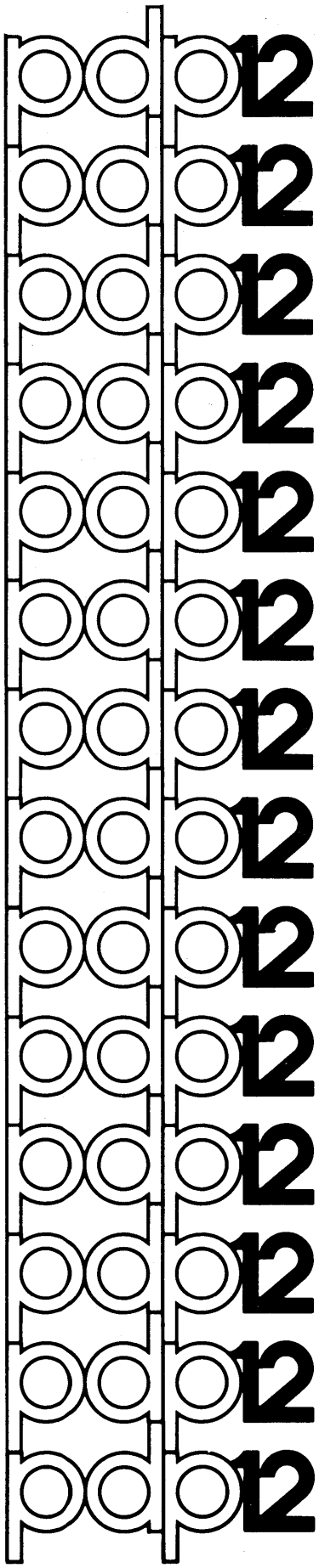
digital

FRED



DEC-12-FZFA-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	COMPUTER LAB

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

TABLE OF CONTENTS

1.0 PROGRAM OVERVIEW

FRED (File Replacement, Entry, and Deletion) is a set of PDP-12 subroutines for manipulation of LAP6-DIAL[1] indices.

There are four 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

FRED occupies two tape blocks and, when in core, uses four LINC memory blocks (2000_8 words), including space for the index. The routines are segment-independent, but must be loaded at a segment boundary. Thus memory addresses of 2000, 4000, 6000, 14000, etc., can be used, but 2400 or 3000 may not.

In this discussion, all locations are relative to the segment into which FRED is loaded.

3.0 USAGE

3.1 The user's program must load FRED from a DIAL tape, 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.

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

2Ø - LOOKUP
22 - ENTER
24 - REPLACE
26 - DELETE
3Ø - READ
35 - WRITE

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

READ and WRITE are called as follows:

```
LIF X                   /SEGMENT INTO WHICH FRED IS LOADED
LDA I                   /LOAD AC WITH PARAMETER POINTER
RWPARM                  /POINTER TO READ/WRITE PARAMETER LIST
JMP 3Ø (JMP 35)         /DO READ (WRITE)
 .
 .
 .
 .
 .
 .
RWPARM,  Y\UNIT         /HIGH-ORDER THREE BITS FOR FIELD
                        /LOW-ORDER THREE BITS FOR TAPE UNIT
BUFFER                  /12-BIT MEMORY ADDRESS OF DATA
BLOCKNO                 /BLOCK NUMBER OF FIRST TAPE BLOCK
COUNT                   /NO. OF BLOCKS TO READ/WRITE
```

The COUNT must not be zero.

Return is to the instruction following the JMP If AC Bit 1 is Ø, RWPARM is taken from the caller's instruction segment if 1, the parameter list is in his data segment. Note: There is no check for attempts by the user to write over LAP6-DIAL, nor is there a check to prevent reading over FRED.

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

```
LIF X                   /SEGMENT WITH FRED
LDA I                   /AC:  POINTER TO FILE DESCRIPTOR VECTOR
FDV                     /GO TO LOOKUP (ENTER, DELETE)
JMP 2Ø (JMP 22, 26)
 .
 .
 .
 .
 .
FDV, UNIT               /LINC TAPE Ø-7
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 FRED
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 310 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 FRED
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 FRED must be reloaded.

4.0 DESCRIPTION OF THE ROUTINES:

4.1 GETFLD: (Level Ø) called to obtain the address of the the user's parameter list, which he placed in AC before the call. AC is stored in PARAM (Beta 1). The save field buffer is read and stored at SAVFLD. Bit 1 of the parameter list is in the caller's data field, which is also FRED's data field, and GETFLD returns.

If zero, the parameter list is in the caller's instruction field. That field is then obtained from SAVFLD and used to construct an LDF instruction at GTFØ1Ø, which is executed to set FRED's data field to the caller's instruction field. Bit 1 is set in PARAM and GETFLD exits.

4.2 READ: (Level 1, entry point 3Ø). The return JMP at Ø is saved at RETURN (Beta 17). An RDC instruction is placed in RDWR (Beta 2), and READ jumps to COMMOM. See WRITE.

4.3 WRITE: (Level 1, entry point 35). Return JMP is saved at RETURN (Beta 17), and a WRC instruction is moved to RDWR (Beta 2). From this point (COMMON), READ and WRITE are the same routine. RET2 (Beta 16) is set to 7777 to indicate that the call was from a user outside this field. GETFLD is called to set up PARAM (Beta 1) to point to the parameter list. The next location, RWENT, is the entry point for internal calls by higher lever routines, which must have performed appropriate setup. The 8-mode field (high-order 3 bits of 15-bit address) is obtained from the parameter list and stored at EXT for use in extended addressing. The two high-order bits of the unit number are then moved to AC_{10-11}, with the low-order bit in Linc. Tape extended operations are then set. The 12 low-order bits of the memory address are obtained from the parameter list and stored at MEMADD. The low-order unit bit is obtained from Link, combined with the READ or WRITE instruction at RDWR, and stored at CMND The first block number is obtained from the parameter list and stored, following the command, at 10BLK.

The block count, also from the parameter list, is complemented and stored at RDWR (Beta 2) as a loop control.

IOLOOP begins by setting the tape memory address register,

then updating the address at MEMADD. The READ or WRITE instruction is executed, the block number is incremented, and RDWR is incremented to test for completion. If more I/O remains, there is a jump to IOLOOP. Otherwise, RET2 is tested to determine whether the call was internal or external. If internal (RET2 $\neq$ 7777), return is immediated. If external, the user's fields are restored before returning.

4.4 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), and GETFLD is called. PARAM (Beta 1) is saved at PARM2 (Beta 3). The current instruction field is obtained, and used to set the address into which the index will be read. Parameters and return address are set up for READ, which is then called at RWENT to bring in the index. Upon return, the index is checked for validity (5757 is 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 later 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 02 (B). If neither, the error return is taken. S causes a jump 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 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.

4.5 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 into 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.

4.6 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 set the 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 accomodate 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 an empty entry is found, the file name from the user's parameter list is moved in. 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 re-writing 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.

4.7 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 length 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.

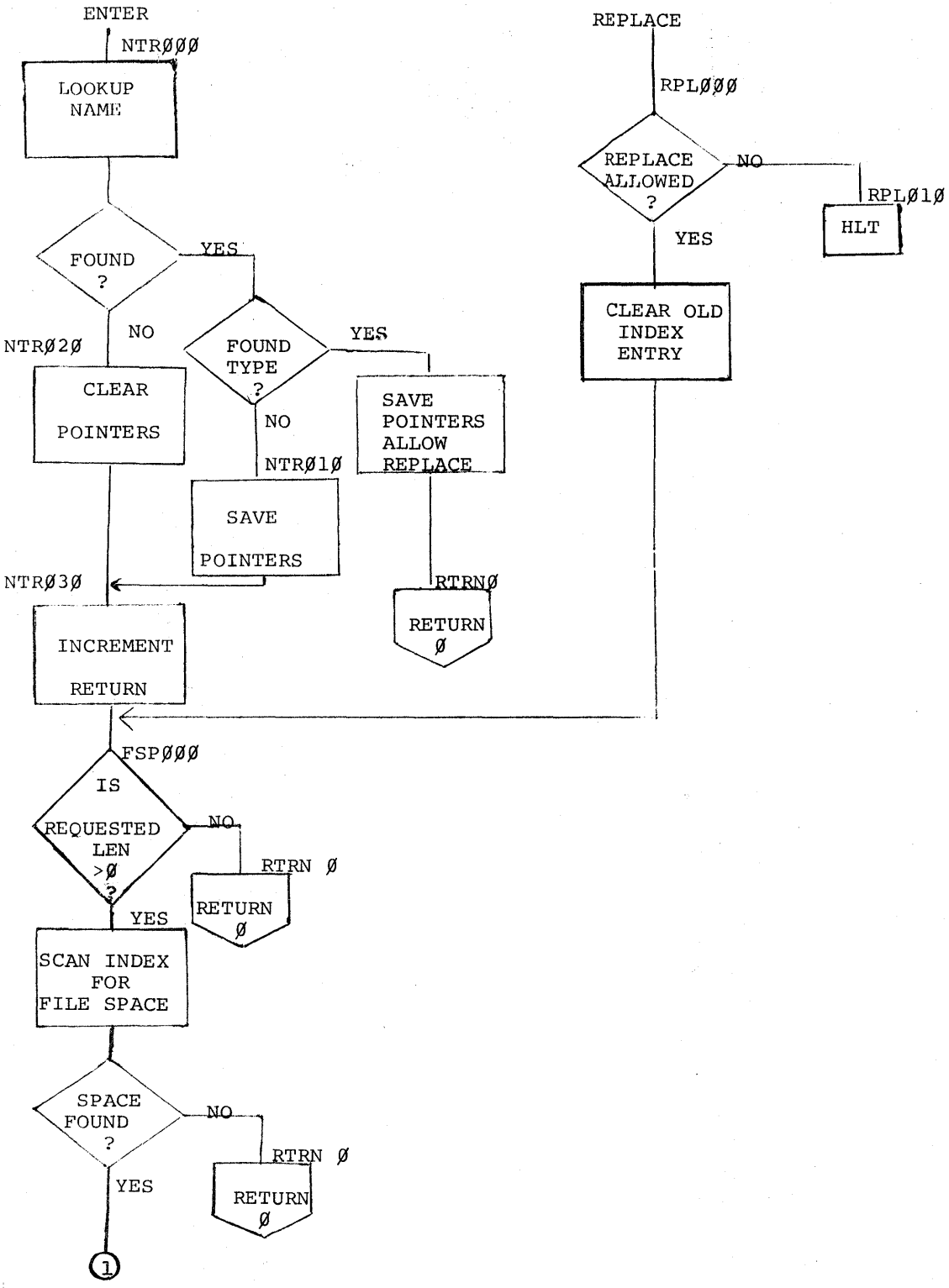
If the name is not empty, XPNT is incremented by 4, to address the source-file pointers, and control flows into FCFØ4Ø. FCFØ5Ø having been set to an APO, or APO I instruction, the start block is compared to the index TBLK to determine whether the file is in the wrong area for this scan. If it is the 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.

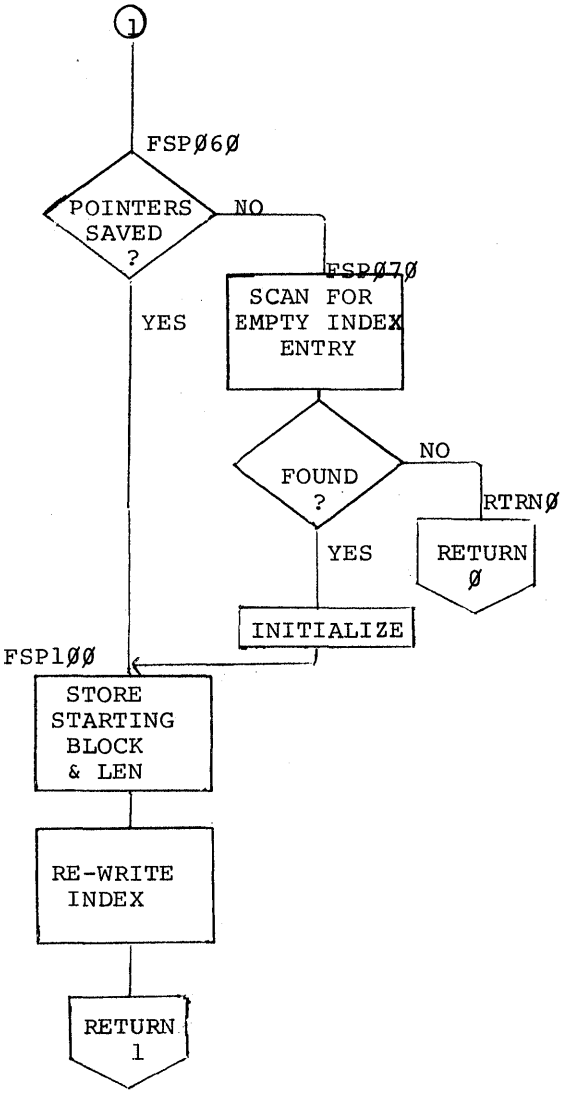
XPNT2 is set to address the length field, which is tested for validity. If negative, there is no file of that type, and the loop is re-entered at FCFØ1Ø. If the length is positive, the starting block is subtracted from TRY. If the result is negative or zero, TRY is below the start of this file by complement 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.

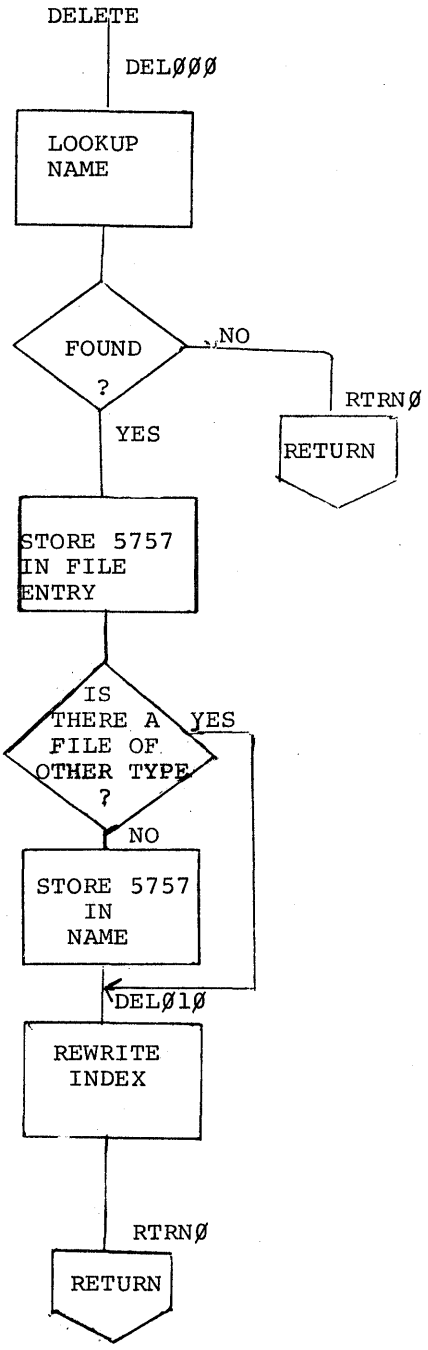
The code at FCFØ6Ø is entered when TRY is below the start of this file. TRYLEN is added to the complement of the block difference. A negative or zero result implies no conflict, and the scan continues at FCFØ1Ø. A positive result is a conflict, so control flows into FCFØ7Ø, which increments RET2 and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

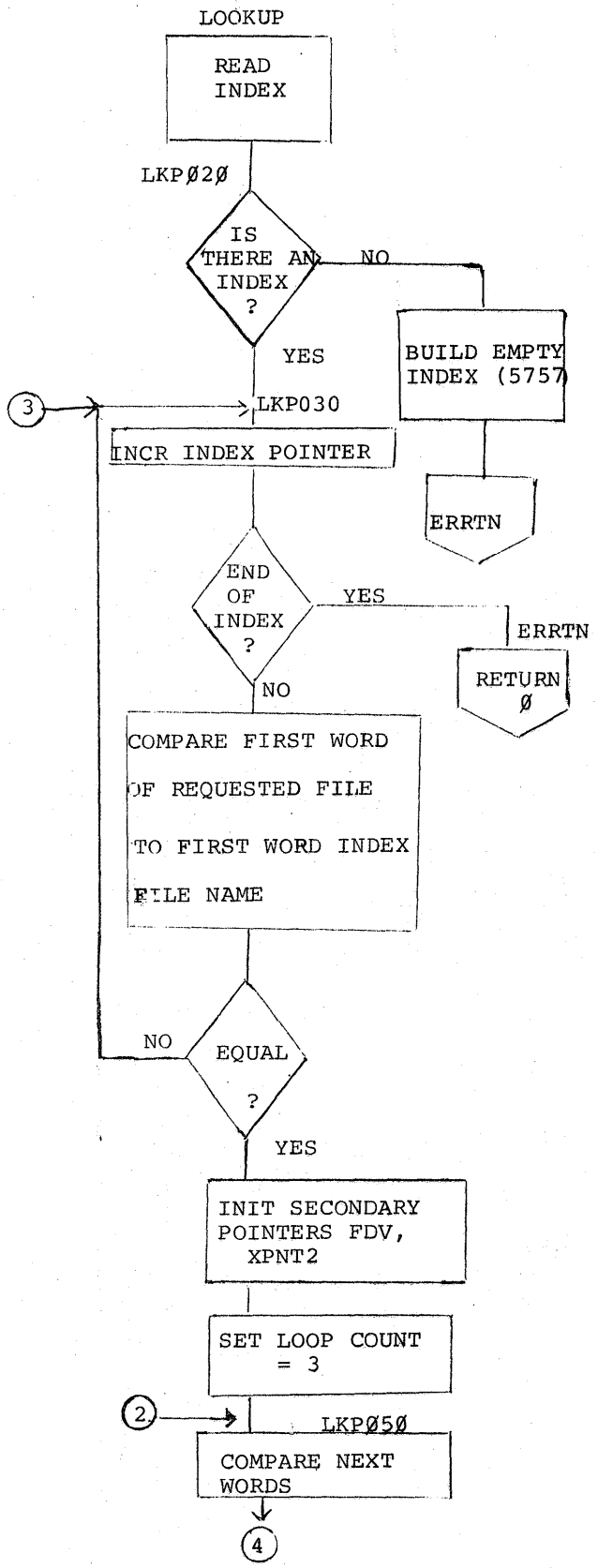
4.8 DELETE: (Level 3, Entry Point 26). Beginning at DELØØØ, the return jump is saved at RET3 (Beta 15). LOOKUP is then called via the internal entry point LKPØØØ to find the name and file to be deleted. On each of the alternate returns, a JMP RTRNØ is taken, since it is unnecessary to delete a file which doesn't 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 DEL1Ø. If negative, there is no file of the other type, so the name area of the index entry is set to 5757. At DEL1Ø, the write code, return jump, and parameter pointers are set up, and the I/O handler is called at RWENT to rewrite the index.

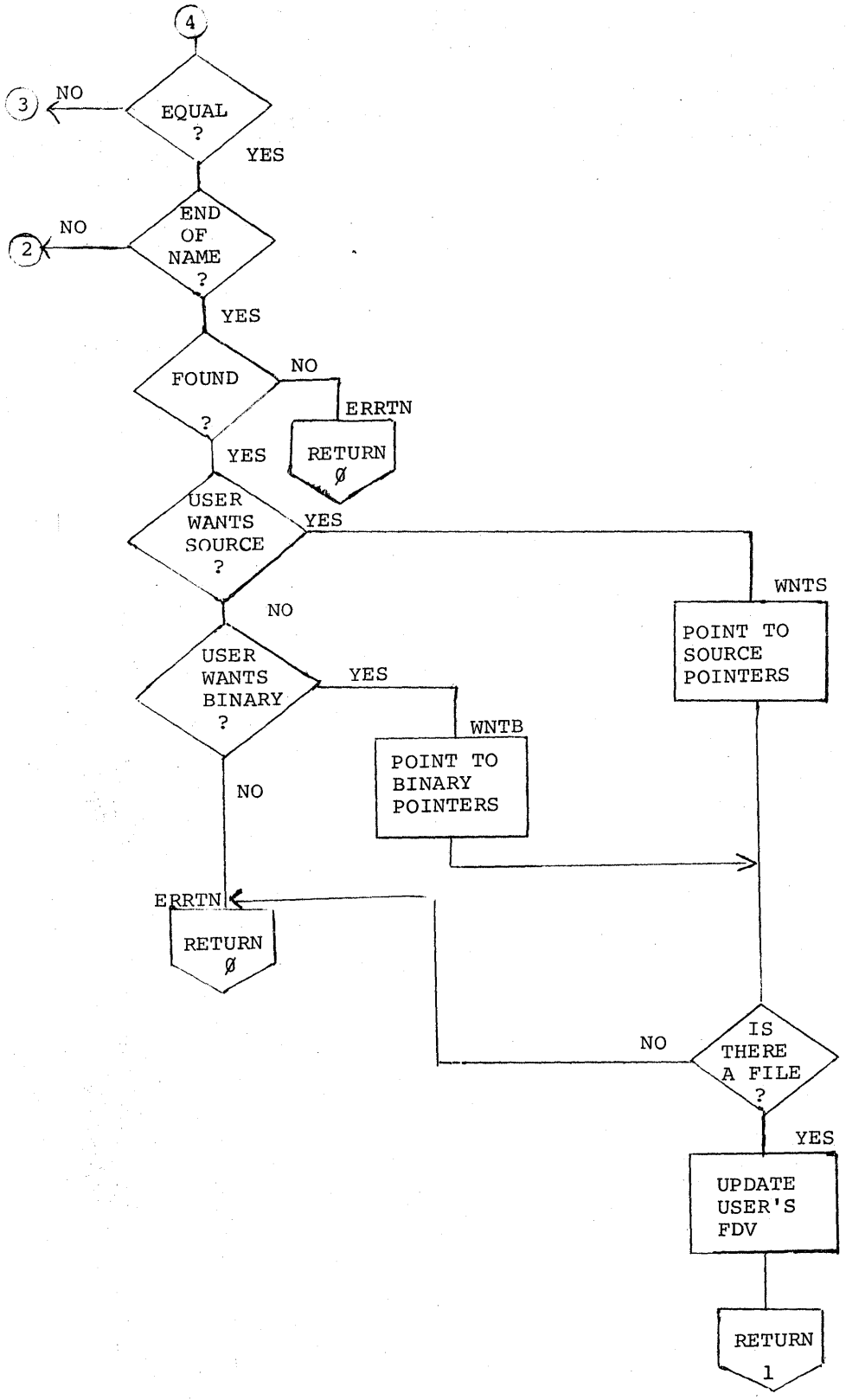
5.0 FLOW CHARTS


ENTER
NTRØØØ
LOOKUP NAME
FOUND ?
YES
NO
NTRØ2Ø
CLEAR POINTERS
FOUND TYPE ?
YES
NO
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
DELØ1Ø
REWRITE INDEX
RTRNØ
RETURN


LOOKUP
READ INDEX
LKPØ2Ø
IS THERE AN INDEX ?
NO
YES
BUILD EMPTY INDEX (5757)
3
LKP030
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
3
NO
EQUAL ?
YES
2
NO
END OF NAME ?
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
NO
IS THERE A FILE ?
YES
UPDATE USER'S FDV
RETURN 1

INDEX

```
0000                    *20
0001                    /       FRED -- FILE REPLACEMENT, ENTRY, AND DELETION
0002                    /       MAR 19, 1970
0003                    /
0004                    /
0005                    /       BETA REGISTER DEFINITIONS
0006                    /               (13 AND 14 ARE UNUSED)
0007                    /
0010                            PARAM=1
0011                            RDWR=2
0012                            FDV=2
0013                            PARM2=3
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                            RETURN=17
0026                    /
0027                    /   LOAD ADDRESS OF DIAL INDEX -- DO NOT MODIFY: RAMIFICATIONS ARE OVERWHELMING
0030                    /
0031                            INDEX=1000
0032                    /
0033                    /   PDP-8 MODE INSTRUCTIONS FOR USE AFTER IOB
0034                    /
0035                            LRMF=6244
0036                            LRIB=6234
0037                            LRIF=6224
0040                    /
0041                    /
0042                    /     ENTRY POINTS FOR MAJOR ROUTINES OF FRED
0043                    /
0044                    /
0045    0020    0006            DJR
0046    0021    6142            JMP LOOKUP
0047    0022    0006            DJR
0050    0023    6321            JMP NTR000
0051    0024    0006            DJR
0052    0025    6303            JMP RPL000
0053    0026    0006            DJR
0054    0027    6641            JMP DEL000
0055                            EJECT
```

```
0060                    /
0061                    /     ENTERED IN LINC MODE, WITH AC CONTAINING POINTER TO PARAMETER LIST.
0062                    /     IF BIT 1 IS ZERO, PARAMETERS ARE IN THE CALLERS INSTRUCTION FIELD.
0063                    /     IF ONE, PARAMETERS ARE IN HIS DATA FIELD.
0064                    /     PARAMETER LIST AS FOLLOWS --
0065                    /
0066                    /   0/ FIELD (3 BITS) /     UNIT      /
0067                    /   1/     MEMORY ADDRESS (12 BITS)   /
0070                    /   2/     FIRST BLOCK NUMBER         /
0071                    /   3/     NUMBER OF BLOCKS           /
0072                    /
0073    0030  0057  READ,   SET RETURN      / SAVE RETURN ADDRESS
0074    0031  0000          0
0075    0032  0062          SET I RDWR     / INDICATE READ OPERATION
0076    0033  0700          700
0077    0034  6041          JMP COMMON     / GO TO COMMON PROCESSING
0100    0035  0057  WRITE,  SET RETURN     / SAVE RETURN
0101    0036  0000          0
0102    0037  0062          SET I RDWR     / INDICATE WRITE OPERATION
0103    0040  0704          704
0104    0041  0076  COMMON, SET I RET2     / SET CODE FOR USER CALL
0105    0042  7777          7777
0106    0043  6116          JMP GETFLD     / SETUP TO ADDRESS PARAMETERS
0107    0044  1001  RWENT,  LDA PARAM      / GET FIRST WORD OF PARAMETERS
0110    0045  0306          ROR 6          / MOVE BANK NO TO RIGHT HALF
0111    0046  1340          STH            / HOLD FOR EXTENDED ADDRESSING
0112    0047  0054          EXT
0113    0050  0266          ROL I 6        / RESTORE UNIT BITS RIGHT
0114    0051  1560          BCL I          / KEEP TWO HIGH-ORDER BITS FOR AXO
0115    0052  7774          7774           /. . .  PLUS LOW-ORDER IN LINK
0116    0053  1620          BSE I          / SET HIGH-ORDER UNIT BITS FOR AXO
0117    0054  0020  EXT,    0020
0120    0055  0001          AXO            / SET EXTENDED OPERATIONS BITS
0121    0056  1021          LDA I PARAM    / GET MEMORY ADDR FROM PARAMS
0122    0057  4072          STC MEMADD     / HOLD FOR TAPE USE
0123    0060  0264          ROL I 4        / GET UNIT BIT INTO POSITION
0124    0061  1600          BSE            / SET DESIRED READ/WRITE OP CODE
0125    0062  0002          RDWR
0126    0063  4077          STC CMND       / SET READ/WRITE INSTRUCTION
0127    0064  1021          LDA I PARAM    / GET START BLOCK NUMBER
0130    0065  4100          STC IOBLK      / SET APPOROPRIATELY
0131    0066  1021          LDA I PARAM    / BLOCK COUNT...
0132    0067  0017          COM            /... MADE NEGATIVE
0133    0070  4002          STC RDWR       / STORE IN INDEX
0134    0071  1020  IOLOOP, LDA I          / GET MEMORY ADDRESS FOR DATA
0135    0072  0000  MEMADD, 0              / ADDRESS FOR NEXT BLOCK
0136    0073  0023          TMA            / TELL THE TAPE WHERE
0137    0074  1120          ADA I          / UPDATE ADDRESS
0140    0075  0400          400            / BY BLOCK LENGTH
0141    0076  4072          STC MEMADD     / STORE NEW ADDR
0142    0077  0700  CMND,   RDC            / MODIFIED FOR DESIRED OPERATION
0143    0100  0000  IOBLK,  0              / BLOCK NUMBER WANTED
0144    0101  1020          LDA I          / GET CONSTANT 1
0145    0102  0001          1
0146    0103  1140          ADM            / UPDATE BLOCK NUMBER
0147    0104  0100          IOBLK
0150    0105  0222          XSK I RDWR     / SKIP IF END OF OPERATION
0151    0106  6071          JMP IOLOOP     / ELSE CONTINUE
0152                        EJECT
```

```
0154                    /       IO IS COMPLETE -- RETURN TO CALLER
0155                    /
0156    0107    0216            XSK RET2        / SKIP IF USER CALL
0157    0110    6017            JMP RETURN      / RETURN NOW - INTERNAL CALL
0160    0111    0500            IOB
0161    0112    6244            LRMF            / RESTORE TO DESIRED STATUS
0162    0113    0040            SET Ø           / RESTORE RETURN JMP
0163    0114    0017            RETURN
0164    0115    6000            JMP Ø           / GO TO CALLER
0165                    /
0166                    /
0167                    /
0170                    /       GETFLD -- GET CALLERS FIELDS AND PARAMETERS
0171                    /
0172                    /       ENTERED IN LINC MODE, WITH 10-BIT PARAMETER POINTER IN AC.
0173                    /       RETURN WITH INDIRECT POINTER IN LOCATION 1 TO
0174                    /       DATA FIELD CONTAINING PARAMETERS.
0175                    /
0176    0116    1620    GETFLD, BSE I           / SET BIT Ø (SOON BECOMES BIT 1)
0177    0117    4000            4000
0200    0120    0262            ROL I 2         / BIT 1 TO LINK
0201    0121    0302            ROR 2           / RESTORE 2-11, SET BIT 1
0202    0122    4001            STC PARAM       / HOLD AC CONTENTS
0203    0123    4011            STC ENTSW       / CLEAR ENTSW
0204    0124    0500            IOB
0205    0125    6234            LRIB
0206    0126    0303            ROR 3           / INST FLD TO AC 7-11
0207    0127    0452            LZE             / SKIP IF PARMS IN INST FLD
0210    0130    0245            ROL 5           / DF TO AC 7-11
0211    0131    1560            BCL I           / DROP OTHER BITS
0212    0132    7740            7740
0213    0133    1620            BSE I           / BUILD LDF INST
0214    0134    0640            LDF
0215    0135    4140            STC GTFØ1Ø      / SAVE IT
0216    0136    0452            LZE             / IS DF ALREADY SET?
0217    0137    6000            JMP Ø           / YES
0220    0140    0640    GTFØ1Ø, LDF             / NO - SET IT
0221    0141    6000            JMP Ø           / RETURN TO CALLER
0222                            EJECT
```

```
0224                    /     LOOKUP - FIND NAMED FILE IN DIAL INDEX
0225                    /
0226                    /     ENTERED IN LINC MODE WITH ADDRESS OF A PARAMETER
0227                    /     LIST IN AC.
0230                    /
0231                    /   0/ UNIT NUMBER  /
0232                    /   1/ FILE NAME    /
0233                    /   2/  CONTD       /
0234                    /   3/  CONTD       /
0235                    /   4/  CONTD       /
0236                    /   5/ TYPE (S OR B)/
0237                    /   6/ STARTING BLOCK
0240                    /   7/ NO OF BLOCKS /
0241                    /
0242    0142    0075    LOOKUP, SET I RET3      / INDICATE EXTERNAL CALL
0243    0143    7777            7777
0244    0144    0056    LKP000, SET RET2        / SAVE RETURN JMP
0245    0145    0000            0
0246    0146    6116            JMP GETFLD      / SETUP PARAMETER POINTER
0247                    /
0250                    /     READ THE INDEX
0251                    /
0252    0147    0043            SET PARM2       / MOVE PARAMETER POINTER TO ALT AREA
0253    0150    0001            PARAM
0254    0151    1020            LDA     I       / INIT AC
0255    0152    0001            1
0256    0153    0500            IOB
0257    0154    6224            LRIF            / GET INSTRUCTION FIELD
0260    0155    0243            ROL 3           / MOVE FIELD NO TO RIGHT HALFWORD
0261    0156    1340            STH             / STORE DATA ADDRESS FOR INDEX READ
0262    0157    0300            XPARM+1
0263    0160    0243            ROL 3           / BANK NO TO BITS 0-2
0264    0161    1560            BCL I           / DROP ALL BUT BANK NO
0265    0162    0777            0777
0266    0163    1603            BSE PARM2       / COMBINE WITH UNIT NO
0267    0164    4277            STC XPARM       / STORE INTO PARAMETER LIST
0270    0165    0062            SET I RDWR      / SETUP READ CODE
0271    0166    0700            700
0272    0167    0077            SET I RETURN    / SETUP RETURN JMP FROM READ
0273    0170    6174            JMP LKP010      / INST MOVED FOR LATER USE
0274    0171    0061            SET I PARAM     / SETUP PARAMETER POINTER
0275    0172    0277            XPARM
0276    0173    6044            JMP RWENT       / GO TO READ INDEX
0277    0174    0064    LKP010, SET I XPNT
0300    0175    1000            INDEX
0301    0176    1004            LDA XPNT        / GET FIRST WORD OF INDEX
0302    0177    1460            SAE I           / SKIP IF VALID FOR INDEX
0303    0200    5757            5757            / FIRST WORD OF INDEX
0304    0201    0467            SKP             / NOT AN INDEX -- BUILD ONE
0305    0202    6213            JMP LKP020      / INDEX IS OK -- GO TO NAME SCAN
0306                    /
0307                    /     THERE IS NO INDEX -- CREATE ONE
0310                    /
0311    0203    1020            LDA I           / GET FILLER WORD
0312    0204    5757            5757
0313    0205    1044            STA XPNT        / STORE IN NEXT INDEX WORD
0314    0206    0224            XSK I XPNT      / INCREMENT AND TEST FOR END
0315    0207    6205            JMP .-2         / ZAP ANOTHER
0316    0210    1044            STA XPNT        / ZAP LAST WORD
0317    0211    0223            XSK I PARM2     / MAKE PARM2 L[illegible] LIKE WE SOUGHT MATCH
0320    0212    6252            JMP ERRTN       / RETURN NO FI[illegible]
0321                            EJECT
```

```
0323                    /      SCAN INDEX FOR NAME
0324                    /
0325    0213  1023  LKP020, LDA I PARM2      / GET FIRST WORD OF NAME
0326    0214  4226          STC WORD1        / SAVE AT COMPARE INST
0327    0215  1020  LKP030, LDA I            / CONSTANT 7
0330    0216  0007          7
0331    0217  1140          ADM              / ADD IT TO INDEX POINTER
0332    0220  0004          XPNT
0333    0221  0204          XSK XPNT         / TEST FOR END OF INDEX
0334    0222  0467          SKP              / SKIP NOT END
0335    0223  6252          JMP ERRTN        / OTHERWISE RETURN NO SUCCESS
0336    0224  1024          LDA I XPNT       / GET A WORD OF INDEX NAME
0337    0225  1460          SAE I            / SKIP OUT OF LOOP IF FIRST WORDS EQUAL
0340    0226  0000  WORD1,   0
0341    0227  6215          JMP LKP030       / LOOP IF NOT EQUAL
0342    0230  0045          SET XPNT2        / FIRST WORDS EQUAL
0343    0231  0004          XPNT             / START FINAL COMPARE
0344    0232  0042          SET FDV          / POINT TEMPORARY FDV INDEX
0345    0233  0003          PARM2
0346    0234  0067          SET I LP2        / SET LOOP COUNTER
0347    0235  7774          -3
0350    0236  1022  LKP050, LDA I FDV        / GET NEXT TWO CHARS
0351    0237  1465          SAE I XPNT2      / COMPARE TO THOSE IN INDEX ENTRY
0352    0240  6215          JMP LKP030       / UNEQUAL - RETURN TO SEARCH LOOP
0353    0241  0227          XSK I LP2        / LOOP TO COMPARE ENTIRE NAME
0354    0242  6236          JMP LKP050
0355                    /
0356                    /      IF WE GET HERE, WEVE FOUND THE NAME
0357                    /
0360    0243  1022            LDA I FDV      / PICK UP SOURCE/BINARY CODE
0361    0244  1420            SHD I          / IS THE CODE S?
0362    0245  2300            2300
0363    0246  6263            JMP WNTS       / YES - GIVE HIM SOURCE
0364    0247  1420            SHD I          / IS THE CODE B?
0365    0250  0200            0200
0366    0251  6261            JMP WNTB       / YES - GIVE HIM BINARY
0367                    /
0370                    /      COME HERE ON DETECTING ERROR, OR UNSUCCESSFUL FIND
0371                    /
0372    0252  0215  ERRTN,   XSK RET3         / TEST FOR INTERNAL CALL
0373    0253  6016          JMP RET2         / RETURN NOW IF INTERNAL
0374    0254  0040          SET 0            / MOVE IN RETURN JMP
0375    0255  0016          RET2
0376    0256  0500          IOB
0377    0257  6244          LRMF
0400    0260  6000          JMP 0            /RETURN TO CALLER
0401                        EJECT
```

```
0403                    /     WEVE FOUND WHAT HE WANTS - GIVE IT HIM
0404                    /
0405    0261  0225  WNTB,    XSK I XPNT2       / CANT SKIP, BUT THIS ...
0406    0262  0225          XSK I XPNT2       /...FAKES OUT THE POINTER
0407    0263  0215  WNTS,    XSK RET3          / IF INTERNAL CALL...
0410    0264  0236          XSK I RET2        /...SETUP FOR THREE RETURNS
0411    0265  1025          LDA I XPNT2       / GET STARTING BLOCK NO FROM INDEX
0412    0266  1062          STA I FDV         / STORE IN CALLERS PARAMETER LIST
0413    0267  1025          LDA I XPNT2       / GET NO OF BLOCKS
0414    0270  0451          APO               / SKIP IF DESIRED DATA EXISTS
0415    0271  6252          JMP ERRTN         / RETURN ERROR IF NO FILE
0416    0272  0215          XSK RET3          / DONT STORE RESULT IF INTERNAL
0417    0273  0467          SKP
0420    0274  1062          STA I FDV         / RETURN NO OF BLOCKS
0421    0275  0236          XSK I RET2        / INCREMENT JMP RETURN ADDRESS
0422    0276  6252          JMP ERRTN         / NOT REALLY AN ERROR
0423                    /
0424                    /     PARAMETERS TO READ ROUTINE
0425                    /
0426    0277  0000  XPARM,    0                 / FIELD AND UNIT
0427    0300  1000          INDEX             / DATA ADDRESS
0430    0301  0346          346               / BLOCK NUMBER
0431    0302  0002          2                 / BLOCK COUNT
0432                        EJECT
-
```

P. 6

P. 6

```
0434                    /     REPLACE -- REPLACE A NAMED ENTRY IN DIAL INDEX
0435                    /
0436                    /     ENTERED IN LINC MODE AFTER CALLING ENTER.
0437                    /     MAY BE CALLED ONLY IMMEDIATELY AFTER ENTER HAS RETURNED TO P+1,
0440                    /     INDICATING THAT A FILE OF THE PROPOSED NAME AND TYPE ALREADY EXISTS.
0441                    /
0442    0303  0055  RPL000, SET RET3          / SAVE RETURN
0443    0304  0000          0
0444    0305  0231          XSK I ENTSW       / TEST FOR NAME FOUND, SET "REPLACE"
0445    0306  0000  RPL010, HLT               / ILLEGAL SEQUENCE -- STOP
0446    0307  0211          XSK ENTSW         / TEST FOR CONTINUE AFTER HALT
0447    0310  6306          JMP RPL010        / BAD BOY -- STOP AGAIN
0450    0311  1000          LDA               / PICK UP LDF INSTRUCTION
0451    0312  0140          GTF010
0452    0313  4314          STC .+1
0453    0314  0000          0                 / BECOMES AN LDF INSTRUCTION
0454    0315  1020          LDA I             / ZAP LENGTH FIELD OF THIS FILE
0455    0316  5757          5757
0456    0317  1045          STA XPNT2
0457    0320  6345          JMP FSP000        / ALL LOOKS GOOD -- DO YOUR THING
0460                    /
0461                    /     ENTER -- ADD A NAMED ENTRY TO DIAL INDEX
0462                    /
0463                    /     ENTERED IN LINC MODE WITH AC POINTING TO A PARAMETER LIST
0464                    /     IDENTICAL TO THAT FOR LOOKUP, EXCEPT THAT THE FILE-LENGTH
0465                    /     FIELD IS FILLED BY THE USER.
0466                    /
0467                    /     RETURN IMMEDIATELY FOLLOWING THE JMP (P+1) IF THE FILE ALREADY EXISTS.
0470                    /     RETURN TO P+2 IF THERE IS NO ROOM FOR THE FILE.
0471                    /     RETURN TO P+3 IF OPERATION COMPLETE (DIAL INDEX HAS BEEN UPDATED).
0472                    /
0473    0321  0055  NTR000, SET RET3          / SAVE RETURN
0474    0322  0000          0
0475    0323  6144          JMP LKP000        / LOOKUP NAME IN INDEX
0476    0324  6342          JMP NTR020        / THIS NAME IS NOT IN INDEX
0477    0325  6335          JMP NTR010        / NAME IS IN INDEX, BUT NOT WITH THIS TYPE
0500                    /
0501                    /     WE FOUND A FILE OF THIS NAME AND TYPE
0502                    /
0503    0326  1020          LDA I             / GET CONSTANT -2
0504    0327  7775          -2
0505    0330  2005          ADD XPNT2         / POINT TO POINTER AREA - 2
0506    0331  4010          STC MARK          / SAVE POINTER FOR REPLACE
0507    0332  0071          SET I ENTSW       / INDICATE READY-FOR-REPLACE
0510    0333  1776          1776
0511    0334  6543          JMP RTRN0         / GO HOME
0512                    /
0513                    /     FOUND THE NAME, BUT NOT TYPE
0514                    /
0515    0335  1020  NTR010, LDA I             / CONSTANT -2
0516    0336  7775          -2
0517    0337  2005          ADD XPNT2         / FROM POINTER INDEX
0520    0340  4010          STC MARK          / SAVE ADDR OF FILE POINTERS
0521    0341  6344          JMP NTR030        / GO BEGIN SCAN FOR FILE SPACE
0522                    /
0523                    /     NOTHING FOUND IN THE LOOKUP SCAN
0524                    /
0525    0342  0070  NTR020, SET I MARK        / INDICATE NO FIND
0526    0343  7777          7777
0527    0344  0235  NTR030, XSK I RET3        / NO NAME CONFLICT, INCREMENT RETURN
0530                        EJECT
```

```
0532                    /     FSP - FIND SPACE ON DIAL TAPE FOR NEW FILE
0533                    /
0534    0345  1020  FSP000, LDA I           / PICK UP SKIP-NEG INSTRUCTION
0535    0346  0471         APO I           / MOVED TO FCF050
0536    0347  4605         STC FCF050      / STORE IN INDEX-SCAN ROUTINE
0537    0350  1020         LDA I
0540    0351  0005         5
0541    0352  2003         ADD PARM2       / POINT TO USERS START BLOCK FIELD
0542    0353  1040         STA             / HOLD FOR LATER
0543    0354  0006         LP1
0544    0355  4007         STC LP2         / STORE POINTER
0545    0356  1027         LDA I LP2       / PICK UP LENGTH
0546    0357  0451         APO             / TEST FOR POSITIVE LEN REQUEST
0547    0360  6543         JMP RTRN0       / ELSE RETURN ERROR
0550    0361  0470         AZE I           / SKIP IF LEN NOT ZERO
0551    0362  6543         JMP RTRN0       / BOMB IF NULL REQUEST
0552    0363  1040         STA             / HOLD LENGTH HANDY
0553    0364  0636         TRYLEN
0554    0365  0017         COM
0555    0366  1120         ADA I           / SUBTRACT LEN FROM HIGHEST BLOCK NO, LOW FILE
0556    0367  0270         270
0557    0370  0470  FSP010, AZE I          / SKIP NOT ZERO
0560    0371  0011         CLR             / FORCE TRUE ZERO IF RESULT IS 7777
0561    0372  0451         APO             / SKIP IF STILL ON THE TAPE
0562    0373  6404         JMP FSP020      / NO SPACE IN LOW FILE AREA
0563    0374  4635         STC TRY         / HOLD TRIAL STARTING BLOCK
0564    0375  6550         JMP FCF000      / GO FIND POSSIBLE CONFLICT
0565    0376  6413         JMP FSP030      / HOORAY -- NO CONFLICT
0566    0377  1000         LDA             / NOPE -- THAT TRY IS NO GOOD
0567    0400  0636         TRYLEN
0570    0401  0017         COM             / SUBTRACT THE SEARCH LENGTH FROM...
0571    0402  1104         ADA XPNT        /... THE START BLOCK OF CONFLICT FILE...
0572    0403  6370         JMP FSP010      /...AND TRY AGAIN
0573                    /
0574                    /     WEVE FOUND NO SPACE LARGE ENOUGH IN THE LOWER FILE AREA
0575                    /
0576    0404  0072  FSP020, SET I NFSW      / SET NO-FIND SWITCH
0577    0405  0000         0
0600    0406  1020         LDA I           / PICK UP MINUS END OF TAPE
0601    0407  6777         -1000
0602    0410  2636         ADD TRYLEN      / MINUS LAST FEASIBLE START BLOCK
0603    0411  4640         STC UPLIM       / HOLD AS SCAN LIMIT
0604    0412  6425         JMP FSP035      / ENTER UPPER SCAN
0605                    /
0606                    /     TRY POINTS TO GOOD SPACE IN LOWER AREA
0607                    /
0610    0413  0072  FSP030, SET I NFSW      / INDICATE SPACE FOUND
0611    0414  7777         7777
0612    0415  1000         LDA             / PICK UP OLD TRY
0613    0416  0635         TRY
0614    0417  1040         STA             / SAVE IT
0615    0420  0637         SVTRY
0616    0421  2636         ADD TRYLEN      / GET END BLOCK
0617    0422  1120         ADA I           / GET COMPARE CONSTANT (TRYEND-2*INDEX LOC)
0620    0423  7061         -716
0621    0424  4640         STC UPLIM       / MINUS HIGHEST DESIRABLE STARTING BLOCK
0622    0425  1020  FSP035, LDA I          / GET SKIP-POS INSTRUCTION
0623    0426  0451         APO             / MOVED TO FCF050
0624    0427  4605         STC FCF050      / STORE IN INDEX SCAN ROUTINE
0625    0430  1020         LDA I           / INITIAL STARTING BLOCK, UPPER FILE
0626    0431  0470         470
0627    0432  1040  FSP040, STA            / SET NEW TRY BLOCK
```

```
0631  0434  2640          ADD UPLIM       / COMPARE TO MAXIMUM USABLE BLOCK
0632  0435  0471          APO I           / SKIP IF STILL IN USEFUL REGION
0633  0436  6444          JMP FSP050      / NO SPACE IN THIS AREA
0634  0437  6550          JMP FCF000      / SEARCH FOR CONFLICT
0635  0440  6451          JMP FSP060      / WEVE FOUND SPACE
0636  0441  1004          LDA XPNT        / THIS TRY WONT WORK...
0637  0442  1105          ADA XPNT2       /...SO TRY AT END OF CONFLICT FILE
0640  0443  6432          JMP FSP040
0641                /
0642                /     THERE WAS NO SPACE FOUND IN UPPER FILE AREA
0643                /
0644  0444  0212  FSP050, XSK NFSW        / SKIP IF FOUND IN LOWER AREA
0645  0445  6543          JMP RTRN0       / RETURN BAD NEWS -- NO SPACE
0646  0446  1000          LDA             / GET START BLOCK OF FILE IN LOWER AREA
0647  0447  0637          SVTRY
0650  0450  4635          STC TRY         / THATS THE ONE WELL USE
0651                /
0652                /     THERE IS SPACE AT TRY
0653                /
0654  0451  0210  FSP060, XSK MARK        / WAS THERE A FILE BY THIS NAME
0655  0452  6525          JMP FSP100      / YES - GO USE THAT ENTRY
0656                /
0657                /     WE MUST SEARCH FOR EMPTY INDEX SPACE
0660                /
0661  0453  0070          SET I MARK      / INITIALIZE MARK POINTER
0662  0454  1000          INDEX
0663  0455  1020  FSP070, LDA I           / BUMP INDEX INDEX
0664  0456  0007          7
0665  0457  1140          ADM
0666  0460  0010          MARK
0667  0461  0210          XSK MARK        / TEST FOR END OF INDEX
0670  0462  0467          SKP             / NOT END
0671  0463  6543          JMP RTRN0       / NO SPACE IN INDEX -- RETURN ZERO
0672  0464  1030          LDA I MARK      / FIRST WORD OF INDEX NAME
0673  0465  1460          SAE I           / CHECK FOR EMPTY ENTRY
0674  0466  5757          5757
0675  0467  6455          JMP FSP070      / NOT EMPTY, TRY NEXT
0676                /
0677                /     MARK POINTS TO AN UNUSED INDEX ENTRY
0700                /     PUT THE NAME INTO IT
0701                /
0702  0470  1003          LDA PARM2       / FIRST WORD OF USER NAME
0703  0471  1050          STA MARK        / PLUNK INTO INDEX
0704  0472  1023          LDA I PARM2     / 2ND
0705  0473  1070          STA I MARK
0706  0474  1023          LDA I PARM2     / 3RD
0707  0475  1070          STA I MARK
0710  0476  1023          LDA I PARM2     / 4TH
0711  0477  1070          STA I MARK
0712  0500  1023          LDA I PARM2     / TYPE CONTROL (S OR B)
0713  0501  1420          SHD I           / IS TYPE S
0714  0502  2300          2300
0715  0503  6510          JMP FSP080      / YES
0716  0504  1420          SHD I           / IS TYPE B
0717  0505  0200          0200
0720  0506  6521          JMP FSP090      / YES
0721  0507  6543          JMP RTRN0       / NO-- BOMB NOW
0722                      EJECT
```

```
0724                    /     HE WANTS A SOURCE FILE -- SCRATCH THE BINARY POINTERS
0725                    /
0726    0510  0047      FSP080, SET LP2          / TEMP POINTER
0727    0511  0010              MARK
0730    0512  0227              XSK I LP2        / CANT SKIP
0731    0513  0227              XSK I LP2        / DITTO
0732    0514  1020              LDA I            / GET 57S...
0733    0515  5757              5757
0734    0516  1067              STA I LP2        / STORE THEM...
0735    0517  1067              STA I LP2        /... IN BINARY POINTERS
0736    0520  6525              JMP FSP100
0737                    /
0740                    /     HE WANTS BINARY FILE
0741                    /
0742    0521  1020      FSP090, LDA I
0743    0522  5757              5757
0744    0523  1070              STA I MARK       / STORE 57S...
0745    0524  1070              STA I MARK       /...IN SOURCE POINTERS
0746                    /
0747                    /     MARK NOW POINTS TO POINTER AREA OF DESIRED TYPE IN A NAMED INDEX ENTRY
0750                    /
0751    0525  1000      FSP100, LDA              / PICK UP STARTING BLOCK
0752    0526  0635              TRY
0753    0527  1070              STA I MARK       / STORE IN INDEX
0754    0530  1046              STA LP1          / STORE IN USERS LIST
0755    0531  1026              LDA I LP1        / GET LENGTH
0756    0532  1070              STA I MARK       / STORE IN INDEX
0757                    /
0760                    /     NOW EVERYBODY IS HAPPY EXCEPT THE TAPE, WHICH HASNT BEEN UPDATED
0761                    /
0762    0533  0062              SET I RDWR       / SETUP WRITE CODE
0763    0534  0704              704
0764    0535  0077              SET I RETURN     / SETUP RETURN JMP
0765    0536  6542              JMP FSP110       / INST MOVED FOR LATER USE
0766    0537  0061              SET I PARAM      / SETUP PARAMETER POINTER
0767    0540  0277              XPARM
0770    0541  6044              JMP RWENT        / GO DO THE WRITE
0771    0542  0235      FSP110, XSK I RET3       / BUMP RETURN ADDR
0772    0543  0040      RTRN0,  SET 0            / MOVE RETURN JMP TO 0
0773    0544  0015              RET3
0774    0545  0500              IOB
0775    0546  6244              LRMF             / RESTORE USERS FIELDS
0776    0547  6000              JMP 0            / GO TO CALLER
0777                            EJECT
-
```

```
1000                    /
1001                    /     FIND POSSIBLE CONFLICT BETWEEN INDEX ENTRY AND TRIAL STARTING BLOCK          //
1002                    /
1003                    /     CALLER MUST SET OR CLEAR I-BIT IN FCF050 TO SELECT DESIRED FILE AREA
1004                    /
1005    0550  0056  FCF000, SET RET2          / SAVE RETURN
1006    0551  0000          0
1007    0552  0064          SET I XPNT       / INITIALIZE POINTER TO INDEX IN CORE
1010    0553  1006          INDEX+6
1011    0554  0224  FCF010, XSK I XPNT       / SKIP ON END OF INDEX
1012    0555  0467          SKP
1013    0556  6016          JMP RET2         / END OF INDEX, RETURN NO CONFLICT
1014    0557  0224          XSK I XPNT       / INCREMENT AGAIN, NO SKIP POSSIBLE
1015    0560  1000          LDA              / PICK UP POINTER
1016    0561  0004          XPNT
1017    0562  0323          ROR I 3          / MOVE BIT 9 TO LINK
1020    0563  0452          LZE              / SKIP IF NAME AREA OF ENTRY
1021    0564  6602          JMP FCF040       / JMP IF POINTER AREA
1022    0565  1004          LDA XPNT         / GET FIRST WORD OF NAME
1023    0566  1460          SAE I            / SKIP IF EMPTY ENTRY
1024    0567  5757          5757
1025    0570  6576          JMP FCF030       / WORD IS VALID NAME
1026    0571  1020          LDA I            / GET CONSTANT 6
1027    0572  0006          6
1030    0573  1140          ADM              / ADDRESS NEXT ENTRY - 2, THIS ONE IS EMPTY
1031    0574  0004          XPNT
1032    0575  6554          JMP FCF010       / TRY NEXT INDEX ENTRY
1033    0576  1020  FCF030, LDA I            / INCREMENT BY 4
1034    0577  0004          4
1035    0600  1140          ADM              /...TO ADDRESS POINTER
1036    0601  0004          XPNT
1037                    /
1040                    /     XPNT NOW ADDRESSES A STARTING BLOCK NO
1041                    /
1042    0602  1004  FCF040, LDA XPNT         / PICK UP STARTING BLOCK
1043    0603  1120          ADA I            / SUBTRACT INDEX LOCATION
1044    0604  7430          -347
1045    0605  0451  FCF050, APO              / REVERSE SENSE BIT MAY BE SET BY CALLER
1046    0606  6554          JMP FCF010       / TRY AGAIN IF WRONG FILE AREA
1047    0607  0045          SET XPNT2        / TEMP POINT TO LENGTH WORD
1050    0610  0004          XPNT
1051    0611  1025          LDA I XPNT2      / PICK UP LENGTH
1052    0612  0451          APO              / SKIP IF LENGTH POS
1053    0613  6554          JMP FCF010       / NEG LEN -- NO FILE HERE
1054                        EJECT
```

```
1056                    /       WE NOW HAVE A VALID INDEX ENTRY
1057                    /       COMPARE IT TO TRY
1060                    /
1061    0614    1004            LDA XPNT        / STARTING BLOCK OF THIS FILE
1062    0615    1560            BCL I           / CLEAR GARBAGE IN HIGH THREE BITS
1063    0616    7000            7000
1064    0617    0017            COM
1065    0620    2635            ADD TRY         / SUBTRACT XSTART FROM TRY
1066    0621    0451            APO             / SKIP IF TRY ABOVE XSTART
1067    0622    6630            JMP FCF060      / JMP IF BELOW
1070                    /
1071                    /       TRY IS ABOVE THE START OF THIS FILE
1072                    /
1073    0623    0017            COM             / MAKE DIFFERENCE NEGATIVE
1074    0624    1105            ADA XPNT2       / SUBTRACT DIFFERENCE FROM FILE LENGTH
1075    0625    0451            APO             / ZERO RESULT WILL BE NEG (7777)
1076    0626    6554            JMP FCF010      / NO CONFLICT, TRY NEXT ENTRY
1077    0627    6633            JMP FCF070      / CONFLICT FOUND -- RETURN
1100                    /
1101                    /       THE TRY IS BELOW OR AT THE START OF THIS FILE
1102                    /
1103    0630    2636    FCF060, ADD TRYLEN      / SUBTRACT STARTING DIFF FROM LEN OF TRY
1104    0631    0451            APO             / SKIP IF CONFLICT
1105    0632    6554            JMP FCF010      / NO CONFLICT - TRY NEXT ENTRY
1106                    /
1107                    /       WE HAVE FOUND A CONFLICT -- RETURN IT TO CALLER
1110                    /
1111    0633    0236    FCF070, XSK I RET2      / INCREMENT RETURN ADDRESS
1112    0634    6016            JMP RET2        / GO BACK
1113                    /
1114                    /       WORK AREA
1115                    /
1116    0635    0000    TRY,    0
1117    0636    0000    TRYLEN, 0
1120    0637    0000    SVTRY,  0
1121    0640    0000    UPLIM,  0
1122                            EJECT
-
```

12

p. 12

```
1124                    /     DELETE -- REMOVE A FILE FROM THE DIAL INDEX
1125                    /
1126                    /     SAME CALLING SEQUENCE AS LOOK  EXCEPT NO ALTERNATE RETURNS
1127                    /
1130    0641    0055    DEL000, SET RET3        / SAVE RETURN
1131    0642    0000            0
1132    0643    6144            JMP LKP000      / LOOKUP NAME IN INDEX
1133    0644    6543            JMP RTRN0       / DIDNT FIND THE FILE
1134    0645    6543            JMP RTRN0       / DITTO
1135    0646    1020            LDA I           / DECREMENT XPNT2
1136    0647    7776            -1
1137    0650    1140            ADM             /...TO ADDRESS START BLOCK FIELD
1140    0651    0005            XPNT2
1141    0652    1020            LDA I           / EMPTY AREA INDICATOR
1142    0653    5757            5757
1143    0654    1045            STA XPNT2       / ZAP START BLOCK FIELD
1144    0655    1065            STA I XPNT2     / DITTO LEN
1145    0656    1000            LDA             / GET POINTER
1146    0657    0005            XPNT2
1147    0660    1660            BCO I           / ADDRESS OTHER TYPE LEN FIELD
1150    0661    0002            2
1151    0662    4005            STC XPNT2
1152    0663    1005            LDA XPNT2       / PICK LENGTH OF OTHER-TYPE FILE
1153    0664    0471            APO I           / SKIP IF OTHER-TYPE EMPTY
1154    0665    6710            JMP DEL030      / GO RE-WRITE INDEX
1155    0666    1020            LDA I           / BOTH TYPES EMPTY -- CLEAR NAME
1156    0667    5757            5757
1157    0670    1044            STA XPNT        / ZAP FIRST WORD OF NAME
1160    0671    1064            STA I XPNT      / 2ND
1161    0672    1064            STA I XPNT      / 3RD
1162    0673    1064            STA I XPNT      / 4TH
1163                    /
1164                    /       WAS THAT THE LAST FILE
1165                    /
1166    0674    0064            SET I XPNT      / POINT TO START OF INDEX
1167    0675    0777            INDEX-1
1170    0676    1464    DEL010, SAE I XPNT      / IS THERE A NON-EMPTY ENTRY
1171    0677    6710            JMP DEL030      / YES - REWRITE
1172    0700    0204            XSK XPNT        / CHECK FOR END
1173    0701    6676            JMP DEL010      / NOT YET - LOOP
1174                    /
1175                    /       INDEX IS EMPTY: MAKE IT A NON-INDEX
1176                    /
1177    0702    0011            CLR
1200    0703    0064            SET I XPNT      / POINT TO INDEX
1201    0704    0777            INDEX-1
1202    0705    1064            STA I XPNT      / ZERO ONE WORD
1203    0706    0204            XSK XPNT        / LAST ONE ?
1204    0707    6705            JMP .-2         / NO - LOOP
1205                    /
1206                    /     RE-WRITE INDEX
1207                    /
1210    0710    0062    DEL030, SET I RDWR      / SET WRITE CODE
1211    0711    0704            704
1212    0712    0077            SET I RETURN    / RETURN JUMP FOR R/W ROUTINE
1213    0713    6543            JMP RTRN0
1214    0714    0061            SET I PARAM
1215    0715    0277            XPARM
1216    0716    6344            JMP RWENT       / DO THE WRITE THING, AND RETURN TO CALLER
1217                            EJECT
```

```
1221                    /       THIS WILL CAUSE AN ASSEMBLY ERROR IF ROUTINE BECOMES SO LARGE
1222                    /       THAT INDEX WILL OVERLAY CODE WHEN IT IS READ IN
1223                              ASMIFM INDEX-.
1224                              NAUGHTY BAD BOY - ROUTINE IS TOO BIG FOR ONE FIELD.
1225                    /         IN CASE OF MINOR SIZE PROBLEMS, REMOVE 5 LINES OF CODE AT FCF042.
1226                    /         REMOVE 3 LINES AT FSP000, AND THREE LINES AT FSP035.  THIS SHOULD
1227                    /         REMOVE ALL REFERENCES TO FCF050.  ALL OTHER TAGS MUST BE RETAINED.
1230                    /
1231                    /
1232                    /
1233                    /         END OF FRED
1234                    /
```

p.14

```
COMMON 4041
DEL000 4641
DEL010 4676
DEL030 4710
ENTSW  0011
ERRTN  4252
EXT    4054
FCF000 4550
FCF010 4554
FCF030 4576
FCF040 4602
FCF050 4605
FCF060 4630
FCF070 4633
FDV    0002
FSP000 4345
FSP010 4370
FSP020 4404
FSP030 4413
FSP035 4425
FSP040 4432
FSP050 4444
FSP060 4451
FSP070 4455
FSP080 4510
FSP090 4521
FSP100 4525
FSP110 4542
GETFLD 4116
GTF010 4140
INDEX  1000
IOBLK  4100
IOLOOP 4071
LKP000 4144
LKP010 4174
LKP020 4213
LKP030 4215
LKP050 4236
LOOKUP 4142
LP1    0006
LP2    0007
LRIB   6234
LRIF   6224
LRMF   6244
MARK   0010
MEMADD 4072
-
```

NFSW 0212
NTR00 4321
NTR010 4335
NTR0[redacted] 4342
NTR0[redacted] 4344
PARAM 2001
PARM2 0003
RDWR 0002
READ 4032
RETURN 0017
RET2 0016
RET3 0015
RPL000 4303
RPL010 4306
RTRN0 4543
RWENT 4044
SVTRY 4637
TRY 4635
TRYLEN 4636
UPLIM 4640
WNTB 4261
WNTS 4263
WORD1 4226
WRITE 4035
XPARM 4277
XPNT 0004
XPNT2 0005

P.17