NBSIR 75-713

A File Management System for a Laboratory Automation Facility

Peter S. Shoenfeld and Lawrence J. Kaetzel

National Bureau of Standards
Department of Commerce
Washington, D. C. 20234

June 1975

Final Report

Prepared for

Analytical Chemistry Division
Institute for Materials Research
National Bureau of Standards
Washington, D. C. 20234

NBSIR 75–713

A FILE MANAGEMENT SYSTEM FOR A LABORATORY AUTOMATION FACILITY

Peter S. Shoenfeld and Lawrence J. Kaetzel

National Bureau of Standards
Department of Commerce
Washington, D. C. 20234

June 1975

Final Report

Prepared for
Analytical Chemistry Division
Institute for Materials Research
National Bureau of Standards
Washington, D. C. 20234

U.S. DEPARTMENT OF COMMERCE, Rogers C.B. Morton, Secretary

NATIONAL BUREAU OF STANDARDS, Ernest Ambler, Acting Director

TABLE OF CONTENTS

Page

1. INTRODUCTION. 1

2. DISK DIRECTORY. 2

3. WORK AREAS FOR OPEN FILES 2

4. GENERAL STRUCTURE 2

5. FMGR -- Executive Scheduling Routine. 2

A. Entry Points. 6
B. Vectors, System Parameters, and Reserved Areas. 6
C. Operation 7
D. Error Conditions. 8
E. Batch Communication with FMGR 9

6. XFIN -- Executive Completion Routine. 10

7. NXACT -- Executive Queue Processing Routine 10

8. NBS DISK ALLOCATOR. 11

9. ACTION PROGRAMS 11

A. Linkage (Subroutine FMLINK) and Conventions . . 12
B. NEWID -- Establish New I.D. No. 14
C. RMVID -- Remove an I.D. No. 15
D. CREATE -- Establish a New File. 16
E. REMOVE -- Remove a File 17
F. OPEN -- Make a File Available for Use 18
G. CLOSE -- Complete Usage of File 19
H. WRITE -- Output Data. 20
I. READ -- Input Data. 21
J. RESUME -- Restart a File where Left Off 22
K. DELETE -- Mark a File for Deletion. 23
L. CHKPT -- Take Checkpoint. 24
M. Others. 25

10. SUBROUTINES 25

A. GDIR and PDIR -- Get and Put I.D. Directory . . 25
B. RETRY -- Retry I/O. 26
C. DCHK -- Wait for Disk Directory 26
D. FSUM -- Form and Check Checksum in Work Area. . 26
E. HFIND -- Locate and Retrieve Record 0 27
F. DKINIT -- Initialize Disk Allocator 27
G. REDFIL and WRTFIL -- Read and Write Archive File 27

TABLE OF CONTENTS (cont.)

Page

H. PFMD -- Print Directory 29
I. FLRET and RLCCM -- Communication with CCM . . . 30

11. CONSOLE UTILITIES 32

A. INIT -- Initialize File Manager 33
B. NWID -- Establish new User I.D. from Console. . 34
C. RMID -- Remove I.D. No. from Console. 35
D. DLET -- Mark File for Deletion from Console . . 36
E. CRFL -- Create New File from Console. 37
F. RMOV -- Remove File from Console. 38
G. RLOT -- Roll out Files on Daily Archive Tape. . 39
H. RLIN -- Roll in Files from Daily Archive Tape . 40
I. SVFL -- Save File on User Archive Tape. 41
J. INDEX -- Produce Index of User of Daily Archive Tape 42
K. RSFL -- Restore File from User of Daily Archive Tape 43
L. DSPL -- Display File Manager Directory. 44
M. CLOS -- Close File from Console 45

12. OPERATING PROCEDURES. 46

A. Initializing. 46
B. User I.D.'s 46
C. Deleting Files. 46
D. Removing Files. 46
E. Closing Files 46
F. Archive Tapes and Reorganization. 47
G. File Manager Status and Indexing. 47

13. ACKNOWLEDGEMENTS. 48

14. LITERATURE. 48

APPENDIX A: ERROR CODES. 49

LIST OF FIGURES

Figure — Page

1. Disk Directory. 3

2. Work Area for Open Files. 4

3. Data Records. 5

4. PBLOCKS 13

5. PFMD Directory Printout 31

A FILE MANAGEMENT SYSTEM FOR A LABORATORY AUTOMATION FACILITY

The National Bureau of Standards' Analytical Chemistry Division operates a centralized laboratory automation facility built around a multiprogrammed minicomputer. A file manager was developed which allows the dynamic creation and manipulation of sequential disk files. Although the system was developed for real-time data acquisition, it is a general purpose addition to the computer's operating system and may be used for a variety of applications. A new operating system function was developed to allow the queued scheduling of programs. This is used to achieve more efficient multiprogramming. A comprehensive file utility package is also provided.

Key Words: Data acquisition; file system; laboratory automation; multiprogramming; operating system; real-time.

1. INTRODUCTION

The National Bureau of Standards' Analytical Chemistry Division operates a centralized laboratory automation facility, using a UNIVAC Series 60 computer, formerly known as the EMR 6135 (see 1). A number of instruments are connected to this computer. Each instrument acquires data in a series of "runs." Six or more instruments may acquire data on a given day, with some instruments performing thirty or more runs. This data must be stored on magnetic disk and catalogued by instrument and run. The file system described in this report was built to fulfill this need. However, it is a general purpose addition to ASSET IV (see 2), the computer's real-time operating system, and may be used for other purposes as well.

This package was designed to meet severe core constraints. This was achieved by development of a new operating system service to allow the queued scheduling of resident and non-resident programs. This service is also of general use.

This system controls disk files consisting of backward and forward linked lists of fixed length disk records. Each such record consists of an integral number of physical disk segments. The user may treat such files as if they were series of fixed length records on magnetic tape. The names of the "Actions" provided are suggestive of their functions -- CREATE, OPEN, CLOSE, READ, WRITE, RESUME, etc. The essential functions of most routines are to transfer data between disk and core and to maintain two types of table -- the Disk Directory and the Work Areas for Open Files.

A console utility package is provided which facilitates the transfer of file manager files to and from magnetic tape and the maintenance of hardcopy records.

2. DISK DIRECTORY

This directory (Figure 1) is disk resident and is organized around User I.D. (Identification) Numbers. It contains a "Record 0" entry for each currently existing file specifying File Names, Logical Unit, Date last used, No. times used, File Length, Record Length, etc. All files assigned to a single User I.D. No. must share the same logical unit. Generally, User I.D. Nos. are associated with instruments and files are associated with data acquisition runs. The "Record 0" entry for an individual file is so named because it is also the zeroth record, or header record, for that file.

3. WORK AREAS FOR OPEN FILES

When a file is opened, the user must supply a nine word work area (Figure 2). This area is subsequently used to maintain pointers and other information needed for transferring data (Figure 3) to and from the disk. This makes it unnecessary to refer to the disk directory except at the beginning and end of a series of file operations. A checksum is maintained in the work area to guard against overwrites.

4. GENERAL STRUCTURE

All user calls are to an executive routine called FMGR with the action desired indicated by an operation code. FMGR queues the calls and dispatches them to the action programs with the aid of two other executive modules, NXACT and XFIN. There is an action program for each allowable operation code.

FMGR, NXACT, and XFIN may be used with subroutine FMLINK to achieve queued scheduling of resident or nonresident programs. This makes it possible, when multiprogramming, to create multiple simultaneous instances of a nonreentrant nonresident program without keeping multiple copies in core.

5. FMGR

FMGR is the main resident scheduling routine for the file manager. The three resident modules FMGR, NXACT, and XFIN are strongly analogous to the three modules RIOS, FNR, and CRR which compose the executive portion of the EMR 6135 ASSET I/O (Input-Output) subsystem. FMGR is a reentrant system routine and resides below the ASSET FENCE boundary.

Figure 1. DISK DIRECTORY

I.D. Directory

I.D. No. (1)
Logical Unit
Addr. 1st File
I.D. No. (2)
Logical Unit
Addr. 1st File
1
1
1
1
1

Record 0

0	Forward Rec. Pointer (Initially -1)	
1	Backward Rec. Pointer (Always -1)	
2	Pointer to Last Rec. (Initially Rec. 0)	
3	Pointer to Next File (-1 for Last File)	
4 7	File Name (4 words)	
8	Logical Unit	
9 11	Date Created or last Used (3 words)	
12	Delete Flag 16	No. Times Opened
13	File Length (records)	
14	Rec. Length (words)	
15	Addr. of Work Area When File was Opened. (-1 if file not open)	
16	I.D. No.	
17 19	Not Used	
20 49	User (30 words) Label	

I.D. No. = -1 marks end

I.D. No. = 0 marks an empty slot

Addr. Pointer negative means no files on disk for that I.D. No.

Max. No. Segments in I.D. Directory in resident location

DSIZE

Logical Unit for I.D. Directory in resident location FLU

The Delete Flag (Bit 16, word 12) is set to mark a file for deletion when the system is next reorganized

Figure 2. WORK AREA FOR OPEN FILES

0	Busy Flag 16	Logical Unit
1	Pointer to record 0	
2	Pointer to current record	
3	Pointer to previous record	
4	Pointer to next record	
5	Pointer to last record	
6	File length (records)	
7	Record length (words)	
8	Check Sum (Sum words 0-8, modulo 16 bits)	

The busy flag is on when an action is in progress.

The check sum is checked at the start of each action and reformed at the end of the action, except that

The entire work area must be zeros going into OPEN or RESUME and the entire work area is set to zeros at the end of CLOSE.

Figure 3. DATA RECORDS

Word	
0	Forward Record Pointer (-1 on last record)
1	Backward Record Pointer
2	DATA
L-1	

L= Record Length. Fixed for any given file.
Always a positive multiple of 50.

A. Entry Points

FMGR has four entry points for user calls.

FMGR1 -- PBLOCK follows call; control returns after schedulling. PBLOCK preceeded by a pointer to NP (No. of parameters in PBLOCK)

FMGR2 -- PBLOCK immediately follows call; no return after scheduling.

FMGR3 -- PBLOCK address in index register 1, control returns after scheduling.

FMGR4 -- PBLOCK address in index register 1, no return after scheduling.

File Manager PBLOCKS are generally similar to RIOS PBLOCKS. They always contain a completion address and a completion priority. If there is no completion routine, FMGR should be furnished a completion address of 0.

B. Vectors, System Parameters, and Reserved Areas

FMGR maintains four vectors, each of which contains an entry for each allowable action type. These vectors are indexed by operation code. These three vectors are called FMQUE, OPWORD, NORD, and ADTAB.

FMQUE -- An Asset queue of PBLOCKS waiting for service from the corresponding action program. PBLOCKS are queued in priority order by ASSET routine QUEIT.

NORD -- Nonresident program number if action program is nonresident, meaningless if action program is resident.

OPWORD -- One bit flags.

Bit 16 - 1 if action uses disk directory, otherwise 0.

Bit 15 - 1 during period when nonresident action program scheduled but still nonresident, 0 at all other times.

Bit 14 - 1 if action program should remain permanently resident after being scheduled the first time. 0 if action program should be removed from core when no PBLOCKS are waiting.

Bit 13 - 1 if action program is in use, 0 otherwise. (action programs are assumed to be nonreentrant)

ADTAB -- Action program starting address while program is in core, 0 when program is not in core. The ADTAB entry should be initialized at SYSGEN time if the action program is permanently resident. Nonresident action programs plant there starting addresses here when they come in from the disk (Subroutine FMLINK).

Other Parameters and Reserved Areas

DFLAG -- 1 when the disk directory is in use, 0 otherwise. Only one action which uses the directory is allowed to proceed at a time. Action programs are responsible for waiting for and setting DFLAG. Subroutine DCHK is used for this purpose. DFLAG is turned off by XFIN when an action program using the directory has completed processing a call.

DSIZE -- Maximum number of 50 word disk segments in the I.D. directory. Used in determining size of dynamic buffer needed when reading I.D. directory into core (Subroutines GDIR and PDIR).

FLU -- Logical unit of I.D. directory.

PRI -- Priority used for loading nonresident action programs.

FLMAX -- Maximum permissible file length, in disk segments.

FRES, POOL, and LPOOL -- A pool and stack arrangement furnishing PBLOCKS to be used in loading nonresident action programs. These PBLOCKS are obtained by system subroutine GETP and returned by system subroutine PUSH.

C. Operation

The program starts off, at each of its four entry points, by initializing the index registers. XR 1 gets the PBLOCK address, XR 4 gets the interrupt stack address (PROP) and XR 2 gets a "return code" -- 0 for no return (FMGR2 and 4), 1 for direct return (FMGR2), and 2 for indirect return (FMGR3).

The four entry paths come together at location FMGRA. The PBLOCK busy bit is tested and set. The running priority, used in queueing the PBLOCK and scheduling action program execution is placed in word 1 of the PBLOCK. The value used is either 0 or RPL (ASSET running priority level) depending on whether the caller is below or above FENCE. The completion address, if there is one, is then checked against the completion priority.

At location OPCD, XR 4 gets the operation code obtained from the PBLOCK. At location FM10, a decision is made as to the disposition of the call. If the action program is in core (ADTAB entry not 0) and if the busy bit is off (OPWORD bit 13) the call will get immediate service and the busy bit is set. This is indicated by zeroing the E register. Otherwise the call will have to be queued for later service. In this case a PBLOCK may be needed to load the action program so one is obtained using system subroutine GETP. The address of this PBLOCK is stored in the E register. A positive value in the E register at this point indicates that the calling PBLOCK is to be queued.

At location T, the return to the calling program after scheduling is set up, if there is to be such a return. Action taken depends on the return code in XR2. If no return is called for, the user is removed from the stack. If a direct (PBLOCK in line) return is called for, the return address is computed from the NP (no. of parameters) value furnished and is placed on the stack. If an indirect return (PBLOCK address in XR1 originally) is called for we simply add one to the address already on the stack. If any return is called for, AMPN (active middleground program number) and RPL values are placed on the stack as required by ASSET protocol.

At location T30, the E register is tested to see if we are to process this call directly or queue it. If the E register is zero, the call gets directly processed. The ADTAB value is placed in word 3 of the calling PBLOCK which is then used to schedule entry to the action program via SRQUE. On the other hand, if the E register is not 0, the calling PBLOCK is placed on the appropriate FMQUE queue by QUEIT. If the action program is not in core (ADTAB = 0) and if it has not been scheduled for loading (bit 15 of OPWORD = 0) its loading is scheduled via RMAS4 using the PBLOCK whose address was in the E register. In this case bit 15 of OPWORD is also set. If the action program is already in core or already has been scheduled, the PBLOCK whose address was in the E register will not be needed and is released via system subroutine PUSH.

D. Error Conditions

FMGR produces console error messages via system subroutine ALM3 with the PBLOCK address typed out.

ERROR 061 -- Calling PBLOCK busy. Control goes to Job Control.

ERROR 062 -- Caller below FENCE has furnished completion address above FENCE with priority 0. Control goes to Job Control.

ERROR 063 -- Completion address below FENCE furnished with nonzero priority. Processing proceeds with priority changed to zero.

ERROR 064 -- No PBLOCKS available for scheduling loading of action program. Control goes to Job Control.

E. Batch Communication with FMGR

Since programs running under control of the Batch Monitor may not directly call programs in foreground (like FMGR), special procedures are required for file manager operations within such programs. Two routines are involved; FMGR2 and FMBPC.

A batch program performs file manager operations by issuing a call to FMGR2 using the same calling sequences as are used in calling the foreground version. However, the call to FMGR2 from the batch has the effect of invoking the batch library routine FMGR2 which is not the same as the routine reached through the foreground entry point FMGR2. FMGR2 copies the PBLOCK along with the contents of the locations pointed to by addresses in the PBLOCK into the areas FMBPPB and FMBA. This copying is directed by a table in FMGR2 which is indexed by operation code. FMGR2 calls FMBC via an SMM (Store Monitor Master) trap. FMBC then calls FMGR to begin the actual file operation. Return is made to FMBC which then passes control back to FMGR2. FMGR2 then copies the information from FMBPPB and FMBA back into the calling program and returns to the calling program.

The routine FMBC is in the protected "resident batch area." This is the portion of memory occupied by batch processor routines throughout the entire period when the batch monitor is active, as opposed to that area which is used by the batch monitor but may be checkpointed when foreground jobs need core.

The resident batch routine)INIT was modified to unprotect areas FMBPPB and FMBA so that information could be passed in from the unprotected batch area.

The size of area FMBA presently places a restriction to 500 words on the maximum record length available when accessing the file manager from the batch monitor.

Status codes returned from batch calls to FMGR2 have the same meanings as in foreground operation with one exception;

a code of -16 means that there was an error in the move-directing table in FMGR2 which would have resulted in moving more data that could be accommodated in FMBA.

6. XFIN

XFIN is the resident completion routine called by all action programs after completing processing for a given call. It is analagous to CRR, which is the completion routine in the I/O subsystem. XFIN is a reentrant system routine and resides below the FENCE boundary. It is entered from an action program with the PBLOCK address in XR1, the operation code in XR4, and the completion status in the E register. Its functions are:

1. DFLAG is turned off, if OPWORD bit 16 indicates that the completing action uses the disk directory.

2. The completion status is placed in the PBLOCK and the PBLOCK threading cell is zeroed.

3. The completion address and priority are set up to schedule the completion routine if the completion address is not 0. If the completion priority is zero, the priority in word 1 of the PBLOCK is left unchanged. If the completion priority is not zero and the completion address is below FENCE the priority in word 1 is zeroed. Otherwise, the completion priority replaces the priority in word 1.

4. The completion routine, if there is one, is scheduled via SRQUE.

XFIN terminates by jumping to NXACT which initiates processing of the next call on the queue.

7. NXACT

NXACT is the resident routine which initiates processing of the next call on queue for any action or terminates processing if the queue is empty. It is analogous to the FNR (Find Next Request) program in the I/O subsystem. It is called from XFIN and at the beginning of nonresident action programs when they are first loaded in core. It is called with the appropriate operation code in XR4. NXACT is a reentrant system routine and resides below the FENCE boundary. Its functions are:

1. FMQUE is checked. If the queue is not empty, the top PBLOCK is removed. The action program address is obtained from ADTAB and placed in word 3 of the PBLOCK. The action program is then scheduled via SRQUE.

2. If the queue is empty, the busy bit (OPWORD bit 13) is turned off. Bit 14 of OPWORD is then tested to see whether or not the action program should be removed from core. If removal is indicated, the core formerly held by the action program is released via PUT and the ADTAB entry is zeroed.

NXACT terminates by calling JC (Job Control).

8. NBS DISK ALLOCATOR

This program is used by the file manager to obtain and return disk space. Calling sequences are as follows:

To obtain disk space

```
CALL QUEDRM(0,0,PRI1,GETDRM,LU,WC,DADD,PRI2)  where
```

PRI1 ignored
GETDRM is an external symbol.
LU = disk logical unit
WC = number of words desired
DADD = disk address, on return (1 word).
PRI2 ignored

To release disk space

```
CALL QUEDRM(0,0,PRI1,RTNDRM,LU,WC,DADD,PRI2)  where
```

PRI1, LU, WC, and PRI2 are as above
RTNDRM is an external symbol
DADD = address of disk area to be released.

The reason for the ignored calling sequence parameters is that this is a compatible replacement for an earlier package. In both cases, a negative DADD value is returned to indicate failures.

9. ACTION PROGRAMS

Action programs may be either resident or nonresident. A resident action program is loaded above FENCE at SYSGEN time with its starting address in ADTAB. Only those actions which are used very frequently (READ and WRITE) should be made resident. Nonresident action programs plant their addresses in ADTAB and call NXACT to process waiting calls when they first arrive in core. Setting bit 14 of OPWORD at SYSGEN will cause a nonresident action program to remain permanently resident after it is loaded.

A. Linkage and Conventions

A resident action program whose starting address was labeled READ would begin execution with

```
      ENT  READ

READ  JMP  *+1
```

If the same program were made nonresident and had operation code 07 it would begin like this

```
OC    VAL  07

INIT  CALL FMLINK(READ,OC)

READ  JMP  *+1
       .
       .
       .
     END   INIT
```

When the program was loaded, subroutine FMLINK would be called. FMLINK first returns the PBLOCK used in loading the program to stack FRES by means of system subroutine PUSH. It then plants the starting address, in this case READ, in the correct entry of ADTAB, as determined by OC. It turns off the scheduling bit (bit 15) in OPWORD and calls NXACT to initiate processing of the calls waiting on queue. In this case, the action program would be entered at location READ each time a call is to be processed.

Action programs expect XR1 to contain the PBLOCK address when entered at the starting address. Action programs generally use XR1 to point to the PBLOCK and XR4 to point to the Open Files Work Area internally. Action programs use the priority obtained from word 1 of the calling PBLOCK to schedule I/O and disk allocator operations. Action programs using the disk directory call subroutine DCHK to delay execution until the directory is available and to set DFLAG to indicate that the directory is tied up. All action programs terminate by calling XFIN with the PBLOCK address in XR1, the operation code in XR4 and the completion status in the E register.

Figure 4. PBLOCKS

Word	NEWID	RMVID	CREATE	REMOVE	OPEN	CLOSE	WRITE	READ	RESUME	DELETE	CHKPT
0					system use						
1					system use						
2				completion priority							
3					system use						
4				completion address							
*5	00	01	02	03	04	05	06	07	08	09	11
6				completion status							
7					work area	work area	work area	work area	work area		work area
8	I.D. No.	I.D. No.	I.D. No.	I.D. No.	I.D. No.	BUFFER	BUFFER	BUFFER	I.D. No.	I.D. No.	
9	Logical Unit	Logical Unit	NAME	NAME	NAME				NAME	NAME	
10			DATE		DATE				DATE		
11			LABEL		BUFFER				BUFFER		
12			REC. LENGTH								

* Word 5 always contains op. code

NAME is always a 4 word field
DATE is always a 3 word field
LABEL is always a 30 word field
BUFFER should have length = REC. LENGTH

Typical FORTRAN Calls

```
   CALL FMGR2(0,0,PRI,0,$100,02,STAT,0,ID,NAME,DATE,LABEL,WC)
$100 SCHENT                                           TO CREATE
   CALL FMGR2(0,0,PRI,0,$200,04,STAT,WA,ID,NAME,DATE,BUFFER)
$200 SCHENT                                           TO OPEN
   CALL FMGR2(0,0,PRI,0,$300,06,STAT,WA,BUFFER)       TO WRITE
$300 SCHENT
   CALL FMGR2(0,0,PRI,$400,05,STAT,WA,BUFFER)         TO CLOSE
$400 SCHENT
```

ACTION DESCRIPTION

B. NAME: NEWID

TYPICAL CALL: CALL FMGR2 (0,0,PRI,0,CA,00,STAT,0,ID,LU)

FUNCTION: Establishes new entry in the I.D. Directory.

Searches I.D. Directory for a vacant slot and establishes new entry using the I.D. No. (ID) and Logical Unit (LU) furnished.

COMPLETION STATUS CODES:

+1 - Success
-1 - I/O Error
-4 - No room in I.D. Directory
-5 - Duplicate I.D. No.
-9 - Illegal (Negative) I.D. No.

ACTION DESCRIPTION

C. NAME: RMVID

TYPICAL CALL: CALL FMGR2 (0,0,0,0,CA,01,STAT,0,I.D.)

FUNCTION: Searches directory for ID and removes ID by writing zeros in the 3 word directory entry.

COMPLETION STATUS CODES:

```
 -1 = IOST - I/O Error
 -6 = MSST - ID NOT IN DIR.
-10 = FEST - FILES EXIST
 +1 = OKST - OK STATUS
```

D. NAME: CREATE

TYPICAL CALL: CALL FMGR2 (0,0,PRI,0,CA,02,0,ID,NAME,DATE, LABEL,WC)

FUNCTION: Establishes a new Record 0 (Header) entry in the Disk Directory.

The I.D. Directory is read and searched for a matching I.D. No. The list of files attached to that I.D. No. is followed to the end. The disk allocator is called to obtain a new 50 word segment and the new Record 0 is built in that segment with values as follows:

Forward Record Pointer = -1
Backward Record Pointer = -1
Last Record Pointer → Record 0
Forward File Pointer = -1
Name - from PBLOCK
Logical Unit - from I.D. Directory
Date - from PBLOCK, ignored if 1st word = 0
No. Times Opened = 0
Delete Flag = off
Length = 0
Word Count (Rec. Length) - taken from PBLOCK (WC); rounded to high multiple of 50 with a minimum of 50.
Pointer to Work Area = -1
I.D. - from PBLOCK
Label - from PBLOCK, ignored if 1st word = 0

The new Record 0 entry is written out. The proceeding Record 0 entry has its Forward File Pointer updated and is rewritten. If this is the 1st file for this I.D. No., the disk pointer in the I.D. Directory is updated and the I.D. Directory is rewritten.

COMPLETION STATUS CODES:

+1 - Success
-1 - I/O Error
-4 - Disk Allocator Error
-5 - Duplicate Name within I.D. No.
-6 - I.D. No. not in Directory

E. NAME: REMOVE

TYPICAL CALL: CALL FMGR2 (O,O,PRI,O,CA,03,STAT,O,ID, NAME)

FUNCTION: Removes a Record 0 entry from the disk directory and deallocates all disk assigned to the file.

The I.D. Directory is read and a matching entry found. The list of Record 0 entries attached is searched using 2 alternating internal buffers. When a match is found, both the matching Record 0 and its predecessor are in core.

If the matching Record 0 is not first on the list, its Forward Record Pointer replaces that of its predecessor and the predecessor is rewritten. If the matching Record 0 is first on the list, its Forward Record Pointer replaces the disk pointer in the I.D. Directory and the I.D. Directory is rewritten.

Deallocation, via the disk allocator, starts with Record 0 and proceeds down the file until one of the following conditions is met:

a) Forward Record Pointer negative.

b) Last Record (as indicated by Record 0) deallocated.

c) No. of records deallocated = File Length (as indicated by Record 0).

COMPLETION STATUS CODES:

+1 - Success
-1 - I/O Error
-6 - Couldn't find matching I.D. No. and Name
-7 - File Open

F. NAME: OPEN

TYPICAL CALL: CALL FMGR2 (0,0,PRI,0,CA,04,STAT,WA,ID, NAME,DATE,BUFFER)

FUNCTION: Builds an Open Files Work Area in User Provided Area (WA)

The matching Record 0 entry is found and read into BUFFER. If there are no data records on the file, the disk allocator is called to reserve disk space for the first data record. The Work Area is set up with values as follows:

Logical Unit - from Record 0
Pointer to Record 0 → Record 0
Pointer to Current Record → Record 0
Pointer to Previous Record = -1
Pointer to Next Record - from Record 0 or Disk Allocator
Pointer to Last Record - from Record 0
File Length - from Record 0
Record Length - from Record 0

Record 0 is updated as follows:

Date - from PBLOCK, ignored if 1st word = 0
No. Times Opened - Incremented by 1
* Pointer to Work Area → Work Area
Forward Record Pointer - from Disk Allocator if previously -1, left alone otherwise

COMPLETION STATUS CODES:

+1 - Success
-1 - I/O Error
-2 - Work Area not initialized to all zeros
-4 - Disk Allocator Error
-6 - Couldn't find matching Name & I.D. No., could also result from an I/O error in the search
-7 - File already open.

* No essential use is made of this pointer. The Work Area may move around in core between an OPEN and a CLOSE.

ACTION DESCRIPTION

G. NAME: CLOSE

TYPICAL CALL: CALL FMGR2 (0,0,PRI,0,CA,05,STAT,WA, BUFFER)

FUNCTION: To terminate the file at its "last record" and to update Record 0 from the Work Area (WA). At completion, WA will be all zeros and BUFFER will contain Record 0.

The "last record", as indicated in the Work Area (WA) is read into BUFFER. If the Forward Record Pointer is not negative, the record pointed to is released by the disk allocator and the "last record" is rewritten with its Forward Record Pointer changed to -1.

Record 0 is read into BUFFER and modified as follows:

Last Record Pointer - from Work Area Pointer to Last Record

Length - from Work Area

Pointer to Work Area = -1 (this indicates that file is closed)

Finally, Record 0 is rewritten and the Work Area is cleared.

COMPLETION STATUS CODES:

+1 - Success
-1 - I/O Error
-2 - Work Area busy or has incorrect check sum

ACTION DESCRIPTION

H. NAME: WRITE

TYPICAL CALL: CALL FMGR2 (0,0,PRI,0,CA,06,STAT,WA, BUFFER)

FUNCTION: Contents of BUFFER are written onto next record.

A new record is obtained from the disk allocator. The contents of BUFFER are written onto the record indicated by the Work Area (WA) "Pointer to Next Record." The pointers in the record written are as follows:

Backward Record Pointer - from Work Area pointer to Current Record

Forward Record Pointer - from disk allocator

The Work Area is modified as follows:

Pointer to Current Record = former Pointer to Next Record.

Pointer to Previous Record = former Pointer to Current Record.

Pointer to Next Record - from disk allocator

Pointer to Last Record = former Pointer to Next Record

COMPLETION STATUS CODES:

+1 - Success
-1 - I/O Error
-2 - Work Area busy or has incorrect check sum
-4 - Disk Allocator Error
-8 - Attempt to write with current record ≠ last record
-12 - Attempt to write when File Length has reached maximum allowable value.

ACTION DESCRIPTION

I. NAME: READ

TYPICAL CALL: CALL FMGR2 (0,0,PRI,0,CA,07,STAT,WA,
BUFFER)

FUNCTION: Reads next record into BUFFER.

The record indicated by the Work Area (WA) "Pointer to Next Record" is read into BUFFER. If this was the last record, one more record is obtained from the disk allocator and the Forward Record Pointer in the record just read is updated to point to this additional record. The Work Area is modified as follows:

Pointer to Current Record = former Pointer to Next Record

Pointer to Previous Record = former Pointer to Current Record

Pointer to Next Record = Forward Record Pointer from record read if this was not last.

= address obtained from disk allocator if record read was last.

COMPLETION STATUS CODES:

+1 - Success
+2 - Success, record just read was last record on file
-1 - I/O Error
-2 - Work Area busy or has incorrect check sum
-4 - Disk Allocator Error
-8 - Attempt to read past last record on file

ACTION DESCRIPTION

I. NAME: RESUME

TYPICAL CALL: CALL FMGR2 (0,0,PRI,0,CA,08,STAT,WA,ID, NAME,DATE,BUFFER)

FUNCTION: Builds on Open Files Work Area in WA, with pointers positioned at end of the file. At completion, BUFFER contains Record 0.

The matching Record 0 entry is found and read into an internal buffer. The last record is read into BUFFER. An additional record is obtained from the disk allocator. The Work Area is set up as follows:

Logical Unit - from Record 0
Pointer to Record 0 → Record 0
Pointer to Current Record → last record
Pointer to Previous Record - from Backward Record Pointer of last record
Pointer to Next Record - from disk allocator
Pointer to Last Record → last record
File Length - from Record 0
Record Length - from Record 0

Record 0 is updated as follows:

Date - from PBLOCK
No. Times Opened - Incremented by 1
Pointer to Work Area → Work Area

The last record is updated as follows:

Forward Record Pointer - from disk allocator

Finally Record 0 and the last record are rewritten and Record 0 is copied into BUFFER

If Record 0 = last record, the action resulting from RESUME is the same as from OPEN

COMPLETION STATUS CODES:

+1 - Success
-1 - I/O Error
-2 - Work Area not initialized to all zeros
-4 - Disk Allocator Error
-6 - Couldn't find matching Name and I.D. No., could also result from an I/O Error in the search.
-7 - File Already Open

ACTION DESCRIPTION

K. NAME: DELETE

TYPICAL CALL: CALL FMGR2(0,0,0,0,CA,09,STAT,0,ID,NAME)

FUNCTION: Finds Record 0, deletes file by setting bit 16 of word 12 and rewrites Record 0.

COMPLETION STATUS CODES:

+1 - Success
-1 - I/O Error
-3 - File Already Deleted
-6 - No such ID in Directory

ACTION DESCRIPTION

L. NAME: CHKPT

TYPICAL CALL: CALL FMGR2(0,0,PRI,0,CA,11,STAT,WA)

FUNCTION: Updates the last record indications in Record 0 from the Work Area to establish a checkpoint. If system is interrupted after a CHKPT operation with the file open and later restored, the restored file will be truncated at the point of the CHKPT.

Record 0 is read into an internal buffer and changed as follows:

Last Record Pointer - from Work Area Pointer to Last Record

Length - from Work Area

Record 0 is then rewritten.

COMPLETION STATUS CODES:

+1 - Success
-1 - I/O Error
-2 - Work Area busy or has incorrect check sum

M. Others

File Manager's flexible design makes it easy to define and build in additional action types. Such additions currently contemplated include:

REWRITE. Writes a record over an existing record with work area pointers positioned somewhere other than at the end of the file.

BACKSPACE. Repositions work area pointers one record behind their current position.

REWIND. Repositions work area pointers at beginning of file. This would eliminate need of an extra OPEN and CLOSE in some applications.

SKIP-TO-END. Repositions work area pointers at end of file. This would eliminate need of an extra CLOSE and RESUME in some applications.

TRUNCATE. Closes a file with the current position of the work area pointers becoming the end position. All succeeding records would be discarded.

EXCISE. Discards a single record at the current work area pointer position, linking up the records at the previous and next positions.

10. SUBROUTINES

Subroutine FMLINK is described in the section on action program linkage and conventions. Other subroutines written specifically for the use of File Manager Action Programs and console utilities are described below.

A. GDIR and PDIR

GDIR is a nonreentrant library subroutine which obtains a dynamic buffer and reads the I.D. Directory into it. The calling sequence is:

```
CALL  GDIR(DLOC,MAX)
```

On return, DLOC will contain the address of the buffer containing the I.D. Directory and MAX will contain the number of words in the Directory - 1. This is useful in setting up a loop to search the Directory. The A register will contain the I/O status on return. The value of MAX and the size of the buffer are computed from system parameter DSIZE.

PDIR is a nonreentrant library subroutine which rewrites the I.D. Directory from a dynamic buffer and releases the buffer. The calling sequence is:

```
CALL PDIR(DLOC)
```

On call, DLOC should contain the buffer address. On return, the A register will contain the I/O status.

B. RETRY

RETRY is a nonreentrant library subroutine used to retry an aberrant I/O operation up to ten times before giving up and going to an error routine. The calling sequence is:

```
CALL  RETRY(STAT,IO,ERR)
```

If STAT is nonnegative, control proceeds to the next instruction. If STAT is negative, control goes to location IO the first nine times RETRY is called and to ERR the tenth time. RETRY maintains its own counter which is reset on a nonnegative status and before jumping to location ERR.

C. DCHK

DCHK is a nonreentrant library subroutine used to delay execution until DFLAG = 0 (Directory free) and then to set DFLAG = 1 (Directory busy). It operates by repeatedly calling library subroutine DELAY. The calling sequence is:

```
CALL DCHK
```

D. FSUM

FSUM is a reentrant subroutine used to form and check the checksum in an open files work area. It is normally called at the beginning and end of action programs. The calling sequence is:

```
CALL  FSUM
```

On call, the work area address should be in XR4. The sum in word eight of the work area is saved. A new sum of words zero through seven is formed, ignoring overflow, and stored in word eight. The old sum and new sum are compared. If they are the same, a +1 is returned in the A register. If they are different, a -1 is returned in the A register.

FSUM is resident and reentrant.

E. HFIND

Subroutine HFIND is used to locate and retrieve Record 0 of a file, given the file I.D. No. and Name. The calling sequence is:

```
CALL  HFIND(ID,NAME,LU,AD,BUF)
```

ID = I.D. No. of file on call.

NAME = Name of File on call.

LU = Logical Unit on which file is located on return.

AD = Disk address of Record 0 on return.

BUF should be a 50 word buffer which will contain Record 0 on return.

HFIND operates by first searching the I.D. Directory and then searching the list of Record 0 entries attached to the matching I.D. No. It returns a status in the A register:

A reg. = +1 means success

-1 means no matching I.D. No.

-2 means no matching Name for this I.D. No.

-3 means file deleted. In this case all parameters are returned normally.

-4 means I/O error.

F. DKINIT -- Initialize Disk Allocator

Subroutine DKINIT is used to force a null condition on the NBS Disk Allocator's core resident tables when initializing or reorganizing the file manager. The calling sequence is:

```
CALL  DKINIT
```

G. RDFIL and WRTFIL -- Read and Write Archive File

An archive file is a magnetic tape containing copies of a variable number of File Manager files. Such tapes begin with a fifty word volume header label, followed by an E.O.F., and end with a fifty word volume trailer label. A detailed description of these labels is available in the documentation for the NBS LABGEN program. Each File Manager file copy consists of a header record, a variable number of data blocks, and finally an E.O.F. The header record is fifty words long and contains a copy of Record 0. The data block length is the least multiple of the file's record length which is greater than or equal to 1000. Each data block is filled with copies of File Manager records. However, the last data block may be only partially filled. The forward record pointer of the last record in each data block is set to -1 and the forward record pointer of all other records is set to +1.

Subroutine WRTFIL is used to transfer a file from the File Manager to an already positioned archive tape. The calling sequence is:

```
      CALL  WRTFIL(IFLMAX,ID,LU,LUTAP,IRECZ,ISTAT)
```

where

IFLMAX = Maximum number of disk segments to be transferred.

ID = I.D. number for file.

LU = Logical Unit for file.

LUTAP = Logical Unit for tape drive.

IRECZ = Record 0

ISTAT = Status on return

The file is transferred with transmission being terminated and an E.O.F. written when IFLMAX is exceeded or when any of the following end of file conditions is recognized:

i. File Length specified in Record 0 reached.

ii. Last record reached, as specified in Record 0.

iii. Record reached with forward pointer negative.

Status codes are returned in ISTAT with the following meanings:

+1 - transfer completely successful

-1 - tape I/O error

-2 - file contains no data; Record 0 and an E.O.F. are written on tape.

-3 - Word Count specified in Record 0 greater than 1000.

-4 - out of range disk address

-5 - IFLMAX exceeded, an E.O.F. is written

-6 - end of file conditions inconsistent, an E.O.F. is written.

Subroutine REDFIL is used to transfer a file from a positioned archive tape to the File Manager. The calling sequence is

```
CALL REDFIL(LUTAP,IRECZ,ISTAT,IFMRS)    where
```

LUTAP = Logical Unit of tape drive.

IRECZ = Record 0.

ISTAT = status on return.

IFMRS = File Manager Status on return if there is a File Manager error.

The file is transferred with transmission normally terminating when an E.O.F. is reached on the tape. Status codes are returned in ISTAT with the following meanings:

+1 - transfer completely successful.

-1 - tape I/O error.

-2 - File Manager error; IFMRS will contain File Manager status code.

-3 - Word count in Record 0 greater than 1000.

-4 - Delete flag set in Record 0, transfer not effected.

-5 - Reached end of block without finding record with forward pointer = -1.

H. PFMD -- Print Directory

This subroutine is called by console utility program DSPL to print out formatted Record 0 images. The calling sequence is

```
CALL  PFMD(PENT)
```

where PENT is an I.D. directory entry in core. PFMD will print Record 0 for each file associated with the I.D. Number referenced on logical unit 14. Sample output is shown in Figure 5. If no files are present for that I.D. number, the message

```
NO FILES FOR THIS ID
```

is printed.

If an I/O error occurs the message

```
ERROR 172  177764
```

is typed on the console teletype. When Record 0 is read in from disk, the backward pointer is checked, since it should always be -1. If this is not the case the message

```
RECORD ERROR...FILE POINTERS ARE INVALID
```

is printed.

I. FLRET and RLCCM -- Communication with CCM

These two subroutines are used by console utility programs operating under control of the Command Control Monitor (CCM).

RLCCM is used to reset the CCM busy flag (CCMBSY), which has been made external, without returning to CCM. This allows the operator to intitiate further CCM actions from the console teletype during long file searches, printouts, etc. The calling sequence is

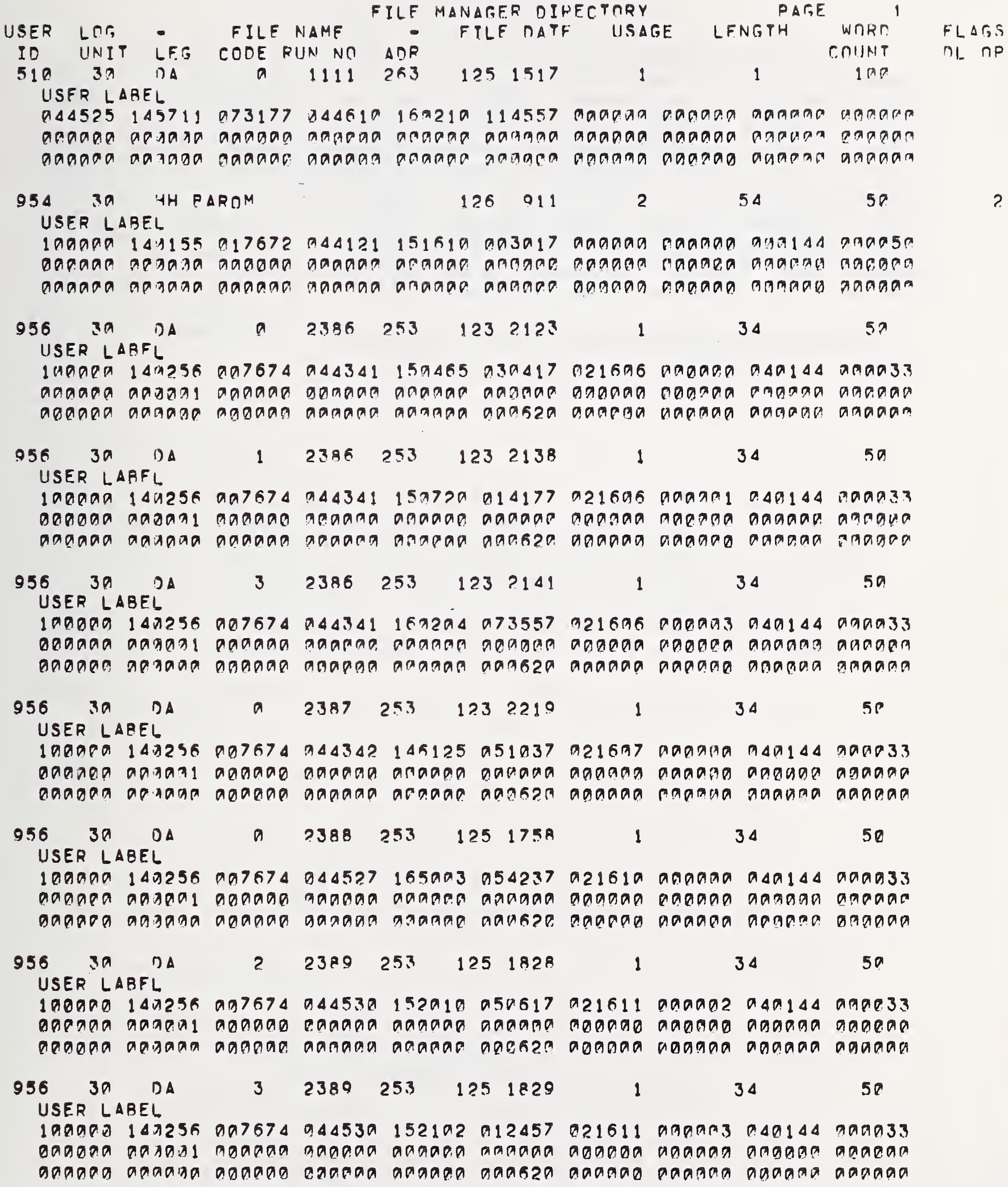
```
CALL RLCCM
```

FLRET is used to return to CCM from a File Manager console utility. The calling sequence is

CALL FLRET with a status code in the A register.

If the status is negative, ERROR 172 is typed with the status. Control then goes to CCMRET. CCM will type OK if the status was positive.

Figure 5. PFMD Directory Printout



```
                                   FILE MANAGER DIRECTORY           PAGE      1
USER  LOG    -      FILE NAME      -   FILE DATE   USAGE   LENGTH    WORD    FLAGS
 ID   UNIT  LEG  CODE RUN NO   ADR                                   COUNT   DL OP
 510   30   DA      0   1111   263    125 1517      1        1        100
  USFR LABEL
  044525 145711 073177 044610 160210 114557 000000 000000 000000 000000
  000000 000030 000000 000000 000000 000000 000000 000000 000000 000000
  000000 000000 000000 000000 000000 000000 000000 000000 000000 000000

 954   30   HH PAROM                 126  911       2       54        50        2
  USER LABEL
  100000 140155 017672 044121 151610 003017 000000 000000 000144 000050
  000000 000030 000000 000000 000000 000000 000000 000000 000000 000000
  000000 000000 000000 000000 000000 000000 000000 000000 000000 000000

 956   30   DA      0   2386   253    123 2123      1       34        50
  USER LABFL
  100000 140256 007674 044341 150465 030417 021606 000000 040144 000033
  000000 000031 000000 000000 000000 000000 000000 000000 000000 000000
  000000 000000 000000 000000 000000 000620 000000 000000 000000 000000

 956   30   DA      1   2386   253    123 2138      1       34        50
  USER LABFL
  100000 140256 007674 044341 150720 014177 021606 000001 040144 000033
  000000 000031 000000 000000 000000 000000 000000 000000 000000 000000
  000000 000000 000000 000000 000000 000620 000000 000000 000000 000000

 956   30   DA      3   2386   253    123 2141      1       34        50
  USER LABEL
  100000 140256 007674 044341 160204 073557 021606 000003 040144 000033
  000000 000031 000000 000000 000000 000000 000000 000000 000000 000000
  000000 000000 000000 000000 000000 000620 000000 000000 000000 000000

 956   30   DA      0   2387   253    123 2219      1       34        50
  USER LABEL
  100000 140256 007674 044342 146125 051037 021607 000000 040144 000033
  000000 000031 000000 000000 000000 000000 000000 000000 000000 000000
  000000 000000 000000 000000 000000 000620 000000 000000 000000 000000

 956   30   DA      0   2388   253    125 1758      1       34        50
  USER LABEL
  100000 140256 007674 044527 165003 054237 021610 000000 040144 000033
  000000 000001 000000 000000 000000 000000 000000 000000 000000 000000
  000000 000000 000000 000000 000000 000620 000000 000000 000000 000000

 956   30   DA      2   2389   253    125 1828      1       34        50
  USER LABFL
  100000 140256 007674 044530 152010 050617 021611 000002 040144 000033
  000000 000001 000000 000000 000000 000000 000000 000000 000000 000000
  000000 000000 000000 000000 000000 000620 000000 000000 000000 000000

 956   30   DA      3   2389   253    125 1829      1       34        50
  USER LABEL
  100000 140256 007674 044530 152102 012457 021611 000003 040144 000033
  000000 000031 000000 000000 000000 000000 000000 000000 000000 000000
  000000 000000 000000 000000 000000 000620 000000 000000 000000 000000
```

11. CONSOLE UTILITIES

All file manager utilities are accessed via CCM (Command Control Monitor), and run at priority 13. Utilities having a long execution time release CCM by calling RLCCM (Release CCM) after console inputs, and type descriptive messages when finished. All error diagnostics are produced through the utility. Shorter utilities terminate by calling FLRET (file return) and passing the status code in the "A" register, at which time an error diagnostic is produced or an "OK" status is returned to CCM. Both methods use the following format for error output: ERROR XXX - YYYYYY; where XXX = error number and YYYYYY = Status code in octal (See listing attached for definitions).

Brief descriptions of individual modules follows.

CONSOLE UTILITY DESCRIPTION

NAME -- INIT

FUNCTION -- Initializes file manager directory and disk allocator

INPUT PARAMETERS -- 1. File logical unit (FLU). Used for file manager directory as defined in SYSGEN listing. (Format I2) Range greater than 19, less than 66.

2. Disk size (DSIZE). Number of segments used for directory. (Format 12). Range greater than 0, less than 11.

OPERATION -- Requests and validates console inputs described above. Stores logical unit number in resident location (FLU) and writes -1 in word one of directory. Stores disk size in resident location DSIZE. Calls DKINIT, which initializes disk allocator and terminates calling FLRET.

```
LOGICAL UNITS -- TTY input = 20
                 FLU       - No. input
```

Error CODES -- None

```
STATUS CODES -- +1 = OK
               -12 = I/O Failure
               -11 = Parameter error
```

CONSOLE UTILITY DESCRIPTION

NAME -- NWID

FUNCTION -- Creates new user ID in file manager directory.

INPUT PARAMETERS -- 1. User ID in decimal. (Format I4).

2. User logical unit. (Format I2).

OPERATION -- Requests and validates inputs described above. Calls FMGR action module NEWID, which writes entry in directory. Loads status from FMGR and terminates by calling FLRET.

LOGICAL UNITS -- TTY input = 20

Error CODES -- None

STATUS CODES -- +1 = OK
-12 = I/O Failure
-11 = Parameter error
Also return error status codes from FMGR (see listing)

CONSOLE UTILITY DESCRIPTION

NAME -- RMID

FUNCTION -- Removes user ID from directory

INPUT PARAMETERS -- 1. User ID in decimal (Format I4).

OPERATION -- Requests user ID and calls FMGR module RMVID (01). Loads return status code and calls FLRET.

LOGICAL UNITS -- TTY Input = 20

Error CODES -- None

STATUS CODES -- +1 = OK
-12 = I/O Failure
Return status from FMGR (See listing)

CONSOLE UTILITY DESCRIPTION

NAME -- DLET

FUNCTION -- Marks delete flag (Bit 16 of word 12 in record zero) for future file delection.

```
INPUT PARAMETERS -- 1. User ID of file to be deleted (Format
                       I4).
                    2. Legend (DA,HH,$$), Format A2).
                    3. Optional, depending on legend:
                       DA: Code (Format I5)
                           Run. No. (Format I5)
                           Lab Addr. (Format O3)
                       HH: Alpha name (Format A2,A2,A2)
                       $$: Octal name (Format O6,O6,O6)
                    4. CODE = -1 (for termination)
```

OPERATION -- Requests file name using the above inputs. Calls FMGR action module DELETE (09). Loads return status code and terminates calling FLRET.

```
LOGICAL UNITS -- TTY input  = 20
                 TTY output = 20
```

Error CODES -- None

```
STATUS CODES -- +1 = OK
               -12 = I/O Failure
               Also, return error status codes from FMGR (See
               lising)
```

CONSOLE UTILITY DESCRIPTION

NAME -- CRFL

FUNCTION -- Creates record zero of new file from console.

```
INPUT PARAMETERS -- 1. User ID of file to be created (Format
                       I4).
                    2. Legend (DA,HH,$$) (Format A2).
                    3. Optional, depending on legend:
                       DA: Code (Format I5)
                           Run. No. (Format I5)
                           Lab Addr. (Format O3)
                       HH: Alpha name (Format A2,A2,A2)
                       $$: Octal name (Format O6,O6,O6)
                    4. User label (Format 30A2).
                    5. Record length (Format I4).
```

OPERATION -- Requests above inputs and calls GDB for file date. Calls FMGR action module CREATE (02), which creates record zero of new file. Loads return status code from FMGR and terminates calling FLRET.

```
LOGICAL UNITS -- TTY input  = 20
                 TTY output = 20
```

Error CODES -- None

```
STATUS CODES -- +1 = OK
               -12 = I/O Failure
               Also, return error status codes from FMGR
               (See listing).
```

CONSOLE UTILITY DESCRIPTION

NAME -- RMOV

FUNCTION -- Removes file from FMGR by deallocating disk.

```
INPUT PARAMETERS -- 1. User ID (Format I4)
                    2. Legend (DA,HH,$$) (Format A2)
                    3. Optional, depending on legend:
                       DA: Code (Format I5)
                           Run No. (Format I5)
                           Lab Addr. (Format 03)
                       HH: Alpha name (Format A2,A2,A2)
                       $$: Octal name (Format 06,06,06)
```

OPERATION -- Request above inputs and calls RLCCM to release CCM. Calls FMGR action module REMOVE (03), which removes file. Outputs error message when necessary via ALM3, types "RMOV FIN" and calls FIN.

```
LOGICAL UNITS -- TTY input   = 20
                 TTY output  = 20
                 ALM3 output = AO(25)
```

Error CODES -- 172

STATUS CODES -- -12 = I/O Failure
Also, return error status codes from FMGR. (See listing)

CONSOLE UTILITY DESCRIPTION

NAME -- RLOT

FUNCTION -- Rolls out file manager files on daily archive.

INPUT PARAMETERS -- 1. Current date (Format 9A2).

OPERATION -- Requests current date. Releases CCM. Positions and writes label on daily archive. Calls GDIR to get directory, calls WRTFIL to write file on archive. Creates a record of ID's where no files exist. Writes trailer label, rewinds, releases directory from dynamic core and terminates, typing "RLOT FIN" and calling FIN. Also types descriptive error messages on line printer.

LOGICAL UNITS -- TTY input = 20
TTY output = 20
Archive LU = 7
LPR error msg. = 14

Error CODES -- 172

STATUS CODES -- -12 I/O Failure
Also, return status codes from WRTFIL (See listing ERROR 173)

CONSOLE UTILITY DESCRIPTION

NAME -- RLIN

FUNCTION -- Rolls in daily archive.

INPUT PARAMETERS -- None

OPERATION -- Reads daily archive header label and outputs contents on console TTY. Calls REDFIL which writes files from daily archive to file manager disk. Creates file manager directory entry for ID's with no files. Loads status code in "A" register and terminates calling FLRET. Also, outputs descriptive error messages on line printer.

LOGICAL UNITS -- Daily Archive LU = 7
TTY output = 20
LPR error msg. = 14

Error CODES -- None

STATUS CODES -- -12 = I/O Failure
+1 = OK
Also, return status codes from REDFIL
(See listing ERRORS 174)

CONSOLE UTILITY DESCRIPTION

NAME -- SVFL

FUNCTION -- Saves files on user archive.

INPUT PARAMETERS -- 1. User ID (Format I4)
2. Legend: DA,HH,$$ (Format A2)
3. Mode?: Manual = -1
Auto = any char.
Term = -2
4. Optional, depending on legend:
DA: Code (Format I5)
Run No. (Format I5)
Lab Addr. (Format O3)
HH: Alpha name (A2,A2,A2)
$$: Octal name (Format O6,O6,O6)
5. Delete option (manual only)
"NO" (A2)
"Any char"

OPERATION -- Manual Mode -- Requests above inputs and releases CCM. Validates user archive with ID input. Reads user archive until "end of archive" (trailer label) has been found. Finds file in directory and writes on archive using WRTFIL. Returns to "MODE?" for next file/s to be saved or terminates request. Terminates by rewriting trailer label at end of archive, releases core held by directory, types "SVFL FIN" and calls FIN.

AUTO MODE -- Validates archive and ID, as above. Requests "Run No. (Low Range)" and "Run No. (Hi Range)". Saves and deletes all files, then returns for next range or terminates.

LOGICAL UNITS -- TTY input = 20
TTY output = 20
User Archive = 7

Error CODES -- 172 - SVFL
173 - WRTFIL

STATUS CODES -- -12 = I/O Failure
-6 = REC O NOT FOUND
-9 = Illegal ID
-15 = Wrong Archive Mounted

CONSOLE UTILITY DESCRIPTION

NAME -- INDX

FUNCTION -- Index's user or daily archive.

INPUT PARAMETERS -- None

OPERATION -- Starts by releasing CCM. Writes headings on line printer then begins reading archive searching for record zero and outputing file name, date, LU, length, and usage on line printer. Terminates when trailer label is found, by writing "INDX FIN" on console and calling FIN.

LOGICAL UNITS -- LPR output = 14
Archive LU = 7

Error CODES -- 172

STATUS CODES -- -12 = I/O Failure

CONSOLE UTILITY DESCRIPTION

NAME -- RSFL

FUNCTION -- Restores file from user or daily archive to file manager.

```
INPUT PARAMETERS -- 1. User ID (Format I4).  Negative ID
                       signifies daily archive is mounted.
                    2. Legend (DA,HH,$$), (Format A2)
                    3. Optional, depending on legend:
                       DA: code (Format I5)
                           run no. (Format I5)
                           Lab Addr. (Format 03)
                       HH: Alpha name (Format A2,A2,A2)
                       $$: Octal name (Format 06,06,06)
                    4. Code: -1 for Terminate
```

OPERATION -- Requests above inputs from console TTY. Releases CCM and determines whether a user or daily archive is mounted by checking for a negative ID number input. Begins searching archive for file name requested. If found, calls REDFIL to write file onto file manager. Returns to "CODE" for next restore or terminate.

```
LOGICAL UNITS -- TTY input  = 20
                 TTY output = 20
                 Archive LU =  7

Error CODES --   172 - RSFL
                 174 - REDFIL

STATUS CODES -- -12 = I/O Failure
                -15 = Wrong Archive mounted
                -14 = Unsuccessful archive search
                Also, return error status codes from REDFIL
                (See listing errors 174).
```

CONSOLE UTILITY DESCRIPTION

NAME -- DSPL

FUNCTION -- Display file manager directory.

INPUT PARAMETERS -- 1. Individual user ID (Format I4) OR general display ("00" - zero).

OPERATION -- Requests mode from console TTY and prints headings on line printer. Gets directory and searches for user ID and existing files. If no ID's exist, "DIRECTORY IS EMPTY" is output on line printer. If ID's exist, PFMD is called to print one of the following:

1. Record zero information
2. "NO FILES FOR THIS ID"
3. "RECORD ERROR"

Returns from PFMD and checks for next directory entry, if requested. Otherwise, terminates printing "END OF DISPLAY" on line printer. Loads return status and calls FLRET.

LOGICAL UNITS -- TTY input = 20
TTY output = 20
LPR output = 14

Error CODES -- none

STATUS CODES -- -12 = I/O Failure
+1 = OK

CONSOLE UTILITY DESCRIPTION

NAME -- CLOS

FUNCTION -- Closes file by turning off open flag.

```
INPUT PARAMETERS -- 1. User ID (Format ID)
                    2. Legend (DA,HH,$$), (Format A2)
                    3. Optional, depending on legend:
                       DA: Code (Format I5)
                           Run No. (Format I5)
                           Lab. Addr. (Format 15)
                       HH: Alpha name (Format A2,A2,A2)
                       $$: Octal name (Format 06,06,06)
```

OPERATION -- Requests above parameters and calls HFIND to find record zero. When found, calls FMGR action module CLOSE (05). Loads return status from FMGR and terminates by calling FLRET.

```
LOGICAL UNITS -- TTY input  = 20
                 TTY output = 20
```

Error CODES -- None

```
STATUS CODES -- -12 = I/O Failure
                +1  = OK
                Also, return error status codes from FMGR
                (See listing).
```

12. OPERATING PROCEDURES

A. Initializing

Initializing the file manager is done by calling CCM module INIT, which requires two parameters. The first is the directory logical unit (FLU), which is determined by SYSGEN. The logical unit is currently named FMR1 (LU27). The second is DSIZE, the number of 50 word disk segments reserved for I.D. directory entries. Initializing is always done before rolling in of daily archives or when a fresh file manager directory is desired. There should be no file manager users running and no files left on the directory which have not been archived, as such files will be lost.

B. User ID's

A New user ID is entered into the directory by calling CCM module NWID. No files can be acquired without there first being a matching ID in the directory. Care should be taken so that one ID will represent a single users data. This is necessary for the validation of ID's and for user archives when files are saved. An ID may be removed from the directory by calling CCM module RMID.

C. Deleting Files

File manager files are deleted by first marking the file for deletion by calling CCM module DLET. The deleting is actually done by calling CCM module RLOT to save the files on a daily archive and calling CCM module RLIN which detects the delete bit and omits the file.

D. Removing Files

Files may be removed at once by calling CCM module RMOV. This is not recommended for long files, however. Problems may arise in the deallocating of large quantities of disk from the disk allocator.

E. Closing Files

Closing file manager files from the console may be done by calling CCM module CLOS, which sets the close flag in the file work area. This may be done when a user is unable to terminate or terminates without closing the file.

F. Archive Tapes and Reorganization

1. Daily archives are produced by calling CCM module RLOT. This should be done under the following conditions:

   a. When a fresh copy of the operating system has been booted into core; a scratch tape or serial disk may be used with no saving of the archive necessary after the system has been initialized.

   b. When the file manager disk area has reached its capacity; archives should be saved and files processed on user archives at a future date.

   c. When a backup copy of file manager files is desired (usually done once a day); archive should be saved and recycled (usually 30 days).

   d. When it is felt that debugging would endanger existing files; archive should be saved until development is completed and system restored.

When restoring the system, as mentioned above, the daily archive is reloaded by calling CCM module RLIN, which creates a file manager directory. Header and trailer labels are generated within RLOT and are compatable with LABGEN (subroutine used to produce user archive labels) produced labels.

2. User archives are produced by calling CCM module SVFL. The file is read from the file manager onto a user archive. This is done periodically, depending on the volume of files. Ideally, files should be saved the day after they are acquired. This eliminates having to roll in a previous daily archive, which cannot be done with users on-line. Files are stacked on the user archive at the "end of tape," which is the trailer label. User archive labels are produced by the batch processing routine LABGEN.

At some point, the user may request a previously acquired file which is on his archive. This file may be restored by calling CCM module RSFL, which searches the archive for the named file and restores it to file manager.

G. File Manager Status and Indexing

At any time the file manager may be interrogated by calling CCM module DSPL. This will produce a list of files and their status on the line printer. A display may be produced for a given ID or the entire directory. The contents of a user or daily archive can be obtained by calling CCM module INDX, which prints header and trailer labels and file identification of the archive files on the line printer.

13. ACKNOWLEDGEMENTS

Peter Shoenfeld served as designer and general supervisor on this project. He programmed the executive routines and the action programs. Lawrence Kaetzel programmed most of the utility package. John Barkley wrote the roll-out and roll-in modules, subroutines REDFIL and WRTFIL, the batch communication feature, and the NBS Disk Allocator. The first functional specifications were due to Richard Freemire, of the NBS Institute for Basic Standards. All work was done under the administrative direction of James R. DeVoe.

14. LITERATURE

1. DeVoe, J. R., Shideler, R. W., Ruegg, F. C., Aronson, J. P., Shoenfeld, P. S., Computer Utility for the Analytical Laboratory, Anal. Chem. 46, 509 (1974).

2. ASSET IV Real-Time Monitor User's Manual, EMR Computer, Bloomington, Minn., 1970.

3. FORTRAN IV User's Manual, EMR Computer, Bloomington, Minn., 1970.

APPENDIX A: ERROR CODES

```
                              RETURN STATUS CODES
                              FOR FILE MANAGER

****************************************************************************

DECIMAL    OCTAL                        DEFINITION/SOURCE

****************************************************************************

FOLLOWING STAT CODES RETURNED FROM CFLUT AND ARUT WITH THE FORMAT:
     ERROR 172 XXXXX


  +1        1                   OK

  -1        177777              I/O FAIL (FMGR)

  -2        177776              WORK AREA NOT INITIALIZED

  -3        177775              FILE ALREADY DELETED

  -4        177774              DISC ALLOCATOR ERROR

  -5        177773              DUPLICATE NAME

  -6        177772              MATCHING ID AND FILE NAME NOT FOUND

  -7        177771              FILE ALREADY OPENED

  -8        17777Ø              FILE MIS-POSITIONED (PAST END)

  -9        177767              ILLEGAL ID

  -1Ø       177766              FILES EXIST

  -11       177765              PARAMETER ERROR - CFLUT

  -12       177764              I/O FAILURE - CFLUT

  -13       177763              WRTFIL ERROR

  -14       177762              UNSUCCESSFUL ARCHIVAL SEARCH

  -15       177761              WRONG ARCHIVE MOUNTED
------------------------------------------------------------------------------

FOLLOWING STAT CODES RETURNED FROM WRTFIL WITH THE FORMAT:
     ERROR 173 XXXXX

  +1        1                   OK

  +2        2                   ONLY RECORD Ø IN FILE

  -1        177777              I/O FAIL (SERIAL UNIT)

  -2        177776              I/O FAIL (DISC)

  -3        177775              WORD COUNT > 1ØØØ

  -4        177774              ILLEGAL DISC ADRS WHILE WRITING OUT



  -5        177773              SEGMENT EXCEEDS FILE MAX

  -6        177772              EOF CONDITIONS DO NOT AGREE

  -7        177771              ID AND LU DO NOT AGREE
------------------------------------------------------------------------------

FOLLOWING STAT CODES RETURNED FROM REDFIL WITH THE FORMAT:
     ERROR 174 XXXXX

  +1        1                   OK

  -1        177777              I/O FAIL (SERIAL DRIVER)

  -2        177776              FMGR STATUS ERROR

  -3        177775              WORD COUNT > 1ØØØ

  -4        177774              DELETE FLAG ON

  -5        177773              ILLEGAL FORMAT TAPE

  EOF
EOJ/
```

USCOMM-NBS-DC

NBS-114A (REV. 7-73)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET	1. PUBLICATION OR REPORT NO. NBSIR 75-713	2. Gov't Accession No.	3. Recipient's Accession No.

4. TITLE AND SUBTITLE

A File Management System for a Laboratory Automation Facility

5. Publication Date

June 1975

6. Performing Organization Code

7. AUTHOR(S)

Peter S. Shoenfeld and Lawrence J. Kaetzel

8. Performing Organ. Report No.

9. PERFORMING ORGANIZATION NAME AND ADDRESS

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234

10. Project/Task/Work Unit No.

310 1113

11. Contract/Grant No.

12. Sponsoring Organization Name and Complete Address *(Street, City, State, ZIP)*

Same as Item 9

13. Type of Report & Period Covered

Final Report

14. Sponsoring Agency Code

15. SUPPLEMENTARY NOTES

16. ABSTRACT *(A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here.)*

The National Bureau of Standards' Analytical Chemistry Division operates a centralized laboratory automation facility built around a multi-programmed minicomputer. A file manager was developed which allows the dynamic creation and manipulation of sequential disk files. Although the system was developed for real-time data acquisition, it is a general purpose addition to the computer's operating system and may be used for a variety of applications. A new operating system function was developed to allow the queued scheduling of programs. This is used to achieve more efficient multiprogramming. A comprehensive file utility package is also provided.

17. KEY WORDS *(six to twelve entries; alphabetical order; capitalize only the first letter of the first key word unless a proper name; separated by semicolons)*

Data acquisition; file system; laboratory automation; multiprogramming; operating system; real-time.

18. AVAILABILITY ☒ Unlimited

□ For Official Distribution. Do Not Release to NTIS

□ Order From Sup. of Doc., U.S. Government Printing Office
Washington, D.C. 20402, SD Cat. No. C13

☒ Order From National Technical Information Service (NTIS)
Springfield, Virginia 22151

19. SECURITY CLASS (THIS REPORT) UNCLASSIFIED	21. NO. OF PAGES 52
20. SECURITY CLASS (THIS PAGE) UNCLASSIFIED	22. Price $4.25

AMERICAN REVOLUTION BICENTENNIAL 1776-1976

AMERICAN REVOLUTION BICENTENNIAL
1776-1976