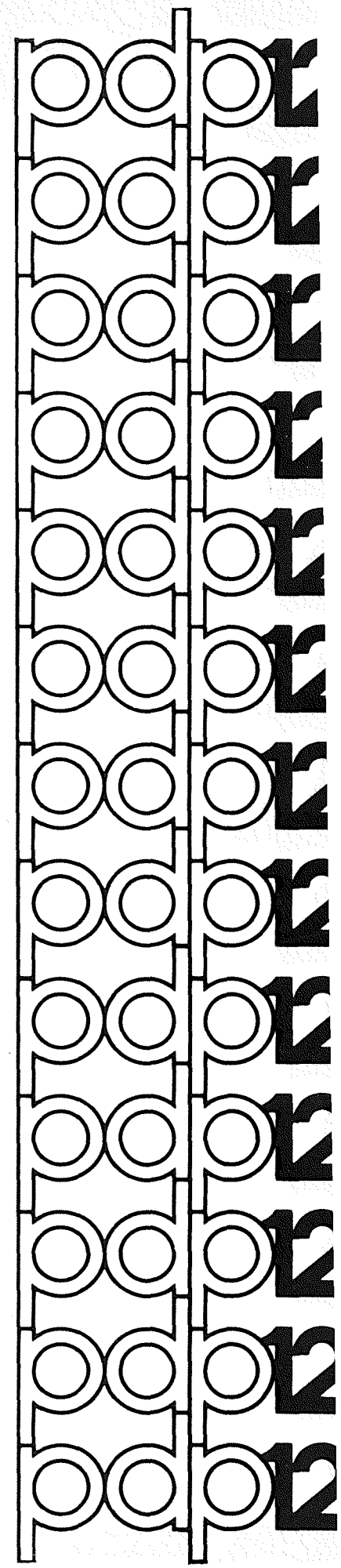
digital

MILDRED



DEC-12-FZDA-D
1st Printing May 1970

The material in this handbook, including but not limited to instruction times and operating speeds, is for information purposes and is subject to change without notice.

The following are trademarks of Digital Equipment Corporation, Maynard, Massachusetts:

DEC	PDP
FLIP CHIP	FOCAL
DIGITAL	COMPUTERLAB

For additional copies, order DEC-12-FZDA-D from Digital Equipment Corporation, Program Library, Maynard, Massachusetts, 01754. Price $1.00

TABLE OF CONTENTS

1.0 PROGRAM OVERVIEW

MILDRED (Multiple Index Lookup, Deletion, Replacement and Entry: Disk) is a set of PDP-12 subroutines for manipulation of LAP6-DIAL[1]/DIAL-MS indexes.

There are three levels of routines, with provision for a routine at any but the lowest level to call any routine of lower level. There is no provision for reentrance or recursion, but the routines are serially reusable (except for REPLACE, as explained later).

Locations 2Ø to 27 contain DJR, JMP pairs to the entry points of each major routine, so that the coding can be modified without changing calls in external routines.

2.0 ENVIRONMENT

MILDRED occupies two tape blocks and, when in core, uses four LINC memory blocks ($2ØØØ_8$ words), including space for the index. The routines are segment-independent, but must be loaded at a segment boundary. Thus memory addresses of 2ØØØ, 4ØØØ, 14ØØØ, etc., can be used, but 24ØØ or 3ØØØ may not. MILDRED requires that the DIAL-MS I/O routines (blocks 322 and 323 of DIAL) reside in field 1, 76ØØ-7777. Because the disk data break locations (775Ø-7751) are in segment 3, MILDRED must not be used in segment 3 (i.e. field Ø, 6ØØØ-7777) when attempting to do disk operations with DF32 or RSØ8 disks. In this discussion, all locations are relative to the segment into which MILDRED is loaded.

3.0 USAGE

3.1 The user's program must load MILDRED from a DIAL tape, add it to his program with Add Binary, or assemble it with his program, at any memory address which is a multiple of $2ØØØ_8$. It may then be reused until it is overlaid. He must also load the DIAL-MS I/O routines from tape Ø before using MILDRED. The DIAL-MS I/O routines may be loaded as follows:

```
LDF     7
RDC
6/322
RDC
7/323
```

3.2 Entry points for the routines of MILDRED start at location 2Ø of the segment into which MILDRED IS loaded, as follows:

2Ø - LOOKUP
22 - ENTER
24 - REPLACE
26 - DELETE

[1] LAP6-DIAL is hereafter referred to as DIAL.

3.3 LOOKUP, ENTER, and DELETE are called as follows:

```
        LIF X               /SEGMENT WITH MILDRED
        LDA I               /AC:  POINTER TO FILE DESCRIPTOR VECTOR
        FDV                 /GO TO LOOKUP (ENTER, DELETE)
        JMP 2Ø (JMP 22, 26)
         .
         .
         .
         .
         .
FDV,    UNIT                /UNIT Ø-17 (SAME AS DIAL-MS UNITS)
        TEXT "NAME????"     /FILE NAME, ENDING WITH 77'S
                            /TO FILL FOUR WORDS (8 CHARS)

        TYPE                /ØØ23 FOR SOURCE, ØØØ2 FOR BINARY

        START               /STARTING BLOCK NO. OF FILE:
                            /FILLED BY LOOKUP, ENTER, REPLACE,
                            /OR DELETE

        LEN                 /LENGTH OF FILE IN BLOCKS: FILLED IN
                            /BY LOOKUP, CALLER MUST SUPPLY IN
                            /ENTER-REPLACE, UNUSED BY DELETE
```

a) LOOKUP has two returns; the first, immediately following JMP 2Ø, is taken if there is an error in the parameter list, or the named file is not found. The second, two words after JMP 2Ø, is taken if the file is found, indicating that the information in the file descriptor vector is correct.

```
        LIF X
        LDA I
        JMP LOOKUP          /GO FIND THE FILE
        JMP NOFIND          /1ST RETURN FILE DOESN'T EXIST
                            /COME HERE WHEN FILE IS FOUND
```

b) ENTER has three returns; the first is taken if there already exists a file of the same name and type. The second is taken on errors in parameter list or insufficient space, either in file space or in the index. The third indicates successful updating of the index.

```
        LIF X               /SEGMENT CONTAINING MILDRED
        LDA I               /POINTER TO PARAMETER LIST
        FDV
        JMP ENTER           /GO ENTER FILE IN INDEX
        JMP EXISTS          /1ST RETURN - FILE ALREADY EXISTS
        JMP NOSPACE         /2ND RETURN - NO SPACE FOR FILE
                            /COME HERE ON SUCCESSFUL COMPLETION
```

Note that the largest file which can ever be stored on a DIAL tape is 31Ø blocks, because that is the length of the largest file area.

c) DELETE has only one return, immediately following the JMP 26.

3.4 REPLACE may be called only immediately after a call to ENTER which took the second return. The parameter list need not be explicitly indicated - REPLACE uses that from the preceding ENTER. But the instruction field must be set again.

There are two returns; the first is taken on error in calling sequence or insufficient space. (This can never occur if the new file is smaller than or equal to the old file). The second indicates successful replacing of the old file entry.

```
LIF X              /SEGMENT CONTAINING MILDRED
JMP REPLAC         /ENTER FOUND A FILE OF SAME NAME
JMP NOSPAC         /NO SPACE FOR NEW ONE
                   /COME HERE ON SUCCESSFUL REPLACE
```

If REPLACE is not able to find space for a new file, the old file remains intact.

If the call to REPLACE is not immediately preceded by a call to ENTER which returns indicating the file exists, the machine will halt and MILDRED must be reloaded.

4.0 DESCRIPTION OF THE ROUTINES:

4.1 LOOKUP: (Level 2, entry point 2Ø) RET3 (Beta 15) is set to 7777 to indicate external call. Internal calls enter immediately following this point, at LKPØØØ. Here, the return JMP is saved at RET2 (Beta 16). The current instruction field is obtained and used to set the address into which the index will be read. The caller's fields are saved and used to set the data field for MILDRED to enable examination of the user's parameter list. Parameters are set up for READ[1], which is then called to bring in the index. Upon return, the index is checked for validity (5757 in first word). If invalid, an empty index is built in core by storing 5757 in each word (from 1ØØØ to 1777), and the error return is taken.

At LKPØ2Ø, the name in the user's parameter list is compared with each name in the index until a match is found or the end of the index is reached. In the latter event, the error return is taken. If a match is found, the type code in the user's parameter list is compared to 23 (S) and Ø2 (B). If neither, the error return is taken. S causes a jump

[1]DIAL-MS READ routines.

to WNTS; B jumps to WNTB, which increments the pointer to the index entry by two and flows into WNTS.

Here RET3 is tested for internal or external call; if internal, RET2, the return JMP, is incremented to allow the caller to distinguish between those cases in which there was a successful name match, but no file of the requested type, and those cases in which the name match was unsuccessful. The starting block number of the file is then moved to the user's parameter list. The length is then picked up and tested to see whether or not there is a file of the requested type. If not, the length will be 5757, and the error return will be taken. If the requested file exists, RET3 is tested to check for external call. If external, the length is stored. The return address is then incremented to indicate a successful find, and LOOKUP jumps to ERRTN. There, RET3 is tested again. If the call is internal, return is immediate. If external, LOOKUP restores the user's fields before returning.

4.2 ENTER (Level 3, Entry Point 22) Starting at NTRØØØ, the return JMP is saved in RET3 (Beta 15). LOOKUP is called at its internal entry point LKPØØØ. Because this is an internal call, there are three returns. The first, indicating that there was no name match, jumps to NTRØ2Ø, where MARK (Beta 1Ø) is set to 7777 to indicate no name match, and flows to NTRØ3Ø, which increments the return address, there being no conflict with existing files. From there, control flows in to FSPØØØ to find space for the file. Subsequent processing is in common with REPLACE, and is described below. The second return indicates that the name was found, but not with the requested file type. A pointer to the matching entry is saved at MARK, and ENTER jumps to NTRØ3Ø to increment the return JMP. The third return from LOOKUP indicates that the named file exists. A pointer to its index entry is saved at MARK, ENTSW (Beta 11) is set to 1776 to allow a REPLACE to follow, and the first return is taken by going to RTRNØ.

4.3 REPLACE: (Level 3, Entry Point 24) starting at RPLØØØ, the return JMP is saved at RET3 (Beta 15), and ENTSW (Beta 11) is tested for 1776 (indicating that ENTER found a file conflict). Any other value indicates a user error, and the program halts. The LDF instruction at GTFØ1Ø is then moved into the instruction stream to again setthe data field appropriately for the user's parameter list, the length field for this file in the index is set to 5757 to eliminate the old file, and REPLACE jumps to FSPØØØ to find space for the file. From this point, processing is in common with ENTER.

The search for file space is performed in two steps; first, a scan is made to find any suitable space in the lower file area. The result, if any, is saved; then a scan of the upper area is made. Because the index is below the middle of the tape, the result of a successful scan of the lower file area can be used to calculate an upper limit for scanning in the upper file area. Beyond this limit, any suitable file space would not be used, since the suitable space in the lower area is closer. Conversely, any space found in the upper area before reaching this limit must be closer to the index than the space found in the lower area. Use of this algorithm eliminates, therefore, the need to compare two possible spaces for closeness to the index, and generally shortens the scan of the upper file area.

Scanning itself is performed by the conflict-search routine, FCT, as follows:

A tentative starting block (TRY) and the length of the desired file (TRYLEN) are set up. Each non-empty index entry is compared to TRY by subtracting its starting block from TRY. If the result is negative, the file specified by this index entry starts at or above TRY. If adding TRYLEN still gives a negative result, the file starts above the end of the tentative file, and there is no conflict. The scan continues to the next index entry.

If there is a conflict, control is returned to the caller to set a new TRY.

If subtracting the starting block from TRY yields a positive result, TRY is above the file specified by this entry, by the value of the result. In other words, AC contains the distance between TRY and the file concerned. This distance is subtracted from the length of the file in question. Here, a negative result implies no conflict, and the scan continues with the next index entry. A positive result represents a conflict, and control returns to the caller.

During the scan of the lower area, TRY moves downward (away from the index). Each time a conflict is found, a new TRY is calculated by subtracting TRYLEN from the starting block of the file causing the conflict. This is the highest possible starting block which will not cause a conflict with this particular file. The conflict search routine is called again, and the whole process repeated. Thus, the maximum number of iterations is the number of

files in the lower file area. During the scan of the upper area, the process is equivalent, except that TRY moves upward. When a conflict is found, a new TRY is calculated as the sum of the start and length of the file in conflict.

In detail, the search is performed as follows:

A SKIP-IF-NEGATIVE instruction (APO I) is moved to FCFØ5Ø in the conflict search routine to make it ignore any files in the upper file area. A pointer to the start block field of the user's parameter list is saved at LP1 (Beta 8). The user's length request is picked up, tested for validity (zero or negative lengths cause a jump to RTRNØ, indicating error), and saved at TRYLEN. The length is then subtracted from 27Ø, to give the block number of highest starting block in the lower file area which could satisfy the request.

This value is in the AC at FSPØ1Ø, the beginning of the search loop for the lower file area. It is tested to assure that this starting block is positive (i.e., that it is on the tape). If not, there is no space large enough for the file in the lower area, and a jump FSPØ2Ø is taken. If the starting block is positive, it is stored at TRY, and the conflict search routine, FCF, is called. Return is to a JMP FCFØ3Ø if no file in the index would overlap one which started at the block number in TRY. If a conflict is found, the second return is taken from FCF; in this case, TRYLEN is subtracted from the starting block number of the file which conflicts with TRY. This value is in the AC when FSP jumps to FSPØ1Ø to make another search.

The code at FSPØ2Ø is entered, as described above, when TRY goes negative before an acceptable space is found indicating that there is insufficient continuous file space in the lower area for a file the size of TRYLEN. NFSW (Beta 12) is set to zero to indicate this. The last block number on the tape is subtracted from TRYLEN and stored at UPLIM so that UPLIM contains the complement of the highest possible starting block which would permit a file of the desired size to fit on the tape. FSP then jumps to FSPØ35 to scan the upper file area.

The code at FSPØ3Ø is entered when FCF is unable to find a file which overlaps with one starting at TRY, thus TRY contains the starting block number of a space in the lower file area large enough to accommodate the desired file. NFSW is set to 7777, indicating space was found in the lower area. TRY is saved at SVTRY. The distance between this space and the

index is [346 - (TRY + TRYLEN - 1)]. The block as far from the index in the upper file area is this number +347. The complement of the latter result is calculated and stored at UPLIM. Thus the search of the upper file area can be stopped and considered unsuccessful if no space can be found closer to the index than the space already found in the lower area. At this point, control flows into FSPØ35, and processing is the same whether space was found in the lower area or not.

A SKIP-IF-POSITIVE instruction (APO) is moved to FCFØ5Ø, to cause files in the lower area to be ignored during the conflict search. The AC is initialized to 47Ø, the first block of the upper area, and the upper area scan is begun at FSPØ4Ø. The AC is stored at TRY, then added to UPLIM. If the result is positive, TRY is too large to be useful because it represents the starting block of a file which would run off the end of the tape, or it is farther from the index than the space found in the lower area. A jump is therefore taken to FSPØ5Ø, which tests NFSW for a find in the lower area. If none, there is no space, and an error return is taken via RTRNØ. If NFSW is set, however, the starting block at SVTRY is restored to TRY, and control flows to FSPØ6Ø.

If the sum of TRY and UPLIM was negative or zero (zero result will always be negative), FCF is called to search for a conflict. On finding one, the sum of the starting block and the length of the conflicting file is taken as the next TRY, and FSP jumps to FSPØ4Ø to begin another scan. If no conflict is found, however, before TRY exceeds the absolute value of UPLIM, TRY represents the best starting block for the new file, and FSP jumps to FSPØ6Ø.

At this point, MARK (Beta 1Ø) is tested to determine whether an index entry with the desired name already exists. If so, control transfers to FSP1ØØ. If not, the index is scanned for an entry containing 5757 in the name field, indicating it is empty. If no empty entry is found, an error return is taken via RTPNØ.

If the name is not empty, XPNT is incremented by 4, to address the source-file pointers, and control flows into FCFØ4Ø. FCFØ5Ø having been set to an APO or APO I instruction, the start block is compared to the index TBLK to determine whether the file is in the wrong area for this scan. If it is the wrong area, the loop is re-entered at FCFØ1Ø. This check, it should be noted, is unnecessary, but was included to speed the scan. Thirteen octal words can be saved by its elimination if space becomes tight.

If an empty entry is found, the file name is moved in from the user's parameter list. The user's type specification (S or B) is examined, and the START and LENGTH pointers for the other file type are filled with 5757. Control flows to FSP1ØØ, where the starting block and LENGTH are stored in the index, and the starting block is stored in the user's parameter list.

The write code, parameter pointer, and return jump are setup for rewriting the INDEX. The I/O handler is called via its internal entry point RWENT. The return address is incremented, to indicate successful completion, the user's fields restored, and control is returned to the caller.

4.4 FCF: (Level 2, no external Entry Point). FCF is the conflict search routine. Given a starting block, TRY, and a length, TRYLEN, its task is to scan the index for a file, one or more of whose blocks is in the range from TRY to (TRY+TRYLEN-1). If it should find such a file, pointers to the starting block and length are returned in XPNT and XPNT2, respectively, and control is returned to (P+2), where P is the address of the calling jump. If no conflict is found, control is returned to (P+1).

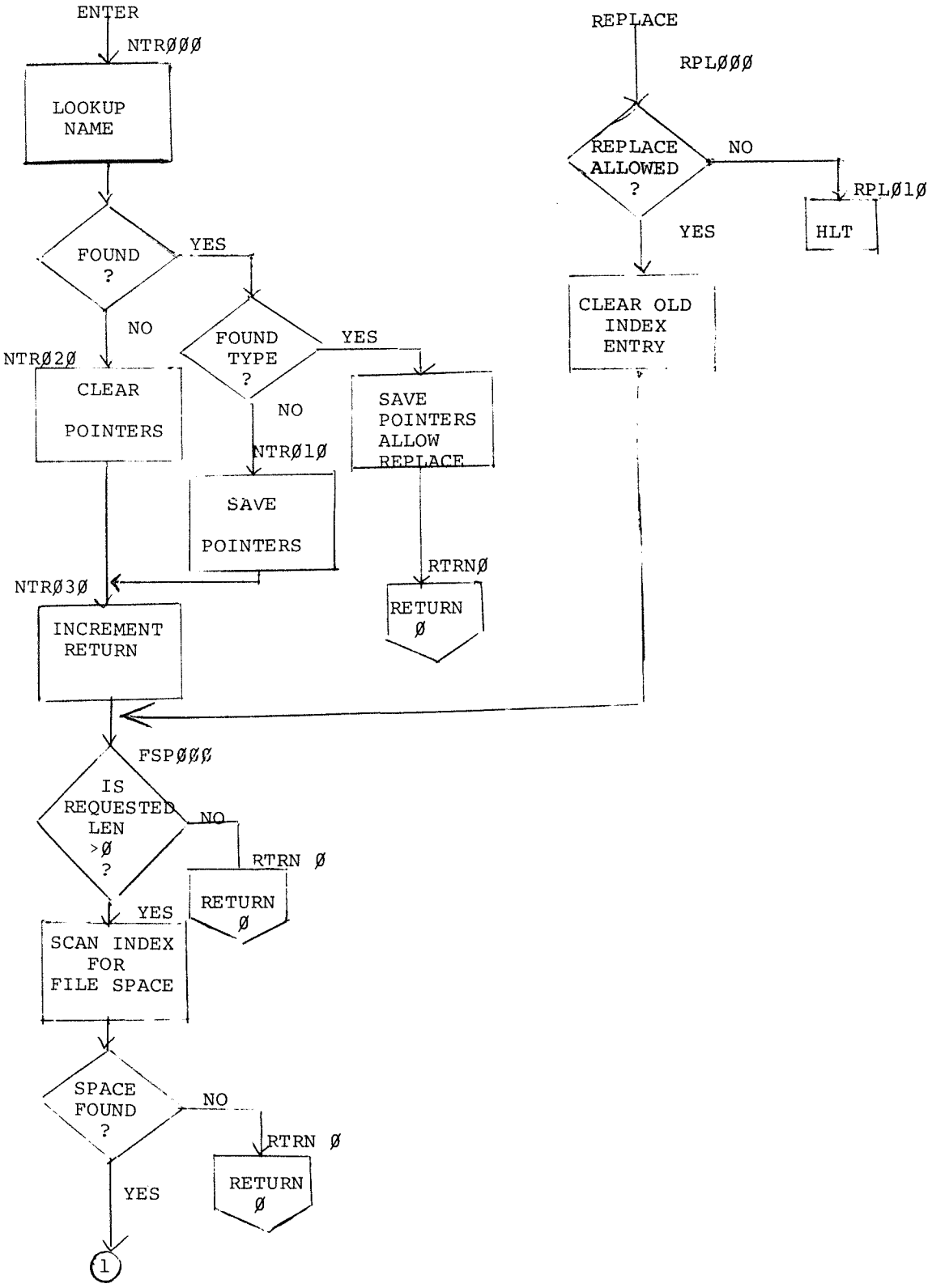
Upon entry at FCFØØØ, the return jump is saved at RET2, and XPNT (Beta 4) is initialized to point to the first index entry. Control flows to FCFØ1Ø, where XPNT is incremented and tested for end-of-index. If the end has been reached, there is no conflict and control returns to the caller via RET2. If the end has not been reached, bit 9 of XPNT is tested to determine whether XPNT is pointing to a file name or the start and len area of the entry. In the latter case, control transfers to FCFØ4Ø. In the former, the name is compared to 5757. If equal, the entry is empty, so XPNT is incremented by 6, and the loop is entered again at FCFØ1Ø.

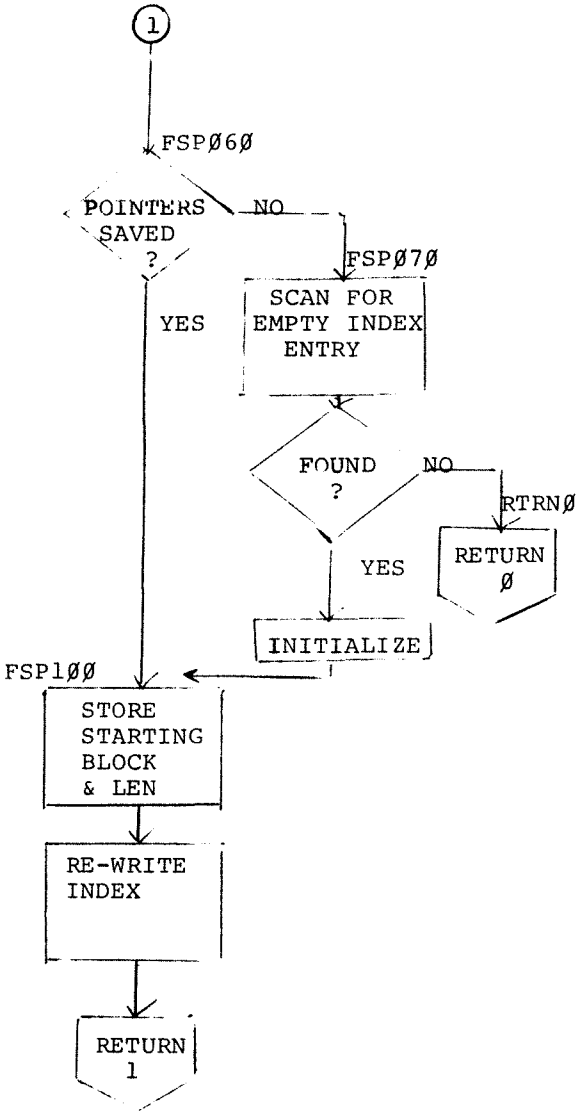
XPNT2 is set to address the length field, which is tested for validity. If negative, there is no file of that type, and the loop is re-entered at FCFØ1Ø. If the length is positive, the starting block is subtracted from TRY. If the result is negative or zero, TRY is below the start of this file by complement of AC, and control transfers to FCFØ6Ø. If the result is positive, TRY is above the start of this file by the contents of AC. Subtracting this value from the file length gives a positive result if there is a conflict, a negative or zero result if none. If there is a conflict, it is returned to the caller via FCFØ7Ø. If not, the scan is resumed at FCFØ1Ø.

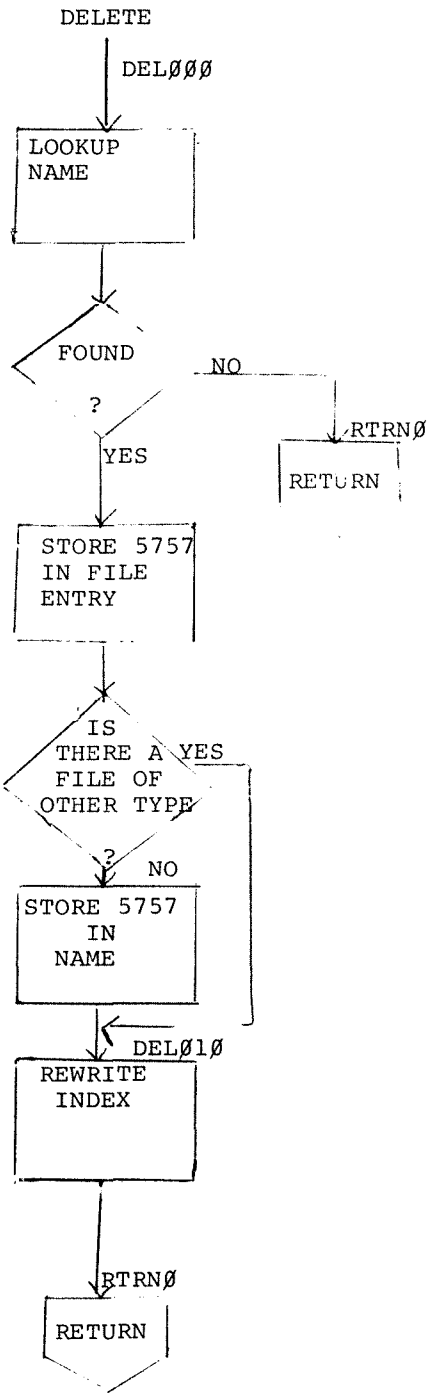
The code at FCFØ6Ø is entered when TRY is below the start of this file. TRYLEN is added to the complement of the block difference. A negative or zero result implies no conflict, and the scan continues at FCFØ1Ø. A positive result is a conflict, so control flows into FCFØ7Ø, which increments RET2 and jumps to it.

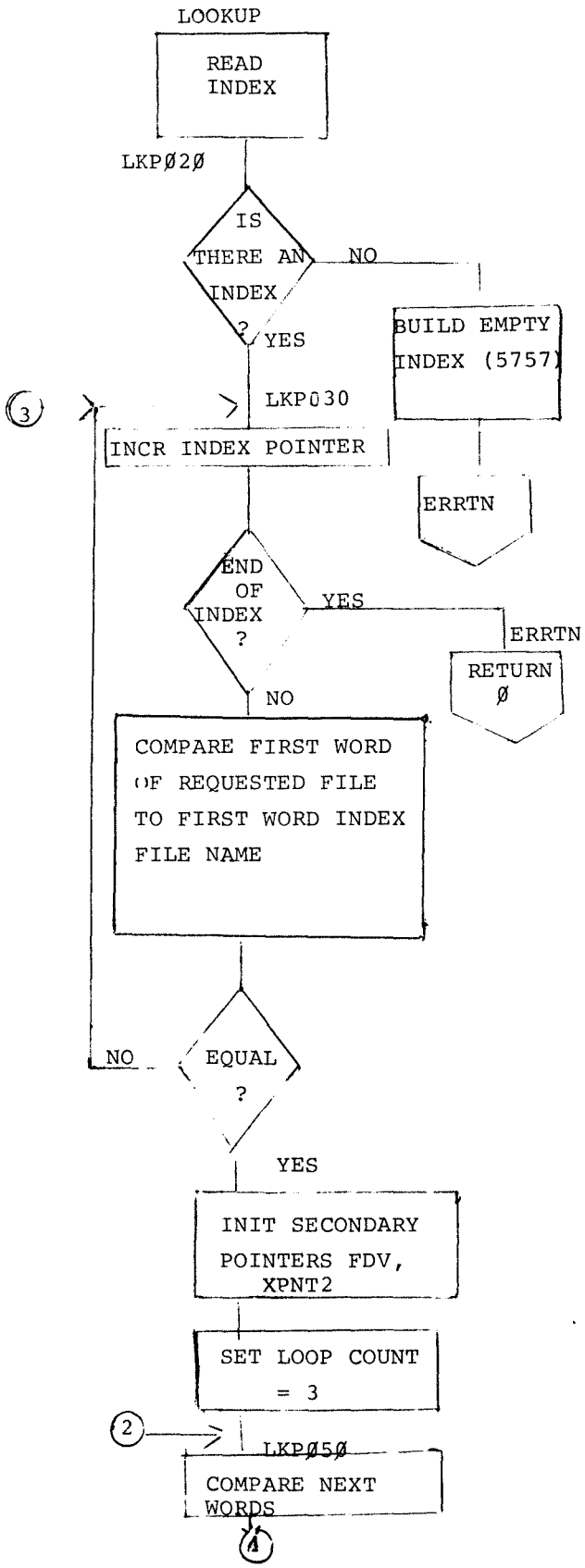
4.8 DELETE: (Level 3, Entry Point 26). Beginning at DELØØØ, the return jump is saved at RET3 (Beta 15). LOOKUP is then called via the internal entry point LKPØØØ to find the name and file to be deleted. On each of the alternate returns, a JMP RTRNØ is taken, since it is unnecessary to delete a file which does not exist. On the third return, the start and length words for this file are filled with 5757 to eliminate the file. Bit 1Ø of the address of the length word is complemented to give the address of the length word for the other file type; that is, if a source file is being deleted, the low order digit of the address of its length word is 5. Complementing bit 1 gives 7, the address of the binary length. The length of the other type file is tested to determine whether such a file exists. If it is positive, a jump is taken to DELØ1Ø. If negative, there is no file of the other type, so the name area of the index entry is set to 5757. At DELØ1Ø, the write code, return jump, and parameter pointers are set up, and the I/O handler is called at RWENT to rewrite the index.

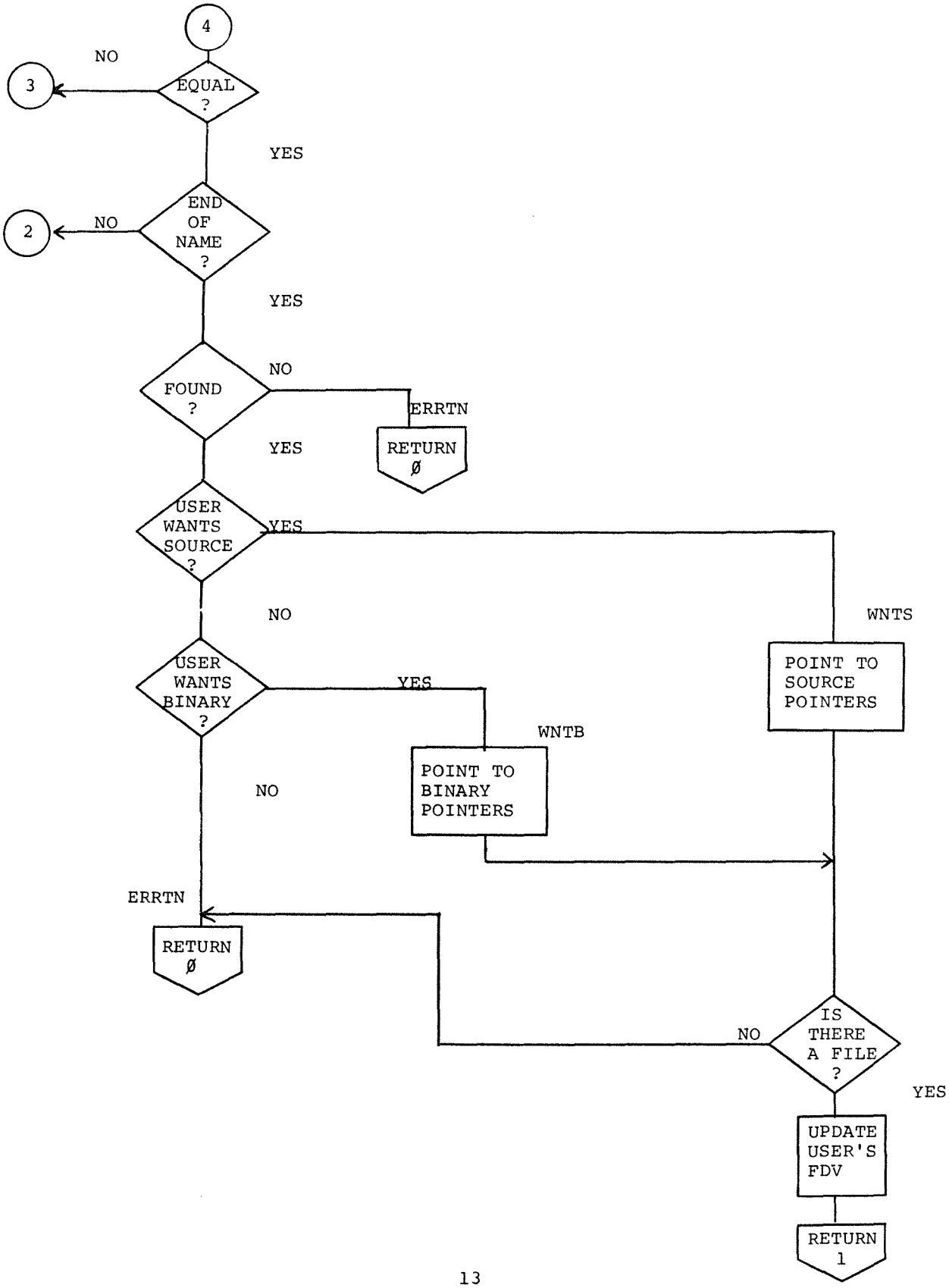
5.0 FLOW CHARTS


ENTER
NTRØØØ
LOOKUP NAME
FOUND ?
YES
NO
FOUND TYPE ?
YES
NO
NTRØ2Ø
CLEAR POINTERS
NTRØ1Ø
SAVE POINTERS
SAVE POINTERS ALLOW REPLACE
RTRNØ
RETURN Ø
NTRØ3Ø
INCREMENT RETURN
FSPØØØ
IS REQUESTED LEN >Ø ?
NO
RTRN Ø
RETURN Ø
YES
SCAN INDEX FOR FILE SPACE
SPACE FOUND ?
NO
RTRN Ø
RETURN Ø
YES
1
REPLACE
RPLØØØ
REPLACE ALLOWED ?
NO
RPLØ1Ø
HLT
YES
CLEAR OLD INDEX ENTRY


1
FSPØ6Ø
POINTERS SAVED ?
NO
YES
FSPØ7Ø
SCAN FOR EMPTY INDEX ENTRY
FOUND ?
NO
RTRNØ
RETURN Ø
YES
INITIALIZE
FSP1ØØ
STORE STARTING BLOCK & LEN
RE-WRITE INDEX
RETURN 1


DELETE
DELØØØ
LOOKUP NAME
FOUND ?
NO
RTRNØ
RETURN
YES
STORE 5757 IN FILE ENTRY
IS THERE A FILE OF OTHER TYPE ?
YES
NO
STORE 5757 IN NAME
DEL01Ø
REWRITE INDEX
RTRNØ
RETURN


LOOKUP
READ INDEX
LKPØ2Ø
IS THERE AN INDEX ?
NO
YES
BUILD EMPTY INDEX (5757)
3
LKPØ3Ø
INCR INDEX POINTER
ERRTN
END OF INDEX ?
YES
ERRTN
RETURN Ø
NO
COMPARE FIRST WORD OF REQUESTED FILE TO FIRST WORD INDEX FILE NAME
NO
EQUAL ?
YES
INIT SECONDARY POINTERS FDV, XPNT2
SET LOOP COUNT = 3
2
LKPØ5Ø
COMPARE NEXT WORDS
4


4
NO
3
EQUAL ?
YES
END OF NAME ?
NO
2
YES
FOUND ?
NO
ERRTN
RETURN Ø
YES
USER WANTS SOURCE ?
YES
WNTS
POINT TO SOURCE POINTERS
NO
USER WANTS BINARY ?
YES
WNTB
POINT TO BINARY POINTERS
NO
ERRTN
RETURN Ø
IS THERE A FILE ?
NO
YES
UPDATE USER'S FDV
RETURN 1

INDEX

```
0000                    *20
0001                    /       MILDRED -- MULTIPLE INDEX LOOKUP, DELETION, REPLACEMENT, AND ENTRY (DISK)
0002                    /           DIRECT DESCENDENT OF:
0003                    /       FRED -- FILE REPLACEMENT, ENTRY, AND DELETION
0004                    /       29 MAY, 1970
0005                    /
0006                    /
0007                    /       BETA REGISTER DEFINITIONS
0010                    /               (3, 13, 14, AND 17 ARE UNUSED)
0011                    /
0012                            PARAM=1
0013                            FDV=2
0014                            XPNT=4
0015                            XPNT2=5
0016                            LP1=6
0017                            LP2=7
0020                            MARK=10
0021                            ENTSW=11
0022                            NFSW=12
0023                            RET3=15
0024                            RET2=16
0025                    /
0026                    /       ENTRY POINTS OF MS I/O HANDLERS
0027                    /
0030                            READ=7774
0031                            WRITE=7775
0032                            RWFLD=10
0033                    /
0034                    /   LOAD ADDRESS OF DIAL INDEX -- DO NOT MODIFY: RAMIFICATIONS ARE OVERWHELMING
0035                    /
0036                            INDEX=1000
0037                    /
0040                    /   PDP-8 MODE INSTRUCTIONS FOR USE AFTER IOB
0041                    /
0042                            LRMF=6244
0043                            LRIB=6234
0044                            LRIF=6224
0045                    /
0046                    /
0047                    /     ENTRY POINTS FOR MAJOR ROUTINES OF FRED
0050                    /
0051                    /
0052    0020  0006              DJR
0053    0021  6230              JMP LKP000
0054    0022  0006              DJR
0055    0023  6265              JMP NTR000
0056    0024  0006              DJR
0057    0025  6252              JMP RPL000
0060    0026  0006              DJR
0061    0027  6604              JMP DEL000
0062                            EJECT
```

```
0063                    /
0064                    /       LOCKUP - FIND NAMED FILE IN DIAL INDEX
0065                    /
0066                    /       ENTERED IN LINC MODE WITH ADDRESS OF A PARAMETER
0067                    /       LIST IN AC.
0070                    /
0071                    /    0/ UNIT NUMBER  /
0072                    /    1/ FILE NAME    /
0073                    /    2/  CONTD       /
0074                    /    3/  CONTD       /
0075                    /    4/  CONTD       /
0076                    /    5/ TYPE (S OR B)/
0077                    /    6/ STARTING BLOCK
0100                    /    7/ NO OF BLOCKS /
0101                    /
0102    0030    0075    LKP000, SET I RET3      / INDICATE EXTERNAL CALL
0103    0031    7777            7777
0104    0032    0056    LKP010, SET RET2        / SAVE RETURN JMP
0105    0033    0000            0
0106                    /
0107                    /       GET CALLERS FIELDS AND PARAMETER LIST
0110                    /
0111    0034    1620            BSE I           / SET BIT 0, LATER BECOMES BIT 1
0112    0035    4000            4000
0113    0036    0262            ROL I 2         / IF-DF BIT TO LINK
0114    0037    0302            ROR 2           / RESTORE REST
0115    0040    1560            BCL I           / CLEAR HALF-WORD BIT
0116    0041    4000            4000
0117    0042    4001            STC PARAM       / HOLD THE REST
0120    0043    4011            STC ENTSW       / CLEAR ENTRY SW
0121    0044    0500            IOB             / DO RIB IOT
0122    0045    6234            LRIB
0123    0046    1060            STA I           / HOLD IT
0124    0047    0000    SAVFLD, 0
0125    0050    0303            ROR 3           / IF TO BITS 7-11
0126    0051    1560            BCL I           / DROP OTHERS
0127    0052    7740            7740
0130    0053    1620            BSE I           / BUILD LIF INST
0131    0054    0600            LIF
0132    0055    4221            STC RESTIF      / FOR RESTORING
0133    0056    2047            ADD SAVFLD      / GET DF BITS IN 9-11, 0-1
0134    0057    0242            ROL 2           / NOW IN 7-11
0135    0060    1560            BCL I           / CLEAR OTHERS
0136    0061    7740            7740
0137    0062    1620            BSE I           / BUILD LDF INST
0140    0063    0640            LDF
0141    0064    4220            STC RESTDF      / SAVE TO RESTORE LATER
0142    0065    0452            LZE             / SKIP IF PARMS IN USER INST FLD
0143    0066    2220            ADD RESTDF
0144    0067    0472            LZE I           / SKIP IF PARMS IN HIS DATA FIELD
0145    0070    2221            ADD RESTIF
0146    0071    1620            BSE I           / MAKE AN LDF INST OF IT
0147    0072    0640            LDF
0150    0073    4241            STC SETUDF      / HOLD FOR ADDRESSING PARAMS
0151                    /
0152                    /       RELOCATE FOR THIS SEGMENT
0153                    /
0154    0074    1020            LDA     I       / INIT AC 11=1
0155    0075    0001            1
0156    0076    0500            IOB
0157    0077    6224            LRIB            / GET INSTRUCTION FIELD IN 6-10
0160    0100    0241            ROL 1           / NOW IN 5-9, 10, 11=0
0161    0101    1040            STA             / HOLD FOR READING
```

```
0162    0102  0247          XPARM+1
0163    0103  0302          ROR 2             / FOR BUILDING LDF
0164    0104  1560          BCL I             / DROP EXTRANEOUS
0165    0105  7740          7740
0166    0106  1620          BSE I
0167    0107  0640          LDF               / BECOMES INSTR FOR SETTING TO THIS SEGMENT
0170    0110  1040          STA               / HOLD FOR LATER USE
0171    0111  0243          SETMDF
0172    0112  1560          BCL I             / GET SEGMENT BITS
0173    0113  7774          7774
0174    0114  0302          ROR 2             / MOVE TO HIGH ORDER
0175    0115  2245          ADD PPOINT        / COMBINE WITH PARAMETER ADDRESS
0176    0116  1040          STA
0177    0117  4132          PPNT1
0200    0120  4504          STC PPNT2         /...FOR INPUT AND OUTPUT
0201                /
0202                /       READ THE INDEX
0203                /
0204    0121  6241          JMP SETUDF        / SET DATA FIELD FOR USERS PARAMS
0205    0122  1001          LDA PARAM         / GET UNIT
0206    0123  1560          BCL I             / ONLY
0207    0124  7760          7760
0210    0125  4246          STC XPARM
0211    0126  6243          JMP SETMDF        / SET DATA FIELD TO US FOR RETURN FROM I/O
0212    0127  0002          PDP               / PMODE FOR READING
0213                        PMODE
0214    4130  6212          CIF RWFLD         / I/O IS IN FIELD 1
0215    4131  4754          JMS I   PREAD
0216    4132  4246  PPNT1,  XPARM             / PARAMETER ADDRESS
0217    4133  6141          LINC
0220                        LMODE
0221    0134  6241          JMP SETUDF        / GET USERS PARAMETER LIST
0222    0135  0064          SET I XPNT
0223    0136  1000          INDEX
0224    0137  1004          LDA XPNT          / GET FIRST WORD OF INDEX
0225    0140  1460          SAE I             / SKIP IF VALID FOR INDEX
0226    0141  5757          5757              / FIRST WORD OF INDEX
0227    0142  0467          SKP               / NOT AN INDEX -- BUILD ONE
0230    0143  6155          JMP LKP020        / INDEX IS OK -- GO TO NAME SCAN
0231                /
0232                /     THERE IS NO INDEX -- CREATE ONE
0233                /
0234    0144  1020            LDA I           / GET FILLER WORD
0235    0145  5757            5757
0236    0146  1044            STA XPNT        / STORE IN NEXT INDEX WORD
0237    0147  0224            XSK I XPNT      / INCREMENT AND TEST FOR END
0240    0150  6146            JMP .-2         / ZAP ANOTHER
0241    0151  0221            XSK I PARAM     / ADJUST PARAM
0242    0152  1044            STA XPNT        / ZAP LAST WORD
0243    0153  6214            JMP ERRTN       / RETURN NO FIND
0244                /
0245                /       ADDR OF READ ROUTINE IN FIELD 1
0246                /
0247    0154  7774  PREAD,  READ
0250                        EJECT
```

```
0251                    /
0252                    /     SCAN INDEX FOR NAME
0253                    /
0254        0155  1021  LKP020, LDA I PARAM      / GET FIRST WORD OF NAME
0255        0156  4170          STC WORD1        / SAVE AT COMPARE INST
0256        0157  1020  LKP030, LDA I            / CONSTANT 7
0257        0160  0007          7
0260        0161  1140          ADM              / ADD IT TO INDEX POINTER
0261        0162  0004          XPNT
0262        0163  0204          XSK XPNT         / TEST FOR END OF INDEX
0263        0164  0467          SKP              / SKIP NOT END
0264        0165  6214          JMP ERRTN        / OTHERWISE RETURN NO SUCCESS
0265        0166  1024          LDA I XPNT       / GET A WORD OF INDEX NAME
0266        0167  1460          SAE I            / SKIP OUT OF LOOP IF FIRST WORDS EQUAL
0267        0170  0000  WORD1,   0
0270        0171  6157          JMP LKP030       / LOOP IF NOT EQUAL
0271        0172  0045          SET XPNT2        / FIRST WORDS EQUAL
0272        0173  0004          XPNT             / START FINAL COMPARE
0273        0174  0042          SET FDV          / POINT TEMPORARY FDV INDEX
0274        0175  0001          PARAM
0275        0176  0067          SET I LP2        / SET LOOP COUNTER
0276        0177  7774          -3
0277        0200  1022  LKP050, LDA I FDV        / GET NEXT TWO CHARS
0300        0201  1465          SAE I XPNT2      / COMPARE TO THOSE IN INDEX ENTRY
0301        0202  6157          JMP LKP030       / UNEQUAL - RETURN TO SEARCH LOOP
0302        0203  0227          XSK I LP2        / LOOP TO COMPARE ENTIRE NAME
0303        0204  6200          JMP LKP050
0304                    /
0305                    /     IF WE GET HERE, WEVE FOUND THE NAME
0306                    /
0307        0205  1022            LDA I FDV      / PICK UP SOURCE/BINARY CODE
0310        0206  1420            SHD I          / IS THE CODE S?
0311        0207  2300            2300
0312        0210  6225            JMP WNTS       / YES - GIVE HIM SOURCE
0313        0211  1420            SHD I          / IS THE CODE B?
0314        0212  0200            0200
0315        0213  6223            JMP WNTB       / YES - GIVE HIM BINARY
0316                    /
0317                    /     COME HERE ON DETECTING ERROR, OR UNSUCCESSFUL FIND
0320                    /
0321        0214  0215  ERRTN,   XSK RET3        / TEST FOR INTERNAL CALL
0322        0215  6016          JMP RET2         / RETURN NOW IF INTERNAL
0323        0216  0040          SET 0            / MOVE IN RETURN JMP
0324        0217  0016          RET2
0325                    RESTOR=.
0326        0220  0640  RESTDF, LDF              / MODIFIED FOR...
0327        0221  0600  RESTIF, LIF              /...USERS FIELDS
0330        0222  6000          JMP 0            / RETURN TO CALLER
0331                            EJECT
-
```

```
0332                        /
0333                        /      WEVE FOUND WHAT HE WANTS - GIVE IT HIM
0334                        /
0335    0223  0225  WNTB,   XSK I XPNT2      / CANT SKIP, BUT THIS ...
0336    0224  0225          XSK I XPNT2      /...FAKES OUT THE POINTER
0337    0225  0215  WNTS,   XSK RET3         / IF INTERNAL CALL...
0340    0226  0236          XSK I RET2       /...SETUP FOR THREE RETURNS
0341    0227  1025          LDA I XPNT2      / GET STARTING BLOCK NO FROM INDEX
0342    0230  1062          STA I FDV        / STORE IN CALLERS PARAMETER LIST
0343    0231  1025          LDA I XPNT2      / GET NO OF BLOCKS
0344    0232  0451          APO              / SKIP IF DESIRED DATA EXISTS
0345    0233  6214          JMP ERRTN        / RETURN ERROR IF NO FILE
0346    0234  0215          XSK RET3         / DONT STORE RESULT IF INTERNAL
0347    0235  0467          SKP
0350    0236  1062          STA I FDV        / RETURN NO OF BLOCKS
0351    0237  0236          XSK I RET2       / INCREMENT JMP RETURN ADDRESS
0352    0240  6214          JMP ERRTN        / NOT REALLY AN ERROR
0353                        /
0354                        /
0355                        /      SETUDF - SET DATA FIELD FOR USERS PARAMETER LIST
0356                        /
0357    0241  0640  SETUDF, LDF              / INSTRUCTION OVERLAID
0360    0242  6000          JMP Ø            / RETURN
0361                        /
0362                        /
0363                        /      SETMDF - SET DATA FIELD FOR MILDRED
0364                        /
0365    0243  0640  SETMDF, LDF              / MODIFIED
0366    0244  6000          JMP Ø
0367                        /
0370                        /
0371    0245  0246  PPOINT, XPARM&777       / LOW-ORDER BITS OF XPARM ADDR
0372                        /
0373                        /    PARAMETERS TO READ ROUTINE
0374                        /
0375    0246  0000  XPARM,   Ø                / UNIT
0376    0247  0000           Ø                / DATA ADDRESS SHIFTED RIGHT 8 BITS
0377    0250  0346           346              / BLOCK NUMBER
0400    0251  0002           2                / BLOCK COUNT
0401                         EJECT
-
```

```
0402                    /
0403                    /       REPLACE -- REPLACE A NAMED ENTRY IN DIAL INDEX
0404                    /
0405                    /       ENTERED IN LINC MODE AFTER CALLING ENTER.
0406                    /       MAY BE CALLED ONLY IMMEDIATELY AFTER ENTER HAS RETURNED TO P+1,
0407                    /       INDICATING THAT A FILE OF THE PROPOSED NAME AND TYPE ALREADY EXISTS.
0410                    /
0411    0252  0055      RPL000, SET RET3          / SAVE RETURN
0412    0253  0000              0
0413    0254  0231              XSK I ENTSW       / TEST FOR NAME FOUND, SET "REPLACE"
0414    0255  0000      RPL010, HLT               / ILLEGAL SEQUENCE -- STOP
0415    0256  0211              XSK ENTSW         / TEST FOR CONTINUE AFTER HALT
0416    0257  6255              JMP RPL010        / BAD BOY -- STOP AGAIN
0417    0260  6241              JMP SETUDF        / GET CORRECT DATA FIELD
0420    0261  1020              LDA I             / ZAP LENGTH FIELD OF THIS FILE
0421    0262  5757              5757
0422    0263  1045              STA XPNT2
0423    0264  6311              JMP FSP000        / ALL LOOKS GOOD -- DO YOUR THING
0424                    /
0425                    /       ENTER -- ADD A NAMED ENTRY TO DIAL INDEX
0426                    /
0427                    /       ENTERED IN LINC MODE WITH AC POINTING TO A PARAMETER LIST
0430                    /       IDENTICAL TO THAT FOR LOOKUP, EXCEPT THAT THE FILE-LENGTH
0431                    /       FIELD IS FILLED BY THE USER.
0432                    /
0433                    /       RETURN IMMEDIATELY FOLLOWING THE JMP (P+1) IF THE FILE ALREADY EXISTS.
0434                    /       RETURN TO P+2 IF THERE IS NO ROOM FOR THE FILE.
0435                    /       RETURN TO P+3 IF OPERATION COMPLETE (DIAL INDEX HAS BEEN UPDATED).
0436                    /
0437    0265  0055      NTR000, SET RET3          / SAVE RETURN
0440    0266  0000              0
0441    0267  6032              JMP LKP010        / LOOKUP NAME IN INDEX
0442    0270  6306              JMP NTR020        / THIS NAME IS NOT IN INDEX
0443    0271  6301              JMP NTR010        / NAME IS IN INDEX, BUT NOT WITH THIS TYPE
0444                    /
0445                    /       WE FOUND A FILE OF THIS NAME AND TYPE
0446                    /
0447    0272  1020              LDA I             / GET CONSTANT -2
0450    0273  7775              -2
0451    0274  2005              ADD XPNT2         / POINT TO POINTER AREA - 2
0452    0275  4010              STC MARK          / SAVE POINTER FOR REPLACE
0453    0276  0071              SET I ENTSW       / INDICATE READY-FOR-REPLACE
0454    0277  1776              1776
0455    0300  6506              JMP RTRN0         / GO HOME
0456                    /
0457                    /       FOUND THE NAME, BUT NOT TYPE
0460                    /
0461    0301  1020      NTR010, LDA I             / CONSTANT -2
0462    0302  7775              -2
0463    0303  2005              ADD XPNT2         / FROM POINTER INDEX
0464    0304  4010              STC MARK          / SAVE ADDR OF FILE POINTERS
0465    0305  6310              JMP NTR030        / GO BEGIN SCAN FOR FILE SPACE
0466                    /
0467                    /       NOTHING FOUND IN THE LOOKUP SCAN
0470                    /
0471    0306  0070      NTR020, SET I MARK        / INDICATE NO FIND
0472    0307  7777              7777
0473    0310  0235      NTR030, XSK I RET3        / NO NAME CONFLICT, INCREMENT RETURN
0474                            EJECT
```

```
0475                    /
0476                    /     FSP - FIND SPACE ON DIAL TAPE FOR NEW FILE
0477                    /
0500    0311  1020  FSP000, LDA I            / PICK UP SKIP-NEG INSTRUCTION
0501    0312  0471          APO I            / MOVED TO FCF050
0502    0313  4550          STC FCF050       / STORE IN INDEX-SCAN ROUTINE
0503    0314  1020          LDA I
0504    0315  0005          5
0505    0316  2001          ADD PARAM        / POINT TO USERS START BLOCK FIELD
0506    0317  1040          STA              / HOLD FOR LATER
0507    0320  0006          LP1
0510    0321  4007          STC LP2          / STORE POINTER
0511    0322  1027          LDA I LP2        / PICK UP LENGTH
0512    0323  0451          APO              / TEST FOR POSITIVE LEN REQUEST
0513    0324  6506          JMP RTRN0        / ELSE RETURN ERROR
0514    0325  0470          AZE I            / SKIP IF LEN NOT ZERO
0515    0326  6506          JMP RTRN0        / BOMB IF NULL REQUEST
0516    0327  1040          STA              / HOLD LENGTH HANDY
0517    0330  0601          TRYLEN
0520    0331  0017          COM
0521    0332  1120          ADA I            / SUBTRACT LEN FROM HIGHEST BLOCK NO, LOW FILE
0522    0333  0270          270
0523    0334  0470  FSP010, AZE I            / SKIP NOT ZERO
0524    0335  0011          CLR              / FORCE TRUE ZERO IF RESULT IS 7777
0525    0336  0451          APO              / SKIP IF STILL ON THE TAPE
0526    0337  6350          JMP FSP020       / NO SPACE IN LOW FILE AREA
0527    0340  4600          STC TRY          / HOLD TRIAL STARTING BLOCK
0530    0341  6513          JMP FCF000       / GO FIND POSSIBLE CONFLICT
0531    0342  6357          JMP FSP030       / HOORAY -- NO CONFLICT
0532    0343  1000          LDA              / NOPE -- THAT TRY IS NO GOOD
0533    0344  0601          TRYLEN
0534    0345  0017          COM              / SUBTRACT THE SEARCH LENGTH FROM...
0535    0346  1104          ADA XPNT         /... THE START BLOCK OF CONFLICT FILE...
0536    0347  6334          JMP FSP010       /...AND TRY AGAIN
0537                    /
0540                    /     WEVE FOUND NO SPACE LARGE ENOUGH IN THE LOWER FILE AREA
0541                    /
0542    0350  0072  FSP020, SET I NFSW       / SET NO-FIND SWITCH
0543    0351  0000          0
0544    0352  1020          LDA I            / PICK UP MINUS END OF TAPE
0545    0353  6777          -1000
0546    0354  2601          ADD TRYLEN       / MINUS LAST FEASIBLE START BLOCK
0547    0355  4603          STC UPLIM        / HOLD AS SCAN LIMIT
0550    0356  6371          JMP FSP035       / ENTER UPPER SCAN
0551                    /
0552                    /     TRY POINTS TO GOOD SPACE IN LOWER AREA
0553                    /
0554    0357  0072  FSP030, SET I NFSW       / INDICATE SPACE FOUND
0555    0360  7777          7777
0556    0361  1000          LDA              / PICK UP OLD TRY
0557    0362  0600          TRY
0560    0363  1040          STA              / SAVE IT
0561    0364  0602          SVTRY
0562    0365  2601          ADD TRYLEN       / GET END BLOCK
0563    0366  1120          ADA I            / GET COMPARE CONSTANT (TRYEND-2*INDEX LOC)
0564    0367  7061          -716
0565    0370  4603          STC UPLIM        / MINUS HIGHEST DESIRABLE STARTING BLOCK
0566    0371  1020  FSP035, LDA I            / GET SKIP-POS INSTRUCTION
0567    0372  0451          APO              / MOVED TO FCF050
0570    0373  4550          STC FCF050       / STORE IN INDEX SCAN ROUTINE
0571    0374  1020          LDA I            / INITIAL STARTING BLOCK, UPPER FILE
0572    0375  0470          470
```

```
0574    0377  0600          TRY
0575    0420  2623          ADD UPLIM       / COMPARE TO MAXIMUM USABLE BLOCK
0576    0401  0471          APO I           / SKIP IF STILL IN USEFUL REGION
0577    0402  6410          JMP FSP050      / NO SPACE IN THIS AREA
0600    0403  6513          JMP FCF000      / SEARCH FOR CONFLICT
0601    0404  6415          JMP FSP060      / WEVE FOUND SPACE
0602    0405  1004          LDA XPNT        / THIS TRY WONT WORK...
0603    0406  1105          ADA XPNT2       /...SO TRY AT END OF CONFLICT FILE
0604    0407  6376          JMP FSP040
0605                  /
0606                  /     THERE WAS NO SPACE FOUND IN UPPER FILE AREA
0607                  /
0610    0410  0212  FSP050, XSK NFSW        / SKIP IF FOUND IN LOWER AREA
0611    0411  6506          JMP RTRN0       / RETURN BAD NEWS -- NO SPACE
0612    0412  1000          LDA             / GET START BLOCK OF FILE IN LOWER AREA
0613    0413  0602          SVTRY
0614    0414  4600          STC TRY         / THATS THE ONE WELL USE
0615                  /
0616                  /     THERE IS SPACE AT TRY
0617                  /
0620    0415  0210  FSP060, XSK MARK        / WAS THERE A FILE BY THIS NAME
0621    0416  6471          JMP FSP100      / YES - GO USE THAT ENTRY
0622                  /
0623                  /     WE MUST SEARCH FOR EMPTY INDEX SPACE
0624                  /
0625    0417  0070          SET I MARK      / INITIALIZE MARK POINTER
0626    0420  1000          INDEX
0627    0421  1020  FSP070, LDA I          / BUMP INDEX INDEX
0630    0422  0007          7
0631    0423  1140          ADM
0632    0424  0010          MARK
0633    0425  0210          XSK MARK        / TEST FOR END OF INDEX
0634    0426  0467          SKP             / NOT END
0635    0427  6506          JMP RTRN0       / NO SPACE IN INDEX -- RETURN ZERO
0636    0430  1030          LDA I MARK      / FIRST WORD OF INDEX NAME
0637    0431  1460          SAE I           / CHECK FOR EMPTY ENTRY
0640    0432  5757          5757
0641    0433  6421          JMP FSP070      / NOT EMPTY, TRY NEXT
0642                  /
0643                  /     MARK POINTS TO AN UNUSED INDEX ENTRY
0644                  /     PUT THE NAME INTO IT
0645                  /
0646    0434  1001          LDA PARAM       / FIRST WORD OF USER NAME
0647    0435  1050          STA MARK        / PLUNK INTO INDEX
0650    0436  1021          LDA I PARAM     / 2ND
0651    0437  1070          STA I MARK
0652    0440  1021          LDA I PARAM     / 3RD
0653    0441  1070          STA I MARK
0654    0442  1021          LDA I PARAM     / 4TH
0655    0443  1070          STA I MARK
0656    0444  1021          LDA I PARAM     / TYPE CONTROL (S OR B)
0657    0445  1420          SHD I           / IS TYPE S
0660    0446  2300          2300
0661    0447  6454          JMP FSP080      / YES
0662    0450  1420          SHD I           / IS TYPE B
0663    0451  0200          0200
0664    0452  6465          JMP FSP090      / YES
0665    0453  6506          JMP RTRN0       / NO-- BOMB  JW
0666                        EJECT
```

```
0667                    /
0670                    /       HE WANTS A SOURCE FILE -- SCRATCH THE BINARY POINTERS
0671                    /
0672    0454  0047  FSP080, SET LP2          / TEMP POINTER
0673    0455  0010          MARK
0674    0456  0227          XSK I LP2       / CANT SKIP
0675    0457  0227          XSK I LP2       / DITTO
0676    0460  1020          LDA I           / GET 57S...
0677    0461  5757          5757
0700    0462  1067          STA I LP2       / STORE THEM...
0701    0463  1067          STA I LP2       /... IN BINARY POINTERS
0702    0464  6471          JMP FSP100
0703                    /
0704                    /       HE WANTS BINARY FILE
0705                    /
0706    0465  1020  FSP090, LDA I
0707    0466  5757          5757
0710    0467  1070          STA I MARK      / STORE 57S...
0711    0470  1070          STA I MARK      /...IN SOURCE POINTERS
0712                    /
0713                    /       MARK NOW POINTS TO POINTER AREA OF DESIRED TYPE IN A NAMED INDEX ENTRY
0714                    /
0715    0471  1000  FSP100, LDA             / PICK UP STARTING BLOCK
0716    0472  0600          TRY
0717    0473  1070          STA I MARK      / STORE IN INDEX
0720    0474  1046          STA LP1         / STORE IN USERS LIST
0721    0475  1026          LDA I LP1       / GET LENGTH
0722    0476  1070          STA I MARK      / STORE IN INDEX
0723                    /
0724                    /       NOW EVERYBODY IS HAPPY EXCEPT THE TAPE, WHICH HASNT BEEN UPDATED
0725                    /
0726    0477  0235          XSK I RET3      / BUMP RETURN ADDR
0727    0500  6243  FSP110, JMP SETMDF      / SET DATA FIELD TO MILDRED
0730    0501  0002          PDP
0731                        PMODE           / PMODE FOR I/O
0732    4502  6212          CIF RWFLD       / ROUTINES IN FIELD ONE
0733    4503  4712          JMS I   PWRITE
0734    4504  4246  PPNT2,  XPARM           / RELOCATED FOR THIS SEGMENT
0735    4505  6141          LINC
0736                        LMODE
0737    0506  0040  RTRN0,  SET 0           / MOVE RETURN JMP TO 0
0740    0507  0015          RET3
0741    0510  0006          DJR
0742    0511  6220          JMP RESTOR      / RETURN TO USER
0743                    /
0744                    /       ADDR OF WRITE ROUTINE IN FIELD 1
0745                    /
0746    0512  7775  PWRITE, WRITE
0747                        EJECT
```

```
0750                    /
0751                    /     FIND POSSIBLE CONFLICT BETWEEN INDEX ENTRY AND TRIAL STARTING BLOCK
0752                    /
0753                    /     CALLER MUST SET OR CLEAR I-BIT IN FCF050 TO SELECT DESIRED FILE AREA
0754                    /
0755      0513  0056  FCF000, SET RET2          / SAVE RETURN
0756      0514  0000          0
0757      0515  0064          SET I XPNT       / INITIALIZE POINTER TO INDEX IN CORE
0760      0516  1006          INDEX+6
0761      0517  0224  FCF010, XSK I XPNT       / SKIP ON END OF INDEX
0762      0520  0467          SKP
0763      0521  6016          JMP RET2         / END OF INDEX, RETURN NO CONFLICT
0764      0522  0224          XSK I XPNT       / INCREMENT AGAIN, NO SKIP POSSIBLE
0765      0523  1000          LDA              / PICK UP POINTER
0766      0524  0004          XPNT
0767      0525  0323          ROR I 3          / MOVE BIT 9 TO LINK
0770      0526  0452          LZE              / SKIP IF NAME AREA OF ENTRY
0771      0527  6545          JMP FCF040       / JMP IF POINTER AREA
0772      0530  1004          LDA XPNT         / GET FIRST WORD OF NAME
0773      0531  1460          SAE I            / SKIP IF EMPTY ENTRY
0774      0532  5757          5757
0775      0533  6541          JMP FCF030       / WORD IS VALID NAME
0776      0534  1020          LDA I            / GET CONSTANT 6
0777      0535  0006          6
1000      0536  1140          ADM              / ADDRESS NEXT ENTRY - 2, THIS ONE IS EMPTY
1001      0537  0004          XPNT
1002      0540  6517          JMP FCF010       / TRY NEXT INDEX ENTRY
1003      0541  1020  FCF030, LDA I            / INCREMENT BY 4
1004      0542  0004          4
1005      0543  1140          ADM              /...TO ADDRESS POINTER
1006      0544  0004          XPNT
1007                    /
1010                    /     XPNT NOW ADDRESSES A STARTING BLOCK NO
1011                    /
1012      0545  1004  FCF040, LDA XPNT         / PICK UP STARTING BLOCK
1013      0546  1120          ADA I            / SUBTRACT INDEX LOCATION
1014      0547  7430          -347
1015      0550  0451  FCF050, APO              / REVERSE SENSE BIT MAY BE SET BY CALLER
1016      0551  6517          JMP FCF010       / TRY AGAIN IF WRONG FILE AREA
1017      0552  0045          SET XPNT2        / TEMP POINT TO LENGTH WORD
1020      0553  0004          XPNT
1021      0554  1025          LDA I XPNT2      / PICK UP LENGTH
1022      0555  0451          APO              / SKIP IF LENGTH POS
1023      0556  6517          JMP FCF010       / NEG LEN -- NO FILE HERE
1024                          EJECT
-
```

```
1025                    /
1026                    /       WE NOW HAVE A VALID INDEX ENTRY
1027                    /       COMPARE IT TO TRY
1030                    /
1031    0557    1004            LDA XPNT        / STARTING BLOCK OF THIS FILE
1032    0560    1560            BCL I           / CLEAR GARBAGE IN HIGH THREE BITS
1033    0561    7000            7000
1034    0562    0017            COM
1035    0563    2600            ADD TRY         / SUBTRACT XSTART FROM TRY
1036    0564    0451            APO             / SKIP IF TRY ABOVE XSTART
1037    0565    6573            JMP FCF060      / JMP IF BELOW
1040                    /
1041                    /       TRY IS ABOVE THE START OF THIS FILE
1042                    /
1043    0566    0017            COM             / MAKE DIFFERENCE NEGATIVE
1044    0567    1105            ADA XPNT2       / SUBTRACT DIFFERENCE FROM FILE LENGTH
1045    0570    0451            APO             / ZERO RESULT WILL BE NEG (7777)
1046    0571    6517            JMP FCF010      / NO CONFLICT, TRY NEXT ENTRY
1047    0572    6576            JMP FCF070      / CONFLICT FOUND -- RETURN
1050                    /
1051                    /       THE TRY IS BELOW OR AT THE START OF THIS FILE
1052                    /
1053    0573    2601    FCF060, ADD TRYLEN      / SUBTRACT STARTING DIFF FROM LEN OF TRY
1054    0574    0451            APO             / SKIP IF CONFLICT
1055    0575    6517            JMP FCF010      / NO CONFLICT - TRY NEXT ENTRY
1056                    /
1057                    /       WE HAVE FOUND A CONFLICT -- RETURN IT TO CALLER
1060                    /
1061    0576    0236    FCF070, XSK I RET2      / INCREMENT RETURN ADDRESS
1062    0577    6016            JMP RET2        / GO BACK
1063                    /
1064                    /       WORK AREA
1065                    /
1066    0600    0000    TRY,    0
1067    0601    0000    TRYLEN, 0
1070    0602    0000    SVTRY,  0
1071    0603    0000    UPLIM,  0
1072                            EJECT
```

```
1073                    /
1074                    /       DELETE -- REMOVE A FILE FROM THE DIAL INDEX
1075                    /
1076                    /       SAME CALLING SEQUENCE AS LOOKUP, EXCEPT NO ALTERNATE RETURNS
1077                    /
1100        0604  0055  DEL000, SET RET3          / SAVE RETURN
1101        0605  0000          0
1102        0606  6032          JMP LKP010        / LOOKUP NAME IN INDEX
1103        0607  6506          JMP RTRN0         / DIDNT FIND THE FILE
1104        0610  6506          JMP RTRN0         / DITTO
1105        0611  1020          LDA I             / DECREMENT XPNT2
1106        0612  7776          -1
1107        0613  1140          ADM               /...TO ADDRESS START BLOCK FIELD
1110        0614  0005          XPNT2
1111        0615  1020          LDA I             / EMPTY AREA INDICATOR
1112        0616  5757          5757
1113        0617  1045          STA XPNT2         / ZAP START BLOCK FIELD
1114        0620  1065          STA I XPNT2       / DITTO LEN
1115        0621  1000          LDA               / GET POINTER
1116        0622  0005          XPNT2
1117        0623  1660          BCO I             / ADDRESS OTHER TYPE LEN FIELD
1120        0624  0002          2
1121        0625  4005          STC XPNT2
1122        0626  1005          LDA XPNT2         / PICK LENGTH OF OTHER-TYPE FILE
1123        0627  0471          APO I             / SKIP IF OTHER-TYPE EMPTY
1124        0630  6500          JMP FSP110        / GO RE-WRITE INDEX
1125        0631  1020          LDA I             / BOTH TYPES EMPTY -- CLEAR NAME
1126        0632  5757          5757
1127        0633  1044          STA XPNT          / ZAP FIRST WORD OF NAME
1130        0634  1064          STA I XPNT        / 2ND
1131        0635  1064          STA I XPNT        / 3RD
1132        0636  1064          STA I XPNT        / 4TH
1133                    /
1134                    /       WAS THAT THE LAST FILE ?
1135                    /
1136        0637  0064          SET I XPNT        / POINT TO INDEX
1137        0640  0777          INDEX-1           /...FOR SCAN
1140        0641  1464  DEL010, SAE I XPNT       / IS THERE ANY ENTRY?
1141        0642  6500          JMP FSP110        / YES - REWRITE INDEX
1142        0643  0204          XSK XPNT          / HAVE WE CHECKED ALL?
1143        0644  6641          JMP DEL010        / NO - LOOP
1144                    /
1145                    /       INDEX IS EMPTY: FILL WITH ZERO
1146                    /
1147        0645  0011          CLR
1150        0646  0064          SET I XPNT        / RESET POINTER
1151        0647  0777          INDEX-1
1152        0650  1064          STA I XPNT        / CLEAR NEXT WORD
1153        0651  0204          XSK XPNT          / DONE?
1154        0652  6650          JMP .-2
1155        0653  6500          JMP FSP110        / RE-WRITE INDEX
1156                            EJECT
-
```

```
1157                /
1160                /     THIS WILL CAUSE AN ASSEMBLY ERROR IF ROUTINE BECOMES SO LARGE
1161                /     THAT INDEX WILL OVERLAY CODE WHEN IT IS READ IN
1162                        ASMIFM INDEX-.
1163                        NAUGHTY BAD BOY - ROUTINE IS TOO BIG FOR ONE FIELD.
1164                /       IN CASE OF MINOR SIZE PROBLEMS, REMOVE 5 LINES OF CODE AT FCFØ4Ø.
1165                /       REMOVE 3 LINES AT FSPØØØ, AND THREE LINES AT FSPØ35.  THIS SHOULD
1166                /       REMOVE ALL REFERENCES TO FCFØ5Ø.  ALL OTHER TAGS MUST BE RETAINED.
1167                /
1170                /
1171                /
1172                /       END OF MILDRED
1173                /
1174                        LISTAPE 12
```

NO ERRORS

```
SYMBOL  VALUE  DEF  REFERENCES

DEL000    )4  1100  0061
DEL010  4641  1140  1143
ENTSW   0011  0021  0120 0413 0415 0453
ERRTN   4214  0321  0243 0264 0345 0352
FCF000  4513  0755  0530 0600
FCF010  4517  0761  1002 1016 1023 1046 1055
FCF030  4541  1003  0775
FCF040  4545  1012  0771
FCF050  4550  1015  0502 0570
FCF060  4573  1053  1037
FCF070  4576  1061  1047
FDV     0002  0013  0273 0277 0307 0342 0350
FSP000  4311  0500  0423
FSP010  4334  0523  0536
FSP020  4350  0542  0526
FSP030  4357  0554  0531
FSP035  4371  0566  0550
FSP040  4376  0573  0604
FSP050  4410  0610  0577
FSP060  4415  0620  0601
FSP070  4421  0627  0641
FSP080  4454  0672  0661
FSP090  4465  0706  0664
FSP100  4471  0715  0621 0702
FSP110  4500  0727  1124 1141 1155
INDEX   1000  0036  0223 0626 0760 1137 1151 1162
LKP000  4030  0102  0053
LKP010  4032  0104  0441 1102
LKP020  4155  0254  0230
LKP030  4157  0256  0270 0301
LKP050  4200  0277  0303
LP1     0006  0016  0507 0720 0721
LP2     0007  0017  0275 0302 0510 0511 0672 0674 0675 0700 0701
LRIB    6234  0043  0122
LRIF    6224  0044  0157
LRMF    6244  0042
MARK    0010  0020  0452 0464 0471 0620 0625 0632 0633 0636 0647 0651 0653 0655 0673 0710 0711 0717 0722
NFSW    0012  0022  0542 0554 0610
NTR000  4265  0437  0055
NTR010  4301  0461  0443
NTR020  4306  0471  0442
NTR030  4310  0473  0465
PARAM   0001  0012  0117 0205 0241 0254 0274 0505 0646 0650 0652 0654 0656
PPNT1   4132  0216  0177
PPNT2   4504  0734  0200
PPOINT  4245  0371  0175
PREAD   4154  0247  0215
PWRITE  4512  0746  0733
READ    7774  0030  0247
RESTDF  4220  0326  0141 0143
RESTIF  4221  0327  0132 0145
RESTOR  0220  0325  0742
RET2    0016  0024  0104 0322 0324 0340 0351 0755 0763 1061 1062
RET3    0015  0023  0102 0321 0337 0346 0411 0437 0473 0726 0740 1100
RPL000  4252  0411  0057
RPL010  4255  0414  0416
RTRN0   4506  0737  0455 0513 0515 0611 0635 0665 1103 1104
RWFLD   0010  0032  0214 0732
SAVFLD  4047  2124  0133
SETMDF  4243  2365  0171 0211 0727
```

```
SYMBOL   VALUE  DEF   REFERENCES

SVTRY    4602   1070  0561 0613
TRY      4600   1066  0527 0557 0574 0614 0716 1035
TRYLEN   4601   1067  0517 0533 0546 0562 1053
UPLIM    4603   1071  0547 0565 0575
WNTB     4223   0335  0315
WNTS     4225   0337  0312
WORD1    4170   0267  0255
WRITE    7775   0031  0746
XPARM    4246   0375  0162 0210 0216 0371 0734
XPNT     0004   0014  0222 0224 0236 0237 0242 0261 0262 0265 0272 0535 0602 0757 0761 0764 0766 0772 1001 1006 1012 1 20 1031 1127
                      1130 1131 1132 1136 1140 1142 1150 1152 1153
XPNT2    0005   0015  0271 0300 0335 0336 0341 0343 0422 0451 0463 0603 1017 1021 1044 1110 1113 1114 1116 1121 1122
```

MILDRED
DEC-12-FZDA-D

READER'S COMMENTS

Digital Equipment Corporation maintains a continuous effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback – your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy, organization, usability, and readability.

_____

_____

_____

_____

_____

Did you find errors in this manual? _____

_____

_____

_____

_____

How can this manual be improved? _____

_____

_____

_____

_____

_____

DEC also strives to keep its customers informed of current DEC software and publications. Thus, the following periodically distributed publications are available upon request. Please check the appropriate boxes for a current issue of the publication(s) desired.

☐ Software Manual Update, a quarterly collection of revisions to current software manuals.

☐ User's Bookshelf, a bibliography of current software manuals.

☐ Program Library Price List, a list of currently available software programs and manuals.

Please describe your position. _____

Name _____ Organization _____

Street _____ Department _____

City _____ State _____ Zip or Country _____

Fold Here

Do Not Tear - Fold Here and Staple

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Digital Equipment Corporation
Software Information Services
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

HOW TO OBTAIN SOFTWARE INFORMATION

Announcements for new and revised software, as well as programming notes, software problems, and documentation corrections are published by Software Information Service in the following newsletters.

Digital Software News for the PDP-8 Family
Digital Software News for the PDP-9/15 Family
PDP-6/PDP-10 Software Bulletin

These newsletters contain information applicable to software available from Digital's Program Library.

Please complete the card below to place your name on the newsletter mailing list.

Questions or problems concerning DEC Software should be reported to the Software Specialist at your nearest DEC regional or district sales office. In cases where no Software Specialist is available, please send a Software Trouble Report form with details of the problem to:

Software Information Service
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

These forms, which are available without charge from the Program Library, should be fully filled out and accompanied by teletype output as well as listings or tapes of the user program to facilitate a complete investigation. An answer will be sent to the individual and appropriate topics of general interest will be printed in the newsletter.

New and revised software and manuals, Software Trouble Report forms, and cumulative Software Manual Updates are available from the Program Library. When ordering, include the document number and a brief description of the program or manual requested. Revisions of programs and documents will be announced in the newsletters and a price list will be included twice yearly. Direct all inquiries and requests to:

Program Library
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

Digital Equipment Computer Users Society (DECUS) maintains a user Library and publishes a catalog of programs as well as the DECUSCOPE magazine for its members and non-members who request it. For further information please write to:

DECUS
Digital Equipment Corporation
146 Main Street
Maynard, Massachusetts 01754

Send Digital's software newsletters to:

Name_____
Company Name_____
Address_____
_____
(zip code)

My computer is a

PDP-8/I ☐		PDP-8/L ☐	
LINC-8 ☐		PDP-12 ☐	
PDP-9 ☐		PDP-15 ☐	Please specify
PDP-10 ☐		OTHER ☐	_____

My system serial number is_____ (if known)

Fold Here

Do Not Tear - Fold Here and Staple

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Digital Equipment Corporation
Software Information Services
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754