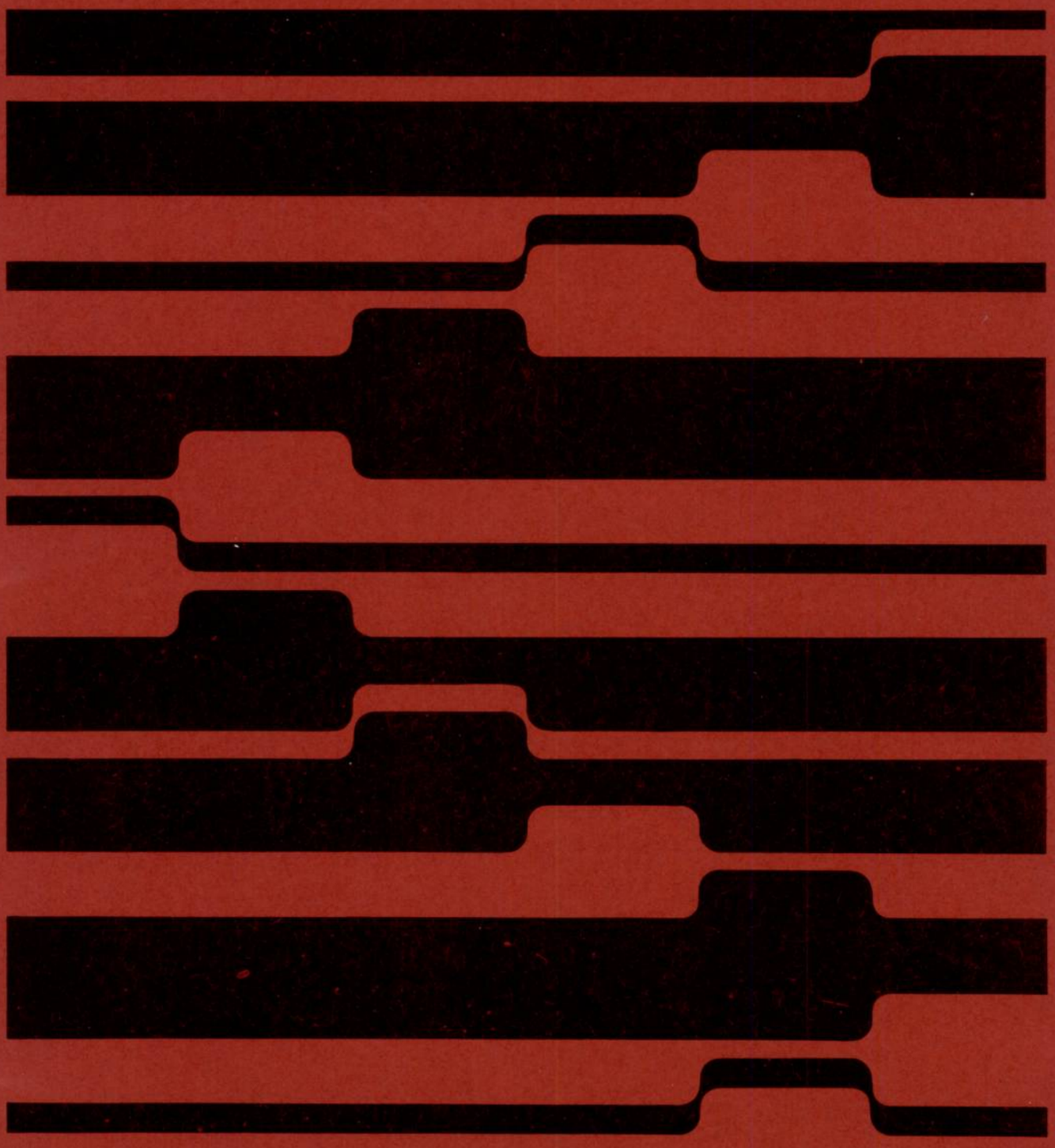
ICL

Data Management Software: Reporting

1900 Series



ICL

Data Management Software: Reporting

1900 Series

A

Technical Publication 4132

First Edition February 1970

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

Preface

This manual describes the two Data Management Software programs which are concerned with reporting. This includes the production of a report on the line printer and the creation of a print tape.

This manual is one of a series describing ICL Data Management software and should be read in conjunction with the manual *Data Management Software - Introduction* which explains the philosophy behind ICL's approach to data management.

This manual is intended to be read by those already using or considering the use of the Data Management Software programs described. These may either be programmers or people with no experience of programming; the software is designed to allow the user to control the programs entirely by parameters in a given format. The manual will also be of use to systems analysts and designers in so far as it explains what functions the programs are capable of performing.

The manual is divided into two parts, each part dealing with a single program, as follows:

Part 1 REPORT, a program for the production of an ordered printout or a print tape from input magnetic tape files.

Part 2 REPRINT, a program for the production of an ordered printout from a prepared print tape.

Contents

Illustrations

PART 1 REPORT

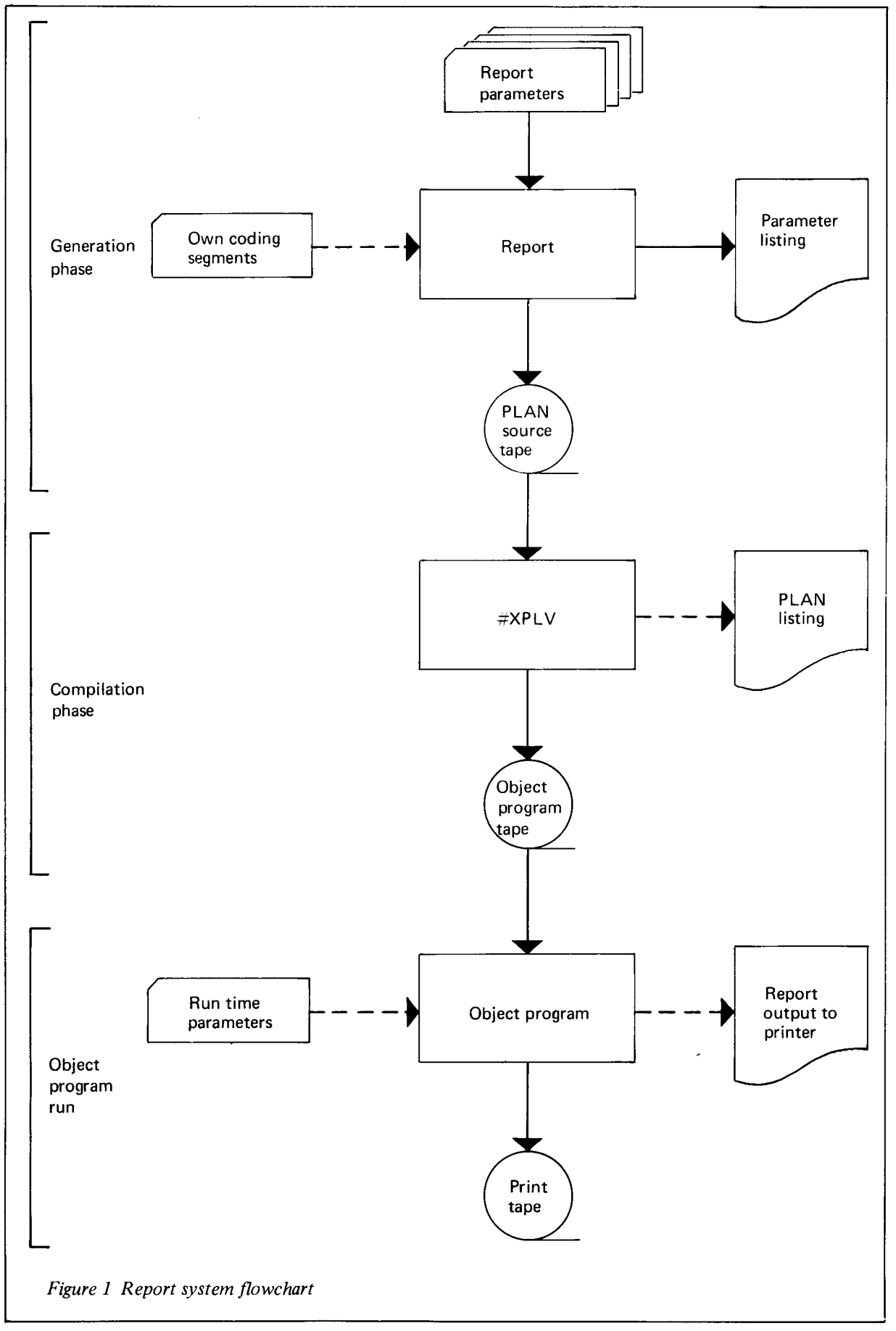


Figure 1 Report system flowchart

Chapter 1 Introduction to REPORT

REPORT generates a program which produces reports, either in the form of a printout or on a print tape, containing information from data stored on magnetic tape files. The production of a report takes two computer runs. On the first run a set of parameters punched either into cards or into paper tape is input and processed to produce a PLAN source program on magnetic tape: this program is then compiled. REPORT calls the compiler #XPLV. The output from this run is a tape which contains an object program generated in accordance with the parameter values input to the run.

The object program tape and the data files to be listed make up the input to the second run, the object program run, which outputs either a listing on the line printer or a print tape which can subsequently be run with REPRINT (see page 43) to produce a listing offline.

The facilities of the program enable the format of the listing produced to be controlled by the user. REPORT has several additional facilities which can be used in the processing of the input data files.

1 Data from up to three data files held on magnetic tape can be merged to produce one report

2 Calculations and analyses can be performed upon the data before it is printed, and the results incorporated into the printout

3 Editing and conversion routines can be used to alter the format of information, insert explanatory headings, convert from binary to decimal, and so on

4 Own coding facilities are available to enable the user to process the data as he requires as part of the REPORT run

MINIMUM CONFIGURATION

The minimum computer configuration required for the generation phase of REPORT is

1 1900 Series processor with 16K words of core store
1 card or paper tape reader
1 line printer
4 magnetic tape decks

The configuration needed for the object program run depends on the parameter set used.

Chapter 2 File input and output

COMPILATION PHASE

The main input to the compilation phase of REPORT is the set of REPORT parameters punched into eit! ɾ cards or paper tape: these are described briefly below but for a detailed description of each separate parameter ›ference should be made to Chapter 3. Also input at this stage are such own coding segments as the user may have ›ritten.

Parameter input

The parameters supplied to REPORT must contain all the information necessary for REPORT to generate ›rrect print program. The main functions of the parameters can be classed as follows:

1 Defining the environment. REPORT is informed of the type of line printer that is to be used

2 Defining the data to be input; this involves naming the files to be used and identifying and locating relevant records and fields within the files

3 Defining any processing to be performed, either upon each record in isolation, or upon groups of related records

4 Defining the format and content of the output of the program

The parameters do not have to define the structure of the program to be generated by REPORT, since this is always the same. Most of the functions found in normal print programs are present in programs generated by REPORT, including facilities:

1 To read run time parameters

2 To check that paper is correctly set up

3 To check for control breaks between records. A control break is a change in the contents of a control field and causes a specified type of processing to begin

4 To perform calculations and totals on fields extracted from records

5 To convert and edit fields on output

6 To specify printing requirements
   (a) at the start of the run
   (b) at the head of each page
   (c) in the body of the text
   (d) at the foot of each page
   (e) at control breaks
   (f) at the end of the run

These points and others are covered in more detail in the descriptions of parameters which follow.

The parameters are specified in *groups*, each group consisting of several related parameters. The groups are arranged to correspond to the sequence in which the user will normally define his print requirements.

PRELIMINARY INPUT GROUP

The preliminary input group defines the four character name to be given to the program generated by the first run of REPORT. The generated program is in COSY mark 2 format, (see *PLAN Reference Manual, Chapter 9*), and occupies a single subfile. The program name is used as the 12 character name of the subfile, the remaining eight characters being space-filled, and also as the name in the first line of the program; the same name will also be used for the object program after compilation.

This group also names the input file or files. The generated program will be so formed that it expects to read from files with the precise labels specified in this group. However, if it is required to use the same program to read files with different labels, for example, from different generations of the same file, the program can be adjusted by means of run time parameters. These are described on page 7.

The generated program can produce output directly on a line printer, or it can write the same output data to a print tape, for subsequent listing by REPRINT. This parameter group may contain details of the label to be written to this print tape. The use of a print tape is optional.

DICTIONARY GROUP

The Dictionary group is described in *Data Management Software, Introduction.* The dictionary parameters define each relevant field in terms of its position within input records, their type (for example, characters or binary), and their size. Only fixed length fields can be addressed, and they must be in positions that are fixed within the record. A further restriction in REPORT is that fields must be defined in sequence, starting from the beginning of the record. However, fields that are not required either for printing or for calculation do not have to be defined; only those fields which are required in the current run need be defined. The same part of a record can be mentioned in more than one dictionary entry, provided that the addresses do not break sequence; this can be useful if a field contains coded information which has to be examined in part in one stage of the program and all together in another stage. For example, a sales analysis might be required showing totals of sales by individual salesmen, with further totals by sales area: the salesman code might consist of three digits, of which the first is also the code of the sales area. REPORT allows the user to define all three digits as a single field, but also to define the first digit on its own as another field.

There is no absolute limit to the number of fields which can be defined in a REPORT dictionary. The program is written so that it will expand to use additional core storage if the original allocation proves too small for the parameter group provided.

ENVIRONMENT GROUP

The main function of the Environment group is to tell the generator program what size of printer the object program will use, and what size of records the object program will have to handle.

The printer may have 96, 120 or 160 print positions, and the appropriate program will be generated, according to the number of print positions specified by the parameters. At the same time the user specifies the depth of the normal printing page, so that the generated program can be equipped with a suitable line count. This means that the generated program can change page in response to particular needs (for example, at a total) or it can automatically change pages during listing, a new page being called for when the current page is full. The generated program will take note of any overflow punching in channel 8 of the printer control loop, and will call for a new page if any overflow is detected; the user can take advantage of this if a program generated for 11-inch paper is to be used with 9-inch paper. However, this is optional, and the program does not assume that overflow punchings will always be present in the printer control loop.

This group of parameters also defines the maximum block size for each input file, and the maximum record size. This allows the generator to set up economical record areas and buffer areas. If, as a file is developed, record sizes increase, the program will have to be re-generated with expanded record areas. It is therefore probably worthwhile to overstate the record sizes somewhat, unless the user can be certain that record size is not going to change during the life of the generated program. Any increase beyond the maximum block size incorporated in the generated program will be detected by the Magnetic Tape Housekeeping package and will be signalled as a long block error. The generated program does not include coding to check that the record fits within the generated record area, and the user must, therefore, check that the correct parameter value is given himself.

The group includes a keys parameter if more than one file is being input to the generated program. In some print programs, keys are used to initiate the printing of totals, a total being printed whenever the value of a key field changes; this is not the case with REPORT. The only function of this parameter in REPORT is to identify the fields which are to be used in relating records from two or three input files. If the generated program will have only a single input file, this parameter is not required. Where more than one file is input to the program, File A is treated as a master file, and records from other files are matched against it. The program will go through the printing and calculating functions specified by the user for each File A record, regardless of whether or not it is matched from other files; unmatched records from other files are ignored.

Also in this group are optional parameters indicating that own coding routines are bring supplied at various points. Own coding is discussed in more detail in Chapter 4.

CONSTANTS GROUP

The Constants group includes all constants that will be required during calculation and printing. These parameters are equivalent to Data Statements under the control of a #LOWER directive in an ordinary PLAN program; they give rise to constants established in the Lower Preset area of the generated program. The user is able to specify that constants should be held in character form, or that they should be converted into binary.

The user will sometimes wish to change the values of particular constants. For example, in an evaluation calculation it might be necessary to alter the rate per unit, or the heading of a printout might have to be varied according to

whether it is the weekly or monthly version. The generated program is able to read parameters at run time to insert revised values in such constant locations: this is described in more detail under the heading of *Run time parameters,* on page 7. However, the revision of constants at run time can be simplied if all constants that are subject to revision are defined together at the start of the Constants group.

The user should note that the generated program will contain certain special fields which do not have to be specified in the Constants group. For example, one such special field contains the date of run obtained from Executive; the user does not have to set up the date of run by inserting a run time parameter in the Constants area. These special fields are described on page 14.

RECORD SELECTION GROUP

The Record Selection group allows the user to specify which records are to be considered for calculation and printing, and which are to be ignored. The user may specify either which particular records are to be selected, or which particular records are to be ignored. If he is using selection, indicated by #ONLY parameters, then the only records to be processed are those which meet the selection criteria; if he is using #SKIP parameters, then all records will be processed *except* those which meet skipping criteria. The user cannot mix SKIP and ONLY conditions; if the record selection is so complex that this would be necessary, it is advisable to do a preliminary run with GAMP in order to produce a suitable subfile for use by REPORT.

When #SKIP parameters are being used, a record is skipped if it meets *any* of the skipping criteria. This is equivalent to *skip this record if, for example, a* OR *b* OR *c.*

When #ONLY parameters are being used; a record is processed only if it meets *all* the selection criteria. This is equivalent to *process this record, if for example, a* AND *b* AND *c.*

It will often be the case that the same print program will be required to print different subsets of records on different occasions. A good example of the use of run time constant parameters is if a program has been generated to skip records when a certain field in the record has a value within the range specified by the two constant fields. The user can then control which records are skipped on any particular run by setting appropriate range values in the two constant fields.

TABLES GROUP

The Tables group allows the user to translate code values into equivalent texts. This can often be useful if the printed report is to be read by members of higher management who are not familiar with the meanings of particular code values. For example, a company installing deep-freeze units might have a sales term code in the records, using 1 for 'sold', 2 for 'rented', 3 for 'leased', 4 for 'free loan' , and so on. The Tables parameters will allow the user to print SOLD, RENTED, and so on, without his having to keep these texts in the records themselves in the input files.

The parameters contain the values that are to be translated and also the texts which are to be inserted in their place in the printout. The generated program holds the code values in lower storage, and the equivalent texts in upper storage; this means that the user need not limit himself in defining texts by consideration of the space which they will occupy.

CALCULATIONS GROUP

The Calculations group contains parameters which define required calculations and comparison actions. These are sufficiently different to require separate descriptions here.

Calculations parameters

Calculations parameters are used to set up subroutines in the generated program which perform the necessary arithmetic functions on fields from the input records, on values taken from constant fields, and so on. Each calculation makes use of three labels: two of these identify the operands in the calculation, and the third identifies the location which is to receive the result of the calculation. The third label may refer only to the Results area; it must not be the label of a field in the record or in the constants area. The generator automatically creates a Results area to receive the results of calculations, and within this area fields are identified by the labels given to them in calculations parameters; at the start of the run all fields in the results area contain zero.

The arithmetic operations of addition, subtraction, multiplication and division can all be carried out. The program can also generate a calculation to express one numerical value as a percentage of another.

Calculations may be used in various ways according to the requirements of the user. For example, in a stock valuation run, a price per unit could be multiplied by the number of units to give the value of each line by one calculation; another calculation could total the values for all lines.

Comparison parameters

Comparison parameters can be used to alter fields at the user's discretion in order to control subsequent calculations

and printing. In general terms, a comparison can be expressed as: "If field A is equal to field B, then move the contents of field C to field D; if they are unequal, then either do nothing, or spacefill field D, or zeroize field D". The four fields quoted in the parameter do not necessarily have to be different fields; for example a constant may be moved to a field in the record if that field equals another constant. In fact, this kind of transposition of values is the most common use of the comparison facility.

Both calculations and comparisons must be numbered. The numbers given in the parameters are used in the generated program to identify the generated subroutines. Calculations can be numbered from 01 to 99; numbers may be omitted but the parameters must be submitted in ascending sequence of these numbers. However, this sequence does not necessarily determine the sequence in which the subroutines will be called during operation of the generated program. A further block of numbers from 01 to 99 can be used for comparisons, and the same rules about sequence apply.

HEADINGS GROUP

The Headings Group includes parameters which define the print format for headings to be printed:

1 At the start of the run; these are called *Main headings*

2 At the head of each page during listing; these are called *Page headings*

Main headings can be used to print a preliminary page giving a title and other identifying information to the printout. This point can usefully be made the subject of an installation standard, so that all printouts produced in the installation observe the same format for Main headings; this greatly simplifies the task of referring to a printout that has been filed along with others.

The Page headings defined in this group will be printed automatically every time that the program has to change to a new page, either because the line-count is exhausted or because a punching has been found in the overflow channel of the printer control loop. The form of page headings most commonly used consists of a line giving date of run, title of the printout, and page number, followed by one or two lines giving column headings to identify the following detailed information.

Both Main headings and Page headings are optional.

RECORD LEVEL GROUP

The Record Level group defines the functions to be carried out by the program for each selected record from File A. The group starts with a list of the calculations and comparisons that are to be performed. These should be quoted in the order in which they are to be performed; where one calculation uses an intermediate result prepared by another calculation this sequence can be very important. The calculations are called in the sequence in which they are to be performed; this does not have to be the same as the sequence in which they were originally defined.

The Line description parameters which follow set out the details of the printing requirements for each record. The user can call for any relevant fields to be printed, either from the record itself, or from the constants area, or even from the results area. Coding in the parameters allows the user to specify editing, conversion, and zero suppression facilities as appropriate. These are defined in full detail in Chapter 3, and the user should refer to that chapter and to the examples in Chapter 5 for a full statement of the various possibilities.

Only one set of print requirements can be defined in the Record group.

CARRY GROUP

The Carry group of parameters is used to define printing requirements at the foot of each page. This can be particularly useful when a printout is likely to be separated into single sheets, but where each sheet must contain enough information about, for example, carried forward totals, to allow it to be related to the other information on other pages of the printout.

The parameters which occur in this group are similar to those which occur in the Record Levels group: the user can call for calculations and comparisons, and can provide line description parameters to define his printing requirements.

LEVELS GROUP

Levels group of parameters define the user's requirements in terms of calculation and printing when control breaks occur. The parameters define the control field and, if necessary, the control values of that field.

In the simplest case of the use of Levels groups, a control field is specified, and the routines associated with this level are called whenever the value in the specified field changes. To take again the example of a sales analysis run, it will often be the case that details of individual sales are printed from individual records, but that a salesman's total is to be printed whenever the salesman's number changes: that a sales area total is to be printed whenever the sales area number changes, and so on to regional, national, and any other levels. If the sales area number changes, a total for

the last salesman in the previous area is formed. The least significant level is the first level at which a total is formed (in this example, salesman's totals); the most significant level is the last level at which a total is formed records (in this case the national sales total). In REPORT, levels are numbered from 0 to 9; 0 being used for the most significant level and 9 for the least significant. If, as usually happens, fewer than ten levels are used, the levels should still be numbered from level 0.

In the alternative case there may be a set of records that do not contain a type code, such as salesman's number but the first record of the set can always be recognized by a specified control value in a particular field. Levels routines are then called whenever the value in the control field of the latest record is equal to one of the specified values.

Further parameters in the group, similar to those used in the Record Levels group, specify which tabulations and comparisons are to be called. If the printing for a level is to be on a separate page, not on the same page as the preceding records, page parameters can be included; these are similar to those parameters in the Headings group which specify the headings that will be printed at the top of the new page. The Levels group may also contain line description parameters giving the print format details of the totals fields that are to be printed.

Dummy levels

In the present parameter structure, there can be only one control field for each level of totalling. In simple cases this is obviously sufficient, but the condition can arise where a total is to occur if there is a change in any or all of two or more fields and the totalling procedure is required to be exactly the same, whichever of the fields causes the change.

This can be achieved by specifying one group of level parameters which refer to one of the control fields, which specify all calculations and all printing requirements; and by specifying dummy level sets for the other control fields, simply specifying the control field without calling for any calculation or printing. However, the level sets should be arranged in the right order because whenever a control break occurs at one level, the program automatically obeys the specified processing for all lower levels. If the level set which contains the calculation and printing details is made lower than the dummy level sets, the correct answer will be produced.

For example, in a sales analysis printout, a total might be taken of the sales of each area and within each area of the sales of each product. A change of area can be recognized by a simple change in the area code number, but a change of product may be indicated by a change in product code, product description, or product size, or more than one of these. Parameters to produce the required printout should be arranged as follows:

Level Group 0 (the most significant level)
Control field: Area code
This group contains calculations and printing to produce the total sales for the area.

Level Group 1
Control field: Product code
This group is a dummy, and contains no calculation or printing.

Level Group 2
Control field: Product description
This group is a dummy, and contains no calculation or printing.

Level Group 3 (the lowest level used in this example)
Control field: Product size
This group contains the calculations and printing to form a total for the sales of the product.

TOTALS GROUP

The Totals group of parameters is in many ways similar to a Levels group, except that the Levels groups define totalling activities that are to take place at intervals during the run, whereas the Totals group defines the final totalling activities that are to take place at the end of run only. The rules for what this group may contain are similar to the rules for the Level group.

Run time parameters

If at any time a generated program is to be run using different input files, it will be necessary to use run time parameters to give the details of the actual labels on the tapes to be read. These parameters are identical in format and content to the main #READ parameters with which the program was generated.

Similarly, a run time #WRITE parameter can be used to alter the label that is to be written to an output print tape from the program. Again, this parameter is in standard form.

If constant values are to be changed at run time, new values can be inserted preceded by a #CONSTANTS group header parameter. At run time the program no longer has any record of the names of individual constant fields;

```
        15/32/01        24/10/69        COMPILED BY XPLY 26D


0001    #PROGRAM        PSR275/GENERATED
0002    #CMODE          MT(0,1,2,3)NULL(4-15)
0003    #               GENERATED BY REPORT MARK 4
0004    #LOWER          COMMON/LEVELNUMBER/
0005                    LEVELNO
0006    #LOWER          COMMON/USER/
0007    ITEMA           0                                                    1288 #00000000
0008    ECOUNT          0                                                    1289 #00000000
0009    PCOUNT          0                                                    1290 #00000000
0010    EPAGE           0                                                    1291 #00000000
0011    DATE            0,0                                                  1292 #00000000

0012    #LOWER
0013    SWB             0                                                     728 #00000000
0014    SWC             0                                                     729 #00000000
0015    VAL             0,0,0,0                                               730 #00000000
                                                                              731 #00000000
                                                                              732 #00000000
                                                                              733 #00000000
0016    BAL             0,0,0                                                 734 #00000000
                                                                              735 #00000000
                                                                              736 #00000000
0017    #LOWER          COMMON/PRINTAREAS/
0018                    CHAR,PRINT(40)
0019    #LOWER          COMMON/CALCULATION/
0020    GPTOTAL2X       0(2)                                                 1294 #00000000
                                                                             1295 #00000000
0021    GPAVRGE2X       0                                                    1296 #00000000
0022    GPCOUNT1X       0                                                    1297 #00000000
0023    GDTOTAL2X       0(2)                                                 1298 #00000000
                                                                             1299 #00000000
0024    #LOWER
0025    FDAA            261/0,0,12HPMMXPERSONNF,0000,0002                     737 #40500000
                                                                              738 #00000000

                                                                              796 #00000000
0029    #LOWER          COMMON/RECORDAREAS/
0030                    RECDA(073),BASICPY2A(1),A001(182)
0031    #LOWER          COMMON/CONSTANTS/
0032    HEAD1XXHY       52HLIST BY AGE OF PERSONNEL RECORDS ON PERSONNEL FILE 1300 #54516364
                                                                             1301 #20427120
                                                                             1302 #41474520
                                                                             1303 #57462060
                                                                             1304 #45626357
                                                                             1305 #56564554
                                                                             1306 #20624543
                                                                             1307 #57624463
                                                                             1308 #20575620
                                                                             1309 #60456263
                                                                             1310 #57565645
                                                                             1311 #54204651
                                                                             1312 #54452020
0033                    08H                                                  1313 #20202020
                                                                             1314 #20202020
0034    RANGEAXHY       02HL1                                                1315 #54012020
0035    RANGEBXHY       02HL9                                                1316 #54112020
0036    RANGECXHY       02HP1                                                1317 #60012020
0037    RANGEDXHY       02HP9                                                1318 #60112020
0038    RANGEEXHY       02HT1                                                1319 #64012020
0039    RANGEFXHY       02HT9                                                1320 #64112020
0040    HEAD2XXHY       52HDATE OF BIRTH  PERSONNEL NO. TITLE AND NAME       1321 #44416445

0050    #LOWER
0051    LEAVCODHAS      0003/0                                                797 #00300000
0052                    /RANGEAXHY,/RANGEBXHY                                 798 #00002443 1315
                                                                              799 #00002444 1316
0053                    /RANGECXHY,/RANGEDXHY                                 800 #00002445 1317
                                                                              801 #00002446 1318
0054                    /RANGEEXHY,/RANGEFXHY                                 802 #00002447 1319
                                                                              803 #00002450 1320
0055    #UPPER
0056    TEXTS           52HSYSTEMS DEPTSERVICE DEPTINFORMATION ACCOUNTING SALA 4997 #63716364
                                                                             4998 #45556320
                                                                             4999 #44456064

                                                                             5009 #63415441
0057                    20HRIES     SALES DEPT.                              5010 #62514563
                                                                             5011 #20202020

0106    #PROGRAM

0109    #ENTRY          3
0110            STOZ    LMARK                                        1502 033 0 0    45
                                                                     1503 124 7 0   256
                                                                     1504 112 7 0  1036
                                                                     1505 020 7 0    30
0111            TEST  7 12
0112            BZE   7 *+3                                          1506 050 7    1509
0113            STOZ    GOTLP                                        1507 033 0 0    46
0114            BRN     JWSCS                                        1508 074 0    1537
0115            ALLOT   LPO                                          1509 156 0 0     2
0116            LDX   7 9                                            1510 000 7 0     9
0117            STO   7 GOTLP                                        1511 010 7 0    46
0118            BPZ   7 *+3                                          1512 054 7    1515
0119            SUSWT   2HLP                                         1513 161 0 0  2864
0120            BRN     *-5                                          1514 074 0    1509
```

Figure 2 Structure of the generated program

these are used during generation, and during compiling, but by run time they have been lost. The program is therefore unable to refer by name to any special location within the constants area. For this reason, when run time constants are stored, they are stored consecutively from the start of the constants area, taking the place of the values that were included at generation time. When changes are made to any of the constants at run time, all the constants up to and including the last constant that has to be changed must be input again. This may mean that some constants whose values have not been changed will have to be input again if they are stored before a constant that is to be changed. Therefore, the user is advised to group together at the start of the constants area all constants that are likely to have to be changed, as this arrangement will minimize the number of constants that have to be input at run time.

Only the values of the constants may be changed at run time; the user should not change either the type of a constant or its length. This is because the processing of the constants during the operation of the program will be as defined in the original parameter group. If, there, the user overwrites by, for example, a new page heading line, a pair of binary words used for selection purposes, the first two sets of four characters for the newly submitted heading would be used for selection purposes. If a constant has originally been compiled as binary, any run time parameter must also be binary of the same type; if a constant has originally been compiled as 17 characters, any run time alterations must also consist of 17 characters.

Tape requirements

The compilation phase also requires three scratch magnetic tapes which are used:

1 To hold a copy of the parameter set in card image format

2 To hold the PLAN coding generated in the first stage of the compilation phase

3 To hold the output of the #XPLV compiler. This is the compiled object program complete with own coding segments for input to the object program run

OUTPUT

The output from the first run(generation and compilation phases) consists of:

1 A listing of the REPORT parameters on a line printer

2 Optionally, a PLAN listing of the generated object program on a line printer

3 The object program tape (Tape 3 above) for input to the object program run of the REPORT program

Structure of the generated program

The program generated by REPORT is not provided with any narrative; nevertheless, with the help of this section of the manual the user should have no difficulty in finding his way about the program and determining which section of the program performs which functions. A printout of the generated program is shown in Figure 2.

The program starts with a #CMODE line which allocates used file numbers to magnetic tape files, and defines all other numbers as null. The input files A, B and C are always known as the program's files 0, 1 and 2. The output file is always file 3.

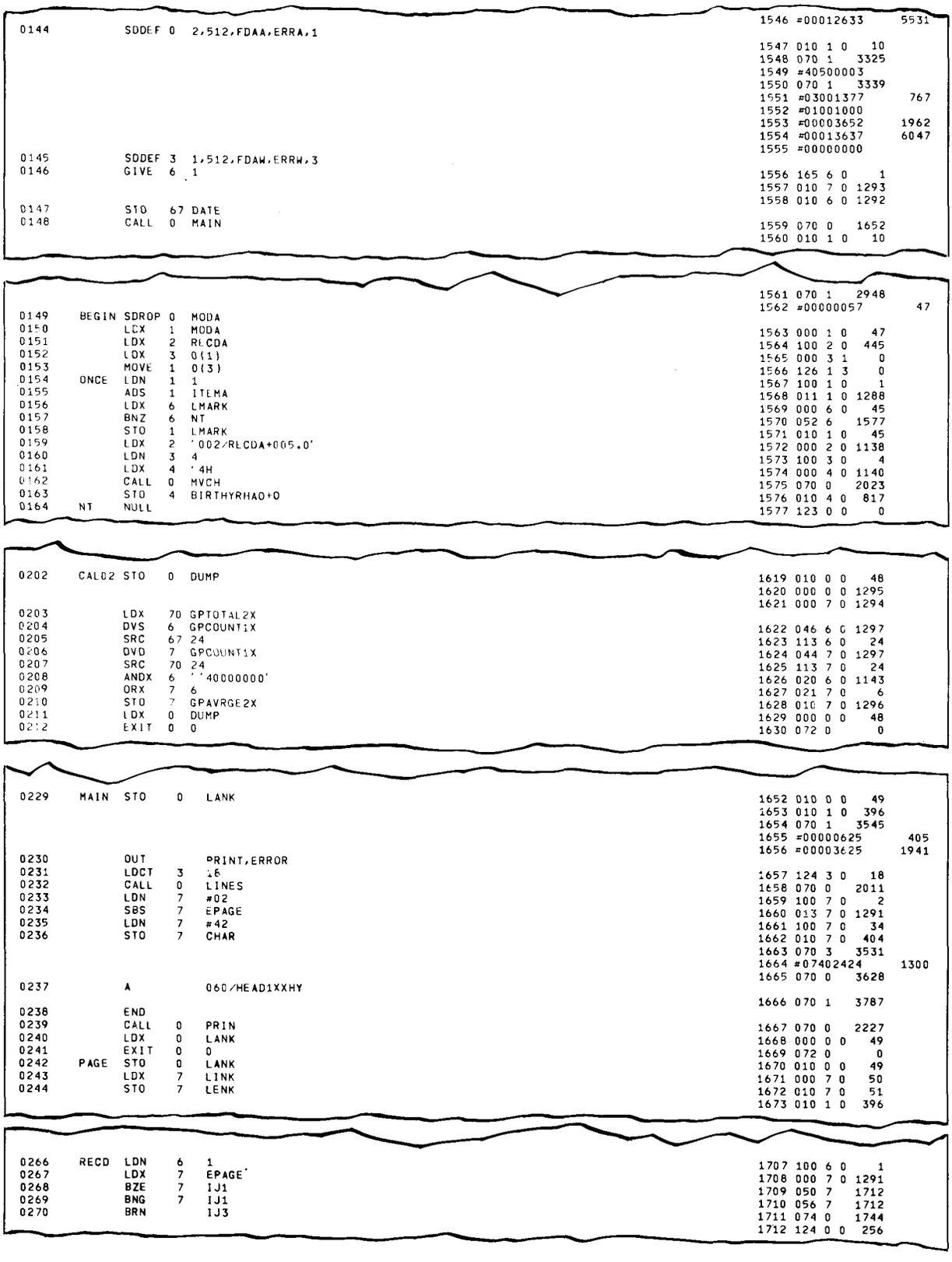
The areas in lower common data, known as /LEVELNUMBER/ and /USER/, are provided for the main program to communicate various values to any own coding segments incorporated by the user, see Chapter Four, page 33.

Lower common /PRINTAREAS/ contains a word for the print feed control character, and an area in which formed printlines are held for printing.

Lower common /CALCULATION/ contains locations for all fields which are named as the result fields in calculation parameters. This area also includes location for the results fields of comparison parameters, unless these parameters call for a result field in either a file area or the constants area. All fields in this area are initially either zeroized or space-filled, as appropriate.

Next is an area of lower preset containing file definition areas for all files. A file definition area is always included for an output file. The user can always define an offline print tape at run time if required. If the user is not taking advantage of the print tape facility, but is outputting direct to the printer, this file definition area can be used whenever it is necessary to dump the program to tape. If a print tape is being used, the file definition area will contain the appropriate filename, reel sequence number, file generation number, and retention period. This area also holds constants used by the program in relating records from more than one input file.

Lower common /RECORDAREAS/, as the name suggests, consists of locations to hold records read from the input files. Most binary fields are represented in this area by modified versions of their dictionary names. Character fields are not so defined, because of the possibility that parts of a character field may be defined in different ways in



```
                                                                    1546 =00012633      5531
0144          SDDEF 0  2,512,FDAA,ERRA,1
                                                                    1547 010 1 0   10
                                                                    1548 070 1   3325
                                                                    1549 =40500003
                                                                    1550 070 1   3339
                                                                    1551 =03001377       767
                                                                    1552 =01001000
                                                                    1553 =00003652      1962
                                                                    1554 =00013637      6047
                                                                    1555 =00000000
0145          SDDEF 3  1,512,FDAW,ERRW,3
0146          GIVE  6  1
                                                                    1556 165 6 0    1
                                                                    1557 010 7 0 1293
                                                                    1558 010 6 0 1292
0147          STO   67 DATE
0148          CALL  0  MAIN
                                                                    1559 070 0   1652
                                                                    1560 010 1 0   10

                                                                    1561 070 1   2948
                                                                    1562 =00000057       47
0149     BEGIN SDROP 0  MODA
0150          LDX   1  MODA                                         1563 000 1 0   47
0151          LDX   2  RLCDA                                        1564 100 2 0  445
0152          LDX   3  0(1)                                         1565 000 3 1    0
0153          MOVE  1  0(3)                                         1566 126 1 3    0
0154     ONCE LDN   1  1                                            1567 100 1 0    1
0155          ADS   1  ITEMA                                        1568 011 1 0 1288
0156          LDX   6  LMARK                                        1569 000 6 0   45
0157          BNZ   6  NT                                           1570 052 6   1577
0158          STO   1  LMARK                                        1571 010 1 0   45
0159          LDX   2  '002/RLCDA+005.0'                            1572 000 2 0 1138
0160          LDN   3  4                                            1573 100 3 0    4
0161          LDX   4  '4H                                          1574 000 4 0 1140
0162          CALL  0  MVCH                                         1575 070 0   2023
0163          STO   4  BIRTHYRHAO+0                                 1576 010 4 0  817
0164     NT   NULL                                                  1577 123 0 0    0

0202     CAL02 STO  0  DUMP                                         1619 010 0 0   48
                                                                    1620 000 0 0 1295
                                                                    1621 000 7 0 1294
0203          LDX   70 GPTOTAL2X
0204          DVS   6  GPCOUNT1X                                    1622 046 6 0 1297
0205          SRC   67 24                                           1623 113 6 0   24
0206          DVD   7  GPCOUNT1X                                    1624 044 7 0 1297
0207          SRC   70 24                                           1625 113 7 0   24
0208          ANDX  6  ''40000000'                                  1626 020 6 0 1143
0209          ORX   7  6                                            1627 021 7 0    6
0210          STO   7  GPAVRGE2X                                    1628 010 7 0 1296
0211          LDX   0  DUMP                                         1629 000 0 0   48
0212          EXIT  0  0                                            1630 072 0      0

0229     MAIN  STO   0  LANK                                        1652 010 0 0   49
                                                                    1653 010 1 0  396
                                                                    1654 070 1   3545
                                                                    1655 =00000625       405
                                                                    1656 =00003625      1941
0230          OUT       PRINT,ERROR
0231          LDCT  3   16                                          1657 124 3 0   18
0232          CALL  0   LINES                                       1658 070 0   2011
0233          LDN   7   #02                                         1659 100 7 0    2
0234          SBS   7   EPAGE                                       1660 013 7 0 1291
0235          LDN   7   #42                                         1661 100 7 0   34
0236          STO   7   CHAR                                        1662 010 7 0  404
                                                                    1663 070 3   3531
                                                                    1664 =07402424      1300
                                                                    1665 070 0   3628
0237          A         060/HEAD1XXHY
                                                                    1666 070 1   3787
0238          END
0239          CALL  0   PRIN                                        1667 070 0   2227
0240          LDX   0   LANK                                        1668 000 0 0   49
0241          EXIT  0   0                                           1669 072 0      0
0242     PAGE STO   0   LANK                                        1670 010 0 0   49
0243          LDX   7   LINK                                        1671 000 7 0   50
0244          STO   7   LENK                                        1672 010 7 0   51
                                                                    1673 010 1 0  396

0266     RECD LDN   6   1                                           1707 100 6 0    1
0267          LDX   7   EPAGE'                                      1708 000 7 0 1291
0268          BZE   7   IJ1                                         1709 050 7   1712
0269          BNG   7   IJ1                                         1710 056 7   1712
0270          BRN       IJ3                                       1711 074 0   1744
                                                                    1712 124 0 0  256
```

Figure 2 Structure of the generated program (continued)

different dictionary entries. This area also contains constants used in extracting key fields from the input records.

Lower common /CONSTANTS/, is the constants area, in which locations and values are made available for all constants in the order in which the constants are specified in parameters. Following these are small areas of lower store for use in storing the control values from the control fields used in detecting when totalling levels are to be called. If any tables are specified their texts are held in upper store and their control details in lower. The remainder of the lower area specified is occupied by miscellaneous control factors, peripheral control areas, and so on.

The program area starts immediately at entry point 3, the normal entry used at the start of run, if run time parameters do not have to be read. The first section of coding is concerned with printing check lines at the head of the form to confirm that the paper has been lined up properly before the run begins. The section of coding forms a loop which may be repeated as many times as necessary, until the user is satisfied that the paper line-up is correct. On continuing, the program opens all files, sets up the date of run, and calls the subroutine to print the main run headings. (If run headings are not being used, this call is omitted).

The program then reads records from the input file or files. If there is more than one input file, a section of coding is included at this stage which matches up records from subsidiary input files with the records from File A. If own coding at INPUT is being used, the program calls this user segment as soon as the matching has been completed. Where there is no File B record to match the File A record, the program empties the File B record area, inserting zeros in all binary fields and spaces in all character fields.

There then follows a section of coding which is concerned with detecting whether a control break has occurred. The least important control field, for the least significant level in use, is tested first, followed by a test of the field for the next level, and so on. When the end of this section is reached, accumulator 1 contains the number of the most significant level at which there has been a control change; if there has been no control break, the program branches straight to the coding concerned with processing the current record. If there has been a control break, the program branches to the next higher level, until all required levels have been processed.

Following the control break testing routines are the calculation and comparison subroutines. These are identified by labels of the form CAL*nn* or COM*nn*. Each subroutine is completely independent.

Next are the routines that contain coding to form the necessary print lines. The first of these routines forms Main headings, the next forms Page headings. Then comes the Record routine. This starts by testing the line count to see whether there is sufficient space left on the current page to print from the new record. If not, the page routine is called to throw to a new page and to reset the line count. Also at the start of the record routine is a test to see whether the user wishes to dump the program before continuing. After these preliminaries the routine calls any required calculations and comparisons, in the order in which they are to be performed. Then print lines are formed and printed as necessary, before the routine branches back to the point where the next records will be read.

The sections concerned with processing control levels are similar. They call any required calculations and comparisons, and then perform the necessary printing distributions. But there is an additional test at the end of each such section: if the most significant level to be processed at the current control break is more significant than the level which has just been processed, the routine branches to the next more significant level; this process will continue until the most significant level required has been processed. The routine then branches to process the record. Users who have written print programs before will be familiar with this structure.

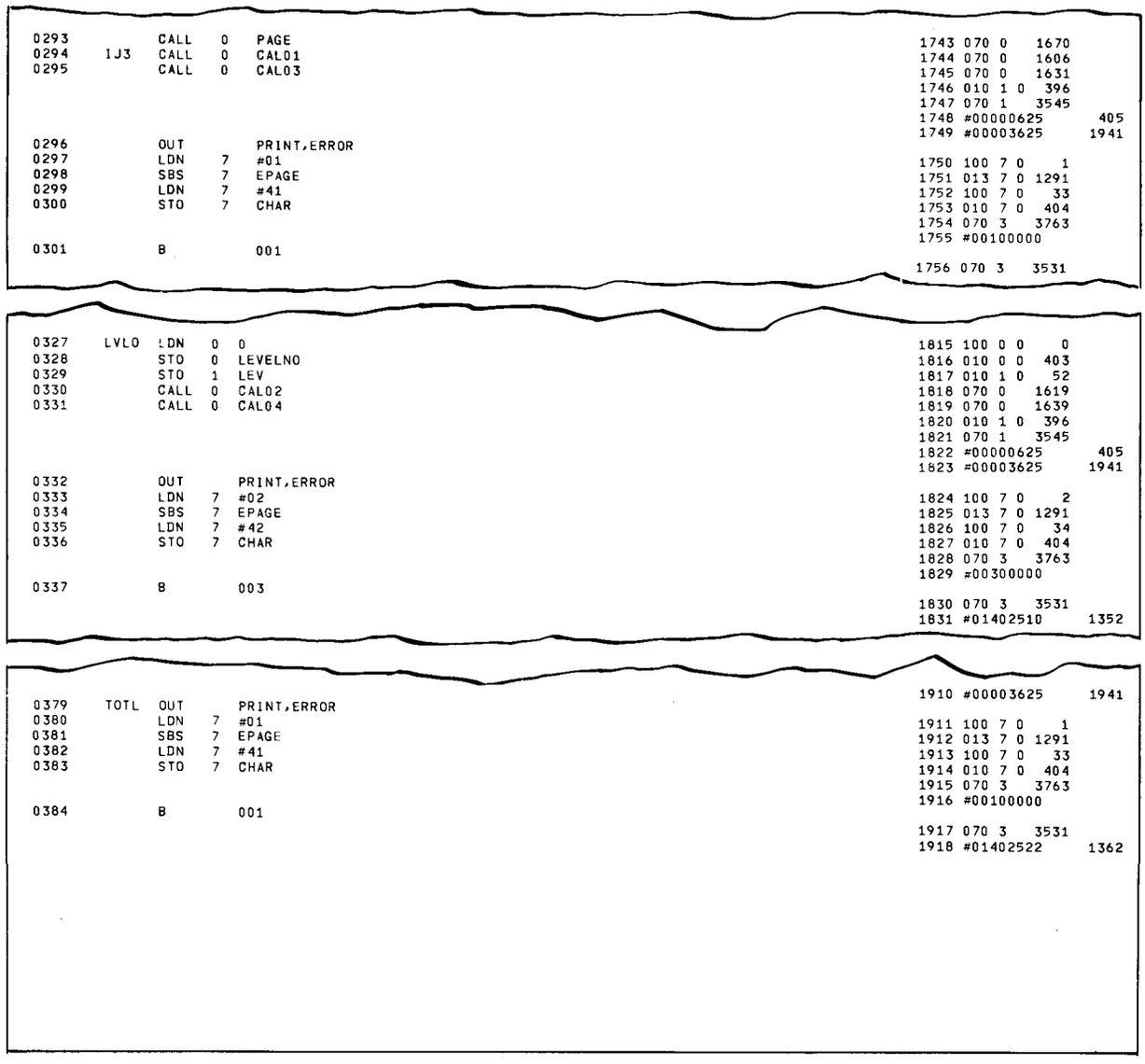
Level processing in this example consists of three functions; accumulation from the current level to the next higher level, printing of the current level, followed by zeroizing of the accumulations at the current level. The first of these functions is performed in REPORT by calculation subroutines, the second by the generated level routine, and the third is also contained in the generated printing routine if it is specified by the user. If the user includes a clear code in the line description parameters he can clear after printing. Otherwise he can accumulate without clearing, producing running accumulative totals throughout the run.

All these print routines use Input/Output Generator statements to perform output conversion and distribution, but they do not use the I/O generator to perform the actual functions of printing. This work is done by the subroutine PRIN, which is generated in different forms according to whether or not the user uses a print tape.

The final print routine is that concerned with the end of run processing, which can be used to print final totals. It is similar in structure and content to the others.

The program ends with miscellaneous routines concerned with tape exception condition processing; reading, checking, and storing run time parameters; and the print and spacefill subroutines.

Where the user has provided own coding routines, these are appended to the program as independent segments. Their structure will naturally be determined by the user's own requirements.



```
0293          CALL   0   PAGE                    1743 070 0    1670
0294    IJ3   CALL   0   CAL01                   1744 070 0    1606
0295          CALL   0   CAL03                   1745 070 0    1631
                                                 1746 010 1 0   396
                                                 1747 070 1    3545
                                                 1748 #00000625       405
                                                 1749 #00003625      1941
0296          OUT        PRINT,ERROR
0297          LDN    7   #01                    1750 100 7 0     1
0298          SBS    7   EPAGE                  1751 013 7 0  1291
0299          LDN    7   #41                    1752 100 7 0    33
0300          STO    7   CHAR                    1753 010 7 0   404
                                                 1754 070 3    3763
                                                 1755 #00100000
0301          B          001
                                                 1756 070 3    3531

0327    LVL0  LDN    0   0                       1815 100 0 0     0
0328          STO    0   LEVELNO                 1816 010 0 0   403
0329          STO    1   LEV                     1817 010 1 0    52
0330          CALL   0   CAL02                   1818 070 0    1619
0331          CALL   0   CAL04                   1819 070 0    1639
                                                 1820 010 1 0   396
                                                 1821 070 1    3545
                                                 1822 #00000625       405
                                                 1823 #00003625      1941
0332          OUT        PRINT,ERROR
0333          LDN    7   #02                    1824 100 7 0     2
0334          SBS    7   EPAGE                  1825 013 7 0  1291
0335          LDN    7   #42                    1826 100 7 0    34
0336          STO    7   CHAR                    1827 010 7 0   404
                                                 1828 070 3    3763
                                                 1829 #00300000
0337          B          003
                                                 1830 070 3    3531
                                                 1831 #01402510      1352

                                                 1910 #00003625      1941
0379    TOTL  OUT        PRINT,ERROR
0380          LDN    7   #01                    1911 100 7 0     1
0381          SBS    7   EPAGE                  1912 013 7 0  1291
0382          LDN    7   #41                    1913 100 7 0    33
0383          STO    7   CHAR                    1914 010 7 0   404
                                                 1915 070 3    3763
                                                 1916 #00100000
0384          B          001
                                                 1917 070 3    3531
                                                 1918 #01402522      1362
```

Figure 2 Structure of the generated program (continued)

OBJECT PROGRAM PHASE

Input

The input to the object program phase of REPORT consists of

1 At least one and up to three input files

2 The object program tape output by the compilation phase

MULTIPLE FILE INPUT

REPORT will accept up to three input files: One file, File A, must always be specified. Two subsidiary files, File B and File C, may also, optionally be input.

All input magnetic tape files must have been created by the standard 1900 Series Magnetic Tape Housekeeping routines using logical processing.

When there is more than one input file, each file must be in the same sequence of keys. The location of the key fields and their order of importance is specified by means of the KEYS parameter. A KEYS parameter must be supplied for each file if more than one is present: it need not be supplied if only one file is present. The actual location of the key fields may differ in each file, but they must be equivalent in type, size and relative importance.

When there is more than one input file, each record from File A is read in turn. Records are then read from File B and File C until a match is obtained. If there is no File B or File C record with identical keys to the particular File A record, then a dummy File B or File C record is created in which all character fields are made blank and all binary fields are zeroized. Further records from File A are read, but until there is a change in the File A keys, no further records are read from File B or File C. When the File A keys change, further records from File B and File C are read until a match occurs or a dummy record is created.

An example of the use of two input files might be an invoicing run. File A would contain records for each item, identified by customer number. File B could contain names and addresses of each customer, again identified by customer number. A new address would be read from File B each time the customer number changes on File A.

End of run occurs when File A ends, whatever the state of the other two files.

All files are identified by standard header labels. File labels are given by means of #READ and #WRITE parameters at the compilation stage, but the object program may also read file label parameters at run time (page 7) which overwrite the existing labels generated at compilation.

Output

Output from the object program run consists of a listing of the report produced either on a line printer, or on a print tape.

PRINTOUT DETAILS

The generated program contains a routine to print check-lines at the head of the first page. These lines consist of a series of 'X's, with asterisks printed in the two middle point positions. If printing is being done on simple listing paper, these lines serve to show that the paper is correctly set of head of form before the main run begins. If the printing is to be done on pre-printed stationery, the line-up is even more important. The check lines are intended to be used in conjunction with a hairline cross printed at the top of each form, in the middle of the form. If the paper is correctly lined up, the four asterisks should register precisely in the four quadrants of the printed cross, as shown:

```
XXXXXXXXXXXXXXXXXXXXXXXXXX*|*XXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXX*|*XXXXXXXXXXXXXXXXXXXXXXXXX
```

This arrangement makes it quite clear if the paper is too high, too low, or displaced to one side or the other. The paper line-up procedure may be repeated as many times as necessary until the operator is satisfied with the line-up obtained.

STRUCTURE OF A PRINT TAPE

The print tape created by REPORT has the structure of a normal data file created by the Magnetic Tape Housekeeping package; that is, there is a standard header label, followed by a standard start of data sentinel; then there are data blocks; and the tape is closed by a standard trailer label.

The data blocks consist of batched records, and each record consists of a single print line, with associated print and feed control character. When the tape is created by REPORT, each line of print is formed in the normal way, but instead of being output to the line printer with a PERI instruction it is written to tape by means of a SDWR instruction. When the print tape is read by REPRINT, each tape record is read in turn with SDRD, and then output to the line printer by means of a PERI instruction. Because each line has been written away with the associated

print feed control character this results in the same pattern of a printout being constituted as if the program had been used directly to produce the printout.

More precise details of the format of records on the print tape are contained in Part 2 (page 43).

Among the important advantages of creating output on a print tape rather than directly on a line printer are:

1 Improved multi-programming possibilities. The program generated by REPORT can be run even though another program is using the printer. The printout can be created on the line printer by REPRINT at any convenient time, and REPRINT uses the minimum number of peripherals

2 Improved repeat and restart facilities. If there is any difficulty with paper line-up or paper tearing, or if the printing run has to be suspended to allow a more urgent job to use the printer, REPRINT can restart at the most convenient point. REPRINT can also be used to obtain multiple copies of a printout without the need to repeat all the processing which created the first copy, and without the need to tie up main files during the printing process

SPECIAL FIELDS

The program generated by REPORT creates and maintains certain fields which can be used for printing as required by the user. These fields do not have to be mentioned in the Dictionary parameters, as they do not occur in the area of the user's records. Some of them are particularly important when using own coding; details of this are available in Chapter 4.

Such fields are:

1 *PCOUNT, which contains the current page count, that is the number of the page which is currently being printed. This is updated by the program every time that the page change subroutine is called

2 *DATE contains the date of run, extracted by the program from Executive at run time, and stored in character form as DD/MM/YY

3 *ITEMA contains a count of the records read from File A; *ITEMB and *ITEMC fulfil a similar function for the other input files, if they are present

4 *EPAGE contains the line-count within the page currently being printed; this is reset at the start of each page to the number specified by the user in the SIZES parameter as being the number of lines in the normal printing depth, and is progressively reduced as printing takes place

The fields may be used in printing and to a limited extent in calculation. If the user performs calculations upon the line count and the page count the pagination of the printout may be modified. When the fields are used in parameters, they should be quoted in place of dictionary labels, as an asterisk followed by the appropriate letters. The letters should be followed by spaces; no type code, source code, or storage code needs to be quoted.

In printing, the fields will occupy the following number of print positions:

*PCOUNT	5
*DATE	8
*ITEMA	5
*ITEMB	5
*ITEMC	5
*EPAGE	5

Chapter 3 The parameter set

The parameter set input to REPORT enables the user to generate a report from up to three input files, and produce either a printout or a listing on to a printout tape for subsequent printing.

Parameters may be input to REPORT either on cards or on paper tape. If some are on one medium and some on the other #SWITCH directives can be used when a change is required.

The parameter set input to REPORT is organized into a series of groups of related parameters. The first parameter of each group, with the exception of the Preliminary Input group, is the group directive, which defines the parameters that follow it: the group directive is given in brackets for each group below.

The REPORT groups are as follows.

1 *Preliminary input group:* the parameters in this group name the object program, and call in tape files for input and output

2 *Dictionary group* (#DICTIONARY): the parameters in this group label fields within each input record to which the program is to refer

3 *Environment group* (#ENVIRONMENT): the parameters in this group define the keys of input file records if more than one input file is being used, specify the printer and page size required, and define entry points for any own coding routines that are being used

4 *Constants group* (#CONSTANTS): the parameters in this group define all the constants to be used in the output and calculations: both heading lines and numeric constants are defined here

5 *Record selection group* (#SKIP or #ONLY): the parameters in this group specify which records from the input tape files are to be printed

6 *Tables group* (#TABLE): the parameters in this group describe tables which are to be used to translate code fields on the input files into printed text

7 *Calculations group* (#CALCULATIONS): the parameters in this group describe any calculations which may be required in the preparation of data for the report

8 *Headings group* (#HEADINGS): the parameters in this group establish headings to be used on pages in the printout

9 *Record level group* (#RECORD): the parameters in this group describe the output required for every record input, including any calculations required

10 *Carry group* (#CARRY): the parameters in this group describe any special output which may be required when a page of printout is full

11 *Levels group* (#LEVEL): the parameters in this group describe the calculations and output required at each control break, and define the occurrence of a control break

12 *Totals group* (#TOTALS): the parameters in this group describe the calculations and output required at the end of the run

13 *End of parameters* (#END): this card signals the end of parameters

Certain parameters may be altered by the input (at run time) of new parameters which can be read by the object program. These run time parameters are described on page 32.

The parameter groups are described more fully below. Each parameter is illustrated by an example.

STANDARD CODES FOR REPORT

Certain fields in some parameters are punched according to standard codes. To avoid having to list the full set of alternatives for each of these codes every time it is mentioned, a full list is given here.

Type codes
This single character code tells the program in what form a field referred to in a parameter is to be interpreted from the input file.

Punching	*Interpretation*
H	Characters
%	Binary numbers
£	Binary pence
@	Sterling Character Field, 2 pence positions:
&	Sterling Character Field, 1 pence position: 10d = '&', 11d = '–'
/	Sterling Character field, 1 pence position: 10d = ':', 11d = ';'
A	Binary days
B	Bit position

Source codes
This single character code tells the program where a field referred to in a parameter is to be found.

Punching	*Interpretation*
A, B, C,	File A, B or C
X	Results area
Y	Constants area

Storage codes
This single character code tells the program in what form a field referred to in a parameter is stored.

Punching	*Interpretation*
1	Single length binary
2	Double length binary
3	Single length binary fraction
4	Double length binary fraction
5	Double length mixed binary number (1 word integer, 1 word fractional)
6	Single length binary pence
7	Double length binary pence
∇ (space)	Characters

Note: In REPORT, a field is fully defined by means of its label, type code, source code, and storage code.

PRELIMINARY INPUT PARAMETERS GROUP
This group of parameters consists of

1 The program label parameter (#NAME), which names the object program output by the compilation phase of REPORT

2 Read tape label parameters (#READ) for each file to be input to REPORT. There must be at least one #READ parameter for any run of REPORT

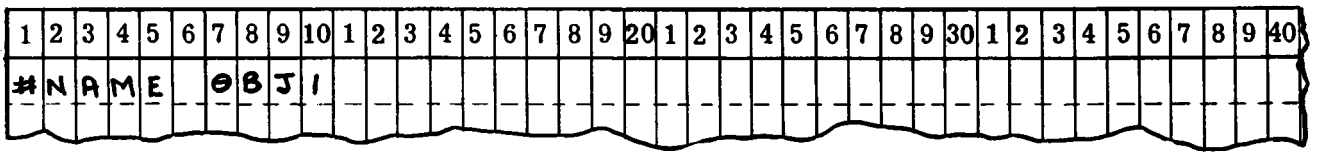
3 A write tape label parameter (#WRITE), naming the file to be used for the output tape. This parameter is optional

These parameters are punched as follows:

Program Label parameter (#NAME)
This parameter names the object program to be output by the compilation phase of REPORT.

Field	*Columns*	*Contents*
A	1 to 5	The directive #NAME
B	7 to 10	Program name to be allocated to the object program. First character alphabetic, others alphanumeric.
C	12 onwards	Steering line to be used with the generated program, see *The Plan Reference Manual, Chapter 7 STEERING LINES.* If this field is left blank FULLIST will be used automatically.

Example

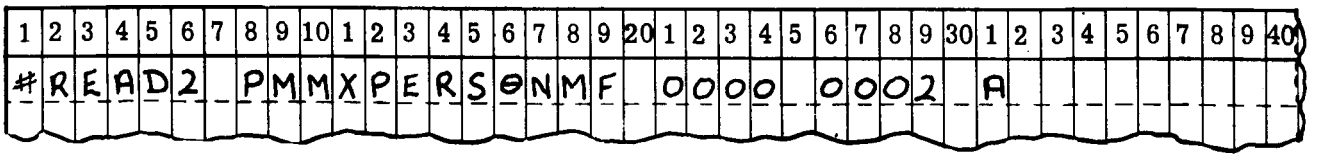


Read tape label parameter (#READ)

This parameter is in standard form. Its function is to define the labels on the input tape files, and the modes in which they are to be opened.

Field	*Columns*	*Contents*
A	1 to 5	The directive #READ
	6	Opening mode: 1 character as follows: 1 = do not check for absence or presence of write permit ring. 2 = check for absence of write permit ring. ∇ (space) in this position is interpreted as 2. Note: These are equivalent to the standard modes of opening tapes that is #100, #200 and #300. Details of additional checks by Executive on the 1904 and above will be found in the ICL manual *Magnetic Tape.*
B	8 to 19	12 character file name
C	21 to 24	Reel sequence number: four digits.
D	26 to 29	File generation number: four digits.
E	31	Designation: one alphabetic character, either A, B, C.

Example



Write tape label parameter (#WRITE)

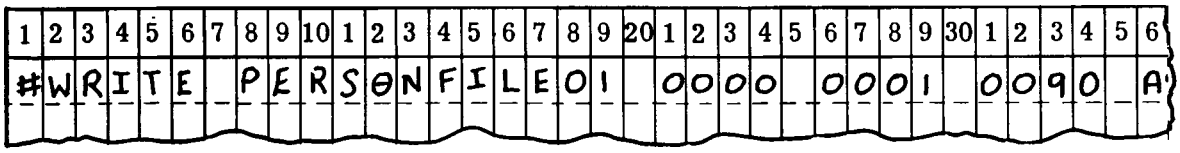
This parameter is in standard form. Its function is to define the labels to be written to output files.

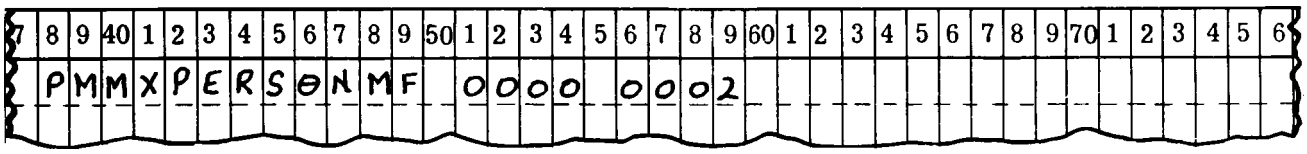
Fields G, H and I are optional. If they are punched the program will search for a tape with the specified label and open it for output: if these fields are blank, the program will open any available scratch tape.

Field	*Columns*	*Contents*
A	1 to 6	The directive #WRITE
B	8 to 19	File name: 12 characters. The name of the file to be output.
C	21 to 24	Reel sequence number: 4 digits. This indicates which reel of the file is to be written first.
D	26 to 29	File generation number: 4 digits.
E	31 to 34	Retention period: 4 digits.
F	36	Designation: one alphabetic character.

Field	*Columns*	*Contents*
G	38 to 49	Old file name: twelve characters. The existing filename on the tape to be used for output. This is the name of the file to be overwritten.
H	51 to 54	Old reel sequence number: 4 digits.
I	56 to 59	Old file generation number: 4 digits.

Example





Note: If this parameter is omitted, the print tape or dump tape (if used) will be given the file name ICLVDUMP*name* where *name* represents the name of the generated program. Reel sequence number and file generation number will both be zero. The tape can also be named, and the reel sequence number and file generation number defined by run time parameters.

DICTIONARY PARAMETERS GROUP

This group of parameters consists of

1 The group directive (#DICTIONARY), which must be present

2 A dictionary identifier parameter for each input file. At least one must be present

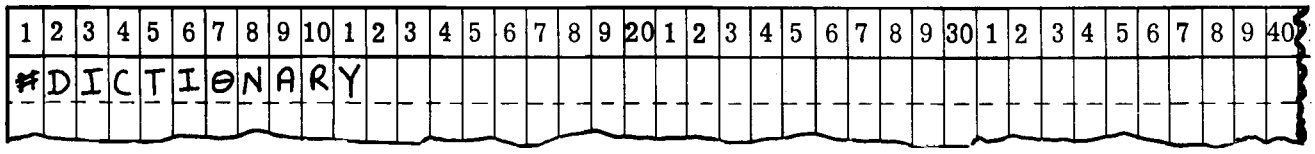
3 Dictionary parameters, which label the fields in the input records to which the REPORT program is to refer

The parameters are punched as follows.

Dictionary group directive parameter (#DICTIONARY)

Field	*Columns*	*Contents*
A	1 to 11	The directive #DICTIONARY

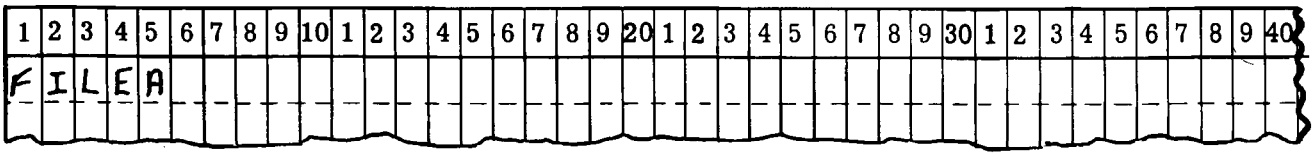
Example



Dictionary identifier parameter (#FILE)

Field	*Columns*	*Contents*
A	1 to 5	#FILE
B	6	File identifier: A, B or C,

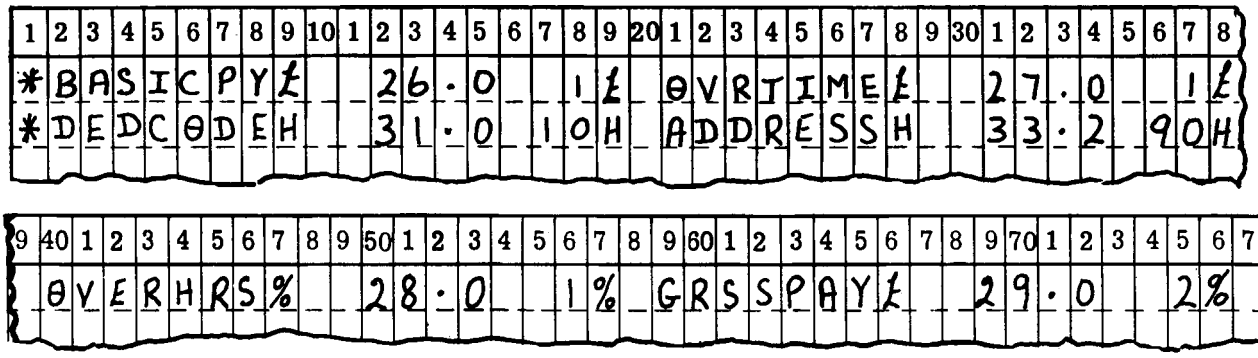
Example



Dictionary parameter

Field	*Columns*	*Contents*
A	1	* (asterisk)
B	2 to 9	Field name Columns 2 to 8 contains a seven character field name, the first character alphabetic, others alphanumeric. Column 9 contains the type code.
C	11 to 13	Field size. If the type code is characters, columns 11 to 12 contain the number of characters, column 13 contains 'H'. If the type code is binary, columns 11 to 12 contains the number of words and column 12 is blank. If the type code is bit position, columns 11 to 12 contain the bit number, column 12 contains 'B'.
D	15 to 19	Start address of the field in words and characters in the form NNN.N. The character position must be zero for either binary or bit fields.
E	21 to 38	As B to D for a second field.
F	40 to 57	As B to D for a third field.
G	59 to 76	As B to D for a fourth field.

Example



Note: It is not necessary to define four fields per card or block but within a card the fields on the card should be defined in adjacent positions since the first blank fieldname is considered to be the end of data on a card or block.

ENVIRONMENT PARAMETERS GROUP

This group of parameters consists of

1 The group directive (#ENVIRONMENT), which must be present

2 Keys description parameters (#KEYS), which must be present if more than one input file is being used, and which identify the positions of the keys in the records on a tape file. If a single input file is used, no #KEYS parameter is needed: otherwise, there must be one for each input file, including the first

3 A record sizes parameter (SIZES), which must be present and which gives the record size for each input file, and the dimensions of the printed page

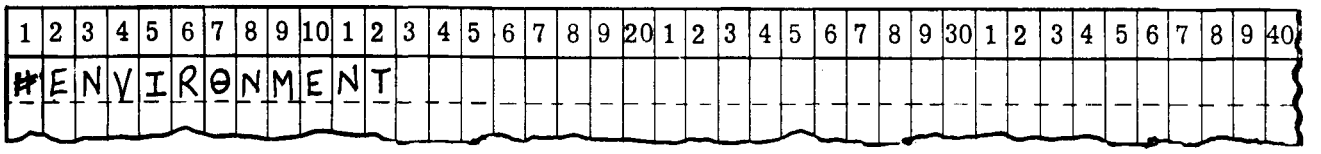
4 Own coding entry parameters: there are up to four possible own coding entries as explained on page 33

The parameters are punched as follows:

Environment group directive parameter (#ENVIRONMENT)

Field	*Columns*	*Contents*
A	1 to 12	The directive #ENVIRONMENT

Example



Keys parameter (#KEYS)

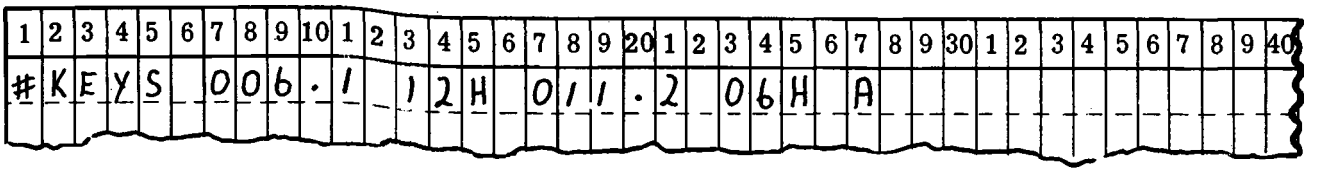
This parameter is in standard single key form. It is used to identify a record by indicating to the program where the keys are to be found on the record and is used only if several files are to be input to the object program.

Field	*Columns*	*Contents*
A	1 to 5	The directive #KEYS
B	7 to 11	The start address of a key, expressed in words and characters relative to the start of the tape record.
C	13 to 15	Field size: Columns 13 and 14 contain a two digit number giving the length Column 15 contains a single character indicating the type of field: H = character ascending % = binary ascending D = character descending * = binary ascending

Fields B and C may be repeated to give a maximum of six keys.

D	67	Designation: File identifier, a single alphabetic character. Note: If there are fewer than six keys, this identifier follows the last named key after a space of one column.

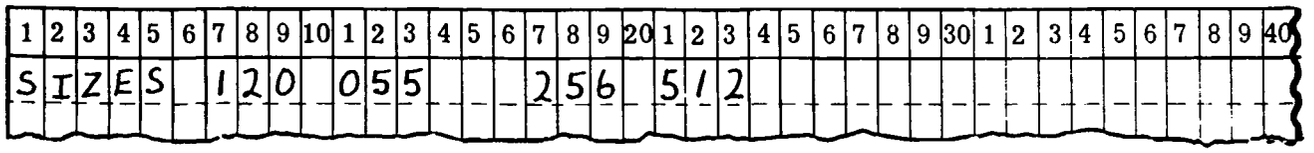
Example



Record sizes parameters

Field	*Columns*	*Contents*
A	1 to 5	The identifier SIZES
B	7 to 9	Line size: number of characters in one line of printout (96, 120 or 160). Three digits, numeric
C	11 to 13	Entries: defines the number of normal printing lines on a page. Three digits, numeric
D	17 to 19	Record size in File A: three digit integer. Record length in words (as in word 0 of record.)
E	21 to 23	Maximum block size in File A: three digit integer
F and G	25 to31	As for Columns 17 to 23 for File B, if necessary.
H and I	33 to 39	As for Columns 17 to 23 for File C, if necessary.

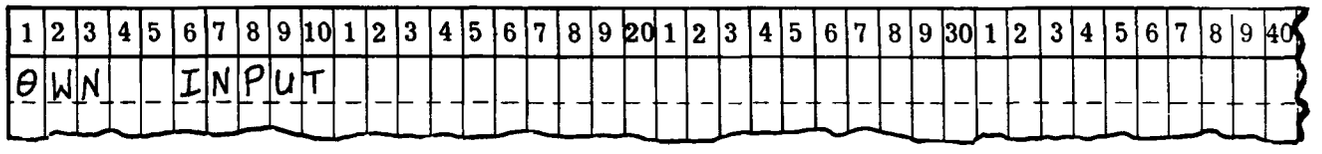
Example



Own coding entry parameter

Field	*Columns*	*Contents*
A	1 to 3	The identifier OWN
B	6 to 10	Entry point.These columns are punched as follows:
		INPUT: enter own coding after each record is read and matched with Files B and C, provided no inhibition occurs.
		PRINT: enter own coding as first routine to be entered in the =RECORD set.
		BREAK: enter own coding as first routine at each control break.
		TOTAL: enter own coding as first routine to be performed at end of file.

Example



CONSTANTS PARAMETER GROUP

This group of parameters, which is optional, consists of

1 The group directive (#CONSTANTS)

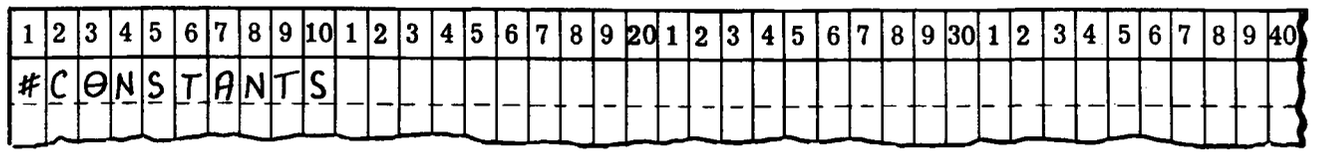
2 A constant description parameter for each constant required, whatever its form. Continuation cards may be punched

These parameters are punched as follows.

Constants group directive parameter (#CONSTANT)

Field	*Columns*	*Contents*
A	1 to 10	The directive #CONSTANTS

Example

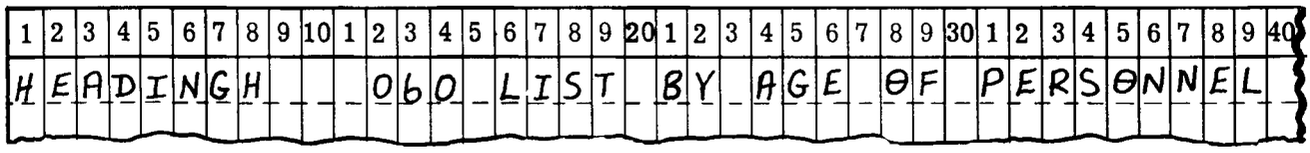


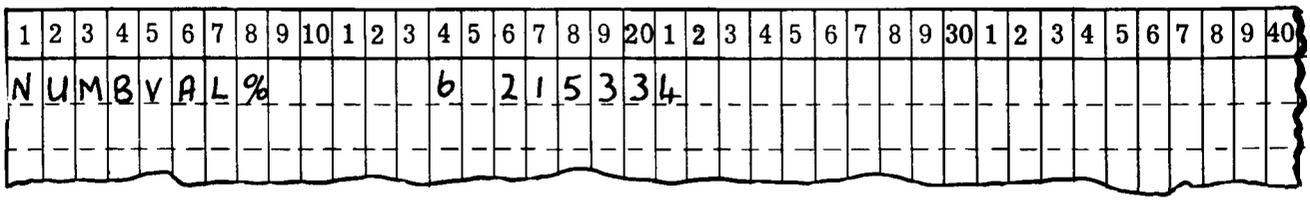
Constant description parameter

Field	*Columns*	*Contents*
A	1 to 8	Constant label: seven characters, first alphabetic.
		Column 8: Type code.
B	10	Storage code

Field	*Columns*	*Contents*
C	12 to 14	Size of constant: number of characters in the value field on cards or paper tape (including any on continuation cards). Note: In whatever form constants are to be held, this field gives the length of the field in its external form in the parameter, not its length in store.
D	16 to 75	Value of the constant in characters; left justified. Note: A sterling value is given in the form of a string of four or more characters. The last four characters will be assumed to be a two-digit shillings value and a two digit pence value. That is, the string is of the form ££££.....££*ssdd*

Examples

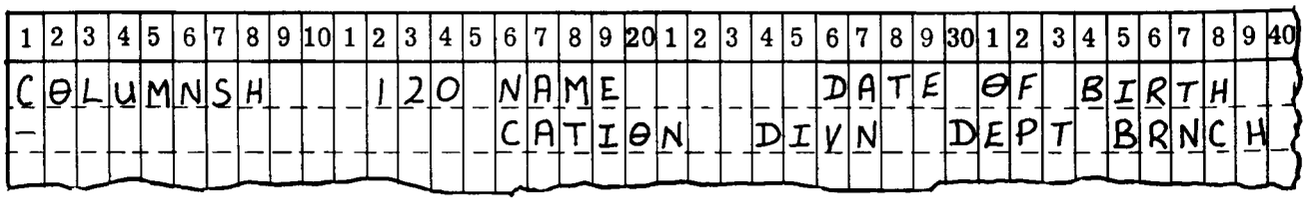


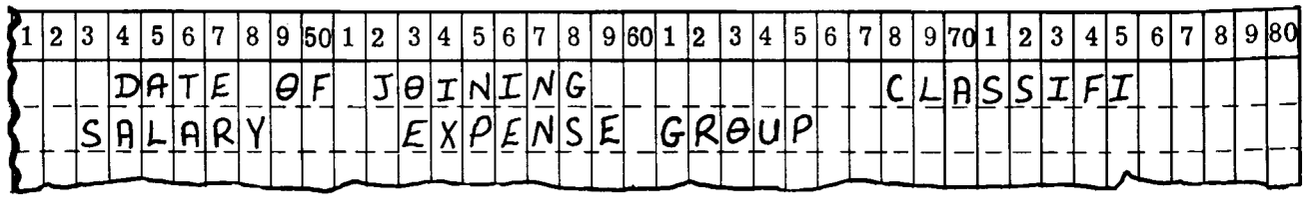


If a continuation card is necessary it is punched as follows:

Field	*Columns*	*Contents*
A	1	- (hyphen)
B	16 to 75	Continued value of constant.

Example





RECORD SELECTION PARAMETERS GROUP

This group of parameters, which is optional, consists of

1 The group directive, which is either #SKIP or #ONLY

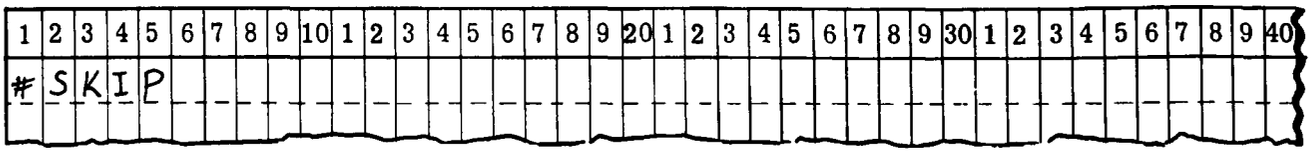
2 Inhibit processing parameters, which determine which records on the input file are to be processed. Fields specified in the Inhibit processing parameter are checked against a range of values

Record selection group directive parameter (#SKIP or #ONLY)

Field	*Columns*	*Contents*
A	1 to 5	The directive #SKIP or #ONLY.

If this is #SKIP, processing is inhibited for the records specified in the inhibit processing parameter: if it is #ONLY, processing is inhibited unless the record is specified.

Example



Inhibit processing parameter

Field	*Columns*	*Contents*
A	1 to 9	Label of field to be checked
		Columns 1 to 7 contain a seven-character label.
		Column 8 contains type code
		Column 9 contains source code
B	12 to 20	Label of field holding minimum value.
		Columns 12 to 18 contain a seven-character label.
		Column 19 contains type code
		Column 20 contains source code.
C	22 to 29	Label of field holding maximum value

Note: Fields B and C refer to the labels of values previously set up in the Constants group. The two values are taken as a range within which the specified field must fall if it is to be operated on by #SKIP or #ONLY. If only a single value is required, both Fields B and C must contain the same label. If more than one inhibit processing parameter is used under #SKIP the satisfaction of any one is sufficient to inhibit processing: if more than one inhibit processing parameter is used under #ONLY all the parameters must be satisfied for processing to be permitted. #SKIP and #ONLY cannot be used together.

Example

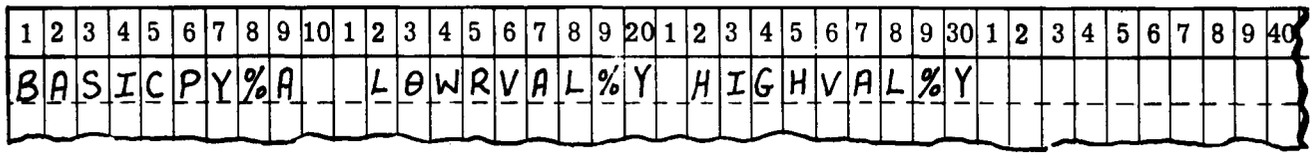


TABLE PARAMETERS GROUP (#TABLE)

This group of parameters, which is optional, consists of

1 The group directive (#TABLE)

2 A table header parameter for each table given, including the table number and the label of the field to be interpreted by it

3 Table description parameters for each table: continuation cards may be punched

Table group directive parameter

Field	*Columns*	*Contents*
A	1 to 6	The directive #TABLE

Example

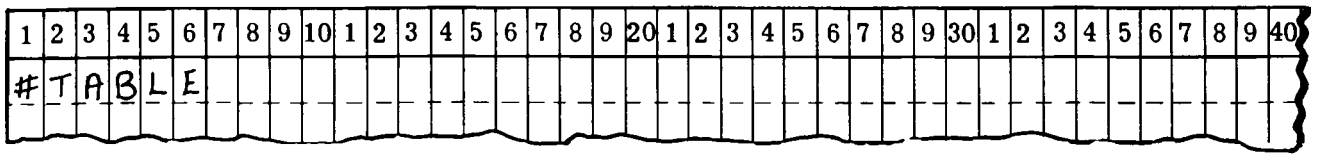


Table header parameter

Field	*Columns*	*Contents*
A	1 to 5	Table identifier. TAB followed by two-digit table number.
B	7 to 15	Code field.
		Columns 7 to 13: seven character field label.
		Column 14: Type code
		Column 15: Source code
		The field label identifies the field on tape to be interpreted.

Example

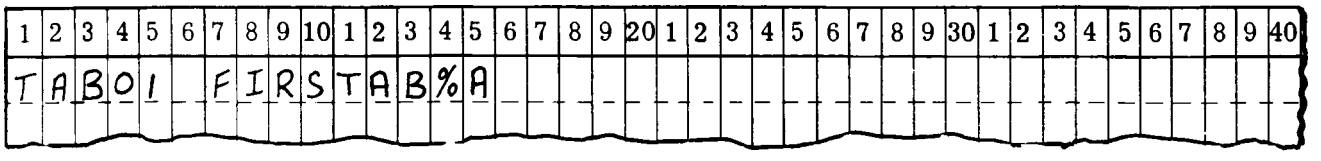
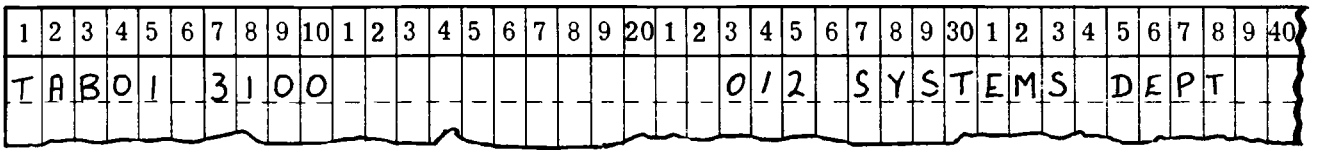


Table description parameter

Field	*Columns*	*Contents*
A	1 to 5	Table identifier. TAB followed by two-digit table number
B	7 to 21	Code value. Fifteen-character field giving the value of the code field to be interpreted by the table. Binary fields should be specified in character form. This field must be left-justified.
C	23 to 25	Length of text field in characters. Three-digit number.
D	27 onwards	Text field to be output in characters. If more than 48 characters are required a continuation card may be punched.

Example



A continuation card should be punched as follows:

Field	*Columns*	*Contents*
A	1	- (hyphen)
B	27 to 74	Continuation of text field to be output.

CALCULATIONS PARAMETERS GROUP

This group of parameters, which is optional, consists of:

1 The group directive (#CALCULATIONS)

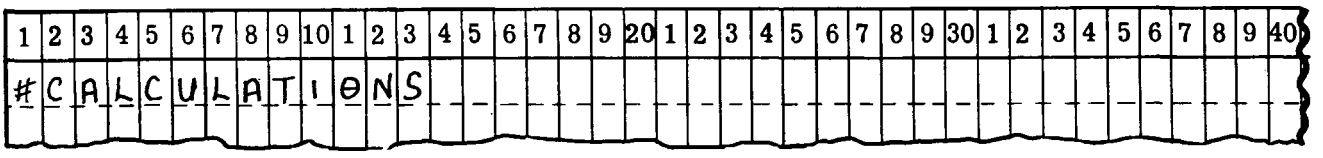
2 A description parameter for each calculation or comparison to be carried out, giving the labels describing where the factors are to be found, defining the operation to be carried out on the factors, and giving a label for the result field.

Calculations or comparisons must be numbered consecutively, beginning with 01.

Calculations group directive parameter (#CALCULATIONS)

Field	*Columns*	*Contents*
A	1 to 13	The directive #CALCULATIONS

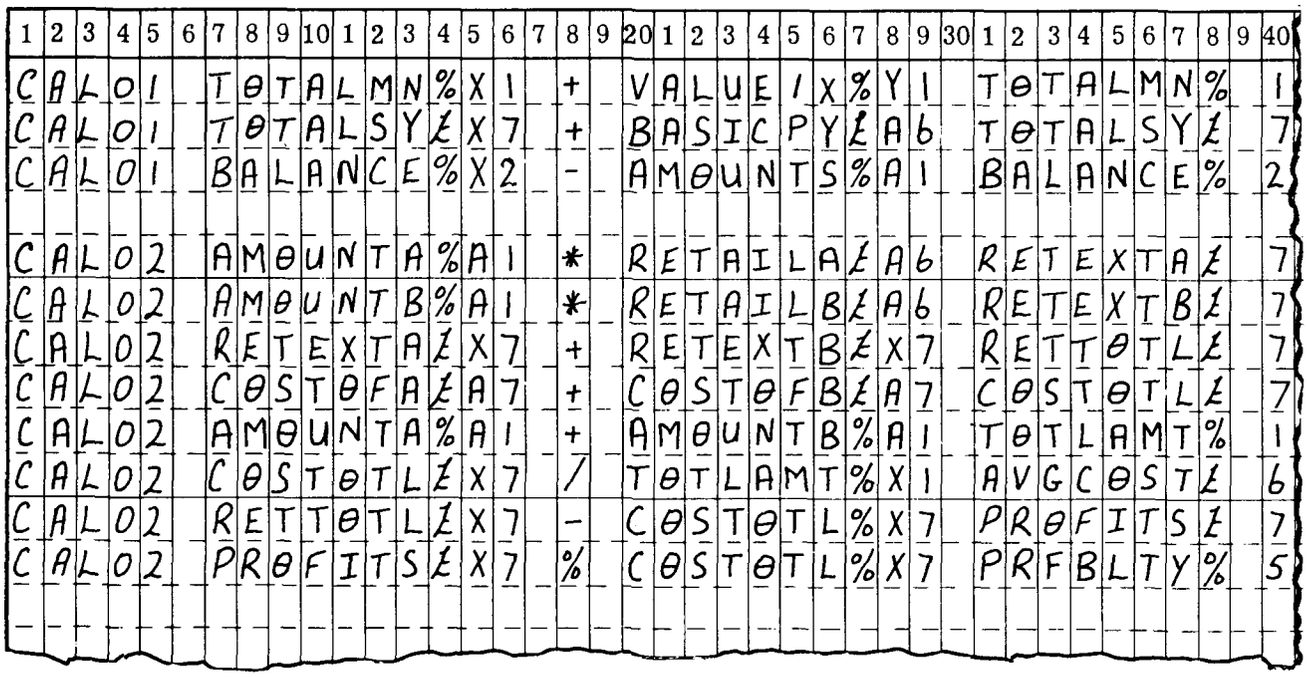
Example



Calculation description parameter

Field	*Columns*	*Contents*
A	1 to 5	Calculation identifier
		CAL followed by a two-digit calculation number
B	7 to 16	Label of first factor of calculation
		Columns 7 to 13 contain a seven character label
		Column 14 contains the type code
		Column 15 contains the source code
		Column 16 contains the storage code
C	18	Operation. Single character, as follows:
		(P = 1st factor, Q = 2nd factor, R = Result)
		+ : Add P to Q, sum in R.
		– : Subtract Q from P, difference in R.
		* : Multiply P and Q, product in R.
		/ : Divide P by Q, unrounded quotient in R.
		% : P as percentage of Q, result in R.
D	20 to 29	Label of second factor of calculation
		Columns 20 to 26 contain a seven character label.
		Column 27 contains the type code
		Column 28 contains the source code
		Column 29 contains the storage code
E	31 to 40	Label of result field.
		Columns 31 to 37 contain a seven-character label (first character alphabetic, other alphanumeric).
		Column 38 contains the type code (% or £ only)
		Column 40 contains the storage code
		Note: Source code need not be specified for the result field (Column 39 is blank), because this field will always be in the results area and source code X is assumed.

Example

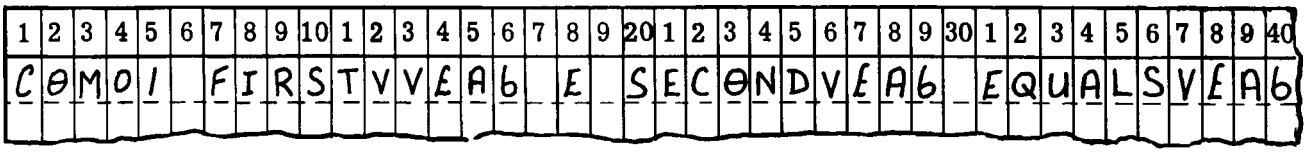


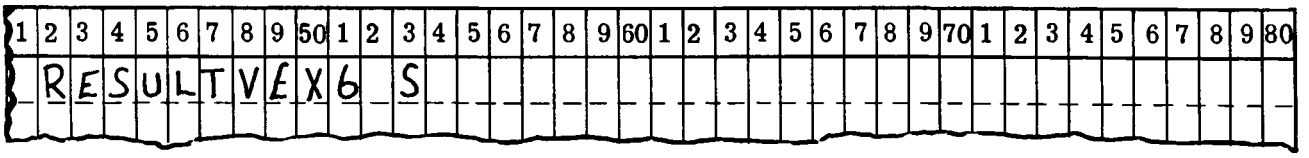
Comparison description parameter

Field	*Columns*	*Contents*
A	1 to 5	Comparison identifier
		COM followed by a two-digit comparison number.
B	7 to 16	Label of first field for comparison
		Columns 7 to 13 contain seven character label.
		Column 14 contains the type code
		Column 15 contains the source code
		Column 16 contains the storage code
C	18	Relation. Single character, as follows:
		L : Less than
		M : less than or equal to
		G : greater than
		H : greater than or equal to
		E : equal to
		N : unequal to
D	20 to 29	Label of second field for comparison
		Columns 20 to 26 contain a seven character label
		Column 27 contains the type code
		Column 28 contains the source code
		Column 29 contains the storage code
E	31 to 40	Label of field to be stored in result field if the relation is satisfied.
		Columns 31 to 37 contain a seven character label
		Column 38 contains the type code

Field	*Columns*	*Contents*
		Column 39 contains the source code
		Column 40 contains the storage code
F	42 to 51	Label of the result field
		Columns 42 to 48 contain a seven character label
		Column 49 contains the type code
		Column 50 contains the source code
		Column 51 contains the storage code
G	53	Clear code. Single character, defining the action to be taken with the result field if the relation is not satisfied.
		∇ : do nothing
		S : spacefill
		C : zerofill

Example





HEADINGS PARAMETER GROUP

This group of parameters, which is optional, consists of:

1 The group directive (#HEADINGS)

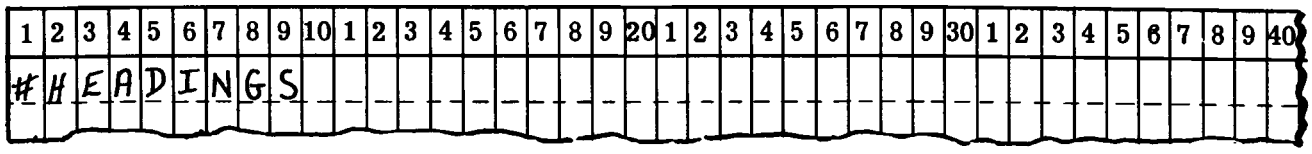
2 A heading identifying parameter, which indicates for each heading whether a main or page heading is being specified.

3 One or more line description parameters, which specify the output format on the printer.

Headings group directive parameter(#HEADINGS)

Field	*Columns*	*Contents*
A	1 to 9	The directive #HEADINGS

Example

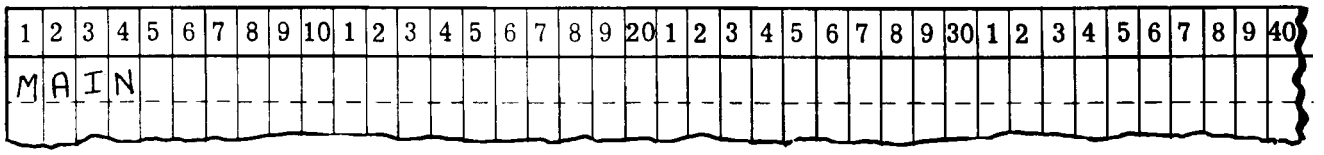


Heading identifier parameter

Field	*Columns*	*Contents*
A	1 to 4	MAIN or PAGE

Field	*Columns*	*Contents*
		A MAIN heading appears only once on the first page of the report. A PAGE heading appears at the top of each subsequent page.

Example

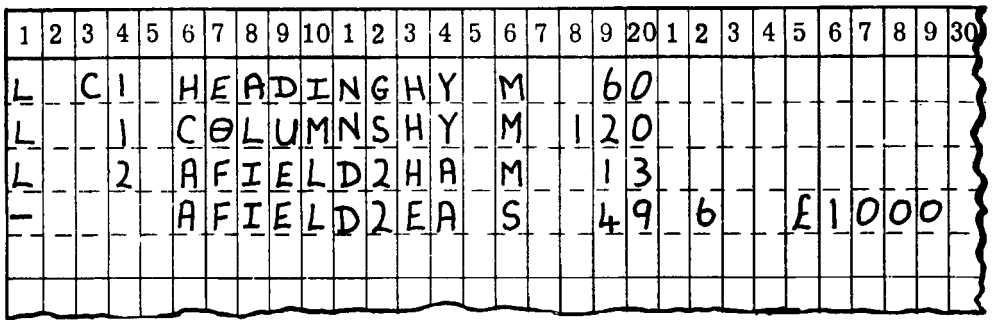


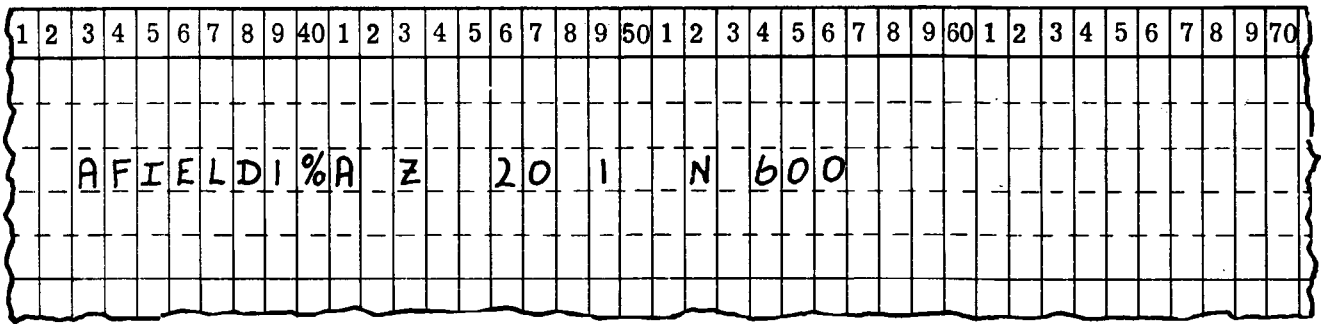
Line description parameter

Field	*Columns*	*Contents*
A	1	L for the first card - (hyphen) for a continuation card
B	3 to 4	Spacing: a two-digit number of lines to be spaced before the line is printed. If a C is punched in Column 3, Column 4 gives the channel in the paper tape loop used to control spacing for this line. This field will be left blank if successive fields are being moved to form the same line.
C	6 to 14	Field label Columns 6 to 12 contain a seven-character label. Column 13 contains type code Column 14 contains source code
D	16	The operation to be performed on the field before it is printed. M : move field unchanged E : (mid point numbers only): insert decimal point and zero suppress Z : zero suppress S : sterling edit
E	18 to 20	Print position: a three-digit number giving the right-hand print position of the field to be printed.
F	22	Storage code
G	24 to 29	Print format. The contents of this field vary according to the type of field to be printed. Character fields : whole field left blank Numeric fields : as follows 24 : ∇ (space) 25 : N 26,27 : number of whole integers 28,29 : integer fields 00 : mid point fields number of places after the decimal point Sterling fields : as follows 24 : * for check protection asterisks : ∇ for floating £ sign 25 : C 26,27 : maximum number of characters in the output field (excluding +, – and £ signs)

Fields	*Columns*	*Contents*
		28, 29 : 00
H	31	Clear code: contains a C if the output field is to be zeroized after printing.
I	33 to 58	Description of second field, as C to H above.

Example





RECORD LEVEL PARAMETERS GROUP

This group of parameters consists of:

1 The group directive (#RECORD)

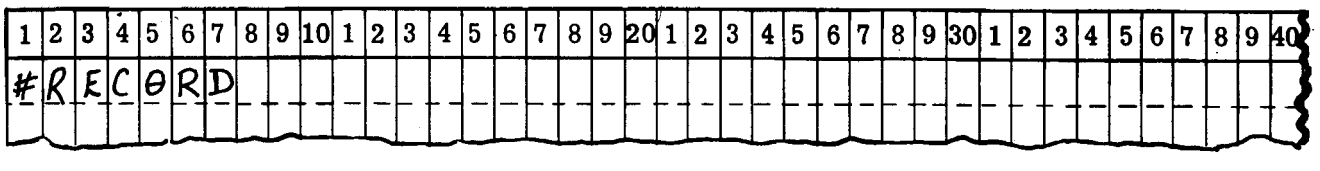
2 Optional calculation list parameters, each listing the identifiers of the calculations and comparisons to be performed in the processing of the record

3 Optional line description parameters, each giving the location of a line to be printed, any operations to be performed on it, and the print formats to be used (as described above)

Record level group directive parameter (#RECORD)

Field	*Columns*	*Contents*
A	1 to 7	The directive #RECORD

Example

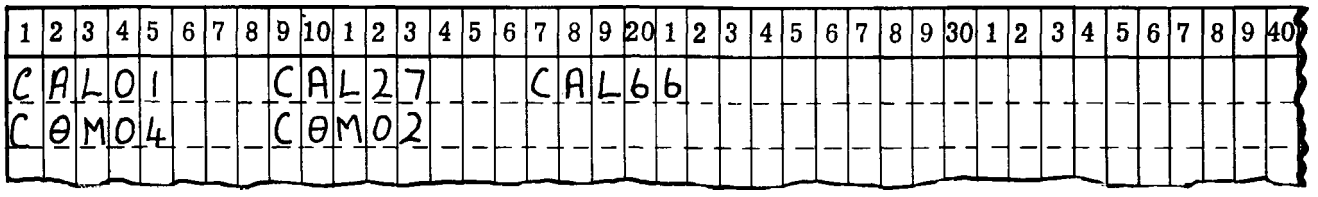


Calculation list parameter

Field	*Columns*	*Contents*
A	1 to 5	First calculation or comparison identifier. CAL or COM and a two-digit number.

Field	*Columns*	*Contents*
B	9 to 13	
C	17 to 21	
D	25 to 29	
E	33 to 37	Further calculation or comparison identifiers: these have the same format as A.
F	41 to 45	
G	49 to 53	
H	57 to 61	
I	65 to 69	

Example



Note: Calculations and Comparisons may be called in any order. The order does not have to correspond to the sequence in which they were originally defined.

CARRY PARAMETERS GROUP

This group of parameters, which is optional, consists of:

1 The group directive (#CARRY)

2 Calculation list parameters, as in the Record levels set (page 29)

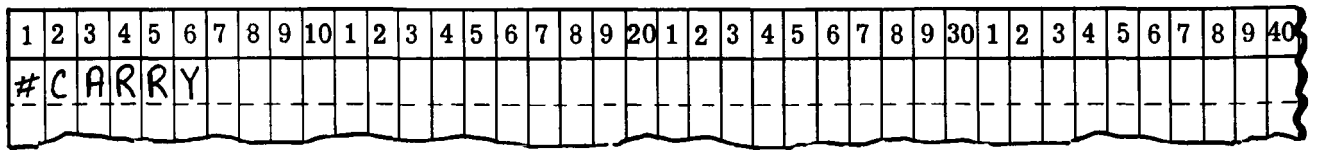
3 Line description parameters, as in the Heading parameter group (page 27)

This set is needed only if special printing action is to be taken at the foot of a page, for example to print carry forward totals.

Carry group directive parameter (#CARRY)

Field	*Columns*	*Contents*
A	1 to 6	The directive #CARRY

Example



LEVELS PARAMETERS GROUP

A separate Levels parameter group is defined for each required level of control break.

1 The group directive (#LEVEL)

2 A control field parameter, which specifies in what circumstances a control break will occur. A continuation card may be punched for this parameter

The remaining parameters of this group describe the calculations and further activities to be undertaken at a control break. All these parameters are identical with the corresponding ones described earlier. They may include:

3 Calculations list parameters, as in the Record levels group (page 29)

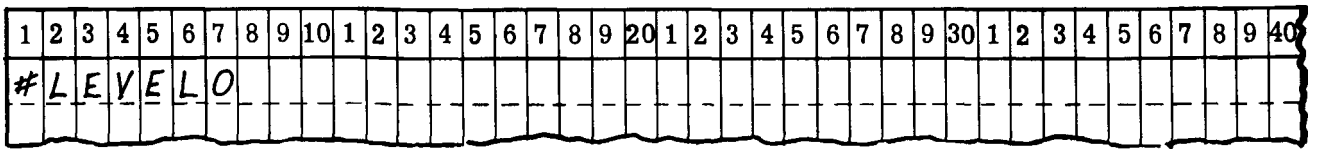
4 A heading identifier parameter (PAGE), as in the Headings group (page 27)

5 Line description parameters as in the Headings group (page 27)

Levels group directive parameter (#LEVEL)

Field	*Columns*	*Contents*
A	1 to 7	Group directive: Columns 1 to 6, the directive #LEVEL; Column 7, the level number, a single digit, 0 to 9

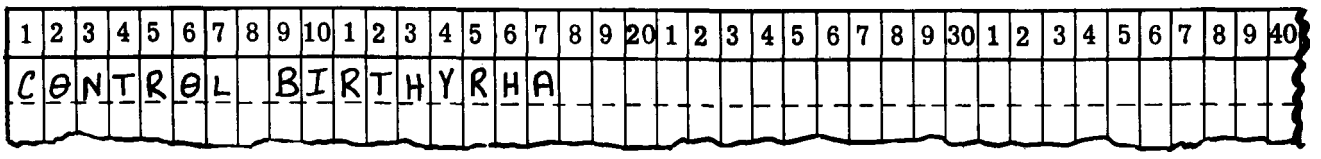
Example



Control field parameter

Field	*Columns*	*Contents*
A	1 to 7	CONTROL for the first card, - (hyphen) in Column 1 for continuation card.
B	9 to 17	Control field label Columns 9 to 15 contain a seven-character label Column 16 contains type code Column 17 contains source code
C	19 to 33	Control value, left justified. If this field is blank, a control break occurs wherever the value of the record control field changes. If this field contains a constant, then a control break occurs when the value of the record control field equals the value of this constant. If the control field is in binary, this field is specified in characters and will be translated before comparison.
D	37 to 51	Second control value, as Columns 19 to 33.
E	55 to 69	Third control value, as Columns 19 to 33.

Example



TOTALS PARAMETERS GROUP

This group of parameters, which is optional, consists of

1 The group directive (#TOTALS)

The remaining parameters which may be included in this group describe the calculations and further activities which are to be carried out at the end of the run. Their formats may be seen on the pages referred to:

2 Calculations list parameters (page 29)

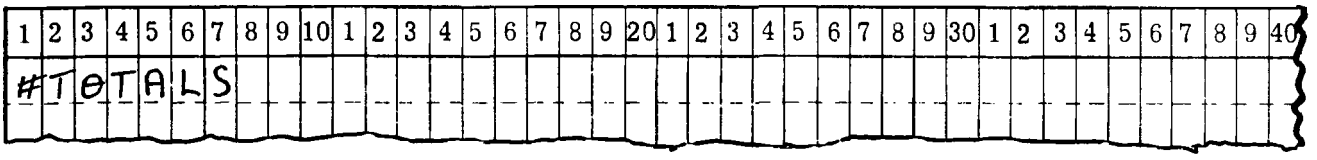
3 Heading identifier parameters (page 27)

4 Line description parameters (page 28)

Totals group directive parameter (#TOTALS)

Field	*Columns*	*Contents*
A	1 to 7	The directive #TOTALS

Example

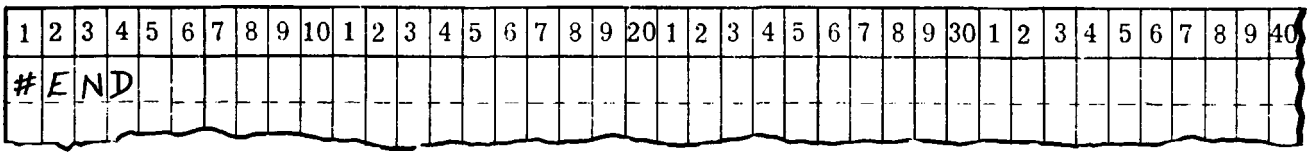


END OF PARAMETERS MARKER

The end of parameters is indicated by a card punched as follows:

Field	*Columns*	*Contents*
A	1 to 4	The directive #END

Example



ERROR MESSAGES

Invalid parameters are flagged by the printing of an error label next to the parameter during the listing of the parameter cards. The following flags can occur.

INV : The parameter is miscoded

LAB : The label quoted has not been properly defined previously (for example in the dictionary set) or is miscoded in some way (for example, it contains non-alphanumeric characters)

MIS : A mandatory parameter is missing

SEQ : Cards are out of sequence, or there are no sequence numbers

When all the parameters have been processed REPORT will halt with the message PARAMETER SET INVALID if an INV, LAB or MIS was detected during the processing.

RUN TIME PARAMETERS

Certain changes may be made to the parameter set at the time of the object program run. The only changes allowed are to the #READ and #WRITE parameters, and to the Constants parameter group.

#READ and #WRITE parameters should be supplied in exactly the same format as for the original parameter set, except for the changes being made.

The Constants parameter group supplied at run time should be exactly the same as that for the equivalent Constants group in the source parameter set, although the value of each constant may be changed. A group directive and all subsequent parameters are required. When changes are made to any of the constants at run time, all the constants up to and including the last constant that has to be changed must be input again. This may mean that some constants whose values have not been changed will have to be input again if they are stored before a constant that is to be changed. It is therefore advisable to place variable constants at the beginning of the constants area.

The run time parameter set must end with a #END parameter in the usual format.

Chapter 4 Own coding facilities

Own coding segments may be included in a REPORT program by specifying the call times in an own coding entry parameter (page 21); they should be input immediately following the parameter set.

Own coding segments are read and checked immediately after the #END parameter of the parameter set. The program checks all segment names and cues against the own coding call names. If an own coding parameter has called for a particular segment of own coding, and if this segment is not provided, then after the own coding has been read the message NEEDS CUENAME is output.

Own coding must consist of standard PLAN segments each beginning with #PROGRAM and ending with #END. Any number of segments may be input; the last segment should be followed by #FINISH.

CALL TIMES

Own coding segments are called according to the punching in the own coding entry parameter. The times are as follows:

1 OWNINPUT: Any own coding segment is called after the input records have been matched, provided that none of them inhibit processing. This own coding is performed before the program makes any of the checks for control breaks

2 OWNPRINT: Any own coding segment is called immediately before any action is taken for the record group of parameters. On exit from the own coding segment the record group will be performed

3 OWNBREAK: Any own coding segment is called immediately before control break action is taken. On exit from the own coding segment the particular levels group and any lower level calls resulting from it will be performed

4 OWNTOTAL: Any own coding segment is called immediately before any action is taken for the totals group of parameters. On exit from the own coding segment the totals group will be performed and the program will then halt

Figure 3 provides a diagrammatic representation of the call times for own coding segments.

The occurrence of an own coding parameter will cause REPORT to generate a call to an own coding segment, as for example

```
CALL        1        OWNPRINT
```

The user's own coding must therefore contain a cue of the form

```
#CUE                 OWNPRINT
```

When the user's own coding has completed whatever processing it has been designed to do, it must return control to the main program by means of an instruction of the form

```
EXIT        1        0
```

Note: Accumulator 1 is used as the link accumulator for all own coding segments.

COMMON AREAS

During the preparation of own coding segments, it may be necessary to refer to common areas held with the REPORT program coding. These areas are named by the generated program.

```
#LOWER      COMMON/RECORDAREAS/
#LOWER      COMMON/PRINTAREAS/
#LOWER      COMMON/CALCULATION/
#LOWER      COMMON/LEVEL NUMBER/
```

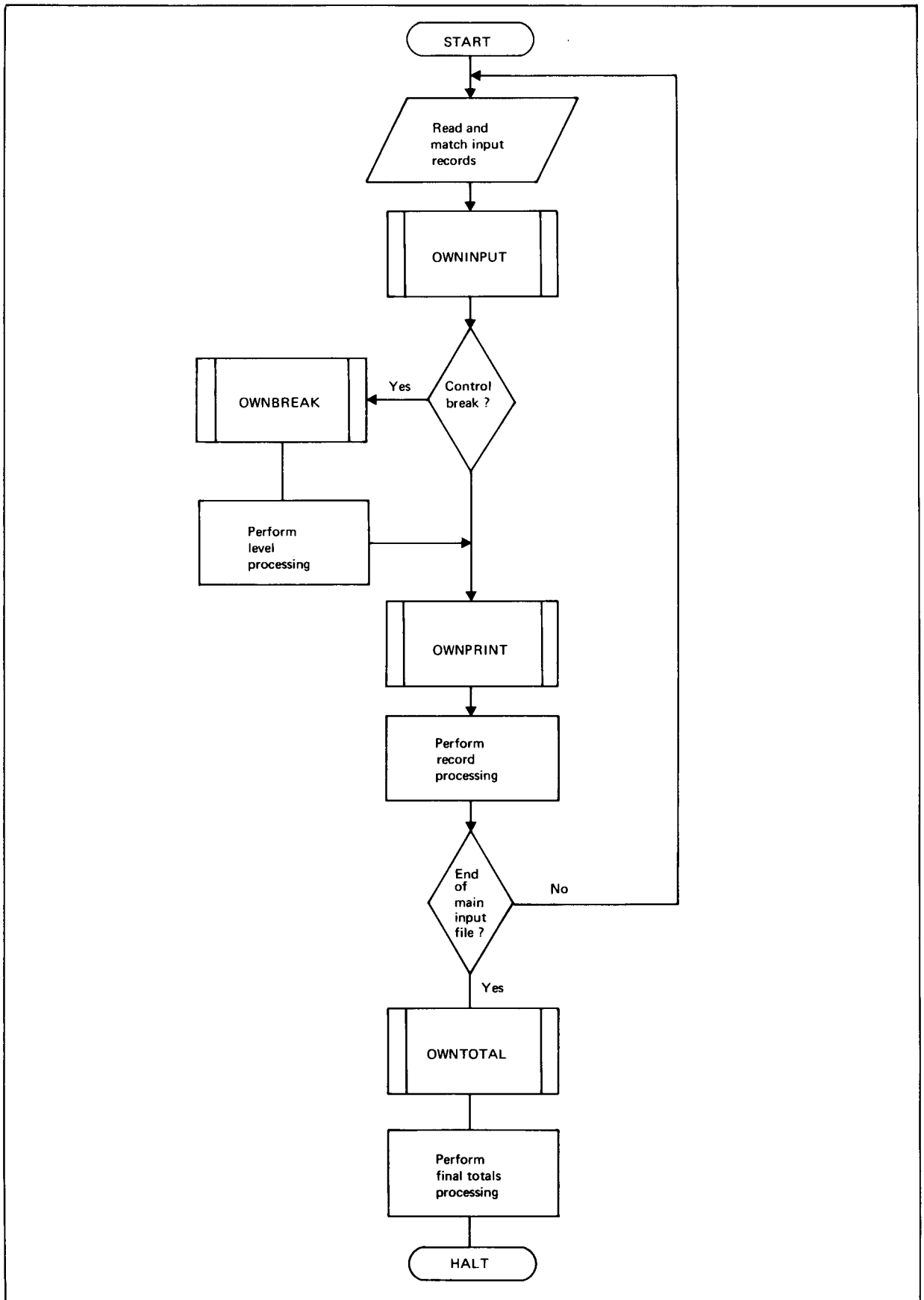


Figure 3 Own coding call times

#LOWER COMMON/USER

As usual, the user may give his own labels to fields within the common areas, but he cannot alter the structure or layout of these areas. The structure of the common areas is described below.

Record areas
Under this heading come all the input file areas of the tapes defined by the #READ parameters. All areas are of the record length defined in the SIZES parameter.

Record areas are named as follows:

RECDA: for File A

RECDB: for File B (if present)

RECDC: for File C (if present)

Note: RECDB and RECDC will not be defined if files B and C are not present.

Print areas
These consist of

1 A one word area called CHAR containing the print control character

2 A 41 word area called PRINT containing the next print line (the line about to be printed) with the *print selection word* as its 41st word (as used by REPRINT, see Chapter 8, page 45).

Calculation area
This area will contain all labels generated by the calculation routine: the storage code will always be X. The labels will be the same as defined, except that the eighth character is translated as follows.

£ = 2

% = 1

and a 9th character of X is added. Double length locations are provided as necessary and the labels will be in the order defined.

Level number area
This is one word area called LEVELNO which contains the level number of the current control break.

User area
This area is made up of the following:

1 A one word area called ITEMA containing the count of records in File A

2 A one word area called ITEMB containing the count of records in File B (if needed)

3 A one word area called ITEMC containing the count of records in File C (if needed)

4 A one word area called ECOUNT containing the count of the number of entries printed

5 A one word area called PCOUNT containing the count of the number of pages

6 A one word area called EPAGE containing the preset line count

7 A two word area called DATE containing the current date in the form DD/MM/YY as supplied by Executive

Example
These lines of coding are an example of the beginning of an own coding segment.

LABEL 1–6	OPERATION 7–12	ACC. 13–15	16–72	PROG. IDENT. 73–75	SEQUENCE 76–80
#PROGRAM			NAME/OWNINPUT		
#LOWER			COMMON/RECORDAREAS/		
			RECDA(3,6)		
#LOWER					
			LINK,WORK		
#PROGRAM					
#CUE			OWNINPUT		
	STO	1	LINK		

PRINT LAYOUT CHART | PROGRAM/SYSTEM | LAYOUT NAME | Design by

```
'MAIN'   LIST BY AGE OF PERSONNEL RECORDS ON PERSONNEL FILE
'PAGE'                                                  PAGE XXXXX
'PAGE'   DATE OF BIRTH   PERSONNEL NO.   TITLE AND NAME          DEPARTMENT          SALARY
'DATA'     DD MM YY        XXXXXX       NNNN N N NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN £XXXXX/XX/XX
           DD MM
           DD
           DD MM YY        XXXXXX
'GROUP'    TOTAL SALARY    £XXXXXX/XX/XX
           AVERAGE SALARY  £XXXXXX/XX/XX
           NO.IN GROUP           XXXXX
'next data' DD MM YY
'last group'  NO. IN GROUP           XXXXX
'TOTAL'    GRAND TOTAL     £XXXXXXX/XX/XX
```

Figure 4 Print layout chart for report example

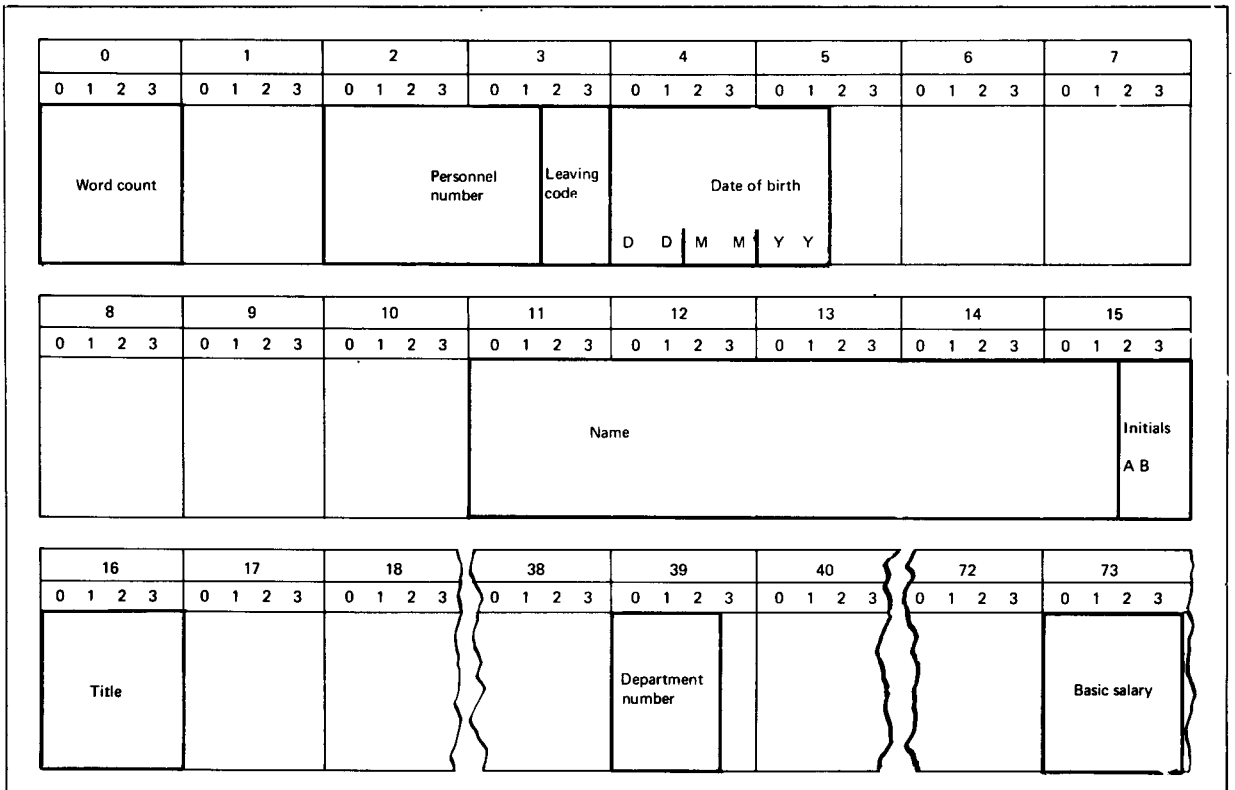


Figure 5 Personnel file record layout

Chapter 5 Example

PLANNING A REPORT

As the user has to prepare a long parameter set for input to REPORT previous planning is usually needed. Much of this planning is concerned with the action to be taken by the program at control breaks. In the initial planning of a report the user should consider:

1 The way in which the program will detect that control breaks have occurred, together with the calculations to be performed and the details to be printed at control breaks

2 The files on which the data to be input in the preparation of the report is held

3 The format in which records have been stored on the input files

4 The maximum block size on the input files

5 The calculations to be performed on each record, and the details which are to be printed from each record

6 The main and page headings required in the printout

7 Which final totals are to be calculated at the end of the run

8 Whether the output is to be in the form of a direct printout or a print tape

The user may find the use of detailed record layout charts and print format charts convenient at this stage.

In the example given in this chapter a personnel file has been chosen to illustrate the use of REPORT.

A detailed listing of all personnel with the exception of those about to leave, is required. The list will be grouped by year of birth and the total basic salary for each group will be printed together with the group's average salary. Figure 4 illustrates the required form of the listing and Figure 5 illustrates the record layout of the personnel file concerned.

PREPARING THE PARAMETER SET

The print layout, Figure 4, shows that all headings and certain annotations, for example TOTAL SALARY, must be inserted as constants. A single line of print is required for each person. All the relevant information for each person can be found in the personnel record. All the necessary fields must be defined in the dictionary, while the department name is held as a code with a simple table to convert it to the full name.

As the only control key is the year of birth the file should have been previously sorted with year of birth as the major key. Any subsidiary keys that were used in sorting are not important to the program but they may have been selected so that within each group the list would be in a logical sequence, for example by department, date of birth, alphabetical sequence and so on, or some combination of these.

The items to be printed at group level, the total and average salaries, are not directly available from the personnel file so they have to be accumulated during the run. The group total will be accumulated as a simple total. The average salary, however, will be obtained by the setting up of a counter, requiring a constant set at 1, to calculate the number of people in the group. This will be followed by the division of the group's total salary by this number. The grand total can be obtained by the addition of either all the basic salaries or all the group salaries.

For this example the following parameters are required.

Preliminary parameters group

The first parameter gives the name PSR2 to the object program.

The second parameter gives the label of the input file. This label may be changed by run time parameters.

Dictionary group

Only one input file, File A, is specified: the fields relevant to this report are described in the dictionary. Nine fields are to be defined for direct printing purposes, one field is to be printed by use of a table and one field is to be defined for record skipping purposes.

The fields are listed in ascending sequence of addresses, with unique 8 character labels. The final character indicates the field type. Field BASICPY£, for example, is a sterling field held as binary pence in one word; the other fields are in character form, 06H being 6 characters.

Environment group
If more than one input file was specified, the keys would.be defined here but in this example only one file is used. The SIZES parameter indicates that the maximum record and block sizes are 256 and 512 words respectively. This parameter also defines the print line size as 120 characters and the number of normal print lines as 55; space is allowed on the page for totals to be printed before the page is changed in the event of page overflow coinciding with a control break.

The Environment group would also define own coding entry points if these were required.

Constants group
Headings required in the printout are specified here together with printing details and values to be used by record selection parameters.

The headings have suitable 8 character labels as in the dictionary. As HEAD2XXH is 120 characters long a continuation card is needed.

The character constants RANGEAXH, RANGEBXH etc. are used in the record selection group to identify the leaving code values.

The constant NUMBONE% is a binary constant of value 1, held as one binary word; this is required in a later calculation.

The last constant, SPACESXH, is used as a dummy print line after the totals to improve the appearance of the report.

Record selection group
In this example, record selection is made under the #SKIP directive. The #ONLY directive could have been used instead.

There are three skip conditions all operating on the same input field. Records in File A with a code field defined as LEAVCODH will be ignored if the code falls within one of three ranges of values. The upper and lower limits of these ranges are given in the constants group, i.e. L1 to L9, P1 to P9 and T1 to T9.

Tables group
This group contains a table of department numbers and names. The field DEPTNUMH contains a department code which is to be translated into suitable text before being printed. The table defines a text for each code value.

Calculations group
Four calculations are described. They involve:

1 total salary for the group

2 average salary for the group, i.e. group total divided by number in the group

3 number of people in the group

4 grand total salary

CAL01 accumulates the salary field BASICPY£ into a work area field GPTOTAL£.

CAL02 divides the accumulated salary field GPTOTAL£ by a field called GPCOUNT% and places the result in work area field GPAVRGE£.

Note: Calculations may be defined in any order. The order in which they are *performed* depends on the order in which they are specified in the output groups. It is, therefore, not important that GPCOUNT% is undefined at this stage.

CAL03 adds a constant 1,NUMBONE%, into work area field GPCOUNT%.

CAL04 adds the accumulated salary field GPTOTAL£ into a work area field GDTOTAL£.

This calculation provides for salary totals at two levels: for each control break and a grand total. The field GPTOTAL£ is cleared to zero after each control change. A calculation is therefore required to accumulate each control group total for the final total.

Headings group
Two headings are described. The Main heading will be printed by itself, 20 lines down on a separate page. The field *PCOUNT is a standard field which contains the number of pages printed.

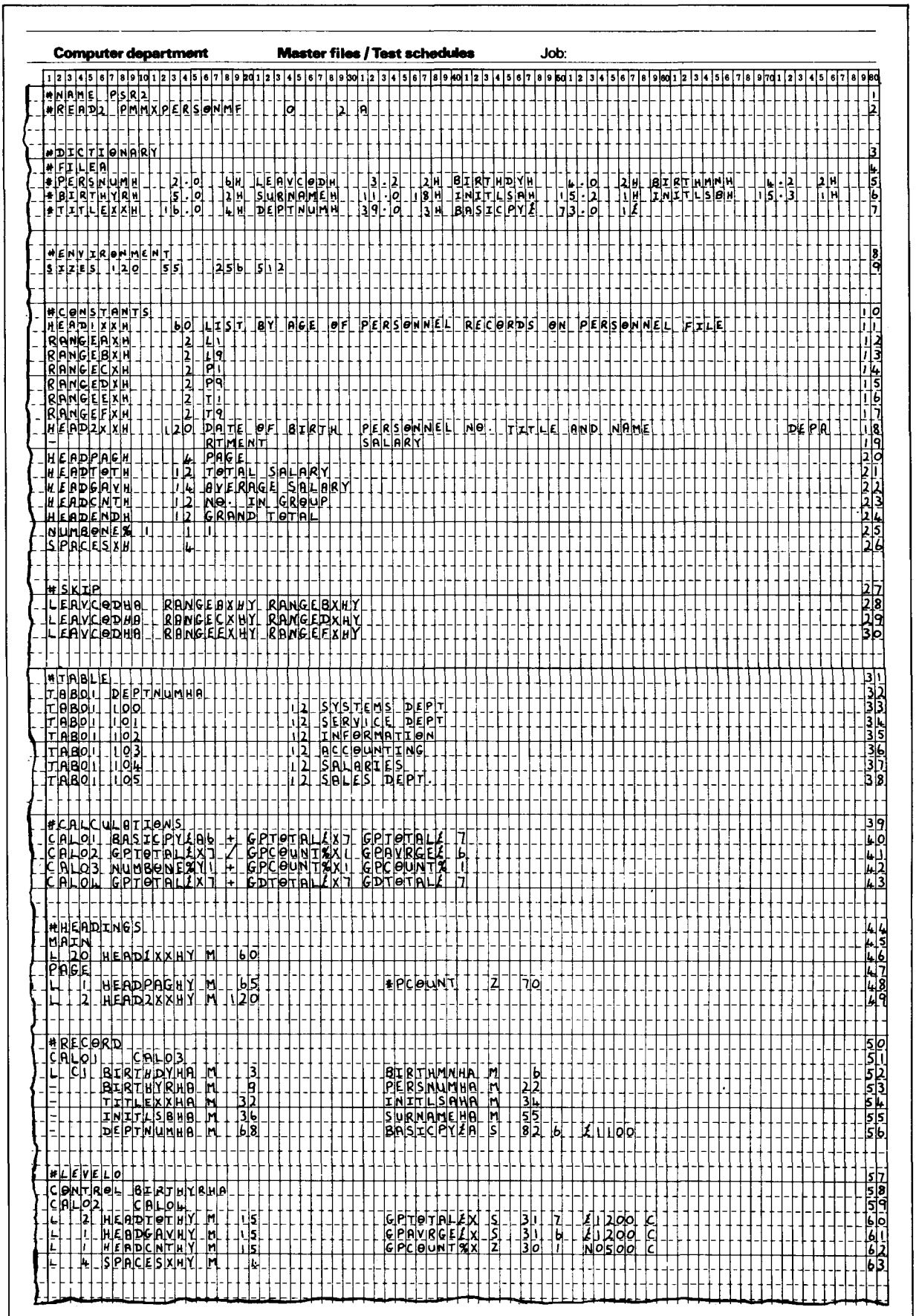


Figure 6 Parameters for report example

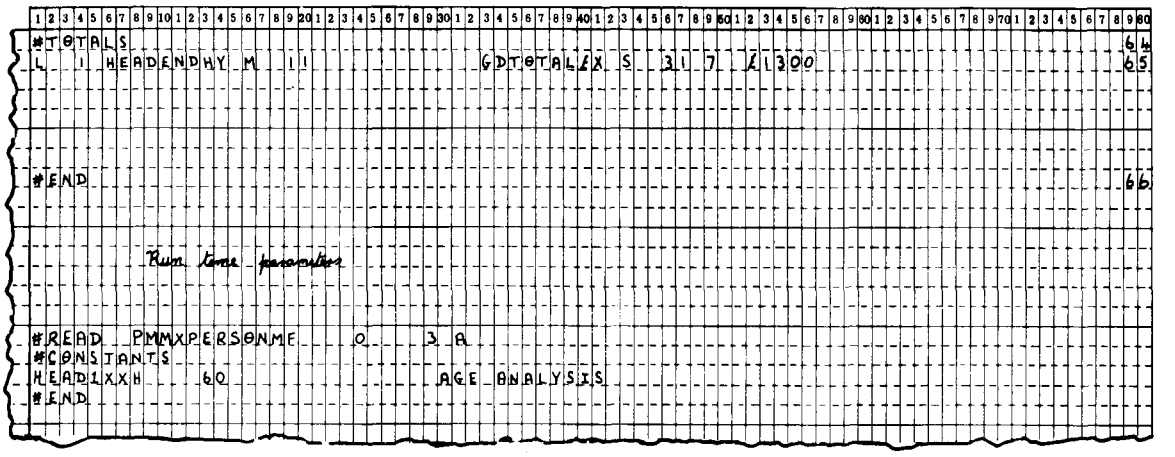


Figure 6 Parameters for report example (continued)

Record levels group
This group describes the actions that are required when each record is read. The calculations CAL01 and CAL03 will be performed and one line will be printed. This line consists of character fields moved unchanged from the input area, and BASICPY£, a single length sterling field held as binary pence. This field is to be edited on output as a character sterling field with preceding £ sign. There will be five pound positions, two shillings and two pence positions with solidus (/) signs between the pound, shillings and pence making eleven characters in all. The edit field is therefore given as £1100. The actual size of the output field in this example will be thirteen characters, since a £ sign and a ± sign position are included on output.

Carry group
In this example no Carry group is given. A Carry group describes any special input required at the bottom of a page upon a page overflow condition, before a new page is printed.

Levels group
The Levels group defines level control breaks in descending order of importance, starting with LEVEL0 as the major level. In this example, only one level control break is defined as the only control key is year of birth.

When the field BIRTHYRH changes, calculations CAL02 and CAL04 are required to obtain the groups average salary and to add the group total into the grand total.

The total and average salaries for the control group will then be printed together with the number of people in the group. Each value field, GPTOTAL, GPCOUNT and GPAVRGE will be zeroized after being printed.

A dummy line is given to obtain extra spacing between the total of one group and the first record of the next group.

Totals group
End of run output is described by the Totals group. A single line giving the final grand total salary is to be printed.

End of parameters marker
The last card is #END.

Run time parameters
When the REPORT parameter set is compiled, it generates a PLAN program named #PSR2. When the program is run with an input file in order to produce a report, it contains an option to read run time parameters.

Run time parameters for this example are illustrated in Figure 6. The input file generation number has been changed from 2, as specified the original parameter set, to 3 and the first heading has been changed. Subsequent parameters are unchanged so no further parameters are required. If, for example, the heading to be changed was HEADENDH, all constants parameters up to and including this one would have had to be input. This is because at run time the CONSTANTS area is overwritten without any check on the actual labels of the headings fields, solely in the order in which the new constants are read.

The generated program
Part of the printout from the resulting program is shown in Figure 2, pages 8, 10 and 12.

Chapter 6 Operating instructions and exception conditions

OPERATING INSTRUCTIONS

Instructions for both the runs involved in the running of the REPORT program are given in this section.

Generation run

The procedure necessary for a generation run with REPORT is as follows:

Narrative	*Console message*
1 Load three scratch tapes	
2 Load the standard Master Library Tape	
3 Load the REPORT program tape	
4 Load the parameters on a card or paper tape reader	
5 Load the program REPORT by inputting	FI#X68G
6 To read in the parameters either (a) if the parameters are on paper tape, input:	GO #X68G 20
or	
(b) if the parameters are on cards, input:	GO #X68G 21
The program prints the parameter listing, halts and outputs the message:	DELETED:- FI#XPLV #TAPE
7 To enter the compiler program #XPLV input:	GO #XPLV 22
#XPLV compiles the object program, halts and outputs the message:	END OF BATCH

Object program run

The generation run being complete, the user may now proceed with the object program run, as follows:

Narrative	*Console message*
1 Load up to three input files	
2 Load a scratch tape for the output file and/or dump file, if required	
3 Load the object program tape	
4 Load the object program *#nnnn* by inputting	FI*#nnnn*
5 Load any run time parameters	
6 If the program is to output to a print tape input:	ON*#nnnn* 11
7 To enter the program (a) if there are parameters on paper tape input:	GO *#nnnn* 20
or	
(b) if there are parameters on cards input:	GO *#nnnn* 21
or	
(c) if no run time parameters are to be used input:	GO *#nnnn* 23
The program prints the check lines and outputs the message:	HALTED:- CH
8 (a) if the paper alignment is correct, input:	GO *#nnnn*

Narrative	*Console message*
(b) if the paper is incorrectly aligned adjust the page and input:	GO *#nnnn* 23
The program prints the report or lists the report to the tape, and outputs the message:	HALTED:- END OF PROGRAM

9 The program can be restarted by returning to step 5.

EXCEPTION CONDITIONS

Generation run

The possible exception condition message which may be output on the console typewriter during the generation run of REPORT are described below.

Message	*Reason*	*Action*
PARAMETER SET INVALID	Errors (which will be flagged) have occurred in the parameter set.	Abandon the run unless the error is merely one of sequence, in which case compilation should not be affected.
PARAMETER SET INCOMPLETE	A #END parameter has been read before all the compulsory parameter groups have been read.	Load the remaining parameters and GO #X68G.
TABLES FULL		Make more core available and GO#X68G.
AB	The first parameter read after the #END parameter was not a #PROGRAM directive with a segment name.	Load own coding segments and GO #X68G.
ER	A tape error has occurred for which there is no recovery.	Abandon the run
LP CR TR		Make the appropriate peripheral available, and GO #X68G, the program will then allocate the peripheral and continue

Object program run

The possible exception condition error messages which may be output on the console typewriter during the object program run of REPORT are described below.

Message	*Reason*	*Action*
EA EB EC	Tape error on File A, B or C for which there is no recovery.	Abandon run.
EW	Tape error on output tape for which there is no recovery.	Abandon run
IΘ ERR *nn* (*nn* is the error type)	Data submitted is imperfect.	GO #X68G to continue or abandon run.
EP	Error in run-time parameter	Correct the faulty parameter, which will be the most recent parameter to have been read. Reload all run-time parameters and GO #X68G from the initial entry point.
LP TR CR		Make the appropriate peripheral available, and GO #X68G; the program will then allocate the peripheral and continue.

PART 2 REPRINT

Chapter 7 Introduction to REPRINT

REPRINT is a program which presents information in the form of a printout from a print file of a standard form stored on magnetic tape. This print tape can be prepared by using an option of the REPORT program (page 13). The advantages of using REPRINT are that the printing can be carried out offline, thus improving multi-programming possibilities, and that repeat and restart facilities are better than these of the less specialised REPORT program. Most of the format of the printout produced is specified by the contents of the print file itself, but the REPRINT parameter set enables the user to control, during the print run:

1 The maximum length of line to be printed

2 The maximum number of lines to be used on a page

3 The number of copies of the printout to be made

4 How many records from the print file are to be printed

The input to REPRINT consists of the print file and a set of parameters either in cards or in paper tape.

MINIMUM CONFIGURATION

The minimum computer configuration required for REPRINT is

1 1900 Series central processor with 8K words of core store

1 card or paper tape reader

1 line printer

1 magnetic tape deck

Chapter 8 File input and output

INPUT

The input to REPRINT consists of

1 The standard print file, consisting of one or more magnetic tape reels

2 The set of REPRINT parameters punched either in cards or in paper tape

These and the order in which they must be input are fully described in Chapter 9.

Print file

Any print file in the standard format specified below may be input to REPRINT, whether the file has been prepared by the REPORT-generated program or by any other appropriate program. Multi-reel files can be dealt with by REPRINT. Each record on this file is in effect a print line ready for output on the line printer. Each line includes a paper feed control character and a print selection word. The record size is 43 words and the block size 512 words. Label, reel sequence number, and file generation number are specified by the program which creates the print file.

The print selection word acts as a 24 bit switch word. The #COPY parameter (page 48) is used to indicate the particular bit of this word which is to be tested in a given printing run and so decide whether the record is to be printed: the record is printed only if the bit is set. This enables up to 24 different printouts to be stored on a single tape.

The format of each record on the file is as follows

Words	*Contents*
0	Word count
1	Paper feed control character in position 1.3
2 to 41	The edited print line of up to 160 print positions
42	Print selection word

OUTPUT

Line printer output

Line printer output from REPRINT consists of

1 A list of parameters with any errors flagged

2 The set up lines for the printout

3 The printout as defined by the parameters

4 A count of the number of pages printed

PARAMETER LIST

Each parameter is listed by the program on the line printer as soon as the card has been read. Any parameter which is found to be incorrect is flagged by having the word WRONG printed beside it.

SET UP LINES

Before the main printout, the program prints two lines of 120 characters (unless the length is altered by the #SIZE parameter). These lines consist of 'X's except for the two central characters, which will be asterisks. The program is then suspended for the operator to check alignment of the paper.

PRINTOUT

When the operator restarts the run after checking the set up lines, the contents of the print file are output on the

line printer.

The #COPY parameter can be used to test for switch settings in Word 42 of each record on the file to decide whether it is to be printed.

The #COPY parameter may also be used to indicate that more than one copy of a printout is required. In this case, either each page is repeated as many times as required before the next page is begun, or the whole printout may be completed and then repeated as required.

It is also possible for the user to specify at what page printing should begin and should end.

COUNT OF PAGES
The program prints

1 Number of records read

2 Number of pages read

3 Last page number

The count of pages at the end is of pages read: it ignores extra pages specified in the #COPY parameter and any generated by restarts. Page counting is carried out by the detection of control characters requesting throws to channel one.

If a further printout from the same file is required, the operator can GO 22. This will produce one further printout from the file, using the same parameter set.

If a further printout from the file using a different parameter set is required, the operator can restart the program (page 53). The program will then read new parameters on the same medium as the original set. All details from earlier sets will be deleted, so the new parameter set must be complete. No new #READ parameter is necessary however, since the input file is the same.

Console typewriter output
Output from REPRINT to the console typewriter consists of error and diagnostic messages: these are described in Chapter 11.

Chapter 9 The parameter set

The parameter input to REPRINT enables the user to set up the format for the printout to be obtained from the print file.

Parameters may be input to REPRINT either on cards or on paper tape. In this chapter they are described in terms of cards: the paper tape format is identical except that each parameter is terminated by a newline character, and any parameter punched on tape must have at least 28 characters, including trailing spaces (*not* zeroes) before the newline.

The parameters are given below in the order in which they must be input. Optional parameters may be omitted, but the order is mandatory.

The main parameters needed for REPRINT are as follows:

1 The #READ parameter handles the magnetic tape file

2 The line and page size parameter (#SIZE) defines the maximum line and page sizes to be printed

3 The printout control parameter (#COPY) enables the number of copies required to be specified, and indicates how much of the input tape is to be printed.

4 The #END parameter terminates the pack of parameter cards

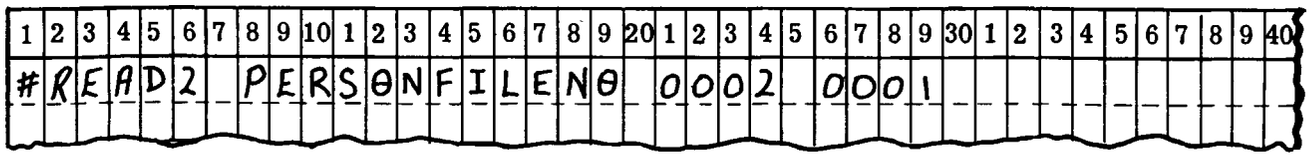
The layout of the parameters is as follows:

READ TAPE LABEL PARAMETER (#READ)

This parameter is in standard form. Its function is to define the label on the input tape file, and the mode in which it is to be opened.

Field	*Columns*	*Contents*
A	1 to 5	The directive #READ
	6	Opening mode: 1 character as follows: 1 = no checks for presence or absence of write permit ring. 2 = check for absence of write permit ring. 3 = check for presence of write permit ring. ∇ (space) in this position is interpreted as 2. Note: These are equivalent to the standard modes of opening tapes that is #100, #200 and #300. Details of additional checks by Executive on the 1904 and above will be found in the ICL manual *Magnetic Tape.*
B	8 to 19	12 character file name.
C	21 to 24	Reel sequence number: four digits
D	26 to 29	File generation number: four digits
E	31	Designation: not used by this program.

Example

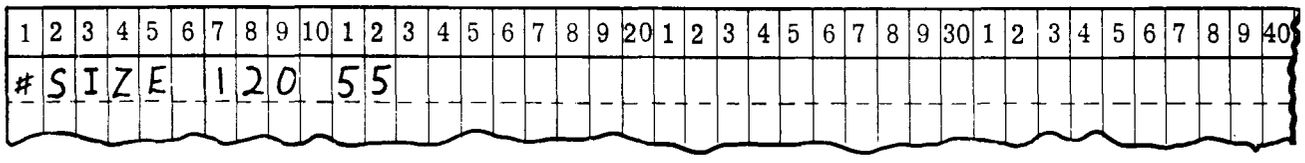


LINE AND PAGE SIZE PARAMETER (#SIZE)

This parameter, which is optional, is used to specify maximum sizes for print lines and pages. If it is omitted, a maximum line of 120 characters will be specified. When the maximum number of lines specified by #SIZE has been printed, the program automatically starts a new page.

Field	*Columns*	*Contents*
A	1 to 5	The directive #SIZE
B	7 to 9	Maximum line size (96, 120 or 160): three digits. If this field is omitted, 120 is assumed
C	11 to 12	Maximum number of lines per page: two digits. If the field is omitted, 60 is assumed.

Example

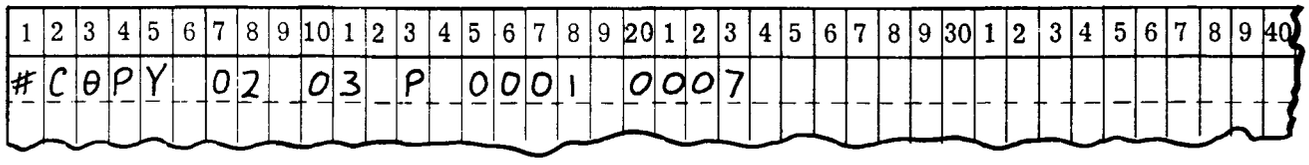


PRINTOUT CONTROL PARAMETER (#COPY)

This parameter, which is optional, enables the user to specify the number of copies of the printout required, and whether all or part of the output tape is to be printed. If this parameter is not present, the program will print every record on the file once only.

Field	*Columns*	*Contents*
A	1 to 5	The directive #COPY
B	7 to 8	Bit position: this field specifies a bit position (00 to 23), in the print selection word, and so identifies the records to be printed.
C	10 to 11	Number of copies to be made: two digits
D	13	P : print as many copies of page 1 as required, then go to page 2, etc. T : print one copy of the whole printout, then a second (if required), etc.
E	15 to 18	Number of first page to be printed: four digits. First page of printout is 0001.
F	20 to 23	Number of last page to be printed: four digits. Alternatively ENDV to continue to the end of the file.

Example



If fields C to F are left blank, each record with the required record selection bit is printed once only.

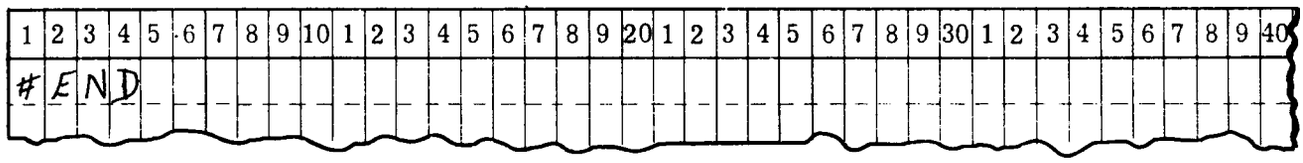
If C and D are punched but E and F left blank, every page is printed.

END OF PARAMETERS MARKER (#END)

The end of parameters is indicated by a card punched as follows:

Field (state)	*Columns*	*Contents*
A	1 to 4	The directive #END

Example



```
REPRINT                                  03/07/68

#READ2 PERSPRINTOUT 0000 0002

#COPY 02 02 P 0003 0005

#END
```

```
                                STAFF PAYROLL ADVICE    03/03/68

  100007
  MR   J.G. MONCRIFFE

  THE GLEN 25 WEST PARK WIMBLEDON

                          PRESENT          CHANGE          EFFECTIVE DATE

BUILDING CODE             BHS1
PAY CENTRE                   1AB
LOCATION CODE             1000006
WEEKLY HOURS                 3700

                           ANNUAL      PERIOD      WEEKLY

PRESENT SALARY          2600  0 0    200  0 0
SALARY CHANGE
NEW SALARY

PRESENT ALLOWANCE       1300  0 0    100  0 0
ALLOWANCE CHANGE
NEW ALLOWANCE

PRESENT TOTAL           3900  0 0    300  0 0
NEW TOTAL

BONUS

ENGAGEMENT

  DATE OF JOINING COMPANY            26 10 66
  CALCULATED DATE OF JOINING         26 10 65
  DATE OF BIRTH                      18 10 42
  NATIONAL INSURANCE NUMBER          A00000009

TERMINATION
  DATE OF LEAVING COMPANY
  TERMINAL REASON

NOTES                               SIGNED
```

```
                                STAFF ADVICE FORM  03/03/68

  100007
  MR   J.G. MONCRIFFE

  THE GLEN 25 WEST PARK WIM-
  BLEDON

FLOOR          23  ROOM          23
INT. TEL.   023  EXT. TEL.   459
BUILDING CODE              BHS1
PAY CENTRE                    1AB
LOCATION CODE              1000006
WEEKLY HRS 3700  EXPENSE SCALE C
LEAVE SCALE   3  LEAVE DAYS    20
STANDARD SALARY SCALE          23A

PRESENT SALARY          2600  0 0
SALARY CHANGE
NEW SALARY

PRESENT ALLOWANCE       1300  0 0
ALLOWANCE CHANGE
NEW ALLOWANCE

PRESENT TOTAL           3900  0 0
NEW TOTAL

BONUS

SALARY EFFECTIVE DATE
ALLCE EFFECTIVE DATE
BONUS EFFECTIVE DATE

JOB TITLE LIASON OFFICER
JOB GRADE XAYB

#ANY QUERIES SHOULD BE REFERRED TO
 YOUR MANAGER OR PERSONNEL OFFICER

SIGNED
```

```
REPRINT                                  03/07/68

RECORDS ON INPUT FILE       533

PAGES PRINTED          3

LAST PAGE NO.          6
```

Figure 7 Reprint printout example

Chapter 10 Example

This chapter includes a run of REPRINT in which the program has been used to make two copies of certain records from a file containing details of personnel. The rest of the chapter consists of detailed comments on the printout.

PARAMETERS

The first part of the printout is a listing of the parameters.

The #READ parameter gives the name of the input file, PERSPRINTOUT, the file generation number (0000) and the reel sequence number (0002)

There is no #SIZE parameter, so the maximum line size is assumed to be 120 characters and the maximum page depth 60 lines.

The #COPY parameter specifies bit position 02 in the print selection word to identify the records to be printed and specifies that two copies are to be made. The "P" in Column 13 indicates that as many copies as are needed of the first page should be printed before going on to the second page. This parameter also specifies that pages 0003 to 0005 are to be printed.

The #END card indicates the end of the parameter set.

SET UP LINES

These lines of 120 characters are printed to enable the operator to align the paper correctly: the four asterisks should appear in the four quadrants of the cross on pre-printed stationery.

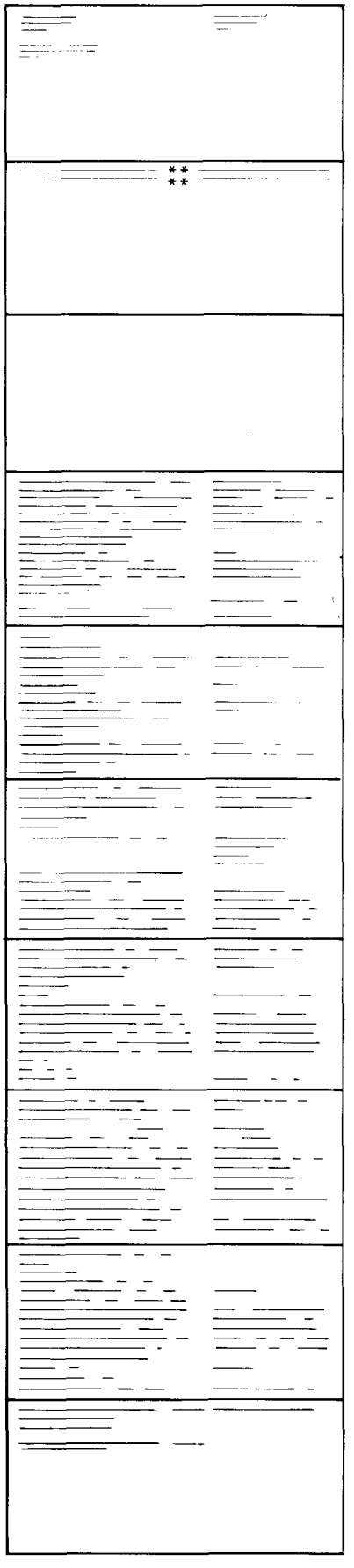
PRINTOUT DETAILS

The program now proceeds to print two copies of the requested pages in the format determined for the print tape.

TOTALS

The program prints out the number of records read from the input file (533), the number of pages read (3) and the number of the last page (6).

REPRINT print out example – Overall layout of a print out produced with this set of parameters.



PARAMETER -- LIST

CHECK-LINES PAGE

BLANK PAGE
This page is omitted, so that the operator can see that the check lines are correctly positioned before main printing starts

PAGE 3, 1st COPY
Note that the copy parameter states that printing is to start from Page 3; Pages 1 and 2 are therefore omitted

PAGE 3, 2nd COPY

PAGE 4, 1st COPY

PAGE 4, 2nd COPY

PAGE 5, 1st COPY

PAGE 5, 2nd COPY

RECORD-COUNTS PAGE

Figure 8 Reprint printout example

Chapter 11 Operating instructions and exception conditions

OPERATING INSTRUCTIONS

The normal procedure necessary for a REPRINT run is as follows:

Narrative	*Console message*
1 To load the program REPRINT input:	FI#X68H
2 Load the magnetic tape containing the input file without a write permit ring	
3 Load the parameters on a card or paper tape reader	
4 To read in the parameters either (a) if the parameters are on paper tape input:	GO #X68H 20
or	
(b) if the parameters are on cards input:	GO #X68H 21
The program reads and lists the parameters, checks the size of printer, prints the check lines and outputs the message:	HALTED:- CH
5 (a) If the paper alignment is correct input:	GO #X68H
(b) If the paper is incorrectly aligned adjust the page and input:	GO #X68H 23
6 The program reads the input tape, prints the required number of copies, and outputs the message:	HALTED:- END OF PROGRAM
7. To read more parameters either (a) if the parameters are on paper tape input:	GO #X68H 20
or	
(b) if the parameters are on cards input:	GO #X68H 21
The parameter set need not include a #READ parameter if the same tape is to be used, but must be terminated by a #END parameter	
8 To obtain another copy of the printout with the existing parameters, input:	GO #X68H 22

Further operating facilities

The operator may make use of the following facilities of REPRINT.

Narrative	*Console message*
1 After a paper wreck, to repeat the last few pages, reload and re-align the paper, and input:	GO #X68H 25
The program performs the checks as in step 4 above and outputs the message:	HALTED:- CH
Restart the program by inputting either: or:	GO #X68H GO #X68H 23
as appropriate. The program repositions the tape and continues printing, reprinting the last six pages	
2 To force a halt at the foot of the current page so that more paper can be loaded, suspend the program, set switch 11 on and input:	GO #X68H

Narrative	*Console message*
The program halts at the foot of the page and outputs the message:	HALTED:- PAPER LOW
3 To close the run early, input:	GO #X68H 26
This closes the input tape, prints the counts and restart information and outputs the message:	HALTED:- RUN CLOSED EARLY

EXCEPTION CONDITIONS

The possible exception condition messages which may be output on the console typewriter in the REPRINT run are described below.

Message	*Reason*	*Action*
INPUT TAPE EXCEPTION CONDITION *nn*:	See the manual *Magnetic Tape* for details of the exception condition	Abandon the run
LAST COMPLETE PAGE *xxxx*	Magnetic tape failure: an additional message gives the number of the last page successfully printed.	Delete REPRINT
INCORRECT PARAMETERS		Correct the parameters and GO #X68H 20 or 21
LOAD INPUT FILE RSN *nnnn*	Only occurs if more than one T-type copy is required, and if input print file is multi-reel	Load the specified reel and GO #X68H
PAPER LOW		Load more paper. The program then re-enters the set-up routine. When this is completed and the operator restarts, processing continues from the point at which it stopped.
NEEDS PRINTER WITH *nnn* POSITIONS		Make a suitable printer available and GO #X68H
LONG BLOCK	Block on magnetic tape file longer than 512 words.	Delete REPRINT
OVERFLOW LAST PAGE *nnnn*	Page overflow. This occurs if a punching is sensed in channel 8 of the printer control loop before the input data has called for a page change. This implies that the input print file is wrongly formed or that the wrong printer control loop is fitted.	Restart with GO #X68H: a paper throw will be forced on the next print instruction but another page modifications will not be made. *Or* Change the paper tape loop to one with no punching in track 8 and GO #X68H.
TRANSFER FAILURE LAST PAGE *nnnn*	Transfer failure on the line printer.	GO #X68H to enable another attempt at outputting the line to be made. If three repetitions, delete REPRINT.
LP TR CR	Peripheral needed.	Put peripheral on line and GO #X68H

Amendment record

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

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