digital

July 7, 1970

TO: PDP-10 Customers

FROM: David Stone
Supervisor - PDP-10 Monitor Development

SUBJECT: The 5.01 Monitor for the PDP-10

This document explains the contents of the software kit accompanying the release of 5.01. Read it first to guide you to an understanding of the rest of the material.

Sincerely,

David L. Stone

David L. Stone

DLS:ll

Encl:

DIGITAL EQUIPMENT CORPORATION, 146 MAIN STREET, MAYNARD, MASSACHUSETTS 01754
(617)897-5111 TWX: 710-347-0212 TELEX: 94-8457

FIVE SERIES MONITOR

INSTALLATION GUIDE

Table of Contents

For additional copies order No. DEC-10-MRZA0D from Program Library, Digital Equipment Corporation, Maynard, Mass. 01754
Price $2.00

TABLE OF CONTENTS

1. WHAT IS 5.01?

1.1 For brief descriptions of the following modules, see Table.TXT.

1.2 A complete list of 5.01 monitor modules:

* = Disk Related N = New

BTHINT		
CCIINT		
CDPSER		
CDRSRX		
CLKCSS		
CLOCK1		
COMCON		
COMMON		
COMMOD	*	
CONFIG		
CORE1		
DATDMP	*	
DISSER		
DLSINT		
DPXKON	*	
DTASRN		
EDDT		
ERRCON		
FHXKON	*	
FILSER	*	
FT40D		feature test switches
FTTM1Ø		feature test switches
FT4ØN		feature test switches
FT5ØS		feature test switches
JOBDAT		
KONPAR	*	
LPTSER		
MDXKON	*	
MOVIE		
MTASRX		
NULSEG		
ONCE		
ONCMOD	*	
PATCH		
PLTSER		
PTPSER		
PTRSER		
PTYSRF		
REFSTR	*	
SCHED1	N	
SCNSRF		

SEGCON
S
SWPSER N*
SYSINI
SYSMAK
TMPUUO
UUOCON

1.3 A complete list of monitor support cusps and their documentation:

* = old cusp which has been updated.
X = old cusp which has been replaced.
"tape N" indicates document in dectape image N.
The Systems Manager's Guide is in the PDP-10 Software Notebook.

CUSP	COMMENTS	WHERE DOCUMENTED
ALCFIL	(called ALCDSK in DSK016.MEM)	DSKØ16
BOOTS	(bootstrap disk file dump and load)	Tape 1Ø
* CHKPNT		Sys.Man.Guide
DATDMP	(dump contents of Level D core blocks)	Tape 1Ø
X DD1Ø	(replaced by FILEX)	--
DMPFIL		Tape 12
DSKC	(Interim routines to print	This letter
DSKCØ	remaining amount of storage	This letter
DSKC1	on file structure)	This letter
DSKC2		This letter
DSKLST		DSKØ16
DSKRAT	(file structure damage assessment program)	Tape 1Ø
FAILCD	(failsafe-Level C format tape to Level D)	Tape 13
FAILDC	(failsafe-Level D to Level C format tape)	Tape 13
FAILSA	(failDD - Level D only)	Tape 1Ø
X FDDD1Ø	(replaced by FILEX)	--
* FILDDT		Sys.Man.Guide
FILEX	(Any dectape format-to-disk conversion)	Tape 1Ø
FILTST	(Test Program)	Tape 12
GRIPE	(submit user complaints)	Tape 1Ø
* LOGIN		Sys.Man.Guide, Level D.MEM
* LOGOUT		Sys.Man.Guide, Level D.MEM
LOOKFL	(type extended entries of a file)	Tape 1Ø
MONEY		Sys.Man.Guide
OMOUNT	(See mount command,DSKØ16)	DSKØ16,
X PAKLOD	(replaced by Boots)	--
* PAL1Ø		Tape 16

CUSP	COMMENTS	WHERE DOCUMENTED
* PLEASE		Tape 1Ø
* PRINT		Sys.Man.Guide
* PRINTR		Sys.Man.Guide
QUOLST	(print a user's quotas)	Tape 1Ø
* REACT		Sys.Man.Guide
SCRIPT	(test system with many jobs)	Tape 12
SETSRC	(print and change search list)	DSKØ16
* SYSTAT		Tape 1Ø
* TENDMP		reference handbook
UMOUNT	(see mount command, DSKØ16)	DSKØ16,

1.4 A list of Level C only monitor routines replaced by Level D routines:

DPCINT
DPCREF
DSKSER
FHDINT
FHDREF
MDFINT
MDFREF
SCHED (now two routines; SCHED1 and SWPSER)

2. A GUIDE TO THE 5.01 DOCUMENTATION

2.1 Existing Documentation.

Except in the area of the disk service, the 5.01 monitor conforms to the same documentation as the previous release (4S72) as modified by the published Monitor Change Orders (MCO's). The definitive document is the PDP-10 Software Notebook which is updated at least quarterly. Other documents which pertain are the PDP-10 Reference Handbook and the newly published PDP-10 Timesharing Handbook. These are modified by the MCO's which are included in this release.

2.2 New Documentation.

2.2.1 Level D.MEM

This memorandum is the best introduction to the new Level D disk service. It is divided into four sections -- each explaining the disk service from a different point of view. The operator,

the console user, the programmer and the system manager are all given an introduction to the details which are more fully described in DSKØ16.MEM.

2.2.2 DSKØ16.MEM

DSKØ16.MEM is the definitive detailed description of the Level D disk service. It should be read by all systems programmers and system managers. Since it is an older document, some changes have occurred. These are described in LEVELD.MEM.

2.2.3 MONITR.OPR

This document contains complete instructions for building and operating the 5.01 monitor. It is extremely valuable to system operators and managers.

2.2.4 TABLE.TXT

TABLE.TXT is a table of contents for the magnetic tape which contains the 5.01 release. It divides the release into sections which correspond to the DEC-tapes on which they are kept at Maynard. Each module is briefly identified by a line of text.

2.2.5 MONSUP.MAN

This document is a supplement to the other documentation and contains information about 5.01 which is not directly related to the disk file system -- e.g. the DC10E handler and the TMPCOR feature.

2.2.6 DOC'S, MEM'S and MAN'S.

On the release tape are many files with extension DOC, MEM and MAN. These files are the original documentation for the monitor support cusps supplied on the tape. In section .1.3 of this letter, references to "tape 1Ø" mean that the cusp in question (say FILEX) is documented by an ascii file named FILEX.MEM to be found in the DEC-tape 10 section of the release.

2.2.7 FLO'S.

The three files in the release with extension FLO are "english flow-charts" of the internal workings of three major portions of the Level D disk service. They are intended to give insight into the algorithms used by the programs but are not at this time completely accurate. However, as a guide to the system programmer they are still valuable, though not as definitive as the listings.

2.2.8 PATCHES

Fixes to the monitor made after 5.01 was frozen are documented in the patch release included in the software kit.

3. A FEW WORDS ABOUT THE LEVEL D DISK SERVICE

The Level D Disk Service included in the 5.01 release is extremely complex and very efficient. It has a large number of options and parameters which can be used to tailor it to a specific configuration. However, since there are so many hardware and software options, it is inevitable that some combinations are never tested together. The standard answers for the once-once dialogue are given in LevelD.MEM. The answers given there have all been tested. Hardware configurations tested have included RM10B drums, RD10 fixed head disks, RP01 disk packs (up to 4) and RP02 disk packs (up to 3). Dual RP10 controllers have been used to drive both RP01 and RP02 disks simultaneously. Both file and swapping storage have been put on all of the above devices, singly and in combination. In short, we have tested a large number of options but have left some out.

As a result of the complexity of the options available, Software Trouble Reports for the disk service will have to include a great deal of pertinent information in order to be serviced. Anomalies of hardware configuration, file structures and software parameters will have to be spelled out explicitly where pertinent.

4. 5.01 RELIABILITY

The 5.01 monitor has been running in some form at Maynard on our development PDP-10 since March of 1970. We have subjected

the monitor to two kinds of stringent testing in addition to the usual unit and integrated tests made for any project. First, we have created a program to generate jobs which perform actions according to directions in an ascii file (for details see SCRIPT.MEM on Tape 12). Using this program to generate job loads as high as 64 users (on an 80K configuration) we have exhaustively tested the various areas which typically cause trouble with monitors -- limit conditions expecially. We have scripts which fill up and fragment the swapping space, load the IO channels, exhaust the available disk space, etc. These scripts, when run simultaneously, cause the monitor algorithms considerable anguish. However, the 5.01 monitor has been taught to live comfortably with them.

Another application of the SCRIPT program has been to generate job loads which closely approximate the user job loads which we see in-house on our production machine and on other machines run by our customers. These scripts, which use all the standard language processors and cusps, also run the line printer and tape drives to simulate an actual user environment. They have been very helpful in debugging the 5.01 release.

Most important of our reliability tools, however, has been our production time-sharing PDP-10. This machine has a user population which typically varies from 28 to 36 during prime shift (8:30 a.m. to 6:00 p.m.), from 15 to 20 during the second shift (6:00 p.m. to 12:00 p.m.) and from 0 to 10 during weekends. These three different intensities of use give us a good evaluation of the reliability of our monitor under different loads. The configuration of the machine is 80K of 1.8 µs memory, one RM10B for swapping on its own DF-10, 3 RP02 disk drives on another DF-10, 3 magnetic tape drives, 8 DECtape drives and a line printer. It handles up to 40 simultaneous users -- primarily on hardwired TTY's but with up to eight remote TTY's. The reasonable response job limit for this configuration is about 30-34 jobs. Most of the users are systems programmers and therefore are capable of putting more strain on the system than most users.

During most of the month of June, precursors to 5.01 have been running on the production machine. Reliability has proven to be roughly linear with load. During the two weeks preceeding the writing of this letter, the crash rate on that machine has been:

	Week 1	Week 2
1st. shift	1 crash/shift	
2nd. shift	1/2 crash/shift	NO CRASHES
Weekends	No Crashes	

During this period, the number of users who can be accommodated with reasonable response has increased over Level C by about 5 - or roughly 20%.

5. EFFICIENCY AND THROUGHPUT MEASUREMENT

5.1 WATCH.

A new feature has been added to the 5.01 monitor to allow the user to observe the service he is getting and the efficiency of his programs. The WATCH command allows each user to have a printout of the time of day at the beginning of each command which uses user core and at the end, a printout of the elapsed real time and cpu time, plus the number of disk reads and writes (in blocks) for his program. Any or all of these printouts may be requested. Once turned on, they continue until another Watch command is given. The default printouts are set table at assembly time and patchable. The monitor is distributed with the default set to no printout. Type WATCH H for a list of the parameter names. See MCO's for a detailed description.

5.2 Level D versus Level C.

Some measurement of the efficiency of Level D versus Level C has been made. While any such tests are bound by a large number of parameters (exact placement on the disk, buffer size, competition from other users), a few general comments apply.

For a sequentially accessed file stored in contiguous blocks on a disk pack, Level C will require a full latency between adjacent blocks if they are requested by two separate input UUO's. Since Level C allows no parallel operations on other disks, the entire system is slowed down by the loss of latency. The maximum throughput for a single block requested at a time (assuming the drive to be always on track) is one block per 27.5 milliseconds. Level D, on the other hand, will not miss the latency if the input request is made in time.

The more disk drives, controllers and channels you have, the more efficient Level D will be over Level C because of parallel seeks and data transfers. With four RP02's on one controller and one channel, if a random track request for a single block is always queued up for each drive, the Level C throughput is one block every 65 ms

or 15.4 blocks per second. The Level D throughput is one block every 15 ms or 67 blocks per second.

On a user benchmark involving a loop of fortran overlays, Level D took less than one-third the real time required by Level C on the same equipment. More comprehensive measurements of Level D's efficiency are under way and will be reported later.

6. SOME NEW PROGRAMS

6.1 Crash Procedures.

To speed up the operation of a computer, the 5.01 release contains two new support cusps which facilitate loading a new monitor, dumping a dead monitor and dealing with the dump; they are BOOTS and FILEX. Both these programs have their own documentation files on DECtape image 1Ø. BOOTS should be kept on a paper tape for loading into the machine after a crash. It will save the remains in "CRASH.SAV" and reload the system from "SYSTEM.SAV". FILEX will expand the CRASH.SAV file into input suitable for FILDDT to analyze.

6.2 SCRIPT and SCP's.

Included in the 5.01 release are the SCRIPT program (mentioned above) and a number of files with extension SCP. These latter are the ascii input to SCRIPT and consist of several of our test programs. We have included them as potentially helpful examples of SCRIPT input.

6.3 FRM's.

Files with extension FRM are examples of the forms which we use at Maynard to report software errors and fixes. They are included because some of our customers indicated that they would be useful.

6.4 DSKC and Friends.

The files DSKC, DSKCØ, DSKC1 and DSKC2 are simple DDT programs to print the number of free blocks remaining on a file structure. They are interim programs to help out until the "LEFT" command is implemented. DSKC prints out the total number of blocks left on logical structure DSKC. Similarly, DSKCØ -1 and -2 print the number of

free blocks left on units Ø, 1 and 2 of logical structure DSKC. By suitable use of the ASSIGN command (.ASSIGN DSKB DSKC) any file structure can be investigated.

6.5 Others.

For descriptions of other new programs see section 1.3 of this letter for the place in which they are documented.

7. DOCUMENTATION CHANGES

7.1 Protection.

The 5.01 file protection scheme is almost completely compatible with Level C. The correct description is given in Level D.MEM Version 2 dated May 27, 1970 on page 27. DSKØ16.MEM has an incorrect description. The default protection has been set to Ø57 rather than Ø55 at the request of our service bureau customers. This means that files in one project cannot be read by another project. To change the default, set the symbol PRVFIL to your choice at MONGEN time.

7.2 LOOKUP, ENTER, RENAME.

7.2.1 Error code #6 may now include BADUFD.

7.2.2 Note that four word LOOKUP'S are still legal. They are distinguished by a non-zero left half of word one.

7.2.3 Rename.

A deficiency in 5.01 is the inability to rename a file and transfer it to a different UFD. It must be copied separately.

7.3 USETO.

When USETO is set past the end of a file, zeroes are written in the file at that time, not later at output time.

7.4 Read Image Mode.

Contrary to DSKØ16.MEM section 2.1, read image mode is not implemented in either hardware or software.

7.5 CRASH.SAV.

A new once only question asks the size of CRASH.SAV on each file structure. This file is for use with BOOTS (see section 6.1 of this letter) and need not be given any room if BOOTS is not to be used.

7.6 ALCDSK vs ALCFIL.

The cusp described as ALCDSK in DSKØ16.MEM is now named ALCFIL.

7.7 FILE command.

There has been an option switch change in FILE. "D" which formerly meant "DIRECTORY" now means delete. "L" is the new switch for DIRECTORY LISTING.

8. DEFAULT CHANGES

8.1 SYS Protection

When files are copied into SYS, the operator should explicitly set the file protection with PIP or FILEX to 155. Otherwise the standard protection of Ø57 will be substituted thereby preventing access by most of the users. FAILSAFE and FAILCD automatically restore files explicitly to their original protection.

8.2 Tape Density.

Default tape density has been set to 800 BPI.

8.3 Memory Speed.

The default memory speed for shuffle time computations has been set to 1 microsecond.

9. RANDOM INFORMATION OF INTEREST

9.1 The DIRECTORY command now requires a colon (":") after the device name.

9.2 There may be a difference between the number of blocks allocated to a user (printed by LOGOUT) and the number

written (printed by DIRECTORY) if the cluster size is is not one.

9.3 If you want to look at other people's files on UFD'S you must explicitly specify any file structures which they have access to which are not on your search list. The DIRECTORY command will only print files on structures in your search list.

9.4 For a file with protection 455, the owner cannot write or delete. He must rename to a lower protection first. 455 is the protection used by Level C LOGOUT to indicate a "preserved" file.

At our installation, many of our more knowledgeable users rename all their files with PIP to have protection 4XX to speed up LOGOUT. If they choose to do this under Level D, they should make the protection be 1XX rather than 4XX.

9.5 LOGOUT when confirming deletion of individual files requires a K and will not accept just a carriage return.

9.6 The number of core blocks used by the monitor is settable at once-only time. Patch symbol CORNUM; each block is worth 4 words.

9.7 If once-only gets a hardware error it puts the controller CONI status in the lights.

9.8 For faster Failsafe operation after refreshing a file structure, use the "/N" switch to prevent LOOKUP'S.

9.9 FTHALT is 1 for this release.

10. IMPLICATIONS OF SOUP

Be sure to keep an inviolate copy of all original sources in this release as a "father" for SOUP updates in the future. We hope to make most future releases with SOUP.

11. 10/40 N and 10/40 D MONITOR

The sources for 5.01 will not produce a 10/40 N or 10/40 D monitor. Some undefined globals occur. This will be rectified in the future with SOUP updates.

12. HOW TO GET ON THE AIR WITH A MAGTAPE DISTRIBUTION

Since you received your monitor distribution on magnetic tape, you will need to read MONITR.OPR sections 1 and 2 and LEVELD.MEM sections 4.2 in order to make a 5 series monitor for your configuration. The magnetic tape contains all of the monitor sources in a Level C FAILSAFE format in project-programmer number 1Ø,7. Thus old customers who already have a disk system, may read the magnetic tape with a Level C FAILSAFE onto their Level C disk system. New customers will receive a monitor already built for their hardware configuration on a DECtape labeled "YOUR MONITOR" (Tape #1). They must read MONITR.OPR section 1 and LEVELD.MEM section 12 in order to start running their system.

MCO 776

symp: Undefined symbol in DPCREF when assembled for 10/40 disk system.

diag: DPCIOC under FTSWAP conditional.

cure: Remove from conditional.

DPCREF

MCO 780

symp: If monitor assembled with FTHALT=-1 monitor halts whenever user has an ILM.

diag: The check in APRINT for PI's in progress must not include the APR PI level itself.

cure: Define a symbol in common (APRNOT) which is all PI's levels except APR for CONSZ PI.

CLOCK1 p. 19
COMMON

MCO 781

symp: Monitor halted in CHKTAL routine with negative CORTAL. Sharable high data segment (write lock off) is destroyed, and then appears to occupy 256K of core.

diag: The sharable high data segment had become idle and got deleted from core under the false assumption that a copy existed on the swapping space (not true for non-write-locked segments).

cure: The FRECR1 routine in SEGCON must be made to recognize this case and cause the segment to be swapped out. However, CORTAL must remain unchanged by this operation.

SCHED p. 43, 46
SEGCON p. 42, 59, 61, 64

MCO 782

symp: FILDDT restricted to searching 48K.

diag: Arbitrary limit imposed in routines SETUP and FETCH.

cure: Remove the limitation.
DELETE {FETCH+6/ CAIL R,30
{FETCH+7/ POPJ P,
CHANGE SETUP+1 MOVEI T,137777 MOVEI T,777777
Return Ø if attempt is made to fetch a word outside the actual CRASH.SAV file

FILDDT

MCO 785

symp: TTY logical names are lost across DETACH, ATTACH sequence.

diag: SCNIN routine of SCNSRF using DEVLOG as a temporary.

cure: Use AC17 instead after saving on the stack.

SCNSRF

MCO 787

symp: PRINT ignored an input if the form *.*)

diag: Code at BLANKQ was ignoring the wrong terminator.

cure: Change BLANKQ:

```
CAIE CH,33
CAIGE CH,15
JRST CMDERR
JRST DONE
```

To BLANKQ:

```
CAIE CH,33
CAIGE CH,15
JRST DONE
JRST CMDERR
```

PRINT

MCO 788

symp: PRINT and UFILE take an inordinate amount of time to store a command file when the system is heavily loaded.

diag: PRINT and UFILE each do a succession of LOOKUP's until a free name is found that increases in length as the queue increases.

cure: Generate command file names at random using the millisecond time UUO.

UFILE (V005)
PRINT (V005)

MCO 789

symp: Deposit to a high segment appears to work, but later previous contents is restored.

diag: Changed high segment is not swapped out when earlier copy exists in the swapping space.

cure: Call ZERSWP after successful deposit to a high segment.

SEGCON p. 27

MCO 790

symp: 1) In REACT, L DEV:FILE.EXT,PROJ,PROG doesn't work. Types "? BAD FILE NAME SYNTAX"

2) ?BAD OCTAL NUMBER TYPED WHEN TERMINATING "]" ENCOUNTERED.

diag: 1) FILSPC routine not recognizing "," as a terminator.

2) OCTRD routine not recognizing "]" as a terminator.

cure: Add additional tests.

REACT

MCO 791

symp: 1) ENTER to DECtape fails for no apparent reason and/or

2) FILE appears in DECtape directory with Ø blocks.

diag: a) Test in ENTER code to prevent ENTER to a file open for reading fails spuriously because index in IBLK is cleared only on RELEASE.

b) LH of IBLK is used by dead-reckoning code.

cure: a) Clear index used for test on INPUT CLOSE as well as on RELEASE.

b) In DTASRN use a temporary for dead-reckoning code rather than IBLK (DEVDAT)

(See further correction in MCO #824)

DTASRN p. 17, 18
DTCSRN p. 17, 34
DTASRN p. 44, 46, 49, 56

MCO 792

symp: HALT in ERRCON in ERRPNT with job # out of range (when FTHALT=-1). (Trying to print swap read error message).

diag: ERRPNT called with high segment # in ITEM and bits set in left half.

cure: Replace segment number with associated job # before calling ERRPNT.

SCHED p. 41
SEGCON p. 60

MCO 793

symp: Parity recovery code is not used (intentional).

diag: 4S74 was distributed with the code purposefully included but never called so that it could be debugged later and small patches sent.

CLOCK1 p. 16

MCO 794

symp: Random locations may be clobbered if operator continues after a parity error halt (i.e. PC was in exec mode or more than one parity error in user mode).

diag: On a parity error halt, the monitor must not attempt to fix up the bad word if parity error did not occur out of the sweep loop.

cure: Check PC on parity error from exec mode. If not in sweep loop, don't try to fix bad word.

CLOCK1 p. 17

MCO 795

symp: Mem parity error message does not print absolute address correctly.

diag: TAC1 instead of TAC was being set up to call OCTPNT.

cure: Set up TAC with absolute address.

ERRCON p. 1, 16, 16-1

MCO 798

symp: High segment MACRO.SHR didn't get marked for later deletion from segment table when MACRO.SHR renamed to MACRO.OLD.

diag: DSKSER now changes filename and extension in DDB after a RENAME so FNDSEG never found correct high segment when it existed.

cure: Call FNDSEG before RENAME and call CLRNM1 after successful RENAME if shared segment being RENAMED.

UUOCON p. 41
SEGCON p. 68, 72
NULSEG P. 17

MCO 801

symp: PAKLOD doesn't work on RPØ2's.

diag: Test for which kind of drive is wrong.

cure: Move 2 bits. (2 other trivial changes)

PAKLOD p. 2, 4, 1Ø

MCO 802

symp: Monitor halts in APRINT with PI in progress on APR channel and some lower priority channel.

diag: Out-of-bounds transfer address for user enabled traps leads to loop on APR channel until a lower priority interrupt occurs.

cure: Treat interrupt occurring at trap instruction fetch as though user not enabled for that error and give an error message.

In addition, more code was added to cope with the situation of an ILM that appears to be in EXEC mode, but really only did a JSR on the lower priority interrupt level. This is treated as though ILM occurred directly and lower channel is given a legal address to dismiss to.

CLOCK1 p. 16, 18, 19

SPECIFICATION CHANGE
MCO 805

symp: Can't determine what's a card punch (needed by Fortran).

diag: Can share CDR/CDP with DVCDR, since one is input and one is output.

cure: Set DVCDR in CDPDDB.

CDPSER p. 1

MCO 806

symp: MONGEN asks for dev:file specification for CONFIG but ignores extension even if supplied.

diag: No code to check for user supplied file extension.

cure: Add code.

MONGEN p. 3

MCO 810

symp: FILDDT output showed undefined symbols that should have been defined.

diag: Final part of monitor symbol table lost because EOF encountered.

cure: Remove test for EOF but zero window contents before reading.

FILDDT

MCO 811

symp: Push down list overflow at channel 7 level.

diag: Decoding RUN command at clock level which required deleting a fragmented high segment.

cure: Increase size of NULPDL.

CLOCK1 p. 21, 22
COMMON p. 21, 23

SPECIFICATION CHANGE
MCO 815

symp: User requests that CDPSER ignore RUB-OUTs in ASCII mode.

diag: Seems like a good idea.

cure: CAIN TAC,177
JRST NOPUN

CDPSER p. 19

SPECIFICATION CHANGE
MCO 817

symp: User desires error return on OUT UUO when gets EOT on MAGTAPE.

diag: Sounds like a winning change.

cure: Change UUOCON exit from OUT UUO to also look for EOT bit.

UUOCON p. 62

MCO 821

symp: DSK not available or
.AS DSK 1 already assigned to JOBØ.

diag: Not enough monitor CORE for DISK DEVICE DATA BLOCKS, particularly when running COBOL jobs which have 8 open files.

cure: Increase standard size of MINCOR.

COMMON

MCO 824

symp: MCO #791 has introduced an error such that files get garbled on transfer from DECtape or block too large error received.

diag: Storing dead-reckoning block # in DTASRN rather than DDB, but DTASRN switches disconnected tapes. Therefore the number must be in DDB.

cure: Store in LH of DLOC (previously unused).

DTASRN p. 2, 30, 32

SPECIFICATION CHANGE
MCO 825

symp: Cannot do IB (mode 13) to card reader or punch.

diag: Allow mode 13 and define as I (mode 1Ø).

cure:
```
CDPSER$:  CDPIOS+2/DVOUT+DVCDR,,144Ø3
CDRSRX$:  CDRIOS+2/DVIN+DVCDR,,144Ø3
          NOTASC/TRNN IOS,4
```

CDPSER p.1
CDRSRX p. 1, 6

MCO 826

symp: No error message from card punch when output attempted and device not ready.

cure: Call HNGSTP to warn user if trouble.

CDPSER p. 2

SPECIFICATION CHANGE
MCO 827

symp: Cannot do image binary mode output to CDP without handling own buffers.

diag: CDP required different buffer sizes for each of 3 modes:
27 - I, IB
26 - B
16 - A, AL

cure: Add an entry to all devices dispatch table (at XXXDSP-3) which UUOCON will call to set up correct buffer size if it finds buffer size = Ø in the device data block.

CDPSER p. 1, 2
UUOCON p. 32
DPCINT p. 1

MCO 828

symp: Halt in CHKTAL claiming core tables (CURTAB) and count of free core (CORTAL) do not agree.

diag: This is produced only on a swap of high sharable data segment. The fix implemented by MCO #781 was not entirely correct.

cure: Instead of using JBTADR (ITEM) to decrement CORTAL, we must use IMGOUT because JBTADR was cleared when core given back.

SEGCON p. 44, 46

MCO 830

symp: Monitor Listing have "," delimiting comments at beginning of a line.

diag: This will not work under new (V43) MACRO-1Ø.

cure: Write TECO macro and run on all monitor files - Files requiring edit are listed below.

SCHED
UUOCON
PTPSER
PTRSER

SPECIFICATION CHANGE
MCO 831

symp: For systems with more than 64K, DECtapes don't have enough room to accomodate the crash dump.

cure: Use Mag Tapes to save the crash dump. Dataline Systems Mag Tape routine for saving crashes is merged with DEC Tape TENDMP routine. The Mag Tape version of the utility can be assembled by defining MAGT feature test switch.

TENDMP p. 1, 2, 4, 5, 10, 11

MCO 837

symp: "PRINTR" does not recognize VTAB as a valid character.

diag: It should.

cure: Make it - add VTAB to list of valid special characters.

PRINTR

MCO 838

symp: OPFILE does not like user to delete UFD when logging off and leave R or D command pending.

diag: When UFD missing, OPFILE cannot restore to user's UFD since it is not there.

cure: When OPFILE finds missing UFD, have it create a new one.

OPFILE

MCO 839

symp: UFILE pending command (C) is inefficient and inaccurate (when user logs off, then on under a new job #).

diag: (1) Search is by job # instead of PPN.

(2) Command file must be read to determine who user is.

cure: (1) Make ID PPN rather than JOB #.

(2) Include a unique derivative of PPN in command file name, so that most LOOKUPS are unnecessary. Also include a random number in name so that multiple requests from same PPN have a "better" probability of finding a unique command file name on 1st try.

UFILE

MCO 840

symp: Incomplete syntax checking in UFILE - bad syntax not diagnosed until OPFILE phase when user has been waiting for some period of time.

diag: OFILE command scan does not check file names and extensions for legal syntax.

cure: Include length check for |NAME| $\leq$ and |EXT| $\leq$ 3 and no DOT following EXT (any other break is legal).

UFILE

SPECIFICATION CHANGE
MCO 841

symp: No convenient way of deleting files from DECtapes thru FILE command.

diag: Seems like a good feature to add.

cure: Add it under "D" command. Change Directory command to "L".

UFILE
OPFILE

SPECIFICATION CHANGE
MCO 852

symp: Customers wanted more features in the Logout Cusp.

cure: The following new features have been added:
D Deletes all files
F Saves all files
U Allows you individually decide to save or protect all unprotected files

Logout treats files of all extensions the same as compared to the previous version that treates .TMP etc. specially.

LOGOUT

MCO 862

symp: Push down list overflow in monitor.

diag: Need more pushdown list space. However, the job data area cannot be expanded.

cure: Dynamically assign PD list space from monitor free core, and move push down list up to it. Move pushdown list back to job data area when returning to user via a UUO. The number of 4 word blocks assigned to an extended push-down list is EPL4WD, which can be redefined using MONGEN. Current setting is 12 (decimal). Pushdown list overflow message now means either: (1) not enough free core, (2) extended list overflowed too. A record of successful and unsuccessful pushdown overflows is kept in COMMON for GETTAB UUO.

ERRCON
COMMON
CORE1

MCO 863

symp: TIME Ø command info should be in SYSTAT.

cure: Add shuffle and zero core time to SYSTAT. Add GETTAB to find nano-seconds per memory cycle. Remove TIME Ø type-out of everything except routine for null job and KILO-CORE-SEC.

CNFTBL table

item	location	use
21	MEMNSP	No. of NANO-SEC PER MEMORY CYCLE (SET BY MONGEN)

COMCON
COMMON

MCO 864

symp: It takes too much core to load the monitor.

diag: A significant savings could be effected by having only one copy of DDT around instead of both EXEC DDT and USER DDT.

cure: Version 24 of DDT has been created which is capable of running in either exec mode or user mode (by containing the code for both options and doing conditional execution instead of being conditionally assembled). This new version will be distributed with the level D disk monitor.

EDDT

MCO 878

symp: MOVIE does not print more than one line.

diag: Buffer is only that long

cure: Make buffer long enough for a 64 job system with 128K of core. Append a CR-LF at end of text and print only the part of buffer stored.

MOVIE

MCO 879

symp: FILDDT takes 18.5 minutes of CPO time.

diag: Symbol table search time is very long.

cure: Make the current symbol table be COMMON most of the time since most symbols are in it and will be found more quickly. Also add local symbols in COMMON .A, .B, ...Z. Use these symbols in FILDDT.TXT for all intermediate symbol definitions. Symbols starting with . are found first since symbol table is sorted in backwards order.

FILDDT.TXT

MCO 880

symp: SYSTAT needed several improvements:

1. Add shuffle, zcore times
a. Add help command
3. Footnote # and @
4. Leading zero blank or suppress times
5. Print (Self) for P,PN if same as this one unless we are not logged in
6. Prepare to make reentrant
7. Define GOD as Proj. 1, CTY or OPR
8. Make User Core be MEMSIZ-SYSSIZ
9. Report login availability
10. Rearrange null time message
11. Remove extra %s.
12. XLIST Hacques

SYSTAT

MCO 883

symp: Improve SYSTAT:

1. Don't say (SELF) if not logged in
2. Remove unused code
3. Round swap ratio correctly
4. Remove extra comma on file structures
5. Add comments.
6. Print n+nk for job size
7. Print #jobs in system/use/logged in/detached
8. Rearrange some code

MCO 884

symp: MOVIE does not print status of highest job logged in. MOVIE wastes 3 columns by printing null job status and wastes the letter A for null job.

cure: Start job status with job 1 and go through HIGHJB. Use A for low segment of job 1 in core map, B for job 2, etc.

MOVIE

MCO 887

symp: Occasionally the crash procedure does not halt with the PC - 525252.

diag: Continue switch bounces and an illegal UUO is executed.

cure: Put hald . in AC 10 and then halt with PC pointing to 1Ø. If MA also set to 10, then continue switch bounced but caused no problem.

MCO #D-2

symp: EDDT prints octal numbers when symbols are requested.

diag: DDT prints just octal if nearest symbol is more than 100 away.

cure: change to 1000 for exec DDT and FILDDT. A ⟶ B

EDDT$:

MCO #D-3

symp: Many tape read errors on FAILSAFE.

diag: standard density is 556 tape written at 800.

cure: change STDENS in COMMON to 3. so standard density is 800. B ⟶ C

MTASRX$:

MCO #D-7 "Level C bug"

symp: NXM @ 26440 (ANYDEV+15)

diag: Item set wrong.

cure: Do a SOS on -1 (PDP) not Ø (PDP), - which is where item is saved. E ⟶ F

SEGCON$:

MCO #D-10 "Level C bug"

symp: System hung not responding to ↑C.

diag: A job in core has swp bit on because high seg is swapped out. The SCNJOB algorithm in swapper does not count jobs in core with SWP bit on. G ⟶ H

cure: Make SCNJOB test JBTHDR for zero rather than SWP bit in order to ignore jobs.

MCO #D-13

symp: FAILSA changes access dates when it saves files.

diag: Not using CLOSE bit 32.

cure: Change CLOSE FIL, to CLOSE FIL,1Ø
CLOSE CHK, to CLOSE CHK,1Ø
Add CLOSE's before RELEAS'S where necessary.

MCO #D-14

symp: Changed message ASSIGN MTA# FAILSA & THEN RESTART to ASSIGN MTA# FAILSA & THEN START.

MCO #D-15

symp: When restoring FAILSA overwrites existing disk files if the same file on tape has an equal creation date and time (on if the disk file is older).

cure: As of V.26A FAILSA only overwrites existing disk files if they are older - not if the tape file is equal.

MCO #D-2Ø

symp: K for CRASH.SAV set to 262143 when a file structure is defined.

diag: When a file structure is first defined, K for CRASH.SAV is set arbitrarily large by once-only. This forces refresher to allocate maximum amount of contiguous space possible in one retrieval pointer - determined by the size of the cluster count field in the retrieval pointer.

cure: Make the documentation clearer.
Note that K for CRASH.SAV is set to 262143 by once-only when a file structure is defined (e.g. when you dissolve and recreate the file structure).

MCO #D-22

symp: Refresher takes a long time.

diag: refresher zeros all unused blocks in all files it creates for security reasons.

cure: Set protection to 557 to take care of security problem and do not clear blocks.

MCO #D-24 "Level C bug"

symp: Garbage names for dormant segs in TBTNAM table.

diag: FNDSEG expects extension in DEVEXT (DEVDAT) to be "SHR" and it is not because UREMAP calls IOWAIT, which changes DEVDAT. Happens on GETSEG UCO of a sharable high seg while active I/O on some channels.

cure: PUSH, POP DEVDAT around call to IOWAIT.

MCO #D-27

symp: Swapping space disappears.

diag: Core parity err lights an error bit in IOS, so swapper tries different loc on disk to swap.

cure: Reference bad loc, so OPU will notice error (and stop)

FHXKON$:

MCO #D-30 "Level C bug"

symp: PI7 glows a bit, expecially in 64 job system.

diag: COMCNT gets very large, but there are no waiting commands, that is no sign bits set in TTYTAB. This happens when ↑C is sent over a PTY, but I don't know why.

cure: A cure that adjusts COMCNT and at least saves PI7 time is to decrement COMCNT each time:
1. the entire TTYTAB is scanner
2. No delayed commands are found
3. No other waiting commands are found

MCO #D-36

symp:

diag: Core blocks have timing problems.

cure: Interlock FNOFIL routine as it it were a sharable resource.

MCO #D-37

symp: Reduce possibility of timing problems at the cost of some speed.

cure: 1 Monitor buffer instead of 2.

MCO #D-38 "Level C bug"

symp: System hangs in tight loop with PI4 (scanner in progress)

diag: Monitor waiting for CCI buffer empty flag to go off. Never does if PDP-8 is stopped.

BTHINT$:

MCO #D-39 "Level C bug"

symp: Loop with PI 4 (scanner) in progress.

diag: Talk ring is not complete.

cure: Count TTPLEN times (no. of TTY+PTY+CTY lines) then quit if not got back to beginning.

SCNSRF

MCO #D-48 "Level C bug"

symp: Reproducible NXM with SPY UUO.

diag: Arg too big so NXM adr set up in user relocation.

cure: Give error return if LH of user arg is non-zero or RH is greater than size of Monitor (SYSSIZ)

symp: Hard to measure response time of programs.

cure: Add a command called WATCH which causes monitor to automatically print incremental job statistics. WATCH DAY causes the monitor to print the time of day as (HH°MM.SS) when the user started or continued a program with a monitor command. e.g. START, CONT, R,RUN,COMPIL,SYSTAT,etc. WATCH RUN, WAIT, READ, WRITE causes the monitor to print the incremental run time, the wait time (time since user started or continued program), the incremental number of disk blocks read and the incremental number of disk blocks written as (SS.HH␣SS.HH RR␣WW) whenever the console is returned to monitor mode via CONTROL C, EXIT, HALT, ERROR IN JOB, DEVICE XXX OK? WATCH with no arguments eliminates the printing of all incremental job statistics. Any combination of the five arguments may be typed in any order. Each occurrence of the WATCH command clears the status of previous commands. The monitor does not print statistics for commands which do not start up jobs, such as ASSIGN, and PJOB. When a user logs in, his job is set to watch all incremental statistics. The system administrator can change this initial setting by redefining MONGEN symbol SETWCH from 37ØØØØ to any other combinations of bits 1 thru 5, as follows:

bit 1 = setting for DAY, bit 2 = setting for RUN
bit 3 = setting for WAIT, bit 4 = setting for READ,
bit 5 = setting for WRITE
Note: That defining SETWCH to be Ø with MONGEN, causes the monitor to print no incremental statistics.

If the user types an invalid argument to WATCH, the monitor responds with:
ARGS ARE: DAY,RUN,WAIT,READ,WRITE
Note that the order of the error message is the same as the order of output. Thus a user who forgets either the arguments or the significance of the statistics can find out. Note also that the incremental commands TIME and DSK terminate an increment in the same way as the incremental job statistics which have been activated by WATCH. These job statistics are included inside brackets as an indication to the user that the type out is incidental to his major interests and is not being typed out by his program. The single space between each pair of number is always typed, whether the number is or not. Thus is is possible to tell which statistics are being typed without seeing the WATCH command.

MCO #D-58 "Level C bug"

symp: A user program can turn off UWP for a spy seg and can write in monitor.

diag: Check for spy seg is not correct.

cure: Give error return if high seg is spy seg.

SEGCON

MCO #D-59 "Level C bug"

symp: FILDOT is very slow. Also typing one control C while it is running returns to command level immediately.

diag: TTCALL UUO is done after every word in a word search.

cure: Do not to TTCALL if input is from a command file.

EDDT

MCO #D-71 "Level C big"

symp: NXM on getting a non-monotonic file.

diag: Expand algorithm in COMCON assumes monotonicity.

cure: Give ADR check in non-monotonic.

EDITED

MCO #D-80 "Level C bug"

symp: Mem parity error in high seg incorrectly reports parity to be beyond end of low seg.

diag: If high seg is below low seg, the difference is negative.

G H

DIGITAL EQUIPMENT CORPORATION

TIME-SHARING MONITOR

VERSION 5N,01, 5D,01, 5S,01

23 JUN 70

V417

```
        TABLE OF CONTENTS  --  PDP-10 MONITOR LISTING
        /TH/CMF  TS      32 APR 70 V414
V411    THE FOLLOWING IS THE ORDER OF CONTENTS OF THE MONITOR SOURCES AND SUPPORT CUSPS
V411    ON THE DEC FURNISHED LISTINGS FOR THIS MONITOR,  THEY ARE ARRANGED IN THIS ORDER FOR
V411    CONVENIENCE TO THE OPERATOR TO AID IN ASSEMBLING THIS MONITOR AND SUPPORTING PROGRAMS,

V411            FILE NAME        DESCRIPTION

V411            TABLE.TXT        TABLE OF CONTENTS (THIS LISTING)

V411            SYSTEM.MAP       LOADER (SYSTEM BUILDER) STORAGE MAP

V411            GLOB.XRF         GLOBAL CROSS REFERENCE (ALL GLOBAL SYMBOLS)

V415            ALL CCL FILES    FILES TO ASSEMBLE AND PRODUCE BINARY
                                 AND LISTING FILES OF ALL MONITOR AND
                                 CUSP SOURCES,

V411            MONGEN.LST       CROSS REFERENCE LISTING OF CONFIGURATION DEFINITION PROGRAM,

V411            MONITOR SOURCE
V411            LISTINGS         CROSS REFERENCE LISTINGS OF MONITOR SOURCE FILES FOUND ON TAPES
V411                             4 THROUGH 6, INCLUSIVE, EXCEPT FOR ROUTINES ASSOCIATED WITH DISKS
V414            DISK LISTINGS    CROSS REFERENCE LISTINGS OF DISK ROUTINES - TAPE 15

V411            CUSP LISTINGS    CROSS REFERENCE LISTINGS OF MONITOR SUPPORT PROGRAMS
V411            (NOT FURNISHED)  USED WITH THIS MONITOR,
```

TABLE OF CONTENTS -- PDP-10 MONITOR
[THESE TAPE NUMBERS CORRESPOND TO THE MASTER DECTAPE NUMBERS]

	FILE NAME	DESCRIPTION
TAPE 1:	YOUR MONITOR	[CREATED FOR DELIVERY OF MACHINE]
		[IS NOT CREATED FOR SUBSEQUENT SOFTWARE DISTRIBUTIONS]
	FAILCD,SAV	RESTORE LEVEL C MAGTAPE TO LEVEL D DISK
		[USED IF MONITOR IS DELIVERED ON MAGTAPE]
TAPE 2:	10/40 MONITOR MAKER	
	4N72.REL	LIBRARY FILE TO MAKE 10/40 MONITORS
	120,16K	STANDARD 16K 10/30 MONITOR [NEEDED TO LOAD MONGEN,SVE]
	SPMON,16K	SPECIAL 16K 10/30 MONITOR [SAVES IN 10/40, 10/50 FORMAT] [NEEDED TO LOAD 10/40, 10/50 MONITORS]
	SPMON,32K	SPECIAL 32K 10/30 MONITOR [NEEDED TO LOAD 10/40, 10/50 MONITORS]
	SPMON,48K	SPECIAL 48K 10/30 MONITOR [NEEDED TO LOAD 10/40, 10/50 MONITORS]
	MONGEN,SVE	MONITOR GENERATOR (CONFIGURATION DEFINITION) [RUNS UNDER 120,16K OR SPMON]
	PIP,SVE	PERIPHERAL INTERCHANGE PROGRAM [RUNS UNDER 120,16K OR SPMON]
	MACRO,SVE	MACRO ASSEMBLY PROGRAM [RUNS UNDER 12016K OR SPMON]
	LOADER,SVE	RELOCATING LOADER [RUNS UNDER 120,16K OR SPMON]
TAPE 3:	ASSEMBLY TAPE	
	TABLE,TXT	THIS FILE (TABLE OF CONTENTS)
	MONITR,OPR	MONITOR ASSEMBLY AND LOADING INSTRUCTIONS
	TENDMP,MAC	LOAD/DUMP MONITOR ON DECTAPE
	MONGEN,MAC	MONITOR GENERATOR-CREATE CONFIG,MAC VIA DIALOG
	MONGEN,SAV	MONITOR GENERATOR (SAV FILE)
	FILDDT,TXT	FILDDT INPUT COMMAND FILE
	S,MAC	MONITOR PARAMETER DEFINITIONS (LISTED ONLY IN COMMON)
	FT40D,MAC	CONFIGURATION DEPENDENT FEATURE TEST SWITCHES FOR (NON-SWAPPING) DISK
	FT40N,MAC	CONFIGURATION DEPENDENT FEATURE TEST SWITCHES FOR NON-DISK (NON-SWAPPING) SYSTEM
	FT50S,MAC	CONFIGURATION DEPENDENT FEATURE TEST SWITCHES FOR DISK (SWAPPING)
	FTTM10,MAC	FEATURE TEST SWITCHES FOR TM10B MAG TAPE CONTROLLER AND DF-10 DATA CHANNEL
	CONFIG,MAC	SAMPLE CONFIGURATION DEFINITION FILE (MONGEN OUTPUT - SEE COMMON)
	S4NRL,CCL	ASSEMBLE ENTIRE 10/40N - REL ONLY
	S4NBTH,CCL	ASSEMBLE ENTIRE 10/40D - REL + CRF
	S4DRL,CCL	ASSEMBLE ENTIRE 10/40D - REL ONLY
	S4DBTH,CCL	ASSEMBLE ENTIRE 10/40D - REL + CRF
	S50RL,CCL	ASSEMBLE ENTIRE 10/50 - REL ONLY
	S50BTH,CCL	ASSEMBLE ENTIRE 10/50 - REL + CRF
	S50RLX,CCL	ASSEMBLE ENTIRE 10/50 ON DSK - REL ONLY
	B00TS,MAC	LEVEL D DISK BOOTSTRAP LOADER (LOAD MONITOR FROM DEVIFILE,EXT[P,PN])

TAPE 41 MONITOR SOURCE FILES

```
ETHINT          DATA LINE SCANNER + COMPUTER-COMPUTER-INTERFACE - DEVICE DEPENDENT PART OF TELETYPE SERVICE
CCINT           COMPUTER-COMPUTER-INTERFACE - DEVICE DEPENDENT PART OF TELETYPE SERVICE
CDPSER          CARD PUNCH SERVICE ROUTINE
CDRSRX          CARD READER SERVICE ROUTINE FOR PDP-10(CR-10)
CLKCSS          JOB SCHEDULING ALGORTIHM FOR NON-SWAPPING, I.E., 10-40, SYSTEMS
CLOCK1          CLOCK,CONTEXT SWITCHING AND JOB START AND STOP ROUTINES
        APRINT          HIGH PRIORITY PROCESSOR INTERRUPR ROUTINE
        CLOCK           LOW PRIORITY CLOCK INTERRUPT ROUTINE
        RUNCSS          ROUTINES TO START AND STOP USER JOBS

COMCON          COMMAND DECODER AND SAVEGET ROUTINES
        COMCON          MONITOR COMMAND DECODER AND COMMAND ROUTINES
        COMCSS          COMMON SUBROUTINES USED BY MONITOR COMMANDS
        SAVGET          THE SAVE AND GET MONITOR COMMANDS
COMMON          COMMON DATA STORAGE FOR MONITOR (INCLUDES SAMPLE CONFIG.MAC)
        S.MAC           SYSTEM PARAMETER FILE ASSEMBLED WITH ALL FILES (LISTED HERE ONLY)
        CONFIG.MAC      SAMPLE CONFIGURATION DEFINITION FILE (MONGEN OUTPUT)
CORE1           CORE ALLOCATION AND SHUFFLING
DISSER          CRT DISPLAY SERVICE ROUTINE FOR MODEL 340 OR TYPE 30 DISPLAY
DLSINT          DATA LINE SCANNER - DEVICE DEPENDENT INT. SERV. FOR USE WITH TELETYPES
S4NRL4.CCL      ASSEMBLE THIS TAPE FOR 10/40N - REL ONLY
S4DRL4.CCL      ASSEMBLE THIS TAPE FOR 10/40D - REL ONLY
S5DRL4.CCL      ASSEMBLE THIS TAPE FOR 10/50 - REL ONLY
```

TAPE 51 MONITOR SOURCE FILES

```
DPCINT          [LEVEL C]
DPOINT          [LEVEL C]
DSKSER          [LEVEL C]
DTASRN          DECTAPE SERVICE FOR PDP-10(TU-55) DECTAPES AND NEW FILE STRUCTURE
EDDT            EXECUTIVE MODE DDT (DYNAMIC DEBUGGING TECHNIQUE)
ERRCON          MONITOR DETECTED ERROR MESSAGE ROUTINES
JOBDAT          SYMBOL DEFINITIONS FOR JOB DATA AREA (BOTH SOURCE CODE AND ASSEMBLY LISTINGS)
LPTSER          LINE PRINTER SERVICE ROUTINE
MOVIE           SNAPSHOT OF SYSTEM IS PRINTED ON LPT BY THIS ROUTINE
MTASRX          MAGTAPE SERVICE ROUTINE FOR PDP-10 MAGTAPE CONTROLLER(TTM10A+TM10B)
NULSEG          DUMMY HIGH USER SEGMENT HANDLING ROUTINES
ONCE            ONCE ONLY OPERATOR DIALOGUE FOR MONITOR START-UP(NON-DISK OR DISK)
PATCH           PATCHING SPACE
PLTSER          CALCOMP PLOTTER SERVICE ROUTINE
PTPSER          PAPER TAPE PUNCH SERVICE ROUTINE
PTRSER          PAPER TAPE READER SERVICE ROUTINE FOR PDP-10(PDP-6)
PTYSRF          PSEUDO-TELETYPE SERVICE ROUTINE (FULL DUPLEX)
SCHED1          SCHEDULAR AND SWAPPER ROUTINES
S4NRL5,CCL      ASSEMBLE THIS TAPE FOR 10/40N - REL ONLY
S4DRL5,CCL      ASSEMBLE THIS TAPE FOR 10/40D - REL ONLY
S50RL5,CCL      ASSEMBLE THIS TAPE FOR 10/50 - REL ONLY
```

TAPE 61 MONITOR SOURCE FILES

```
SCNSRF          TELETYPE SERVICE - NEW FULL DUPLEX - SCANNER INDEPENDENT (USES CCIINT, OR DLSINT)
SEGCON          HIGH USER SEGMENT HANDLING ROUTINES
SYSINI          MONITOR INITIALIZATION
SYSMAK          MAKE JOB 1 BECOME THE NEW MONITOR (OVERLAY EXISTING MONITOR)
TMPUUO          IN CORE STORAGE ROUTINE FOR CCL FILES (TMPCOR UUO)
UUOCON          UUO TRAP HANDLER AND DEVICE INDEPENDENT UUO ROUTINES
        UUOCON          UUO TRAP HANDLER AND DEVICE INDEPENDENT UUO ROUTINES
        IOCSS           COMMON IO SUBROUTINES
S4NRL6,CCL      ASSEMBLE THIS TAPE FOR 10/40N - REL ONLY
S4DRL6,CCL      ASSEMBLE THIS TAPE FOR 10/40D - REL ONLY
S50RL6,CCL      ASSEMBLE THIS TAPE FOR 10/50 - REL ONLY
```

TAPE 7: 10/50 MONITOR REL FILES

COMMON.REL	COMMON DATA STORAGE (SAMPLE SYSTEM CONFIGURATION)
COMMOD.REL	LEVEL D COMMON DISK DATA STORAGE (SAMPLE DISK CONFIGURATION)
SS01.REL	10/50 RELOCATABLE BINARY LIBRARY FILE

TAPE 8: MONITOR SUPPORT CUSPS

ALCFIL.MAC	ALLOCATE FILE SPACE
ALCFIL.SAV	
CHKPNT.MAC	CHECK POINT CURRENT CHARGE FILE AND START NEW ONE
CHKPNT.SAV	
ACCT.SYS	SAMPLE PASSWORD FILE READ BY LOGIN
AUXACC.SYS	SAMPLE PUBLIC DISK QUOTA FILE READ BY LOGIN
DSKLST.MAC	LIST DISK FILE STRUCTURE (MFD,UFDS,SATS,ETC)
DSKLST.SAV	
FAILSA.MAC	SAVE AND RESTORE DISK USING MAGTAPE
FAILSA.SAV	
LOGIN.MAC	VALIDATE PASSWORDS AND CONTROL SYSTEM ACCESS
LOGIN.SHR	
LOGOUT.MAC	LOG USER OFF SYSTEM
LOGOUT.SHR	
FILDDT.SAV	FILE DDT - EXAMINE A FILE (CRASH.SAV) [ASSEMBLE FROM EDDT.MAC]
C8BTH.CCL	ASSEMBLE THIS TAPE FOR BOTH REL AND LISTING
LOOKFL.MAC	LOOK AT A FILE'S EXTENDED LOOKUP/ENTER ARGUMENTS
LOOKFL.SAV	
GRIPE.MAC	ENTER USER GRIPES IN [3,3] UFD AS A FILE
GRIPE.SAV	

TAPE 9: MONITOR SUPPORT CUSPS

MONEY.MAC	LIST CHARGE FILES AND PRINT TOTALS
MONEY.SAV	
OMOUNT.MAC	PROVIDE OPERATOR INTERFACE FOR FILE AND MOUNT COMMANDS
OMOUNT.SAV	
PLEASE.MAC	PROVIDE OPERATOR CONVERSATION (PLEASE COMMAND)
PLEASE.SAV	
PRINT.MAC	ENTER FILE NAMES IN PRINTR QUEUE FOR LPT
PRINT.SAV	
PRINTR.MAC	OPERATOR CUSP TO PRINT FILES FROM QUEUE
PRINTR.SAV	
REACT.MAC	PREPARE PASSWORD AND QUOTA FILES (ACCT.SYS,AUXACC.SYS,QUOTA.SYS,STRLST.SYS)
REACT.SAV	
SETSRC.MAC	SET AND/OR PRINT JOB FILE STRUCTURE SEARCH LIST
SETSRC.SAV	
SYSTAT.MAC	TYPE SYSTEM SUMMARY ON TTY (SYSTAT COMMAND)
SYSTAT.SAV	
UMOUNT.MAC	ENTER FILE NAMES (FILE COMMAND) OR REMOVABLE VOLUME NAMES (MOUNT COMMAND) IN OPERATOR QUEUE
UMOUNT.SAV	
C9BTH.CCL	ASSEMBLE THIS TAPE FOR BOTH REL AND LISTING
NOTICE.TXT	SAMPLE NOTICE OF THE DAY PRINTED BY LOGIN
FILEX.MAC	
FILEX.SAV	CORE IMAGE PROCESSING PROGRAM

TAPE 10: DOCUMENTATION

DSK016.MEM	LEVEL D&E DISK PROJECT SPECIFICATION
SYSTAT.MEM	DESCRIPTION OF SYSTAT COMMAND PRINTOUT
FAILSA.DOC	DOCUMENTATION FOR LEVEL D FAILSAFE
ERROR.FRM	SOFTWARE ERROR FORM (CASH AND OTHER MONITOR AND CUSP ERRORS)
UNRMCO.FRM	UNRELEASED MONITOR CHANGE ORDER FORM (ABBREVIATED MCO).
CSPSUB.FRM	CUSP SUBMISSION FORM FOR UPDATING SYS
BOOTS.MEM	DOCUMENTATION FOR BOOTS
LOOKFL.MEM	DOCUMENTATION FOR LOOKFL
GRIPE.MEM	DOCUMENTATION FOR GRIPE
DATDMP.MEM	DOCUMENTATION FOR DATDMP
DSKRAT.MEM	DOCUMENTATION FOR DSKRAT
TSTRUN.FRM	SOFTWARE TEST RUN RESULTS FORM
FILEX.MEM	DOCUMENTATION FOR FILEX
QUOLST.MEM	DOCUMENTATION FOR QUOLST
PLEASE.MEM	DOCUMENTATION FOR PLEASE
UMOUNT.MEM	DOCUMENTATION FOR UMOUNT
OMOUNT.MEM	DOCUMENTATION FOR OMOUNT

TAPE 11: DOCUMENTATION

MONSUP.MAN	DESCRIPTION OF DC10E AND TMPCOR UUO
ONCE.FLO	ONCE ONLY FLOW LOGIC
FILSER.FLO	LEVEL D FILE SERVICE FLOW LOGIC
REFSTR.FLO	LEVEL D DISK REFRESHER FLOW LOGIC
LEVELD.MEM	GUIDE TO 5 SERIES MONITOR FILE SYSTEM(LEVEL D)

TAPE 12: TEST PROGRAMS

PARTST.MAC	TEST MEMORY PARITY ERROR RECOVERY
FILTST.MEM	DESCRIPTION OF FILTST LANGUAGE
FILTST.MAC	FILE SYSTEM TEST INTERPRETER
TESTS2.MAC	SERIES OF LEVEL D FILE SYSTEM TESTS
SCRIPT.MEM	DESCRIPTION OF HOW TO RUN SCRIPT
SCRIPT.MAC	TEST SYSTEM BY SIMULATING MANY JOBS FOLLOWING A SCRIPT
TOTAL.F4	PRINT SCRIPT STATISTICS
LINFOR.SCP	LINED/FORTRAN SCRIPT
CPYSYS.SCP	COPY SYS1*.SHR,*.SAV UNTIL QUOTA FILLS UP SCRIPT
DELALL.SCP	DELETE ALL FILES IN TEST DIRECTORY SCRIPT
DMPFIL.MAC	DUMP DISK OR DECTAPE BLOCKS IN OCTAL
FRGSWP.SCP	SCRIPT TO FRAGMENT SWAPPING SPACE
DMPFIL.MEM	DESCRIPTION OF DMPFIL PROGRAM
TECMAC.SCP	TECO/MACRO SCRIPT
PARIO.SCP	
DSKC.SAV	PRINT NO. OF BLOCKS LEFT ON FILE STRUCTURE DSK
DSKC0.SAV	PRINT NO. OF BLOCKS LEFT ON UNIT DSKC0
C12BTH.CCL	ASSEMBLE THIS TAPE FOR BOTH REL AND LISTING
DSKRAT.MAC	EXAMINE A LEVEL D FILE STRUCTURE FOR ERRORS
DSKC1.SAV	PRINT NO. OF BLOCKS LEFT ON UNIT DSKC1
QUOLST.MAC	TYPE JOB'S QUOTAS ON ALL FILE STRUCTURES IN SEARCH LIST
FILLUP.MAC	FILLUP DISK TO TEST THIS CONDITION

TAPE 13: LEVEL D CUSP TAPE

```
FAILCD.MAC      RESTORE LEVEL C MAGTAPE TO LEVEL D DISK
FAILCD.SAV
FAILCD.DOC      DOCUMENTATION FOR FAILCD
FAILDC.MAC      SAVE LEVEL D DISK ON LEVEL C MAGTAPE
FAILDC.SAV
FAILDC.DOC      DOCUMENTATION FOR FAILDC
C13BTH.CCL      ASSEMBLE THIS TAPE FOR BOTH REL AND LISTING
```

TAPE 14: LEVEL D MONITOR FILE SYSTEM SOURCES

```
COMMOD.MAC      COMMON DATA BASE FOR DISK DATA
        S.MAC           SYSTEM PARAMETER FILE ASSEMBLE WITH ALL FILES [LISTED WITH COMMON ONLY]
        CONFIG.MAC      SAMPLE CONFIGURATION DEFINITION FILE (MONGEN OUTPUT)
        COMMOD.MAC      LEVEL D FILE SYSTEM DATA BASE
        DATDMP.MAC      LEVEL D FILE SYSTEM DATA BASE DUMPER
DATDMP.MAC      DATA BASE DUMPER (EXEC OR USER MODE)
ONCMOD.MAC      DISK ONCE ONLY DIALOG
REFSTR.MAC      DISK REFRESHER
SWPSER.MAC      LEVEL D SWAPPER INTERFACE ROUTINES
S4DR14.CCL      ASSEMBLE THIS TAPE FOR 10/40D - REL ONLY
S50R14.CCL      ASSEMBLE THIS TAPE FOR 10/50 - REL ONLY
```

TAPE 15: LEVEL D MONITOR FILE SYSTEM SOURCES

```
FILSER.MAC      DISK INDEPENDENT FILE SYSTEM
KONPAR.MAC      CONTROLLER PARAMETER FILE
DPXKON.MAC      RP10 KONTROLLER(S) ROUTINE
FHXKON.MAC      RC10 KONTROLLER(S) ROUTINE
MDXKON.MAC      RA10 KONTROLLER(S) ROUTINE
S4DR15.CCL      ASSEMBLE THIS TAPE FOR 10/40D - REL ONLY
S50R15.CCL      ASSEMBLE THIS TAPE FOR 10/50 - REL ONLY
```

TAPE 16: 680/680I COMMUNICATIONS

```
S680I.PAL       PDP-8I ROUTINES FOR 680 USED WITH PDP-10
S680I.LST       LISTING OF PDP-8I 680 ROUTINES
PAL10.141       SOURCE OF PAL ASSEMBLER (PDP-8) TO RUN ON PDP-10.
PAL10.OPR       PAL ASSEMBLER INSTRUCTIONS
PAL10.SAV       PAL ASSEMBLER (PDP-8) TO RUN ON PDP-10.
```

TAPE 17: MORE MONITOR TEST PROGRAMS

```
WORKER.SCP      SCRIPT TO SIMULATE IN-HOUSE T.S. SYSTEM
UPDATE.SAV      PROGRAM USED IN SCRIPT TO TEST SIM. UPDATE
UPDATE.SCP      SCRIPT TO TEST SIMULTANEOUS UPDATE
LOGLOG.SCP      SCRIPT TO TEST SIMULTANEOUS LOGIN/LOGOUT
FILER.SCP       SCRIPT TO TEST USE OF FILE COMMAND.
```

CHANGES LISTED IN ORDER OF MOST RECENT FIRST
FIRST VERSION IN WHICH CHANGE APPEARED WILL BE PUT ON EVERY LINE
SO CHANGED AT LEFT HAND MARGIN .

CHANGE FROM V416 TO V417
DELETED REFERENCES TO PDP-6 AND
HALF DUPLEX SCANNER 29 JUN 70

CHANGE FROM VERSION V415 TO V416 23 JUN 70

TAPE 3: ADDED BOOTS,MAC
TAPE 8: ADDED C8BTH,CCL,LOOKFL,MAC,LOOKFL,SAV,FILEX,SAV
TAPE 9: ADDED PRINTR,MAC,PRINTR,SAV,GRIPE,MAC,GRIPE,SAV,C9BTH,CCL,FILEX,MAC
TAPE 10: ADDED ERROR,FRM,UNRMCO,FRM,CSPSUB,FRM,BOOTS,MEM,
LOOKFL,MEM,GRIPE,MEM,DATDMP,MEM,DSKRAT,MEM,TSTRUN,FRM,QUOLST,MEM
TAPE 12: ADDED LINFOR,SCP,TESTS2,SAV,SCRIPT,MEM,DMPFIL,MAC
DMPFIL,MEM,DMPFIL,SAV,TECMAC,SCP,DD10F,MAC,DD10F,SAV
DD10F,MEM,C12BTH,CCL,DSKRAT,MAC,DSKRAT,SAV,QUOLST,MAC,QUOLST,SAV
REMOVED SCR10S
TAPE 13: ADDED FAILDC,MAC,FAILDC,SAV,FAILDC,DOC,C13BTH,CCL,PIP030,DOC
TAPE 17: ADDED THIS WHOLE TAPE OF TEST PROGRAMS

CHANGE FROM VERSION V414 TO V415 6 MAY 70

TAPE 8: ADDED FILDDT,SAV
TAPE 12: ADDED SCR10S

THE TAPES WERE ENTIRELY REORGANIZED FOR 5 SERIES MONITOR,
THE TAPE NUMBERS WERE ARRANGED TO BE THE SAME AS THE LIBRARY TAPE NUMBERS
FOR ORDERING TAPES

CHANGE FROM VERSION V411 TO V412 12 JAN 70

1. REMOVE IJOBS FROM TAPE 5
2. REMOVE RENMON,MAN,MONSPL,MAN AND RENMON,S01 FROM TAPE 8,
3. REMOVE S,MAC AND COMMON,MAC FROM TAPE 9,
4. CHANGE 120,16K TO 121,10K ON TAPE 9,

CHANGE FROM VERSION V410 TO V411 11 JAN 70

1. ADDED TABLE OF CONTENTS OF A DEC FURNISHED LISTING FOR THIS MONITOR,
2. REMOVED SYSTEM MAP AND GLOB,XRF FROM TAPE 1,

CHANGE FROM VERSION V407 TO V410 6 JAN 70

1. MOVED ALL DISK RELATED ROUTINES TO TAPE 4,
2. ADDED IJOB01,02,04 - THE BATCH MONITOR MAKER
3. ADDED TAPE 7 WITH PDP-8I 680 ROUTINES + PAL10
4. RENAMED LIBRARY FILES
5. ADDED LEVELC,MEM, SYSTAT,MEM TO MONITOR DOCUMENTATION
6. ADDED OPFILE,UFILE,PLEASE TO SUPPORT CUSPS
7. ADDED PAKLOD - UTILITY DISK PACK LOADER

CHANGED FROM VERSION V406 (-00) 01 JUN 69 TO V407 (-00) 30 JUN 69

TAPE 1, REMOVED MONITR,OPR BECAUSE RUNNING OUT OF ROOM

TAPE 6, ADDED MONITR,OPR

END OF FILE TABLE.TXT

100-100-007-02

"MONITR.OPR"

MONITOR ASSEMBLY AND LOAD INSTRUCTIONS

TOM HASTINGS/TL/RAP/PEC 08 JUL 1970

V017

CHANGES LISTED IN ORDER MOST RECENT FIRST
FIRST VERSION IN WHICH CHANGE APPEARED WILL BE PUT ON EVERY LINE
SO CHANGED.

CHANGE FROM VERSION V015(-01) TO VERSION V016 (-02) 06 JUL 70

1.5.2 ADDED HOW TO GET ON THE AIR FROM MAGTAPE

2.0.2.2.0.3.2.0.4 ADDED HOW TO MAKE MONITOR FROM MAGTAPE

3.1.2 ADDED HOW TO ASSEMBLE ENTIRE MONITOR ON DSK WITH ONE CCL FILE

6.1 ADDED CRASH PROCEDURE TO WRITE ON DISK WITH BOOTS

CHANGE FROM VERSION V014(-01) TO VERSION V015(01)

CHANGE FROM VERSION V013(-02) TO VERSION V014(-01) 12 APR 70
CONVERT TO LEVEL D.

CHANGE FROM VERSION V012(-00) TO VERSION V013(-00) 13 MAR 70

4.2.5 SAVE (DUMP) PROCEDURE CORRECTED WITH ADDITION OF LINE
"140$" BEFORE D$4S45S SAV

CHANGES FROM VERSION V010(-00) TO VERSION V011(-00) 6 JAN 70

1.3.3 REVISED REFRESH PROCEDURE

3.1.9.1 DESCRIPTION OF MONITOR SOURCES

CHANGES FROM VERSION V007(-00) TO VERSION V010(-00) 30 JUN 69

6. HOW TO SAVE A MONITOR CRASH AND DUMP SYMBOLICALLY WITH FILDDT
ADDED THIS NEW SECTION

QUICK INDEX TO MONITOR.OPR

[ ]'S INDICATE STEPS NEEDED FOR DISK SYSTEMS (10/40D AND 10/50)
WHICH ARE NOT NEEDED FOR NON-DISK SYSTEMS (10/40N).

3. HOW TO ASSEMBLE MONITOR SOURCES, CREATE LIBRARY FILE, TO PRODUCE A NEW MONITOR

3.0 REQUIRED COMPONENTS

3.1 ASSEMBLE SOURCES USING MACRO

3.1.1-3.1.6 ASSEMBLE ALL SOURCES AUTOMATICALLY WITH CCL FILES

3.1.7 ASSEMBLE EACH SOURCE BY HAND

3.2 COMBINE RELOCATABLE BINARIES TO MAKE LIBRARY FILE USING PIP

3.3 LOAD MONITOR USING LOADER

3.4 SAVE MONITOR USING MONITOR SAVE COMMAND

4. HOW TO MAKE DDT PATCHES TO YOUR MONITOR

4.0 REQUIRED COMPONENTS

4.1 PATCHING WITH EXEC DDT IN USER MODE WHILE TIME SHARING

4.2 PATCHING WITH EXEC DDT STAND-ALONE EXEC MODE

4.3 PATCHING CONVENTIONS

5. HOW TO ASSEMBLE MONITOR SUPPORT CUSPS

6. HOW TO SAVE A MONITOR CRASH AND DUMP SYMBOLICALLY WITH FILDDT

```
1.      HOW TO GET ON THE AIR WITH AN EXISTING TIME SHARING
        MONITOR ON A DECTAPE, CUSPS ON DECTAPE [OR MAGTAPE]

        1.Ø COMPONENTS REQUIRED

        1.Ø.Ø A SAV COPY OF THE MONITOR ON A DECTAPE

        1.Ø.1 A CLEARED MACHINE [AND A DISK]

        1.Ø.2  "16K [OR 32K OR 48K] TENOMP" ON PAPER TAPE (MAYBE NEEDED)

V17     IF CUSPS AND MONITOR SUPPORT CUSPS ARE ON DECTAPE!

V17       1.Ø.3.Ø  MONITOR SUPPORT CUSP TAPE (TAPE #8)
V17       1.Ø.3.1  CUSP SAV FILES 1
V17       1.Ø.3.2  CUSP SAV FILES 2
V17       1.Ø.3.3  MONITOR SUPPORT CUSPS (TAPE #9)
V17       1.Ø.3.4  TEST PROGRAMS (TAPE #12)
V17       1.Ø.3.5  LEVEL D CUSP TAPE(TAPE #13)
V17       1.Ø.3.6  MORE MONITOR TEST PROGRAMS (TAPE #17)

V17     [IF CUSPS AND MONITOR SUPPORT CUSPS ARE ON MAGTAPE!

V17       1.Ø.4.Ø  FAILCD.SAV ON A DECTAPE(YOUR MONITOR TAPE TAPE # 1)
V17       1.Ø.4.1  CUSPS ON A MAGTAPE (LEVEL C FORMAT)
V17       1.Ø.4.2  MONITOR SUPPORT CUSPS ON A MAGTAPE(LEVEL C FORMAT)
V17     ]
```

1.1 READ-IN YOUR MONITOR FROM DECTAPE USING TENDMP

```
V10 BRIEF STEP BY STEP OPERATOR INSTRUCTIONS TO LOAD MONITOR FROM
V10 DECTAPE WITH TENDMP,  THESE ARE BRIEF ENOUGH TO BE PLACED ON THE
    COMPUTER CONSOLE,  THE NEXT FEW PAGES DESCRIBE THIS PROCESS
    IN MORE DETAIL.

V10 1.  PUT MONITOR TAPE ON DECTAPE UNIT 0 (DIALED AS 8)
V10 2.  PUSH STOP DOWN(CONSOLE SWITCH)
V10 3.  PUSH RESET DOWN(CONSOLE SWITCH)
V10 4.  PUSH NON EX MEM UP(CONSOLE SWITCH)
V10 5.  PUSH MEMORY PARITY UP(CONSOLE SWITCH)(UNLESS YOU HAVE HAD MEM
V10     PROBLEMS RECENTLY)
V10 6.  PUT UNIT   ON WRITE LOCK(DECTAPE UNIT SWITCH)
V10 7.  SET READ-IN DEVICE TO 320(SWITCHES 4,5, AND 7 DOWN, REST UP
V10     ON MAINTENANCE PANEL JUST ABOVE DIGITAL PDP-10 NAME PLATE ON
V10     CONSOLE)
V10 8.  PUSH READ-IN(CONSOLE SWITCH)
V10 9.  UNIT 0 SHOULD MOVE,  IF IT DOES, GO ON TO STEP 10 BELOW
V10     IF PC JUST COUNTS UP, LOAD TENDMP
V10     FROM PAPER TAPE READER:
V10     A. PUSH STOP
V10     B. PUSH RESET
V10     C. PUT TENDMP TAPE IN READER
V10     D. SET READ-IN SWITCHES TO 104(SWITCHES 5 AND 9 DOWN ON MAINT-
V10        ENANCE PANEL JUST ABOVE DIGITAL PDP-10 NAME PLATE ON CONSOLE)
V10     E. PUSH READ-IN
V10     F. PAPER TAPE SHOULD MOVE
V10     G. TENDMP WILL RESPOND WITH CARRIAGE RETURN WHEN LOADED IN
V10     H. READ DIRECTORY BY TYPING:
V10             <ALT-MODE>
V10 10. THE TENDMP ON FRONT OF TAPE AUTOMATICALLY READS IN DIRECTORY
V10 11. LIST THE DIRECTORY (UNNECESSARY IF YOU KNOW ITS CONTENTS) BY TYPING
        THE FOLLOWING ON THE CONSOLE TELETYPE (CTY):
V10     F<ALT-MODE>
V15 12. TYPE XXXMON SAV<CR> TO LOAD FILE XXXMON SAV.  THE DECTAPE
V15     WILL SPIN AND START UP THE MONITOR.  TENDMP WILL TYPE A BELL
V15     IF IT CANNOT FIND THE FILE TYPED IN.
V10 13. THE MONITOR WILL PRINT OUT ITS NAME, DATE, AND VERSION NUMBER,
V10     THEN IT WILL ASK FOR TODAY'S DATE.  TYPE IT IN AS MM-DD-YY
V10     FOLLOWED BY A CARRIAGE RETURN.  IF THE MONITOR DISCOVERS
V10     SYNTAX ERROR, IT WILL REPEAT QUESTION.
V10 14. THEN THE MONITOR WILL ASK FOR THE TIME OF DAY.  TYPE IT IN AS
V10     A FOUR DIGIT 24-HOUR TIME HHMM FOLLOWED BY A CARRIAGE RETURN.
V10 15. MONITOR IS READY TO TIME SHARE.
V10     THE PROGRAM COUNTER LIGHTS ON THE CONSOLE WILL BE 1(THE NULL
V10     JOB) AND ACCUMULATOR 0 WILL COUNT BY ONES.
V15 16. PUSH NXM SWITCH DOWN (UNLESS YOU WISH THE MONITOR TO ATTEMPT
V15     TO RECOVER AUTOMATICALLY).
```

EXPLANATION OF INSTRUCTIONS TO READ-IN YOUR MONITOR WITH TENDMP.
YOU HAVE BEEN PROVIDED WITH A MONITOR WHICH HAS BEEN BUILT FOR AND TESTED ON YOUR MACHINE, (NEW MACHINES ONLY). THIS MONITOR IS ON A DECTAPE LABELED "YOUR MONITOR". TO LOAD THIS MONITOR:

1.1.1 MOUNT THIS TAPE ON DECTAPE DRIVE 8 (REFERRED TO IN OCTAL AS DTA0 - IN ADDRESSING THIS DRIVE THROUGH THE TELETYPE, IT WILL ALWAYS BE CALLED DTA0 ALTHOUGH THE DRIVE IS DIALED TO 8).

NOTE: NO OTHER DRIVE SHOULD BE DIALED TO 8.

1.1.2 ASSUME THAT TENDMP HAS BEEN WRITTEN ON BLOCKS 0 AND 1 OF THE DECTAPE. ON THE MAINTENANCE PANEL JUST ABOVE THE DIGITAL PDP-10 NAMEPLATE IS A ROW OF SWITCHES NUMBERED FROM 3 THROUGH 9. SET SWITCHES 4, 5, AND 7 DOWN (BOTTOM PART PUSHED IN) AND ALL THE REST UP (TOP PART PUSHED IN)(DTA DEVICE CODE=320).

1.1.3 ON THE LOWER LEFT SIDE OF THE CONSOLE IS A ROCKER SWITCH LABELED NXM STOP. TURN OFF THIS SWITCH (FAR END DOWN). PRESS THE FOLLOWING ROCKER SWITCHES, LOCATED ON THE LOWER PANEL OF THE CONSOLE ON THE LEFT-HAND SIDE, IN THE ORDER GIVEN.

STOP RESET READ IN

DECTAPE UNIT 0 SHOULD MOVE. IF PROGRAM COUNTER LIGHTS (TOP ROW ON RIGHT HAND SIDE OF CONSOLE) START TO COUNT UP, THERE WAS NO TENDMP ON FRONT OF TAPE (USE COPY CUSP /T SWITCH OR PIP /U SWITCH), SO PUSH RESET TO STOP THE RUNAWAY TAPE AND PROCEED TO STEP 1.1.3.1. OTHERWISE THE DECTAPE WILL READ IN THE LOADER PROGRAM TENDMP. WHEN FINISHED, THE TAPE WILL STOP, AND THE TELETYPE WILL MAKE AN AUDIBLE CLICK (PROVIDED IT WAS TURNED ON), THEN PROCEED TO STEP 1.1.4

1.1.3.1 ON THE MAINTENANCE PANEL JUST ABOVE THE DIGITAL PDP-10 NAMEPLATE IS A ROW OF SWITCHES NUMBERED FROM 3 THROUGH 9. SET SWITCHES 5 AND 9 DOWN; SET ALL THE REST UP (PTR DEVICE CODE=104).

1.1.3.2 LOAD INTO THE PAPER TAPE READER THE TAPE LABELED "TENDMP", INCLUDED IN YOUR SOFTWARE PACKAGE, PRESS THE FOLLOWING ROCKER SWITCHES, LOCATED ON THE LOWER PORTION OF THE CONSOLE ON THE LEFTHAND SIDE, IN THE ORDER GIVEN,

STOP RESET READ IN

THE PAPER TAPE WILL READ AND LOAD THE LOADER PROGRAM TENDMP, WHEN FINISHED, THE TAPE WILL STOP, AND THE TELETYPE WILL MAKE AN AUDIBLE CLICK,

NOTE- A WRITEUP OF TENDMP IS INCLUDED IN THIS PACKAGE,

1.1.4 WHEN THE CONSOLE TELETYPE RESPONDS WITH A CLICK, TYPE IN

<ALT-MODE>

NOTE- ON MODEL 37 TELETYPES, THE ALTMODE KEY IS LABELED PREFIX,

DECTAPE 0 (DIALED 8) WILL SPIN FOR A FEW SECONDS, AND THE DIRECTORY WILL BE READ INTO CORE, WHEN IT HAS STOPPED, TYPE

F <ALT-MODE>
("F" FOR FILE NAMES)

THE DIRECTORY OF THE TAPE WILL BE PRINTED,

1.1.5 IN THIS DIRECTORY LISTING WILL APPEAR THE NAME OF YOUR MONITOR, IN THE FORM

XXXMON SAV

WHERE THE FIRST THREE LETTERS (XXX) ARE AN ATTEMPT TO REPRESENT YOUR COMPANY'S NAME (E.G., DECMON SAV), TYPE IN THIS NAME (FOLLOWED BY A CARRIAGE RETURN), EXACTLY AS IT APPEARS ON THE LISTING, ON THE CONSOLE TELETYPE, (DO NOT SEPARATE NAME FROM EXTENSION WITH A PERIOD,) THIS ACTION CAUSES DECTAPE 0 TO SPIN AS THE MONITOR IS READ AND LOADED, TENDMP WILL RESPOND WITH A BELL IF IT CANNOT FIND THE FILE OR GETS A PARITY ERROR WHILE READING,

1.2 MONITOR ONCE ONLY OPERATOR DIALOG

1.2.1 THE MONITOR WILL REQUEST THE DATE AND TIME.

TYPE TODAY'S DATE AS ABOVE
MM-DD-YY<CR> (MM = MONTH; DD = DAY; YY = YEAR;
E.G., 12-3Ø-68) (<CR> IS A CARRIAGE
RETURN)

TYPE A 4-DIGIT TIME
TTTT (TTTT = TIME IN 24-HOUR FORMAT;
E.G., 1435 = 2:35 PM)
SEE NOTE BELOW

IF AFTER PERFORMING STEP 1.1.5, THE OPERATOR IS NOT
ASKED BY THE MONITOR FOR DATE AND TIME:

V15 (Ø) CHECK THAT YOU PUT NXM STOP UP (FRONT). IF YOU DID
V15 NOT, PUT IT UP (BACK SIDE DOWN), THEN PUSH CONTINUE
V15 AND GO ON TO STEP 1.2.2.

(1) SET THE PROGRAM COUNTER SWITCHES TO ØØØ14Ø(8).

(2) PRESS STOP RESET START (IN THAT ORDER).

(3) RESPOND TO DATE AND TIME QUERIES.

NOTE- THE DATE AND TIME RESPONSES MUST BE TERMINATED BY A
CARRIAGE RETURN UNLESS YOURS IS A 1Ø-5Ø SWAPPING SYSTEM
OR A 1Ø/4Ø SYSTEM WITH DISK. IN EITHER OF THESE CASES,
YOU WILL WANT TO GO THROUGH THE COMPLETE INITIALIZA-
TION DIALOGUE IN ORDER TO DEFINE YOUR FILE STRUCTURE(S)
AND "REFRESH" (INITIALIZE THE FILE STRUCTURE(S) OF) THE
DISK(S). THE COMPLETE DIALOGUE IS ENABLED BY TYPING
ALTMODE INSTEAD OF A CARRIAGE RETURN AFTER THE FOUR DIGITS
OF TIME.

1.2.2 IF YOUR SYSTEM CONTAINS A DISK SUCH THAT YOU TYPED ALTMODE AFTER TIME-OF-DAY, GO TO STEP 1.3. OTHERWISE (YOU TYPED A CARRIAGE RETURN), YOUR MONITOR IS ALREADY RUNNING.

IT IS ADVISABLE TO RUN THE SYSTEM WITH THE NXM STOP SWITCH DOWN SO THAT THE MONITOR WILL STOP IF IT TRIES TO REFERENCE NON-EXISTENT MEMORY (OR A MEMORY BECOMES INOPERATIVE), HOWEVER THE MONITOR WILL TRY TO RECOVER FROM SUCH A PROBLEM AS BEST IT CAN, IF THE NXM SWITCH IS NOT DOWN OR IF THE OPERATOR PUSHES CONTINUE AFTER A MEMORY STOP CAUSED BY A NON-EXISTANT MEMORY REFERENCE. USUALLY ONLY ONE JOB WILL BE AFFECTED (VERY LIKELY IF NO PI'S IN PROGRESS) AND THE USER WILL RECEIVE AN "?ERROR IN JOB" NXM AT EXEC NNNNNN) UUO AT USER MMMMMM.

IT IS ADVISABLE TO RUN WITH PAR STOP SWITCH UP (NEAR SIDE UP), RATHER THAN DOWN SINCE MACHINE WILL RUN ABOUT 10
V15 PERCENT FASTER. IF A MEMORY PARITY ERROR OCCURS, THE
V15 MONITOR WILL USUALLY BE ABLE TO RECOVER. THE MONITOR
V15 WILL HALT IF THE PARITY ERROR OCCURRED WHILE A PI WAS
V15 IN PROGRESS OR THE MACHINE WAS IN EXEC MODE SINCE THIS
V15 IS DIFFICULT TO RECOVER FROM RELIABLY. IF THE ERROR
V15 OCCURRED IN USER MODE, THE MONITOR SWEEPS THROUGH ALL OF
V15 CORE. IF IT FINDS EXACTLY ONE PARITY ERROR AGAIN (USUAL)
V15 IT STOPS THE JOB AND PRINTS AN ERROR MESSAGE ON THE
V15 USERS CONSOLE AFTER ATTEMPTING TO FIX THE ERROR BY
V15 RESTORING THE VALUE IT READ OUT OF MEMORY. IF NO
V15 ERRORS OCCUR DURING THE SWEEP, IT IS COUNTED AS A
V15 SPURIOUS ERROR AND THE CURRENT USER CONTINUES. IF
V15 MORE THAN ONE ERROR OCCURS, THE MONITOR HALTS SINCE
V15 A WIDESPREAD MEMORY PROBLEM IS LIKELY. A COUNT OF
V15 THE NUMBER OF PARITY ERRORS IS KEPT ALONG WITH THE
V15 LOCATION OF THE LAST ERROR (SEE GETTAB UUO).

BY SETTING THE ADDRESS SWITCHES (LOCATED IN THE UPPER ROW OF SWITCHES, ON THE RIGHT SIDE OF THE CONSOLE) TO ZERO, THE USER CAN OBSERVE THE NULL JOB
V15 COUNTING. EVERY TIME THE NULL JOB IS STARTED UP AFTER
V15 ANOTHER JOB HAS BEEN RUNNING, AC 0 IS RESET TO ZERO.
V15 BY WATCHING AC 0 COUNT, AN ESTIMATE OF NULL TIME BURTS
V15 CAN BE ESTIMATED.

1.2.3 REMOVE THE MONITOR TAPE FROM DRIVE DTA0 AND MOUNT ON THE SAME DRIVE THE TAPE IN YOUR PACKAGE LABELED "CUSP SAV FILES".

TYPE ↑C (HOLD DOWN THE CTRL KEY WHILE STRIKING C), THE MONITOR WILL RESPOND WITH THE ECHO "↑C" AND START A NEW LINE WITH A ","
THEN TYPE

.ASSIGN DTA0 <CR> THIS INFORMS THE MONITOR THAT THE CUSP TAPE IS ON DECTAPE 0 WHERE IT WILL RESIDE.

1.2.4 SINCE YOURS IS A NONDISK SYSTEM, YOU ARE FINISHED AND MAY PROCEED TO USE YOUR TIME-SHARING PDP-10 SYSTEM.

1.3 MONITOR DISK FILE STRUCTURE DEFINITION AND REFRESH DIALOG (DISK SYSTEMS ONLY)

1.3.1 UPON RECEIVING AN ALTMODE FOLLOWING TYPEIN OF THE TIME-OF-DAY, YOUR MONITOR WILL TYPE A SUMMARY OF THE I/O CONFIGURATION FOR WHICH IT WAS BUILT, AND WILL PROCEED TO ASK FOR THE OPERATOR'S CONSOLE NAME. THE DEVICE WHICH YOU SPECIFY WILL BECOME THE ONE TO WHICH MESSAGES FOR LOGICAL DEVICE "OPR" WILL BE SENT. A RECOMMENDED RESPONSE TO THIS QUESTION IS

TTY <CR> (<CR> IS A CARRIAGE RETURN)

1.3.2 NEXT, YOU WILL BE GIVEN THE OPPORTUNITY TO RETAIN
V14 IN CORE THREE OPTIONAL MONITOR COMPONENTS CALLED MOVIE, SYSMAK,
V14 AND EXEC DDT. THESE ARE USEFUL FOR EXAMINING THE CURRENT MONITOR, FOR CREATING NEW MONITORS, AND FOR DEBUGGING THE CURRENT MONITOR, RESPECTIVELY; THEY ARE USUALLY NOT RETAINED.

A RESPONSE OF CARRIAGE RETURN WILL DELETE THESE ROUTINES AND MAKE THE SPACE THEY OCCUPY AVAILABLE AS USER CORE.

V14 YOUR ANSWERS TO THESE THREE QUESTIONS WILL DETERMINE THE FINAL SIZE OF THE MONITOR (THIS SIZE IS TYPED AT THE END OF THE ONCE ONLY DIALOG.

V14 1.3.3 NEXT YOU MUST SET UP THE DISK FILE STRUCTURE. FOR
V14 DETAILS SEE LEVELD,MEM SECT. 4,2, AND DSKØ16,MEM
V14 SECT. 4,2 AND 4,3.

1.3.4 IT IS COMMON PRACTICE TO INCLUDE THE ACCESS PROTECTION FEATURE CALLED LOGIN IN ALL SYSTEMS CONTAINING DISKS.
V14 THE LOGIN MECHANISM REQUIRES THE PRESENCE OF THREE FILES ON THE DISK: THE LOGIN CUSP (LOGIN,SAV) AND THE SYSTEM
V14 ACCOUNTING FILES (ACCT,SYS AND AUXACC,SYS). ALTHOUGH YOUR MONITOR IS NOW RUNNING, THESE REQUIRED FILES ARE MISSING FROM THE NEWLY REFRESHED DISK. FOR THIS REASON, AN AUTOMATIC LOGIN IS ALLOWED FOR THE FIRST USER TO TYPE THE LOGIN COMMAND AFTER THE DISK HAS BEEN REFRESHED.

TYPE

LOGIN <CR>

THE SYSTEM RESPONDS WITH

JOB1 <YOUR MONITOR NAME>

YOU ARE NOW LOGGED IN UNDER THE PRIVILEGED FAILSAFE
V14 NUMBERS 1,2 SO THAT YOU MAY WRITE ON ANY DISK AREA
V15 WITH ANY CUSP.

SET NXM SWITCH DOWN, MEM PARITY UP, AND ADDRESS SWITCHES TO 0. SEE SECTION 1.2.2 FOR DISCUSSION OF THESE SWITCHES AND WHEN IT IS ADVISABLE TO SET THEM DIFFERENTLY.

V15 SAV FILES, SEE FILEX.MEM FOR A COMPLETE DESCRIPTION, THE
1.4 COPY CUSPS FROM DECTAPE TO LEVEL D DISK USING FILEX

V15 (SKIP TO SECTION 1.5 FOR MAGTAPE TO DISK COPYING, IF
V15 YOU RECEIVED THE CUSPS ON MAGTAPE IN FAILCD OR FAILSA FORMAT)

V15 FILEX IS A UTILTIY CUSP LIKE PIP WHICH COPIES FILES FROM
V15 DECTAPE TO DISK AND BACK IN AN EFFICIENT MANNER, IT
V15 ALSO HAS A NUMBER OF OTHER FEATURES SUCH AS CONVERSION TO
V15 AND FROM PDP-6 DECTAPE FORMAT AND EXPANSION OF ZERO-COMPRESSED
V15 SAV FILES, SEE FILEX.MEM FOR A COMPLETE DESCRIPTION, THE
V15 FOLLOWING INSTRUCTIONS WILL SUFFICE FOR COPYING THE CUSPS.

V15 AFTER THE DISK FILE STRUCTURE(S) HAVE BEEN REFRESHED, THE
V15 FIRST USER TO TYPE LOGIN IS LOGGED IN AS 1,2 (THE PRIVILEGED
V15 FAILSAFE NUMBERS).

V15 1.4.1 MOUNT THE MONITOR SUPPORT CUSP WHICH CONTAINS FILEX.SAV
V15 (TAPE #8) ON DTA0.

```
V15     1.4.2 TYPE
V15           .ASSIGN DTA0
V15           -
V15           DTA0 ASSIGNED
V15           -------------
V15
V15           .RUN DTA0 FILEX 10
V15           -
V15           *DSKB:[1,4]<155>/Q-DTA0:*,SAV,*,SHR
V15           -
V15
V15           WHERE DSKB IS WHERE YOU WANT YOUR CUSPS STORED BECAUSE
V15           IT IS A LARGE (BUT POSSIBLY SLOWER) FILE STRUCTURE THAN DSKA,
V15           IT CAN BE REFERENCED LATER BY USERS SIMPLY
V15           AS DEV SYS.  THE /Q SWITCH STANDS FOR QUICK AND CAUSES
V15           FILEX TO MAKE JUST ONE PASS OVER THE DECTAPE AND
V15           COPY IT QUICKLY ONTO THE DISK.  THERE UPON IT SORTS THE
V15           SAV AND SHR FILES AND SETS THE PROTECTION TO 155.  THE
V15           1 IS A FLAG TO LOGOUT TO PRESERVE THIS FILE.  THE 55
V15           ALLOWS USER'S IN OTHER PROJECTS THAN PROJECT 1 TO READ
V15           THE CUSP FILES.  THIS IS NECESSARY ONLY IF THE DEFAULT
V15           FILE PROTECTION IS LEFT AT ITS VALUE OF Ø57 AS DISTRIBUTED
V15           BY DIGITAL.  THE CORE ARGUMENT OF 1Ø SPEEDS UP THE TAPE
V15           COPY OPERATION.  A LARGE CORE ARGUMENT MAY BE USED.  IF
V15           NO ARGUMENT IS PRESENT, FILEX WILL EXPAND ITSELF TO A
V15           REASONABLE SIZE FOR USE UNDER REGULAR TIME SHARING
V15           (MUCH LESS THAN 10).

V15           FILEX WILL RESPOND WITH AN * WHEN IT IS THROUGH, JUST
V15           LIKE PIP.  MOUNT THE FOLLOWING TAPES IN ORDER AND TYPE
V15           THE COMMAND STRING INDICATED:

V15           CUSP SAV FILES 1
              *DSKB:[1,4]<155>/Q-DTA0:*,*
V15           CUSP SAV FILES 2
              *DSKB:[1,4]<155>/Q-DTA0:*,*
V15           MONITOR SUPPORT CUSPS (TAPE #9)
              *DSKB:[1,4]<155>/Q-DTA0:*,SAV,*,SHR
V15           TEST PROGRAMS (TAPE #12)
              *DSKB:[1,4]<155>/Q-DTA0:*,SCP,*,SAV,*,HGH,*,SHR
V15           INTERIM LEVEL D CUSP TAPE (TAPE #13)
              *DSKB:[1,4]<155>/Q-DTA0:*,SAV,*,SHR
V15           MORE MONITOR TEST PROGRAMS (TAPE #17)
              *DSKB:[1,4]<155>/Q-DTA0:*,SCP,*,SAV,*,SHR
```

1.5 COPY CUSPS AND MONITOR SUPPORT CUSPS FROM MAGTAPE TO LEVEL D DISK USING FAILCD

1.5.1 IF YOUR PDP-12 CONFIGURATION INCLUDES DISK AND MAGTAPE
YOU WILL PROBABLY HAVE RECEIVED YOUR MONITOR AND CUSP SOFTWARE
V15 ON TWO SEPARATE MAGNETIC TAPES IN FAILCD FORMAT (IE LEVEL C
V15 FAILSAFE FORMAT WHICH CAN BE READ BY A LEVEL C FAILSAFE
V15 ONTO A LEVEL C DISK SYSTEM, OR CAN BE READ BY FAILCD
V15 ONTO A LEVEL D DISK SYSTEM,) MAC, REL, SAV, AND SHR FILES
HAVE ALL BEEN COLLECTED TOGETHER IN UFD 10,7 ON EACH TAPE.

V15 THE FOLLOWING TABLE SHOWS THE DIFFERENT VERSIONS OF
V15 FAILSA CUSPS:

V15	FAILSAFE	TAPE TO	DISK TO	TAPE
V15	FAILSA,V16	C	C	C
V15	FAILCD	C	D	-
V15	FAILDC	-	D	C
V15	FAILSA,V27	D	D	D

V15 NOTE: IT IS SUGGESTED THAT EACH OLD CUSTOMER CHANGE
V15 THE NAME OF THE LEVEL C FAILSAFE CUSP HE ALREADY HAS,
V15 TO FAILCC, TO DISTINGUISH IT FROM THE FAILSA BEING
V15 DISTRIBUTED NOW WHICH IS LEVEL D TO D ONLY.

1.5.2 COPY CUSPS FROM MAGTAPE TO LEVEL 3 DISK

AFTER HAVING LOADED THE MONITOR, REFRESHED THE DISK AND
V15 AUTOMATICALLY LOGGED IN UNDER THE FAILSAFE NUMBERS 1,2
(AS DESCRIBED IN SECTION 1.3 ABOVE) MOUNT DECTAPE CONTAINING
FAILCO.SAV ON DECTAPE DRIVE 2 AND THE CUSP MAGTAPE ON MAGTAPE
DRIVE 0, TYPE TO THE MONITOR:

ASSIGN MTA2 FAILSA

FOLLOWED BY A CARRIAGE RETURN, THE MONITOR WILL RESPOND
MTA2 ASSIGNED

(BE SURE THAT THE TAPE AT THE LOAD POINT, ON LINE AND
WRITE LOCKED - RING REMOVED)
ASSIGN DSKB DSK
WHERE DSKB IS YOUR LARGEST FILE STRUCTURE.
(FAILCO WRITES ALL FILES ON DEVICE DSK)

THEN TYPE TO THE MONITOR:
V15 RUN MTA2 FAILCO

FOLLOWED BY A CARRIAGE RETURN, FAILCO WILL TYPE:

FOR HELP, TYPE: /H
*

TYPING /H WILL CAUSE FAILSAFE TO PRINT A LIST OF SWITCH
OPTIONS AND OPERATING INSTRUCTIONS. YOU MAY WISH TO DO
THIS AT SOME TIME BUT FOR NOW IT IS ONLY NECESSARY TO
TYPE THREE COMMANDS

/8

FOLLOWED BY A CARRIAGE RETURN, THIS WILL INFORM FAIL-
SAFE THAT THE MAGTAPE IS RECORDED AT 800 BITS PER INCH.
FAILSAFE WILL RESPOND WITH AN ASTERISK.
V15 TYPE:

V15 /G1 ,7

V15 FOLLOWED BY CARRIAGE RETURN TO TELL FAILCO YOU WISH
V15 TO OPERATE ON THE 10,7 UFDS ONLY

```
V16          IF YOU HAVE A LARGE ENOUGH DISK (1 RP02 OR MORE)
V16          IT WILL BE FASTER TO COPY ALL OF THE 1?,7 AREA FROM
V16          MAGTAPE TO DISK (SOURCE AND SAVE FILES).  HOWEVER IF
V16          YOU HAVE LIMITED SPACE, YOU WILL HAVE TO MAKE 6 PASSES
V16          OVER THE FAILSAFE TAPE IN ORDER TO SELECTIVELY COPY
V16          JUST THE *,REL,*,HGH,*,LOW,*,SAV,*,SHR,QPIP FILES.

V16          IF YOU HAVE A LARGE DISK, TYPE: (COMPUTER OUTPUT IS
V16          UNDERLINED)

V16          **,*
V16          -
V16          *
V16          -

V16          IF YOU HAVE A SMALL DISK, TYPE:

V16          **,REL,*HGH,*,LOW,*,SAV,*,SHR,QPIP
V16          -
V16          *
V16          -
V16

             AFTER FAILCD HAS FINISHED, TYPE:
V16          <CONTROL>C
V16          .RUN DSK PIP[10,7]
V16          -
V16          *DSKB:[1,4]/X/B*DSK:*,REL,*,HGH,*,LOW,*,SAV,*,SHR,QPIP
V16          -
V16          *
V16          -
V16          YOU MAY WISH TO PUT THE FORTRAN LIBRARY ON A FASTER FILE
V16          STRUCTURE THAN DSKB,  TO DO THIS, TYPE:

V16          *DSKA:[1,4]/X/B*DSK:LIB40,REL,DDT,REL

V16          NOTE:  IF DSKB IS LARGE ENOUGH AND DSKA APPEARS BEFORE
V16          DSKB IN THE SYS SEARCH LIST, YOU MAY KEEP DUPLICATE COPIES
V16          OF THE LIBRARY ON DSKB (IN CASE DSKA SHOULD GO DOWN),
```

V16 1.5.3 COPY MONITOR SUPPORT CUSPS FROM MAGTAPE TO DISK

V16 NEXT DISMOUNT THE CUSP MAGTAPE AND MOUNT THE MONITOR
V16 MAGTAPE ON THE SAME DRIVE, (SEE SECTION 1.5.2 FOR MORE
V16 INFORMATION).

V16 THEN TYPE TO THE MONITOR:
V16 .RUN DTA2 FAILCU
V16 -
V16 FOR HELP, TYPE: /H
V16 ----------
V16 */H
V16 -
V16 */C1:,7
V16 -

V16 IF YOU HAVE A LARGE DISK, TYPE:

V16 **,*
V16 -

V16 IF YOU HAVE A SMALL DISK, TYPE:
**,SYS,*,SAV,*,SHR,*,HGH
V16 -
V16 *
V16 -

V16 AFTER FAILCU HAS FINISHED, TYPE:
V16 <CONTROL>C
V16 .R PIP
V16 -
V16 *DSKB:[1,4]/X/B*DSK:*,SYS,*,SAV,*,SHR,*,HGH
V16 -
V16 *
V16 -

V16 IF YOU ALREADY HAVE AN ACCT,SYS AND AUXACC,SYS ON [1,4],
V16 OMIT *,SYS IN ABOVE COMMAND STRING. YOU NOW
V16 HAVE RESTORED THE CUSP AREA EXCEPT FOR YOUR OWN ACCT,SYS,
V16 AND AUXACC,SYS ACCOUNTING FILES.

V16 NOW RESTORE YOUR ACCOUNTING FILES FROM DECTAPE OR DISK
V16 WHEREVER YOU HAVE WRITTEN THEM. IF YOU ARE A NEW
V16 CUSTOMER, YOU WILL NEED TO CREATE THEM. TO DO THIS SEE
V16 SECTIONS 5.6.2 IN LEVELD,MEM, SECTION 13.15 IN DSK016,MEM.

V16 1.5.4 RESTORE YOUR ACCOUNTING FILES

ALONG WITH THE CUSPS AN ACCOUNTING FILE (ACCT.SYS) WAS LOADED WITH THE FOLLOWING ENTRIES:

PROJECT NO.	PROG. NO.	PASSWORD
1	2	FAILS
V14 1	4	CUSP
10	7	DIST
7	7	OPER
6	6	MAINT
100	100	DEMO1

YOU WILL PROBABLY WANT TO RUN THE CUSP REACT TO ADD YOUR OWN NUMBERS AND CHANGE THE PASSWORD FOR FAILSAFE (1,2) AND CUSP (1,4). YOU SHOULD ASSIGN PROJECT NUMBERS AND PROGRAMMER NUMBERS STARTING WITH 11 TO YOUR USERS. DIGITAL HAS RESERVED 1-10 FOR SPECIAL PURPOSES. YOU WILL ALSO NEED TO MAKE ADDITIONS TO THE DISK QUOTA ADMINISTRATIVE FILE AUXACC.SYS. THIS IS ALSO DONE WITH REACT. SEE SOFTWARE NOTEBOOK, LEVELD,MEM SECTION 5,6,2, DSK016,MEM SECTION 13,15. THE AUXACC.SYS FILE HAS QUOTAS FOR [1,*],[6,*],[7,*],[10,*],[100,*] OF 0 RSRVD, 100000 FCFS, 100000 LOGGED OUT ON DSKA,DSKB, AND DSKC.

THE SOURCE FILES FOR ALL THE SYSTEM SOFTWARE HAVE BEEN SAVED WITH FAILSAFE ON MAGNETIC TAPES IN THE FOLLOWING MANNER:

USER AREA	CONTENTS
ON MONITOR MAGTAPE:	
10,7	SOURCES, MONITOR AND MONITOR SUPPORT CUSPS DOCUMENTATION, SAV FILES
ON CUSP MAGTAPE:	
10,7	CUSP SOURCES AND RELATED .OPR FILES DOCUMENTATION, SAV FILES

THESE FILES MAY BE ACCESSED BY LOGGING IN UNDER THE RELATED NUMBER AND RESTORING THE DESIRED FILES WITH FAILCD. FILES MAY BE RESTORED EITHER COMPLETELY FOR A USER AREA OR INDIVIDUALLY. AFTER DOING THIS, IT IS RECOMMENDED YOU COPY THE FILES FROM DISK TO YOUR OWN DECTAPES FOR SAFE KEEPING WITH PIP. FOR THE MONITOR SOURCES, BE SURE TO
V14 COPY THEM IN THE PROPER ORDER TO MAKE 17 MONITOR SOURCE DECTAPES. SEE TABLE.TXT FOR A LISTING OF FILE NAMES FOR EACH DECTAPE. THE MONITOR ASSEMBLY INSTRUCTIONS ASSUME THAT THE MONITOR SOURCES ARE ON DECTAPES AND DESCRIBE HOW TO COPY THEM ONTO THE DISK, IF YOU WISH TO SPEED UP THE ASSEMBLY PROCESS.

(1.5.2.2 CONT'D)

IF YOU HAVE A SINGLE BURROUGHS DISK (RD-10) IT WILL NOT HOLD THE CONTENTS OF THE FAILSAFE TAPE. SINCE IT IS NEVER NECESSARY TO RESTORE ALL THE SOURCE FILES AT ONE TIME THIS SHOULD NOT BE AN INCONVENIENCE.

EXAMPLE:
TO RESTORE ALL LIBRARY SOURCES

```
LOGIN                          USER TYPES LOGIN
JOB 3  45,45 TS MONITOR        SYSTEM ASSIGNS JOB NO,
#1 ,7                          USER GIVES PROJ,PROG NOS
PASSWORD:                      PASSWORD- DOES NOT PRINT
1015 4-JUN-69 TTY3             JOB IS LOGGED IN
*C
.AS MTA FAILSA                 LOGICAL NAME
MTA: ASSIGNED                  FAILSA REQUIRED
.R FAILSA
FOR HELP, TYPE: /H             MESSAGE FROM FAILSAFE
**,*                           FIRST * TYPED BY FAILSAFE,
                               USER TYPES *,* TO RESTORE ALL
                               LIBRARY SOURCES
          OR
FOR HELP, TYPE: /H
*AXIS,F4,ALPHI.MAC,ALPHO.MAC
                               USER TYPES NAMES OF SELECTED
                               FILES.  ONLY THESE ARE RESTORED
                               TO THE DISK IN USER AREA 10,7.
```

2. HOW TO MAKE A MONITOR FOR YOUR CONFIGURATION FROM LIBRARY FILE

2.0.1 REQUIRED COMPONENTS - IF MONITOR SOURCES ARE ON DECTAPE:

2.0.1.1 10/40 MONITOR MAKER TAPE [TO MAKE 10/40 OR 10/50 SYSTEM]

	SPRON,16K	MINIMAL 16K SPECIAL 10/30 MONITOR
	SPRON,32K	MINIMAL 32K SPECIAL 10/30 MONITOR
V15	SPRON,48K	MINIMAL 48K SPECIAL 10/30 MONITOR
V15	12..,16K	MINIMAL STANDARD 10/30 MONITOR
V15		(NEEDED TO MAKE MONGEN,SVE)
	PIP,SVE	STANDARD 10/30 PIP
	MACRO,SVE	STANDARD 10/30 MACRO
	LOADER,SVE	STANDARD 10/30 LOADER
	MONGEN,SVE	CONFIGURATION DEFINITION PROG. TO RUN UNDER 10/30
V14	5N##,REL	MONITOR LIBRARY BINARY TO PRODUCE 10/40 TIME SHARING SYSTEM [WHERE ## IS THE LOAD NUMBER OF THE MULTIPROGRAMMING NON-DISK MONITOR]

2.0.1.2 ASSEMBLY TAPE (TAPE #5)

S,MAC	SYMBOL DEFINITION FILE ASSEMBLED WITH MOST OTHER FILES

2.0.1.3 MONITOR SOURCE FILES (TAPE #4)

COMMON,MAC	COMMON DATA STORAGE FOR MONITOR

2.0.1.4 10/50 MONITOR REL FILES [NOT NEEDED FOR 10/40 SYSTEM] (TAPE #7)

V14	5S##,REL	MONITOR LIBRARY BINARY TO PRODUCE 10/50 TIME SHARING SYSTEM [WHERE ## IS THE LOAD NUMBER OF THE SWAPPING MONITOR]

2.0.1.5 LEVEL D MONITOR FILE SYSTEM SOURCES (TAPE #14) [NEEDED ONLY IF DISK SYSTEM]

COMMOD,MAC	COMMON DATA BASE FOR DISK SYSTEMS
DATDMP,MAC	(OPTIONAL) CORE BLOCK DUMPER FOR DEBUGGING

2.0.1.6 MONITOR SUPPORT CUSP,SAV FILES

2.0.1.7 ONE BLANK DECTAPE

2.0.1.8 3 DECTAPE DRIVES (2 CAN BE USED, ALTHOUGH THE INSTRUCTIONS ARE WRITTEN FOR 3 UNITS)

2.0.1.9 OPTIONAL - A LINE PRINTER

2.0.1.10 OPTIONAL - A RUNNING TIME SHARING SYSTEM

SKIP TO SECTION 2.1 IF SOURCES ARE ON DECTAPE

```
V16     2.2.2 REQUIRED COMPONENTS - IF MONITOR SOURCES ARE ON MAGTAPE

V16           2. .2.1 A RUNNING LEVEL C (3 OR 4 SERIES) OR LEVEL
V16                   D (5 SERIES) MONITOR

V16           2. .2.2 A MONITOR SOURCES MAGTAPE (FAILCD FORMAT
V16                   UFD 1Ø,7)

V16           2. .2.3 A DECTAPE WITH FAILCD.SAV ON IT IF THE RUNNING
V16                   MONITOR IS A LEVEL D MONITOR.  NEW CUSTOMERS
V17                   WILL FIND FAILCD.SAV ON THE "YOUR MONITOR TAPE"
V17                   (TAPE #1).    OTHERWISE THE
V16                   LEVEL C FAILSA.SAV ALREADY ON THE CUSP WILL
V16                   SUFFICE. THEN DO SECTION 2.Ø.3 FOLLOWING:

V16     2. .3 COPY MONITOR SOURCES FROM MAGTAPE TO LEVEL C OR D DISK

V16           2. .3.1 IF YOU ARE RUNNING UNDER A LEVEL C DISK MONITOR
V16                   YOU MUST ADD A PASSWORD FOR PROJECT, PROGRAMMER
V16                   NUMBER [1Ø,7].  BE SURE TO USE LEVEL C (NOT DIS-
V16                   TRIBUTED LEVEL D REACT) SINCE YOU ARE RUNNING
V16                   UNDER A LEVEL C MONITOR.  ANY PASSWORD WILL DO.

V16                   IF YOU ARE RUNNING UNDER A LEVEL D DISK MONITOR,
V16                   THE ACCT.SYS AND AUXACC.SYS ADMINISTRATIVE FILES
V16                   ALREADY HAVE PASSWORD AND DISK QUOTAS RESPECTIVELY
V16                   FOR [1Ø,7].  THE PASSWORD IS DIST

V16           2. .3.2 LOGIN UNDER 1Ø,7

V16           2. .3.3 ASSIGN A MAGTAPE, BY TYPING:

V16                   .ASSIGN MTA FAILSA
V16                   -
V16                   MTAØ ASSIGNED
V16                   -------------

V16           2. .3.4 MOUNT THE MONITOR FAILSAFE MAGTAPE ON THE UNIT
V16                   TYPED OUT BY THE MONITOR (EG MTAØ).

V16           2. .3.5 IF RUNNING UNDER LEVEL C, TYPE:

V16                   .R FAILSA
V16                   -

V16                   (WHERE FAILSA IS LEVEL C FAILSA, NOT DISTRIBUTED LEVEL D FAILSA)
V16                   ELSE IF RUNNING UNDER LEVEL D, TYPE:

V16                   .ASSIGN DTA
V16                   -
V16                   DTA3 ASSIGNED
V16                   -------------

V16                   MOUNT DECTAPE CONTAINING FAILCD.SAV ON IT
V16                   (TAPE #1) FOLLOWED BY:

V16                   .RUN DTA3 FAILCD
```

```
V16          2.0.3.6 TYPE
V16                  ASSIGN DSKB DSK
V16                  WHERE DSKB IS THE FILE STRUCTURE ON WHICH YOU
V16                  WISH TO WRITE THE MONITOR FILES.

V16          2.0.3.7 IF YOU HAVE ONLY 32K OR HAVE 48K OF CORE BUT
V16                  ARE RUNNING LEVEL D, THERE WILL NOT BE SUF-
V16                  FICIENT USER CORE TO LOAD A 5 SERIES MONITOR
V16                  UNDER TIME SHARING.  THEREFORE YOU WILL HAVE
V16                  TO LOAD USING SPMON, THE SPECIAL SINGLE USER
V16                  MONITOR.  THEREFORE TYPE:

V16                  *SPMON,32K,SPMON,48K,*,SVE
V16                  -

V16                  WHICH WILL READ SPMON AND ITS CUSPS ONTO THE DISK.

V16                  WHEN FAILSA (OR FAILCD) TYPES *, RESTORE THE
V16                  FOLLOWING FILES:

V16                  *MONGEN.SAV,S.MAC,COMMON.MAC,5S01.REL,COMMOD.MAC,DATDMP.MAC
V16                  -

        2.0.4 DEFINE YOUR CONFIGURATION USING MONGEN UNDER TIME SHARING

                TYPE:

V16                  .RUN DSK MONGEN
V16                  *

V16                  MONGEN WILL ASK QUESTIONS ABOUT YOUR CONFIGURATION
V17                  SEE MONITR.OPR SECTION 2.1 (STARTING WITH
V16                  2.1.12). (PUT CONFIG.MAC ON DSK BY TYPING JUST
V16                  CARRIAGE RETURN.)

        2.0.5 ASSEMBLE COMMON [AND COMMOD] UNDER TIME SHARING

V16          2.0.5.1 ASSEMBLE COMMON BY TYPING:

V16                  .COM S+CONFIG+COMMON/C
V16                  *

V16          2.0.5.2 ASSEMBLE COMMOD BY TYPING:
                     (IF YOUR ARE MAKING A DISK SYSTEM)

V16                  .COM S+CONFIG+COMMOD/C
V16                  *
V16                       -OR-

V16                  .COM S+CONFIG+DATDMP+COMMOD/C
V16                  *

V16                  IF YOU WANT TO BE ABLE TO LOOK AT FILSER CORE
V16                  BLOCKS IN EXEC MODE FOR DEBUGGING; DATDMP IS
V16                  700 LOCS.
```

```
V16          2. .5.3 OBTAIN CREF LISTINGS BY TYPING:

V16                  .CREF
V16                  -

V16          2. .5.4 IF YOU HAVE SUFFICIENT USER CORE TO LOAD A
V16                  MONITOR UNDER TIME SHARING, GO TO SECTIONS
V16                  2.3 AND 2.4.  OTHERWISE YOU MUST COPY SOME
V16                  FILES TO DECTAPE BEFORE USING SPMON, AS FOLLOWS:

        2.7.6 COPY FROM DISK TO DECTAPE TO BE USED BY SPMON
                (IF NOT ENOUGH USER CORE TO LOAD MONITOR UNDER TIME SHARING)

V16                  .ASSIGN DTA3
V16                  -
V16                  DTA3 ASSIGNED
V16                  -------------

V16                  WHERE DTA3 CAN BE ANY DECTAPE.  REMOVE DECTAPE
V16                  CONTAINING FAILCD.SAV IF YOU WISH.

V16                  MOUNT A SCRATCH TAPE AND ZERO THE DIRECTORY.

V16                  .R PIP
V16                  -
V16                  *DTA3:*/Z
V16                  *DTA3:/X/B*DSK:SPMON.*.*.SVE
V16                  -

V16                  LABEL THIS TAPE "10/40 MONITOR MAKER" BECAUSE
V16                  IT HAS MOST OF THE FILES WHICH ARE ON THAT
V16                  DECTAPE IMAGE.  MOUNT A SECOND SCRATCH TAPE
V16                  ON DTA3 AND TYPE:

V16                  .ASSIGN DTA3
V16                  -
V16                  .R PIP
V16                  -
V16                  *DTA3:*/Z
V16                  -
V16                  *DTA3:/X/B*DSK:COMMON.REL.COMMOD.REL.5S01.REL
V16                  -

V16          2. .4.6 NOW, USING TENDMP,  LOAD SPMON FROM "10/40 MONITOR
V16                  MAKER" DECTAPE YOU HAVE JUST WRITTEN.  USE
V16                  SPMON.32K IF YOU HAVE 32K, USE SPMON.48K IF
V16                  YOU HAVE 48K OR MORE PHYSICAL CORE.  SEE SECT 1.1
V17                  FOR TENDMP INSTRUCTIONS.

V16          2. .4.7 LOAD YOUR MONITOR WHILE RUNNING UNDER SPMON.
V16                  USING LOADER.SVE FROM "10/40 MONITOR MAKER"
V16                  DECTAPE.  SEE SECTION 2.3 FOR LOADER INSTRUCTIONS.

V16          2. .4.8 SAVE YOUR MONITOR ON DECTAPE: SEE SECTION 2.4.
V16                  TO LOAD YOUR MONITOR FROM DECTAPE WITH TENDMP
V16                  DO SECTION 1.
```

2.1 DEFINE CONFIGURATION FILE USING MONGEN

2.1.1 MONGEN IS A PROGRAM WHICH WILL ASK YOU QUESTIONS
ABOUT YOUR HARDWARE AND SOFTWARE CONFIGURATIONS,
YOUR ANSWERS WILL BE WRITTEN AS A MACRO SOURCE FILE NAMED
CONFIG.MAC. THEN YOU WILL ASSEMBLE IT WITH S.MAC AND
V14 COMMON.MAC TO PRODUCE COMMON.REL. [FOR 10/50 AND 10/40D
(LEVEL D DISK) YOU WILL ALSO COMBINE S.MAC, CONFIG.MAC,
V14 AND COMMOD.MAC TO FORM COMMOD.REL.] FINALLY YOU
WILL LOAD COMMON.REL [AND COMMOD.REL] AND PERFORM
V14 A SEARCH ON THE APPROPRIATE MONITOR LIBRARY FILE
V14 5N##.REL OR 5S##.REL. MONGEN MAY BE RUN UNDER A REGULAR
V14 10/40 OR 10/50 TIME SHARING SYSTEM (IF YOU HAVE ONE ALREADY).
HOWEVER, THE FOLLOWING INSTRUCTIONS HAVE BEEN WRITTEN
FOR THE USER STARTING FROM SCRATCH WITH JUST THE MONITOR
DECTAPE SOURCES AND SPMON - THE SPECIAL 10/30
SINGLE USER MONITOR. THE OPERATING INSTRUCTIONS
FOR RUNNING MONGEN UNDER TIME SHARING ARE THE SAME AS
UNDER SPMON - THE SPECIAL 10/30 MONITOR.
NOTE: THE 10/30 MONITORS LOOK FOR AND WRITE A FILE EXTENSION
OF .SVE WHILE 10/40 AND 10/50 MONITORS LOOK FOR AND WRITE .SAV.
THEY CANNOT BE INTERCHANGED BECAUSE THE JOB DATA AREAS
V15 ARE INCOMPATIBLE. ALTHOUGH SPMON IS A 10/30 MONITOR,
V15 IT IS SPECIAL IN THAT IT WRITES SAVE FILES WHICH ARE
V15 COMPATIBLE WITH 10/40 AND 10/50 MONITORS INSTEAD OF
V15 10/30 MONITORS. IT IS USED TO CREATE .SVE MONITOR
V15 FILE WHICH CAN BE LOADED WITH A 10/40 OR 10/50 GET
V15 COMMAND FOR DDT PATCHING.

2.1.2 LOAD TENDMP INTO CORE ACCORDING TO THE DETAILED
V15 INSTRUCTIONS GIVEN IN SECTION 1.1. IF YOU HAVE 16K
V15 OF CORE, USE SPMON.16K; 32K OF CORE, USE SPMON.32K;
V15 48K OR MORE, USE SPMON.48K.

2.1.3 MOUNT "10/40 MONITOR MAKER" (EVEN IF MAKING A 10/50 SYSTEM)
ON DECTAPE UNIT 0. SET WRITE SELECT SWITCH ON TOP
LEFT ON DECTAPE UNIT TO WRITE LOCK (BOTTOM PUSHED IN).
THEN TYPE TO TENDMP
0$ ($=ALT-MODE)
TENDMP WILL THEN READ THE DIRECTORY FROM DECTAPE 0.

2.1.4 TYPE
SPMON 16K [OR SPMON 32K OR SPMON 48K]
FOLLOWED BY CARRIAGE RETURN ON THE CONSOLE TELETYPE.
TENDMP WILL LOAD AND ACTIVATE A SMALL 10/30 MONITOR.
SPMON. TYPE THE DATE AND TIME AS REQUESTED
BY SPMON AND SPMON WILL TYPE A PERIOD ON THE CONSOLE
TELETYPE TO INDICATE THAT THE USER CAN TYPE MONITOR

COMMANDS. THE 10/30 MONITOR COMMANDS ARE ALMOST
IDENTICAL TO THE 10/40 AND 10/50 MONITOR COMMANDS.

2.1.5 TYPE
ASSIGN DTA2
FOLLOWED BY CARRIAGE RETURN TO SPMON TO ASSIGN DECTAPE UNIT 2 (DIAL SET TO 8). NOTE: SPMON'S CONFIGURATION HAS ONLY DECTAPE UNITS DTAØ,DTA1, AND DTA2.

TYPE
ASSIGN DTA1
FOLLOWED BY CARRIAGE RETURN TO SPMON TO ASSIGN DECTAPE UNIT 1.

TYPE
ASSIGN DTA2
FOLLOWED BY CARRIAGE RETURN TO ASSIGN DECTAPE UNIT 2.
V15 NOTE: IT IS IMPERATIVE THAT YOU ASSIGN ALL THREE
V15 DECTAPES BEFORE YOU USE ANY OF THEM. THIS INFORMS
V15 SPMON THAT YOU INTEND TO USE ALL THREE AND PREVENTS
V15 SPMON FROM TRYING TO USE THE DIRECTORY BUFFER SPACE.

2.1.7 MOUNT A BLANK TAPE ON DECTAPE UNIT 1. THIS TAPE WILL RECEIVE FILE CONFIG.MAC; SO SET WRITE SELECT SWITCH ON TOP LEFT OF DECTAPE UNIT TO WRITE (TOP PUSHED IN, WRITE ENABLED LIGHT SHOULD BE ON).

2.1.8 TYPE
R PIP
FOLLOWED BY CARRIAGE RETURN TO SPMON, TO LOAD AND START PIP.SVE, THE PERIPHERAL INTERCHANGE PROGRAM. WHEN DTAØ STOPS SPINNING, PIP WILL TYPE AN * AND WAIT FOR INPUT. [IF YOU ARE RUNNING UNDER A 10/40 OR 10/50 MONITOR, ALSO TYPE R PIP].

2.1.9 CLEAR DTA1 AND COPY CONFIGURATION DEPENDENT SOURCE FILES FROM OTHER TAPES. IF THE SOURCES ARE ALREADY
V16 ON THE DISK COPY THEM DIRECTLY FROM THE DISK TO DTA1,
V16 RATHER THAN FROM DTA2 TO DTA1 AS DESCRIBED BELOW, FOR USE
V16 WITH SPMON (SINCE SPMON CANNOT ACCESS THE DISK).

2.1.9.1 TYPE
DTA1:←/Z
FOLLOWED BY CARRIAGE RETURN TO PIP TO CLEAR (ZERO) THE DIRECTORY ON UNIT 1 IN CASE IT HAS SOME FILES ON IT. DTA1 WILL MOVE AND PIP WILL RESPOND WITH * WHEN IT HAS FINISHED.

2.1.9.2 TYPE
<CONTROL>C
BY HOLDING DOWN THE CONTROL KEY (LEFT HAND CORNER OF KEYBOARD) AND PUSHING C, SPMON WILL RESPOND BY TYPING ↑C FOLLOWED BY TWO CARRIAGE RETURN LINE FEEDS AND A DOT.

```
V15   2.1.9.3 COPY S.MAC FROM "ASSEMBLY TAPE" (TAPE #3)
V15           TYPE:
V15           ASSIGN DTA2
V15           TO INFORM SPMON THAT A NEW TAPE IS ABOUT TO BE MOUNTED
V15           MOUNT "ASSEMBLY TAPE" (TAPE #3) ON DTA2
V15           TYPE:
V15           START
V15           TO START PIP OVER AGAIN,  PIP WILL RESPOND WITH *,
V15           TYPE:
V15           DTA1:/X/B-DTA2:S.MAC
V15           PIP WILL RESPOND WITH AN * WHEN IT HAS FINISHED
V15           THEN TYPE: (DO NOT TYPE AHEAD WITH SPMON, IE WAIT UNTIL *
V15           IS TYPED) <CONTROL>C

V15   2.1.9.4 COPY COMMON.MAC FROM MONITOR SOURCE FILES TAPE (TAPE #4)
V15           TYPE:
V15           ASSIGN DTA2
V15           TO INFORM SPMON THAT A NEW TAPE IS ABOUT TO BE MOUNTED,
V15           MOUNT "MONITOR SOURCE FILES TAPE" (TAPE #4) ON DTA2
V15           TYPE:
V15           START
V15           TO START PIP OVER AGAIN,  PIP WILL RESPOND WITH AN *,
V15           TYPE:
V15           DTA1:/X/B-DTA2:COMMON.MAC
V15           PIP WILL RESPOND WITH AN * WHEN IT HAS FINISHED
V15           THEN TYPE:
V15           <CONTROL>C

V15   2.1.9.5 COPY COMMOD.MAC AND DATDMP.MAC FROM "LEVEL D FILE SYSTEM
V15           SOURCES" (TAPE #14)--ONLY FOR DISK SYSTEMS (10/40D, 10/50)
V15           TYPE:
V15           ASSIGN DTA2
V15           TO INFORM SPMON THAT A NEW TAPE IS ABOUT TO BE MOUNTED
V15           MOUNT "LEVEL D FILE SYSTEM SOURCES" (TAPE #14) ON DTA2
V15           TYPE:
V15           START
V15           TO START PIP OVER AGAIN,  PIP WILL RESPOND WITH A *,
V15           TYPE:
V15           DTA1:/X/B-DTA2:COMMOD.MAC,DATDMP.MAC
V15           PIP WILL RESPOND WITH AN * WHEN IT HAS FINISHED
V15           THEN TYPE:
V15           <CONTROL>C

V15    2.1.11 MOUNT AND START MONGEN
V15           TYPE:
V15           ASSIGN DTA2
V15           TO INFORM THAT A NEW TAPE IS ABOUT TO BE MOUNTED
V15           MOUNT "ASSEMBLY TAPE" (TAPE #3) ON DTA2
V15           TYPE:
V15           RUN DTA2 MONGEN
              FOLLOWED BY CARRIAGE RETURN TO SPMON TO LOAD
              AND START THE CONFIGURATION DEFINITION DIALOG
              PROGRAM, MONGEN.
```

2.1.12 MONGEN WILL RESPOND WITH
TYPE "DEVICE:NAME<CR>" FOR WHERE TO PUT RESULTS
OF THIS DIALOG, OR ASSUMES "DSK:CONFIG.MAC"

SINCE SPMON DOES NOT HAVE A DSK (LARGER 10/30 MONITORS
HAVE A DISK FILE STRUCTURE WHICH IS INCOMPATIBLE WITH
1 /40 AND 10/50 SYSTEMS), TYPE
DTA1:CONFIG
FOLLOWED BY CARRIAGE RETURN TO MONGEN (AFTER YOU
HAVE MADE YOUR MONITOR, YOU WILL BE ABLE TO
RUN MONGEN.SAV UNDER IT INSTEAD OF MONGEN.SVE
UNDER SPMON)

2.1.13 MONGEN WILL TYPE
ANSWER THE FOLLOWING QUESTIONS WITH Y OR N
OR A DECIMAL NUMBER
SHORT DIALOG? [N=LONGER QUESTIONS]

IT IS RECOMMENDED THAT YOU TYPE N FOLLOWED BY
CARRIAGE RETURN UNTIL YOU ARE MORE FAMILIAR
WITH MONGEN, BECAUSE MONGEN WILL EXPLAIN THE
QUESTIONS MORE FULLY BY TYPING EXTRA COMMENTS
INSIDE SQUARE BRACKETS.

2.1.14 MONGEN WILL WRITE EACH QUESTION,
ANSWER AND SYMBOL DEFINITION IN FILE CONFIG.MAC,
BECAUSE MOST OF THE QUESTION ARE STRAIGHT-
FORWARD, THEY HAVE NOT BEEN INCLUDED HERE, SEE THE
LISTING OF CONFIG.MAC IN MONITOR LISTING OF COMMON
FOR A SAMPLE DIALOG, ONLY A FEW ARE EXPLAINED MORE
FULLY BELOW.

2.1.14.1 TYPE "SYMBOL,VALUE" (VALUE IN DECIMAL)
FOR ANY SYMBOLS TO BE DEFINED, TYPE EXTRA
V14 CARRIAGE RETURN WHEN THROUGH. THEN IT WILL ASK THE
V14 SAME QUESTION FOR OCTAL VALUES.

V15 IF YOU DISCOVER THAT YOU ARE RUNNING MONGEN OFTEN,
V15 IT IS RECOMMENDED THAT YOU EDIT THE BEGINNING OF
V15 COMMON.MAC [AND/OR COMMOD.MAC] SO THAT YOU WILL NOT
V15 ACCIDENTALLY BUILD A MONITOR WITH AN INCORRECT SYMBOL
V15 VALUE. HOWEVER FOR THE FIRST FEW TIMES, CHANGING
V15 THE SYMBOL VALUES WITH MONGEN WILL SUFFICE.

THE FOLLOWING SYMBOLS CAN BE CHANGED TO BE DIFFERENT
FROM THE STANDARD, THEY ARE SHOWN WITH THEIR STANDARD
(DECIMAL) VALUES. IF YOU ARE SATISFIED WITH THE STANDARD,
JUST TYPE CARRIAGE RETURN LINE FEED. IF YOU DO DECIDE
TO CHANGE ONE OR MORE OF THE FOLLOWING SYMBOLS, TYPE
THE SYMBOL FOLLOWED BY ITS DESIRED VALUE SEPARATED BY
A COMMA. PUT EACH SYMBOL DEFINITION ON A SEPARATE
LINE. TYPE AN EXTRA CARRIAGE-RETURN LINE-FEED WHEN DONE.

2,1,14,1,3 JIFSEC,60
POWER FREQUENCY IN HERTZ (CYCLES PER SECOND),
SOME INSTALLATIONS SHOULD CHANGE THIS TO
50 SO THAT ACCOUNTING AND TIME OF DAY WILL BE
CORRECT,

2,1,14,1,4 DTTRY,4
NO, OF TIMES TO TRY ON DECTAPE ERRORS

2,1,14,1,5 MTSIZ,128
SIZE OF MAGTAPE RECORDS

2,1,14,1,6 LPTSIZ,26
SIZE OF LPT BUFFER+2,
THIS SHOULD BE MADE 29 IF YOU HAVE 132 CHAR
LINE PRINTER AND WANT TO PRINT ON ALL COLUMNS
WITH FORTRAN,

V14 2,1,14,1,7 NSPMEM,1000
NUMBER OF NANO-SECONDS PER MEMORY CYCLE
V15 (SHOULD BE 1760 FOR MB10 MEMORIES), THIS
IS ONLY USED TO COMPUTE THE AMOUNT OF TIME
SPENT CORE SHUFFLING AS PRINTED BY THE
V15 SYSTAT COMMAND,

V15 2,1,14,1,8 MINCOR,JOBN*54 [LEVEL-C ONLY]
THE MONITOR RESERVES A TABLE OF AT LEAST MINCOR WORDS
(POSSIBLY MORE UP TO THE NEXT 1K BOUNDARY FOR ALLOCATING
DISK DEVICE DATA BLOCKS, MINCOR IS NORMALLY 54*JOBN
WORDS ALLOWING 1.5 OPEN DISK FILES PER JOB, IF THE
VALUE FOR MINCOR IS TOO SMALL IT MAY BE SUPPLIED
DURING THE MONGEN DIALOGUE, THE TOTAL SIZE OF THE
MONITOR IS PRINTED OUT AFTER THIS SPACE IS RESERVED
IN THE LONG ONCE ONLY DIALOG, (SEE 1,3),

2,1,14,1,9 LOGSIZ,2
MINIMUM AMOUNT OF VIRTUAL CORE REQUIRED IN ORDER
TO ALLOW A USER TO BE LOGGED IN, IF THIS AMOUNT
IS NOT AVAILABLE, USER WILL RECEIVE USUAL CORE
UNAVAILABLE MESSAGE INCLUDING AMOUNT OF VIRTUAL
CORE LEFT, THIS VALUE MUST BE AT LEAST AS BIG AS
LOGIN CUSP (CURRENTLY 2K)

```
V14     2.1.14.1.11     STRMAX,14
V14                     THE MAXIMUM NUMBER OF DISK FILE STRUCTURES
V14                     WHICH CAN BE ON LINE AT ONE TIME.  DECREASING
V14                     THIS VALUE SAVES ONE WORD PER VALUE.

V14     2.1.14.1.12     CCWMAX,10
V14                     THE MAXIMUM LENGTH OF A DISK CHANNEL COMMAND
V14                     LIST, ALTERING THIS VALUE AFFECTS ONLY
V14                     THE EFFICIENCY AND SPACE [1 WORD PER CHANNEL].

V14     2.1.14.1.13     SWPMAX,8
V14                     THE MAXIMUM NUMBER OF DISK UNITS WHICH MAY BE
V14                     USED FOR SWAPPING.  [1 WORD PER UNIT].

V14     2.1.14.1.14     SWCLSN,7
V14                     THE HIGHEST CLASS NUMBER FOR SWAPPING.

V14     2.1.14.1.15     DSKTRY,3
V14                     NO. OF TIMES TO TRY ON DISK ERRORS.

V14     2.1.14.1.18     CHVIFP,10
V14                     STANDARD FAIRNESS COUNT FOR POSITIONING.
V15                     [SEE DSK016,MEM SECTION 10.3.3]

V14     2.1.14.1.19     CH0IFP,CHVIFP ... CH7IFP,CHVIFP
V14                     FAIRNESS COUNT FOR POSITIONING ON CHANNEL 0,...,7.

V14     2.1.14.1.20     CHVIFT,10
V14                     STANDARD FAIRNESS COUNT FOR TRANSFERS.

V14     2.1.14.1.21     CH0IFT,CHVIFT ... CH7IFT,CHVIFT
V14                     FAIRNESS COUNT FOR TRANSFERS ON CHANNEL 0,...,7.
V14                     [SEE DSK016,MEM, SECTION 10.3.3]

V14     2.1.14.1.22     PTRLEN,10
V14                     NUMBER OF IN-CORE RETRIEVAL POINTERS PER DISK
V14                     DDB.

V14     2.1.14.1.23     EPL4WD,12
V14                     ONE-FOURTH THE LENGTH OF THE MAXIMUM EXEC PUSH
V14                     DOWN LIST EXCURSION.
```

```
V15     2.1.14.1.24     FIL4WD,10
V15                     NO. OF 4 WORD BLOCKS ALLOCATED PER JOB IN A COMMON
V15                     POOL OF MONITOR FREE CORE.  THE ONCE ONLY CODE
V15                     MULTIPLIES THIS FACTOR TIMES THE NUMBER OF
V15                     JOBS THE SYSTEM IS BUILT FOR TO ASSIGN THIS SPACE.
V15                     THESE BLOCKS ARE USED BY THE LEVEL D DISK
V15                     SERVICE FOR ACTIVE, DORMANT, AND FREE ACC, AKB, NMB, PPB,
V15                     AND UFB BLOCKS.  THIS POOL IS PERMANENTLY
V15                     RESERVED FOR THESE, BLOCKS AND IS NOT USED FOR
V15                     ANY OTHER PURPOSE.  ANOTHER POOL IS USED FOR
V15                     VARIABLE LENGTH CORE BLOCKS SUCH AS DISK
V15                     DEVICE DATA BLOCKS, AND EXTENDED EXEC PUSH DOWN
V15                     LISTS.  ENTERUUD ERROR CODE 16 (.ERNET) IS
V15                     SET IF THIS POOL FILLS UP.  IF THIS
V15                     HAPPENS REGULARLY, INCREASE THE VALUE OF FIL4WD.
V15                     THE MINIMUM NUMBER OF 4 WORD BLOCKS IS 50 FOR THE SYSTEM.

V14     2.1.14.1.25     UNVRSF,500
V14                     THAT RECIPROCAL FACTOR OF THE TOTAL DISK SIZE
V14                     ON EACH UNIT ONTO WHICH USERS COULD NEVER WRITE THEIR DATA.
V14                     THIS SAFETY FACTOR ENSURES THERE IS ALWAYS ROOM
V14                     TO WRITE THE SECOND RIB, ETC.  I.E. ONE FIVE
V14                     HUNDRETH OF THE DISK SPACE IS RESERVED FOR THIS PURPOSE.

V14     2.1.14.1.26     LBNHOM,1
V14                     LBZHOM,10
V14                     STANDARD LOGICAL BLOCK NUMBERS ON EACH UNIT
V14                     CONTAINING THE HOM BLOCK.

V14     2.1.14.1.27     MFDSIZ,8
V14                     NUMBER OF BLOCKS ALLOCATED TO THE MFD IN EACH
V14                     FILE STRUCTURE BY THE REFRESHER.  THE MFD CAN GROW
V14                     LONGER THAN THIS.  HOWEVER A SPEED ADVANTAGE
V14                     OCCURS IF THE MFD IS CONSECUTIVE BLOCKS.  INCREASE
V14                     THIS VALUE IF YOU DISCOVER YOUR MFD IS USUALLY
V14                     LONGER THAN 8 BLOCKS.

V14     2.1.14.1.30     MBFN,1
V14                     NUMBER OF 128-WORD MONITOR BUFFERS.  (USED
V14                     FOR READING AND WRITING NON-USER DATA).
```

2,1,14,2 TYPE "SYMBOL,VALUE" (VALUE IN OCTAL)

FOR ANY SYMBOL VALUES TO BE CHANGED FROM THE STANDARD
LISTED BELOW, TYPE AN EXTRA CARRIAGE RETURN WHEN THRU.

V15 2,1,14,2,1 STDENS,3 [OCTAL]

STANDARD MAGTAPE DENSITY IF USER PROGRAM DOES NOT
OVERRIDE WITH NON-ZERO VALUE IN INIT, OPEN, OR
SETSTS UUO.

1 BINARY (ODD) PARITY + 200 BPI
2 BINARY (ODD) PARITY + 556 BPI
3 BINARY (ODD) PARITY + 800 BPI

5 BCD (EVEN) PARITY + 200 BPI
6 BCD (EVEN) PARITY + 556 BPI
7 BCD (EVEN) PARITY + 800 BPI

V15 2,1,14,2,2 INDPPN,Ø [OCTAL]
V15 IF INDPPN=Ø, THEN EACH PROGRAMMER NUMBER REFERS
V15 TO THE SAME PERSON IN EVERY PROJECT.
V15 IF INDPPN=777777, THEN PROGRAMMER NUMBERS MAY BE
V15 ASSIGNED INDEPENDENTLY WITHIN EACH PROJECT.
V15 THIS AFFECTS ONLY THE DISK FILE ACCESS PROTECTION
V15 MECHANISM. [SEE DSKØ16.MEM, SECTION 7,2,2]

V15 2,1,14,2,3 PRVFIL,057 [OCTAL]
V15 STANDARD FILE PROTECTION.

V15 2,1,14,2,4 PRVUFD,775 [OCTAL]
V15 STANDARD UFD PRIVILEGE.

V15 2,1,14,2,5 SETWCH,ØØØØØØ [OCTAL]
V15 INITIAL SETTING OF WATCH PARAMETERS FOR EACH JOB IS ALL TURNED
V15 OFF. SEE WATCH COMMAND FOR A DESCRIPTION OF JOB STATISTICS.
V15 THESE JOB STATISTICS ARE TYPED INSIDE BRACKETS
V15 WHEN A USER STARTS TO WAIT FOR THE SYSTEM OR
V15 WHEN THE CONSOLE RETURNS TO COMMAND LEVEL.
V15 BIT 1 = DAY (2ØØØØØ)
V15 BIT 2 = RUN (1ØØØØØ)
V15 BIT 3 = WAIT (4ØØØØ)
V15 BIT 4 = READ (2ØØØØ)
V15 BIT 5 = WRITE (1ØØØØ)

2.1.15 PI CHANNEL ASSIGNMENTS CAN BE CHANGED FROM THEIR STANDARD VALUES WHEN ADDING SPECIALIZED I/O ROUTINES FOR CUSTOMERS' NON-STANDARD DEVICES.

PI ASSIGNMENTS ARE MADE BY GROUPING I/O DEVICES BY RELATIVE SPEED OF INTERRUPTS, AS SHOWN BELOW. IF ANY DEVICE IN A GROUP IS PRESENT, A PI CHANNEL WILL BE ASSIGNED TO THAT GROUP. THUS THE EXACT PI ASSIGNMENT FOR A GIVEN DEVICE VARIES DEPENDING ON THE PRESENCE OF OTHER DEVICES.

IN ADDITION, THE DEVICES ADDED BY A CUSTOMER DURING THE MONGEN DIALOG WILL BE CHAINED ON THE REQUESTED CHANNEL. IF A DEVICE REQUIRES THE EXCLUSIVE USE OF A PI CHANNEL, THAT MAY BE DECLARED TO MONGEN AFTER DECLARING THE NAME OF THE SPECIAL DEVICES.

THE DEVICE GROUPS ARE AS FOLLOWS (THESE MAY BE CHANGED BY RE-ARRANGING THE DEVICES IN INTTAB, IN THE SOURCE OF COMMON):

	GROUP NUMBER (NOT PI CHANNEL)	DEVICE MNEMONIC	NAME
	1	DCB	136 DATA CONTROL FOR 270 DISK
	2	MTA	TM10A DATA CHANNEL
	3	DCT	136 DATA CONTROL FOR 551 OR 516 TAPE CONTROLS
	4	DTA	TD10 DECTAPE DATA CHANNEL
	5	CDR	461 OR CR10 CARD READER
	5	APR	KA10 OR 166 ARITHMETIC PROCESSOR
	6	SCN	DC10,680, OR 630 TELETYPE SCANNER
	6	PTR	PAPER TAPE READER
	6	LPT	LINE PRINTER(S)
	6	DTA	DECTAPE FLAG CHANNEL (DTA OR DTC)
	6	MTA	MAGTAPE FLAG CHANNEL
	6	CTY	CONSOLE TELETYPE
	7	DSK	DISK FLAG CHANNEL (FHD,MDF,DPC, OR DPD)
V11	7	PEN	LIGHT PEN
	7	PTP	PAPER TAPE PUNCH
	7	CDP	CARD PUNCH
	7	PLT	PLOTTER
	8	DIS	DISPLAY DATA CHANNEL
	9	CLK	SCHEDULER, CLOCK ROUTINES, (ALWAYS ASSIGNED TO CHANNEL 7.)

2.2 ASSEMBLE S.CONFIG.COMMON [AND S.CONFIG.COMMOD] USING MACRO

2.2.1 WHEN MONGEN TYPES "EXIT ^C", YOU HAVE CREATED
FILE CONFIG.MAC (ON DTA1 IF USING SPMON), NOW
YOU MUST ASSEMBLE IT WITH S.MAC AND COMMON.MAC
TO PRODUCE COMMON.REL.

2.2.2 TYPE
ASSIGN DTA2
FOLLOWED BY CARRIAGE RETURN TO SPMON TO
ASSIGN DECTAPE UNIT 2.

2.2.3 MOUNT A BLANK TAPE ON DTA2
SET THE UNIT TO WRITE ENABLED
TYPE:
R PIP
CLEAR THE DIRECTORY ON DTA2 BY TYPING:
DTA2:=/E
PIP WILL RESPOND WITH A * WHEN IT IS DONE
THEN TYPE:
(CONTROL) C

2.2.4 TYPE
.R MACRO
FOLLOWED BY CARRIAGE RETURN TO SPMON, TO LOAD
AND START MACRO, THE SYMBOLIC ASSEMBLER.
MACRO WILL TYPE AN * AND WAIT FOR INPUT.
[IF YOU ARE RUNNING UNDER A 10/40 OR
10/50 MONITOR INSTEAD OF SPMON, ALSO TYPE
R MACRO].

2.2.5 TYPE

V15 DTA2:COMMON=DTA1:S,CONFIG,COMMON
FOLLOWED BY CARRIAGE RETURN TO MACRO.
THE ASSEMBLER WILL MAKE TWO PASSES OVER THE
SOURCE FILES AND PRODUCE COMMON.REL ON
DECTAPE UNIT 2. WHEN MACRO IS FINISHED, IT
SHOULD PRINT NO ERRORS DETECTED
FOLLOWED BY *. IF THE ASSEMBLER SHOULD DETECT
SOME ERRORS, CHECK YOUR MONGEN DIALOG,
ESPECIALLY WHERE YOU DEFINED SPECIAL SYMBOLS
OR DEVICE ROUTINES. MAKE SURE THAT YOU
SEPARATED EACH FIELD WITH A COMMA AND
TYPED NO SPACES. (MONGEN TAKES EACH SUCH
LINE AND PASSES IT DIRECTLY TO CONFIG.MAC).
IF YOU CANNOT FIND YOUR ERROR, PERFORM
THE ASSEMBLY OVER AGAIN WITH A LISTING:
DTA2:COMMON,LPT:=DTA1:S,CONFIG,COMMON
AND LOOK TO SEE WHERE ERROR OCCURRED.

V15 [FOR LEVEL D DISK SYSTEMS ONLY. ALSO TYPE:
V15 DTA2:COMMOD=DTA1:S,CONFIG,COMMOD
V15 TO MACRO AFTER IT HAS FINISHED PREVIOUS ASSEMBLY
V15 AND HAS TYPED *
V15 IF YOU ARE INTERESTED IN LOOKING AT THE LEVEL D CORE
V15 BLOCKS WHILE RUNNING IN EXEC MODE, TYPE THE FOLLOWING
V15 COMMAND STRING INSTEAD:
V15 DTA2:COMMOD=DTA1:S,CONFIG,DATDMP,COMMOD
V15 THIS WILL INCLUDE DATDMP WHICH IS ABOUT 700 OCTAL LOCATIONS LONG
V15]

2.2.6 TYPE
<CONTROL>C
V14 TO MACRO AFTER IT HAS FINISHED ASSEMBLY AND HAS
TYPED * TO
RETURN TO SPMON COMMAND LEVEL.

2.3 LOAD COMMON.REL [AND COMMOD.REL] FOLLOWED BY MONITOR LIBRARY FILE USING LOADER. (SEE REFERENCE HANDBOOK FOR EXPLANATION OF LOADER COMMANDS AND SYMBOLS.)

2.3.1 TYPE
R LOADER
FOLLOWED BY CARRIAGE RETURN TO SPMON WHICH WILL LOAD LOADER.SVE WHICH IS A STANDARD LOADER SAVED TO RUN UNDER A 10/30 MONITOR INSTEAD OF A 10/40 OR 10/50 MONITOR. [IF YOU ARE LOADING UNDER TIME SHARING INSTEAD OF SPMON ALSO TYPE R LOADER]. [IF YOU ARE MAKING A 10/50 SYSTEM YOU MUST REPLACE THE "10/40 MONITOR MAKER" (TAPE #2) TAPE WITH THE "12/50 MONITOR REL FILES" TAPE (TAPE #7) AFTER YOU
V14 HAVE GOTTEN THE LOADER OFF OF IT (WITH THE R COMMAND). TO DO THIS TYPE <CONTROL>C FOLLOWED BY:
ASSIGN DTA0
SO THAT SPMON WILL KNOW TO READ THE DIRECTORY AGAIN. THEN REPLACE "10/40 MONITOR MAKER" TAPE (TAPE #2) WITH "10/50 MONITOR REL FILES" TAPE (TAPE #7) AND FINALLY TYPE:
START
TO SPMON TO RETURN TO THE LOADER]

2.3.2 NEXT YOU MUST TYPE THE LOADER COMMAND STRING TO CAUSE IT TO LOAD YOUR TAILOR MADE COMMON.REL FROM DTA2 AND
V14 DO A LIBRARY SEARCH OF 5N##.REL OR 5S##.REL ON DTA0. THE LONG MONGEN DIALOG TYPED OUT THE APPROPRIATE COMMAND STRING FOR YOU TO USE. THE FOLLOWING SECTIONS, 2.3.2 THROUGH 2.3.4 DESCRIBE THE LOADER COMMAND STRINGS, IN CASE YOU DID NOT SPECIFY THE LONG DIALOG. IF YOU DISCOVER YOU MADE A MISTAKE BEFORE TYPING /G, YOU MAY START LOADER OVER AGAIN MERELY BY TYPING <CONTROL>C
START
FOLLOWED BY CARRIAGE RETURN TO MONITOR.

2.3.3 TYPE
/S
FOLLOWED BY CARRIAGE RETURN, TO LOAD LOCAL SYMBOLS FOR DEBUGGING THE MONITOR WITH EXEC DDT OR PATCHING WITH EXEC DDT (STAND-ALONE IN EXEC MODE OR UNDER TIME SHARING IN USER MODE).

2.3.4 TYPE
V14 DTA2:COMMON,DTA0:5N##/L
FOLLOWED BY CARRIAGE RETURN IF YOU ARE BUILDING A 10/40 NON-DISK SYSTEM OR TYPE:
V14 DTA2:COMMON,COMMOD,DTA0:5S##/L
FOLLOWED BY CARRIAGE RETURN IF YOU ARE BUILDING A LEVEL D DISK SYSTEM (10/40 D OR 10/50) SYSTEM. THE /L TELLS THE LOADER TO DO A LIBRARY SEARCH RATHER THAN LOAD EVERY FILE
V14 IN 5N##.REL OR 5S##.REL. COMMON.REL CONTAINS THE PROPER EXTERNAL DECLARATIONS TO LOAD JUST THE ROUTINES NEEDED FOR YOUR CONFIGURATION.

2.3.5 TYPE:

LPT:-/A/M/P/G

V15 NOTE: EXEC DDT (EDDT V24) WILL EXECUTE EITHER IN EXEC
V15 MODE OR USER MODE, SO THAT ONLY ONE COPY OF DDT WILL
V15 SUFFICE FOR PATCHING UNDER TIME SHARING OR STAND-ALONE.

THE LOADER WILL PRINT A STORAGE MAP ON THE LINE
PRINTER AND WILL TYPE
EXIT
WHEN IT IS THROUGH, IF YOU DO NOT HAVE A LINE PRINTER,
TYPE
/P/G
FOLLOWED BY CARRIAGE RETURN

THE /A PRECEDING THE /M CAUSES
THE LOADER MAP TO INCLUDE ALL GLOBAL SYMBOLS
INCLUDING ZERO LENGTH FILES (JOBDAT). THE /P SWITCH
PREVENTS THE LOADER FROM SEARCHING THE LIBRARY IN
CASE THERE ARE UNDEFINED GLOBALS. THE /G IS EQUIVALENT
TO ALTMODE AND SEEMED LESS CONFUSING TO INCLUDE HERE
SINCE THE USER CAN END ALL COMPUTER INPUT WITH CARRIAGE
RETURN.

INSTEAD OF PUTTING THE STORAGE MAP ON THE LINE
PRINTER YOU MAY WISH TO PUT IT ON DTA2 INSTEAD FOR
LATER PRINTING. IF SO TYPE

DTA2:XXXMON-/A/M/P/G

WHERE XXXMON IS THE NAME OF YOUR MONITOR. THE LOADER WILL
WRITE STORAGE MAP AS FILE XXXMON.MAP ON DTA2. IF YOU GET
ANY MULTIPLY DEFINED GLOBALS CHECK TO SEE THAT YOU TYPED
V14 /L IMMEDIATELY FOLLOWING 5N## OR 5S##. IF LOADER
RUNS OUT OF CORE, IT WILL PRINT CORE EXCEEDED FILE
V14 5N## [OR 5S##] CHECK TO MAKE SURE YOU ARE USING THE
LARGEST SPMON FOR YOUR MEMORY SIZE (SPMON.16K,SPMON.32K,
SPMON.48K) IF YOU WERE NOT, RELOAD LARGEST SPMON WHICH
WILL FIT IN YOUR MEMORY AND GO BACK TO STEP 2.3. CHECK
TO MAKE SURE YOU TYPED /L AFTER THE LIBRARY FILE. OTHER-
WISE LOADER WAS ATTEMPTING TO LOAD ALL FILES. TO RECOVER,
TYPE <CONTROL>C START TO MONITOR. IF CORE IS STILL
EXCEEDED TYPE <CONTROL>C START AND REPEAT STEPS 2.3.3
THRU 2.3.5 EXCEPT DO NOT LOAD LOCAL SYMBOLS(/S.).
IF THIS FAILS TRY LOADING WITHOUT EXEC DDT (MUST REPEAT
MONGEN DIALOG AND ANSWER N), AND WITHOUT LOCAL SYMBOLS.

2.4 SAVE MONITOR USING MONITOR SAVE COMMAND

2.4.1 YOU HAVE JUST LOADED YOUR MONITOR INTO CORE USING LOADER. NOW YOU MUST SAVE IT ON DECTAPE SO THAT IT CAN BE LOADED BY TENDMP.

TYPE
V15 SAVE DTA2 XXXMON

FOLLOWED BY A CARRIAGE RETURN TO SPMON, WHERE XXXMON IS THE NAME YOU WOULD LIKE TO GIVE TO YOUR MONITOR. SPMON WILL WRITE FILE XXXMON.SVE ON DTA2. IF YOU ARE RUNNING UNDER TIME SHARING, THE MONITOR WILL WRITE FILE XXXMON.SAV INSTEAD. TENDMP WILL BE ABLE TO LOAD EITHER.

WARNING - NEVER COPY A .SVE OR .SAV FILE ONTO A DECTAPE WITH PIP WHICH YOU MAY WISH TO LOAD WITH TENDMP (EG MONITOR OR SOME MAINTENANCE PROGRAMS). ALWAYS USE THE MONITOR GET AND SAVE COMMANDS INSTEAD. PIP (WITH /B SWITCH) IS FINE FOR ORDINARY USER PROGRAMS AND
CUSPS EVEN THOUGH THEY ALSO HAVE EXTENSIONS OF SAV AND
SVE SINCE YOU WILL NEVER LOAD THEM WITH TENDMP.
V15 THE REASON FOR THIS RESTRICTION IS THAT TENDMP
V15 ASSUMES THAT THE FIRST BLOCK OF A FILE IS THE
V15 LOWEST ONE ON THE TAPE FOR THAT FILE. TENDMP
V15 WOULD HAVE TO BE LARGER IF IT WAS TO BE MADE
V15 COMPATIBLE.

2.4.2 DECTAPE DTA2 SHOULD NOW CONTAIN THE FOLLOWING FILES

COMMON.REL
V14 COMMOD.REL [IF LEVEL-D DISK]
XXXMON.MAP [IF YOU WROTE STORAGE MAP HERE]
XXXMON.SVE [OR XXXMON.SAV IF UNDER TIME SHARING]

V16 2.4.3 AFTER BRINGING UP YOUR LEVEL D MONITOR,
V16 LOGIN UNDER 1,4 AND SAVE YOUR MONITOR ON A SLOW SPEED FILE
V16 STRUCTURE WITH THE NAME SYSTEM.SAV. THEN YOU WILL BE ABLE
V16 TO LOAD IT MERELY BY TYPING CARRIAGE RETURN TO BOOTS.

3. HOW TO ASSEMBLE MONITOR SOURCES, AND CREATE LIBRARY FILE TO PRODUCE A NEW MONITOR.

3.0 REQUIRED COMPONENTS

3.0.1 A RUNNING TIME SHARING SYSTEM EITHER 10/40 OR 10/50,
V14 YOU WILL USE THE FOLLOWING CUSPS: MACRO(VERSION 42
V14 OR LATER), PIP, FUDGE2(OPTIONAL), COMPIL(OPTIONAL),

3.0.2 THE MONITOR SOURCE DECTAPES LABELED:
V14 MONITOR SOURCE 3
V14 MONITOR SOURCE 4
V14 MONITOR SOURCE 5
V14 MONITOR SOURCE 6
V14 MONITOR SOURCE 14 [LEVEL-D SOURCE]
V14 MONITOR SOURCE 15 [LEVEL-D SOURCE]

V16 - OR -

V16 MONITOR SOURCES ON A FAILSAFE MAGTAPE.

3.1 ASSEMBLE MONITOR SOURCES AUTOMATICALLY USING THE MACRO INDIRECT FEATURE

3.1.1 TWO METHODS OF ASSEMBLING THE MONITOR ARE DESCRIBED HERE, ONE IS AUTOMATIC AND ASSEMBLES EVERY ROUTINE OF THE MONITOR WITH A SINGLE COMMAND STRING. IT USES THE INDIRECT FEATURE OF MACRO TO READ A FILE OF COMMANDS. IT WILL EVEN
V14 WORK ON 10/40 SYSTEMS PROVIDED THAT 7 DECTAPES ARE AVAILABLE. THE SECOND METHOD (SECTION 3.2) DESCRIBES HOW TO ASSEMBLE EACH ROUTINE INDIVIDUALLY AND IS INCLUDED FOR COMPLETENESS ONLY. REFER TO IT IF YOU WISH TO MAKE UP A MACRO INDIRECT COMMAND FILE WHICH ASSEMBLES ONLY THE SOURCES FOR YOUR CONFIGURATION

MACRO INDIRECT FILES APPEAR ON MONITOR SOURCE TAPE 3
V14 FOR 10/40 SYSTEMS, AND EACH SOURCE TAPE FOR 10/50 SYSTEMS,
V14 ALL WITH EXTENSION ",CCL", THE XXXRL4.CCL, XXXRL5.CCL,
V14 XXXRL6.CCL FILES PRODUCE JUST BINARY FROM SOURCE TAPES
V14 4,5,6, THE XXXBTH.CCL FILES PRODUCE BOTH BINARY AND CREF
V14 INPUT FROM ALL SOURCE TAPES, S50RLX.CCL ASSEMBLES ALL
V16 SOURCES FROM DSK AND PUTS BINARIES ON DSK. IT IS
V16 THE RECOMMEND WAY IF YOU HAVE ENOUGH DISK SPACE(5000 BLOCKS).
MORE THAN ONE FILE IS REQUIRED BECAUSE OF THE LIMITATION OF 22 FILES ON A DECTAPE AND THE DESIRE TO BE ABLE TO ASSEMBLE BINARIES WITH JUST 4 DECTAPES. THE FOLLOWING CHART SHOWS THE DEVICES NEEDED IN ORDER TO PERFORM THE ASSEMBLY.

	FILE NAME	REL	LST	#DTA	#MTA	INPUT	OUTPUT
V14	S4 RLN,CCL	Y	NO	3	Ø	3,N	RLN
V14	S4 BTH,CCL	Y	Y	7	1	3-6	RL4,RL5,RL6,M
V14	S5 RLN,CCL	Y	NO	3	Ø	3,N	RLN
V14	S5 BTH,CCL	Y	Y	11	1	3-6	RL4,RL5,RL6,M
V14						14-15	RL14,RL15
V16	S5ØRLX,CCL	Y	NO	Ø	Ø	DSK	DSK

THE INDIRECT FILES HAVE BEEN DESIGNED SO THAT A DISK
IS NOT REQUIRED. HOWEVER THE ASSEMBLY PROCESS CAN BE
SPED UP BY COPYING SOME OR ALL OF THE SOURCES ONTO THE
DISK AND PUTTING THE BINARY ONTO THE DISK.

ALL OF THE FILES EXPECT THE FOUR MONITOR SOURCES TO
V14 APPEAR ON LOGICAL DEVICES 3,4,5,6,14 AND 15 RESPECTIVELY.
V14 SEE LISTING OF TABLE.TXT ON MONITOR SOURCE TAPE 3
FOR DIRECTORY LISTING OR CONSULT THE DECTAPE DIRECTORIES
THEMSELVES. THE BINARIES ARE PUT ON LOGICAL DEVICES
V14 RL4, RL5, RL6, RL14 AND RL15. UNFORTUNATELY 2 DEVICES
ARE REQUIRED SINCE THERE WILL BE MORE THAN 22 BINARY FILES.
THE CREF INPUT IS WRITTEN ON LOGICAL DEVICE M WHICH MUST
BE A MAGTAPE UNLESS YOU HAVE A VERY LARGE DISK.

IF YOU HAVE A DISK THE GREATEST SAVINGS CAN BE ACCOM-
V14 PLISHED BY COPYING ALL OF MONITOR SOURCE TAPE 3
V14 ONTO THE DISK AND ASSIGNING DSK THE LOGICAL NAME "3".
THIS IS BECAUSE ALMOST EVERY ASSEMBLY MUST READ S.MAC
AND ABOUT HALF THE APPROPRIATE FEATURE FILE ON LOGICAL
V14 DEVICE 3. IF YOU HAVE ENOUGH ROOM, YOU CAN COPY ALL
V14 6 TAPES OF THE MONITOR SOURCES ONTO THE DISK AND
V14 ASSIGN THE DSK AS LOGICAL DEVICES "3","4","5","6","14",
V14 AND "15".

V14 SECTIONS 3.1.2 THROUGH 3.1.6 DESCRIBE HOW TO USE EACH
OF THE EIGHT CCL FILES TO PRODUCE RELOCATABLE BINARY
AND/OR LISTINGS OF THE MONITOR.

V16 3.1.2 FILE S5ØRLX.CCL WILL PRODUCE JUST MONITOR BINARY
V16 FILES FOR A 10/50 SYSTEM WITH A DISK(5000 BLOCKS NEEDED).
V16 NO DECTAPES ARE NEEDED.

V16 3.1.2.1 COPY ALL MAC FILES ONTO DISK. (USE FAILCD LOOGED-IN UNDER 10,7).
V16 3.1.2.2 TYPE :
V16 .R MACRO
V16 -
V16 *S5ØRLX.CCL@
V16 -

3.1.3 FILES S40RL4.CCL AND S40RL5.CCL S40RL6.CCL WILL PRODUCE
JUST MONITOR BINARY FILES FOR A 10/40
V14 SYSTEM WITH 3 DECTAPES.

3.1.3.1 TYPE
V14 ASSIGN DTA 3
V14 ASSIGN DTA 4
V14 ASSIGN DTA RL4

TO ANY TIME SHARING MONITOR. THE MONITOR WILL RESPOND
WITH THE PHYSICAL DEVICES ACTUALLY ASSIGNED. MOUNT
V14 MONITOR SOURCE TAPE 3 ON LOGICAL UNIT "3",
V14 MONITOR SOURCE TAPE 4 ON LOGICAL UNIT "4",
V14 AND A BLANK TAPE ON LOGICAL UNIT "RL4". MAKE SURE
V14 "3", AND "4" ARE WRITE LOCKED AND THAT "RL4" IS SET
V14 TO WRITE.

3.1.3.2 COPY THE CCL FILE ONTO "RL4":
V14 .R PIP
V14 *RL4:/Z*DTA3:S40RL4.CCL/B/X
V14 *<CONTROL>C

3.1.3.3 ASSEMBLE SOURCE TAPE 4
BY TYPING:

.R MACRO

TO THE MONITOR, FOLLOWED BY:

V14 RL4:S40RL4.CCL@

FOLLOWED BY A CARRIAGE RETURN TO MACRO. WHERE @ IS THE
"EACH" SIGN (NOT ALT-MODE). THIS WILL CAUSE MACRO TO READ
V14 COMMAND FILE S40RL4.CCL. IT WILL TYPE OUT THE TITLE OF
EACH PROGRAM AS IT IS ASSEMBLED. IT WILL PRINT THE NUMBER
OF ERRORS ONLY IF IT FINDS ANY (WHICH IT SHOULDN'T).
WHEN ALL THE ASSEMBLIES ARE COMPLETE, MACRO WILL PRINT
EXIT AND THE TELETYPE WILL BE RETURNED TO MONITOR MODE.

3.1.3.4 LIST THE DIRECTORY OF LOGICAL DECTAPE "RL4", BY TYPING:

.R PIP
V14 *TTY:*RL4:/L
*<CONTROL>C

TO THE MONITOR AND PIP. CUT OUT AND TAPE THE DIRECTORY
V14 ON THE UNIT RL4.

3.1.3.5 REMOVE LOGICAL DECTAPE "RL4", MOUNT A SECOND
v14 BLANK TAPE ON LOGICAL UNIT "RL4"

3.1.3.6 TELL THE MONITOR THAT YOU HAVE MOUNTED NEW DECTAPES BY
TYPING THE ASSIGN COMMAND FOR EACH PHYSICAL UNIT. THIS
IS IMPORTANT! IF YOU DO NOT DO THIS, THE MONITOR WILL
USE THE DIRECTORY IT HAS IN CORE RATHER THAN THE ONE ON
TAPE.
EXAMPLE: ASSUME THAT THE MONITOR HAD ASSIGNED PHYSICAL
v14 DEVICE DTA7 AS LOGICAL UNIT "RL4".
TYPE
v14 ASSIGN DTA7 RL5
v14 SIMILARLY, ASSIGN "5" ON THE DRIVE USED FOR "4".

3.1.3.7 REPEAT STEPS 3.1.3.2 THROUGH 3.1.3.6 FOR EACH TAPE
v14 TO BE ASSEMBLED.

3.1.4 FILE S40BTH.CCL WILL PRODUCE BOTH MONITOR REL (RELOCA-
TABLE BINARY) FILES AND CREF INPUT IN PARALLEL FOR A
V14 10/40 SYSTEM USING A 10/40 SYSTEM WITH 1 MAGTAPE AND 7
DECTAPES.

3.1.4.1 USE THE MONITOR ASSIGN COMMANDS TO ASSIGN SEVEN DECTAPES WITH
V14 LOGICAL NAMES "3","4","5","6", "RL4", "RL5", AND "RL6";
V14 ALSO ASSIGN A MAGTAPE "M", MOUNT MONITOR SOURCES 3,4,5,
V14 AND 6 ON LOGICAL UNITS "3","4","5", AND "6" AND BLANK
V14 DECTAPES ON LOGICAL UNITS "RL4", "RL5", AND "RL6", CLEAR
V14 THE DIRECTORIES ON "RL4", "RL5" AND "RL6" USING PIP AND
V14 COPY S40BTH.CCL FROM LOGICAL UNIT "3" TO "RL4".
V14 INITIATE THE ASSEMBLY PROCESS BY TYPING:

```
        .R MACRO <CR>
V14     *RL4:S40BTH.CCL@<CR>
```

V14 3.1.5 FILES S50RL4.CCL, S50RL5.CCL AND S50RL6.CCL, ETC., WILL
PRODUCE JUST REL (RELOCATABLE BINARY) FOR A 10/50
SYSTEM USING EITHER A 10/40 OR A 10/50 SYSTEM WITH
V14 3 DECTAPES.

3.1.5.1 TO ASSEMBLE THE MONITOR WITH JUST FOUR DECTAPES PROCEED
AS FOLLOWS:

```
V14     .ASSIGN DTA 3
V14     .ASSIGN DTA 4
V14     .ASSIGN DTA RL4 [MAKE LOGICAL ASSIGNMENTS CORRESPOND TO .CCL FILES]
V14     [MOUNT MONITOR SOURCES 3,4 ON LOGICAL UNITS "3","4",
        MOUNT A BLANK DECTAPE ON LOGICAL UNIT "RL4" WRITE ENABLED]
        .R PIP
V14     *RL4:/Z*4:S50RL4.CCL/B/X          [COPY CCL FILE]
        *<CONTROL>C
        .R MACRO
V14     *RL4:S50RL4.CCL@   [INITIATE FIRST PORTION OF ASSEMBLIES]
        EXIT''             [MACRO TERMINATES]
        .R PIP
V14     *TTY:=RL4:/L       [LIST THE DIRECTORY OF UNIT "RL1"]
        *<CONTROL>C
V14     .ASSIGN DTAN RL5  [WHERE DTAN IS PHYSICAL UNIT ASSIGNED TO "RL1"]
        [REMOVE TAPE FROM RL1, ATTACH IT'S DIRECTORY AND MOUNT A BLANK
        DECTAPE ON WHAT IS NOW LOGICAL UNIT RL2]
V14     REPEAT ABOVE SUBSTITUTING "RL5" FOR "RL4" AND "5" FOR "4"
V14     REPEAT AGAIN FOR EACH TAPE.
```

THE FOLLOWING INSTRUCTIONS ASSUME THAT YOU HAVE A DISK
V14 WHICH WILL HOLD AT LEAST MONITOR SOURCE TAPE 3,
ZERO DISK DIRECTORY.

3.1.5.2 COPY THE MAC FILES ON MONITOR SOURCE TAPE 3 TO
THE DISK BY TYPING

V14 .ASSIGN DTA 3

V14 TO THE MONITOR AND MOUNTING MONITOR SOURCE TAPE 3 ON
THE PHYSICAL UNIT THE MONITOR ASSIGNED,

TYPE

.R PIP

TO THE MONITOR FOLLOWED BY

V14 *DSK:/X/B*3:*.MAC
TO PIP WHICH WILL CAUSE PIP TO TRANSFER ALL OF THE FILES
WITH EXTENSION .MAC TO THE DISK, THE /X SWITCH CAUSES PIP
TO RETAIN THE SAME NAMES ON THE DISK, RATHER THAN
COMBINING ALL OF THE FILES INTO ONE, THE /B SWITCH MEANS
COPY IN BINARY AND SHOULD ALWAYS BE USED WHEN COPYING WITH
V14 /X AS A HABIT BECAUSE WITHOUT IT, PIP DOUBLES THE LENGTH OF YOUR FILE
WHEN GOING FROM DISK TO DECTAPE, USING THE /B SWITCH
IS ALSO A GOOD HABIT SINCE YOU NEVER HAVE TO WORRY ABOUT
WHETHER THE DATA IS REALLY BINARY OR NOT,

3.1.5.3 COPY AS MANY OF THE OTHER MONITOR SOURCES ONTO THE DISK AS
V14 YOU HAVE ROOM FOR ALTHOUGH JUST TAPE 3 GREATLY INCREASES
EFFICIENCY AS S.MAC AND THE FEATURE FILES ARE ON IT, BE
SURE TO ASSIGN THE DECTAPE UNIT BEFORE MOUNTING EACH
V14 TAPE, AT LEAST THE CCL FILES MUST BE COPIED TO DISK,

3.1.5.4 FOR EVERY MONITOR SOURCE TAPE THAT YOU ARE GOING TO LEAVE
ON THE DECTAPE DRIVES, MAKE SURE THAT YOU HAVE GIVEN THEM
V14 THE APPROPRIATE LOGICAL NAMES "4","5", AND "6",ETC, FOR
EVERY TAPE YOU COPIED TO THE DISK TYPE:

V14 .ASSIGN DSK 3
V14 .ASSIGN DSK 4
V14 .ASSIGN DSK 5
V14 .ASSIGN DSK 6
V14 ETC,

IT IS POSSIBLE TO GIVE THE DISK MANY LOGICAL NAMES
SIMULTANEOUSLY WHILE EACH DECTAPE CAN HAVE ONLY ONE
LOGICAL NAME, IF MONITOR PRINTS OUT LOGICAL NAME ALREADY
IN USE, DEVICE XXX ASSIGNED, FIRST DEASSIGN OLD NAME
V14 (SAY IT WAS 3) BY TYPING
V14 DEASSIGN 3
FOLLOWED BY:
V14 ASSIGN DSK 3
OTHERWISE YOUR LOGICAL NAME WILL BE FOR A DEVICE YOU DID
NOT INTEND,

3.1.5.5 FINALLY GIVE THE DISK THE LOGICAL NAMES RL4, RL5, RL6, ETC,, BY TYPING:

```
V14          .ASSIGN DSK RL4
V14          .ASSIGN DSK RL5
V14          .ASSIGN DSK RL6
V14            ETC,
```

V14 MACRO WILL WRITE THE REL FILES ON DEVICES "RL4", "RL5" AND "RL6".

3.1.5.6 PERFORM THE ASSEMBLIES BY TYPING:

```
             .R MACRO<CR>
V14          *S5=RL4,CCL@<CR>
                          [MACRO TYPES PROGRAM NAMES]
             EXIT         [MACRO NOTES TERMINATION]
             .START<CR>
```

V14 REPEAT FOR EACH TAPE.

```
V14     3.1.6 FILE S50BTH.CCL PRODUCES BOTH REL (RELOCATABLE BINARY)
V11           FILES AND A CREF INPUT LISTING FILE FOR A 10/50 SYSTEM
V11           USING A 10/40 OR A 10/50 SYSTEM WITH 1 MAGTAPE AND
V14           11 DECTAPES.  IF YOU DO NOT HAVE A DISK PROCEED AS FOLLOWS:
V14           .ASSIGN DTA 3
V14           .ASSIGN DTA 4
V14           .ASSIGN DTA 5
V14           .ASSIGN DTA 6
V14           .ASSIGN DTA 14
V14           .ASSIGN DTA 15
V14           .ASSIGN DTA RL4
V14           .ASSIGN DTA RL5
V14           .ASSIGN DTA RL6
V14           .ASSIGN DTA RL14
V14           .ASSIGN DTA RL15
V11           .ASSIGN MTA M    [ASSIGN MAGTAPE FOR LISTING]
V14           [MOUNT MONITOR SOURCES 3,4,5,6,14,15 ON LOGICAL UNITS "3","4","5","6","14","15",
V14           MOUNT BLANK DECTAPES IN LOGICAL UNITS "RL4","RL5","RL6","RL14","RL15",
V11           MOUNT A BLANK MAGTAPE WITH THE WRITE RING IN, ON THE
V11           PHYSICAL MAGTAPE THE MONITOR ASSIGNED TO YOU.]
V11           .R PIP
V14           *RL4:←/Z
V14           *RL5:←/Z
V14           *RL6:←/Z          [CLEAR THE DIRECTORIES OF UNITS "RL4","RL5",
V14           *RL14:←/Z
V14           *RL15:←/Z
V14           *RL4:/X/B←3:S50BTH.CCL [COPY S50BTH.CCL TO "RL4"]
V11           *<CONTROL>C
V11           .R MACRO
V14           *RL4:S50BTH.CCL*   [START THE ASSEMBLY PROCESS]
V11           [MACRO WILL TYPE EXIT WHEN IT HAS FINISHED ASSEMBLING AND
V11           LISTING THE ENTIRE MONITOR]
V11           IF YOU DO HAVE A DISK COPY AS MANY OF MONITOR
V11           SOURCES TAPES 1 THROUGH 4 AS WILL FIT ON THE DISK AND
V11           GIVE THE DISK LOGICAL NAMES CORRESPONDING TO THE SOURCE TAPE
V11           NUMBERS RATHER THE DECTAPES AS ABOVE, THE DISK MAY
V11           ALSO BE USED TO RECEIVE THE REL FILES BY TYPING
V11           .ASSIGN DSK RL1
V14           .ASSIGN DSK RL6
V14           .ASSIGN DSK RL14
V14           .ASSIGN DSK RL15
```

3.1.7 ASSEMBLE MONITOR SOURCES BY HAND USING MACRO
[INCLUDED HERE FOR COMPLETENESS ONLY SEE SECTION 3.1.1 THROUGH 3.1.6 FOR AUTOMATIC ASSEMBLY OF ENTIRE MONITOR]

3.1.7.1 DESCRIPTION OF MONITOR SOURCES

THERE ARE OVER 30 MACRO SOURCE FILES TO THE COMPLETE MONITOR. SOME OF THE FILES MAY BE ASSEMBLED BY THEMSELVES WHILE OTHERS MUST BE ASSEMBLED WITH A SYSTEM SYMBOL DEFINITION FILE CALLED S.MAC

WITHIN S, CERTAIN SPECIAL SYMBOLS ARE DEFINED TO HAVE THE VALUE 0 OR -1. THESE VALUES CONTROL THE ASSEMBLY OF CERTAIN FEATURES IN THE MONITOR. THESE SYMBOLS ARE CALLED FEATURE SWITCHES AND ALL BEGIN WITH THE LETTERS FT. IF A SWITCH IS ASSIGNED THE VALUE -1, THE ASSOCIATED FEATURE WILL APPEAR IN THE ASSEMBLY. IF ASSIGNED THE VALUE 0, THE FEATURE WILL NOT APPEAR. SEE LISTING OF S.MAC FOR A DESCRIPTION OF EACH FEATURE.

FILES ASSEMBLED WITH THE VERSION OF S.MAC SUPPLIED WILL CONTAIN THE FULL DUPLEX FEATURE. IN ORDER TO REMOVE THE FULL DUPLEX FEATURE, I.E. PRODUCE A HALF-DUPLEX VERSION OF THE FILES, IT IS NECESSARY TO SET THE SWITCH FTCCL=0 THE VALUE OF THE FEATURE SWITCH FTTTYSER=0. THIS MAY BE DONE WITH TECO. WHEN ASSEMBLING A HALF DUPLEX LIBRARY, IT IS NECESSARY TO SET THE SWITCH FTCCL=0 IN FT50S.MAC IF IT IS REQUIRED IN THE ASSEMBLY OF YOUR PARTICULAR CONFIGURATION.

SOME OF THE FILES MUST BE ASSEMBLED WITH BOTH S AND AN APPROPRIATE FEATURE SYMBOL DEFINITION FILE WHICH CONTROLS CONDITIONAL ASSEMBLY DETERMINED BY WHICH OF THE 3 SYSTEMS IS BEING ASSEMBLED FOR. EACH FILE CONTAINS FEATURE SWITCHES WHICH SPECIFY WHICH OF THE 3 POSSIBLE SYSTEMS ARE TO BE ASSEMBLED.] THESE FEATURE SWITCHES ARE IN SEPARATE FILES CALLED:

	SYSTEM	FILE NAME	
	40N	FT40N.MAC	10/40 NON-DISK
V11	40D	FT40D.MAC	10/40 DISK (ALL TYPES)
V11	50D	FT50S.MAC	10/50 SWAPPING (ALL DISK TYPES)

(CONT'D)

EXPLANATION OF SYSTEM NAMES:
THE FIRST 2 CHARACTERS SAY WHETHER THE SYSTEM IS NON-SWAPPING (4) OR SWAPPING (50)
THE THIRD CHARACTER SAYS NON-DISK (N), NON-SWAPPING DISK (D), SWAPPING (S)

AFTER ASSEMBLING THE REQUIRED SOURCES, THE RELOCATABLE BINARY(.REL) FILES MUST BE COMBINED INTO A LIBRARY FILE USING PIP SO THAT THEY MAY BE LOADED USING THE REGULAR LOADER BEFORE ASSEMBLING THE MONITOR THE USER MUST DECIDE WHETHER HE WANTS FULL OR HALF DUPLEX TELETYPE SERVICE. WE RECOMMEND FULL DUPLEX BECAUSE IT ALLOWS CCL IN DISK SYSTEMS AND HAS MANY USER CONVENIENCES. THE LIBRARY FILE CANNOT BE MADE TO HANDLE BOTH TYPES TOGETHER DEPENDING ON MONGEN ANSWER SO THE DECISION MUST BE MADE BEFORE ASSEMBLY. THE FOLLOWING INSTRUCTIONS ARE FOR ASSEMBLING FOR 1 OF 3 LIBRARY FILES:

	SYSTEM	LIBRARY FILE NAME
V11	4[illegible]	5[illegible]XX.REL
V11	4[illegible]	50XX.REL
V11	5[illegible]	55XX.REL

(CONT'D)

THE USER MUST DECIDE WHICH OF THE ABOVE THREE FILES HE WISHES TO CREATE IF HE IS CREATING A NEW LIBRARY; OTHERWISE, HE MAY MERELY DETERMINE THE SYSTEM DESIGNATION FOR HIS "EXISTING LIBRARY" FILE FROM THE ABOVE TABLE.

ALTHOUGH IT IS POSSIBLE TO ASSEMBLE, PIP, AND LOAD A MONITOR WITH JUST 2 DECTAPES PLUS A CUSP TAPE THE FOLLOWING DIRECTIONS ASSUME THAT THERE ARE TWO UNITS FOR SOURCES, ONE FOR BINARY, AND ONE SCRATCH TAPE FOR CREF INTERMEDIATE OUTPUT(IF LISTING DESIRED), PLUS THE CUSP TAPE.

MOUNT MONITOR SOURCE TAPE 3 ON UNIT WITH LOGICAL NAME 3 AND MONITOR SOURCE TAPE 4 ON UNIT WITH LOGICAL NAME 4, MOUNT TAPE TO RECEIVE BINARY FILES ON UNIT WITH LOGICAL NAME RL4. IF YOU HAVE A DISK, COPY S.MAC AND FT----.MAC FROM 3 TO DSK USING PIP:

```
.R PIP
*DSK:/X/B←3:S,MAC,FT----,MAC
*↑C
```

AND IN THE FOLLOWING ASSEMBLY INSTRUCTIONS SUBSTITUTE DSK:S,FT---- FOR 3:S,FT----

DURING THE ASSEMBLIES, SUBSEQUENT SOURCE TAPES 5,6,14, AND 15 WILL BE SUBSTITUTED FOR 4. HOWEVER SOURCE TAPE 3 MUST ALWAYS BE PRESENT UNLESS THE FILES S, AND THE DESIRED FT---- FILES ARE COPIED FROM SOURCE TAPE 3 TO SOURCE TAPE 4, 5, 6, 14, AND 15 OR YOU HAVE A DISK, SINCE THEY ARE RE-READ FOR EACH ASSEMBLY. COPYING ALL OF THE OTHER SOURCES ONTO THE DISK REDUCES THE ASSEMBLY TIME FROM 20 MINUTES TO ABOUT 10 MINUTES ELAPSED TIME, SO IT NEED BE DONE ONLY IF YOU HAVE A LOT OF DISK SPACE. FOR EACH ASSEMBLY, THE USER SHOULD ALTERNATIVELY RUN MACRO AND CREF. OF COURSE IF THE USER DOES NOT WANT A CREF(CROSS-REFERENCE) LISTING, HE NEED ONLY RUN MACRO. HOWEVER, THE INSTRUCTIONS ASSUME THAT A CREF LISTING IS WANTED FOR EACH SUB-PROGRAM OF THE MONITOR. THUS THE PATTERN OF TYPE-IN IS:

(CONT'D)

```
,R MACRO

*RLXI------,/C+SIS,FT----,------
                    [SOMETIMES S IS OMITTED, MOSTLY
                    FT---- IS OMITTED] [MACRO PUTS CREF
                    INTERMEDIATE ON LOGICAL DEVICE DSK
                    WITH FILE NAME CREF.LST; IF CREF IS
                    NOT RUN, OMIT /C BUT INCLUDE ","]

THERE ARE NO ERROS

PROGRAM BREAK IS ------

-K CORE USED


(INSTRUCTIONS ENCLOSED WITHIN < > OMITTED IF NOT RUNNING
CREF)

<
*↑C

,R CREF

*[USER TYPES JUST CARRIAGE RETURN WHICH CAUSES CREF TO
READ FILE CREF.TMP FROM LOGICAL DEVICE DSK AND PUT CREF
LISTING ON LOGICAL DEVICE LPT]

-K CORE

*↑C

,R MACRO >

*[NEXT FILE TO BE ASSEMBLED, ETC]
```

(CONT'D)

```
            THE REASON THAT MACRO AND CREF ARE RUN ALTERNATELY,
            IS TO REDUCE THE AMOUNT OF CREF INTERMEDIATE STORAGE
            REQUIRED.  IF A USER HAS A MAG-TAPE, HE CAN PUT ALL THE
            CREF INTERMEDIATE ON MAGTAPE BY RUNNING ALL THE ASSEM-
            BLIES FIRST, FOLLOWED BY ALL THE CREF'S. THIS MAKES
            FOR MUCH LESS TYPING, SINCE THERE IS NO NEED TO TYPE
            CONTROL C FOLLOWED BY A MONITOR COMMAND ON EACH ASSEMBLY,
            WHEN THE CREF'S ARE RUN THE USER MERELY TYPES CARRIAGE
            RETURN AS INPUT TO EACH ASSEMBLY OUTPUT USING THE MAG
            TAPE AS INPUT AND LPT AS OUTPUT.

            MONITOR COMMANDS:
            ASSIGN DTA 3                [SOURCE TAPE 3]
              DTAN ASSIGNED             [MONITOR ASSIGNS DTA #N WITH
                                        LOGICAL NAME 3]
            ASSIGN DTA 4                [SOURCE TAPE 4]
              DTAN ASSIGNED             [MONITOR ASSIGNS TAPE WITH
                                        LOGICAL NAME 4]
            ASSIGN DTA DSK              [MACRO INTERMEDIATE OUTPUT,
                                        CREF INPUT]
              DTAN ASSIGNED             [MONITOR ASSIGNS DTA WITH
                                        LOGICAL NAME DSK]
            ASSIGN DTA RL4              [BINARY OUTPUT FROM MACRO]

              DTAN ASSIGNED             [MONITOR ASSIGNS TAPE WITH
                                        LOGICAL NAME BIN]
            .R PIP              [RUN PIP CUSP]
            *RL4:←/Z                    [CLEAR DIRECTORY FOR BINARY]
            *DSK:←/Z                    [CLEAR DIRECTORY FOR TMP LISTING
                                        (IF NO DISK)]
            *
COMMON        COMMON DATA STORAGE FOR MONITOR
              CONFIG                    CONFIGURATION DEPENDENT MONGEN
                                        DIALOG OUTPUT
              COMMON                    CONFIGURATION INDEPENDENT MONITOR
                                        DATA AREA
              [ALWAYS REQUIRED]
              [APPEARS ON DECTAPE RL4, SINCE IT CONTAINS ALL THE
              EXTERNALS WHICH
              LOAD THE PROPER DEVICE ROUTINES FOR YOUR CONFIGURATION]

              ALL:    RL4:COMMON,FIR,/C←IS,CONFIG,4:COMMON

V11     BTHINT  DATA LINE SCANNER + COMPUTER-COMPUTER-INTERFACE
V11                DEVICE DEPENDENT PART OF TELETYPE SERVICE
V11           [ASSEMBLE ONLY IF BOTH DC10 AND DA10 ARE PART OF
V11             YOUR HARDWARE CONFIGURATION]
V11
V11           ALL:    RL4:BTHINT,/C←3:S,4:BTHINT
```

```
(CONT'D)

CCIINT          COMPUTER-COMPUTER-INTERFACE - DEVICE
                        DEPENDENT PART OF TELETYPE SERVICE
                [ASSEMBLE ONLY IF CCI IS PART OF HARDWARE CONFIGURATION]

                ALL:    RL4:CCIINT,/C-3:S,4:CCIINT

CDPSER                  CARD PUNCH SERVICE (CP-10)
                        [ASSEMBLE ONLY IF PART OF YOUR CONFIGURATION]
                ALL:    RL4:CDPSER,/C-3:S,4:CDPSER

CDRSRX                  CARD READER SERVICE ROUTINE FOR PDP-10
                        (CR-10)
                        [ASSEMBLE ONLY IF PART OF HARDWARE CON-
                        FIGURATION]

                ALL:    RL4:CDRSRX,/C-3:S,4:CDRSRX

CDRSR6                  CARD READER SERVICE ROUTINE FOR PDP-6
                        [ASSEMBLE ONLY IF PART OF HARDWARE
                        CONFIGURATION]

                ALL:    RL4:CDRSR6,/C-3:S,4:CDRSR6

CLOCK1                  CLOCK,CONTEXT SWITCHING AND JOB START
                        AND STOP ROUTINES
                APRINT  HIGH PRIORITY PROCESSOR INTERRUPT
                        ROUTINE
                CLOCK   LOW PRIORITY CLOCK INTERRUPT ROUTINE
                RUNCSS  ROUTINES TO START AND STOP USER JOBS
                [ALWAYS REQUIRED]

V11             40N:    RL4:CLOCK1,/C-3:S,FT40N,4:CLOCK1
V11             40D     "     -3:S,FT40D,4:CLOCK1
V11             50S:    "     -3:S,FT50S,4:CLOCK1

CLKCSS                  JOB SCHEDULING ALGORTIHM FOR NON-SWAPPING
                        I.E., 10-40, SYSTEMS [REQUIRED FOR NON
                        SWAPPING SYSTEMS ONLY] [USES FEATURE
                        SWITCH FTDISK]

V11             40N:    RL4:CLKCSS,/C-3:S,FT40N,4:CLKCSS
V11             40D:    "     -3:S,FT40D,4:CLKCSS
V11             50S:[NEVER]
```

(CONT'D)

```
COMCON                  COMMAND DECODER AND SAVGET ROUTINES
        COMCON          MONITOR COMMAND DECODER
        COMCSS          COMMON SUBROUTINES USED BY MONITOR COMMANDS
        SAVGET          THE SAVE AND GET MONITOR COMMANDS
                [ALWAYS REQUIRED]

V11             40N:            RL4:COMCON,/C+3:S,FT40N,4:COMCON
V11             40D:                    "       +3:S,FT40D,4:COMCON
V11             50S:                    "       +3:S,FT50S,4:COMCON

CORE1           CORE ALLOCATION AND SHUFFLING
                [ALWAYS REQUIRED]

V11             40N:            RL4:CORE1,/C+3:S,FT40N,4:CORE1
V11             40D:                    "       +3:S,FT40D,4:CORE1
V11             50S:                    "       +3:S,FT50S,4:CORE1

DCSINT          DATA COMM. SYSTEM - DEVICE DEPENDENT
                                INT, SERV, (PDP-6 630)
                [ASSEMBLE ONLY IF PART OF HARDWARE CONFIGURATION]
            ALL:                RL4:DCSINT,/C+3:S,4:DCSINT

DISSER          TYPE 340 DISPLAY SERVICE ROUTINES
                [ASSEMBLE ONLY IF PART OF HARDWARE CONFIGURATION]
            ALL:        RL4:DISSER,/C+3:S,4:DISSER

DLSINT          DATA LINE SCANNER - DEVICE DEPENDENT
                                INT, SERV, FOR USE WITH TELETYPES
                [ASSEMBLE ONLY IF PART OF HARDWARE CONFIGURATION]
            ALL:                RL4:DLSINT,/C+3:S,4:DLSINT
```

(CONT'D)

```
DTASR.              DECTAPE SERVICE FOR PDP-10(TU-55)
                          DECTAPES, (NEW FORMAT)
            [ASSEMBLE ONLY IF PART OF HARDWARE CONFIGURATION]

             40N:           RL5:DTASRN,/C+3:S,FT40N,5:DTASRN
             40D:                  "          FT40D
             50S:                  "          FT50S


DTCSRN              DECTAPE SERVICE FOR PDP-6(555) DECTAPES
                    WITH PDP-10 COMPATIBLE FILE STRUCTURE.
            [ASSEMBLE ONLY IF PART OF HARDWARE CONFIGURATION AND
            NEW FILE STRUCTURE IS DESIRED]

        ALL:                RL5:DTCSRN,/C+3:S,5:DTCSRN

EDDT        EXECUTIVE MODE DDT (DYNAMIC DEBUGGING TECHNIQUE)
            [ASSEMBLE ONLY IF EDDT MAYBE WANTED
            AS SPECIFIED IN MONGEN DIALOG]

V12     ALL:                RL5:EDDT,LAS,/C+5:EDDT

ERRCON      MONITOR DETECTED ERROR MASSAGE ROUTINES
            [ALWAYS REQUIRED]

V12     ALL:                RL5:ERRCON,/C+3:S,5:ERRCON


JOBDAT      SYMBOL DEFINITIONS FOR JOB DATA AREA (BOTH
            SOURCE CODE AND ASSEMBLY LISTINGS)
            [ALWAYS REQUIRED]

V12     ALL:                RL5:JOBDAT,/C+3:S,5:JOBDAT

LPTSER      LINE PRINTER SERVICE ROUTINE
            [ASSEMBLE ONLY IF PART OF HARDWARE CONFIGURATION]

V12     ALL:                RL5:LPTSER,/C+3:S,5:LPTSER


MOVIE       ROUTINE TO PRINT SNAPSHOT OF SYSTEM ON LPT
            [ALWAYS REQUIRED]
V12     40N:                RL5:MOVIE,LAS,/C+3:S,FT40N,5:MOVIE
V12     40D:                      "            FT40D     "
V12     50S:                      "            FT50S     "
```

(CONT'D)

```
MTASRX          MAGTAPE SERVICE ROUTINE FOR PDP-10 MAGTAPE
                CONTROLLER(TU-20)
                [ASSEMBLE ONLY IF PART OF HARDWARE CONFIGURATION]

V12        ALL:                 RL5:MTASRX,/C+3:S,5:MTASRX


MTBSRX          MAGTAPE SERVICE ROUTINE FOR PDP-10 MAGTAPE
                CONTROLLER TM10B (DATA CHANNEL)
                [ASSEMBLE ONLY IF PART OF HARDWARE CONFIGURATION]

V12        ALL:                 RL5:MTBSRX,/C+3:S,FTTM10,5:MTASRX

MTCSR6          MAGTAPE SERVICE FOR PDP-6 MAGTAPE CONTROLLER(516)
                [ASSEMBLE ONLY IF PART OF HARDWARE CONFIGURATION]

V12        ALL:                 RL5:MTCSR6,/C+3:S,5:MTCSR6

NULSEG          DUMMY HIGH USER SEGMENT HANDLING ROUTINES
                [REQUIRED ONLY IF NO HIGH SEGMENT CODE MAY BE WANTED
                AS SPECIFIED IN MONGEN DIALOG]

V12             40N:     RL5:NULSEG,/C+3:S,FT40N,5:NULSEG
V11             40D:             "          FT40D
V11             50S:             "          FT50S

ONCE       ONCE ONLY OPERATOR DIALOGUE FOR
                MONITOR START-UP


                [ALWAYS REQUIRED]

                40N:     RL5:ONCE,LAS,/C+3:S,FT40N,5:ONCE
V11             40D:             "ONCE            FT40D
V11             50S:             "ONCE            FT50S

PATCH           PATCHING SPACE
                [ALWAYS REQUIRED]

V12        ALL:                 RL5:PATCH,LAS,/C+5:PATCH

PLTSER          CALCOMP PLOTTER SERVICE ROUTINE
                [ASSEMBLE ONLY IF PART OF HARDWARE CONFIGURATION]

V12        ALL:                 RL5:PLTSER,/C+3:S,5:PLTSER

PTPSER          PAPER TAPE PUNCH SERVICE ROUTINE
                [ASSEMBLE ONLY IF PART OF HARDWARE CONFIGURATION]

V12        ALL:                 RL5:PTPSER,/C+3:S,5:PTPSER
```

(CONT'D)

```
PTRSER          PAPER TAPE READER SERVICE ROUTINE FOR PDP-10
                OR PDP-6 [ASSEMBLE ONLY IF PART OF HARDWARE CONFIGURATION]

V12        ALL:                 RL5:PTRSER,/C-3:S,5:PTRSER

PTYSRF          PSEUDO-TELETYPE SERVICE ROUTINE FOR FULL
                DUPLEX TELETYPE SOFTWARE

V12        ALL:                 RL5:PTYSRF,/C-3:S,5:PTYSRF

PTYSRH          PSEUDO-TELETYPE SERVICE ROUTINE FOR HALF DUPLEX
                TELETYPE SOFTWARE

V12        ALL:                 RL5:PTYSRH,/C-3:S,5:PTYSRH

SCHED1          SCHEDULING ALGORITHM FOR THE TIME-SHARING SWAPPING SYSTEM,

CLKCSW          SCHEDULING ALGORITHM

QCSS            QUEUE HANDLING SUBROUTINES
                [SWAPPING SYSTEMS ONLY]

50E:            RL5:SCHED1,/C-3:S,FT50S,5:SCHED1

SCNSRF          TELETYPE SERVICE - FULL DUPLEX -
                SCANNER INDEPENDENT (USES CCIINT,
                DCSINT, OR DLSINT)
                [EITHER SCNSRF OF SCNSRH IS ALWAYS REQUIRED]
                [NEVER BOTH]
```

(CONT'D)

```
SCNSRH          TELETYPE SERVICE - HALF DUPLEX-
                SCANNER INDEPENDENT (USES CCIINT,
                DCSINT, OR DLSINT)
                EITHER SCNSRF OF SCNSRH IS
                ALWAYS REQUIRED]
                [NEVER BOTH]
                [USES FEATURE SWITCHES FTSWAP,
                FTLOGIN]
V14             40N:            RL6ISCNSRF(H),/C*3:S,FT40N,6:SCNSRF(H)
V14             40D:                    "               FT40D
V14             50S:                    "               FT50S

SEGCON          HIGH USER SEGMENT HANDLING
                ROUTINES
                [ALWAYS REQUIRED]
                [USES FEATURE SWITCHES FTSWAP,
                FTDISK]
V14             40N:            RL6ISEGCON,/C*3:S,FT40N,6:SEGCON
V14             40D:                    "               FT40D
V14             50S:                    "               FT50S

SYSINI          MONITOR INITIALIZATION
V14             [ALWAYS REQUIRED]
V14             [USES FEATURE SWITCHES FTSWAP,
V14             FTDISK]
V14             40N:            RL6ISYSINI,/C*3:S,FT40N,6:SYSINI
V14             40D:                    "               FT40D
V14             50S:                    "               FT50S
```

(CONT'D)

```
SYSMAK          MAKES JOB #1 OVERLAY THE MONITOR
                AND BECOME THE NEW MONITOR
V14             ALL:  RL6:SYSMAK,LAS,/C-3:S,6:SYSMAK

TMPUUO          HANDLER FOR IN CORE STORAGE OF SHORT (CCL) FILES,
                [THIS ROUTINE REQUIRED IF USER WANTS TO SPEED
                UP CCL OPERATION AT THE EXPENSE OF ABOUT 28
                WORDS/JOB IN THE MONITOR]
                [USES FEATURE SWITCH FTTMP]
V14             40N:            RL6:TMPUUO,/C-3:S,FT40N,6:TMPUUO
V14             40D:                    "         FT40D
V14             50S:                    "         FT50S

UUOCON          UUO TRAP HANDLER AND DEVICE
                INDEPENDENT UUO ROUTINES
                UUOCON  UUO TRAP HANDLER AND DEVICE
                INDEPENDENT UUO ROUTINES
                IOCSS   COMMON TO SUBROUTINES
                [ALWAYS REQUIRED]
                [USES FEATURE SWITCHES FTSWAP,
                FTDISK]
V12             40N:            RL6:UUOCON,/C-3:S,FT40N,6:UUOCON
V11             40D:                    "         FT40D
V11             50S:                    "         FT50S
```

```
V14     3.1,9,2 DISK RELATED ROUTINES [LEVEL 0]

V14     COMMOD  DISK DATA BASE; CAN INCLUDE THE DISK DATA BASE DUMP
V14             ROUTINE DATDMP
V14             [WITHOUT DATDMP]RL14:COMMOD,FIR,/C*3:S,CONFIG,14:COMMOD
V14             [WITH DATDMP]   RL14:COMMOD,FIR,/C*3:S,CONFIG,TTY:,14:COMMOD,DATDMP
V14                             FTEXTERN==-1
V14                             ↑Z
V14                             ↑Z

V14     ONCMOD  DISK PART OF ONCE ONLY

V14                             RL14:ONCMOD,LAS,/C*3:S,14:ONCMOD

V14     REFSTR  DISK REFRESHER

V14                             RL14:REFSTR,LAS,/C*3:S,14:REFSTR

V14     SWPSER  SWAPPING SERVICE

V14                             RL14:SWPSER,/C*3:S,14:SWPSER

V14     FILSER  DEVICE INDEPENDENT FILE SYSTEM FOR DISK

V14                             RL15:FILSER,/C*3:S,15:FILSER

V14     DPXKON  DISK PACK CONTROLLER ROUTINE
V14             [ASSEMBLE ONLY IF YOU HAVE RP01 OR RP02]
V14                             RL15:DPXKON,/C*3:S,15:KONPAR,DPXKON

V14     FHXKON  FIXED HEAD/DRUM CONTROLLER ROUTINE
V14             [ASSEMBLE ONLY IF YOU HAVE RD10 OR RM10B]
V14                             RL15:FHXKON,/C*3:S,15:KONPAR,FHXKON

V14     MDXKON  BRYANT DISK CONTROLLER ROUTINE
V14             [ASSEMBLE ONLY IF YOU HAVE RB10A]
V14                             RL15:MDXKON,/C*3:S,15:KONPAR,MDXKON
```

3.2 MAKE A MONITOR LIBRARY FILE WITH PIP

3.2.1 ORDERING CONSTRAINTS FOR MONITOR LIBRARY FILE AND FOR LOADING

THE ORDERING CONSTRAINTS ON THE MONITOR LIBRARY FILE ARE VERY FEW.

3.2.1.1 COMMON MUST BE LOADED FIRST SINCE IT HAS EXTERNAL STATEMENTS TO LOAD THE DESIRED ROUTINES FROM THE REST OF THE FILE. IT ALSO HAS THE 140 STARTING LOCATIONS IN IT SO IT MUST BE LOADED FIRST. IT SHOULD NOT BE PART OF THE LIBRARY FILE (ALTHOUGH IT WILL DO NO HARM IF IT IS.)
V14 COMMOD SHOULD BE PLACED AFTER COMMON IF IT IS NEEDED.

3.2.1.2 ALL THE REQUIRED AND OPTIONAL ROUTINES MAY BE LOADED IN ANY ORDER.

3.2.1.3 THE LIBRARY MUST END IN THE FOLLOWING ORDER:

3.2.1.3.1 PATCH MUST BE THE FIRST OF THE LAST ROUTINES, BECAUSE IT CONTAINS THE FIRST UNUSED INSTRUCTION IN MONITOR.

3.2.1.3.2 MOVIE MUST BE NEXT

3.2.1.3.3 SYSMAK MUST BE NEXT

3.2.1.3.4 EDDT, IF LOADED, MUST BE NEXT

3.2.1.3.5 ONCE AND REFRESHER MUST FOLLOW EDDT SO THAT IT WILL NOT TAKE UP ANY ROOM WHEN EDDT IS DESIRED TO BE AVAILABLE FOR EXEC MODE DEBUGGING (ONCE ONLY DIALOG QUESTION).
V14 THIS REQUIRES ONCMOD
V14 REFSTR
V14 ONCE IN THAT ORDER.
V15 IF ONCE IS NOT LAST, THE MONITOR WILL PROBABLY HALT
V15 AFTER DESTROYING THE PROGRAM LOADED AFTER ONCE.

3.2.1.5 KNOWING WHAT THE ORDERING CONSTRAINTS ARE YOU WILL HAVE AN OPPORTUNITY TO DISCOVER SHORT CUTS TO THE ABOVE PROCEDURES. FOR EXAMPLE, IT IS POSSIBLE TO KEEP YOUR OLD LIBRARY FILE AND LOAD A FEW ADDITIONAL AND/OR REPLACEMENT ROUTINES FIRST, SINCE THE GLOBAL REQUESTS WILL BE SATISFIED THE LOADER WILL NOT LOAD THE OLDER LIBRARY ROUTINE. OF COURSE SINCE COMMON MUST BE THE VERY FIRST LOADED ROUTINE, IT MUST PRECEDE ANY OF YOUR ADDITIONAL AND/OR REPLACEMENT ROUTINES.

3.2.2 NOW THAT YOU HAVE CREATED ALL OF THE RELOCATABLE BINARY FILES WITH THE ASSEMBLER, YOU SHOULD COMBINE THEM INTO A MONITOR LIBRARY FILE 5N##.REL FOR 10/40 SYSTEM AND 5S##.REL FOR A 10/50 SYSTEM.

3.2.3 ASSUMING THAT THE BINARIES ARE STILL ON LOGICAL DEVICES
V14 RL4, RL5, RL6,RL14 AND RL15. IF LOGICAL DEVICES RL4, RL5,
V14 RL6,RL14 AND RL15 ARE THE SAME DEVICE, SAY DSK AND IF
V14 MAKING A 10/40 LIBRARY FILE, TYPE:

FOR 10/40 SYSTEMS: (ALL RELS ON DISK):

```
      .R PIP
V14   *DSK:LAST/B*DSK:PATCH.LAS,MOVIE.LAS,SYSMAK.LAS,EDDT.LAS,ONCE.LAS
V14   *DSK:5N##.REL/B*DSK:*.REL,DSK:LAST
V16   *DSK:*.REL/R*.PIR
```

V14 IF LOGICAL DEVICES RL4,RL5, RL6,RL14 AND RL15 ARE THE SAME
V14 DEVICE SAY DSK AND IF MAKING A 10/50 LIBRARY FILE TYPE:

FOR 10/50 SYSTEMS: (ALL RELS ON DSK):

```
      .R PIP
V14   *DSK:LAST/B*DSK:PATCH.LAS,MOVIE.LAS,SYSMAK.LAS,EDDT.LAS,ONCMOD.LAS,REFSTR.LAS,ONCE.LAS
V14   *DSK:5S##.REL/B*DSK:*.REL,DSK:LAST
V16   *DSK:*.REL/R*.PIR
```

V14 IF LOGICAL DEVICES RL4, RL5, RL6,RL14 AND RL15 ARE DIFFERENT
V14 DEVICES, AND IF MAKING A 10/40 LIBRARY FILE, TYPE:

FOR 10/40 SYSTEMS (RELS ON DECTAPE):

```
      .R PIP
V14   *RL4:LAST/B*RL5:PATCH.LAS,MOVIE.LAS,RL6:SYSMAK.LAS,RL5:EDDT.LAS,ONCE.LAS
V14   *RL4:5S##.REL/B*RL4:*.REL,RL5:*.REL,RL6:*.REL,RL14:*.REL,RL15:*.REL,RL4:LAST
```

V14 IF LOGICAL DEVICES RL4, RL5, RL6, RL14 AND RL15 ARE DIFFERENT,
AND IF MAKING A 10/50 LIBRARY FILE, TYPE:

FOR 10/50 SYSTEMS (RELS ON DECTAPE):

```
      .R PIP
V14   *RL4:LAST/B*RL5:PATCH.LAS,MOVIE.LAS,RL6:SYSMAK.LAS,RL5:EDDT.LAS,RL14:ONCMOD.LAS,REFSTR.LAS,RL15:ONCE.LAS
V14   *RL4:5S##.REL/B*RL4:*.REL,RL5:*.REL,RL6:*.REL,RL14:*.REL,RL15:*.REL,RL4:LAST
      *
```

WHERE THE /B SAYS COPY BINARY (WHICH THIS IS) AND THE
*.REL SAYS COPY ALL REL FILES, FILES PATCH.LAS, MOVIE.LAS
SYSMAK.LAS,EDDT.LAS,ONCMOD.LAS,REFSTR.LAS AND ONCE.LAS MUST
BE LAST AND IN THAT ORDER, ALL OF THE OTHER FILES MAY
BE IN ANY ORDER. HOWEVER THEY HAVE BEEN ARRANGED ALPHA-
BETICALLY.

3.3 LOAD MONITOR USING LOADER

TO LOAD THE MONITOR FOLLOW SECTIONS 2.3.1. THROUGH 2.3.4.

3.4 SAVE MONITOR USING MONITOR SAVE COMMAND

3.4.1 TYPE

SAVE DTA1 XXXMON

WHERE XXXMON IS THE NAME OF YOUR MONITOR.

4. PATCHING YOUR MONITOR WITH DDT

4.0 PATCHING COMPONENTS

4.1 PATCHING WITH USER DDT UNDER TIMESHARING

4.2 PATCHING WITH EXEC DDT STAND-ALONE

4.3 PATCHING CONVENTIONS (EITHER DDT).

4.0 REQUIRED COMPONENTS

4.0.1 FOR PATCHING WITH USER DDT UNDER TIMESHARING

4.0.1.1 MONITOR ON DECTAPE WHICH WAS LOADED WITH USER DDT (/D SWITCH TYPED TO LOADER.)

V14 4.0.1.2 UP TO 38K OF USER CORE AVAILABLE TO A SINGLE USER UNDER TIMESHARING

4.0.1.3 A RUNNING TIME SHARING MONITOR

4.0.2 FOR PATCHING WITH EXEC DDT OUT OF TIME SHARING

4.0.2.1 MONITOR ON DECTAPE WHICH WAS LOADED WITH EXEC DDT (ANSWER Y TO MONGEN DIALOG)

4.0.2.2 JUST ENOUGH PHYSICAL CORE TO LOAD MONITOR. THIS TAKES LESS CORE THAN PATCHING WITH USER DDT BUT MACHINE CANNOT BE TIMESHARED.

4.1 PATCHING WITH USER DDT UNDER TIME SHARING.

4.1.1 USER DDT IS RECOMMENDED FOR PATCHING OVER EXEC DDT SINCE MACHINE MAY BE TIME SHARED DURING PATCHING PROCESS. (NOTE A COPY OF THE MONITOR IS PATCHED, NOT THE ONE CONTROLLING THE MACHINE.) HOWEVER IT REQUIRES MORE CORE ON YOUR SYSTEM THAN DOES PATCHING WITH EXEC DDT.
V14 THERE MUST BE AVAILABLE UP TO 38K OF USER CORE AVAILABLE TO A SINGLE JOB UNDER TIME SHARING IN ORDER TO MAKE THE PATCHES WITH USER DDT. IN ORDER TO HAVE LOADED USER DDT WITH YOUR MONITOR YOU MUST HAVE USED THE /D SWITCH. IF YOU DID NOT, YOU MUST RELOAD YOUR MONITOR. YOU DO NOT NEED TO GO THROUGH THE MONGEN DIALOGUE AGAIN, HOWEVER, BE SURE TO ALSO SPECIFY THE /S SWITCH SO THAT LOCAL SYMBOLS ARE LOADED. PATCHING WITH-OUT LOCAL SYMBOLS IS NOT RECOMMENDED.

4.1.2 GET COPY OF YOUR MONITOR BY TYPING:

V14 GET DTAN 5S45A

TO THE RUNNING TIME SHARING SYSTEM. WHEN THE MONITOR RESPONDS WITH:

```
JOB SETUP
.
```

TYPE:

```
DDT
```

WHICH WILL START UP USER DDT.

4.1.3 AFTER ALL PATCHES ARE MADE, TYPE <CONTROL>C, FOLLOWED BY:

V14 SAVE UTAN 5S45B

BE CAREFUL NOT TO TYPE <CONTROL>C TWICE, LEAST YOU RETURN TO MONITOR MODE BEFORE DDT MIGHT MAKE ITS LAST MODIFICATION (SWAPPING SYSTEMS ONLY).

4.1.4 YOU MAY SAVE THE PATCHED MONITOR ON THE DISK TOO, SINCE
V14 THE FORMATS ARE THE SAME IN THE 5 SERIES MONITOR BE SURE TO WRITE MONITOR ON DECTAPE USING MONITOR SAVE COMMAND RATHER THAN COPYING IT WITH PIP, SINCE THIS SAVED FILE MUST BE LOADED WITH TENDMP. ALL SAVED FILES WHICH WILL NEVER BE LOADED BY TENDMP MAY BE COPIED BY PIP (USING /B SWITCH OF COURSE).

4.2 PATCHING WITH EXEC DDT STAND-ALONE

4.2.1 EXEC DDT IS RECOMMENDED ONLY IF YOU DO NOT HAVE ENOUGH CORE FOR PATCHING WITH USER DDT. IT IS ALSO RECOMMENDED THAT ALL PATCHES BE MADE WITH EITHER ONE OR THE OTHER BUT NOT BOTH. THIS IS BECAUSE THE SYMBOL TABLE POINTER BECOMES CONFUSED AND TENDMP WRITES BLOCKS CONSECUTIVELY WHICH CAUSES THE TAPE TO ROCK ON A MONITOR GET FOR EACH BLOCK. HOWEVER, SWITCHING CAN BE DONE IF THESE RESTRICTIONS ARE OBSERVED.

4.2.2 IF THE MONITOR TO BE PATCHED WAS PREVIOUSLY PATCHED BY EXEC DDT, THEN PATCHED BY USER DDT, ONE SPECIAL ACTION MUST BE TAKEN BEFORE EXEC DDT WILL BE ABLE TO REFERENCE THE MONITOR SYMBOL TABLE: THE CONTENTS OF ABSOLUTE LOCATION 116(JOBSYM) MUST BE PLACED IN ABSOLUTE LOCATION 36(DDTSYM)(THE PLACE WHERE DDT EXPECTS TO FIND ITS SYMBOL TABLE POINTER). THIS IS CONVENIENTLY ACCOMPLISHED WITH THE FOLLOWING COMMAND TO EXEC DDT.

```
MOVE    116$X
MOVEM   36$X
```

NOTE: THE FIRST TIME CONTROL WENT TO 141, C(116) IS COPIED TO 36, THEN THE COPY CODE IS OVER WRITTEN.

4.2.3 LOAD THE MONITOR FROM DECTAPE WITH TENDMP, WITHOUT STARTING BY TYPING:

V14 L$5S45A SAV

FOLLOWED BY CARRIAGE RETURN TO TENDMP.
THEN SPECIFY EXEC DDT STARTING ADDRESS AND START IT UP BY TYPING:

141$
G$

4.2.4 WHEN YOU ARE FINISHED PATCHING, TYPE:

774005G

TO RETURN TO 32K TENDMP (37400 IF 16K, 137400 IF 48K, ETC.).

4.2.5 SAVE (DUMP) THE MONITOR ON DECTAPE BY TYPING:

V13 140$
V14 D$5S45S SAV

FOLLOWED BY CARRIAGE RETURN TO TENDMP.
WARNING - DO NOT TYPE A PERIOD BETWEEN NAME AND EXTENSION, BECAUSE TENDMP WILL MAKE IT BE A PART OF NAME, AND MONITOR AND CUSPS WILL NOT BE ABLE TO GET FILE.

4.3 PATCHING CORRECTIONS (EITHER DDT)

THE FOLLOWING TECHNIQUES FOR MONITOR PATCHING ARE PART CONVENTION AND PART NECESSITY; IT IS WELL TO FOLLOW THEM CAREFULLY.

4.3.1 EXEC DDT BEHAVES ALMOST EXACTLY LIKE USER DDT AS DESCRIBED IN THE DDT MANUAL (DEC-1Ø-CDDA-D). THE USER SHOULD BE THOROUGHLY FAMILIAR WITH THIS DOCUMENT BEFORE TRYING TO USE EXEC DDT. EXEC DDT IS ENTERED BY STARTING THE MACHINE AT, OR TRANSFERRING TO, ABSOLUTE LOCATION 141 IN ALL MONITORS. THIS MAY BE ACCOMPLISHED THROUGH COMMANDS TO TENUMP OR BY MEANS OF THE CONSOLE SWITCHES.

4.3.2 ALWAYS PATCH A FRESHLY LOADED COPY OF THE MONITOR-ONE WHICH HAS NOT YET RUN THROUGH THE INITIALIZATION DIALOGUE.

4.3.3 LOADED INTO EACH MONITOR IS A BLOCK OF PATCHING SPACE BEGINNING AT GLOBAL LOCATION PATCH. BY CONVENTION, THE VALUE OF THIS SYMBOL IS NOT CHANGED WHEN PATCHES ARE MADE; INSTEAD, ANOTHER SYMBOL, PAT, IS REDEFINED BY THE USER AFTER EACH PATCH SO AS TO POINT AT ALL TIMES TO THE NEXT FREE LOCATION IN THE PATCHING AREA. THUS, THE VERY FIRST PATCH IN ANY SYSTEM WILL BEGIN AT SYMBOLIC LOCATION PATCH; ALL SUBSEQUENT PATCHES WILL BE MADE STARTING AT SYMBOLIC LOCATION PAT, BUT THIS SYMBOL WILL BE CHANGED TO POINT TO THE NEXT FREE LOCATION AS THE LAST STEP IN MAKING ANY GIVEN PATCH. BY THIS MEANS, THE NEXT PERSON TO PATCH THE MONITOR WILL BE ABLE TO FIND THE PATCHING SPACE.

4.3.4 SUGGESTED PATCHES ARE DISTRIBUTED IN A NOTATION CONSISTENT WITH THE ABOVE MECHANISM. EVERYTHING THE USER MUST TYPE TO EXEC DDT IS INCLUDED IN THE SUGGESTED PATCH INCLUDING THE DDT COMMAND WHICH UNLOCKS THE PROPER SET OF LOCAL SYMBOLS.

4.3.5 THE TIME SHARING MONITORS BEGIN TO ALLOCATE FREE CORE FOR VARIOUS INTERNAL FUNCTIONS JUST ABOVE THE LAST PATCH LOCATION IN USE. FOR THIS REASON IT IS VERY IMPORTANT THAT, AS A LAST STEP IN ANY PATCHING SESSION, THE USER INFORM THE MONITOR OF HOW MUCH ADDITIONAL PATCH SPACE WAS USED. OTHERWISE, THIS SPACE WILL BE ALLOCATED AND THE PATCHES WILL BE DESTROYED. THE USER SHOULD HAVE REDEFINED LOCATION PAT AS THE FIRST FREE LOCATION; THIS HAVING BEEN DONE, IT IS ONLY NECESSARY TO OPEN LOCATION PATSIZ (LOCATION ONCE+1 IN OLDER MONITORS) AND RETYPE ITS CONTENTS AS

```
        MOVEI   TAC,PAT
```

(4.3 CONT'D)

4.3.6 IT IS CONSIDERED GOOD PRACTICE TO CHANGE THE NAME OF A
V14 EVERY TIME IT IS PATCHED. FOR EXAMPLE, 5.18A MIGHT
V14 BECOME 5.18B WHEN PATCHED FOR THE SECOND TIME. IF ALL
APPLICABLE DEC-RELEASED PATCHES THROUGH LEVEL X HAVE
BEEN APPLIED, THEN IT IS OUR CONVENTION TO NAME THE FIELD
V14 IMAGE MONITOR 5.18X. THE NEW NAME SHOULD BE ENTERED AS
7 BIT ASCII TEXT IN LOCATION CONFIG (AND THE SEVERAL
LOCATIONS WHICH FOLLOW - UP TO 24 CHARACTERS).

NOTE: IF THE NAME IS A MULTIPLE OF 5 CHARACTERS LONG,
MAKE SURE THE NEXT WORD CONTAINS A ZERO TO DELIMIT THE
TEXT STRING. LIKEWISE, THE SYSTEM DATE IN LOCATION
SYSDAT AND SYSDAT+1 SHOULD BE UPDATED (IN 7 BIT ASCII TEXT).

4.3.7 NEVER SAVE A MONITOR WHICH HAS ALREADY BEEN RUN, AS IT
WILL BE USELESS FOR RELOADING. IF THE SYSTEM IS ACCIDENTLY
RUN BEFORE SAVED, START AGAIN WITH A FRESH COPY AND
REMAKE THE PATCHES.

4.3.8 INSTRUCTIONS AND AN EXAMPLE FOR PATCHING THE MONITOR.
THE TIME TO MAKE PATCHES IS AFTER THE MONITOR HAS
BEEN LOADED BUT BEFORE IT HAS BEEN RUN. IN ORDER TO PATCH,
IT IS NECESSARY TO HAVE A COPY OF EXEC MODE DDT LOADED
WITH THE MONITOR. IF YOUR SAVED VERSION OF THE
MONITOR DOES NOT CONTAIN DDT, YOU MUST REPEAT THE MONGEN
DIALOG AND RELOAD YOUR MONITOR AND ANSWER YES TO THE
QUESTIONS

```
  LOAD EXEC DDT?
        AND
  LOCAL SYSBOLS?
```

SINCE BOTH ARE NECESSARY FOR PATCHING THE MONITOR. IT IS
POSSIBLE TO PATCH WITHOUT LOCAL SYMBOLS, BUT IT IS NOT
RECOMMENDED. DISTRIBUTED PATCHED WILL ASSUME LOCAL SYMBOLS.

(4,3,Ø CONT'D)

THE PATCHES THEMSELVES ARE GIVEN IN THE FORMAT OF DDT TELETYPE DIALOGUE. THE LOCATIONS ARE RENAMED BY THEIR SYMBOL NAMES. IF A USER DOES NOT HAVE LOCAL SYMBOLS LOADED, HE SHOULD REFER TO HIS MONITOR STORAGE MAP AND LISTING TO TRANSLATE LOCAL SYMBOLS TO GLOBAL SYMBOLS AND LITERALS. THE FIRST COMMAND IN EACH GROUP IS THE COMMAND TO UNLOCK THE LOCAL SYMBOL TABLE IN THE NAMED ROUTINE.

NOTE: $ IN THIS SECTION STANDS FOR THE ALTMODE KEY

```
  EX:     USER TYPES CDRSR6$:      TTY OUTPUT TAB AND
                                   DDT NOW MAY REFERENCE
                                   THE SYMBOL TABLE OF THE
                                   ROUTINE CDRSR6.
```

AT THE END OF THE MONITOR IS A SPACE OF ABOUT 3ØØ(8) LOCATIONS WHICH IS RESERVED FOR MONITOR PATCHES. THIS PATCH BUFFER BEGINS AT GLOBAL LOCATIONS PATCH. TO INSERT A PATCH WITHIN A MONITOR ROUTINE, THE PROCESS IS TO BEGIN AND THEN TO WRITE THE PATCH BEGINNING AT LOCATION PATCH. THE LAST INSTRUCTION IN THE PATCH SHOULD BE A JRST BACK TO THE MONITOR.

```
  EX:     IN THE ROUTINE BAH YOU WOULD LIKE TO CHANGE FROM
          BAH2:   JFCL              BAH2:   JFCL
                  HRRS Ø,Ø TO               ADDI Ø,Ø
                  FMPR 1,1                  HRRS Ø,Ø
                                            FMPR 1,1
```

IN DDT YOU WOULD GIVE THE COMMAND

```
          BAH$:   WHICH UNLOCKS THE LOCAL SYMBOLS IN BAH
```

THEN TYPE

```
  BAH2+1/         WHICH OPENS AND TYPES OUT THE CONTENTS
                  OF BAH2+1
```

THE TTY THEN OUTPUTS
(TAB) HRRS Ø,Ø (TAB)
YOU WOULD THEN TYPE IN

```
JRST PATCH(TAB)   THE TAB CLOSES THE LOCATION BAH2+1
                  WHICH IS NOW MODIFIED AND OPENS LOCATION
                  PATCH
```

(4,3,8 CONT'D)

```
TTY RESPONSE IS
PATCH/ (TAB)Ø(TAB)
YOU THEN INSERT

ADDI Ø,Ø(LINE FEED)
                    LINE FEED CLOSES THE LOCATION
                    AND OPENS THE NEXT ONE

TTY OUTPUTS
PATCH+1/(TAB)Ø(TAB)

YOU INSERT
HRRS Ø,Ø(LINE FEED)

TTY OUTPUTS
PATCH+2/(TAB)Ø(TAB)

YOU INSERT
JRST BAH2+2(LINE FEED)

TTY OUTPUTS'
PATCH+3/(TAB)Ø(TAB)

YOU INSERT
PAT: (CARRIAGE RETURN)
                    (THIS RENAMES THE CURRENT LOCATION WITH
                    GLOBAL SYMBOL PAT

THE ENTIRE DIALOGUE WOULD LOOK LIKE THE FOLLOWING:

BAH3: BAH2+1/      HRRS Ø,Ø        JRST PATCH
PATCH/    Ø        ADDI Ø,Ø
PATCH+1/  Ø        HRRS Ø,Ø
PATCH+2/  Ø        JRST BAH2+2
PATCH+3/  Ø        PAT:
```

THIS LAST STEP IS QUITE IMPORTANT, THE GLOBAL PATCH ALWAYS POINTS TO THE BEGINNING OF THE PATCH AREA, ONLY THE FIRST PATCH IN THIS AREA WILL BEGIN AT LOCATION PATCH, AFTER EACH PATCH IS MADE, THE FIRST UNUSED LOCATION SHOULD BE RENAMED PAT, THIS IS NECESSARY TO DETERMINE WHERE SUCCESSIVE PATCHES SHOULD BEGIN.

(4.3.8 CONT'D)

WHEN THE PATCHING SESSION IS COMPLETE, THE GLOBAL LOCATION PATSIZ SHOULD THEN BE OPENED AND SHOULD BE ALTERED THUS

```
PATSIZ/ MOVEI X,Y        MOVEI TAC,PAT
```

THIS INFORMS THE MONITOR WHERE THE CURRENT END OF THE PATCHES IS. A COMMON ERROR TO FORGET THIS STEP, WHICH WILL CAUSE THE MONITOR TO LOOP WHEN STARTED UP INSTEAD OF RUNNING THE NULL JOB. THIS IS BECAUSE THE MONITOR WILL HAVE CREATED COPIES OF MULTIPLE DEVICE DATA BLOCKS ON TOP OF YOUR PATCH. IN ADDITION, THE GLOBAL LOCATIONS CONFIG AND SYSDAT SHOULD BE UPDATED TO IDENTIFY THE NEW MONITOR VERSION AND DATE OF UPDATE.

```
V14     EX:     CONFIG$7T/      527D    "/527E/
```

$7T IS A COMMAND TO PRINT THE CONTENTS OF THE PRECEDING ADDRESS AS A 7 BIT ASCII CHARACTER.

" MEANS INSERT THE ASCII TEXT FOLLOWING THE / IN THE CURRENT OPENED LOCATIONS TERMINATING THE TEXT WITH THE SECOND /.

```
V14     ALSO    SYSDAT$7T/      01-01   "/04-31/
V14             SYSDAT+1$7T/      -64   "/-70/
```

THIS ALSO COMPLETES THE PATCHING PROCESS. AT THIS POINT THE MONITOR IS READY TO RUN. IT SHOULD NOT BE STARTED, HOWEVER, UNTIL A COPY WITH THE PATCHES IS SAVED ON A DECTAPE FOR FUTURE RELOADING.

AS AN AID TO PATCHING, IN THE DISTRIBUTED PATCH RELEASES, USER INPUTS TO THE TTY ARE IN LOWER CASE; DDT OUTPUT IS IN UPPER CASE.

6. CRASH PROCEDURES IN MONITOR 55.01

V10 THERE IS A NEW, EASIER CRASH PROCEDURE IN MONITOR 55.01. IT IS
V10 DOCUMENTED IN MONCSP.MAN BUT MANY PEOPLE HAVE NOT SEEN IT. IT
V10 IS IMPERATIVE THAT CRASHES BE TAKEN CORRECTLY, ESPECIALLY NOW
V10 THAT CRASHES ARE OCCURRING SO INFREQUENTLY. THE NEW PROCEDURE
V10 SAVES THE HARDWARE STATE OF THE MACHINE, INCLUDING PC; ALL IO
V10 DEVICE STATUS AND AC'S.

V16 A. FILL OUT A SOFTWARE ERROR REPORT FORM(SER). SEE ERROR.FRM
V16 DISTRIBUTED AS A SOURCE FILE.

V16 1. FIND FIRST EMPTY FORM IN SOFTWARE LOGBOOK AND
V16 FILL OUT NEXT HIGHEST SER NUMBER WHERE IT SAYS "ERROR NO."
V16 2. FILL OUT MONITOR VERSION, PATCH LEVEL, DATE, TIME, AND WHO.
V16 3. WRITE DOWN PC, MA, INSTR, REG., AND PI IN PROGRESS
T16 IN BALNKS PROVIDED.
V16 4. CHECK BOX FOR REASON FOR CRASH, IE HALT, NXM, LOOP, HUNG,
V16 CUSP, OTHER

V10 B. FIND CURRENT USER JOB NUMBER IF NOT ZERO.

V10 1. IF SWITCHES ARE NOT SET TO 150 (LOCATION JOB)
V10 SET THEM TO 150 AND PUSH EXAMINE.
V10 2. CONVERT OCTAL LIGHTS TO DECIMAL
V10 3. SHOUT WHO IS JOB #N7, WHERE N IS THE DECIMAL NUMBER.
V10 (IF 0 DO NOT BOTHER, NULL JOB WAS RUNNING)
V10 4. GO TO THAT USER'S TELETYPE AND COLLECT HIS OUTPUT. DO
V10 THIS QUICKLY AS USER'S TEND TO CLEAN UP WHEN SYSTEM
V10 CRASHES, THEREBY DESTROYING USEFUL INFORMATION.

V10 C. SET MONITOR TO TAKE AUTOMATIC DUMP

V10 1. SET ADDRESS SWITCHES TO 30
V10 2. PUSH EXAMINE THIS TO MAKE SURE 30 WAS 0.
V10 3. CHECK MEMORY ADDRESS LIGHTS TO MAKE SURE 30 APPEARS THERE.
V10 (MAYBE NOT IF INTERMITTENT CONTACT OF CONSOLE SWITCHES)
V10 4. SET ONE OR MORE DATA SWITCHES TO NON-ZERO.
V10 5. PUSH DEPOSIT "THIS".
V16 6. PUSH "CONT". MONITOR SHOULD HALT WITH PC=10.

6.1 SAVE CRASHED MONITOR ON DISK WITH BOOTS

V16 THERE ARE MANY ADVANTAGES TO SAVING CRASHED MONIOTRS ON DISK RATHER
V16 THAN DECTAPE.

V16 A. FASTER
V16 B. SAVE ALL JOB DATA AREAS IF MACHINE HAS MORE THAN
V16 64K OF CORE

V16 THE ONLY DISADVANTAGE IS THAT THE SPACE FOR CRASH.SAV MUST BE
V16 PREALLOCATED. THE ONCE ONLY DIALOG ASKS THE NUMBER OF K TO
V16 BE ALLOCATED TO CRASH.SAV ON EACH FILE STRUCTURE. IF YOU HAVE
V16 A SMALL DISK, YOU WILL PROBABLY CHOOSE TO USE TENDMP TO SAVE
V16 CRASH AND RELOAD MONITOR (SEE 6.2 BELOW).

V16 D. LOAD PAPER TAPE BOOTS ASSEMBLED FOR BIGGEST SIZE OF MEMORY
V16 (NO LIMIT) SO THAT ALL OF MEMORY WILL BE DUMPED.

V16 1. SET READ IN DEVICE SWITCHES TO 104 (BITS 5 AND 7 DOWN)
V16 2. PUT BOOTS IN PAPER TAPE READER
V16 3. PUSH STOP, RESET READIN (PAPER TAPE WILL READ-IN)

V16 E. DUMP CORE ON DISK

V16 1. TYPE: DSKB:/D
V16 FOLLOWED BY CARRIAGE RETURN, WHERE DSKB: CONTAINS A
V16 CRASH.SAV ON [1,4] WHICH IS BIG ENOUGH TO HOLD ALL OF CORE.
V16 IF BOOTS RESPONDS WITH A CLICK IMMEDIATELY, IT MAY BE THAT
V16 CRASH.SAV WAS MISTAKENLY ASSIGNED 0 K WHEN FILE STRUCTURE
V16 WAS DEFINED, OR FILE STRUCTURE HAS SPECIFIED WHICH
V16 HAS A CRASH.SAV WITH 0K.

V16 F. RELOAD MONITOR - DO NOT RESTART

V16 1. TYPE JUST CARRIAGE RETURN TO BOOTS. BOOTS WILL LOAD SYSTEM.SAV
V16 FROM SYS.

(6. CONT'D)

6.2 SAVE CRASHED MONITOR ON DECTAPE WITH TENDMP

V10 D. LOAD PAPER TAPE TENDMP ASSEMBLED FOR BIGGEST SIZE OF MEMORY
V10 (UP TO 64K IF YOU HAVE 64K OF MEMORY) SO DUMP ALL OF CORE.

V10 1. SET READ IN DEVICE SWITCHES TO 104 (BITS 5 AND 9 DOWN).
V10 2. PUT TENDMP IN PAPER TAPE READER
V10 3. PUSH STOP, RESET, READIN (PAPER TAPE WILL READIN).

V10 E. DUMP CORE ON DECTAPE

V10 1. MOUNT DECTAPE ON ANY CONVENIENT UNIT, SAY N.
V10 2. SET WRITE ENABLED.
V10 3. TYPE N$ (WHERE $ = ALT-MODE), TAPE N WILL SPIN.
V10 4. TYPE Z$ (WHICH ZERO'S TAPE).
V10 5. TYPE DSCRASH SAV FOLLOWED BY CARRIAGE RETURN. TENDMP
V10 WILL WRITE ALL OF CORE ONTO DECTAPE.

V10 F. LABEL TAPE

V10 1. REMOVE TAPE FROM REEL AND MARK REEL # ON CRASH FORM.
V10 2. PUT LABEL ON TAPE CONTAINING MONITOR NAME, DATE, AND
V10 CRASH SAV.
V16 3. PUT TAPE IN CRASH RACK.

V16 6.3 COPY CRASH FROM DECTAPE TO DISK AND EXPAND.

V10 1. LOGIN UNDER 1,2
V10 2. MOUNT MONITOR CRASH PROGRAMS ON DECTAPE. IF FILEX.
V14 FILDDT.TXT. FD5S01.SAV ARE NOT ON CUSP. AND ASSIGN DECTAPE
V10 (SAY DTAN). (SEE MONCSP.MAN IF YOU HAVE NOT MADE A
V14 FD5S01.SAV FOR YOUR SYSTEM).
V10 3. ASSIGN DECTAPE (SAY DTAX) CONTAINING CRASH.SAV
V10 4. TYPE R FILEX OR RUN DTAN FILEX.

V16 *SYS:SER###.XPN/E/Q+DTAX:CRASH.SAV
V16 -
V16 WHERE ### IS THE SOFTWARE ERROR NUMBER OF THIS CRASH.

```
V16  6.4        IF CRASH WAS DUMPED ON DISK WITH BOOTS ON SYS:CRASH.SAV
V16             EXPAND USING FILEX:

V16             *SYS:SER###.XPN<055>/E←DSK8:CRASH.SAV[1,4]
V16             -
V16             WHERE ### IS THE SOFTWARE ERROR NUMBER OF THIS CRASH.

V10     I. PRODUCE FILDDT SYMBOLIC OUTPUT OF CRASH

V16        1. TYPE:
V14           ASSIGN DTAN SYS IF FD5S01 NOT ON CUSP.
V14        2. TYPE ASSIGN DSK LPT (SO THAT FD5S01 WILL WRITE FILE
V10           SNAP.LST ON DSK).
V14        3. TYPE R FD5S01.
V14        4. FD5S01 WILL COMPUTE FOR 14 TO 18 MINUTES.  SEE WHEN DONE.
V16           YOU MAY TYPE ONE <CONTROL>C AHEAD WHILE IT IS COMPUTING
V16           AND TYPE AHEAD ANOTHER COMMAND OR TWO. SUCH AS:
V16           .LIST SNAP.LST
V10        6. PUT LINE PRINTER LISTING UNDER WHITE BOX.

V16             END OF "MONITR.OPR"
```

8. SYSTAT MONITOR COMMAND

8.1 THE MONITOR COMMAND SYSTAT CAUSES A CUSP TO BE RUN WHICH PRINTS STATUS INFORMATION ABOUT THE SYSTEM. IT MAY EVEN BE TYPED WHEN THE USER IS NOT LOGGED-IN, THUS ALLOWING THE USER TO DETERMINE THE LOAD ON THE SYSTEM BEFORE LOGGING-IN. SOME INSTALLATIONS MAY CONSIDER THE SYSTAT INFORMATION PROPRIETARY, IN WHICH CASE THEY SHOULD CHANGE THE PROTECTION OF THE SAVE FILE (SYSTAT.SAV) ACCORDINGLY, SO THAT ONLY OPERATIONS PEOPLE MAY USE IT.

TO DIVERT THE SYSTAT OUTPUT TO THE LINE PRINTER, ASSIGN THE LPT AND GIVE IT LOGICAL NAME SYSTAT. TO WRITE THE OUTPUT ON THE DISK AS A FILE WITH NAME SYSTAT.TXT, ASSIGN DEVICE DSK AND GIVE IT LOGICAL NAME SYSTAT.

THE BEST WAY TO DESCRIBE THE SYSTAT OUTPUT IS WITH AN EXAMPLE:

```
Status of 5S.0016 LEVEL D at 11:59:11 on 08-May-70

Uptime 01:19:07, 7% Null time = Idle+Lost = 1% + 6%

Job     Who      Where   What    Size    State    RunTime

1       10,612   TTY2    SYSTAT  2K      RN       00:00:01
2 #     20,574   TTY6    PIP     1K      TI       00:02:18
3       **,**    TTY15   SYSTAT  0K      *C       00:00:00
4       **,**    DET     F40     1K      *C SW    00:00:01
5       **,**    DET     LOGOUT  1K      *C SW    00:03:42
6       20,574   TTY12   SYSTAT  2K      *C SW    00:00:01
7       11,554   TTY4    F4021B  9K      TI SWF   00:13:43
8       **,**    DET     PRINTR  4K      RN       00:00:11
9       1,2      TTY13   OMOUNT  5K      SL SWF   00:00:03
10      2,172    TTY1    FDS016  18K     RN       00:35:51
11      30,110   CTY     PIP     1K      TI SW    00:00:05
12      2,171    TTY3    SYSTAT  2K      *C SW    00:00:01
13      11,131   TTY5    TECO    2K      RN       00:00:06
14      **,**    TTY0    LOGOUT  1K      TI       00:00:01

High Segments:
Program Device   Owner   High K  Users

(PRIV)           Job 5   2K SW   1
LOGOUT  DSK      SYS     2K      1
PIP     DSK      SYS     3K SW   1
TECO    DSK      SYS     2K      1
F40     DSK      SYS     9K SW   1
PIP     DSK      20,574  3K      1

Dormant Segments:
Program Device   Owner   High K

COMPIL  DSK      SYS     3K SW
MACRO   DSK      20,574  5K SW
MACRO   DSK      SYS     5K SW
CREF    DSK      SYS     1K SW
LOGIN   DSK      SYS     1K SW
RUNOFF  DSK      20,574  2K SW
COMPIL  DSK      20,574  3K SW
TECO    DSK      20,574  2K SW
```

```
LOADER  DSK     20,574  3K SW

% Swapping space used = 67/475 = 14%
% Virt. Core used = 70/475 = 15%
Swapping Ratio = 70/39 = 1.8

Busy devices:
Device  Job     Why

LPT     8       INIT
DTA0    2       AS
DTA1    13      AS+INIT
DTA2    10      AS
DTA3    9       AS
DTA4    2       AS
DTA5    12      AS
DTA7    7       AS

System Filestructures:
DSKA,DSKB,DSKC,
```

TITLE LINE - THE NAME OF SYSTEM AS SPECIFIED IN MONGEN DIALOG QUESTION AND PATCHED AS ASCII TEXT IN LOCATIONS CONFIG., CONFIG+4, FOLLOWED BY THE TIME OF DAY AND THE DATE.

CPU USAGE - THE NUMBER OF HOURS, MINUTES, AND SECONDS SINCE SYSTEM WAS LOADED INTO CORE. 140 AND 143 RESTARTS DO NOT RESET THIS QUANTITY TO ZERO. THE PERCENT OF UP TIME THAT SYSTEM WAS RUNNING THE NULL JOB. NULL TIME IS DIVIDED INTO TWO CATEGORIES, IDLE AND LOST. THE IDLE TIME IS THE PERCENT OF UPTIME THAT NO JOB WANTED TO RUN, I.E., ALL JOBS WERE HALTED OR WERE IN A WAIT FOR SOME DEVICE. THE LOST TIME IS THE PERCENT OF UPTIME THAT THE NULL JOB WAS RUNNING BUT AT LEAST ONE OTHER JOB WANTED TO RUN (I.E., WAS NOT WAITING FOR A DEVICE) BUT COULD NOT BE RUN BECAUSE OF ONE OF THE FOLLOWING CAUSES:

1. BEING SWAPPED OUT
2. BEING SWAPPED IN
3. ON DISK WAITING TO BE SWAPPED IN
4. MOMENTARILY STOPPED SO DEVICES CAN BECOME INACTIVE IN ORDER TO SHUFFLE JOB IN CORE.

IN A SENSE THE IDLE TIME RATHER THAN ALL OF NULL TIME REPRESENTS THE EXCESS CAPACITY OF THE SYSTEM WHICH CAN BE ABSORBED BY ADDING MORE USERS. THE LOST TIME CANNOT BE USED UNLESS THE JOB MIX IS CHANGED.

(8.1 CONT'D)

JOB DESCRIPTION:

EACH JOB WHICH IS LOGGED-IN HAS THE FOLLOWING INFORMATION TYPED ABOUT ITSELF:

1. JOB NUMBER (JOB)
   - 1A. IF THE JOB HAS A HIGH SEGMENT WHICH HAS BEEN SUPERCEDED, AN EACH SIGN (@) IS PRINTED AFTER THE JOB NUMBER.
   - 1B. IF THE JOB IS USING A HIGH SEGMENT WHICH IS FROM A DIRECTORY OR DEVICE OTHER THAT THE CUSP DIRECTORY ON DEVICE SYS, A NUMBER SIGN (#) IS PRINTED AFTER THE JOB NUMBER.
2. PROJECT-PROGRAMMER NUMBER OF THE USER (WHO). IF THE USER HAS DETACHED HIS TTY FROM THE JOB, **,** IS PRINTED INSTEAD SO THAT ANOTHER USER CANNOT ATTACH TO THE DETACHED JOB. FOR THE OPERATOR AND THE USER HIMSELF, THE NUMBER IS PRINTED.
3. TELETYPE NUMBER (WHERE), CTY MEANS CONSOLE TELETYPE, DET MEANS THE TELETYPE HAS BEEN DETACHED FROM THE JOB.
4. PROGRAM NAME (WHAT) AS SET BY THE GET, R,RUN COMMANDS AND THE LOADER CUSP (SETNAM UUO). IT IS USUALLY THE LOW SEGMENT NAME.
5. PROGRAM SIZE (SIZE) IN THOUSANDS (K=1024 WORDS) OF WORDS.
6. JOB STATE AND SWAPPED STATE (STATE)

↑C	USER HAS TYPED CONTROL C, JOB HAS HAD ERROR OR EXITED.
TI	TELETYPE IO WAIT
DI	DISK IO WAIT
IO	IO WAIT FOR ANY OTHER DEVICE
RN	RUNNING (MAYBE SWAPPED OUT OR ON WAY AS WELL AS IN CORE)
WS	WAIT SATISFIED
TS	TTY WAIT SATISFIED
DS	DISK WAIT SATISFIED
ST	SYSTEM TAPE WAIT
AU	ALTER UFD WAIT
MQ	MONITOR BUFFER WAIT
DA	DISK ALLOCATION WAIT
CB	DISK CORE BLOCK SCAN WAIT
DT	DECTAPE CONTROL WAIT
DC	DATA CONTROL WAIT
MT	MAGTAPE CONTROL WAIT
SL	PROGRAM IS SLEEPING
SW	LOW SEGMENT IS SWAPPED OR ON WAY IN OR OUT
SWF	LOW SEGMENT IS SWAPPED OR ON WAY IN OR OUT AND BECAUSE SWAPPING SPACE IS NEAR FULL THE LOW SEGMENT IS FRAGMENTED ON THE DISK.

7. JOB RUN TIME (RUNTIME) SINCE LOGGED IN.

(8.1 CONT'D)

HIGH SEGMENTS

EACH HIGH SEGMENT CURRENTLY IN AT LEAST ONE JOB'S VIRTUAL ADDRESSING SPACE HAS THE FOLLOWING INFORMATION TYPED OUT:

1. HIGH SEGMENT NAME (PROGRAM), IF THE HIGH SEGMENT IS NOT SHARABLE, THE UNLIKELY NAME (PRIV) FOR PRIVATE IS TYPED, IF THE HIGH SEGMENT HAS BEEN SUPERCEDED, THE UNLIKELY NAME (OBS) FOR OBSOLETE IS TYPED.
2. DEVICE OR FILE STRUCTURE FROM WHICH THE SEGMENT CAME.
3. DIRECTORY NAME (OWNER) FROM WHICH THE HIGH SEGMENT CAME. IF ONE OF THE JOBS USING THE HIGH SEGMENT IS DETACHED AND THE PROJECT-PROGRAMMER NUMBER OF THE HIGH SEGMENT IS THE SAME AS HIS, **,** IS SUBSTITUTED TO FURTHER PROTECT THE USER FROM HAVING HIS DETACHED JOB ATTACHED TO BY SOMEONE ELSE.
4. SIZE OF HIGH SEGMENT (HIGH K) IN THOUSANDS OF WORDS (K=1024 WORDS).
   - 4A. IF HIGH SEGMENT IS SWAPPED OUT AND IS NOT IN CORE, SW IS PRINTED.
   - 4B. IF HIGH SEGMENT IS SWAPPED OUT AND IS NOT IN CORE AND IS FRAGMENTED, SWF IS PRINTED.
   - 4C. IF HIGH SEG IS IN CORE BUT IS FRAGMENTED ON DISK TOO, F IS PRINTED.
5. NO OF USERS IN WHOSE VIRTUAL ADDRESSING SPACE THE HIGH SEGMENT APPEARS.

DORMANT SEGMENTS:

EACH SHARABLE HIGH SEGMENT WHICH IS CURRENTLY NOT IN ANY JOB'S VIRTUAL ADDRESSING SPACE HAS THE FOLLOWING INFORMATION TYPED OUT:

1. HIGH SEGMENT NAME (PROGRAM)

2. DEVICE NAME FROM WHICH IT CAME.

3. DIRECTORY NAME (OWNER) FROM WHICH THE HIGH SEGMENT CAME

4. SIZE OF HIGH SEGMENT (HIGH K) IN THOUSANDS OF WORDS (K=1024 WORDS).
   - 4A. IF HIGH SEGMENT IS SWAPPED OUT AND IS NOT IN CORE, SW IS PRINTED.
   - 4B. IF HIGH SEGMENT IS SWAPPED OUT AND IS NOT IN CORE, AND IS FRAGMENTED, SWF IS PRINTED.
   - 4C. IF HIGH SEGMENT IS IN CORE BUT IS FRAGMENTED ON THE DISK TOO, F IS PRINTED.

PERCENT SWAPPING SPACE USED BY ACTIVE AND DORMANT HIGH AND LOW SEGMENTS IS COMPUTED BY TAKING THE RATIO OF K OF SWAPPING SPACE USED DIVIDED BY THE TOTAL K PREALLOCATED AT REFRESH TIME FOR SWAPPING (EACH HIGH SEGMENT IS ONLY COUNTED ONCE). THIS STATISTIC GIVES THE SYSTEM ADMINSTRATOR SOME INFORMATION FOR DETERMINING THE OPTIMUM AMOUNT OF SWAPPING SPACE TO ASSIGN. TOO MUCH SWAPPING SPACE WASTES DISK SPACE, WHILE TOO LITTLE SPACE CAN CAUSE BAD FRAGMENTATION (HENCE SLOWER SWAPPING) OR EXHAUSTION OF VIRTUAL CORE.

PERCENT VIRTUAL CORE USED BY ACTIVE HIGH AND LOW SEGMENTS IS COMPUTED BY TAKING THE RATIO OF K OF VIRTUAL CORE USED (EACH HIGH SEGMENT IS ONLY COUNTED ONCE NO MATTER HOW MANY USERS ARE SHARING AND DORMANT SEGMENTS ARE NOT COUNTED AT ALL) DIVIDED BY THE TOTAL K PREALLOCATED AT REFRESH TIME FOR SWAPPING. THIS PERCENTAGE CAN BE HIGHER OR LOWER THAN THE PERCENT OF SWAPPING SPACE USED.
THE SWAPPING RATIO IS A MEASURE OF HOW MANY TIMES PHYSICAL CORE IS EXCEED BY THE TOTAL SIZE OF ALL JOBS IN SYSTEM.
THE SWAPPING RATIO IS COMPUTED BY TAKING THE RATIO OF K OF VIRTUAL CORE USED BY ACTIVE HIGH AND LOW SEGMENTS TO THE SIZE OF USER CORE. EACH ACTIVE HIGH SEGMENT IS ONLY COUNTED ONCE AND DORMANT SEGMENTS ARE NOT COUNTED AT ALL).

IF PHYSICAL CORE EXCEEDS THE CURRENT USAGE, THE NUMBER OF K OF CORE LEFT IS REPORTED INSTEAD.

THE PERCENT OF VIRTUAL CORE SAVED BY SHARING IS COMPUTED BY SUMMING THE NO. OF K IN ACTIVE HIGH SEGMENTS MULTIPLIED BY THE NUMBERS OF JOBS MINUS ONE USING THE HIGH SEGMENT AND DIVIDING BY THE SUM OF THAT SAME QUANTITY PLUS THE TOTAL VIRTUAL CORE USED.

BUSY DEVICES:

1. DEVICE NAME (DEVICE)

2. JOB NUMBER USING THE DEVICE (DJOB)

3. HOW DEVICE IS ASSIGNED (WHY)

    ASSIGNED BY CONSOLE ASSIGN COMMAND - AS
    ASSIGNED BY PROGRAM (INIT OR OPEN UUO) - INIT
    ASSIGNED BOTH WAYS - AS+INIT

SYSTEM FILE STRUCTURES:

EACH FILE STRUCTURE KNOWN TO THE SYSTEM IS PRINTED.

SELECTED OUTPUT:

OFTEN A USER IS ONLY INTERESTED IN PART OF THE OUTPUT.
ANY SUBSET OF THE OUTPUT MAY BE SELECTED BY TYPING
ONE OR MORE SINGLE LETTERS AS AN ARGUMENT TO THE SYSTAT
COMMAND. THE LETTERS MAY BE TYPED IN ANY ORDER. THE LETTERS
ARE BDFHJS

B - JUST BUSY DEVICES
D - DORMANT SEGMENTS
F - FILE STRUCTURES
H - EVERYTHING EXCEPT JOB INFORMATION
J - JUST JOB INFORMATION
S - SHORT JOB PRINTOUT - STATE AND RUN TIME ARE NOT PRINTED

EXAMPLE:

.SYSTAT BD
PRINTS BUSY DEVICES AND DORMANT SEGMENTS.

SUBJECT: FAILSAFE VERSION 27 DATE: JUNE 10, 1970

TO: FROM: C. MCCOMAS

THE INFORMATION IN THIS MEMORANDUM IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

FAILSAFE VERSION 27

FAILSAFE VERSION 27 AND ALL LATER VERSIONS ARE DESIGNED TO RUN ONLY WITH THE LEVEL D MONITOR, AND ARE INCOMPATIBLE WITH PRECEDING MONITOR LEVELS.

FAILSAFE VERSION 27 DIFFERS FROM FAILSAFE VERSION 16 (THE LAST LEVEL C RELEASE) IN THE FOLLOWING RESPECTS.

1) THE TAPE FORMAT IS DIFFERENT. THE NEW FORMAT IS AS FOLLOWS:

```
HEADER RECORD:          XWD VER,4
(FOLLOWED BY EOF)       SIXBIT/*FAILS/
                        XWD <SIXBIT/AFE/>,<TAPE #>
                        USUAL FORM OF CREATION TIME & DATE
                        XWD 1,2

                        WHERE VER= THE POSITIVE VERSION# OF
                        THE FAILSAFE USED TO MAKE THE TAPE
```

LAST RECORD OF LAST FILE ON EACH TAPE IS A TRAILER RECORD WHICH IS IDENTICAL TO THE HEADER ON THAT TAPE, EXCEPT WORD 3 RIGHT HALF = -TAPE #.

USER AREA FILES ARE ROUGHLY JUST AS BEFORE EXCEPT THAT INSTEAD OF EACH USER HAVING JUST ONE FILE ON THE TAPE CONTAINING ALL HIS DISK FILES, HE NOW HAS ONE FILE FOR EACH DISK STRUCTURE ON WHICH HE HAS FILES.

EXAMPLE OF ARRANGEMENT OF USERS ON TAPE:

```
DSKA [1,1]      (DSKA UFD'S)
DSKA [1,4]
DSKA [10,10]
DSKA [1,2]
DSKB [1,1]      (DSKB UFD'S)
DSKB [1,4]
DSKB [10,2]
DSKB [10,10]
DSKB [10,14]
DSKB [1,2]
DSKB [11,1411]
```

WITHIN EACH DISK STRUCTURE USER AREA THE FILES ARE THE SAME FORMAT AS BEFORE EXCEPT THAT THE OLD 4 WORD BLOCK IN THE FIRST RECORD OF EACH FILE HAS BEEN EXPANDED FROM THE OLD 4 WORD DIRECTORY ENTRY LOOKUP TO THE NEW 32 WORD EXTENDED LOOKUP ENTRY. IT NOW LOOKS LIKE THIS:

```
FIRST RECORD OF FILE:   XWD -1,N
                        SIXBIT/STRUCTURE NAME/
                        32
                        USER AREA PROJ-PROG#
                        SIXBIT/FILNAM/
                        XWD<SIXBIT/EXT/>,ACC. DATE
                        PROT. MODE. CREAT. DATE
                        26 MORE WORDS OF LOOKUP INFO.
                        FIRST 745 DATA WORDS OF THE FILE
```

UFD'S ARE NOW SAVED, BUT ONLY THE LOOKUP INFO, IS SAVED FOR UFD FILES. NO DATA WORDS ARE SAVED FOR UFD'S. THE 1,1 AREAS CONTAIN ONLY UFD'S & ARE THE FIRST AREA FOR EACH STRUCTURE.

THØSE FILES MARKED BY THE MONITOR AS "DO NOT FAILSAFE" FILES (SAT.SYS, BADBLK.SYS, ETC) ARE NOT SAVED.

2) ALL VERSION 27 COMMANDS ARE THE SAME AS IN VERSION 16, EXCEPT AS FOLLOWS:

/S SAVES ALL STRUCTURES

/SDSKA,DSKC,DPA6 SAVES ONLY THOSE STRUCTURES & DEVICES NAMED.

/U SAVES ALL OF THE USER'S AREAS

/UDSKB SAVES JUST USER'S AREAS ON NAMED STRUCTURES.

A MAX. OF 16 ARGS MAY BE USED AFTER /S OR /U.

RESTORE COMMANDS WORK AS FOLLOWS:

IF POSSIBLE EACH FILE IS RESTORED TO THE STRUCTURE IT CAME FROM. IF THIS IS NOT POSSIBLE, AN ATTEMPT IS MADE TO RESTORE THE FILE TO SOME OTHER STRUCTURE; THE ONE WITH THE MOST ROOM IF POSSIBLE.

THE NEW /G COMMAND HAS BEEN INCLUDED. THIS ENABLES THE USER TO RUN THE /U AND /L COMMANDS FOR A USER OTHER THAN HIMSELF. THE FORMAT IS */GMMM,NNN<C,R,>
THIS CHANGES THE SINGLE-USER PROJECT-PROGRAMMER NUMBER SWITCH FROM THAT OF THE FAILSAFE USER (THE VALUE IT HAS INITIALLY) TO MMM,NNN. THIS NEW VALUE IS RETAINED UNTIL THE NEXT /G COMMAND.

3) OTHER CHANGES IN THE OPERATION OF FAILSAFE:

FAILSAFE DOES NOT CHANGE THE ACCESS DATE OF FILES WHICH IT SAVES OR RESTORES.

ON RESTORING WITHOUT THE /N SWITCH, FAILSAFE DOES NOT OVERWRITE EXISTING DISK FILES WHICH HAVE THE SAME CREATION DATE AND TIME AS A CORRESPONDING FILE ON TAPE.

4) TO RESTORE OLD FORMAT TAPES USE R FAILCD
(A RESTORE ONLY FAILSAFE FOR RESTORING
LEVEL C TAPES TO A LEVEL D DISK).

FLOW CHART FOR /R FOLLOWS:

```
R:      READ UFD INFO FROM TAPE
        IF SOURCE FILE STR. NO LONGER EXISTS, GO TO R5
        IF THIS UFD DOES NOT EXIST ON THIS STR., GO TO R3
R2:     IF NEXT FILE WILL NOT FIT IN EXISTING QUOTA, ERROR
        OTHERWISE GO TO R4
R3:     CREATE A UFD ON THIS STR. WITH ORIGINAL QUOTA
R4:     RESTORE FILE TO THIS STR. (IF IT FITS)
        IF STR. OVERFLOWS, GO TO R6
        OTHERWISE GO TO R2

R5:     IF THIS UFD DOES EXIST IN SOME OTHER FILE STR, GO TO R8
R6:     IF NO OTHER STRUCTURE EXISTS, ERROR
        CHOOSE EMPTIEST STRUCTURE IN SYSTEM
        CREATE UFD WITH INFINITE QUOTA
R7:     RESTORE NEXT FILE (IF IT FITS)
        IF STR. OVERFLOWS, GO TO R6
        OTHERWISE GO TO R7

R8:     IF NEXT FILE DOES NOT FIT IN QUOTA, GO TO R9
        RESTORE FILE (IF IT FITS)
        IF STR. OVERFLOWS, GO TO R9
        OTHERWISE GO TO R8

R9:     IF THIS UFD DOES NOT EXIST IN ANY OTHER STR., ERROR
        OTHERWISE GO TO R8
```

FAILSAFE DOCUMENTATION CHANGES

(CHANGES TO CHAPTER 5, SYSTEM MANAGER'S GUIDE, NOTEBOOK 3)

PAGE

5-1 CHANGE *SAT*.SYS TO SAT.SYS

DELETE "EXCEPT UFD'S" ON /S

ADD PRIMARY FUNCTION /M (AUTOMATICALLY REPEAT LAST SAVE FUNCTION AT STATED INTERVALS)

ADD /G (ENABLE /U, /L, & SELECTIVE RESTORE OF AN AREA DIFFERENT FROM THE USER'S)

5-2 THE TAPE FORMAT HAS CHANGED AS ABOVE.

USRCTD BIT IS NOT USED

5-17 /2 SETS MTA DENSITY TO 200 BPI (INSTEAD OF /Z)

*** FOR MORE EFFICIENT USE OF FAILSAFE ***

WHEN RESTORING AFTER REFRESHING THE DISK, USE THE /N SWITCH TO SPEED THINGS UP. E.G., */N/R
THIS AVOIDS DOING A LOOKUP FOR EACH FILE.

NOTES ON RUNNING FAILSAFE FOR THE NON-[1,2] USER

1) TO SAVE ALL HIS FILES ON A TAPE

```
.AS MTA# FAILSA
.R FAILSA
*/U
```

2) TO LIST THE NAMES OF HIS FILES ON A FAILSAFE TAPE

```
.AS MTA# FAILSA
.R FAILSA    (DIRECTORY GOES TO TTY)
*/L

.AS MTA# FAILSA
.AS DSK LST
.R FAILSA
*/P          (DIRECTORY GOES TO DSK:FAILSA.DIR)
```

3) TO RESTORE SOME OF HIS FILES TO DISK

```
.AS MTA# FAILSA
.R FAILSA
**.*         (RESTORES ALL FILES)

OR

*NAME.*       (RESTORES ALL FILES WITH NAME "NAME"
               BUT ARBITRARY EXTENSION)

OR

**.EXT        (RESTORES ALL FILES WITH EXTENSION "EXT"
               BUT ARBITRARY NAME)

OR

*FIL.1,FIL.2,FIL.3 (RESTORES SPECIFIED FILES)
```

SUBJECT: FAILCD VERSION 6 DATE: MAY 27, 1970

TO: FROM: C. MCCOMAS

THE INFORMATION IN THIS MEMORANDUM IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

FAILCD VERSION 6

(ESSENTIALLY THE SAME AS VERSION 1)

FAILCD IS A RESTORE ONLY FAILSAFE FOR RESTORING OLD FORMAT FAILSAFE TAPES (VERSION 16 & EARLIER) TO A LEVEL D SYSTEM.

ALL FAILSAFE VERSION 16 COMMANDS ARE AVAILABLE EXCEPT:

/S, /U, /M, AND /K

OPERATION IS THE SAME AS FAILSAFE VERSION 16.

THE NEW /G COMMAND HAS BEEN INCLUDED. THIS ENABLES THE USER TO RUN THE /U AND /L COMMANDS FOR A USER OTHER THAN HIMSELF. THE FORMAT IS */GMMM,NNN<C,R,>
THIS CHANGES THE SINGLE-USER PROJECT-PROGRAMMER NUMBER SWITCH FROM THAT OF THE FAILSAFE USER (THE VALUE IT HAS INITIALLY) TO MMM,NNN. THIS NEW VALUE IS RETAINED UNTIL THE NEXT /G COMMAND.

LEVEL C PROTECTION CODES ARE CHANGED TO LEVEL D PROTECTION CODES AS FOLLOWS:

LEVEL C PROTECTION	LEVEL D PROTECTION
0	0
1	5
2	7
3	7
4	1
5	5
6	2
7	7

FILES WITH PROJECT-PROGRAMMER NUMBER 1,1 ARE RESTORED (BY BOTH /R & SELECTIVE RESTORE) TO THE 1,4 AREA.

SUBJECT: FAILDC VERSION 4 DATE: MAY 27, 1970

TO: FROM: C. MCCOMAS

THE INFORMATION IN THIS MEMORANDUM IS
SUBJECT TO CHANGE WITHOUT NOTICE AND
SHOULD NOT BE CONSTRUED AS A COMMIT-
MENT BY DIGITAL EQUIPMENT CORPORATION.

(ESSENTIALLY THE SAME AS VERSION 1)
FAILDC VERSION 4

FAILDC IS A SAVE ONLY FAILSAFE FOR SAVING LEVEL D
DISK FILES ON MAGNETIC TAPE IN THE OLD FORMAT (COMPATIBLE
WITH FAILSAFE VERSION 16 & EARLIER).

ALL FAILSAFE VERSION 16 COMMANDS ARE AVAILABLE
EXCEPT:

/R AND SELECTIVE RESTORE

OPERATION IS THE SAME AS FAILSAFE VERSION 16.

THE NEW /G COMMAND HAS BEEN INCLUDED. THIS ENABLES THE USER
TO RUN THE /U AND /L COMMANDS FOR A USER OTHER THAN HIMSELF.
THE FORMAT IS */GMMM,NNN<C,R,>
THIS CHANGES THE SINGLE-USER PROJECT-PROGRAMMER NUMBER SWITCH
FROM THAT OF THE FAILSAFE USER (THE VALUE IT HAS INITIALLY)
TO MMM,NNN. THIS NEW VALUE IS RETAINED UNTIL THE NEXT /G
COMMAND.

LEVEL D PROTECTION CODES ARE CHANGED TO LEVEL C
PROTECTION CODES AS FOLLOWS:

LEVEL D PROTECTION	LEVEL C PROTECTION
0	0
1	4
2	4
3	4
4	4
5	5
6	5
7	0

ALL FILES IN THE 1,4 AREA ARE SAVED (BY BOTH /S & /U) WITH
PROJECT-PROGRAMMER NUMBER 1,1.

100-470-002-00

BOOTS

Level D Disk Bootstrap

Program Description

Revision Date: 10 June 70

Author: R. Clements

Introduction

BOOTS is a bootstrap program for use with the level D disk service of the PDP-10 timesharing monitor. BOOTS has two main functions:

1) Loading a program into core from a disk SAVE file; and
2) Dumping core out as a SAVE file (for later analysis of a crashed monitor).

BOOTS runs in EXEC mode, of course, and is loaded into the top 1K of core. Due to the lack of READ-IN mode for the various disk controllers, BOOTS is usually loaded from paper tape.

Operation:

The operation of BOOTS is as follows:

1) Load paper tape of BOOTS into paper tape reader.
2) Set READ-IN DEVICE switches to 104, press STOP, RESET, READIN.
3) When paper tape has been read, BOOTS starts and types a carriage return.
4) Type in command string, terminated by carriage return.
5) After performing the command, BOOTS either restarts itself, or transfers to the newly loaded program, as appropriate for the command.

Command string format:

The general form of a BOOTS command is:

STRUCTURE:FILE.EXT[PROJ,PROG]/SWITCH

For any fields not supplied, defaults are assumed. The SWITCH field determines what operation is to be performed, the STRUCTURE field is a file structure name within the disk file system, NOT a physical device name.

Available operations:

1) No switch - means find the specified file, clear core, read the file into core, set the PROGRAM START ADDRESS as specified by the file, and start at that address. I.e., load and run the program. The default FILE name is SYSTEM.SAV.

2) /L - means do all of the above, including setting the PROGRAM START ADDRESS, except do not start the loaded program. I.e., Load the program. Default FILE name is SYSTEM.SAV.

3) /D - means find the specified file, write out core (from location 20 through the base of BOOTS) onto the file, and write the current PROGRAM START ADDRESS onto the file. I.e., Dump core.

NOTE THAT THE SPECIFIED FILE MUST ALREADY EXIST! It is updated, not created. The file, when read, will not have correct checksums. The default file name is CRASH.SAV.

4) /n - where "n" is an octal number - means set the PROGRAM START ADDRESS to n. This command should be used prior to dumping core if the program is to be loaded and run by BOOTS in the future. (This is not necessary for dumping system crashes.)

5) /G - means Go to the current PROGRAM START ADDRESS.

Defaults:

The default STRUCTURE: is DSK: (see "search technique" below).

The default FILE is SYSTEM, except for the /D operation, where it is CRASH.

The default .EXT is .SAV (a blank extension may be specified by explicitly typing the dot with no extension following it).

The default [PROJ,PROG] is [1,4].

The default /SWITCH is "none", i.e., a load and run command.

Search Technique

BOOTS searches for a specific file structure (e.g., DSKB) by looking for all possible disks in a predetermined order. That order is: FHA0, FHA1, ..., FHA3, FHB0, ..., FHB3, DPA0, ..., DPA7, DPB0, ..., DPB7. Note that these devices are searched for a structure name written on the device, not for their physical names. BOOTS will not find device DPA0, for example.

The only exception to the rule that the STRUCTURE: requested must be a defined file structure is the special case of DSK: (or blank) as a structure name. In this case, BOOTS will search for the file on structures DSKA:, DSKB:, DSKC:, ..., DSK0:, in that order. As a consequence of this, the common case of dumping a crash on CRASH.SAV would attempt to write the file on DSKA since each structure normally has a file called CRASH.SAV. If the crash is desired on DSKB, (because DSKB is larger, for example), then the DSKB: must be explicitly typed.

Note that the LEVEL D monitor does not require disk structures to be named DSKA:, DSKB:, etc. These are only recommendations. Nothing prevents an installation from creating file structures called SAM: and IRVING:. However, BOOTS would only find such a structure if its name were explicitly typed: BOOTS will not find it in the default structure DSK:, because IRVING: is not in BOOTS's default definition of DSK:. (The LEVEL D monitor would find IRVING: in DSK:, since the monitor is smarter than BOOTS.)

Core allocation:

BOOTS may be assembled for any size of core. When running, it occupies the top 1000 octal words of the core for which it was assembled. It uses all of the next lower 1000 words as buffers, except for the region from 0 to 137 in that 1000 words, which might be a job's JOB DATA AREA under a crashed monitor, and thus might be needed for analyzing a dump.

The starting address of BOOTS is 1000 below the top of core. For example, 64K BOOTS starts at 177000.

Recapitulation:

To dump a crashed system on DSKB:CRASH.SAV[1,4], type

DSKB:/D

To run a fresh monitor from DSK:SYSTEM.SAV[1,4], type just carriage return.

LOOKFL.MEM

V001
JUNE 18 70

LOOKFL IS A PROGRAM WHICH DOES AN EXTENDED LOOKUP ON A FILE AND PRINTS OUT ALL THE ARGUMENTS. WHEN IT TYPES

FILE: ,TYPE IN A FILE NAME AND RETRIEVE YOUR LISTING FROM DEVICE LPT (FILE NAME LOOKFL.TXT).

FILE NAME HAS FORM DEV:NAME.EXT[PROJ,PROG]. UFD'S MAY BE SPECIFIED.

GRIPE.MEM

V001
JUNE 18 70

GRIPE IS A PROGRAM WHICH READS TEXT FROM ITS USER AND RECORDS IT IN
A DISK FILE, THIS ENABLES USERS TO RECORD COMMENTS AND COMPLAINTS.
TO RUN IT, RUN GRIPE, WHEN IT TYPES YES? TYPE ANY TEXT YOU WISH
TERMINATED BY AN ALTMODE. THE TEXT IS WRITTEN AS FILE
CMP###.CMP (### IS A RANDOM NUMBER), CURRENTLY IN THE 3,3 AREA, IT ALSO
HAS A HEADER INCLUDING THE DATE AND TIME AND USER WRITING THE COMMENT.

DATDMP.MEM

V001
JUNE 18 70

CODE HAS BEEN ADDED TO DATDMP TO ALLOW A SUBSET OF ITS LISTING TO BE PRINTED. IN USER MODE, IT TYPES * AND WAITS FOR THE SUBSET DESIRED TO BE TYPED IN. SUBSET MAY BE FILE.EXT[PROJ,PROG] TO LIST ENTRIES FOR FILE.EXT [PROJ,PROG]. ANY OR ALL MAY BE *, BUT IF ANY FILE IS SPECIFIED AND NO PPN IS SPECIFIED, THE USER'S PPN IS ASSUMED. IF NO FILES OR EXTS ARE SPECIFIED, ALL FILES FOR PPN'S SPECIFIED WILL BE LISTED. CARRIAGE RETURN GIVES THE WHOLE LISTING (I.E. *.*[*,*]).

FOR THOSE PPN'S SPECIFIED, PPB AND UFD BLOCKS ARE PRINTED, AND FOR FILES SPECIFIED, NMB BLOCKS AND ACCESS TABLES ARE PRINTED. THUS, *.MAC[10,*] LISTS ALL PPB'S AND UFB'S FOR ALL PROGRAMMERS IN PROJECT 10 WHO CURRENTLY HAVE PPB'S AND GIVES NMB BLOCKS AND ACCESS TABLES FOR ALL FILES WITH EXTENSION MAC FOR THOSE PROGRAMMERS.

IN EXEC MODE, SET LOCATIONS PPNTST, NAMTST, AND EXTTST FOR THE DESIRED SUBSET.

PPNTST CONTAINS PPN'S DESIRED, WITH 0 FOR *.

NAMTST CONTAINS SIXBIT NAME TO CHECK, WITH 0 FOR *.

EXTTST CONTAINS 0 FOR * OR, IF NON-ZERO, RH=SIXBIT EXT TO CHECK.

THESE LOCATIONS ARE CLEARED AFTER EACH DUMP SO THAT THEY MUST BE RESET EACH TIME.

DSKRAT.MEM

V001
JUNE 18 70

DSKRAT IS A DAMAGE ASSESSMENT PROGRAM FOR LEVEL D DISK FILE STRUCTURES. IT SCANS THE FILE STRUCTURE, USING SUPER USETI'S TO READ FILES AND REPORTS ANY INCONSISTENCIES DETECTED TO DEVICE LPT (FILE NAME RAT.LST IF LPT IS A DIRECTORY DEVICE).

TO RUN DSKRAT, LOG IN AS 1,2 (REQUIRED FOR SUPER USETI), ASSIGN A FILE STRUCTURE LOGICAL NAME STR, ASSIGN LPT IF YOU WISH IT TO BE SOME DEVICE OTHER THAN THE PRINTER, AND RUN DSKRAT. DSKRAT OPENS STR AND LPT, READS SAT BLOCKS INTO CORE AND IF EVERYTHING IS ACCEPTABLE TYPES "RUNNING" AND BEGINS TO SCAN THE FILE STRUCTURE.

FOR EACH FILE ON THE FILE STRUCTURE, DSKRAT LOOKS UP THE FILE AND REPORTS ANY FAILURES, READS AND VERIFIES THE FIRST RIB OF THE FILE, CHECKSUMS EACH GROUP AND REPORTS ERRORS, READS THE RETRIEVAL INFORMATION FROM THE RIB AND CONSTRUCTS ITS OWN SAT BLOCKS. IF THERE ARE ANY DISAGREEMENTS BETWEEN SATS READ FROM DISK AND SATS CONSTRUCTED BY DSKRAT, ERROR MESSAGES ARE OUTPUT; I.E. IF ANY CLUSTER IS IN MORE THAN ONE FILE, OR IN A FILE BUT NOT MARKED IN THE SAT, THAT FACT IS REPORTED, IDENTIFYING THE CLUSTER AND THE FILE TO WHICH IT BELONGS. ONE LINE IS PRODUCED FOR EACH ERROR, INCLUDING THE FILE NAME, CLUSTER NUMBER AND LOGICAL BLOCK NUMBER OF THE CLUSTER IN QUESTION, AND AN ERROR COMMENT.

WHEN DSKRAT HAS GONE COMPLETELY THROUGH THE FILE STRUCTURE, IT PRINTS A LIST OF CLUSTERS IN MORE THAN ONE FILE, CLUSTERS IN FILES BUT NOT MARKED IN SATS, AND CLUSTERS MARKED IN SATS BUT NOT IN ANY FILE. THEN, IF ANY CLUSTERS ARE IN MORE THAT ONE FILE, IT TYPES

"END OF PASS 1, BEGINNING PASS 2"

AND STARTS OVER. THE SECOND PASS WILL PRODUCE AN ERROR LINE FOR EVERY FILE CLAIMING CLUSTERS USED BY MORE THAN ONE FILE (OBVIOUSLY THE FIRST SUCH FILE IS NOT KNOWN IN PASS 1 UNTIL THE SECOND IS FOUND). IF YOU DO NOT WANT PASS 2, TYPE CONTROL C TWICE AND REENTER. THIS WILL CLOSE LPT AND EXIT.

IF NO CLUSTERS ARE IN MORE THAN ONE FILE, IT TYPES

"END OF PASS 1, NO NEED FOR PASS 2"

AND EXITS.

NOTE THAT SINCE DSKRAT READS IN SATS AT THE BEGINNING OF THE PROGRAM, IF OTHER USERS ARE REFERENCING THE DISKS (WRITING OR DELETING FILES OR READING FILES MARKED FOR DELETION) THE SAT BLOCKS WILL NOT BE CURRENT AND YOU WILL GET SPURIOUS ERRORS. HOWEVER, TRUE ERRORS WILL NOT BE MISSED.

FILEX

FILEX IS A GENERAL FILE TRANSFER PROGRAM, INTENDED TO CONVERT BETWEEN VARIOUS CORE IMAGE FORMATS AND TO READ AND WRITE VARIOUS DECTAPE DIRECTORY FORMATS, AS WELL AS STANDARD DISK FILES.

THE COMMANDS TO FILEX ARE SIMILAR TO A PIP COMMAND STRING. FILES ARE TRANSFERRED AS 36-BIT BINARY DATA. NO PROCESSING IS DONE ON THE DATA ITSELF, EXCEPT THAT NECESSARY TO CONVERT BETWEEN VARIOUS CORE IMAGE REPRESENTATIONS.

RAPID TAPE PROCESSING, VIA A DISK SCRATCH FILE, IS AVAILABLE.

"WILD-CARD" FILE NAMES (*) ARE PERMITTED.

DEVICE FORMATS AVAILABLE:

NON-DECTAPE DEVICES ARE READ AND WRITTEN IN BINARY. DEVICE, FILE-NAME, EXTENSION, PROJECT-PROGRAMMER NUMBER, AND PROTECTION ARE SUPPLIED IN THE USUAL WAY.

DECTAPES IN THE USUAL PDP10 DIRECTORY FORMAT MAY BE READ OR WRITTEN IN BINARY IN THE USUAL WAY, AND THEY MAY BE READ VIA A DISK SCRATCH FILE, WHICH IS MUCH FASTER FOR EITHER A TAPE WITH MANY FILES, OR A TAPE WHICH HAS BEEN WRITTEN BY TENDMP (WITH CONSECUTIVE BLOCKS ALLOCATED TO THE SAME FILE.)

SIMILARLY, DECTAPES MAY BE READ, WITH OR WITHOUT USE OF A SCRATCH FILE, AND MAY BE WRITTEN, IN EITHER THE OLD DEC PDP-6 DECTAPE FORMAT, OR THE MIT PROJECT MAC PDP6/10 DECTAPE FORMAT. FOR BOTH OF THESE FORMATS, THE MONITOR'S DECTAPE SERVICE ROUTINE CANNOT BE MADE TO RUN EFFICIENTLY, SO THE SCRATCH FILE TECHNIQUE IS ADVISED. THE /O (OLD) AND /M (MAC) SWITCHES SPECIFY THESE FORMATS. /T (TEN) RETURNS TO PDP10 FORMAT TAPES.

DATA FORMATS AVAILABLE:

UNLESS ONE OF THE FOLLOWING SPECIAL FORMATS APPLIES, ALL FILES ARE TRANSFERRED UNMODIFIED, AS 36-BIT BINARY DATA.

CORE IMAGE FILES ARE THE SPECIAL CASES HANDLED. PROCESSING IS AVAILABLE TO CONVERT FROM ANY OF THE FOLLOWING FORMATS TO ANY OTHER OF THEM. (OF COURSE, IF THE INPUT AND OUTPUT FORMATS ARE IDENTICAL, THE FILE IS SIMPLY COPIED.)

EACH OF THE FOLLOWING CORE IMAGE FORMATS IS INDICATED BY SPECIFIC EXTENSIONS, WHICH MAY BE OVERRIDDEN BY SWITCHES.

1) SAVE-FILE FORMAT: ASSUMED FOR FILES WITH EXTENSIONS .SAV, .LOW, AND .SVE. CAN BE FORCED BY THE /C SWITCH (COMPRESSED CORE IMAGE.) THE DEFAULT OUTPUT EXTENSION FOR A /C FILE IS .SAV.

2) EXPANDED CORE IMAGE FILE (AS USED BY FILDDT): ASSUMED FOR FILES WITH EXTENSION .XPN. CAN BE FORCED BY THE /E SWITCH (EXPANDED). THE DEFAULT OUTPUT EXTENSION FOR A /E FILE IS .XPN.

3) DUMP FORMAT (OLD PDP6 VERSION OF SAVE): ASSUMED FOR FILES WITH EXTENSION .DMP. CAN BE FORCED BY THE /D SWITCH.

4) SBLK FORMAT (SIMPLE BLOCK-PROJECT MAC'S EQUIVALENT OF DEC'S .SAV FORMAT): NOT ASSUMED FOR ANY EXTENSION, BUT IS FORCED BY THE /S SWITCH. THE DEFAULT OUTPUT EXTENSION FOR A /S FILE IS .BIN.

5) THE /B SWITCH CAUSES BINARY PROCESSING EVEN THOUGH A FILE FILE HAS ONE OF THESE SPECIAL EXTENSIONS.

COMMAND FORMAT:

A FILEX COMMAND IS OF THE FORM:

```
        * OUTPUT SPECIFIER = INPUT SPECIFIER(S)
                             OR
        * OUTPUT SPECIFIER = INPUT SPECIFIER(S)
```

OUTPUT SPECIFIER IS:

```
        DEV: NAME .EXT [P,PN]<PROT>/S  OR ...(S(1)S(2)S(3))
        WHERE /S INDICATES ANY SWITCH
```

INPUT SPECIFIER IS:

```
        DEV: NAME .EXT [P,PN]/S, ...  OR
        DEV: [P,PN]/S NAME .EXT, NAME2 .EXT, ...
```

IF THE [P,PN] AND/OR /S APPEAR AFTER A DEVICE, THEY APPLY TO ALL FOLLOWING FILES. IF THEY APPEAR AFTER A FILE NAME, THEY APPLY ONLY TO THAT FILENAME.

THE INPUT NAME OR EXT MAY BE *, IN WHICH CASE THE USUAL WILDCARD PROCESSING OCCURS.

THE OUTPUT NAME OR EXT MAY BE *, IN WHICH CASE THE NAME OR EXT OF THE INPUT FILE IS COPIED.

IF THE OUTPUT NAME OR EXT ARE MISSING, ALMOST THE SAME OCCURS AS FOR *, EXCEPT THAT ALL CORE IMAGE FILES WILL BE WRITTEN WITH THE DEFAULT EXTENSION AND FORMAT APPROPRIATE TO THE OUTPUT DEVICE (UNLESS OVERRIDDEN BY SWITCHES). THAT IS,:

```
*DSK: <DTA1: FOO,DMP/C  WOULD CAUSE
```

THE DMP FORMAT FILE TO BE COMPRESSED /C AND WRITTEN AS FOO,SAV.

TO CAUSE AN INPUT DECTAPE TO BE PROCESSED QUICKLY (VIA A SCRATCH FILE), USE THE /Q SWITCH (FOR QUICK).

TO CAUSE THE /Q PROCESSING AND PRESERVE THE SCRATCH FILE AFTER PROCESSING, FOR USE BY ANOTHER COMMAND, USE THE /P (PRESERVED QUICK) SWITCH.

TO REUSE A SCRATCH FILE PRESERVED BY /P IN A PREVIOUS COMMAND, USE THE /R (RE-USE) SWITCH.

TO IGNORE READ ERRORS ON THE INPUT DEVICE, USE THE /G (GO ON) SWITCH.

FILEX CHECKS THE ALWAYS-BAD-CHECKSUM BIT IN THE LEVEL D DISK FORMAT, SO /G IS NOT NEEDED FOR THOSE FILES WITH ,RPABC ON (E.G. CRASH,SAV).

TO COPY A CRASH,SAV TO AN EXPANDED FORMAT FILE FOR FILDDT TO EXAMINE, TYPE (FOR EXAMPLE):

```
DSK: SER105.SAV[10,10]/E<DSKC: CRASH,SAV[1,4]
```

WHILE LOGGED IN AS [1,2] (TO BE ABLE TO READ CRASH,SAV, WHICH IS READ-PROTECTED BY THE REFRESHER).

THE /Z SWITCH ON AN OUTPUT FILE, IF IT IS A DECTAPE, CAUSES THE APPROPRIATE FORMAT OF ZEROED DIRECTORY TO BE WRITTEN ON THE TAPE. IF THE STRING

```
↑TAPEID
```

APPEARS IN THE OUTPUT SPECIFIER, THEN TAPEID IS WRITTEN AS THE TAPE IDENTIFIER IN THE DIRECTORY. TAPEID MAY BE 6 CHARACTERS ON A PDP10 TAPE, 3 CHARACTERS ON A PROJECT MAC TAPE, AND IS NOT PRESENT ON A PDP6 TAPE.

THE /L SWITCH ON AN INPUT DECTAPE FILE CAUSES THE TAPE DIRECTORY TO BE TYPED ON THE TTY. (DO NOT PUT TTY: IN THE OUTPUT FILE SPECIFIER. THAT WOULD TRY TO WRITE FILES ON THE TTY IN BINARY.)

SUMMARY OF FILEX SWITCHES

SWITCH	MEANING	DEFAULT EXTENSION
1.	DEC TAPE FORMAT SPECIFIERS	
	M - MAC	
	O - OLD, PDP-6	
	T - TEN; NORMAL PDP-10 DIRECTORY FORMAT	
2.	FILE FORMAT SPECIFIERS	
	B - BINARY; OVERRIDES DEFAULT EXTENSION	
	C - COMPRESSED; SAVED FILE FORMAT	SAV
		LOW
		SVE
	D - DUMP; OLD PDP-6	DMP
	E - EXPANDED; FOR FILDDT	XPN
	S - SBLK; PROJECT MAC'S SAVE	BIN
3.	DEC TAPE PROCESSING SWITCHES	
	G - GO ON; IGNORE READ ERRORS	
	L - LIST; TYPE DIRECTORY ON TTY	
	P - PRESERVED; Q PLUS KEEP FILE	
	Q - QUICK; USE SCRATCH FILE FOR DIRECTORY	
	R - REUSE; SCRATCH DIRECTORY FROM P	
	Z - ZERO; DECTAPE DIRECTORY	
	+N(1),,,N(6)- SPECIFY TAPE IDENTIFIER	

QUOLST.MEM

JULY 1 1970
V001

QUOLST IS A PROGRAM WHICH TYPES ON DEVICE TTY ITS USER'S QUOTAS (RESERVED, FIRST COME FIRST SERVED, AND LOGGED OUT), AND BLOCKS FREE WITH RESPECT TO LOGGED IN QUOTA (RESERVED + FIRST COME FIRST SERVED). ONLY THOSE FILE STRUCTURES CURRENTLY IN THE FILE STRUCTURE SEARCH LIST ARE CHECKED.

PLEASE

FUNCTION
THE PLEASE COMMAND IS AVAILABLE TO PROVIDE NON-CONFLICTING COMMUNICATION BETWEEN AN OPERATOR AND SYSTEM USERS VIA TTYØ. USE OF THE PLEASE COMMAND IS PREFERRED TO "TALK OPR" BECAUSE PLEASE PREVENTS SIMULTANEOUS TRANSMISSION TO THE OPERATOR VIA TTYØ.

1. USER INTERFACE

FOR THIS DESCRIPTION LET $ REPRESENT <ALTMODE> AND LET * REPRESENT ANY ARBITRARY TEXT EXCLUDING CONTROL CHARACTERS. THE OPERATION OF PLEASE FOLLOWS:

```
USER TYPES:      PLEASE *<CR>
SYSTEM RESPONDS:          A)  OPERATOR HAS BEEN NOTIFIED
   OR            B)  OPERATOR BUSY, HANG ON PLEASE
```

IN CASE A THE USER MAY IMMEDIATELY BEGIN TWO-WAY TTY COMMUNICATION WITH THE OPERATOR. COMMUNICATION IS TERMINATED WHEN EITHER END TYPES $. IF THE OPERATOR IS BUSY THE USER MAY ELECT TO WAIT FOR A MESSAGE AT WHICH TIME HE MAY COMMUNICATE WITH THE OPERATOR. HOWEVER, HE MAY TYPE ↑C OR $ TO ABORT AND RETURN TO MONITOR MODE.

2. OPERATOR INTERFACE

TTYØ AT THE COMPUTER SITE MUST BE DEDICATED TO THE USE OF THE PLEASE FUNCTION. ON TTYØ THE OPERATOR RECEIVES JOB IDENTIFYING INFORMATION AND THE USER'S MESSAGES AND TRANSMITS HIS RESPONSES.

THE BASIC FORM OF A PLEASE CONVERSATION AS IT APPEARS TO THE OPERATOR IS SHOWN BELOW:

PREAMBLE
JOB IDENTIFYING INFORMATION
PLEASE*
CONVERSATION
TERMINATING MESSAGE

THE JOB IDENTIFYING INFORMATION IS PRESENTED AS:

JOBN [PROJECT#,PROGRAMMER#] TTYN T1 T2

THAT IS, THE INITIATING JOB NUMBER, PROJECT PROGRAMMER NUMBER AND TELETYPE. T1 IS THE TIME THE USER TYPED HIS PLEASE REQUEST, WHEREAS T2 IS THE TIME THE OPERATOR RECEIVED THE REQUEST.

WHEN EITHER PARTY TYPES <ALT-MODE> THE CONVERSATION PORTION TERMINATES, AND THE TERMINATING MESSAGE "FINISHED T3" IS TYPED TO THE OPERATOR. T3 IS NATURALLY THE TIME WHEN TRANSMISSION TERMINATES.

FILE(UMOUNT) - INTRODUCTION

THE FILE COMMAND PROVIDES REMOTE CONTROL OF DECTAPE TO DISK AND DISK TO DECTAPE TRANSFERS ON OPERATOR HANDLED DECTAPES. THIS COMMAND ALLOWS DECTAPE TO BE USED AS CONVENIENT BACKUP STORAGE FOR A SMALL DISK SYSTEM. THE COMMAND IS IMPLEMENTED IN THE UMOUNT CUSP. THE USER REQUESTS FILEING OF DATA ONTO DECTAPE, AND RECALLING DATA FROM DECTAPE, BY USING THE FILE COMMAND.

THE USER DESCRIBES THE OPERATION HE WISHES PERFORMED, AND THEN IS FREE TO DO OTHER WORK WHILE THE OPERATOR HANDLES THE TAPE OPERATIONS. HE CAN CHECK ON THE PROGRESS OF HIS REQUESTS, BUT IS NOT SPECIFICALLY INFORMED WHEN THE REQUESTS ARE COMPLETED.

THERE ARE SIX OPERATIONS WHICH CAN BE PERFORMED VIA THE FILE COMMAND: F, Z, R, L, D AND C. THESE ARE DESCRIBED BELOW. THE OTHER ARGUMENTS REQUIRED BY THE FILE COMMAND ARE TAPE ID'S AND FILENAMES. THE TAPE ID IS THE IDENTIFICATION NUMBER OF THE ROLL OF DECTAPE TO BE USED. THIS IS USUALLY A NUMBER (STARTING AT 1 FOR EACH USER), BUT MAY BE A SHARED TAPE, DESCRIBED BY A LETTER AND A NUMBER: FOR EXAMPLE, "A123" MIGHT BE A TAPE SHARED BY ALL USERS.

FILENAMES ARE THE USUAL NAME AND EXTENSION PAIR FOR DISK AND DECTAPE FILES. THE "*" CONVENTION IS ALSO A LEGAL NAME OR EXTENSION: FOR EXAMPLE, *.MAC MEANS "ALL FILE WITH EXTENSION MAC".

1. OPTIONS OF THE FILE COMMAND

1) F (FOR FILE) OPTION: THIS OPTION IS A REQUEST TO FILE INFORMATION ON DECTAPE. IT TAKES A TAPE ID AND A LIST OF FILENAMES AS ARGUMENTS.

EXAMPLE:

.FILE F,1, TEST.MAC, DATA.BIN

IS A REQUEST TO PUT THE FILES TEST.MAC AND DATA.BIN ONTO THE USER'S DECTAPE NUMBER 1. AT THE COMPLETION OF THE FILE OPERATION, AN AUTOMATIC FILE D (DIRECTORY) COMMAND WILL BE PERFORMED (SEE BELOW). ALTHOUGH A USER COULD LOGOUT AFTER A FILE REQUEST AND EXPECT THE PROCESS TO GO TO COMPLETION (IF HE SAVED THE FILES DURING THE LOGOUT DIALOGUE), THE
V023 PRACTICE IS NOT RECOMMENDED, BECAUSE THE FILES WILL NOT BE DELETED
V023 FROM THE DISK.

2) Z (ZERO) OPTION: THIS OPTION IS IDENTICAL TO THE F (FILE) OPTION, EXCEPT THAT THE DIRECTORY OF THE DECTAPE WILL BE CLEARED (ZEROED) BEFORE THE FILES ARE COPIED. AGAIN, A FILE D COMMAND WILL BE PERFORMED AFTER THE FILES ARE COPIED.

3) R (RECALL) OPTION: THIS OPTION IS A REQUEST TO RECALL INFORMATION FROM DECTAPE TO THE DISK. THE ARGUMENTS ARE THE SAME AS FOR THE F AND Z OPTIONS. AGAIN, A FILE D COMMAND WILL BE PERFORMED AFTER THE FILES ARE TRANSFERRED.

EXAMPLE:

.FILE F,1, *.*

IS A REQUEST TO RESTORE ALL FILES FROM THE USER'S DECTAPE NUMBER 1.

(1 CON'T)

4) L (LIST DIRECTORY) OPTION: THIS OPTION IS A REQUEST TO READ THE DIRECTORY OF A DECTAPE. THE D OPTION REQUIRES ONLY ONE ARGUMENT, THE DECTAPE TAPE ID.

EXAMPLE:

.FILE D,2

IS A REQUEST TO READ THE DIRECTORY OF THE USER'S TAPE NUMBER 2. THE DIRECTORY WILL BE PLACED IN THE USER'S DISK AREA AS AN ASCII FILE WITH THE NAME TAPEID.DIR, WHERE TAPEID IS THE NAME OF THE USER'S DECTAPE. THUS, IN THE ABOVE EXAMPLE, THE USER MAY READ THE DIRECTORY FOR TAPE 2 BY THE MONITOR COMM↑AAND:

.TYPE 2.DIR

A DIRECTORY OPTION IS PERFORMED AT THE COMPLETION OF EACH F, Z, OR R OPTION.

5) D (DELETE) DELETES FILES MENTIONED FROM DECTAPE.

6) C (CHECK) OPTION: THIS OPTION CAUSES THE QUEUE OF FILE COMMANDS TO BE READ TO DETERMINE WHETHER ANY OF THE USER'S REQUESTS ARE STILL PENDING. IF NONE ARE PENDING, THE MESSAGE "NONE PENDING" IS TYPED. IF THERE ARE SOME REQUESTS PENDING, THEY WILL BE LISTED. THIS OPTION DOES NOT USE ANY TAPE OR FILE ARGUMENTS. THE CURRENT POSITION IN THE QUEUE OF EACH REQUEST IS TYPED AT THE BEGINNING OF THE LINE DESCRIBING THAT REQUEST.

EXAMPLE:

.FILE C

2. DIALOGUE FORM

IF THE FILE COMMAND IS EXECUTED WITHOUT AN ARGUMENT, A BRIEF DIALOGUE WILL BE PERFORMED. THIS MAY BE EASIER FOR A BEGINNER TO FOLLOW. AN EXAMPLE FOLLOWS:

```
.FILE
C, D, F, R OR Z (? FOR HELP)
*F
TAPE ID:  1
FILES:  A, B, C
REQUEST STORED
```

1. OPERATOR REQUIRMENTS

1.1. HARDWARE ENVIRONMENTS

OPERATION OF THE FILE-RECALL SYSTEM REQUIRES
A) A DEDICATED DECTAPE DRIVE
B) A CONTROLLING TTY
C) DISK FILE STORAGE OF SOME TYPE

1.1. STARTUP

THE OPERATOR LOGS IN UNDER [1,2] ON AN AVAILABLE LOCAL TTY AND ASSIGNS ANY PARTICULAR DECTAPE DRIVE SAY 2, TO BE USED BY THE FILE-RECALL SYSTEM. HE TYPES R OMOUNT AND STARTUP IS COMPLETE.

2. OPERATION

USER REQUESTS ARE STACKED AS ASCII COMMAND FILES UNDER 3,3,UFD WITH ORDER OF RECEIPT (AND HENCE EXECUTION) INDICATED BY THE COMMAND FILE NAMES: E.G., FIL1.CMD,FIL2.CMD,... THE PROGRAM OPFILE READS THESE REQUESTS IN ORDER, COMMUNICATING WITH THE OPERATOR ON THE CONTROLLING TTY.

A TYPICAL MESSAGE FROM OPFILE IS OF THE FORM:

```
*-*-*-*
PLEASE MOUNT TAPE # FOR USER #,# ON DRIVE:
```

(2. CONT'D)

THE OPERATOR RETRIEVES THE SPECIFIED DECTAPE
MOUNTS IT UPON THE PREVIOUSLY
ASSIGNED DRIVE AND COMMANDS OPFILE TO PROCEDE BY TYPING THE
DRIVE NUMBER TERMINATED BY A CARRIAGE RETURN (IN THIS CASE, 2).
OPFILE REPLIES WITH THE CURRENT ASCII COMMAND FILE AND SIGNIFIES
COMPLETION OF THE FILE-RECALL OPERATION WITH AN APPROPRIATE
MESSAGE.

3. UNUSUAL ACTIONS

OMOUNT INFORMS THE OPERATOR OF ANY ERRORS OCCURRING WHILE
TRANSFERRING FILES. THE OPERATOR MAY REPEAT THE CURRENT FILE
OPERATION BY RESTARTING OMOUNT OR MAY DISCARD THE CURRENT REQUEST
BY TYPING:

```
↑C
.REENTER
```

AT THE TIME WHEN OMOUNT IS WAITING FOR A DRIVE #,
THE OPERATION WOULD APPEAR AS FOLLOWS:
PLEASE MOUNT TAPE # FOR USER #,# ON DRIVE: ↑C

```
REENTER
```

WHEN ↑C, REENTER IS RECEIVED IN THIS FASHION OPFILE RESPONDS
BY PRINTING THE DISCARDED COMMAND LINE AND AN APPROPRIATE
IDENTIFYING MESSAGE.

1.0 DC10E HANDLER

1.1 BACKGROUND

THIS SPECIFICATION DESCRIBES THE INITIAL IMPLEMENTATION OF THE DC10E DATASET-CONTROL SERVICE ROUTINE, AND ITS REQUIRED HARDWARE ENVIRONMENT. THIS ROUTINE IS INTENDED AS THE NECESSARY LEVEL OF SOFTWARE SUPPORT FOR EARLY DC10E'S. FEATURES DESCRIBED HERE WILL BE EXPANDED UPON IN LATER WORK IN THE LOCAL COMMUNICATIONS SOFTWARE PROJECT.

1.2 HARDWARE ENVIRONMENT

1.2.1 THIS ROUTINE WILL SUPPORT UP TO FOUR DC10E DATASET HANDLERS ON A DC10 DATALINE SCANNER. EACH LINE TO BE CONNECTED TO THE DC10E IS ASSUMED DO BE A BELL-SYSTEM

(1.2.1 CONT'D)

103A DATASET, OR ITS EQUIVALENT.

1.2.2 THE AUTOMATIC-DIALLER FEATURES OF THE DC10E ARE NOT SUPPORTED.

1.2.3 LINE-NUMBERS OF THE EQUIPMENT MUST BE ASSIGNED AS FOLLOWS:

1.2.3.1 THE FIRST LINE-GROUP(S) ON THE DC10 (LINES 0-7, 10,17, ETC.) ARE ASSIGNED TO DC10B 8-LINE GROUPS.

1.2.3.2 DC10E EXPANDED DATASET CONTROLS ARE ASSIGNED TO THE NEXT LINE GROUPS IMMEDIATELY ABOVE THE DC10B'S.

1.2.3.3 ANY SPARE LINE GROUPS WILL BE THE REMAINING HIGHEST-NUMBERED LINES (X0-77).

1.2.3.4 THE ASSOCIATION BETWEEN THE DC10B LINE NUMBER BY WHICH THE DATA IS PASSED, AND TAG DC10E LINE NUMBER BY WHICH THE SUPERVISION OF THE DATASET IS MAINTAINED, IS ARBITRARY, THIS ASSOCIATION IS SPECIFIED BY THE CUSTOMER TO THE MONITOR VIA THE MONGEN PROGRAM, AND IS RETAINED BY THE MONITOR IN A TABLE IN SUB-ROUTINE COMMON, THE PHYSICAL NAME "TTYN" OF THE LINE IS DETERMINED BY THE DC10B LINE NUMBER.

1.2.4 EACH LINE OF THE DC10E AND ASSOCIATED DC10B SHALL BE WIRED AS FOLLOWS: (REFERENCE THE DC10 TECHNICAL MANUAL DEC-10-I8AA-D AS REVISED, AND BELL SYSTEM DATA COMMUNICATIONS TECHNICAL REFERENCE MANUAL, DATA SET 103A INTERFACE SPECIFICATION).

1.2.4.1 CIRCUIT BA, TRANSMITTED DATA, SHALL BE CONNECTED TO THE DC10B CIRCUIT LN PNTR EIA.

1.2.4.2 CIRCUIT BB, RECEIVED DATA, SHALL BE CONNECTED TO THE DC10B CIRCUIT LN KYBD EIA.

1.2.4.3 CIRCUITS AA AND AB, SIGNAL GROUND AND PROTECTIVE GROUND SHALL BE CONNECTED TO THE DC10E CIRCUIT LN AB (GND).

1.2.4.4 CIRCUIT CD, DATA TERMINAL READY SHALL BE CONNECTED TO THE DC10E CIRCUIT LN DATA DATA TRM RDY EIA.

1.2.4.5 CIRCUIT CB, CLEAR TO SEND, SHALL BE CONNECTED TO THE DC10E CIRCUIT LN CLR TO SND EIA.

1.2.4.6 CIRCUIT CE, RINGING INDICATOR SHALL BE CONNECTED TO THE DC10E CIRCUIT LN RSTRN DETCTD EIA.

1.2.4.7 THIS INFORMATION IS DESCRIBED IN FIGURE 2-8-G OF THE DC10 MANUAL.

1.2.5 THE DATASET AS SUPPLIED BY THE TELEPHONE COMPANY MUST HAVE THE FOLLOWING OPTIONS, FOR SATISFACTORY OPERATION:

2.5.1 AUTOMATIC ANSWER MUST BE PRESENT.

2.5.2 INITIATE DISCONNECT MUST BE PRESENT.

2.5.3 RESPOND TO DISCONNECT MUST BE PRESENT.

2.5.3 IF THE DATASET IS A 103E OR 103G, THEN "CB-CF INDICATION COMMON" MUST BE PRESENT.

1.2.6 THE "DTR OS" SWITCH ON THE CONTROL PANEL OF THE DC10 MUST BE IN THE ON (UP) POSITION AT ALL TIMES.

1.3 FUNCTIONS PROVIDED BY THE SOFTWARE:

1.3.1 INITIALIZATION:

WHEN THE MONITOR IS STARTED (OR RESTARTED), EACH DC10E LINE WILL BE INTERROGATED FOR THE PRESENCE OF A CARRIER. IF NO CARRIER IS PRESENT, THE DATASET WILL BE HUNG UP. IF A CARRIER AS PRESENT, THE DATASET WILL BE LEFT ENABLED. IN EITHER CASE, THE STATE OF THE CARRIER FLAG IS REMEMBERED IN CORE.

1.3.2 RESPONSE TO RINGING:

WHEN A LINE RINGS, LOCATION "STATES" WILL BE CHECKED FOR THE ABSENCE OF BIT 34. UNLESS INHIBITED BY BIT 34 OF STATES, DATA TERMINAL READY WILL BE SET FOR THIS LINE, CAUSING THE CALL TO BE ANSWERED BY THE DATASET. A FIFTEEN-SECOND COUNT WILL BE STARTED, SO THAT THE CALL MAY BE ABANDONED IF NO CARRIER IS RECEIVED AFTER THAT TIME. (THE RIGHT HALF OF LOCATION STATES MAY BE SET TO AN OCTAL NUMBER "N" BY THE MONITOR COMMAND "SCHEDULE N" TYPED AT THE OPERATOR'S CONSOLE.)

1.3.3 RESPONSE TO CARRIER-ON:

WHEN A CARRIER AS RECEIVED ON A LINE THE RECEIPT OF A "CONTROL-C" CHARACTER WILL BE SIMULATED. THERE WILL NOT BE AN AUTOMATIC LOGIN OPERATION IN THE INITIAL SYSTEM.

1.3.4 RESPONSE TO CARRIER-OFF:

WHEN A CARRIER-OFF SIGNAL IS RECEIVED, ANOTHER "CONTROL-C" WILL BE SIMULATED. THERE WILL NOT BE AN AUTOMATIC DETACH OR LOGOUT OPERATION IN THE INITIAL SYSTEM.

1.4 IMPLEMENTATION:

THE DATA STRUCTURES ASSOCIATED WITH THE DATA-SETS ARE DESCRIBED IN SUBROUTINES COMMON, DLSINT AND MONGEN. THERE ARE NO CONDITIONAL ASSEMBLIES ASSOCIATED WITH THE DC10E CODE, BUT TABLE SPACE WILL BE SAVED IF NO DC10E IS PRESENT.

2.2 UUO'S

2.2.1 TEMPORARY FILE STORAGE FOR JOB UUO, TMPCOR (44)

THE "TMPCOR" UUO IS USED TO ENABLE A JOB TO LEAVE SEVERAL SHORT FILES IN CORE FROM THE RUNNING OF ONE USER PROGRAM OR CUSP TO THE NEXT. THESE FILES MAY BE REFERRED TO BY A THREE CHARACTER FILE NAME, AND ARE UNIQUE TO EACH JOB, I.E. A JOB CAN ONLY REFERENCE ITS OWN FILES. ALL FILES ARE ALWAYS DELETED WHEN A JOB IS KILLED.

EACH FILE APPEARS TO THE USER AS ONE DUMP MODE BUFFER. THE ACTUAL SIZE OF A TEMPORARY FILE, THE NUMBER OF TEMPORARY FILES A USER CAN HAVE, AND THE TOTAL CORE SPACE A USER CAN TIE UP ARE PARAMETERS DETERMINED AT MONGEN TIME. ALL TEMPARARY FILES RESIDE IN A FIXED AREA IN THE MONITOR, BUT THE SPACE IS DYNAMICALLY ALLOCATED AMOUNG DIFFERENT JOBS AND THE SEVERAL DIFFERENT FILES OF ANY GIVEN JOB.

THE PRIMARY PURPOSE OF THE TEMPORARY STORAGE SYSTEM IS FOR SHORT CONTROL FILES, E.G. CCL FILES, TO LIVE IN CORE, THEREBY SPEEDING UP RESPONSE TIMES AND REDUCING DISK OPERATIONS. ACCORDINGLY, SHOULD A PROGRAM ATTEMPT TO WRITE A FILE WHEN THERE IS INSUFFICIENT SPACE, EITHER IN THE ENTIRE BUFFER AREA OR BECAUSE THE USER HAS EXCEEDED HIS QUOTA, THE UUO GIVES AN ERROR RETURN. THE CUSP CAN THEN WRITE THE DATA AS A SHORT DISK FILE. SIMILARLY, SHOULD A PROGRAM FAIL TO FIND A FILE UPON READING IT, IT WILL GET AN ERROR RETURN AND CAN THEN LOOKUP A SHORT DISK FILE.

IT IS VERY IMPORTANT TO REALIZE THE TEMPORARY NATURE OF THESE FILES. FOR EXAMPLE, UPON WRITING, THE OLD FILE IS DELETED BEFORE CHECKING FOR SPACE FOR A NEW VERSION. THE OLD FILE COULD BE LOST WITHOUT A NEW ONE REPLACING IT. ALSO, THERE CAN BE NO GUARANTEE THAT FILES WILL FIT IN CORE.

THE TMPCOR UUO IS NOT INTENDED TO REPLACE A FUTURE, MORE GENERAL, DEVICE INDEPENDENT SERVICE ROUTINE FOR "CORE". HOWEVER, THE SPACE TAKEN UP BY DEVICE DATA BLOCKS, ETC., IN THAT MORE GENERAL ROUTINE WOULD REPRESENT UNNECESSARY OVERHEAD FOR EXTREMELY SHORT DATA, SUCH AS CCL COMMAND FILES.

FORMAT OF TEMPORARY FILE STORAGE UUO.

```
CALL AC, [SIXBIT /TMPCOR/]      ;CALLI INDEX=44
;ERROR RETURN
;NORMAL RETURN
```

C(AC) MUST ALWAYS BE SET UP BY THE USER PROGRAM PRIOR TO EXECUTING THE UUO. IT IS CHANGED BY THE UUO AND RETURNS A VALUE THAT DEPENDS ON THE PARTICULAR FUNCTION PERFORMED.

```
        C(AC) = XWD CODE,BLOCK

BLOCK:  XWD     NAME,0          ;NAME IS FILE NAME
        IOWD    BUFLEN,BUFFER   ;USER BUFFER AREA (ZERO  FOR NO BUFFER)
```

CODE=0 -- GET FREE SPACE

THE IS THE ONLY FORM OF THE TEMP UUO THAT DOES NOT USE A TWO WORD PARAMETER BLOCK. C(AC) WOULD ORDINARLY BE SET TO ZERO FOR THE GET FREE SPACE UUO. THE USER PROGRAM ALWAYS GETS A NORMAL RETURN (UNLESS THE SYSTEM DOES NOT HAVE THE TEMP UUO). C(AC) IS SET TO THE NUMBER OF WORDS OF FREE SPACE AVAILABLE TO THE USER.

CODE=1 -- READ FILE

IF THE SPECIFIED FILE NAME IS NOT FOUND, C(AC) IS SET TO THE NUMBER OF FREE WORDS OF SPACE AVIALABLE FOR TEMP FILES, AND THE ERROR RETURN IS TAKEN.

IF THE FILE IS FOUND, C(AC) IS SET TO THE LENGTH OF THE FILE IN WORDS, AND AS MUCH OF THE FILE AS WILL FIT IS COPIED INTO THE USERS BUFFER. THE USER CAN CHECK FOR TRUNCATION BY COMPARING C(AC) WITH BUFLEN UPON SUCCESSFUL RETURN FROM THE TEMP UUO.

CODE=2 -- READ AND DELETE FILE

THIS IS THE SAME AS CODE=1, EXCEPT THAT IF A FILE WAS FOUND IT IS ALSO DELETED AND ITS SPACE RECLAIMED.

CODE=3 -- WRITE FILE

IF THERE IS ALREADY A FILE OF THE SPECIFIED NAME, IT IS DELETED AND ITS SPACE IS RECLAIMED.

THE REQUESTED SIZE OF THE FILE IS SPECIFIED BY BUFLEN. IF THERE IS NOT ENOUGH SPACE TO WRITE THE ENTIRE FILE, NOTHING IS WRITTEN, C(AC) IS SET TO THE NUMBER OF FREE WORDS OF SPACE AVAILABLE TO THE USER, AND THE ERROR RETURN IS TAKEN.

IF THERE IS ENOUGH SPACE, THE FILE IS WRITTEN. C(AC) IS SET TO THE AMOUNT OF SPACE LEFT AFTER THE FILE HAS BEEN WRITTEN AND THE NORMAL RETURN IS TAKEN. FILES ARE ALWAYS FILLED UP WITH ZEROS TO THE NEXT EVEN MULTIPLE OF THE BLOCK LENGTH (TMPBL).
THIS EVEN LENGTH IS READ BACK IN.

CODE=4 -- READ DIRECTORY

THE ERROR RETURN IS NEVER TAKEN.

C(AC) IS SET TO THE NUMBER OF DIFFERENT FILES IN THE JOB'S TEMPORARY FILE AREA. IN ADDITION, AN ENTRY IS MADE FOR EACH FILE IN THE USER BUFFER AREA UNTIL THERE IS NO MORE SPACE OR ALL FILES HAVE BEEN LISTED. THE USER PROGRAM CAN CHECK FOR TRUNCATION BY COMPARING C(AC) UPON RETURN WITH BUFLEN.

DIRECTORY ENTRY FORMAT
XWD NAME,SIZE ;NAME=FILE NAME, SIZE =FILE LENGTH IN WORDS.

CODE=5 -- READ AND CLEAR DIRECTORY

THIS IS THE SAME AS CODE=4 EXCEPT THAT ANY FILES IN THE JOB'S TEMPORARY STORAGE AREA ARE ALSO DELETED AND THEIR SPACE RECLAIMED.

THIS UUO IS EXECUTED BY THE LOGOUT CUSP.

IMPLEMENTATION

MASTER DIRECTORY

THIS IS A TABLE JOBN+1 ENTRIES LONG.

```
JBTTMP: XWD FREE,IDLE
JBTTM1: XWD SPACE,LINK
          .
          .
          .
```

```
MREE = NO. OF FREE BLOCKS IN MONITOR BUFFER AREA
IDLE  =  LINK TO FIRST FREE BLOCK OR Ø IF NO FREE BLOCKS
SPACE  =  NO OF FREE BLOCKS REMAINING IN JOBS QUOTA
LINK  =  LINK TO FIRST BLOCK OF FIRST FILE OF JOB, Ø IF NONE.
```

IDLE BLOCK FORMAT

```
XWD Ø,LINK
REPEAT TMPBL, <Ø
                >
```

LINK = LINK TO NEXT BLOCK ON IDLE CHAIN, Ø IF NO MORE.

USER BLOCK FORMAT

```
XWD NAME,LINK
BLOCK TMPBL      ;USER DATA OR ZERO FILL.
```

NAME = USER FILE NAME.
LINK = LINK TO NEXT BLOCK IN THIS FILE OR NEXT FILE OF THIS USER

IF A FILE IS SEVERAL BLOCKS LONG, EACH BLOCK HAS THE FILE NAME. A LINK OF Ø INDICATES NO MORE DATA IN THE FILE, AND NO MORE FILES FOR THIS USER.

THEREFORE, A FILE ENDS WHEN ITS LAST BLOCK HAS A ZERO LINK, OR WHEN IT LINKS TO A FILE OF DIFFERENT NAME.

MONITOR BUFFER AND PARAMETERS

```
TMPBUF: BLOCK TMPBKS=<TMPBL+1>  ;BUFFER AREA FOR ALL FILES.
```

TMPBKS IS THE NUMBER OF BLOCKS THE STORAGE AREA IS COMPUTED.
IT IS COMPUTED BY MACRO DURING THE ASSEMBLY OF COMMON.
TMPBL IS A PARAMETER IN S.MAC.

```
   000000000          NNN          NNN          CCCCCCCCCCCC       EEEEEEEEEEEEEEEE
   000000000          NNN          NNN          CCCCCCCCCCCC       EEEEEEEEEEEEEEEE
   000000000          NNN          NNN          CCCCCCCCCCCC       EEEEEEEEEEEEEEEE
000          000      NNN          NNN       CCC                   EEE
000          000      NNN          NNN       CCC                   EEE
000          000      NNN          NNN       CCC                   EEE
000          000      NNNNNNN      NNN       CCC                   EEE
000          000      NNNNNNN      NNN       CCC                   EEE
000          000      NNNNNNN      NNN       CCC                   EEE
000          000      NNN   NNN    NNN       CCC                   EEEEEEEEEEEE
000          000      NNN   NNN    NNN       CCC                   EEEEEEEEEEEE
000          000      NNN   NNN    NNN       CCC                   EEEEEEEEEEEE
000          000      NNN      NNNNNN       CCC                   EEE
000          000      NNN      NNNNNN       CCC                   EEE
000          000      NNN      NNNNNN       CCC                   EEE
000          000      NNN          NNN       CCC                   EEE
000          000      NNN          NNN       CCC                   EEE
000          000      NNN          NNN       CCC                   EEE
   000000000          NNN          NNN          CCCCCCCCCCCC       EEEEEEEEEEEEEEEE
   000000000          NNN          NNN          CCCCCCCCCCCC       EEEEEEEEEEEEEEEE
   000000000          NNN          NNN          CCCCCCCCCCCC       EEEEEEEEEEEEEEEE


FFFFFFFFFFFFFFF       LLL                          000000000
FFFFFFFFFFFFFFF       LLL                          000000000
FFFFFFFFFFFFFFF       LLL                          000000000
FFF                   LLL                       000          000
FFF                   LLL                       000          000
FFF                   LLL                       000          000
FFF                   LLL                       000          000
FFF                   LLL                       000          000
FFF                   LLL                       000          000
FFFFFFFFFFFF          LLL                       000          000
FFFFFFFFFFFF          LLL                       000          000
FFFFFFFFFFFF          LLL                       000          000
FFF                   LLL                       000          000
FFF                   LLL                       000          000
FFF                   LLL                       000          000
FFF                   LLL                       000          000
FFF                   LLL                       000          000
FFF                   LLL                       000          000
FFF                   LLLLLLLLLLLLLLL              000000000
FFF                   LLLLLLLLLLLLLLL              000000000
FFF                   LLLLLLLLLLLLLLL              000000000
```

ONCE .FLO CREATED: 17-DEC-1969 00:00 PRINTED: 02-JUL-1970 16:44

```
        FILOPT,FILCHG  DETAILED FLOW FOR OPTIONAL ONCE ONLY CODE
        FILCHG CALLED AS SUBROUTINE IF MANDATORY ONCE ONLY DISCOVERS A NEED TO REFRESH
        * INDICATES SUBROUTINE IN FILSER (NEEDED AFTER ONCE ONLY) RATHER THAN ONCE

FILOPT: SET AND CLEAR ALL SOFTWARE FLAGS AND DISH QUERIES [CALL DSKINI*]
        READ ALL HOME BLOCKS FROM ALL UNITS IN SYSTEM AND SETUP STR BLOCKS(UPPER CASE)[CALL REDHOM]
        IGNORE ERROR RETURN
        TYPE "DISK FILE STRUCTURES:" CR-LF

FILCHG: TYPE ALL STR NAMES AND PHYSICAL UNITS WITHIN STR [CALL TYPSYS]
        TYPE ALL UNITS NOT IN STRS[CALL TYPUNS]
        ASK FOR STR NAME FOR PARAMETERS TO BE TYPED [CALL ASKPAR] (LOOP UNTIL CR)
        ASK "DO YOU WANT TO CHANGE ANYTHING (CR IF NO)?" [CALL ASKYCR]
        IF CR RETURN, TO RFRLSH

CHGLUP: ASK IF WANT TO DISSOLVE ANY STRS [CALL ASKDIS](LOOP UNTIL CR)
        ASK IF WANT TO DEFINE ANY NEW STRS [CALL ASKDEF](LOOP UNTIL CR)
        ASK FOR STR NAME FOR PARAMETERS TO BE CHANGED [CALL CHGPAR](LOOP UNTIL CR)
        TYPE ALL STR NAMES FOR PARAMETERS TO BE CHANGED [CALL CHGPAR] (LOOP UNTIL CR)
        TYPE ALL STR NAMES AND PHYSICAL UNITS WITHIN STR [CALL TYPSYS]
        TYPE ALL UNITS NOT IN STRS [CALL TYRNS]
        TYPE "BEFORE HOME BLOCKS ARE WRITTEN."
        ASK FOR STR NAMES FOR PARAMETERS TO BE TYPED [CALL ASKPAR] (LOOP UNTIL CR)
        ASK "DO YOU WANT TO CHANGE ANYTHING (CR IF NO)? [CALL ASKYCR]
        IF NOT CR, CHGLUP
        DO SCN SYS, FOR EVERY UNIT IN SYSTEM
          IF HOME BLOCK NEEDS CHANGING [UNPCHG]
            READ HOME BLOCK
            UPDATE FROM STR AND UNI DATA BLOCKS [CALL HOMUPD]
            REWRITE HOME BLOCK IN BOTH PLACES
            READ HOME BLOCK AND CHECK NAME [HOMNAM] AND CODE [HOMCOD] IN BOTH PLACES [CALL REDRUN]
            IF ERROR (BOTH BAD), HALT
          END

SCNSYS: CONTINUE
        TYPE "HOME BLOCKS WRITTEN" CR-LF
        TO FILOPT

RFRESH: TYPE "TYPE STR NAME TO BE REFRESHED (CR IF NONE)" [CALL ASKSTR]
        IF CR, TO REFEND
        REFRESH SPECIFIED FILE STRUCTURE [CALL REFSTR]
        TO RFRESH

 REFEND: RETURN

DSKINI* FLOW FOR ROUTINE TO SET AND CLEAR FLAGS AND QUEQUES
        PART OF FILSER SINCE CALLED ON ALL RESTARTS.

CLEAR ALL UNPCHG GLAGS
```

```
REDHOM: DETAILED FLOW FOR ONCE ONLY READ HOME BLOCKS AND SETUP STR DATA BLOCK
        ROUTINE ALSO CALLED FROM MANDATORY ONCE ONLY CODE
        CHECK ALL CONTROLLERS FOR BEING UP
        SETUP STR DATA BLOCKS
        LINK UNIT DATA BLOCKS TO THEM
        INITIALIZE LOCATIONS IN UNI, ICON, STR DATE BLOCK FROM HOME BLOCK

        ERROR RETURN AFTER TYPING ALL UNITS, IF ANY HAVE PROBLEMS
        INITIALIZE ALL DISK FLAGS, QUEUES, ETC [CALL DSKINI*] [DINITF 0]
        SET ONCE DISK INITIALIZATION IN PROGRESS, SO ERRORS WILL BE HANDLED DIFF.
        DO SCNUNI, FOR EACH UNIT DATA BLOCK IN SYSTEM [SYSUNI=UNISYS]
        IF CONTROLLER HAS ALREADY BEEN MARKED AS DOWN [KOPDWN=1], TO FLGDNN
TRYKON: CHECK TO SEE IF CONTROLLER IS OFF-LINE [CALL KONUPA(K)]
        IF OFF-LINE RETURN
          TYPE "CONTROLLER XXX IS OFF-LINE,
               "DO YOU WANT IT TO BE (1) ON-LINE, OR (2) DOWN? (TYR#)
          IF NOT "2", TO TRYKON
          FLAG CONTROLLER AS DOWN [KOPDWN]
          TO FLGDWN
        END
        DISPATCH TO ONCE ONLY ROUTINE TO SEE IF UNIT EXISTS AND IS ON-LINE
        IF OFF-LINE RETURN FOR THIS UNIT
          TYPE "UNIT XXX IS OFF-LINE"
               "DO YOU WANT IT TO BE (1) ON-LINE, (2) OFF-LINE, OR (3) DOWN (TYA #)
          IF "2", TO FLGOFL
          IF NOT "3", TO TRYKON (THE DUMMY MAY HAVE TURNED CONTROLLER OFF)
FLGDWN:   FLAG UNIT AS DOWN [UNYDST=UNVDWN]
FLGOFL:   FLAG  UNIT AS OFF-LINE [UNPOFL]
          TO SCNUNI
        END
        DISPATCH TO ONCE ONLY ROUTINE INDEXED BY CONTROLLER TYPE TO CHECK WRITE LOCK UNIT
        IF WRITE LOCK RETURN FOR THIS UNIT
          TYPE "UNIT XXX IS WRITE-LOCKED,
               "DO YOU WANT IT TO BE (1)WRITEABLE (2)WRITE LOCKED? (TYPE #)
          IF NOT "2", TO TRYKON (THE DUMMY MAY HAVE TURNED CONTROLLER OFF)
          FLAG UNIT AS WRITE LOCKED [UNPHWP]
        END
          READ BOTH HOME BLOCKS INTO UPPER CORE AND PRINT ANY ERRORS [CALL REDRUN]
          IF ERROR RETURN (SOFTWARE OR HARDWARE ERRORS ON BOTH BLOCKS)
            MARK THIS UNIT AS NOT IN AN STR [UNISTR=0]
            CLEAR LOGICAL UNIT WITHIN STR NAME [UNILOG]
            CLEAR HOMEBLOCK IDENTIFICATION [UNIHID]
            TO SCNUNE
          END
        MOVE PARAMETERS FROM HOMEBLOCK TO UNIT DATA BLOCK [(ALL MOVUNI]
        DETERMINE VIA IO, SIEE OF UNIT AND STORE IN UNIT DATA BLOCK [UNIBPU]
        STORE TYPE OF UNIT (RP01 VS RP02 OR RD10 VS RM10)[UNYUTP]
          SCAN STR DATA BLOCKS (UPPER CORE) FOR MATCH WITH STR NAME IN HOME BLOCK [HOMSNM] [CALL FNDSTR]
          IF FOUND RETURN, TO OLD STR

          CREATE ANOTHER STR DATA BLOCK IN UPPER CORE [CALL GETSTR]
          APPEND NEW STR DATA BLOCK TO END OF STR LIST [STRSYS]
          STORE DESTINATION (LOWER CORE) ADR. OF STR DATA BLOCK IN SYSTEM TABLE [TABSTR]
          STORE NEXT FREE FILE STRUCTURE NUMBER IN THIS FILESTR DATA BLOCK (STRFSN]
          MOVE FOLLOWING PARAMETERS FROM HOME BLOCK TO STR DATA BLOCK [CALL MOVSTR]
```

```
OLDSTR: IF THIS IS LAST UNIT IN STR [HOMNXT=0]
          IF NO. OF UNITS IN STR HAS ALREADY BEEN STORED INTO [STRUNM]
                TYPE "STR XXX HAS MORE THAN ONE LAST UNIT"
                SET ERROR RETURN FLAG
                SET STR NEEDS REFRESHING [RH(STRREF)]
                TO SCNUNI
        END
          STORE NO. OF UNITS IN THIS STR [STRNUM=UNILUN+1]
        END
        DO SCNUNS, FOR EACH UNIT DATA BLOCK IN THIS STR SO FAR [STRUNI=UNISTR]
          IF LOG UNIT NO. OF NEXT UNIT IN THIS STR [UNILUN] IS GREATER, TO INSERT
SCNUNS: CONTINUE
        APPEND THIS UNIT DATA BLOCK TO END OF UNIT LIST FOR STR[STRUNI]
        TO SETUNS

INSERT: INSERT UNIT DATA BLOCK JUST READ IN FRONT OF FIRST UNIT LARGER
SETUNS: SET UNIT DATA BLOCK TO POINT UPWARD TO ITS STR DATA BLOCK [RH(UNISTR)]
SCNUNI: CONTINUE
        DO CHKSTR, FOR EACH STR IN SYSTEM [SYSSTR=STRSYS]
          IF LAST UNIT IN STR WASN'T READ [STRNUM=0]
                TYPE "LAST UNIT WASN'T FOUND IN STR XXX"
                SET ERROR RETURN FLAG
          END
                SET STR NEEDS REFRESHING [RH(STRREF)
          END
          SET NO. OF BLOCKS IN STR TO =NO. OF BLOCKS FOR SWAPPING[STRHGH=STRK4S+BLKBPK]
          DO CHKCUN=S FOR EACH UNIT DATA BLOCK IN THIS STR
            INCREASE NO. OF BLOCK IS STR BY NUMBER IN THIS  UNIT[STRGH=STRHGH + UNIBPU]
            IF NEXT UNIT IS MORE THAN ONE LOGICAL UNIT HIGHER
                TYPE "LOGICAL UNIT N MISSING FROM STR COR"
                SET ERROR RETURN FLAG
                SET STR NEEDS REFRESHING[RH(STRREF)
          END
                IF NEXT UNIT IS SAME AS PREVIOUS UNIT
                TYPE "TEND LOGICAL UNIT U FOUND IN STR AAA"
                SET ERROR RETURN FLAG
                SET STR NEED REGRESHING [RH(STRREF)
          END
CHKUNI: CONTINUE
CHKSTR: CONTINUE
        IF ERROR WHILE READING A UNIT, ERROR RETURN
        OK RETURN

        FNDSTR=ONCE ONLY ROUTINE TO SEARCH STR DATA BLOCKS (UPPER CORE)
        ARE=STR NAME
        VAL=ADR. OF STR DATA BLOCK IN UPPER,
        VAL=ADR OF PREDESSOR (MAYBE=SYSSTR IN LOWER CORE)
        VAL=SYSTEM STR #
        NO SKIP RETURN IF CANNOT FIND STR
FNDSTR: DO SCNSTR, FOR ALL STR DATA BLOCKS (UPPER CORE)[SYSSTR=STRSYS]
        IF FIND MATCH [STRNAM], OK RETURN UPPER CORE ADR. THIS + PRED
SCNSTR: CONTINUE
        ERROR RETURN
```

```
          SUBROUTINE TO READ DOUBLEY WRITTEN SPECIAL BLOCKS [REDRUN]
          ARGS LOGICAL BLOCKS NOS, OF EACH
                SIXBIT NAME CHECK AND FOR ERROR MESSAGE
                SPECIAL CODE TO CHECK FOR
                ADDRESS OF WHERE TO READ
                UNIT DATA BLOCK ADDRESS
        ERROR RETURN OR CH RETURN
        ONCE ONLY IN PROGRESS  [DINIZE IS NOT EQUAL TO Ø]
        READ SECOND BLOCK
        IF HARDWARE ERROR
          SET LIGHTS TO CONTROLLER STATUS WORD STORED IN UNIT DATA  BLOCK BY FILSER
          TYPE "HARDWARE ERROR-SECOND XXX BLOCK ON YYYN
        END
        IF NAME WORD OR CODE WORD DO NOT AGREE
          TYPE "CONSISTANCY ERROR-SECOND XXX BLOCK ON YYYN
        END
        READ FIRST BLOCK
        IF HARDWARE ERROR
        SET LIGHTS TO CONTROLLER STATUS WORD STORED IN UNIT DATA BLOCK BY FILSEN
          TYPE "HARDWARE ERROR-FIRST XXX BLOCK ON YYYN
        END
        IF NAME WORD OR CODE WORD DO NOT AGREE
          TYPE "CONSISTANCY ERROR-FIRST XXX BLOCK ON YYYN
        END
        IF NEITHER BAD ERROR, OK RETURN
        IF FIRST BLOCK WAS BAD BUT SECOND BLOCK WAS GOOD
          READ SECOND BLOCK
          OK RETURN
        END
        ERROR RETURN
        TYPSUS-ROUTINE TO TYPE ALL STRS AND UNITS IN STRS IN SYSTEM
TYPSYS: DO TYPALL, FOR ALL STR DATA BLOCKS (UPPER CORE)[SYSSTR-STRSYS]
          TYPE THIS STR NAME, IF NEEDS REFRESHING, PHYSICAL (AND LOGICAL) UNITS [CALL TYPSTR]
TYPALL: CONTINUE
        RETURN

        TYPSTR-ROUTINE TO PRINT STR NAME, IF NEEDS REFRESHING, PHYSICAL (AND LOGICAL NAMES)
        ARG-ADDRESS OF STR DATA BLOCK (UPPER CORE)

TYPSTR: IF THIS STR NEEDS REFRESHING [RH(STRRET)], TYPE "NEEDS REFRESHING:"
        TYPE STR NAME [STRNAM]
        DO TYPUNI, FOR ALL UNIT DATA BLOCKS (LOWER CORE) IN THIS STR [STRUNI-UNISTR]
          IF FIRST TIME THRU LOOP
            TYPE ":"
          ELSE
            TYPE","
          END
          TYPE PHYSICAL UNIT NAME [UNINAM]
          TYPE "("
          TYPE UNIT HOME ID NAME [UNIHID]
          TYPE ")"
TYPUNI: CONTINUE
        TYPE CR-LF
        RETURN
```

```
        EXAMPLE:
        NEEDS REFRESHING:DSKA:FHAØ(),FHA1(),FHA2()
        DSKB:DPAØ(PØ3),DPA1(PØ7),DPA2(NX375)

        TYPUNS=ROUTINE TO TYPE ALL PHYSICAL UNITS NOT IN AN STR

TYPUNS: TYPE "UNITS NOT IN A FILE STRUCTURE:"CR-LF
        DO SCNUNI, FOR ALL UNIT DATA BLOCKS (LOWER CORE) IN SYSTEM [SYSUNI=UNISYS]
          IF THIS UNIT IS NOT IN AN STR [UNISTR=Ø]
                IF THIS IS NOT THE FIRST TIME THROUGH LOOP, TYPE ","
                TYPE PHYSICAL UNIT NAME [UNINAM]
                TYPE "("
                TYPE HOME IN) NAME [UNIHID] (TYPE NOTHING BETWEEN PAR IF Ø)
                TYPE ")"
        END
SCNUNI: CONTINUE
        TYPE CR-LF
        RETURN

        ASKPAR,CHGPAR=ROUTINES TO TYPE OR TYPE/CHANGE PARAMETERS
CHGPAR: TYPE "TYPE STR NAME TO CHANGE ITS PARAMETERS (CR=NONE, DSK=ALL)[CALL ASKSTR]
        IF CR RETUR, RETURN
        TYPE "AFTER EACH PRINTING OF CURRENT VALUE, TYPE NEW VALUE OR CR", CR-LF
        SET TYPPAR FLAG SO TYPPAR ROUTINE WILL TYPE AND CHANGE PARAMETERS [TYPONL=Ø]
        TO ASKLUP

ASKPAR: TYPE "TYPE STR NAME FOR LIST OF STR PARAMETERS (CR=NONE, DSK=ALL)[CALL ASKSTR]
        IF CR RETURN, RETURN
        SET TYPPAR FLAG SO TYPPAR ROUTINE WILL ONLY TYPE PARAMETERS [TYPON1=-1]
        IF "DSK" WAS TYPED [Ø ADDRESS RETURNED]
                DO SCNSTR, FOR ALL STR DATA BLOCKS (UPPER CORE)
                  TYPE STR NAME [STR NAM]
                  PRINT PARAMTERES FOR THIS STR [CALL TYPPAR]
SCNSTR: CONTINUE
          RETURN
        ELSE
          PRINT PARAMETERS FOR STR [CALL TYPPAR]
        END
        IF TYPONL SET, TO ASKPAR
        TO CHGPAR
          ASKSTR=ROUTINE TO TYPE MESSAGE AND ACCEPT STR NAME AND CHECK IF LEGAL
          ARG=ADR. OF MESSAGE
          VAL=ADDRESS OF STR DATA BLOCK1, Ø IF DSK TYPED
          VAL=ADDRESS OF PREDESSOR, STR DATA BLOCK (MAYBE SYSSTR)
          RET=NON-SKIP IF CR, REPEAT QUESTION IF NOT A VALID STR TYPED

ASKSTR: TYPE MESSAGE WHICH IS PASSED AS AN ARGUMENT
        ACCEPT INPUT FROM CTY
        IF CR, NO SKIP RETURN
        IF "DSK" WAS TYPED, GIVE SKIP RETURN WITH Ø VALUE
        SEARCH UPPER CORE FOR STR DATA BLOCK [CALL FNDSTR]
        IF NOT FOUND RETURN, TO ASKSTR (ASK QUESTION AGAIN)
        OK RETURN
```

```
        TYPPAR←ROUTINE TO TYPE OR TYPE/CHANGE PARAMETERS
        IF C(TYPONL) IS NONZERO PARAMETERS ARE TYPED
        IF C(TYPONL) IS ZERO PARAMETERS ARE TYPED AND CHANGES ARE ACCEPTED
        ARG←ADR, OF FILE STRUCTURE DATA BLOCK (UPPER CORE)

TYPPAR: TYPE "PARAMETERS WHICH MAY BE CHANGED WITHOUT REFRESHING,"CR←LF
        ASK "NO, CONSECUTIVE CLUSTERS TRIED FOR ON SEQUENTIAL OUPUT="
            TYPE ITS VALUE AND ACCEPT CHANGE [CALL ASKDEC]
        IF NO CHANGE OR CR OR TYPE ONLY RETURN, TO ASKBGA
        DO SCNUNI, FOR ALL UNITS IN THIS STR
            STORE NEW NO. OF CLUSTERS TRIED ON OUTPUT FOR THIS STR [LH(UNIGRP)]
SCNUNI: CONTINUE
ASKBGA: ASK "SUM OF BLOCKS GUARRANTEED TO USERS=",VALUE [STRGAR], ACCEPT DEC [CALL ASKDEC]
        IF NO CHANGE, OR CR OR TYPE ONLY RETURN, TO ASKBOU
        STORE NO. OF BLOCKS GUARRANTEED FOR THIS STR [STR GAR]

ASKBOU: ASK "NO. OF BLOCKS OVERDRAW/USER=",VALUE [←STROVR], ACCEPT DEC NO. [CALL ASKDEC]
        IF NO CHANGE, OR CR, OR TYPE ONLY RETURN, TO ASKNSC
        STORE NO. OF BLOCKS OVERDRAW PER USER FOR THIS STR [←STROVR]

ASK:    ASK "NO OF SAT BLOCKS IN CORE PER UNIT=", VALUE [UNY
        TYPPAR←CONT
        TYPE "PARAMETERS WHICH REQUIRE REFRESHING IN ORDER TO CHANGE: "CR←LF
ASKK4S: TYPE "K FOR SWAPPING ON LAST UNIT=",VALUE[STRK4S],ACCEPT DEC[CALL ASKDEC]
        IF NO CHANGE, CR, OR TYPE ONLY RETURN, TO TYPBPC
        IF VALUE EXCEEDS LAST UNIT [VALUE*BLKBPK EXCEEDS UNYBPU]
          TYPE "SWAPPING CANNOT EXCEED LAST UNIT=",VALUE[UNYBPU/BLKBPK]
          TO ASKK4S
        END
        STORE NO. OF K FOR SWAPPING ON THIS STR[STRK4S]
        SET NO. OF BLOCKS IN THIS STR TO *NO. OF BLOCKS FOR SWAPPING [STRHGH←-STRK4S*BLKBPK]
        DO SCNUNI, FOR ALL UNIT DATA BLOCKS IN THIS STR
          FLAG THIS UNIT AS REQUIRING HOME BLOCK TO BE REWRITTEN[UNPCHG]
          INCREASE NO. OF BLOCKS IN STR BY NO OF BLOCKS PER UNIT
SCNUNI: CONTINUE
ASKBPC: ASK "BLOCKS/CLUSTER=", TYPE VALUE [UNYBPC], ACCEPT DEC. [CALL ASKDEC]
        IF NO CHANGE, CR, OR TYPE ONLY RETURN TO TYPBCA
        IF VALUE IS ZERO, TYPE "CANNOT BE 0", TO ASKBPC
        IF VALUE EXCEEDS UPPER LIMIT [LIMBPC], TYPE "CANNOT EXCEED NNN" [TYPDEC], TO ASKBPC
        STORE DIFFERENT NO. OF BLOCKS PER CLUSTER IN EVERY UNIT DATA BLOCK [UNYBPC][CALL STOUNI]
        FLAG STR AS NEEDING REFRESHING [RH(STRREF)]
        COMPUTE SIZE OF CLUSTER ADDRESS FIELD FOR RETRIEVAL POINTERS
            [IE MAX. NO. BITS TO REPRESENT LAST CLUSTER ON UNIT[(UNIBPU-1)/UNIBPC]]
            [USING JFFO FIND FIRST 1, SUBTRACT FROM 36 GIVES NO. OF BITS REQUIRED]
        STORE NUMBER OF BITS REQUIRED FOR CLUSTER ADDRESS [BITS 6-11 STRCLP]
        STORE NUMBER OF BLOCKS PER SUPER CLUSTER [STRBSC←UNYBPC*(((STRHGH-8)/(2↑18))+1]
        STORE NUMBER OF SUPER CLUSTERS PER UNIT [STRSCU←((UNIBPU-1)/STRBSC)+1]
        STORE NO. OF CLUSTERS PER SAT BLOCK [STRSSZ←MIN(128*36,((UNIBPU-1)/UNYBPC)+1]
        CLEAR MULTIPLE SAT PER UNIT FLAG [UNPMSB] FOR ALL UNITS IN STR [CALL STOVNI]
```

```
        IF MORE THAN 1 SAT BLOCK PER UNIT [128*36 LESS THAN (((ONIBPU-1)/UNYBPC)+1)]
            SET MULTIPLE SAT PER UNIT FLAG [UNPMSB] FOR ALL UNITS IN STR [CALL STOUNI]
        END

TYPBCA: TYPE "THEREFORE BITS/CLUSTER ADR.=", VALUE[BITS 6-11 STRCLP][CALL TYPDEC]
        TYPE "THEREFORE WORDS/SAT=",VALUE[((STRSSZ-1)/36)+1][CALL TYPDEC]
        TYPE "THEREFORE BLOCKS/SUPER CLUSTER=",VALUE[STRBSC][CALL TYPDEC]
        TYPE "THEREFORE SATS/UNIT=",VALUE[(((UNIBPU-1)/UNYBPU)/(128*36))+1][CALL TYPDEC]
```

```
        TYPPAR←CONT

ASKBCC: TYPE "BITS/CLUSTER COUNT=",VALUE[BITS 6-11 STRCNP],ACCEPT DEC[CALL ASK DEC]
        IF NO CHANGE CR, OR TYPE ONLY RETURN, TO TYPBCK
        IF VALUE IS 0, TYPE "CANNOT BE 0", TO ASKBCC
        IF VALUE IS GREATER THAN UPPER LIMIT[LIMCNP←18], OR 36,-(BITS 6-11 STRCLP)
            TYPE "CANNOT EXCEED", MIN(LIMCNP, 36-(BITS 6-11 STRCLP))[CALL TYPDEC]
            TO ASKBCC
        END
        STORE NO. OF BITS PER CLUSTER COUNT FIELD IN RETRIEVAL POINTERS [BITS 6-11 STYCNP←VALUE]
        STORE RIGHT MOST BIT ADR OF SAME FIELD [BITS 0-5 STYCNP←36,-BITS 6-11 STYCNP]
        STORE NO. OF BITS PER CHECKSUM FIELD IN RET. PTR[BITS 6-11 STYCKP←36,-BITS 6-11 STRCNP-BITS 6-11 STYCLP]
        STORE RIGHT MOST BIT ADR OF SAME FIELD [BITS 0-5 STYCKP←BITS 0-5 STYCNP+BITS 6-11 STYCKP]

TYPBCK: TYPE "THEREFORE BITS/CHECKSUM=", VALUE [BITS 6-11 STYCKP][CALL TYPDEC]
```

```
        ASKDEC - ROUTINE TO PRINT MESSAGE, DECIMAL NO, ANY ACCEPT A DEC. VALUE
                 NO SKIP RETURN IF CR, VALUES ARE SAME, OR TYPONL FLAG ON
                 REPEATS MESSAGE IF NOT A DECIMAL NUMBER
                 ARGS - DEC. NO., ADR. OF MESSAGE

ASKDEC: SAVE ADDRESS OF MESSAGE IN CASE ERROR AND VALUE PASSED
ASKAGN: TYPE MESSAGE PASSED AS AN ARGUMENT AND TYPE VALUE [CALL TYPDEC]
        ACCEPT DECIMAL INPUT [CALL GETDEC]
        IF ERROR RETURN, RESTORE MESSAGE ADR AND VALUE, TO ASKAGN
        IF INPUT MINUS(CR), NO SKIP RETURN
        IF ORIGINAL VALUE AND THIS ONE ARE SAME, NO SKIP RETURN
        SKIP RETURN

        TYPDEC - ROUTINE TO TYPE MESSAGE AND DECIMAL VALUE

TYPDEC: TYPE MESSAGE PASSED AS ARGUMENT
        TYPE DECIMAL VALUE PASSED AS ARGUMENT [CALL TYPNUM]
        RETURN

        GETDEC - ROUTINE TO ACCEPT DECIMAL NO.
        ERROR RETURN IF NOT A DECIMAL NUMBER
        RETURN - VALUE   VALUE=-1 IF CR TYPED OR TYPE ONLY FLAG ON

GETDEC: IF TYPE ONLY FLAG ON[TYPONL=-1], SKIP RETURN WITH -1 VALUE
        GET NEXT CHAR
        IF CR, SKIP RETURN WITH -1 VALUE
        SET VALUE TO Ø
CHRLUP: IF NOT A NUMBER, ERROR RETURN
        MULTIPLY VALUE BY 1Ø AND ADD THIS CHAR - "Ø"
        GET NEXT CHAR
        IF CR, SKIP RETURN WITH VALUE
        TO CHRLUP

        STOUNI - ROUTINE TO STORE A BYTE IN ALL UNIT DATA BLOCKS
                 WITHIN AN STR
          ARGS - ADDRESS OF STR DATA BLOCK
               - VALUE TO BE STORED
               - BYTE POINTER WITH INDEX U

STOUNI: DO SCNUNI, FOR ALL UNITS IN THIS STR
          STORE VALUE USING BYTE POINTER
          FLAG THIS UNIT AS NEEDING HOME BLOCK REWRITTEN BECAUSE CHANGE[UNPCHG]
SCNUNI: CONTINUE
```

```
        ASKDIS - FLOW FOR ONCE ONLY SUBROUTINE TO ASK ABOUT DISSOLVING STRS
        NO ARGS, NO VALUES, LOOPS UNTIL DONE
        ON LINKS UNIT DATA BLOCKS (LOWER CORE) RETURNS STR DATA BLOCKS (UPPER CORE) TO FREE STORAGE

ASKDIS: ASK "TYPE STR NAME TO DISSOLVE (CR IF NONE, DSK IF ALL)", ACCEPT AND CHECK STR[CALL ASKSTR]
        IF CR RETURN, RETURN
        IF "DSK WAS TYPED [Ø ADDRESS RETURNED]
          DO SCNSTR, FOR ALL STR DATA BLOCKS (UPPER CORE)
            DISSOLVE THIS STR AND UNLINK UNIT DATA BLOCKS [CALL DISSTR]
SCNSTR:   CONTINUE
          RETURN
        ELSE
          DISSOLVE THIS STR AND UNLINK UNIT DATA BLOCKS [CALL DISSTR]
        END
        TO ASKDIS

        DISSTR - FLOW FOR ONCE ONLY SUBROUTINE TO DISSOLVE AN STR
        ARG - ADDRESS OF STR DATA BLOCK (UPPER CORE), ADDRESS OF PREDESSOR (RETURNED BY ASKSTR)
        VOL - NONE

DISSTR: DO SCNUNI, FOR ALL UNIT DATA BLOCKS (LOWER CORE) IN STR TO BE DISSOLVED[STRUNI←UNISTR]
          MARK UNIT AS NOT IN AN STR[UNISTR←Ø,UNILOG←Ø]
          MARK UNIT AS NEEDING REFRESHING [UNPREF←1]
SCNSTR: CONTINUE
        (PREDESSOR RETURNED BY ASKSTR FROM FNDSTR)
        MAKE PREDESSOR STR BLOCK POINT TO STR DATA BLOCK AFTER STR BEING DISSOLVED
            [LH(STRSYS(PREDESSOR))←LH(STRSYS(STR BEING DISSOLVED))]
        RETURN STR TO FREE STORAGE (SPACE WILL BE REUSED IF NEEDED)[CALL    ]
        RETURN

        ASKDEF - FLOW FOR ONCE ONLY SUBROUTINE TO ASK ABOUT DEFINING STRS
        NO ARG, NO VALUES, LOOP UNTIL DONE
        CREATES STR DATA BLOCKS (UPPER CORE) AND LINKS UNIT DATA BLOCKS (LOWER CORE)

ASKDEF: TYPE "TYPE STR NAME TO DEFINE A NEW ONE(CR IF NONE)"
        ACCEPT INPUT FROM CTY
        IF ONLY CR TYPED, RETURN
        CHECK TO SEE IF THIS STR NAME ALREADY EXISTS[CALL TNDSTR]
        IF NOT FOUND, TO DEFNEW
        TYPE "STR NAME ALREADY EXISTS" CR-LF
        TO ASKDEF
```

```
DEFNEW: GET FREE CORE(UPPER CORE) FOR STR DATA BLOCK[CALL   WHICH CLEARS IT]
        (LAST STR DATA BLOCK RETURNED BY FNDSTR)
        LINK LAST STR DATA BLOCK TO THIS NEW ONE(SUBTRACT OFFSET BEFORE STORING)[STRSYS]
        REMEMBER STR DATA BLOCK ADDRESS (UPPER CORE) AS IF UNIT DATA BLOCK
ASKNXT: ASK "TYPE NEXT PHYSICAL UNIT(CR WHEN DONE)"[CALL ASKUNI]
        IF CR RETURN, RETURN
        APPEND THIS UNIT DATA BLOCK TO LAST ONE IN THIS STR[STRUNI←UNISTR]
        REMEMBER THIS ONE AS NEW LAST ONE
        MAKE SURE THIS UNIT IS FLAGGED AS LAST UNIT[LH(STRUNI)←0]
        SET THIS UNITS UPWARD POINTER TO POINT TO STR DATA BLOCK[RH(STRUNI)]
        STORE UNIT LOGICAL NUMBER WITHIN STR[UNYLUN←STRUNM]
        STORE UNIT LOGICAL NAME WITHIN STR[UNILOG←STRNAM&STRUNM]
        INCREASE HIGHEST LOGICAL BLOCK IN THIS STR[STRHGH←STRHGH+UNIBPU]

        INCREASE NUMBER OF UNITS IN THIS STR[STRUNM](LAST SO CAN USE ABOVE A LOGICAL NO.)
        TO ASKNXT

        ASKUNI - FLOW FOR ONCE ONLY SUBROUTINE TO ASK FOR PHYSICAL UNIT
        ARG - MESSAGE ADDRESS
        VAL - UNIT DATA BLOCK ADDRESS (LOWER CORE)
        NO SKIP RETURN IF JUST CR TYPED

ASKUNI: REMEMBER MESSAGE ADDRESS IN CASE ERROR
ASKAGN: TYPE MESSAGE PASSED AS AN ARGUMENT
        ACCEPT CTY INPUT
        IF JUST CR, NO SKIP RETURN
        SCAN UNIT DATA BLOCKS LOOKING FOR MATCH[CALL FNDUNI]
        IF NOT FOUND RETURN
          TYPE "NOT A PHYSICAL UNIT"
          TO ASKAGN
        END
        SKIP RETURN

        FNDUNI - FLOW FOR ONCE ONLY SUBROUTINE TO FIND UNIT DATA BLOCK ADR
        ARG - PHYSICAL NAME
        VEL - UNIT DATA BLOCK ADDRESS (LOWER CORE)
        NO SKIP RETURN IF NOT FOUND

FNDUNI: DO SCNUNI, FOR ALL UNITS IN SYSTEM [SYSUNI←UNISYS]
        IF FIND MATCH [UNINAM], OK RETURN
SCNUNI: CONTINUE
        ERROR RETURN
```

```
;FILFLO.MEM - ENGLISH FLOW CHARTS FOR FILSER - V005  3 DEC 69
;THE ENGLISH FLOW CHARTS AND SUPPORTING DOCUMENTATION
; IS KEPT AS A SEPARATE FILE WHICH CAN BE ASSEMBLED WITH FILSER
LOGIC==0                ;MAKE COMMENTS BE IGNORED


REPEAT  LOGIC,<
;ADD LOTS OF < TO ALLOW < AND > DURING THE TEXT; END WITH LOTS OF >
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<


                TABLE OF CONTENTS

SYMBOLIC PARAMETER DEFINITIONS - APPEAR IN A SEPERATE FILE NAMED DSKPAR,MAC,
THEY DEFINE ALL TABLE AND DISK BLOCK LAYOUTS AND ARE THE SOLE SOURCE
OF THAT SPECIFICATION,  IT SEEMS BETTER TO HAVE ONLY ONE PLACE FOR SUCH
DOCUMENTATION WHICH MUST BE CORRECT RATHER THAN 2 PLACES WHICH CAN DIFFER,

1, IMPLEMENTATION POLICY AND IMPLEMENTATION GOALS

2, ACCUMULATOR USAGE

3, SEQUENCING OF OPERATIONS

4, UUO FLOW IN UUOCON
        ENTER
        LOOKUP
        INPUT CLOSE
        OUTPUT CLOSE
        INIT

5, UUO FLOW IN FILSER
        LOOKUP
        ENTER
        INPUT
        OUTPUT

6, GLOSSARY
```

1. IMPLEMENTATION POLICY IMPLEMENTATION GOALS

1. TRY TO MAKE UUO'S RESTARTABLE AT ANY TIME
   A. ESPECIALLY BETWEEN IO WAITS

   B. BETWEEN INSTRUCTIONS

UUO CODE MUST NOT MODIFY DATA BASE IN DDB UNTIL A PROCESS HAS REACHED SUCCESSFUL TERMINATION. SAME IDEA HOLDS FOR UUOCON AND UUO PROGRESS BITS.

2. MAKE CODE DEFENSIVE, FOR EXAMPLE INPUT TESTS THAT DATA IN DDB IS SELF CONSISTENT, IF POSSIBLE.

3. DON'T TRUST HARDWARE, BUT DEMAND THAT HARDWARE STATUS CORRESPOND TO COMPUTED EXPECTED STATUS OR REMEMBERED STATUS.

4. KEEP MAXIMUM AMOUNT OF DEVICE STATUS INFORMATION IN CORE.

   A. MAKES INITIAL DEBUGGING EASIER

   B. FACILITITATES PROGRAM MAINTENANCE.

5. TRY TO MAKE AS MUCH DATA SWAPPABLE AS POSSIBLE INCLUDING MOST OF DATA IN DSKDDB'S

6. MAKE PROGRAM EASIER TO MAINTAIN BY

   A. SHARPLY DEFINED MODULES

   B. CONVENTIONAL CALLING SEQUENCES

V002 1. DEFINE PJRST TO BE OPCODE(S.MAC) TO REPLACE PUSHJ
V002 POPJ PAIR AT END OF A SUBROUTINE INSTEAD OF USING
V002 JRST. JRST WILL REMAIN TO BE USED FOR TRANSFER
V002 OF CONTROL TO TAGS WHICH ARE NOT SUBROUTINES.

V002 2. USE MACRO !STOPCD ARG! INSTEAD OF HALT FOR HALTS.
V002 IN S.MAC DEFINE ARG AS AN OCTAL SYMBOL USE ONE
V002 HALT CODE PER EACH STOPCD. DEFINE STOPCD AS
V002 ANY OF! HALT
V002 UUO
V002 JFCL

   C. ACCUMULATOR CONVENTIONS

2. ACCUMULATOR USAGE

2.1 INTER MODULE CONVENTION

V002 THE DISK SERVICE WILL BE DIVIDED INTO 20 OR SO MODULES. THESE
V002 MODULES WILL FOLLOW THE FOLLOWING SUBROUTINE CONVENTION WHEN
V002 CALLING EACH OTHER. SECTION 2.2 DESCRIBES TWO EXCEPTIONS WHICH
V002 ARE ALLOWED ONLY WITHIN MODULES.
WE DEFINITELY REQUIRE A MORE DISCIPLINED USE OF ACCUMULATORS WITHIN THE MONITOR NOT ONLY TO REDUCE EXTRANEOUS PUSH'S AND POP'S BUT ALSO TO ENABLE ALL SUBROUTINES TO ASSUME INTEGRITY OF DATA IN AC'S. THREE CLASSES OF ACCUMULATORS ARE DEFINED:

1. GLOBAL
2. TEMPORARY
3. PRESERVED

GLOBAL ACCUMULATORS SUCH AS THE CURRENT F,P,R ALWAYS CONTAIN THE SAME TYPE OF INFORMATION. ONLY A SELECT SET OF ROUTINES EVER EXPLICITLY MODIFIES THE CONTENTS OF GLOBAL AC'S. GLOBAL ACS ARE IMPLICIT ARGUMENTS TO MANY ROUTINES.

TEMPORARY ACCUMULATORS SUCH AS CURRENT T,T1, ETC., ARE USED FOR WORKING STORAGE AS WELL AS BEING THE STANDARD PLACE TO PASS ARGUMENTS AND RETURN VALUES.

PRESERVED ACCUMULATORS FORM A NEW AND MUCH NEEDED CLASS. PRESERVED AC'S MAY BE USED AS DESIRED AFTER BEING SAVED. THEIR INTEGRITY IS ASSURED AS ALL SYSTEM SUBROUTINES WILL USE A COMMON SET OF ROUTINES TO SAVE AND RESTORE THESE AC'S. THEY
V002 WILL NOT BE USED TO RETURN VALUES. HOWEVER THEY MAY BE USED
V002 TO PASS ARGUMENTS WHEN THE QUANTITIES ARE THE TYPE OF THING WHICH
V002 WANT TO STAY AROUND ACCROSS A NUMBER OF SUBROUTINE CALLS.

SUBROUTINES SAV1, SAV2, SAV3 AND SAV4 WILL BE USED TO PRESERVE 1, 2, 3 OR 4 OF THE COMMON PRESERVED AC'S. PRESERVED AC'S MUST BE USED IN ORDER, THAT IS IF ONE IS REQUIRED P1 IS USED, IF TWO ARE REQUIRED P1, P2 ARE USED AFTER CALLING SAV1, SAV2 RESPECTIVELY. THE STANDARD POPJ P, OR JRST CPOPJ1 RETURNS WILL RESTORE ALL OF THE PRESERVED AC'S UPON EXIT RELIEVING EACH SUBROUTINE OF THAT TEDIOUS DUTY.

V003 THE NAMES OF ALL SUBROUTINES SATISFYING THESE CONVENTIONS WILL
V003 CONTAIN SIX CHARACTERS, WHILE VIOLATORS OF CONVENTION (SEE 2.2)
V003 WILL BE GIVEN 5 OR FEWER CHAR NAMES. THIS MAKES IT CLEAR TO
V003 THE READER WHEN HE SEES A PUSHJ WHAT IS WHAT WITHOUT GOING TO
V003 THE SUBROUTINE.

EXAMPLE OF AC SAVING ROUTINE AND USE

```
SUBR:   JSP T3,SAV2
        JRST CPOPJ1

SAV2:   PUSH P,P1
        PUSH P,P2
        PUSHJ P,(T3)
        JRST .+2
        AOS - 2(P)
        POP P,P2
        POP P,P1
        POPJ P,
```

```
V002    NO SUBROUTINE WILL "POP UP MORE THAN ONE LEVEL" ON A RETURN.
V002    TPOPJ AND TPOPJ1 MAY BE USED ONLY FOR POPPING OFF DATA (RATHER
V006    THAN A RETURN).  THUS THE PROGRAM FLOW IS ALWAYS APPARANT
V006    TO THE READER BECAUSE IT ALWAYS RETURNS IMMEDIATELY AFTER
V006    PUSHJ OR ONE LOCATION AFTER IT IF SUBROUTINE CAN GIVE A SKIP RETURN.
        NO SUBROUTINE WILL PASS ARGUMENTS OR RETURN VALUES
        ON THE PUSH DOWN LIST, NO SUBROUTINE WILL
        LEAVE THE PUSHDOWN LIST IN A DIFFERENT CONDITION
        OR DIFFERENT DEPTH THAN WHEN IT WAS CALLED.
        (EXCEPT OF COURSE SUCH ROUTINES AS SAV1...SAV4,
        CPOPJ,CPOPJ1,TPOPJ, AND TPOPJ1 WHICH EXIST
        SOLELY FOR THE PURPOSES OF HANDLING THE PUSH DOWN LIST.


2.2     INTRA MODULE CONVENTIONS

V002    BECAUSE THE ABOVE CONVENTIONS DO USE UP INSTRUCTIONS (PUSH'S AND
V002    POP'S AND MOVE T,P1 ETC.) AND PUSHDOWN LIST SPACE, THE
V002    FOLLOWING TWO EXCEPTIONS ARE ALLOWED:
V003       (THESE SUBROUTINES ARE FLAGGED BY GIVING THEM FIVE OR FEWER
           SYMBOLS IN THEIR NAMES.)

V002    A. A SUBROUTINE (AND ALL THE ONES IT CALLS) MAY BE DESIGNED
V002       TO RESPECT THE CONTENTS OF SPECIFIED TEMPORARY ACS.  RESPECT
V002       MEANS THAT A SUBROUTINE (AND ALL THE ONES IT CALLS) EITHER DO
V002       NOT CHANGE THE CONTENTS OF AN AC (IT MAY OR MAY NOT BE AN
V002       ARGUMENT) OR IT PUSH'S IT BEFORE MODIFICATION AND POP'S IT
V002       BEFORE RETURN.  WHEN THIS IS THE CASE THE COMMENTS AT THE
V002       BEGINNING OF THE SUBROUTINE WILL SAY:
V002       "THIS SUB. AND ALL THE ONES IT CALLS RESPECT THE CONTENTS
V002       OF TEMP ACS T1,T3." - FOR EXAMPLE

V003    B. A SUBROUTINE MAY RETURN A VALUE IN A PRESERVED AC (SO THAT
V003       THE AC IS NOT PRESERVED).  THIS EXCEPTION WILL ALSO BE
V003       CLEARLY STATED AT THE BEGINNING OF A SUBROUTINE.  THIS TECH-
V003       NIQUE WILL BE USED INFREQUENTLY AND WILL BE USED FOR SETTING
V003       UP A PRESERVED AC FOR A RELATIVELY GLOBAL QUANTITY WITHIN
V003       A MODULE.

V003    ALL JRST'S WILL BE TO TAGS DOWN PAGE, EXCEPT LOOPS.  THUS FLOW
V003    GOES DOWN THE PAGE, AND THE READER MAY START AT THE BEGINNING
        OF THE CODE AND READ THE PROGRAM WITHOUT BECOMING CONFUSED
        ABOUT THE PROGRAM FLOW.
```

ACCUMULATOR USAGE

	OLD NAME	NEW NAME	NEW ASSIGNMENT	
	IOS	S	0	
↑	PDP	P	1	↑
↑	ITEM	U	2	↑
↑	PROG	R	3	GLOBAL
↑	DEVDAT	F	4	
↑	DAT	C	5	↑
INTERRUPT AND UUO LEVELS	TAC	T	6	TEMPORARY
	TAC1	T1	7	
	TEM	T2	10	
↑	DSER	T3	11	↑
↑	UCHN	UCHN	12	GLOBAL
↑	UUO	UUO	13	
INTERRUPT AND UUO LEVELS	AC1	P1	14	↑
	AC2	P2	15	"PRESERVED"
	AC3	P3	16	
↑	BUFWRD	P4	17	

```
V004    TYPING CONVENTIONS:
V004            ONE TAB INSTEAD OF ONE SPACE BETWEEN INSTR, AND AC
V004            MULTI-LINE COMMENTS WILL BE INDICATED BY ONE SPACE
V004               AFTER THE SEMI COLON ON ALL BUT THE FIRST LINE.
V004            SUBROUTINE WILL BE PRECEDED WITH A DESCRIPTION OF
V004               FUNCTION, ARGS, AND VALUES,  THE WORD "SUBROUTINE"
V004               WILL BE THE FIRST WORD ON THE FIRST LINE.
V004            TAGS WHICH APPEAR FIRST ON A PAGE AND THEREBY
V004               LOOK LIKE SUBROUTINES WILL BE INDICATED
V004               OTHERWISE BY A COMMENT STARTING AT LEFT
V004               MARGIN STARTING WITH WORD "HERE".
                MOST TAGS WILL BE PRECEDED BY A FULL LINE COMMENT
                   STARTING WITH ";HERE TO ... " TO DESCRIBE WHAT
                   THE CODE IS ABOUT TO DO OR ";HERE WHEN ... "TO
                   DESCRIBE THE CONDITIONS WHICH CAUSED CONTROL TO
                   GOT TO THE TAG
V004            THE INSTRUCTION OF A NON-SKIP SUBROUTINE RETURN WILL
V004               BE INDENTED 2 SPACES SO THE READER WILL
V004               BE ABLE TO TELL SUCH SUBROUTINES WHICH HAVE 2
V004               RETURNS.
                ALL SUBROUTINES WILL BE PRECEDED BY COMMENTS STARTING AT LEFT MARGIN
                   WITH "SUBROUTINE TO ...", ALONG WITH DESCRIPTION OF ARGS
                   ALSO ANY IMPORTANT OVERVIEW COMMENTS.
                ALL TAGS WHICH ARE LOOPS WILL BE PRECEDED BY A FULL LINE COMMENT
                   "LOOP TO .." DESCRIBING WHAT THE LOOP IS DOING
                IF THERE ARE ANY TAGS (VIOLATIONS OF JRST DOWN PAGE) WHICH ARE TRANSFERRED
                   TO FROM FURTHER DOWN THE PAGE, SHALL INDICATE THIS BY A PRECEDING
                   FULL LINE COMMENT "BACK HERE TO ... "
```

3. SEQUENCING OF OPERATIONS

THE OPERATIONS OF THE FILE SYSTEM WILL BE CAREFULLY DESIGNED SO THAT A SYSTEM CRASH AT ANY TIME WILL NOT LEAVE THE DISK IN A STATE WHICH WILL CAUSE MORE INFORMATION TO BE LOST. THUS THE ORDER OF WRITING DISK BLOCKS OF FILES WHICH DESCRIBE THE FILE STRUCTURE IS IMPORTANT (MFD, UFD,SAT,RIB). ALSO IMPORTANT IS THE ORDER IN WHICH CORE MEMORY LOCATIONS ARE UPDATED AND CHANGED. AN ATTEMPT WILL BE MADE TO ALLOW A JOB TO BE STOPPED (CONTROL C) OR RESCHEDULED AT ANY TIME DURING DISK IO OR BETWEEN ANY TWO INSTRUCTIONS. THE NO SCHEDULE, SCHEDULE MACRO WILL BE USED AROUND GROUPS OF INSTRUCTIONS FOR WHICH NO SCHEDULING (AND, THEREFORE STOPPING OF THIS JOB) CAN OCCUR. SOME DATA LOCATIONS ARE CHANGED AT UUO AND INTERRUPT LOCATIONS. THE NUMBER OF SUCH LOCATIONS SHALL BE MINIMIZED AND SHALL BE CLEARLY LABELLED.

THE DISK SERVICE WILL BE PROGRAMMED SO THAT IT WILL BE ABLE TO RECOVER FROM ANY CONDITION IN WHICH THE USER TYPED CONTROL C AND STARTED HIS PROGRAM OVER AGAIN. TO ASSIST IN THIS, EACH ACCESS TABLE ENTRY (ONE PER ACTIVE OR DORMANT FILE) AND EACH DDB (ONE PER ACTIVE FILE-USER PAIR) WILL HAVE A SIMPLE STATE CODE IN IT, RATHER THAN BITS, SO THAT A SIMPLE DISPATCH ON PREVIOUS STATE CODE WHEN AN EVENT OCCURS WILL MAKE SURE THAT ALL POSSIBILITIES ARE ACCOUNTED FOR. IF ANY SEQUENCE OF INSTRUCTIONS (NO IO WAIT) MUST BE EXECUTED WITHOUT INTERRUPT THEN THE NOSHEDULE, SCHEDULE MACRO WILL BE USED.

SEQUENCING OF DISK BLOCK WRITES

THIS PROBLEM BECOMES MORE DIFFICULT BECAUSE THE OPTIMIZATION CODE WILL NOT NECESSARILY WRITE BLOCKS IN THE ORDER IN WHICH THEY ARE QUEUED. THE PROBLEM BECOMES EVEN MORE COMPLEX WHEN DIFFERENT SPEED DEVICES ARE PART OF THE SAME FILE STRUCTURE (E.G., DRUM AND BRYANT DISK). HOWEVER A REQUEST TO WRITE A GIVEN BLOCK FOLLOWED BY A REQUEST TO READ THE SAME BLOCK WILL ALWAYS OCCUR IN FIRST COME FIRST SERVE ORDER.

ORDER OF READING, WRITING, AND READ PAUSE WRITE OF BLOCKS

<> MEAN POSSIBLE REPEATED READS OR WRITES. EXIT FROM LOOP NOT NECESSARILY AT BOTTOM.
[] MEAN MAY BE UNNECESSARY BECAUSE OF RETRIEVED INFORMATION IN CORE. THE REASON IS INDICATED IN A SECOND SET OF BRACKETS.

```
V003    4.      FILE, UNIT, CHANNEL, CONTROLLER STATUS

V003    I  - IDLE
V003    SW - SEEK WAIT - POSITIONING DUE TO SEEK UUO, OF NO DATA
V003         TRANSFER AFTER
V003    S  - SEEK
V003    PW - POSITION WAIT
V003    P  - POSITION
V003    TW - TRANSFER WAIT
V003    T  - TRANSFERRING

V003    WAIT MEANS FILE IS IN SEEK/POSITION QUEUE FOR UNIT OR TRANSFER
V003    QUEUE FOR CHANNEL

V003    NOT ALL ENTITIES CAN BE IN ALL STATES:

V003                     FILE    UNIT    CHANNEL   KONTROLLER
V003            I         I       I        I         I
V003            SW                SW
V003            S                 S
V003            PW        PW      PW
V003            P         P       P
V003            TW        TW      TW
V003            T         T       T        B         B

V003            B STANDS FOR BUSY AND HAPPENS WHEN A DATA TRANSFER

V003            IF A UNIT IS IN SW, ONLY ONE FILE CAN BE IN QUEUE
V003               AND IT IS LEFT IN IDLE STATE,  THE ONLY WAY
V003               A UNIT CAN GET INTO SEEK WAIT IS IF THE CONTROLLER
V003               IS BUSY WHEN A USER DOES A SEEK UUO.  SUBSEQUENT
V003               SEEKS OR POSITIONS FOR THE SAME OR DIFFERENT FILES
V003               ON THE SAME UNIT WILL CAUSE THE SEEK/POSITION
V003               QUEUE FOR THE UNIT (ONLY ONE FILE WHEN UNIT IN SW)
V003               TO BE CLEARED).
```

```
        |--------------|
SYSPPB  |  *   |       |                                  *<<<<<<<<<<<<<<<<*
        |--+-----------|                                  V                 *
           V                    |--------------|         |--------------|  *    |--------------|         |--------------|
PPBNAM     V                    |              |         |              |  *    |              |         |              |
           V                    |--------------|         |--------------|  *    |--------------|         |--------------|
PPBSYS/KNO *>>>>>>>>>>>>>>>>>>>>>|   >>>+>>>>>>>+>>>>>>>>|   >>>+>>>>>>>+>>>+>>>>|   >>>+>>>>>>>+>>>>>>>>|   0  |       |
                                |--------------|         |--------------|  *    |--------------|         |--------------|
PPBUFB/(18)                     |              |      *<+<<*             |  *    |              |         |              |
                                |--------------|      V |--------------|  *    |--------------|         |--------------|
PPBNMB/YES,NLG                  |              |      V |  *           |  *    |              |         |              |
                                |--------------|      V |--+-----------|  *    |--------------|         |--------------|
                                                      V    V                *
                          *<<<<<<<<<<<<<<<<<<<<<<<<+<<<<*                    *
                          V                           V                     *
                          V                           V                     *
                          V                           V                     *
                          V                           V |--------------|    *    |--------------|         |--------------|
UFBTAL                    V                           V |              |    *    |              |         |              |
                          V                           V |--------------|    *    |--------------|         |--------------|
UFBPPB/PRV,1PT,UN1,FSN    V                           *>|   >>>+>>>>>>>+>>>+>>>>|   >>>+>>>>>>>+>>>>>>>>|   0  |       |
                          V                             |--------------|    *    |--------------|         |--------------|
UFBPT1                    V                             |              |    *    |              |         |              |
                          V                             |--------------|    *    |--------------|         |--------------|
UFBTL1,WRT                V                             |              |    *    |              |         |              |
                          V                             |--------------|    *    |--------------|         |--------------|
                          V                                                 *
                          V                                                 *
                          V                                                 *
                          V                                                 *
                          V                                                 *
                          V |--------------|         |--------------|  *    |--------------|         |--------------|
NMBNAM                    V |              |         |              |  *    |              |         |              |
```

```
                                 V |--------------|        |--------------|   *   |--------------|        |--------------|
NMBPPB/KNO,FSN                   *>|   >>>+>>>>>>>+>>>>>>>>|   >>>+>>>>>>>+>>>+>>>>|   >>>+>>>>>>>+>>>>>>>>|   0  |       |
                                   |--------------|        |--------------|   *   |--------------|        |--------------|
NMBEXT/CFP                         |              |        |              |   *   |              |        |              |
                                   |--------------|        |--------------|   *   |--------------|        |--------------|
NMBACC/YES,GRB                     |              |      *<*<<<           |   *   |              |      *<*<<<           |
                                   |--------------|      V |--------------|   *   |--------------|      V |--------------|
                                                         V    *                *                        V    *
                                                         V    *<<<<<<<<<<<<<<<<*<<<<<<<<<<<<<<<<<<<<<<<<*  V    *<<<<<<<<<<<<<<<<*
                                                         V                     *                        *  V                     *
                                                         V                     *<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*
                                                         V                     *                 *      *  V                 *     *
                                                         V |--------------|   *   |-----------+--|   *  V |-----------+--|   *
ACCAKB,ACCPPB                            V |      |   *>>>+>>>>*    |      |  *   |   *  V |      |  *   |   *
                                         V |--------------|        |--------------|   *   |--------------|   *
ACCNMB/CNT,STS,BIT,FSN                   *>|   >>>+>>>>>>>+>>>>>>>>|   >>>+>>>>>>>+>>>*  *>|   >>>+>>>>>>>+>>>*
                                           |--------------|        |--------------|        |--------------|
ACCWRT                                     |              |        |              |        |              |
                                           |--------------|        |--------------|        |--------------|
ACCDOR                                     |              |        |              |        |              |
                                           |--------------|        |--------------|        |--------------|
```

```
                                        *<<<<<<<<<<<<<<<<<<
                                        V                 ^
                                      !---------------!   ^    !---------------!        !---------------!
ACCAKB,ACCPPB                       *<+<<*   !       !   ^  *<+<<*   !       !      *<+<<*   !       !
                                    V !---------------!   ^  V !---------------!      V !---------------!
ACCNMB/CNT,STS,BIT,FSN              V !               !   ^  V !               !      V !               !
                                    V !---------------!   ^  V !---------------!      V !---------------!
ACCWRT                              V !               !   ^  V !               !      V !               !
                                    V !---------------!   ^  V !---------------!      V !---------------!
ACCDOR      *<>><><><><><><><><><>V>!><>><>!<>><>!<><^<>V>!><>><>!<>><>!<><><>V>!><>><>!<>><>!
          ^       ^                 V !---------------!   ^  V !---------------!      V !---------------!
        !---------------!           V                     ^  V                         V
SYSDOR  !   *   !   *   !           V                     ^  V                         V
        !---------------!           V                     ^  V                         V
                                    V                     ^  V                         V
                                    V                     ^  V                         V
                                    *>>>>*                ^  *>>>>*                    *>>>>*
                                        V                 ^      V                         V
                                      !---------------!   ^    !---------------!        !---------------!
AKBALC                                !               !   ^    !               !        !               !
                                      !---------------!   ^    !---------------!        !---------------!
AKBADT/LBS,1PT,UN1                          !               ^  !   ^   !               !        !               !
                                      !---------------!   ^    !---------------!        !---------------!
AKBPT1                                !               !   ^    !               !        !               !
                                      !---------------!   ^    !---------------!        !---------------!
AKBPRV,MOD,CTM,CDT                    !               !   ^    !               !        !               !
                                      !---------------!   ^    !---------------!        !---------------!
                                                          ^
                                                          ^
                                                          ^
        !---------------!                                 ^
DEVLST  !   *   !       !                                 ^<<<<<<<<<<<<<<<<<<<<<<<*
        !---+-----------!                                 ^                       ^
          V                           !---------------!   ^    !---------------! ^
          V                           !               !   ^    !               ! ^
          V                           !---------------!   ^    !---------------! ^
DEVSER    *>>>>>>>>>>>>>>>>>>>>>>>!   >>>+>>>>>>>+>>>+>>>>!   >>>+>>>>>>>+>+>>
                                      !---------------!   ^    !---------------! ^
DEVAČC                                !   >>>+>>>>>>>+>>>^    !   >>>+>>>>>>>+>^
                                      !---------------!        !---------------!
                                      !               !        !               !
                                      !---------------!        !---------------!
```

READ A FILE

```
LOOKUP  [READ MFD RIB][NEVER NEEDED BECAUSE MFD ALWAYS HAS A DDB]
        [<READ MFD BLOCK>][NOT NEEDED IF JBTUFD ENTRY]
        [READ UFD RIB]["]
        [READ UFD BLOCK>][NOT NEEDED IF FILE ACTIVE OR DORMANT]
        <[READ FILE RIB][NOT NEEDED IF SHORT LOOKUP OR MEDIUM LOOK-
         UP AND FILE WAS ACTIVE OR DORMANT]
INPUT   <READ FILE BLOCK>>
CLOSE   [READ-PAUSE-WRITE RIB][NOT NEEDED UNLESS FIRST ACCESS OF DAY
         AND INPUTS DONE]
```

CREATE A FILE, SUPERSEDE INDICATED BY *

```
ENTER   [READ MFD RIB][NEVER NEEDED BECAUSE MFD ALWAYS HAS A DDB]
        [READ MFD BLOCK>][NOT NEEDED IF JBTUFD ENTRY]
        [READ UFD RIB]["]
        [READ UFD BLOCK>][NOT NEEDED SINCE FILE ACTIVE OR DORMANT
        *[READ FILE RIB][NOT NEEDED SINCE FILE DOESN'T EXIST ON A CREATE
        [READ SAT BLOCK][NOT NEEDED IF SAT TABLE ALREADY IN CORE]
        <WRITE NEW RIB BLOCK>
OUTPUT<<<WRITE FILE BLOCK>
        [WRITE OTHER SAT BLOCK READ SAT BLOCK]>[NOT NEEDED IF SAT TABLE
         ALREADY IN CORE]
        READ-PAUSE-WRITE RIB BLOCK>
        [READ SAT BLOCK][NOT NEEDED IF SAT TABLE NOT DIFFERENT FROM DISK]
CLOSE   WRITE PARTIAL FILE BLOCK
        READ-PAUSE-WRITE NEW FILE RIB BLOCK
        [WRITE SAT TABLE][NOT NEEDED IF NOT DIFFERENT FROM DISK]
        READ-PAUSE-WRITE UFD BLOCK
        <READ OLD FILE RIB BLOCK
        [[WRITE OTHER SAT BLOCK][NOT NEEDED IF NOT DIFFERENT FROM DISK]
        *READ SAT TABLE]][NOT NEEDED IF SAT TABLE ALREADY IN CORE]
        *WRITE SAT TABLE>
```

APPENDING, UPDATE A FILE

```
LOOKUP  [READ MFD RIB][
        [<READ MFD BLOCK>][
        [READ UFD RIB]["]
        [<READ UFD BLOCK>]
        [READ FILE RIB][NOT NEEDED IF SHORT LOOKUP OR
         MEDIUM LOOKUP AND FILE WAS ACTIVE OR DORMANT]
ENTER   <[READ FILE RIB][NOT NEEDED IF RIB ALREADY READ BY LOOKUP OR
         MEDIUM ENTRY AND FILE WAS ACTIVE OR DORMANT]
INPUT   <<READ FILE BLOCK>>
        [WRITE ALLOCATED BUT UNWRITTEN]>[NOT NEEDED UNLESS USER OVER-
         SHOOTS ON AN APPEND]
OUTPUT<<<WRITE FILE BLOCK>
        [WRITE OTHER SAT BLOCK
        READ SAT BLOCK]>[NOT NEEDED UNLESS APPENDING AND DESIRED SAT NOT IN CORE]
        READ-PAUSE-WRITE RIB BLOCK>
CLOSE   [READ-PAUSE-WRITE RIB BLOCK][NOT NEEDED UNLESS RIB CHANGED OR
         FIRST ACCESS OF DAY]
        [WRITE SAT TABLE][NOT NEEDED IF NOT DIFFERENT FROM DISK]
```

RENAME A FILE PREVIOUSLY CLOSED

```
LOOKUP  SAME AS READ IN OLD DIRECTORY
RENAME  [LOOKUP IN NEW UFD][NOT NECESSARY IF OLD AND NEW UFD ARE THE SAME]
        READ-PAUSE-WRITE [NEW] UFD
        [READ-PAUSE-WRITE OLD UFD][NOT NECESSARY IF OLD AND NEW UFD
         ARE THE SAME]
        READ-PAUSE-WRITE FILE RIB
```

RENAME A FILE BEING CREATED

```
ENTER   SAME AS CREATE IN OLD DIRECTORY
RENAME  [LOOKUP IN NEW UFD][NOT NECESSARY IF OLD AND NEW UFD ARE THE SAME]
        READ-PAUSE-WRITE [NEW] UFD
        READ-PAUSE-WRITE FILE RIB
```

DELETE A FILE PREVIOUSLY CLOSED

```
LOOKUP  SAME AS READ
RENAME  [READ-PAUSE-WRITE UFD]
        <<[[WRITE OTHER SAT BLOCK][NOT NEEDED IF NOT DIFFERENT FROM DISK]
        READ SAT BLOCK]]>[NOT NEEDED IF SAT TABLE ALREADY IN CORE]
        READ RIB>
```

DELETE A FILE BEING CREATED

```
LOOKUP  SAME AS READ
RENAME  <<[[WRITE OTHER SAT BLOCK][BUT NEEDED IF NOT DIFFERENT FROM DISK]
        READ SAT BLOCK]]>[NOT NEEDED IF SAT TABLE ALREADY IN CORE
```

4. UUOCON FLOW

```
ENTER (UUO LEVEL - UUOCON)

        PREPARE TO SUPPRESS INPUT CLOSE
        IF FILE OPEN FOR OUTPUT [OCLOSB=0], CLOSE OUTPUT SIDE OF FILE [OCLOSB←1]
        INITIALIZE IOS
        STORE UUO BITS
        ATTEMPT AN ENTER WITH SERVICE ROUTINE
        IF ENTER FAILS, EXIT
        FLAG FILE OPEN FOR OUTPUT [ENTRB←1,OCLOSB←0]
        STORE UUO BITS
        EXIT

LOOKUP (UUO LEVEL - UUOCON)

        PREPARE TO SUPPRESS OUTPUT CLOSE
        IF FILE OPEN FOR INPUT [ICLOSB=0], CLOSE INPUT SIDE OF FILE [ICLOSB←1]
        INITIALIZE IOS
        STORE UUO BITS
        ATTEMPT A LOOKUP WITH SERVICE ROUTINE
        IF LOOKUP FAILS, EXIT
        FLAG FILE OPEN FOR INPUT [LOOKB←1,ICLOSB←0]
        STORE UUO BITS
        EXIT

INPUT CLOSE (UUO LEVEL - UUOCON)

        ....

        FLAG SUCCESSFUL COMPLETION [LOOKB←0,ICLOSB←1]
        EXIT

OUTPUT CLOSE (UUO LEVEL - UUOCON)

        ...

        FLAG SUCCESSFUL COMPLETION [ENTRB←0,OCLOSB←1]
        EXIT

INIT (UUO LEVEL - UUOCON)

        FLAG SUCCESSFUL COMPLETION [INITB←1,ICLOSB←1,OCLOSB←1]
        (SET ICLOSB,OCLOSB PRETENDING AN EARLIER CLOSE TO PREVENT
        SUPERFLUOUS CALLS TO CLOSE AT LOOKUP, ENTER TIME)
```

5. FILSER FLOW

FLOW FOR LOOKUP

```
        IF ZERO FILE NAME, ERROR RETURN
        IF AN ENTER IN FORCE ON THIS USER CHANNEL, ERROR RETURN
        SETUP FILE STRUCTURE SEARCH SPECIFICATION FROM INIT PHYSICAL NAME[CALL SETSRC]
        SEARCH CORE AND DISKS FOR FILE NAME, SETUP CORE BLOCKS IF FOUND[CALL FNDFIL]
        IF ANY ERROR EXCEPT FILE NOT FOUND, ERROR RETURN TO USER(FNDFIL DID NOT CREATE ACCESS BLOCK)
        IF FOUND, TO FOUND
        IF DEVICE NAME WAS DSK WITHOUT A PROJ-PROG SPECIFIED
          IF THIS JOB WANTS PROJECT LIBRARY SEARCHED
            SEARCH PROJECT LIBRARY IN CORE AND DISK USING SYSTEM SEARCH LIST[CALL FNDFIL]
            IF FILE FOUND, TO FOUND
          END
          IF THIS JOB WANTS SYSTEM LIBRARY(1,3) SEARCHED
            SEARCH SYSTEM LIBRARY IN CORE AND DSIK USING SPECIAL SYS LIST[SYSSRC][CALLFNDFIL]
            IF FILE FOUND, TO FOUND
          END
        END
        ERROR RETURN TO USER (FNDFIL DID NOT CREATE ACCESS BLOCK)

FOUND:  SET DEVICE DATA BLOCK[DEVRSU,DEVFLT,DEVBLK,DEVREL,DEVRET,DEVFLR]
        SET FILE STATE TO READ[DEVSTS]
        COPY VALUES USER WANTS FROM ACCESS, NAME, UFD, AND PROJ-PROG BLOCKS TO USER
        IF USER WANTS MORE VALUES THAN IN CORE BLOCKS
          IF RIB IS NOT ALREADY IN A MONITOR BUFFER, READ RIB INTO MONITOR BUFFER
          SET DEVICE DATA BLOCK[DEVRSU,DEVFLT,DEVBLK,DEVREL,DEVRET,DEVRLC,DEVFLR]
        SET ACCESS BLOCK DATA[AC1PT1,AC1ALC,ACCWRT,AC1DTE,AC1PRT]
          COPY ALL ARGUMENTS USER WANTS FROM MONITOR BUFFER TO USER
        END
        IF A MONITOR BUFFER IS IN USE BY THIS JOB, RELEASE IT TO SYSTEM[IOSMBF]
        OK RETURN TO USER
```

FLOW FOR ENTER

```
        IF ZERO FILE NAME, ERROR RETURN
        IF LOOKUP IN FORCE ON THIS USER CHANNEL, TO UPDATE
        SETUP SEARCH SPECIFICATION ACCORDING TO INIT NAME[CALL SETSRC]
        SEARCH CORE AND DISK FOR FILE NAME[CALL FNDFIL]
        (IF FILE FOUND BY FNDFIL PROJ-PROG,UFD,NAME, AND ACCESS BLOCKS SETUP)
        IF ERROR OTHER THAN FILE NOT FOUND, ERROR RETURN TO USER(FNDFIL DID NOT CREATE BLOCK ACCESS)
        IF FILE NOT FOUND, TO CREATE
SUPSED: SET STATE FILE TO SUPERCEDE[DEVSTS](ACCESS TABLE ALREADY SET TO SUPERCEDE BY FIDFIL)
        IF THIS PROJ-PROG IN THIS STR IS OUT OF SPACE[UFBTAL],TO STRLOP
        TO UNILOP

CREATE: SET STATE OF FILE TO CREATE[DEVSTS](ACCESS TABLE ALREADY, SET TO CREATE BY FNDFIL)
STRLOP: IF ONLY ONE STR IN SEARCH SPECIFICATION
          IF THAT STR HAS NO ROOM[STRTAL],ERROR RETURN TO USER
          TO USESTR
        END
STILOP: DO FNDSTR, FOR EACH FILE STRUCTURE IN SEARCH SPECIFICATION
          IF THIS PROJ-PROG IN THIS STR HAS NO MORE ROOM[UFBTAL], TO FNDSTR
          IF THIS JOB DOES NOT WANT CREATES IN THIS STR[JBTUFB],TO FNDSTR
          TO USESTR
FNDSTR: CONTINUE
        ERROR RETURN TO USER(NO ROOM IN ANY STR)

USESTR: SET ACCESS BLOCK LOCATIONS TO THIS STR NUMBER[ACCFSN]
        IF DEVICE NAME SPECIFIES A PARTICULAR UNIT [DEVNAM]
          IF THAT UNIT HAS SOME SPACE[UNITAL],TO USEUNI
        END
UNILOP: DO FNDUNI, FOR ALL UNITS IN THIS STR[STRUNI-UNISTR]
          IF THIS UNIT IS THE MOST EMPTY SO FAR[UNITAL],REMEMBER IT
FNDUNI: CONTINUE
USEUNI: SET FIRST-POINTER-IN-FILE UNIT WITH STR IN ACCESS BLOCK [ACCUNI]
        SET FIRST RETREIVAL POINTER IN DEVICE DATA BLOCK TO THIS UNIT NUMBER[DEVPTI,DEVUNI]
        IF USER SPECIFYING ALLOCATION, TRY TO ALLOCATE N OR LESS BLOCKS
        IF CANNOT GET EXACTLY N, REMEMBER ERROR
        SEARCH FOR KONGRP OR LESS BLOCKS ON THIS UNIT
        IF DID NOT FIND ANY, SPACE IN ANY UNIT IN STR TO STRLOP
        SET SECOND POINTER IN DEVICE DATA BLOCK TO POINT TO THIS SPACE[DEVPT2]
        QUEUE FOR MONITOR BUFFER
        COPY USER ARGS INTO MONITOR BUFFER
        COPY THE FIRST UNIT AND FIRST RETREIVAL POINTER INTO MONITOR BUFFER
        WRITE FIRST RIB
        COPY AS MANY VALUES FROM MONITOR BUFFER AS USER WISHES
        GIVE UP MONITOR BUFFER
        IF ALLOCATION FAILURE, ERROR RETURN TO USER
        OK RETURN TO USER
```

```
UPDATE: IF FILE NAMES DO NOT MATCH[DEVFIL,EXT,PPBNAM=USER ARG],ERROR RETURN
*       (DO NOT CHECK IF THIS STR IS WRITE LOCKED FOR THIS USER[DEPWLK]
           SINCE HE MAY JUST WANT TO READ AND LOCK OUT OTHER WRITERS)
        CHECK PRIVILEGES TO AT LEAST APPEND (UPDATE HIGHER)[CALL CHKPRV]
        IF CANNOT APPEND, ERROR RETURN TO USER
        IF USER CHANGING ALLOCATION
*         IF THIS FILE STRUCTURE IS WRITE LOCKED FOR THIS USER, ERROR RETURN
          IF ALLOCATING [AC1ALC]
            CHECK PRIVILEGES TO ALLOCATE[CALL CHKPRV]
          OR IF DEALLOCATING [AC1ALC,ACCWRT]
            CHECK PRIVILEGE TO DEALLOCATE, [CALL CHKPRV]
          ELSE (MUST BE TRUNCATING, I.E., THROWING AWAY DATA BLOCKS)
            CHECK PRIVILEGES TO TRUNCATE [CALL CHKPRV]
          END
          IF CANNOT DO OPERATION, ERROR RETURN TO USER
*         NOSCHEDULE
*         IF FILE IS ALREADY BEING UPDATED [ACPUPD] OR RENAMED [ACPREN], SCHEDULE AND ERROR RETURN
*         SET FILE TO BEING UPDATED
*         SCHEDULE

          IF ALLOCATING
            SEARCH FOR K=N OR LESS BLOCKS ANYWHERE IN STR OR AT A PARTICULAR LOGICAL BLOCK
            IF NOT FIND ALL K=N BLOCKS, REMEMBER ERROR
            IF COULD NOT FIND ANY BLOCKS, ERROR RETURN TO USER
            ADD ONE POINTER TO DEVICE DATA BLOCK
            IF RUN OUT OF ROOM IN DEVICE DATA BLOCK [DEVRET=DEVRBN(F)]
              READ RIB INTO MONITOR BUFFER
              APPEND CORE POINTERS TO POINTERS IN RIB
              IF RUN OUT OF ROOM IN RIB [DEVRSU], ERROR RETURN TO USER (TEMPORARY IN LEVEL D)
              WRITE MONITOR BUFFER BACK ONTO FIRST RIB ONLY(SECOND RIB ON CLOSE)
              GIVE UP MONITOR BUFFER
            END
          OR IF DEALLOCATING OR TRUNCATING
            DO A USETI TO POSITION THE FILE AT PLACE TO START RETURNING SPACE
            READ THE RIB INTO MONITOR BUFFER
            MARK THE FILE IN CORE AS SHORTER [ACCWRT AC1ALC]
            MARK THE RIB AS NEW SHORTER EOF IN DEVICE DATA BLOCK AND MONITOR BUFFER
            REWRITE THE MONITOR BUFFER ONTO RIB ON DISK (SECOND RIB ON CLOSE)
            DO DELGRP, FOR ALL GROUPS BEING DELETED INCLUDING PART OF CURRENT ONE
              RETURNS SPACE IN CORE BY CLEARING CLUSTER BITS
              WRITE SAT BLOCKS ON DISK ONLY IF NEEDED IN ORDER TO READ OTHERS IN
DELGRP:     CONTINUE
            GIVE UP MONITOR BUFFER
          END
*       ELSE (USER NOT CHANGING ALLOCATION)
*         NOSCHEDULE
*         IF FILE IS ALREADY BEING UPDATED,SUPERCEDED, OR CREATED OR RENAMED, SCHEDULE AND ERROR RETURN
*         SET FILE TO BEING UPDATED
*         SCHEDULE
        END
        OK RETURN
```

ARGUMENTS TO FNDFIL

A, SEARCH SPECIFICATION (LIST OF STR TO BE SEARCHED IN ORDER)
B, JOB NUMBER (TO GET SOFTWARE WRITE-LOCK
C, FILE NAME
D, EXTENSION
E, PROJECT-PROGRAM NUMBER
F, LOOKUP, ENTER, CHANGE NAME
G, ADDRESS OF DEVICE DATA BLOCK

VALUES FROM FNDFIL
A, OK RETURN (CORE BLOCKS CREATED AND INITIALIZED)
IF FOUND (LOOKUP AND ENTER)
NOT FOUND(ENTER, CHANGE NAME)
B, ERROR RETURN (CORE BLOCKS NOT NECESSARILY SET UP, CALLER DOES NOT HAVE TO DELETE)
1. CAN'T FIND FILE (ON LOOKUP)
2. USER CAN'T CREATE IN THIS UFD (ENTER)
3. USER CAN'T READ, UPDATE, SUPERSEDE THIS FILE BECAUSE OF ITS STATE
4. PRIVILEGES DO NOT ALLOW LOOKUP, ENTER, CHANGE NAME
5. FOUND FILE ON CHANGE NAME
C, SETS UP CORE BLOCKS IF OK RETURN
1. PROJECT PROGRAMMER NUMBER WITHIN SYSTEM BLOCK[PPB]
2. USER FILE DIRECTORY WITH FILE STRUCTURE BLOCK[NMB]
4. ACCESS BLOCK WITHIN NAME WITHIN PROJ PROG WITHIN SYSTEM BLOCK [ACC]
5. SECOND HALF OF ACCESS BLOCK [AKB]

OPERATION OF FNDFIL
A, FNDFIL IS ONLY CREATOR AND DELETER OF
CORE BLOCKS (ACCESS, NAME, PROJ-PROG, UFD BLOCKS)
B, SO FNDFIL DOES NOT CREATE CORE BLOCKS FOR LOOKUP NOT FOUND
C, BUT DOES FOR ENTER NOT FOUND
D, FNDFIL ALSO CHECKS PRIVILEGES AND DELETES CORE BLOCKS IF
IT WISHED ON ERROR DEVACC=Ø IS FLAG THAT ACCESS BLOCK
IS DELETED ON ERROR
E, FNDFIL SETS STATE OF ACCESS BLOCK ON INCREASES READ COUNT, ETC SO
F, FNDFIL INITIALIZES ALL LOCATIONS IN ANY CORE BLOCKS IT CREATES,
G, FNDFIL RETURNS ERROR CODES ON ALL ERROR RETURNS; THESE ARE PASSED
BACK TO USER
H, FNDFIL LEAVES RIB IN MONITOR BUFFER ONLY IF IT HAD TO READ IT [IOSMBF SET]

FLOW FOR FNDFIL CALLED BY LOOKUP, SEARCH, ENTER, RENAME

AN ACCESS BLOCK CAN BE IN EXACTLY ONE OF THE FOLLOWING STATES WHEN ENCOUNTERED BY FNDFIL:
BECAUSE OF THE RESTRICTIONS ON ENTER, MOST OF THE TIME THERE CAN BE AT MOST ONE ACCESS BLOCK WITH A GIVEN FILE NAME (AND FILE STRUCTURE, PROJ,PROG AND EXTENSION), THE EXCEPTIONS ARE THAT R AND S MAY OCCUR SIMULTANEOUSLY IN TWO DIFFERENT ACCESS BLOCKS, ALSO AN ARBITRARY NUMBER OF EXTRA ENTRIES MAY BE MARKED FOR DELETION (K) BECAUSE THEY HAVE BEEN SUPERCEDED, BUT SOME SLOW READERS ARE STILL READING OLDER VERSIONS, THESE ACCESS BLOCKS HOWEVER ARE NOT SEEN BY FNDFIL SINCE THEY ARE REMOVED FROM THE RING OF ACCESS BLOCKS FOR THE NAME BLOCK WHEN THEY ARE MARKED FOR DELETION, FINALLY IF THE PROJECT-PROGRAMMER NUMBER HAS MORE THAN ONE FILE BY THE SAME NAME IN DIFFERENT FILE STRUCTURES, THERE WILL BE A SEPARATE ACCESS BLOCK FOR EACH

AN ACCESS ENTRY	LOOKUP SEARCH	ENTER (CREATE, SUPERCEDE)	RENAME (SEARCH FOR NEW NAME IF DIFF.)
N-NON EXISTANT FILE	ERROR	CREATE	OK
D-DORMANT	USE	USE REMEMBER	ERROR
R-1 OR MORE READERS(MAYBE ANOTHER S)	USE REMEMBER FOR POSSIBLE ERROR	KEEP LOOKING(S)	ERROR
C-BEING CREATED	KEEP LOADING	ERROR	ERROR
S-SUPERSEDING (MAYBE ANOTHER R)	KEEP LOOKING(R)	ERROR	ERROR
UR-BEING UPDATED AND READ	USE	ERROR	ERROR
K-MARKED FOR DELETION(IE SUPERSEDED)	(NOT SEEN BY FNDFIL)		

IN ORDER TO SAVE DISK SEEKS, FNDFIL REMEMBERS RECENT FILE NAMES ALONG WITH THE PROJECT PROGRAMMER NUMBERS AND FILE STRUCTURES IN WHICH THEY WERE FOUND.

THIS ASSOCIATIVE CORE INFORMATION HAS A RATHER COMPLEX STRUCTURE IN ORDER TO SPEED UP SEARCHES THROUGH IT. THE LISTS ARE AS FOLLOWS:

THE SYSTEM HAS A LIST OF RECENTLY ACCESSED PROJECT-PROGRAMMER NUMBERS [SYSPPB-PPBSYS].
EACH PROJECT PROGRAMMER NUMBER BLOCK [PPB] HAS TWO LISTS:

A. A UFD BLOCK [UFB] FOR EVERY FILE STRUCTURE IN WHICH THE PROJECT-PROGRAMMER NUMBER HAS A UFD [PPBUFB-UFBPPB]
B. AND A NAME BLOCK [NME] FOR EVERY NAME WITH THIS PROJECT PROGRAMMER NUMBER (INDEPENDENT OF FILE STRUCTURE)[PPBNME-NMEPPB]

EACH NAME BLOCK HAS AN ACCESS BLOCK[ACC], LIST WITH AN ENTRY FOR EVERY ACTIVE FILE WITH THAT PROJECT PROGRAMMER NUMBER AND NAME. [NMBACC-ACCNMB]

IN ORDER TO ELIMINATE FRUITLESS SEARCHES,
THE PROJECT PROGRAMMER NUMBER BLOCK HAS A BIT FOR EACH
OF THE UP TO 14 ON-LINE FILE STRUCTURES WHICH
SAY WHETHER UFD EXISTS FOR THE PROJECT-PROGRAMMER NUMBER
OR NOT OR WHETHER WE DON'T KNOW YET, SIMILARLY THE
NAME BLOCK [NMB] HAS A BIT FOR EACH OF UP TO 14
ON-LINE FILE STRUCTURES WHICH SAY WHETHER FILE EXISTS
FOR THE PROJECT PROGRAMMER NUMBER OR NOT OR WHETHER WE DON'T
KNOW YET.

FLOW FOR FNDFIL ABBREVIATED VERSION

```
FNDFIL: IF PROJ-PROG BLOCK NOT IN CORE FOR SYSTEM, CREATE ONE AND APPEND TO LIST FOR SYSTEM[PPB]
        IF NAME BLOCK NOT IN CORE FOR PROJ-PROG, CREATE ONE AND APPEND TO LIST FOR PROG-PROGENMB]
FNDFL1: DO SCNSTR FOR EACH STRUCTURE IN ORDER OF SEARCH LIST SPECIFICATIONS
          IF EXISTENCE OF FILE IN THIS STRUCTURE IS DEFINITELY NO, TO SCNSTR
          IF EXISTENCE OF FILE IN THIS STRUCTURE IS UNKNOWN, TO CFPCHK
          DO SCNACC FOR ALL ACCESS BLOCKS IN RING FOR THIS NAME BLOCK (INDEPENDENT OF STRUCTURE)
            IF ACCESS BLOCK NOT FOR THIS STRUCTURE, TO SCNACC
            IF LOOKUP (INCLUDING EXECUTE)
              IF DORMANT OR BEING READ AND/OR UPDATED
                IF ACCESS PRIVILEGES DO NOT ALLOW DESIRED ACTION, ERROR RETURN
                INCREMENT READ COUNT
                IF DORMANT, REMOVE FROM DORMANT LIST
                OK RETURN
              END (DON'T QUIT ON ACCESS BLOCK BEING CREATED SINCE FILE MAY EXIST IN LATER STRUCTURE)
            OR IF ENTER (CANNOT BE UPDATE FOR THIS USER)
              IF BEING CREATED, SUPERSEDING OR BEING UPDATED, ERROR RETURN
              IF ACCESS PRIVILEGES DO NOT ALLOW SUPERSEDE, ERROR RETURN
              IF DORMANT, SET STATE TO SUPERSEDING, REMOVE FROM DORMANT LIST, AND OK RETURN
              REMEMBER THIS ACCESS ENTRY SO FILE CAN BE MARKED FOR DELETION OR SUPERSEDING CLOSE(FILE BEING READ)
            ELSE (MUST BE CHANGING NAME)
              ERROR RETURN (SINCE ONLY EXISTING NAMES MARKED FOR DELETION ARE LEGAL AND THOSE CANNOT BE FOUND)
            END
SCNACC:   CONTINUE
          (MIGHT HAVE FOUND ACCESS BLOCK CREATING OR SUPERSEDING ON LOOKUP, OR READING ON ENTER)
CFPCHK:   IF COMPRESSED POINTER TO RETRIEVAL BLOCK IN THIS NAME BLOCK FOR THIS STRUCTURE, TO FILRRB
UFBLUP:   IF CAN FIND A UFB BLOCK FOR THIS STRUCTURE, TO SCNUFD
          IF UFD NAME NOT FOUND IN SEARCH OF MFD FILE, TO SCNSTR
CREUFB:   READ UFB RETRIEVAL BLOCK, CREATE UFB BLOCK AND APPEND TO LIST FOR THIS PROJ-PROG(IF ONE DID NOT SNEAK IN)
SCNUFD:   IF FILE NAME FOUND IN SEARCH OF UFD FILE, TO FILRRB
          MARK FILE NAME AS DEFINITELY NOT IN THIS STRUCTURE
SCNSTR: CONTINUE
        (FILE NOT FOUND OR FOUND READING ON ENTER)
        IF LOOKUP, MARK NAME BLOCK AS GRABBABLE AND ERROR RETURN
        IF ONE DID SNEAK IN, BACK TO FNDFL1(REPEAT FNDFIL OPERATION)
        IF CHANGING NAME OR DIRECTORY FUNCTION (RENAME UUO)
*         IF USER NOT PRIVILEGED TO DO THESE, ERROR RETURN
*         CHANGE NAME AND DIRECTORY IN ACCESS BLOCK
          OK RETURN
        END
        CREATE AN ACCESS BLOCK AND APPEND TO RING FOR THIS NAME BLOCK(IF ONE DID NOT SNEAK IN)
        IF PREVIOUS ACCESS BLOCK WAS SEEN(READING ON ENTER), SET STATE TO SUPERSEDE AND OK RETURN
        IF NOT PRIVILEGED TO CREATE FILE IN UFD, ADD ACCESS BLOCK TO DORMANT LIST AND ERROR RETURN
        SET STATE CODE TO CREATE AND OK RETURN
```

```
        (FILE EXISTS BUT ACCESS BLOCK NOT IN CORE)
FILRRB: IF CHANGING NAME, ERROR RETURN (NEW NAME ALREADY EXISTS)
        READ FILE RETRIEVAL BLOCK, CREATE ACCESS BLOCK AND APPEND TO RING FOR THIS NAME(IF ONE DID NOT SNEAK IN)
        IF ONE DI SNEAK IN, BACK TO FNDFL1(REPEAT FNDFIL OPERATION)
        MARK FILE NAME AS DEFINITELY IN THIS STRUCTURE
        IF PRIVILEGES DO NOT ALLOW INTENDED ACTION(MUST BE EXECUTE, READ OR SUPERSEDE)
          FLUSH ACCESS BLOCK AND ERROR RETURN
        END
        IF LOOKUP(READ EXECUTE) INCREMENT READ COUNT AND OK RETURN
        SET STATE TO SUPERSEDING AND OK RETURN
```

DETAILED FNDFIL FLOW

```
FNDFIL: QUEUE FOR CB RESOURCE
PPBLOP: DO SCNPPB, FOR ALL PROJ-PROG BLOCKS[PPB] IN SYSTEM[SYSPPB-PPBSYS]
          IF FIND MATCHING PROJ-PROG NUMBER, TO NMBLUP
SCNPPB: CONTINUE
FNDPPB: CREATE PROJ-PROG BLOCK
        STORE PROJECT PROGRAMMER NUMBER [PPBNAM]
        MARK PPB AS NOT LOGGED IN [PPBNLG]
        MARK THAT PROJ-PROG MAYBE IN EACH STR IN SYSTEM[PPBKNO=Ø]
        MARK THAT PROJ-PROG DOES NOT APPEAR IN ANY STR(EVEN THROUGH IT WAY)[PPBYES=Ø]
        MARK THAT NO NAME OR UFD BLOCK LISTS [PPBUFB,PPBNMB=Ø]
        APPEND PROJECT PROGRAMMER NUMBER BLOCK TO END OF PROJ-PROG LIST FOR SYSTEM [PPBSYS]

NMBLUP: DO SCNNMB, FOR ALL FILE NAMES IN CORE IN THIS PROJ-PROG (NO MATTER WHAT STR)[PPBNMB-NMBPPB]
          IF FILE NAMES AND EXTENSION MATCH, TO STRLOP
SCNNMB: CONTINUE
CRENMB: CREATE A FILE NAME BLOCK [NMB] AND ADD TO END OF THIS PROJ-PROG LIST [PPBNMB]
        MARK IN FILE NAME BLOCK IS NOT GRABBABLE BY FNDFIL[NMPGRB=Ø]
        MARK THAT FILE NAME IS DEFINITELY NOT IN ANY STR IN WHICH PROG-PROG'S IS DEFINITELY NOT
        MARK THAT FILE NAME MAYBE IN EACH STR IN SYSTEM IN WHICH PROJ-PROG DEFINITELY YES
        MARK THAT FILE NAME DOES NOT APPEAR IN ANY STR(EVEN THROUGH IT MAY)[NMBYES=Ø]
        APPEND NAME BLOCK TO END OF NAME LIST FOR THIS PROJ PROG [NMBPPB]
```

```
STRLUP: DO SCNSTR, FOR EACH STR IN SEARCH SPECIFICATION IN ORDER [STRTAB]
          IF EXISTENCE OF FILE IN THIS STR IS DEFINITLEY NO [NMBKNO,NMBYES[, TO SCNSTR
          IF EXISTENCE OF FILE IN THIS STR IS MAYBE, TO CFPCHK(CAN'T BE AN ACCESS BLOCK)
          QUEUE FOR CORE BLOCK RESOURCE
          DO SCNACC,FOR ALL ACCESS BLOCKS IN THIS NAME/PROJ-PROG LIST (INDEP OF STR)
            IF THIS ACCESS ENTRY IS NOT FOR THIS STR, TO SCNACC
            IF LOOKUP (INCL EXECUTE)
              IF DORMANT OR BEING READ AND/OR UPDATED
                IF ACCESS PRIVILEGES, DO NOT ALLOW DESIRED ACTION, RELEASE CB RESOURCE, ERROR RETURN
                INCREMENT READ COUNT
                IF DORMANT, REMOVE FROM DORMANT LIST AND MARK NAME BLOCK AS NOT GRABBABLE[NMBGRB]
              RELEASE CB RESOURCE AND OK RETURN
              END
              (KEEP LOOKING IF ACCESS BLOCK WAS SUPERSEDING OR BEING CREATED)
            OR IF ENTER (CANNOT BE UPDATE FOR THEN USER)
              IF BEING CREATED, SUPERSEDING, OR BEING UPDATED, RELEASE CB RESOURCE AND ERR RET
              IF ACCESS PRIVILEGES DO NOT ALLOW THIS FILE TO BE SUPERSEDED, RELEASE CB AND ERR RET
              IF DORMANT
                REMOVE ACCESS BLOCK FROM SYSTEM DORMANT LIST [ACCDOR]
                ZERO DORMANT LIST POINTERS AS FLAG [ACCDOR]
                SET STATE TO SUPERSEDING
                RELEASE SCAN LIST RESOURCE [CBREQ]
                OK RETURN
              END
              REMEMBER THIS STR SO FILE CAN BE MARKED FOR DELETION ON SUPERSEDING CLOSE [DEVFSN]
              (FILE ALREADY BEING READ, KEEP SCANNING STRS IN CASE A LATER CREATE, SUPERSEDING, UPDATE)
            ELSE (MUST BE CHANGING NAME)
              RELEASE SCAN LIST RESOURCE [CBREQ]
              ERROR RETURN (SINCE ILLEGAL TO CHANGE A NAME TO AN EXISTING FILE, UNLESS MARKED FOR DELETION)
            END

SCNACC:   CONTINUE
          (COULD HAVE FOUND CREATING OR SUPERSEDING ON LOOKUP, READING ON ENTER)
CFPCHK:   IF STR NUMBER IN NAME BLOCK IS NOT THIS STR [NMBFSN], TO UFBLOP
          GET COMPRESSED VFD POINTER TO FILE RIB [NMBCFP]
          RELEASE CB RESOURCE
          TO FILRRB (SAVED A SEARCH OF UFD SINCE KNOW WHERE FILE RIB IS)

UFBLOP:   DO SCNUFB, FOR EACH UFB BLOCK WITH THIS PROJ-PROG[PPBUFB-UFBPPB]
            IF FIND UFD BLOCK WITH THIS STR NUMBER [UFBFSN], TO SCNUFD (SAVED MFD SEARCH)
SCNUFB:   CONTINUE
```

```
          RELEASE CORE BLOCK RESOURCE
          QUEUE FOR MONITOR BUFFER
          IF DIRECTORY NAME (PROJ-PROG) IS REALLY MFD(1,1), TO UFDLOP
MFDLOP:   DO REDMFD, FOR EACH U DISK BLOCK IN MFD IN THIS STR [STRPT1]
            READ MFD DISK BLOCK INTO MONITOR BUFFER
            SCAN MFD BLOCK FOR UFD NAME
            IF UFD NAME FOUND IN MFD BLOCK, TO UFDRIB
            IF RUNOUT OF MFD RETRIEVAL POINTERS, READ MFDRIB INTO SAME MONITOR BUFFER
REDMFD:   CONTINUE
          RELEASE MONITOR BUFFER
          MARK IN PROJ-PROG BLOCK THAT UFD DOES NOT EXIST IN THIS STR [PPBRNO,PPBYES]
          MARK IN FILE NAME BLOCK THAT THIS FILE NAME DOES NOT EXIST IN THIS STR
          TO SCNSTR

UFDRIB:   READ UFD RIB INTO MONITOR BUFFER
          IF ERROR WHILE READING RIB, ERROR RETURN
*         COMPARE FILE NAME, EXT, AND PROJECT PROGRAMMER NO, STORED IN RIB
*         IF DOES NOT AGREE WITH EXPECTED, ERROR RETURN

          QUEUE FOR CB RESOURCE
          IF A UFB BLOCK SNUCK IN FOR THIS PROJ-PROG, TO STRLUP (REPEAT FNDFIL)
          CREATE A UFB BLOCK FOR THIS FILE STRUCTURE AND APPEND TO LIST FOR PROG-PROG
          MARK FILE STRUCTURE NUMBER IN UFB BLOCK [UFBFSN]
          STORE FIRST UNIT NUMBER WITHIN STR OF UFD [UFBUN1←C(C(RIBFIR))]
          STORE FIRST RETRIEVAL POINTER TO UFD IN UFB BLOCK [UFBPT1,UFB1PT]
          STORE NUMBER OF FREE BLOCKS LEFT IN QUOTA FOR THIS UFD [UFBTAL]
          STORE NUMBER OF OVERDRAWN BLOCKS ALLOWED FOR THIS UFD [UFBOVR]
          STORE ACCESS PRIVILEGES FOR UFD [UFBPRV]
          MARK IN PROJ-PROG DATA BLOCK THAT THIS UFD EXISTS IN THIS STR
```

```
SCNUFD:   IF THIS JOB DOES NOT ALREADY LEAVE A MONITOR BUFFER[MNBFPT], QUEUE FOR ONE
          RELEASE CB RESOURCE
UFDLOP:   DO REDUFD FOR ALL DISK BLOCKS IN UFD IN THIS STR[UFBPT1]
            READ UFD DISK BLOCK INTO MONITOR BUFFER
            SCAN UFD BLOCK LOOKING FOR FILE NAME AND EXTENSION
            IF FILE NAME FOUND IN UFD BLOCK, TO FILRIB
            IF RUN OUT OF UFD RETRIEVAL POINTERS, READ UFD RIB INTO MONITOR BUFFER
REDUFD:   CONTINUE
          RELEASE MONITOR BUFFER
          QUEUE FOR CB RESORNCE [CBREQ]
          MARK FILE AS DEFINITELY NOT IN THIS STR[NMBKNO,NMBYES]
SCNSTR: CONTINUE
        (FILE NOT FOUND, OR FOUND READING ON ENTER)
        IF NO UFDS FOUND FOR THIS SEARCH LIST, ERROR RETURN (UFD DOES NOT EXIST)

        IF LOOKUP (OR EXECUTE)
          IF NAME BLOCK HAS NO ACCESS BLOCKS, MARK NAME BLOCK AS GRABBABLE
          RELEASE CB RESOURCE, ERROR RETURN
        END
        IF AN ACCESS BLOCK HAS SNUCK IN, BACK TO STRLUP (REPEAT FNDFIL)
        IF CHANGING NAME OR DIRECTORY FUNCTION (RENAME UUO)
*         IF FILE IS ALREADY BEING RENAMED [ACPREN IN ACYSTS], ERROR RETURN
          IF FILE IS ALREADY BEING CREATED, SUPRESSING, OR UPDATED BY SOME USER CHANNEL
            IF NOT THIS USER CHANNEL [ENTRB], ERROR RETURN
          IF OLD FILE NAME [NMBNAM[ACCNMB(RING)] IS DIFFERENT FROM NEW NAME [DEVFIL]
            IF THIS USER CANNOT CHANGE NAMED [FNCCNM], ERROR RETURN [CALL CHKPRV]
          END
          IF OLD DIRECTORY [PPBNAM[ACCPPB]] IS DIFFERENT FROM NEW DIRECTORY [FNDFIL AC ARG]
            IF THIS USER CANNOT CREATE IN THIS DIRECTORY [FNCCRT], ERROR RETURN [CALL CHKPRV]
            IF THIS CHANNEL IS NOT AN UPDATER [ENTRB], ERROR RETURN
            IF NEW DIRECTORY DOES NOT HAVE ENOUGH FREE BLOCKS [UFBTAL] TO TAKE A FILE [ACCALC], ERROR RETURN
            INCREMENT OLD DIRECTORY FREE BLOCK COUNT [UFBTAL] BY ALLOCATED SIZE OF FILE [AKBALC]
            DECREMENT NEW DIRECTORY FREE BLOCK COUNT [UFBTAL] BY ALLOCATED SIZE OF FILE [AKBALC]
          END
          SET POINTER [ACCPPB] IN OLD ACCESS BLOCK [DEVACC] TO POINT TO THIS PROJ PROG BLOCK (IN CASE CHANGING DIR.)
          UNLINK OLD ACCESS BLOCK [DEVACC] FROM NAME RING[ACCNMB]
          FLAG OLD NAME BLOCK AS GRABBABLE [NMBGRB]
          FLAG OLD FILE NAME AS DEFINITELY NOT PRESENT IN THIS STR [NUMKNO=1,NMBYES=0]
          LINK OLD ACCESS BLOCK [ACCNMB] TO NEW NAME RING [DEVFIL](IN CASE CHANGING NAME)
          FLAG NEW NAME BLOCK AS NOT GRABBABLE (IN CASE IT WAS)
          FLAG NEW FILE NAME AS DEFINITELY PRESENT IN SAME STR [NMBKNO=1,NMBYES=1]
          FLAG THIS FILE AS IN PROCESS OF BEING RENAMED [ACPREN ORED INTO ACYSTS]
          (FILE MAY OR MAY NOT ALREADY BEING CREATED [ACPCRE], SUPERSEDING [ACPSUP], OR BEING UPDATED [ACPUPD] [ACYSTS])
          (BY THIS USER CHANNEL - BUT NEVER BY ANY OTHER USER CHANNEL)
          RELEASE CORE BLOCK RESOURCE [CBREQ] AND OK RETURN
        END
*       IF AN ACCESS BLOCK HAS SNUCR IN AND IS CREATE, SUPERSEDING, OR UPDATE, OR RENAME, ERROR RETURN
        CREATE AN ACCESS BLOCK AND APPEND TO RING FOR THIS NAME BLOCK
        IF FOUND AN ACCESS BLOCK BEING READ ON ENTER, SET STATE TO SUPERSEDE, RELEASE CB, OK RETURN
```

```
  (ACCESS PRIVILEGES ALREADY CHECKED INSIDE LOOP SINCE ACCESS BLOCK WAS FOUND)
IF USER NOT PRIVILEGED TO CREATE FILES IN UFD
  FLUSH ACCESS BLOCK AND ADD TO SYSTEM FREE LIST AND CLEAR POINTER TO IT IN DDB[DEVACC]
  RELEASE CB RESOURCE AND ERROR RETURN
END
SET STATE OF ACCESS BLOCK TO CREATE, RELEASE CB RESOURCE AND OK RETURN
```

```
FILRRB: (FILE EXISTS, BUT ACCESS BLOCK NOT IN CORE, USER DOES NOT HAVE CB RESOURCE)
        IF CHANGING NAME, ERROR RETURN (NEW NAME ALREADY EXISTS)
        IF DONT ALREADY HAVE MONITOR BUFFER[IOSMBF], QUEUE FOR IT
        READ FILE RIB INTO MONITOR BUFFER (LEAVE IT THERE FOR RETURN TO CALLER)
        IF ERROR WHILE READING RIB, RECORD SYSTEM ERROR, RELEASE MB AND ERROR RET. TO USER
        QUEUE FOR CB RESOURCE
        IF AN ACCESS BLOCK SNUCK IN, BACK TO STRLUP (REPEAT FNDFIL)
        CREATE AN ACCESS BLOCK AND APPEND TO RING FOR THIS NAME BLOCK
        MARK THAT THIS FILE DEFINITELY EXISTS IN THIS STRUCTURE[NMBKNO,NMBYES]
          MARK FILE STRUCTURE NUMBER IN ACCESS BLOCK[ACCFSN]
          COPY FIRST UNIT NUMBER WITHIN STR OF FILE IN ACCESS BLOCK[AC1UN1←C)C)RIBFIR))]
          COPY FIRST RETRIEVAL POINTER TO FILE IN ACCESS BLOCK[AC1PT1←C(C(RIBFIR)+1)]
          COPY NUMBER OF BLOCKS WRITTEN IN ACCESS BLOCK[ACCWRT←RIBSIZ/128]
          COPY NUMBER OF BLOCKS ALLOCATED IN ACCESS BLOCK[AC1ALC←RIBALC/128+1]
          COPY ACCESS PRIVILEGES, MODE, CREATION TIME, CREATION DATE[AC1ATT←RIBATT]
          COPY ACCESS DATE (DAY) IN ACCESS BLOCK[AC1ACD←RIBDTE]

          STORE FIRST LOGICAL BLOCK WITHIN UNIT TO BE READ OR WRITTEN(ADD 1 FOR RIB) [DEVBLK]
          COPY FIRST BUNCH [PTRLEN] OF RETRIEVAL POINTERS INTO DEVICE DATA BLOCK[DEVRB1←DEVRBN]
          SET GROUP INDEX OF FIRST GROUP POINTER IN CORE TO Ø[DEVRLC]
          SET CORE ADDRESS OF CURRENT POINTER TO CORE ADDRESS OF FIRST POINTER[DEVRET←DEVRB1(F)]
          SET RELATIVE BLOCK NUMBER TO READ OR WRITTEN (ADD FOR RIB)[DEVBLK]
          SET RELATIVE BLOCK NUMBER TO READ OR WRITE ON NEXT UUO TO 1 (SKIP RIB)[DEVREL]
          STORE CORE ADDRESS OF ACCESS BLOCK IN DEVICE DATA BLOCK[DEVACC]
          STORE NUMBER OF BLOCKS LEFT IN FIRST GROUP (SUBTRACT 1 FOR RIB)[DEVLFT]
          SET UPWARD POINTER TO UFD BLOCK IN DEVICE DATA BLOCK[DEVUFB]
          SET UPWARD POINTER TO UNIT BLOCK IN DEVICE DATA BLOCK[DEVUNI]
          IF PRIVILEGES DO NO ALLOW INTENDED ACTION, (MUST BE EXECUTE, READ OR SUPERSEDE)
            FLUSH ACCESS BLOCK, ADD IT TO SYSTEM FREE LIST, AND CLEAR POINTER TO IT IN DDB[DEVACC]
            IF THIS IS ONLY ACCESS BLOCK FOR NAME BLOCK, MARK NAME BLOCK OR GRABBABLE
          END
          IF LOOKUP (INCL EXECUTE) SET READ TO 1 [ACYRDC], RELEASE CB RESOURCE, OK RETURN
          SET SLATE TO SUPERSEDING, RELEASE CB RESOURCE, OK RETURN
```

```
FLOW FOR CHKPRV SUBROUTINE - CHECK ACCESS PRIVILEGES FOR THIS USER

        IF MOST POWERFUL OPERATION THIS USER CAN PERFORM ON THIS FILE HAS BEEN STORED, TO CHECK[DEVFNC]
        IF THIS USER IS ALSO THE OWNER OF THE FILE[JBTPPB=DEVACC(ACCPPB)]
          IF USER IS TRYING TO "CHANGE PROTECTION"[FNCCPR], OK RETURN (DO NOT STORE HIGHEST FUNCTION)
          GET FILE PROTECTION AGAINST THE OWNER[AKBPRV]
        OR IF THIS USER IS A MEMBER OF THE OWNER'S PROJECT
          GET FILE PROTECTION AGAINST THE REST OF THE OWNER'S PROJECT[AKBPRV]
        ELSE
          GET FILE PROTECTION AGAINST ALL PROJECTS EXCEPT THE OWNER'S
        END
        IF THIS FILE IS A DIRECTORY FILE (MFD OR UFD)[AKBDIR=1]
          IF THIS USER JOB IS PRIVILEGED FILE SYSTEM CUSP [JBTSTS[JACCT]=1], OK RETURN (DO NOT STORE FUNCTION)
          IF USER IS TRYING TO DO MORE THAN JUST "READ" DIRECTORY AS A FILE, ERROR RETURN (DO NOT STORE)
          IF THIS USER IS THE OWNER OF THE FILE (UFD,MFD)
            IF OWNER CANNOT READ DIRECTORY AS A FILE[UFBARD=0], ERROR RETURN
          OR IF THIS USER IS A MEMBER OF THE PROJECT IN WHICH THE FILE(UFD,MFD) BELONGS
            IF REST OF OWNER'S PROJECT CANNOT READ DIRECTORY AS A FILE[UFBPRD=0], ERROR RETURN
          ELSE (THIS USER NOT IN OWNER'S PROJECT)
            IF UNIVERSE CANNOT READ DIRECTORY AS A FILE[UFBURD=0], ERROR RETURN
          END
          STORE "READ" FUNCTION AS MOST POWERFUL THAT THIS USER CAN PERFORM ON THIS FILE(UFD,MFD)[DEVFNC]
          TO CHECK
        END
        (FILE IS A DATA FILE RATHER THAN A DIRECTORY FILE)
        IF THIS USER IS ALSO THE OWNER OF THE FILE[JBTPPB==DEVACC(ACCPPB)]
          GET DIRECTORY PROTECTION AGAINST THE OWNER
        OR IF THIS USER IS A NUMBER OF THE OWNER'S PROJECT
          GET DIRECTORY PROTECTION AGAINST THE OWNER'S PROJECT
        ELSE
          GET DIRECTORY PROTECTION AGAINST ALL PROJECTS EXCEPT THE OWNER'S
        END
        IF DIRECTORY PROTECTION (AGAINST OWNER, PROJECT, UNIVERSE) SAYS THIS USER CANNOT DO LOOKUP, TO CHKERR
        IF USER IS TRYING TO "CREATE" (RENAME TO NEW DIRECTORY OR ENTER)
            IF DIRECTORY PROTECTION (OWNER, PROJECT, OR UNIVERSE) SAYS THIS USER CAN "CREATE" IN THIS DIRECTORY, OK RETURN
          TO CHKERR
        END
        CONVERT FILE PROTECTION (AGAINST OWNER, PROJECT, OR UNIVERSE) TO MOST POWERFUL FUNCTION ALLOWED
TORE:   STORE HIGHEST FUNCTION THIS USER IS ALLOWED TO PERFORM
HECK:   IF FUNCTION USER IS TRYING TO PERFORM IS NO MORE POWERFUL THAN HIGHEST ALLOWED[DEVFNC], OK RETURN
CHKERR: IF DIRECTORY NAME (PROJ-PROG) OF FILE IS SAME AS DIRECTORY FORM WHICH PROGRAM CAME [?]
          IF FILE NAME OF PROGRAM[JBTNAM] IS SAME AS FILE NAME OF FILE[DEVFIL OR NMBNAM?]
            IF USER HAS NOT MEDDLED WITH PROGRAM[?] NOR CHANGED PROGRAM NAME[?], OK RETURN
          END
        END
        ERROR RETURN
```

```
FILE PROTECTION CODES[AKBPTR]   FUNCTION CODES[DEVFNC]

Ø NO ACCESS PRIVILEGES           Ø        FUNCTION NOT STORED YET
1 PRTEXC EXECUTE ONLY            1 FNCEXC EXECUTE ONLY
2 PRTRED READ                    2 FNCRED READ
3 PRTAPP APPEND                  3 FNCALL ALLOCATE
3                                4 FNCDLL DEALLOCATE
3                                5 FNCAPP APPEND
4 PRTUPD UPDATE                  6 FNCUPD UPDATE
       (NO FILE YET)             7 FNCCRT CREATE
5 PRTWRT WRITE                  1Ø FNCSUP SUPERSEDE
5                               11 FNCTRN TRUNCATE
6 PRTREN RENAME                 12 FNCCAT CHANGE ATTRIBUTE(EXCEPT PROTECTION NAME)
6                               13 FNCDEL DELETE FILE
6                               14 FNCCNM CHANGE NAME
7 PRTCPR CHANGE PROTECTION      15 FNCCPR CHANGE PROTECTION
```

FLOW FOR ENTER/RENAME ALLOCATION,

```
        IF USER IS TRUNCATING FILE (N .LE. K), TO TRUNKT
        IF INCREASED SIZE WILL EXCEED UFD QUOTA [C(UFBTAL)-N NEG.]
          CHANGE USER ARG N TO NUMBER, FREE BLOCK LEFT [UFBTAL]
          FLAG FOR ERROR RETURN
        END
        IF USER SPECIFIED WHERE TO ALLOCATE
          CONVERT LOGICAL BLOCK NO. WITHIN F.S. TO UNIT+LOGICAL BLOCK NO. WITH IN UNIT
          TRY TO ALLOCATE (N-K) BLOCKS OR LESS ONLY AT SPECIFIED STARTING PLACE
          ERROR OR OK RETURN TO USER
        END
        IF FILE NOT CURRENTLY ASSIGNED TO ARRAY UNIT [DEVUNI=Ø]
          PICK THE MOST EMPTY UNIT AN ANY FILE STR. ALLOWED BY USER DEVICE NAME
        END
        TRY TO ALLOCATE EXACTLY (N-K) BLOCKS NEXT TO END OF FILE ON CURRENT UNIT
        IF CAN'T GET(N-K), TRY TO ALLOCATE (N-K) OR LESS BLOCKS ANYWHERE ON UNIT
        IF (N-K) FOUND, OK RETURN TO USER
        REMEMBER WHETHER MAX. FOUND ON CURRENT UNIT WAS GREATER THAN MINIMUM [KONFRG]
        DO UNILOP, FOR REST OF UNITS SPECIFIED BY USER DEVICE NAME
        TRY TO ALLOCATE EXACTLY (N-K) CONSECUTIVE ANYWHERE ON UNIT
        IF FOUND, TO NEWUNI
UNILOP: CONTINUE
        IF MAX FOOUND ON ORIGINAL UNIT WAS GREATER THAN MINIMUM[KONFRG]
          USE SMALLER-THAN-DESIRED HOLE ON ORIGINAL UNIT
        ELSE
          USE LARGEST SMALLEST-THAN-DESIRED HOLE ON BEST UNIT SCANNED
        END
        ERROR RETURN TO USER

NEWUNI: ADD NEW UNIT POINTER TO RETRIEVAL POINTERS BEFORE RETRIEVAL POINTER
        OK RETURN TO USER

TRUNKT: READ DISK RIB BLOCK
        SCAN RETRIEVAL POINTERS FOR FILE (DO USEI FOR LAST DESIRED BLOCK)
        CHANGE CURRENT RETRIEVAL POINTER AND SET NEXT ONE TO Ø AS EOF FLAG
        REWRITE DISK RIB (BEFORE ANY SPACE RETURNED IN CASE SYSTEM GOES DOWN)
        RETURN REMAINING SPACE OF CURRENT RETRIEVAL POINTER
        DO DELLOP, FOR ALL REMAINING RETRIEVAL POINTERS IN FILE
          IF NEW UNIT, CHANGE UNITS
DELLOP:   RETURN ALL CLUSTERS WHICH THIS POINTER POINTS TO
        RETURN TO USER
```

ALLOCATION FLOW FOR OUTPUT

```
        IF UFD QUOTA HAS BEEN DRAWN BY MORE THAN MAX OVERDRAW, TO ERRFUL
        TRY TO ALLOCATE A NOMINAL GROUP OR LESS[KONGRP] ON UNIT STARTING AT EOF
        IF NO BLOCKS AVAILABLE STARTING AT END OF FILE
          TRY TO ALLOCATE A NOMINAL GROUP OR LESS[KONGRP] ANYWHERE ON THIS UNIT
          IF NO BLOCKS AVAILABLE
            TRY TO ALLOCATE AND MARK A NOMINAL GROUP OR LESS[KONGRP] ON ANY USER UNIT ALLOED BY THE NAME
              IF NO BLOCKS AVAILABLE
ERRFUL:         SET ERROR FLAG FOR USER [IOBKTL]
                ERROR RETURN TO USER
              END
            END
          END
        END
```

ALLOCATION FLOW FOR USETO

```
        IF AN ENTER HAS NOT BEEN DONE, TO SUPERS
        IF FILE INCREASE WOULD USE UP ALL BLOCK IN UFD QUOTA [UFBTAL]
          SET ERROR FLAG [IOBKTL]
          RETURN TO USER
        END
        TRY TO ALLOCATE (N-K) OR LESS BLOCKS AT END OF FILE [ACCALC]
        IF ALL (N-K) BLOCKS ALLOCATED, RETURN
        DO ALCLOP, UNTIL ALL UNITS TRYED AS SPECIFIED BY USER DEVICE NAME STARTING WITH THIS UNIT
          DO ALCLOP, UNTIL ENOUGH GROUPS ALLOCATED
            TRY TO ALLOCATE ALL OR PART OF REMAINING DESIRED SPACE ON THIS UNIT
            IF ENOUGH ALLOCATED, RETURN
ALCLOP: CONTINUE
        SET ERROR FLAG (IOBKTL)
        RETURN TO USER

SUPERS:
```

```
2 ROUTINES FOR ALLOCATION CALLED BY ENTER,RENAME,USETO,OUTPUT
1.      TRY TO ALLOCATE EXACTLY M(M=N-K) CONSECUTIVE BLOCKS ON THIS UNIT
          ARGS:
            FILE (F)
            UNIT (U)
            NUMBER OF CONSECUTIVE BLOCKS WANTED (EXACTLY)
            LOGICAL BLOCK NUMBER TO START WITHIN THIS UNIT (Ø MEANS ANYWHERE
                ON UNIT WILL DO)

2.      TRY TO ALLOCATE M OR LESS (M=N-K) CONSECUTIVE BLOCKS ON THIS UNIT
          ARGS: SAME

        IF SUCCESSFUL RETURN, THESE ROUTINES MARK STORAGE IN SAT TABLES
        ANS EXPAND LAST GROUP POINTER OR ADD UNIT AND/OR GROUUP POINTERS,
        ALSO THE COUNT OF FREE BLOCKS IN UFD WILL BE DECREMENTED
          (POSSIBLY PART ZERO IF OVERDRAW) ONLY AFTER SAT TABLE HAS BEEN
          SUCCESSFULLY MARKED.
        IF ERROR RETURN, NO SPACE MARKED AS IN USE.

FLOW FOR USETI,USETO
        IF USETI
          IF USER ARG PAST HIGHEST RELATIVE BLOCK WRITTEN [ACCWRT]
            SET EOF
          END RETURN TO USER
        END
        IF DESIRED BLOCK IS BEFORE FIRST IS CORE POINTER [.L.C(DEVFLR)], TO USTRIB
        CALL PTRSCN, (IN=CORE POINTER AREA)
        IF FOUND, TO RANFND
USTRIB: READ RIB BLOCK FOR FILE (AS IF INPUT UUO FOR RELATIVE BLOCK Ø OF FILE)
        (HERE WHEN RIB BLOCK IN ONE OF MONITOR CORE BUFFERS)
        CALL PTRSCN, (MONITOR CORE BUFFER)
        IF FOUND
          COPY WINDOW SURROUNDING DESIRED POSITION INTO FILE DOB
          TO RANFND
        END
        IF USETI UUO
          SET EOF
        ELSE (USETO UUO)
          ALLOCATE ENOUGH GROUPS TO FILL OUT FILE
        END
        EXIT

RANFND: SET UP FILE DATA BLOCK POINTERS SO NEXT SEEK OR INPUT OR OUTPUT UUO
                WILL ACCESS THIS BLOCK
        EXIT
```

```
;SUBROUTINE TO SCAN RETRIEVAL POINTERS IN FILE DATA BLOCK OR MONITOR BUFFER
        INTERNAL SUBROUTINE PTRSCN, (CORE LIMITS)
        DO PTRLOP, UNTIL RUN OUT OF POINTERS
          IF CLUSTER COUNT IS Ø
            IF WORD IS ZERO (MUST BE EOF)
              NOT-FOUND RETURN
            ELSE (MUST BE CHANGED IN UNIT)
              SETUP FOR NEW UNIT
              TO
            ELSE (MUST BE REAL RETRIEVAL POINTER)
              IF DESIRED POSITION IS IN THIS GROUP, FOUND RETURN
            END
PTRLOP: CONTINUE
        NOT FOUND RETURN
```

```
V004    UUO CODE FOR SEEK UUO

V004            IF RUN OUT OF CLUSTER POINTERS, READ CLUSTER PTRS
V004            IF NEXT PTR IN CORE AN EOF, RETURN
V007            IF KONTROLLER DOESN'T POSITION OR UNIT IS IDLE, EXIT
V007            INHIBIT SCHEDULING
V004            TURN ALL DISK PI'S OFF
V004            IF CHANNEL IS IDLE, TO UUOSEK
V004            SET STATE OF UNIT TO SW
V004            (LEAVE STATE OF FILE AS IT IS)
V004            PUT FILE IN SW/PW QUEUE (ONLY ONE THERE)
V004            TO SEKRT1

V004    UUOSEK: IF A UNIT IS ALREADY AT DESIRED POSITION, TO SEKRT1
V004            SET STATE OF UNIT TO S
V004            (DO NOT FLAG ACTIVE IO)
V004            TURN ALL DISK PI'S BACK ON
V007            ENABLE SCHEDULING
V004            ISSUE POSITIONING COMMAND

V004    SEKRT1: TURN ALL DISK PI'S ON
V004    SEKRT2: SCHEDULE
V007            EXIT
```

```
                    ABBREVIATED RENAME FLOW

        IF USER CHANNEL HAS NEVER BEEN CONNECTED TO A FILE BY LOOKUP OR ENTER [DEVFIL=Ø], ERROR RETURN
        IF USER CHANNEL HAS BEEN CLOSED, FIND OLD FILE USING USER'S LOOKUP OR ENTER NAME [DEVFIL]
        IF USER IS TRYING TO DELETE FILE AND HAS THE PRIVILEGES TO DO SO
          MARK FILE FOR DELETION WHEN READ COUNT GOES TO Ø
          TO CALCLS
        END
        IF FILE IS MARKED FOR DELETION, ERROR RETURN
        IF USER IS CHANGING FILE NAME, EXTENSION, OR DIRECTORY, ERROR RETURN IF NEW NAME ALREADY EXISTS [CALL FNDFIL]
        IF FILE ALREADY BEING RENAMED (BY ANOTHER USER CHANNEL)[ACPREN], ERROR RETURN
        IF FILE ALREADY BEING CREATED, SUPERCEDING, OR UPDATED BY ANOTHER USER CHANNEL [ACYSTS;ENTRB], ERROR RETURN
        READ OLD FILE RIB
        CHECK PRIVILEGES FOR EACH ATTRIBUTE USER IS TRYING TO CHANGE AND PERFORM ANY ACTIVITY REQUIRED
        CHANGE ACCESS TABLE (NAME, EXT, DIR, PRIVILEGES) AND MONITOR BUFFER TO REFLECT ALL NEW ATTRIBUTES
CALCLS: CALL CLOSE1 WHICH CALLS CLOSE INPUT, THEN CLOSE OUTPUT (USES SAME MONITOR BUFFER)
        OK RETURN TO USER

                    ABBREVIATED CLOSE INPUT

        IF LOOKUP NOT IN FORCE ON THIS USER CHANNEL [LOOKB], RETURN
        IF FILE READ COUNT IS NOT UP FOR THIS USER CHANNEL, RETURN
        DECREMENT FILE READ COUNT [ACCCNT] AND CLEAR "READ COUNT UP" FOR THIS USER CHANNEL [DEPRDC]
        IF READ COUNT IS NOW Ø FOR FILE (EVEN THOUGH UPDATE MAY STILL BE IN PROGRESS ON THIS OR ANOTHER CHANNEL)
          IF FILE IS MARKED FOR DELETION, ZERO DIRECTORY ENTRY THEN RECLAIM DISK SPACE AND RETURN
          IF OUTPUT CLOSE WILL BE DONE [ENTRB OR RENAB], RETURN SO THAT IT WILL BE DONE
          IF USER DID SOME INPUT UUOS [INPB](AND NOT JUST A LOOKUP TO LIST DIRECTORY)
            IF ACCESS DATE IS NOT TODAY AND THIS STR NOT WRITE-LOCKED FOR THIS USER [DEPWLK], UPDATE RIB AND ACCESS TABLE
          END
        END
        RETURN
```

ABBREVIATED OUTPUT CLOSE

```
        IF NEITHER ENTER NO LONGER IN FORCE ON THIS USER CHANNEL[ENTRB]NOR FILE BEING RENAMED[RENMB], RETURN
        (NEED TO MODIFY UUOCON TO SET RENMB IN DEVPAT, IS USED TO STORE IN MEMORY)
        IF NEITHER DUMP MODE NOR CALL RESET [DEPRST]
          SAVE NO. OF WORDS OF LAST BUFFER [AKBLBC]
          WRITE LAST BUFFER [PUSHJ OUT]
        END
        IF RIB NOT ALREADY IN A MONITOR BUFFER (FROM RENAME)[IOSMBF],READ RIB[AKBPT1]
UPDATE? IF NOT AT LEAST ONE ALLOCATED BLOCK BEYOND LAST BLOCK WRITTEN, ALLOCATE ONE MORE BLOCK(CLUSTER)FOR RIB
        IF USER IN NOT INHIBITING DEALLOCATION OF UNWRITTEN SPACE[CLSDLL],DEALLOCATE ALL BUT ONE BLOCK (CLUSTER)
        MERGE DEVICE DATA BLOCK RETRIEVAL POINTERS INTO MONITOR BUFFER
        IF CALL RESET UUO[DEPRST], RECLAIM DISK SPACE [CALL RECLAM]THEN TO CLRSTS
        WRITE MONITOR BUFFER(RIB)AS ØTH AND LAST+1ST RELATIVE BLOCK WRITTEN OF FILE
        FOR EACH UNIT WHICH FILE PASSES THROUGH, WRITE ALL SAT TABLES WHICH HAVE CHANGED

        IF FILE IS BEING CREATED [ACPCRE] (IE NOT IN A DIRECTORY YET),TO NOTOLD
        (FILE IS BEING UPDATED OR SUPERCEDED OR RENAMED SO OLD NAME ALREADY EXITS IN A DIRECTORY)
        IF NEW DIRECTORY [PPBNAM[ACCPPB]]IS DIFFERENT FROM OLD DIRECTORY [DEVPPN],TO NOTOLD
        REWRITE DIRECTORY BLOCK WITH NEW NAME[NMBNAM[ACCNMB]RING],EXTENSION, AND COMPRESSED FILE POINTER
        READ OLD RIB BLOCK IN MONITOR BUFFER AND RECLAIM DISK SPACE [CALL DELRIB]
        TO CLRSTS

NOTOLD: FIND A FREE WORD PAIR IN NEW DIRECTORY[DEVUFB]
        REWRITE DIRECTORY BLOCK WITH NEW ENTRY IN IT
        IF CHANGING DIRECTORIES, READ OLD UFD AND DELETE OLD NAME FROM IT

CLRSTS: CLEAR UNARY MODIFY STATE CODE FOR FILE [ACYSTS] AND RENAME-IN-PROGRESS FLAG [ACPREN]
        IF ACCESS ENTRY IS DORMANT (READ COUNT=Ø AND NO WRITERS), APPEND ACCESS BLOCK TO SYSTEM DORMANT LIST
        RETURN
```

FLOW FOR ROUTINE TO TEST BAD BLOCKS[TSTBAD]

```
CALL:    MONITOR BUFFER HAS RIB IN IT (TSTBAD WILL WRITE IT)
         BAD LOGICAL BLOCK NUMBER WITHIN UNIT[DEVELB]
         BAD LOGICAL UNIT NUMBER WITHIN STR [DEVEUN]
TSTBAD:  CONVERT FROM LOGICAL UNIT WITHIN STR OF ERROR[DEVEUN]TO GET UNIT DATA BLOCK ADR.
         DO REDBAD, FOR BLOCKS STARTING AFTER BAD BLOCK ON THIS UNIT
           READ NEXT BLOCK(NOT INTO MONITOR BUFFER-SKIP READ)
           IF IT READS OK, TO BADEND
REDBAD:  CONTINUE
         (END OF UNIT)
BADEND:  IF BAD REGION NOT YET STORED IN RIB[[RIBELB=0]
           STORE NO OF BLOCKS IN BAD REGION [RIBNBB], LOGICAL UNIT WITHIN STR[RIBEUN],LOG. BLOCK ADR OF REGION[RIBEUB]
         WRITE MONITOR BUFFER (RIB) AS 0TH AND LAST+1 WRITTEN BLOCKS OF FILE
         IF PREVIOUS BAD REGION STORED IN RIB WAS SAME [RIBEUN,RIBNBB,RIBELB] AS THIS, RETURN
         QUEUE FOR DISK ALLOCATION RESOURCE [DAREQ]
         READ BAD ALLOCATION BLOCK FOR THIS WAIT [LBNBAT]
         IF ERROR, READ OTHER BAD ALLOCATION BLOCK FOR THIS UNIT [LBOBAT]
         DO SCNBAD, FOR ALL BAD REGIONS INSERTED BY MONITOR FOR THIS UNIT [[BATFIR, BATFIR,BATCNT]]
           IF NEW BAD REGION STARTS BEFORE OR AT THIS BAD REGION'S START (THIS START-NEW START)
             IF NEW REGION ENDS BEFORE THIS BAD REGION BEGINS, TO SCNBAD
             INCREASE THIS BAD REGION COUNT OF BAD BLOCKS BY THIS START-NEW START
             STORE NEW BAD REGION START ON TOP OF THIS BAD REGION START
             TO INCREG
           OR IF NEW BAD REGION STARTS BEFORE OR AT THE END OF THIS BAD REGION
INCREG:      IF END OF NEW BAD REGION-END OF THIS BAD REGION IS POSITIVE
               INCREASE THIS BAD REGION COUNT BY DIFFERENCE
             END
             IF THIS ARITH. PROCESSOR NUMBER [ ] IS SAME AS ONE WHICH FOUND BAD REGION [BAYAPN]
               IF THIS CONTROLLER NO. [UNYKNM] IS SAME AS ONE WHICH FOUND BAD REGION [BAYNM]
                 TO STOPOB
             END END
             TO WRTBAT
           END
SCNBAD:  CONTINUE
         (THIS IS A BAD REGION NOT RECORDED BEFORE)
         IF THERE IS STILL ROOM TO BAT BLOCK FOR BAD REGIONS[BAFFIR]
           INCREASE NO. OF DISTINCT BAD REGIONS FOUND BY MONITOR [BAFCNT]
           STORE NO. OF BAD BLOCKS IN THIS BAD REGION [BAYBBC]
           STORE LOGICAL CONTROLLER NO. WITHIN THIS TYPE [BAYKNM-UNYKNM]
           STORE ARITHMETIC PROCESSOR NUMBER[BAYAPN- ]
           STORE FIRST LOGICAL DISK ADR WITHIN UNIT OF BAD REGION [BAFBLB]
STOPUB:  OR-TO-MEMORY PHYSICAL UNIT NUMBER BIT WITHIN CONTROLLER [BAPPUB]
WRTBAT:  WRITE OUT THE BAD ALLOCATION BLOCK IN BOTH PLACES [LBNBAT,LBOBAT]
         END
         RELEASE DISK ALLOCATION RESOURCE [DAREQ]
         RETURN
```

```
REPEAT LOGIC.<

                2 SPACE RECLAIMING SUBROUTINES - DELRIB,RECLAM

        DELRIB IS CALLED WITH AN EMPTY MONITOR BUFFER AND A COMPRESSED FILE POINTER TO RIB AS AN ARG
        RECLAM IS CALLED WITH MONITOR BUFFER ALREADY SETUP WITH FILE RIB

DELRIB: CONVERT OLD COMPRESSED FILE POINTER TO LOGICAL BLOCK NUMBER AND LOGICAL UNIT
        READ OLD RIB BLOCK INTO MONITOR BUFFER

RECLAM: QUEUE FOR DISK ALLOCATION RESOURCE[AUREQ]
        DO ALLPTR, FOR ALL RETRIEVAL POINTERS OF OLD FILE (IN MONITOR BUFFER) UNTIL EOF (ALL 0 POINTER)
          IF NEXT POINTER HAS ZERO CLUSTER COUNT AND UNIT CHANGE BIT IS ON [
            CHANGE TO NEW LOGICAL UNIT WITHIN SAME FILE STRUCTURE
        ELSE
          DO SCNSAT, FOR ALL SAT BLOCKS IN CORE FOR THIS UNIT
            IF FIND SAT BLOCK IN CORE WHICH DESCRIBES SPACE TO BE FREED UP, TO ZERBIT
SCNSAT:     CONTINUE
            IF FIRST SAT BLOCK HAS BEEN MODIFIED [SABCHG], WRITE IT BACK ONTO DISK
            READ DESIRED SAT BLOCK INTO CORE
            ADVANCE POINTER TO FIRST SAT BLOCK IN RING FOR THIS UNIT TO NEXT SAT BLOCK IN RING
ZERSAT:     MARK ALL CLUSTERS REFERRED TO BY THIS RETRIEVAL POINTER AS NOW FREE
        END
ALLPTR: CONTINUE
*       IF THIS FILE HAD A BAD REGION[RIBELB NON-ZERO]
*         CONVERT BAD LOGICAL BLOCK ADR[RIBELB] TO CLUSTER ADDRESS
*         CONVERT NO OF BAD BLOCKS TO CLUSTERS[RIBNBB](ROUNDING UP)
*         MARK ALL CLUSTERS AS IN USE ON BAD UNIT WITHIN STR [RIBEUN]
*       END
        RELEASE DISK ALLOCATION RESOURCE [DAREQ]
        RETURN
```

DETAILED FLOW FOR RENAME

```
IF USER CHANNEL HAS NEVER BEEN CONNECTED TO A FILE BY LOOKUP OR ENTER[LOOKB,ENTRB],ERROR RETURN
IF FILE NAME[DEVFIL]STORED IN DEVICE DATA BLOCK IS 0, SYSTEM ERROR
IF ACCESS TABLE IS NO LONGER ATTACHED TO USER CHANNEL(CLOSE DONE)[DEVACC=0]
  SETUP FILE STRUCTURE SEARCH LIST FROM DEVICE NAME USER USED ON LAST LOOKUP OR ENTER ON CHANNEL [DEVNAM]
  FIND OLD FILE NAME IN FILE SYSTEM[CALL FNDFIL AS IF LOOKUP, CREATE ACCESS BLOCK AND INCREMENT READ COUNT]
  IF FILE NOT FOUND, RELEASE CB RESOURCE AND ERROR RETURN
*   PRETEND LIKE USER HAD DONE A LOOKUP[LOOKB←1,OCLOSB←0 IN AC NOT MEMORY YET]
END
IF USER IS TRYING TO DELETE FILE[NAME ARG=0]
  IF USER IS NOT PERMITTED TO DELETE FILE [CHKPRV[FNCDEL], ERROR RETURN TO USER
  MARK FILE TO BE DELETED WHERE READ COUNT GOES TO ZERO [AKBDEL]
  TO CALCLS
END
IF FILE IS MARKED FOR DELETION[AKBDEL],ERROR RETURN
IF USER IS CHANGING DIRECTORY[ARG DIFF. ACCUFB], FILE NAME[ARG DIFF. NMBNAM[ACCNMB]],OR FILE EXT[ARG DIFF NMBEXT]
  IF USER CHANNEL HAS MONITOR BUFFER(CALL TO FNDFIL),GIVE IT UP
  STORE NEW DIRECTORY, FILE-NAME AND EXTENSION [DEVPPN,DEVFIL,DEVEXT] IN DEVICE DATA BLOCK
  SCAN FILE SYSTEM TO SEE IF NEW NAME[DEVFIL,DEVEXT]EXISTS IN NEW DIRECTORY[ACCUFB][CALL FNDFIL[RENAME]]
  [FNDFIL GIVES ERROR RETURN FOR ANY OF THE FOLLOWING REASONS:
    A. NEW NAME EXISTS IN FILE SYSTEM
    B. EVEN THOUGH NEW NAME DOES NOT EXIST, USER DOES NOT HAVE PRIVILEGES TO CREATE IN DIRECTORY
    C. USER DOES NOT HAVE PRIVILEGES TO CHANGE NAME AND/OR DIRECTORY
    D. USER WANTED TO CHANGE DIRECTORY BUT WAS NOT AN UPDATER [ENTRB]
    E. DISK QUOTA FOR NEW DIRECTORY WOULD BE EXCEEDED
  FNDFIL CHANGES NAME RING[ACCNMB], DIRECTORY[ACCPPB], AND[DEVUFB]FOR ACCESS BLOCK IF OK RETURN].
*   FNDFIL MARKS THIS FILE AS BEING RENAMED [ACPREN] MAYBE ALREADY CREATED, SUPERSEDING, OR BEING UPDATED)
  IF ERROR RETURN FROM FNDFIL
*   RESTORE OLD FILE NAME, EXT AND DIRECTORY
    RELEASE MONITOR BUFFER
    ERROR RETURN TO USER
  END
ELSE
*   NO SCHEDULE
  IF FILE ALREADY BEING RENAMED[ACPREN],SCHEDULE AND ERROR RETURN TO USER
  FLAG FILE IS BEING RENAMED[ACPREN] IN ACCESS BLOOCK
*   SCHEDULE
END
```

```
        CLEAR RIB IN MONITOR BUFFER FLAG (IN CASE ON FROM FNDFIL FOR NEW FILE NAME)
        IF USER-CHANNEL DOES NOT LEAVE A MONITOR BUFFER, QUEUE FOR ONE AND FLAG [IOSMBF IN IOS]
        READ OLD FILE RIB [AKBPT1] INTO MONITOR BUFFER AND FLAG USER-CHANNEL AS RIB IN MONITOR BUFFER[DEPRIB]
        SET RIB IN MONITOR BUFFER FLAG[DEPRIB] FOR THIS USER CHANNEL
        EXCHANGE OLD FILE IN RIB[RIBNAM] WITH NEW NAME IN DEVICE DATA BLOCK [DEVFIL]
        EXCHANGE OLD  EXT IN RIB[RIBEXT] WITH NEW EXT IN DEVICE DATE BLOCK [DEVEXT]
        EXCHANGE OLD PROJ-PROG IN RIB[RIBPPN] WITH NEW PROJ-PROG IN DEV DATA BLOCK[DEVPPN]
        SET LOGICAL BLOCK NO WITHIN STR OF NEW UFB BLOCK[UFBPT1[ACCUFB]]
        IF USER IS TRYING TO CHANGE FILE PROTECTION
          IF USER IS NOT PERMITTED TO CHANGE THE FILE'S PROTECTION[CHKPRV[FNCCPR]],ERROR RETURN
          MODIFY MONITOR BUFFER AND ACCESS BLOCK WITH NEW PROTECTION
        END
        IF USER IS TRYING TO CHANGE OTHER ATTRIBUTES
          IF USER IS NOT PERMITTED TO CHANGE OTHER ATTRIBUTES [CHKPRV[FNCCAT]],ERROR RETURN
          MODIFY MONITOR BUFFER AND ACCESS BLOCK WITH NEW ATTRIBUTES
        END
        IF USER IS TRYING TO ADD ALLOCATION TO THE FILE [ARG,GR,AKBALC]
          IF USER IS NOT PERMITTED TO ALLOCATE ADDITIONAL SPACE, ERROR RETURN
          ALLOCATE SPACE AND MODIFY MONITOR BUFFER AND ACCESS BLOCK
        OR IF USER IS TRYING TO DEALLOCATE SPACE FROM THE FILE [ARG,L,AKBALC]
          IF USER IS NOT PERMITTED TO DEALLOCATE, ERROR RETURN
          DEALLOCATE SPACE AND MODIFY MONITOR BUFFER AND ACCESS BLOCK
        OR IF USER IS TRYING TO TRUNCATE DATA FROM THE FILE[ARG,L,ACCWRT]
          IF USER IS NOT PERMITTED TO TRUNCATE DATA FROM THE FILE, ERROR RETURN
          TRUNCATE SPACE AND MODIFY MONITOR BUFFER AND ACCESS BLOCK
        END
        IF FILE IS BEING CREATED OR SUPERSEDING BY THIS USER CHANNEL[ENTRB,AND,(ACCCRE,OR,ACCSUP)]
          SET CLOSE TO DO AUTOMATIC DEALLOCATION [CLEAR SUPPRESS DEALLOCATION BIT IN UUO]
        ELSE (LOOKUP-RENAME OR UPDATE)
          SET CLOSE TO SUPPRESS AUTOMATIC DEALLOCATION [SET SUPPRESS DEALLOCATION BIT IN UUO]
        END
CALCLS: (DO NOT TOUCH FLAGS SAYING LOOKUP AND/OR ENTER IN FORCE(ONE MUST BE ON)
        CALL CLOSE1 IN UUOCON WHICH CALLS CLOSE INPUT THEN CLOSE OUTPUT(USES MONITOR BUFFER)
        OK RETURN TO USER
```

CLOSE INPUT - DETAILED FLOW

```
        IF LOOKUP NOT IN FORCE ON THIS USER CHANNEL [LOOKB], RETURN
        MARK LOOKUP NO LONGER IN FORCE IN AC BUT NOT YET IN MEMORY[LOOKB←Ø](IN CASE NOT GO TO COMPLETION)
        PREVENT USER FROM HIBITING OUTPUT CLOSE[CLSOUT IN UUO](SINCE CAN'T GO FROM UPDATE TO WRITE)
        IF THERE IS NO ACCESS BLOCK FOR THIS USER CHANNEL, RETURN(SYSTEM ERROR?)
        IF FILE READ COUNT IS NOT UP FOR THIS USER CHANNEL [DEPRDC],RETURN(SYSTEM ERROR?)
        QUEUE FOR CORE BLOCK RESOURCE [CBREQ]
        NO CONTROL C
        DECREMENT FILE READ COUNT [ACCCNT]
        CLEAR "READ COUNT UP" FLAG[DEPRDC] FOR THIS USER CHANNEL
        IF READ COUNT IS NOW Ø (EVEN THOUGH UPDATE MAY STILL BE IN PROGRESS ON THIS OR ANOTHER CHANNEL)
          IF FILE IS MARKED FOR DELETION [UKBDEL]
        IF USER CHANNEL DOESN'T HAVE A MONITOR BUFFER, QUEUE FOR ONE [MQREQ]
        QUEUE FOR ALTER UFD RESOURCE[AUREQ]
        DO ZERDIR, FOR EACH BLOCK IN UFD(IN FILE STRUCTURE WHICH FILE STARTS IN)
          IF FIND A MATCH FOR NAME AND EXTENSION[NAMARG],TO ZERNAM
ZERDIR: CONTINUE
        RELEASE ALTER UFD RESOURCE
        TO FREACC

ZERNAM: REMEMBER COMPRESSED FILE POINTER FOUND WITH NAME AND EXTENSION IS UFD(TO READ RIB AND RECLAIM SPACE)
        CLEAR NAME AND EXTENSION WORD IN UFD BLOCK
        BLOCK MOVE ALL ENTRIES AFTER UP TWO[NAMSIZ==2] IN UFD BLOCK AND ZERO THE LAST TWO
        (THIS ORDER IS PRESERVED(WITHIN EACH UFD BLOCK ONLY) AND ZERO FLAGS EARLY END OF BLOCK)
        REWRITE UFD BLOCK
        RELEASE ALTER UFD RESOURCE
            READ RIB IN MONITOR BUFFER AND RECLAIM SPACE [CALL DELRIB]
FREACC:     PUT ACCESS BLOCK ON FREE LIST
            RETURN
          END
        IF FIRST FILE STRUCTURE FOR FILE[DEVFSN] IS WRITE LOCKED FOR THIS USER [DEPWLK], RETURN (DO NOT UPDATE ANY INFORMATION)
        IF OUTPUT CLOSE WILL BE DONE [ENTRB OR RENMB], RETURN (SO OUTPUT BUFFERS CAN BE PREPARED AND/OR RIB UPDATED)
        IF USER DID SOME INPUT UUOS [INPB](AND NOT JUST LOOKUP TO LIST DIRECTORY)
          IF ACCESS DATE IS NOT TODAY [THSDAT]
            CHANGE IT TO TODAYS DATE IN ACCESS BLOCK
            QUEUE FOR MONITOR BUFFER[MQREQ]
            READ FILE RIB INTO MONITOR BUFFER
            CHANGE ACCESS DATE IN MONITOR BUFFER
            REWRITE RIB
            RELEASE MONITOR BUFFER
          END
        END
```

```
          DETAILED FLOW FOR CLOSE OUTPUT (RESET,RELEASE,CLOSE,RENAME,ENTER,INIT,OPEN)

        IF NEITHER ENTER NO LONGER IN FORCE ON THIS USER CHANNEL [ENTRB] NOR RENAME UUO[RENMB], RETURN
        (NEED TO MODIFY UUOCON TO SET RENMB IN AC NO NEED TO STORE)
        MARK ENTER NO LONGER IN FORCE IN AC BUT NOT YET IN MEMORY [ENTRB]
        IF THIS USER CHANNEL NO LONGER HAS AN ACCESS ENTRY [DEVACC=0], RETURN (SYSTEM ERROR?)
        IF THIS USER CHANNEL MODE IS DUMP [D,DR] OR THIS UUO IS RESET [DEPRST], TO NOOUTP (DO NOT OUTPUT)
        IF NO OUTPUT BUFFERS HAVE BEEN SET UP, TO NOOUTP
        [SEE LINES 1019-1041 OF 4S,50 DISK SERVICE]
        STORE WORD COUNT OF LAST BUFFER IN ACCESS TABLE [AKBLBC]
        IF LAST BUFFER HAS 0 WORD COUNT, TO NOOUTP
        WRITE LAST PARTIAL BUFFER [PUSHJ OUT]
        WAIT UNTIL IO FINISHED [PUSHJ WAIT]
NOOUTP: IF THIS USER CHANNEL DOES NOT HAVE A MONITOR BUFFER [IOSMBF]
          QUEUE FOR MONITOR BUFFER
          SET THIS JOB HAS A MONITOR BUFFER FLAG [IOSMBF]
        END
        IF RIB NOT IN A MONITOR BUFFER [DEPRIB], READ RIB INTO MONITOR BUFFER [ACCPT1]
        MERGE DEVICE DATA BLOCK POINTERS INTO RIB IN MONITOR BUFFER
        IF RESET UUO [DEPRST]
          CALL RECLAIM (RECLAIMS ALL STORAGE FOR RIB IN MONITOR BUFFER)
          TO CLRSTS
        END
        IF NO MORE ALLOCATED BLOCKS [AKBALC] BEYOND THE HIGHEST WRITTEN BLOCK [ALCWRT]
          ALLOCATE 1 MORE BLOCK (CLUSTER) FOR SECOND RIB
          ADD NEW POINTER OR INCREASED LAST POINTER TO DEVICE DATA BLOCK
          DECREMENT NO. OF FREE BLOCKS IN UFD [UFBTAL]
          INCREMENT NO. OF BLOCKS ALLOCATED TO FILE [AKBALC] (BUT NOT NO. OF BLOCKS WRITTEN[ACCWRT])
        END
        IF USER IS NOT INHIBITING DEALLOCATION OF UNWRITTEN SPACE [CLSDLL]
          DEALLOCATE ALL BUT FIRST UNWRITTEN BLOCKS FROM END OF FILE (MAYBE MORE THAN ONE POINTER)
          CHANGE POINTERS IN DEVICE DATA BLOCK AND MONITOR BUFFER
          STORE NEW HIGHEST ALLOCATED BLOCK (NOT COUNTING SECOND RIB) IN ACCESS BLOCK
        END
        MERGE DEVICE DATA BLOCK POINTERS WITH RIB IN MONITOR BUFFER
        STORE NO. OF WORDS (NOT BLOCKS) WRITTEN [ACCWRT*128+AKBLBC] IN MONITOR BUFFER [RIBSIZ]
*       IF USER CHANNEL HAD ANYERRORS WHILE ACCESSING FILE WHICH INDICATED DATA IS BAD
*       (SO THAT USER SHOULD GET A LOGIN AND LOGOUT ERROR MESSAGE)
*       [IOSSCE,IOSBRE,IOSHRE,IOSHWE,IOSHPE BUT NOT DEVICE OR CHANNEL ERRORS]
*          SET CORRESPONDING FLAG FOR FILE IN RIB[RIPSCE,RIPBRE,RIPHRE,RIPHWE,RIPHPE]
*       END
*       IF THIS USER CHANNEL HAD TROUBLE WITH FILE INDICATING THAT SURFACE IS PROBABLY BAD
*       (HARDWARE DETECTED PARITY ERRORS[IOSHRE,IOSHWE] OR POSITIONING ERROR[IOSHPE])
*       (BUT NOT DEVICE ERROR OR CHANNEL ERROR OR SOFTWARE CHECKSUM ERROR OR BAD RETREIVAL ERROR)
*          TEST BAD REGION FOR EXTENT [CALL TSTBAD], RECORD IN RIB THEN WRITE RIBS
*       ELSE
*         WRITE MONITOR BUFFER(RIB) AS 0TH AND LAST+1 WRITTEN BLOCKS OF FILE
*       END
*       IF ANY ERRORS FOR FILE[IOSSCE,IOSBRE,IOSHRE,IOHWE,IOSHPE]NEEDING LOGIN MESSAGE
*         QUEUE FOR ALTER UFD RESOURCE [AUREQ]
*         READ UFD RIB
*       SET COORESPONDING ERROR FLAG FOR LOGIN TO SEE (RIQSCE,RIQBRE,RIQHRE,RIQHWE,RIQHPE]
```

```
*          WRITE UFD RIB
*        END
```

```
        QUEUE FOR DISK ALLOCATION RESOURCE [DAREQ](STILL HAVE MONITOR BUFFER WITH RIB IN IT [MQREQ])
        DO ALLPTR, FOR ALL RETRIEVAL POINTERS IN MONITOR BUFFER
          IF POINTER IS A NEW UNIT AND THE SAT TABLE FOR THAT UNIT HAS BEEN MODIFIED [SAT↑CHG]
            WRITE OUT SAT BUFFER
            CLEAR SAT BUFFER BEEN MODIFIED BIT
          END
ALLPTR: CONTINUE
        RELEASE DISK ALLOCATION RESOURCE [DAREQ]

        IF FILE IS BEING CREATED [ACPCRE](I.E. NOT IN A DIRECTORY YET), TO NOTFOLD
        (FILE BEING UPDATED, SUPERCEDED OR RENAMED SO OLD NAME ALREADY EXISTS IN A DIRECTORY)
        IF NEW DIR NAME(PROJ-PROG)[PRBNAM[ACCPPB]] IS DIFFERENT FROM OLD DIREDEVPPN](RENAME UUO CHANGED), TO NOTOLD
        DO OLDPPB, FOR EACH PROJ-PROG BLOCK IN CORE FOR SYSTEM
          IF FIND OLD DIRECTORY NAME (PROJ PROG)[PPBNAM=DEVPPN], TO FNDPPB
OLDPPB: CONTINUE
        SYSTEM ERROR
FNDPPB: DO SAMSTR, FOR ALL UFB CORE BLOCKS WITH THIS PROJ-PROG NUMBER
          IF FIND A UFD BLOCK [UFBFSN] IN SAME FILE STRUCTURE AS OLD (AND NEW) FILE TO FNDUFB
SAMSTR: CONTINUE
        SYSTEM ERROR
FNDUFB: QUEUE FOR ALTER UFD RESOURCE [AVREQ]
        DO SCNUFD, FOR ALL BLOCKS OF UFD
          READ NEXT UFD BLOCK INTO MONITOR BUFFER
          IF FIND OLD NAME [DEVNAM,DEVEXT] IN UFD BLOCK, TO FNDNAM
SCNUFD: CONTINUE
        (OLD FILE NAME NOT FOUND IN DIRECTORY - CAN HAPPEN IF ANOTHER USER RENAMED SAME FILE WHILE THIS USER READING
        RELEASE ALTER UFD RESOURCE [AUREQ]
        TO NOTOLD (PRETEND LIKE THERE IS NO OLD FILE)

FNDNAM: REMEMBER OLD COMPRESSED FILE POINTER FOUND WITH OLD NAME IN UFD (FOR CALL TO DELRIB)
        CHANGE FILE NAME IN DIRECTORY TO NEW NAME [NMBNAM[ACCNMB]RING] (IN CORE DIFFERENT)
        CHANGE FILE EXTENSION IN DIRECTORY TO NEW EXTENSION [NMBEXT[ACCNMB]RING] (IN CORE DIFFERENT)
        CHANGE COMPRESSED FILE POINTER IN DIRECTORY TO NEW COMPRESSED POINTER [AKBPT1,ALBUN1] (MUST BE DIFFERENT)
        REWRITE UFD BLOCK FROM MONITOR BUFFER
        RELEASE ALTER UFD RESOURCE [AUREQ]
        CALL DELRIB WITH ARG OF REMEMBERED OLD COMPRESSED FILE POINTER (READ RIB AND RECLAIM ALL SPACE)
        TO CLRSTS
```

```
NOTOLD: QUEUE FOR ALTER UFD RESOURCE [AUREQ]
        DO SCNFRE, FOR ALL UFD BLOCKS IN NEW DIRECTORY [UFBPT1[DEVUFB]]
          IF LAST NAME PAIR IN UFD BLOCK IS UNUSED (Ø), TO FNDFRE
SENFRE: CONTINUE
        READ UFD RIB INTO MONITOR BUFFER
        IF NOT ANOTHER FREE BLOCK IN FILE, ALLOCATE ANOTHER [RIBALC] TO UFD FILE
        UPDATE NO. OF BLOCKS WRITTEN [RIBWRT] IN UFD FILE
        DECREMENT NO. OF FREE BLOCKS IN THIS UFD IN THIS FILE STRUCTURE. [UFBTAL]
        REWRITE UFD RIB
        STORE NEW FIRST RETRIEVAL POINTER TO UFD IN UFD BLOCK [UFBPT1]
        CONVERT CLUSTER COUNT TO BLOCK COUNT[UFBPT1]
        IF UFD NOW HAS MORE THAN ONE POINTER OR BLOCK COUNT OVERFLOWED COUNT FIELD
        CLEAR THIS UFD HAS ONLY ONE RETRIEVAL POINTER [UFB1PT]
        END
        READ NEWLY ALLOCATED BLOCK INTO MONITOR BUFFER AND CLEAR IT OUT
FNDFRE: DO SCNZER, FOR ALL NAME PAIRS IN THIS UFD BLOCK
            IF FIND A ZERO ENTRY, TO FNDZER
SCNZER: CONTINUE
        SYSTEM ERROR
FNDZER: STORE NEW NAME[NMBNAM[ACCNMB]RING] AND EXTENSION IN NEW SLOT IN DIRECTORY
        COMPUTE AND STORE NEW COMPRESSED FILE POINTER[AKBPT1,AKBUN1]
        REWRITE DIRECTORY BLOCK WITH NEW ENTRY ADDED TO IT
        RELEASE ALTER UFD RESOURCE[AUREQ]
CLRSTS: CLEAR UNARY MODIFY STATE CODE FOR FILE[ACYSTS] IN ACCESS BLOCK AND RENAME IN PROGRESS[ACPREN]
        IF ACCESS BLOCK IS NOW DOMINANT (READ COUNT=Ø,MODIFY=Ø)
          IF ACCESS BLOCK NOT ALREADY ON DORMANT LIST (SYSTEM ERROR), APPEND ACCESS BLOCK TO SYSTEM DORMANT LIST
        END
        RELEASE MONITOR BUFFER[MQREQ]
        RETURN
```

```
THE UUO CODE FOR INPUT AND OUTPUT IS DIVIDED INTO 5 PARTS:

UUOPTR: COMPUTE THE LOGICAL BLOCK NEEDED FOR I/O
UUOPWQ: PUT FILE INTO POSITION WAIT QUEUE, IF POSITIONING CANNOT BE
        DONE NOW.
UUOPOS: SET UP AND ISSUE POSITIONING COMMAND IF NEEDED
UUOTWQ: PUT FILE INTO TRANSFER WAIT QUEUE, IF TRANSFER CANNOT BE DONE NOW,
UUOTRN: SET UP AND ISSUE TRANSFER COMMAND

UUOPTR:
*       IF OUTPUT
*         IF THIS STR IS WRITE LOCKED FOR THIS USER [DEVWLK], SET ERROR FLAG [IOBKTL] AND RETURN TO USER

*          IF NO. OF FREE BLOCKS IN UFD GONE TO 0 [UFBTAL]
*             TYPE WARNING MESSAGE BUT ALLOW WRITING TO CONTINUE
*             TO UUOPWQ
*          OR IF UFD LOGGED-IN DISK QUOTA GONE PAST FILE STRUCTURE OVERDRAW
*             SET SOFTWARE ERROR FLAG [IOBKTL]
*             EXIT
*          END (WAIT TILL INTERRUPT TO DECREMENT)
        IF RAN OUT OF CLUSTER PTRS, READ CLUSTER POINTERS
        IF THIS PTR IN CORE IS NOT AN EOF, TO UUOPWQ
        FLAG EOF
        EXIT

UUOPWQ: FLAG ACTIVE USER IO
        INHIBIT SCHEDULING
        TURN ALL DISK PI'S OFF
        IF THIS IS FIXED HEAD DEVICE, TO UUOTWQ
        IF UNIT IS IN SW STATE, CLEAR THE QUEUE (ONLY ONE FILE CAN BE IN IT)
        IF KONTROLLER IS IDLE, TO UUOPOS
        SET FILE STATE TO PW
        IF UNIT IS IDLE, SET IT TO PW STATE
        ADD FILE TO END OF SW/PW QUEUE FOR UNIT
        TO UUOEXT

UUOPOS: IF UNIT ALREADY AT DESIRED POSITION, TO UUOTWQ
        SET STATE OF UNIT TO P
        SET STATE OF FILE TO P
        TURN ALL DISK PI'S ON
        ENABLE SCHEDULING
        ISSUE POSITIONING COMMAND
        EXIT

UUOTWQ: IF CHANNEL IS IDLE, TO UUOTRN
        SET UNIT STATE TO TW
        SET FILE STATE TO TW
        ADD FILE (DDB) TO END OF TW QUEUE FOR CHANNEL
        TO UUOEXT

UUOTRN: SET CHANNEL STATE TO BUSY
        SET KONTROLLER STATE TO BUSY
        SET UNIT STATE TO T
        SET FILE STATE TO T
        TURN ALL DISK PI'S ON
        ENABLE SCHEDULING
        TO STARTT (COMPUTE CHANNEL COMMAND BIT AND START TRANSFER)
UUOEXT: TURN ALL DISK PI'S ON
```

```
ENABLE SCHEDULING
EXIT
```

```
INTERRUPT PROCESSING FILINT:

1.      POSITIONING INTERRUPTS
POSTST: START DATA TRANSFER

2:      DATA TRANSFER INTERRUPT
POSTST: QUEUE ANY UNITS WHICH MAY HAVE REACHED POSITION DURING DATA TRANSFER
POSDON: FINISH BOOKEEPING ON DATA TRANSFER, WAKE UP USER IF IN IO WAIT, ETC.
PIKPOS: FOR EACH UNIT WHICH NEEDS POSITIONING ON THE CONTROLLER PICK A
        WAITING FILE TO START POSITIONING.
PIKTRN: SCAN ALL UNIT ON ALL CONTROLLERS ON THIS CHANNEL AND
        PICK THE FASTEST ONE TO START TRANSFER

        NOTE:   PIKPOS USUALLY PICKS POSITIONING AND TRANSFER ON THE
                BASIS OF WHICH WILL MINIMIZE THE IDLE TIME OF THE DEVICE.
                HOWEVER, EVERY MTH DATA TRANSFER PIKPOS PICKS A FILE
                WHICH HAS BEEN WAITING THE LONGEST ON EACH UNIT, (FRONT
                OF PW QUEUE FOR EACH UNIT) INSTEAD OF THE FASTEST IN
                THE ENTIRE PW QUEUE FOR EACH UNIT.  IN OTHER WORDS,
                EVERY MTH DATA INTERRUPT POSITIONINGS ARE SELECTED
                ON THE BASIS OF BEING FAIR INSTEAD OF BEING FAST.
                A DIFFERENT COUNTER IS KEPT FOR PIKTRN, SO THAT EVERY
                NTH TIME THE LONGEST WAITING TRANSFER RATHER THAN THE
                SHORTEST LATENCY IS PICKED.

;DEPENDENT ROUTINE FLOW

HERE ON AN INTERRUPT
        IF ANY PECULIAR TYPE OF ERRORS, TAKE CORRECTIVE MEASURES (SUCH AS RECALIBRATE)
        IF END OF CYLINDER REACHED AND MORE TO TRANSFER
                STORE NEW LOGICAL BLOCK WITHIN UNIT [C(UNIBLK)]
                STORE NEW CYLINDER POSITION FOR UNIT [C(UNICYL)]
                ISSUE IO COMMAND TO POSITION TO NEXT CYLINDER
                COMPUTE NEW RESIDUE CHANNEL COMMAND LIST
                DISMISS INTERRUPT
        END

        IF THIS IS A POSITIONING INTERUPT CAUSED BY A MID DATA TRANSFER
        POSITIONING
                START UP REST OF DATA TRANSFER
                DISMISS INTERRUPT
        END

SETUP THE FOLLOWING INFORMATION FOR DEVICE INDEPENDENT CODE:
        A,  BITS FOR POSITIONS WHICH JUST FINISHED ON THIS CONTROLLER
        B,  IF A DATA TRANSFER INTERRUPT, UNIT NUMBER (JUST FOR CHECKING)
        C,  ERROR FLAG IF ANY ERRORS IN T, HARDWARE ERRORS IN T1
        D,  FUNCTION CODE LAST GIVEN TO CONTROLLER (REASON FOR INTERRUPT)
PUSHJ TO DEVICE INDEPENDENT CODE [FILINT]
DISMISS INTERRUPT
```

```
V006    FILINT:

V003    POSTST: IF NO MORE POSITIONER COMPLETED BITS, TO POSDON
V003            GET NEXT POSITIONER COMPLETED UNIT NUMBER ON THIS CONTROLLER
V003            IF UNIT WAS IDLE
V003               IF UNIT IS NOT A DISK PACK, HALT
V003*              FLAG TO READ HOME BLOCK BEFORE NEXT IO
V003            OR IF UNIT JUST FINISHED SEEK (IN STATE S)
V003               IF NO MORE FILES IN PW-SW QUEUE FOR THIS UNIT
V003               SET STATE OF UNIT TO IDLE
V003            ELSE
V003               SET STATE OF UNIT TO PW
V003               END
V003            OR IF CHANNEL IS BUSY
V006               TURN OFF ALL DISK PI'S
V003               SET STATE OF UNIT TO TW
V003               SET STATE OF FILE TO TW
V003               ADD FILE TO END OF TWQ FOR CHANNEL
V006               TURN ON ALL DISK PI'S
V003            ELSE
V003               SET STATE OF CHANNEL TO BUSY
V003               SET STATE OF CONTROLLER TO BUSY
V003               SET STATE OF UNIT TO T
V003               SET STATE OF FILE TO T
V003               COMPUTE CHANNEL COMMAND LIST
V003               START DATA TRANSFER
V003            END
V003            TO POSTST
```

```
V003    POSDON: IF INTERRUPT WAS JUST FOR POSITIONING AND NO ERRORS, RETURN AND DISMISS INT, (TRANSFER ALREADY STARTED)
        IF ANY ERRORS
          OR-TO-MEMORY ALL HARDWARE ERROR FLAGS TO SOFT ERROR WORD FOR UNIT [UNISOF]
          INCREMENT ERRORS ON THIS CHANNEL [CHNECT]
          IF STILL NOT TOO MANY CONSECUTIVE SOFT ERRORS, TO START [CHNECT ,LESS DSKTRY]
          OR-TO MEMORY ALL HARDWARE ERROR FLAGS TO HARD ERROR WORD FOR UNIT [UNIERR]
          COMPUTE HOW MANY BLOCKS TRANSFERRED BEFORE ERROR OCCURRED USING CHANNEL CONTROL WORD [C(C(RONIOC)+',)]
          IF BUFFERED MODE AND NOT MONITOR IO, ADVANCE BUFFERS BY NUMBER OF GOOD BLOCKS
          IF CHANNEL ERROR [CHNERR], SET USER CHANNEL DEVICE ERROR [IODEER]
          IF PARITY ERROR [PARERR], SET USER CHANNEL DATA ERROR [IODTER]
          IF POSITIONING ERROR [POSERR], SET USER CHANNEL DEVICE ERROR [IODERR]
          IF OTHER DEVICE ERROR [DEVERR], SET USER CHANNEL DEVICE ERROR [IODERR]
          IF BUFFERED MODE AND NOT MONITOR IO, ADVANCE BUFFER CONTAINING ERROR (AND STORE ERROR BITS IN BUFFER)
          IF PARITY ERROR [PARERR]
            IF READING, SET USER CHANNEL HARDWARE DETECTED READ ERROR [IOSHRE]
            IF WRITING, SET USER CHANNEL HARDWARE DETECTED WRITE ERROR [IOSHWE]

          OR IF POSITIONING ERROR [POSERR]
            SET USER CHANNEL POSITIONING ERROR [IOSHPE]
          END
          IF LOGICAL BLOCK OF ERROR IS NOT ALREADY STORED IN DEVICE DATA BLOCK [DEVELB]
            COMPUTE LOGICAL BLOCK IN WHICH ERROR OCCURRED USING CHANNEL CONTROL WORD
            STORE BAD LOGICAL BLOCK NUMBER IN DEVICE DATA BLOCK FOR MARKING BAT BLOCK AT CLOSE [DEVELB]
            STORE LOGICAL UNIT NUMBER WRITTEN ON WHICH THIS ERROR OCCURRED [DEVEUN-UNYLUN]
          END
          IF THIS IS IN IO WAIT, WAKE IT UP [SETIOD]
          IF ERROR LOGGING PROGRAM IS RUNNING, AND IS IN ERROR WAIT, WAKE IT UP
          TO SETIDL
        END
        IF THIS IS FIRST BLOCK OF A GROUP [IOSFIR] AND THIS OPERATION WAS A READ [IO-1] AND NOT MONITOR IO [MNIOBT]
          CLEAR FIRST BLOCK OF A GROUP FLAG [IOSFIR]
          IF DUMP MODE, GET CORE ADDRESS OF BLOCK [DEVDMP]
          IF BUFFERED MODE, GET CORE ADDRESS OF BLOCK FROM
          COMPUTE 36 BIT EXCLUSIVE OR OF SOME [UNYCKN] OF THE WORDS OF THIS FIRST BLOCK
          COMPARE THE LOW ORDER [ ] BITS OF RESULT WITH CHECKSUM STORED IN RETREVIAL POINTER
          IF NOT A MATCH,
            INCREASE COUNT OF NUMBER OF SOFTWARE CHECKSUM ERRORS FOR THIS UNIT [UNINSC]
            SET BLOCK TO LARGE (RATHER THAN DEVICE OR DATA ERROR) FOR USER CHANNEL[IOBKTL]
            SET UNCHANNEL ERROR FLAG FOR CLOSE [IOSSCE]
            IF ERROR LOGGING PROGRAM IS RUNNING AND IS IN ERROR WAIT, WAKE IT UP
          END
        END
        IF SOFT ERROR COUNT IS NOW ZERO [CHNECT] OR SOFTWARE CHECKSUM FAILED
          IF ERROR LOGGING PROGRAM IS RUNNING AND IS IN ERROR WAIT, WAKE IT UP
        END
V003            DECREASE FAIRNESS COUNTS FOR POSITIONING AND TRANSFERS ON THIS CONTROLLER
V003            IF FILE MODE IS DUMP MODE
V003            ELSE
V003               IF JOB IN IO WAIT, WAKE JOB UP
V003               ADVANCE BUFFERS BY NUMBER OF BLOCKS TRANSFERRED
```

```
V003                IF NO MORE EMPTY INPUT BUFFERS OR FULL OUTPUT BUFFERS, TO SETIDL
V003             END
V003             IF EXHAUSTED THIS GROUP RETRIEVAL POINTER
V003     NEXPTR:    IF NEXT GROUP RETIREVAL POINTER NOT IN CORE, TO SETIDL
V003                IF NEXT PART OF FILE IS ON ANOTHER LOGICAL UNIT WITHIN FILE STRUCTURE
V003                  SETUP POINTERS TO NEW UNIT [DDBUNI]
V003                      TO NEXPTR
V003                END
V003                IF NEXT POINTER IS AN EOF, TO SETIDL
V003             END
V003             IF THIS CONTROLLER DOES NOT DO POSITIONING
V003                ADD FILE TO END OF TWQ
V003                SET UNIT TO PW STATE
V003                TO PIKPOS
V003             END

V003     SETIDL: SET STATE OF UNIT TO IDLE
V003             SET STATE OF FILE TO IDLE
V003             IF CONTROLLER IS FIXED HEAD, TO PIKTRN
```

```
V003    PIKPOS: LOOK AT NEXT UNIT IN RING ON THIS CONTROLLER

V003            IF THIS UNIT IN NEITHER PW NOR SW STATE, TO PIKPND

V003    PWQLUP: LOOK AT NEXT FILE IN PW-SW QUEUE FOR THIS UNIT
V003            IF THIS FILE ACCESS IS FOR CURRENT CYLINDER OF ITS UNIT
V003               DELETE FILE FROM PW-SW QUEUE
V003               SET UNIT STATE TO TW
V003               SET FILE STATE TO TW
V006               TURN OFF ALL DISK PI'S
V003               ADD FILE TO END OF TW QUEUE FOR THIS CHANNEL
V006               TURN ON ALL DISK PI'S
V003               TO PIKPOS
V003            OR IF THIS FILE IS THE CLOSEST SO FAR TO CURRENT CYLINDER
V003               REMEMBER THIS FILE
V003               REMEMBER PREVIOUS FILE IN QUEUE
V003               REMEMBER HOW CLOSE THIS FILE IS TO CURRENT CYLINDER
V003            END
V003            IF NOT TIME TO BE FAIR AND MORE FILES ON THIS UNIT, TO PWQLUP
V007            REMOVE FILE FROM PW-SW QUEUE FOR THIS UNIT
V003            SET UNIT TO P STATE
V003            SET FILE TO P STATE
V003            START POSITIONING ON THIS CONTROLLER
V003    PIKPND: IF MORE UNITS ON THIS CONTROLLER RING TO LOOK AT, TO PIKPOS

V003    PIKTRN: SET CONTROLLER WHICH FINISHED DATA TRANSFER TO IDLE
V003            IF NO FILES IN TWQ FOR THIS CHANNEL, TO KONIDL
V003    TWQLUP: LOOK AT NEXT FILE IN TWQ FOR THIS CHANNEL
V003            IF THIS FILE HAS SHORTEST LATENCY SO FAR
V003               REMEMBER LATENCY TIME
V003               REMEMBER THIS FILE
V003               REMEMBER PREVIOUS FILE IN QUEUE
V003            END
V003            IF NOT TIME TO BE FAIR AND MORE FILES IN TWQ, TO TWQLUP

V003            SET CONTROLLER TO BUSY STATE
V006            TURN OFF ALL DISK PI'S
V003            REMOVE FILE FROM TW QUEUE FOR CHANNEL
V007            SET UNIT TO T STATE
V007            SET FILE TO T STATE
V006            TURN ON ALL DISK PI'S
V003    STARTT: COMPUTE CHANNEL COMMAND LIST
*       IF THIS IS FIRST BLOCK OF NEW GROUP [JOSFIR] AND OUTPUT [IO] AND NOT MONITOR IO [MNIOBT]
*         IF DUMP MODE, GET CORE ADDRESS OF FIRST BLOCK IN GROUP [DEVMP]
*         IF BUFFERED MODE, GET  CORE ADDRESS OF FIRST BLOCK IN GROUP [ ]
*         COMPUTE 36 BIT EXCLUSIVE OR OF SOME [UNYCKN] OF THE WORDS
*         STORE LOW ORDER [ ] BITS IN BYTE POSITIONS [ ] IN CALLED RETREIVAL POINTER IN DATA BLOCK
*       END
        CLEAR SOFT ERROR COUNT FOR THIS CHANNEL [CHNECT]
V003    STARTE: START DATA TRANSFER
V003            TO SETFAR

V003    KONIDL: SET STATE OF CONTROLLER TO IDLE
V003    SETFAR: IF FAIRNESS COUNT FOR POSITIONING ON THIS CHANNEL COUNTED OUT, RESET IT
V003            IF FAIRNESS COUNT FOR TRANSFERS ON THIS CHANNEL COUNTED OUT, RESET IT
V003            RETURN (DISMISS INTERRUPT)
```

FLOW FOR FILE STRUCTURE UUO - STRUUO

CALL:

```
        MOVE    AC,[XWD N,LOC]
        CALL    AC,[SIXBIT .STRUUO.]  OR CALLI AC,50
```

N IS THE NUMBER OF WORDS IN THE ARGUMENT LIST STARTING AT LOCATION LOC. FOR FIXED LENGTH ARGUMENT LISTS, N MAY BE 0.

THE FIRST WORD SPECIFIES THE FUNCTION REQUESTED.

```
FUNCTION=       0 FOR SETSRC
                1 FOR DEFFST
                2 FOR LOKFST
                3 FOR REMFST
```

SETSRC DEFINES THE FILE STRUCTURE SEARCH LIST FOR THE JOB

NOTE: THE USER MAY DETERMINE HIS CURRENT FILE STRUCTURE SEARCH LIST BY USING THE JOBSTR UUO. HE MAY UNLINK HIMSELF FROM A FILE STRUCTURE BY LEAVING IT OFF THE NEW FILE STRUCTURE SEARCH LIST. THE USER MAY SPECIFY HIS PROJECT LIBRARY OR SYS TO BE INCLUDED IN HIS FILE STRUCTURE SEARCH LIST. PROJECT LIBRARY MUST BE FILE STRUCTURE NAME DSK FOR LEVEL D, AND MUST BE HIS PROJECT NUMBER AND PROGRAMMER NUMBER Ø. MAXIMUM NUMBER OF FILE STRUCTURES, NOT INCLUDING PROJECT LIBRARY OR SYS, IS 9.

ARGUMENTS: THREE WORD BLOCKS:

```
FIRST WORD = SIXBIT FILE STRUCTURE NAME LEFT JUSTIFIED
SECOND WORD = PROJECT, PROGRAMMER NUMBER
THIRD WORD: BIT Ø=1 IF SOFTWARE WRITE PROTECTION REQUESTED
            BIT 1=1 IF NO CREATE ON THIS FILE STRUCTURE REQUESTED
            (UNLESS THE SPECIFIC FILE STRUCTURE IS OPENED)
```

ERROR RETURNS:
```
1 OR MORE FILE STRUCTURES DON'T EXIST
1 OR MORE FILE STRUCTURES SINGLE ACCESS ONLY
TOO MANY ENTRIES
```

TASKS:

```
IF ARGUMENT LIST OUT OF BOUNDS, TO ERR

VERIFY THAT ALL FILE STRUCTURES IN THE USER'S SEARCH LIST EXIST

IF ANY FILE STRUCTURE IS SINGLE ACCESS AND MOUNT COUNT NO Ø, TO ERR

FOR EACH FILE STRUCTURE ON THE NEW SEARCH LIST

   IF THE FILE STRUCTURE IS NOT ON THE USER'S OLD FILE
      STRUCTURE SEARCH LIST, INCREMENT THE MOUNT COUNT

FOR EACH FILE STRUCTURE NO LONGER ON THE SEARCH LIST,
   DECREMENT THE MOUNT COUNT

MAKE THE NEW LIST THE SEARCH LIST

EXIT
```

```
DEFSTR  DEFINES A FILE STRUCTURE FOR THE SYSTEM

ARGUMENTS:        FILE STRUCTURE NAME
                  BITS
                  PACK ID / DRIVE  PAIRS

ERROR RETURNS:    1 OR MORE DRIVES NOT AVAILABLE
                  CANT REDEFINE BECAUSE MOUNT COUNT NOT Ø
                  FILE STRUCTURE ALREADY BEING MOUNTED

TASKS:            IF ARGUMENT LIST OUT OF BOUNDS, TO ERR

                  IF THE FILE STRUCTURE IS ON THE PHYSICAL DEVICE LIST

                     IF N GR 3 (NEW DEFINTION), TO ERR

                     IF THE MOUNT COUNT NOT Ø, TO ERR

                     IF DRIVES ARE BEING MOUNTED

                        CHANGE STATE TO MOUNTED

                        SET MOUNT COUNT TO 1

                        EXIT

                     (REDEFINITION)       SET NEW FLAGS

                     SET MOUNT COUNT TO 1

                     EXIT

                  END
```

```
IF ANY DRIVE IN THE FILE STRUCTURE IS ALREADY IN A FILE
   STRUCTURE OR DOWN, TO ERR

SET FLAGS

APPEND THE FILE STRUCTURE NAME TO THE PHYSICAL DEVICE LIST

APPEND THE FILE STRUCTURE NAME TO THE DISK DEVICE LIST

APPEND THE PACK ID'S TO THE PHYSICAL DEVICE LIST

PUT THE PACK ID AND FILE STRUCTURE NAME IN THE TABLE
   FOR EACH DRIVE

PUT EACH DRIVE IN BEING MOUNTED STATE

SET THE FLAG TO REREAD THE HOME BLOCK FOR EACH DRIVE

SET THE MOUNT COUNT TO Ø

EXIT
```

LOKSTR LOCKS OUT FURTHER LOOKUP'S, ENTER'S, AND INIT'S

ARGUMENTS: FILE STRUCTURE NAME

TASKS: SET FLAG TO LOCK OUT FURTHER LOOKUP'S, ENTER'S AND INIT'S

EXIT

REMSTR REMOVES A FILE STRUCTURE FROM THE SYSTEM

ARGUMENTS: FILE STRUCTURE NAME

TASKS: IF THE FILE STRUCTURE DOESN'T EXIST, EXIT

DELETE THE FILE STRUCTURE NAME FROM ALL FILE STRUCTURE SEARCH LISTS

DELETE THE FILE STRUCTURE NAME FROM THE PHYSICAL DEVICE LIST

DELETE THE PACK ID'S FROM THE PHYSICAL DEVICE LIST

DELETE THE PACK ID'S AND FILE STRUCTURE NAME FROM THE TABLE FOR EACH DRIVE IN THE FILE STRUCTURE

PUT THE DRIVES IN A NOT IN A FILE STRUCTURE STATE

EXIT

FORM OF MOUNT UUO:

```
        MOVE    A,[XWD NUMWRD,LOC]
        CALL    A,[SIXBIT ,MOUNT,]  OR  CALL A,
```

NUMWRD=THE NUMBER OF WORDS IN THE ARGUMENT LIST
LOC=ADDRESS OF THE FIRST ARGUMENT

THE ARGUMENT LIST HAS THE FOLLOWING FORM:

```
        LOC/    FILE STRUCTURE NAME
        LOC+1/  STATUS BITS
        LOC+2/  FIRST PACK ID
        LOC+3/  FIRST DRIVE
        LOC+4/  SECOND PACK ID
        LOC+5/  SECOND DRIVE
                ...
```

IF NUMWRD LESS THAN OR EQUAL TO 2, THERE ARE NO PACKS LISTED. THIS WILL BE THE CASE IF THE FILE STRUCTURE IS BEING MOUNTED OR IS ALREADY MOUNTED.

```
MOUNT:      CHECK BOUNDS OF ARGUMENT LIST
            IF BOUNDS ERROR, TO ERR
            IF NUMBER OF WORDS IN ARGUMENT LIST LESS OR EQUAL 2, TO NOPAKS
            GET FILE STRUCTURE NAME
            SEARCH THE PHYSICAL DEVICE LIST FOR THIS FILE STRUCTURE
            IF THE FILE STRUCTURE ALREADY EXISTS, TO ERR
            DO MNT1 FOR ALL DRIVES SPECIFIED
               IF THIS DRIVE IS IN A FILE STRUCTURE, TO ERR

               IF THIS DRIVE IS DOWN OR BEING MOUNTED, TO ERR
MNT1:       CONTINUE
            SET SINGLE/MULTIPLE ACCESS AND WRITE LOCK FLAGS
            PUT THE FILE STRUCTURE NAME ON THE PHYSICAL DEVICE LIST
            PUT THE FILE STRUCTURE NAME ON THE FILE STRUCTURE SEARCH LIST
               FOR THIS JOB
            PUT THE FILE STRUCTURE NAME ON THE DISK DEVICE LIST
            DO MNT2 FOR ALL PACK, DRIVE PAIRS
               PUT THIS PACK ID IN THE TABLE FOR THIS DRIVE
               PUT THE FILE STRUCTURE NAME IN THE TABLE FOR THIS DRIVE
               PUT THIS PACK ID ON THE PHYSICAL DEVICE LIST
               PUT THE DRIVE IN A BEING MOUNTED STATE
               SET THE FLAG TO REREAD THE HOME BLOCK BEFORE NEXT I/O
MNT2:       CONTINUE
            CLEAR THE FLAG ALLOWING ACCESS TO THE FILE STRUCTURE
            SET THE MOUNT COUNT TO 1
            SKIP RETURN
```

```
NOPAKS:     SEARCH THE PHYSICAL DEVICE LIST FOR THIS FILE STRUCTURE
            IF THE FILE STRUCTURE DOESN'T EXIST, TO ERR
            IF THE FIRST DRIVE OF THE FILE STRUCTURE IS NOT BEING MOUNTED, TO MNTED
            PUT ALL DRIVES IN THIS FILE STRUCTURE IN FINISHED BEING MOUNTED STATE
            ENABLE ACCESS TO THE FILE STRUCTURE
            SKIP RETURN
MNTED:      IF MOUNT COUNT = Ø, TO MNTED1
            IF SINGLE REQUEST, TO ERR
            IF FILE STRUCTURE IS SINGLE ACCESS, TO ERR
            IF WRITE LOCK FLAG DIFFERS FROM REQUEST, TO ERR
            TO MNTED2
MNTED1:     SET SINGLE/MULTIPLE ACCESS FLAG
            SET WRITE LOCK FLAG
MNTED2:     SEARCH FILE STRUCTURE SEARCH LIST FOR THIS JOB FOR
               FILE STRUCTURE NAME
            IF ALREADY THERE, SKIP RETURN
            ADD THE FILE STRUCTURE NAME TO THE FILE STRUCTURE SEARCH LIST
               FOR THIS JOB
            ENABLE ACCESS TO THE FILE STRUCTURE
            INCREMEN THE MOUNT COUNT
            CLEAR BIT SAYING MOUNT COUNT=Ø
            SKIP RETURN
```

5. GLOSSARY

DATA STRUCTURE

SKELETON FILE STRUCTURE

2 HOME BLOCKS

HOME BLOCK OR VITAL STATISTICS BLOCK CONTAINS A PACK ID AND OTHER BASIC INFORMATION. HOME BLOCKS ARE ALWAYS WRITTEN IN KNOWN FIXED LOCATIONS.

MFD

THE MASTER FILE DIRECTORY. A FILE NAMED 11,UFD OF THE SAME FORM AS ALL OTHERS CONSISTING OF RETRIEVAL INFORMATION AND DATA.

SAT

STORAGE ALLOCATION FILE. THE FILE *SAT*.SYS CONSISTING OF RETRIEVAL INFORMATION AND DATA DESCRIBING THE INITIAL ALLOCATION OF SPACE.

SWAPPING FILE

THE OPTIONAL FILE *SWAP*.SYS CONSISTING OF ALLOCATTED SPACE ONLY. CONSIDERING THE SPACE AS A FILE ALLOWS A FILE VERIFICATION PROGRAM TO MAKE THE CORRECT INTERPRETATION THAT SWAPPING SPACE IS BOTH ASSIGNED AND USED IN SOME FILE.

MAINTENANCE AREA

THE OPTIONAL FILE *MAINT*.SYS CONSISTS ONLY OF ALLOCATED SPACE ON THE BRYANT DISK AND DISK PACKS DENOTING THESE BLOCKS RESERVED FOR MAINTENANCE OR DIAGNOSTIC USE.

SPARSE FILE

A FILE WHOSE DATA IS CONTAINED ONLY IN THE SECOND BLOCK OF EVERY GROUP (HOME.SYS AND SAT.SYS). ALL OTHER FILES ARE DENSE SO THAT THE SECOND BLOCK THROUGH THE LAST BLOCK (EXCEPT LAST GROUP OF FILE) CONTAIN DATA.

COMPRESSED UFD POINTER (CUP)

SUPER CLUSTER NUMBER AND UNIT NUMBER FOUND ONLY IN UFD'S. COMPRESSED TO FIT INTO 18 BITS.

ROOT RIB

THE FIRST RIB OF A FILE
THE RETRIEVAL POINTERS POINT TO "GROUP RIB" AND MAY ALSO POINT TO A NEXT "CONTINUATION RIB" OF A PREVIOUS "CONTINUATION FILE"

GROUP RIB

THE RIB IN FRONT OF EVERY GROUP EXCEPT IF A ROOT OR CONTINUOUS RIBS

CONTINUATION RIB

OCCURS ONLY FOR LONG FILES WHEN ROOT RIB RUNS OUT OF ROOM TO HOLD RETRIEVAL POINTERS

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
RRRRRRRRRRRR        EEEEEEEEEEEEEEE     FFFFFFFFFFFFFFF        SSSSSSSSSSSS     TTTTTTTTTTTTTTT     RRRRRRRRRRRR
RRRRRRRRRRRR        EEEEEEEEEEEEEEE     FFFFFFFFFFFFFFF        SSSSSSSSSSSS     TTTTTTTTTTTTTTT     RRRRRRRRRRRR
RRRRRRRRRRRR        EEEEEEEEEEEEEEE     FFFFFFFFFFFFFFF        SSSSSSSSSSSS     TTTTTTTTTTTTTTT     RRRRRRRRRRRR
RRR         RRR     EEE                 FFF                 SSS                     TTT         RRR         RRR
RRR         RRR     EEE                 FFF                 SSS                     TTT         RRR         RRR
RRR         RRR     EEE                 FFF                 SSS                     TTT         RRR         RRR
RRR         RRR     EEE                 FFF                 SSS                     TTT         RRR         RRR
RRR         RRR     EEE                 FFF                 SSS                     TTT         RRR         RRR
RRR         RRR     EEE                 FFF                 SSS                     TTT         RRR         RRR
RRRRRRRRRRRR        EEEEEEEEEEEE        FFFFFFFFFFFF           SSSSSSSSS            TTT         RRRRRRRRRRRR
RRRRRRRRRRRR        EEEEEEEEEEEE        FFFFFFFFFFFF           SSSSSSSSS            TTT         RRRRRRRRRRRR
RRRRRRRRRRRR        EEEEEEEEEEEE        FFFFFFFFFFFF           SSSSSSSSS            TTT         RRRRRRRRRRRR
RRR   RRR           EEE                 FFF                          SSS             TTT         RRR   RRR
RRR   RRR           EEE                 FFF                          SSS             TTT         RRR   RRR
RRR   RRR           EEE                 FFF                          SSS             TTT         RRR   RRR
RRR      RRR        EEE                 FFF                          SSS             TTT         RRR      RRR
RRR      RRR        EEE                 FFF                          SSS             TTT         RRR      RRR
RRR      RRR        EEE                 FFF                          SSS             TTT         RRR      RRR
RRR         RRR     EEEEEEEEEEEEEEE     FFF                 SSSSSSSSSSSS            TTT         RRR         RRR
RRR         RRR     EEEEEEEEEEEEEEE     FFF                 SSSSSSSSSSSS            TTT         RRR         RRR
RRR         RRR     EEEEEEEEEEEEEEE     FFF                 SSSSSSSSSSSS            TTT         RRR         RRR

FFFFFFFFFFFFFFF     LLL                    000000000
FFFFFFFFFFFFFFF     LLL                    000000000
FFFFFFFFFFFFFFF     LLL                    000000000
FFF                 LLL                 000         000
FFF                 LLL                 000         000
FFF                 LLL                 000         000
FFF                 LLL                 000         000
FFF                 LLL                 000         000
FFF                 LLL                 000         000
FFFFFFFFFFFF        LLL                 000         000
FFFFFFFFFFFF        LLL                 000         000
FFFFFFFFFFFF        LLL                 000         000
FFF                 LLL                 000         000
FFF                 LLL                 000         000
FFF                 LLL                 000         000
FFF                 LLL                 000         000
FFF                 LLL                 000         000
FFF                 LLL                 000         000
FFF                 LLLLLLLLLLLLLLL        000000000
FFF                 LLLLLLLLLLLLLLL        000000000
FFF                 LLLLLLLLLLLLLLL        000000000
```

REFSTR.FLO CREATED: 14-APR-1970 00:00 PRINTED: 02-JUL-1970 16:48

```
RFRESH: ASK "TYPE STR NAME TO REFRESH (CR IF NONE)"[CALL ASKSTR]
        IF JUST CR, RETURN

        READ 1 HOME BLOCK TO GET LENGTHS OF FILES TO CREATE

;CREATE SAT,SYS

        ALLOCATE CORE FOR READING BAT BLOCKS
        ALLOCATE CORE FOR CREATING SAT BLOCKS (SIZE OF 1 DISK BLOCK)
        ALLOCATE CORE FOR SAT.SYS RIB AND HOME,SYS RIB
        SET UP RIB IN CORE FOR SAT,SYS
        SET UP RIB IN CORE FOR HOME,SYS
        DO REDUNI FOR ALL UNITS IN STR
          READ BAT BLOCKS FOR THIS UNIT [CALL REDRUN]
          STORE NEW UNIT RETIEVAL POINTER IN SAT,SYS RIB IN CORE
          STORE NEW UNIT POINTER IN HOME,SYS RIB IN CORE

;CREATE SAT BLOCKS

          DO MRKSAT, FOR ALL SAT BLOCKS FOR THIS UNIT
            SET TO ONES THE NON-EXISTENT RESIDUE BITS IN LAST WORDS
                [WORD SABBIT+SABTAL/36, BITS REMAINDER SABTAL/36-35
                  ,PLUS THE REST OF THE WORDS IN THE BLOCK]

;MARK BAD BLOCKS

            DO SCNBAT FOR ALL BAD REGIONS IN BAT BLOCK

              MARK OFF BAD CLUSTERS IN THIS BAD REGION IN THIS SAT
                        [MAX (BEGSAT, ((BAYLBN-1)/UNYBPC+1)) UP TO
                        MIN (BEGSAT+SVSBTL-1,((BAYLBN+BAYBKN-2)/UNYBPC+1))]
SCNBAT:     CONTINUE

;MARK SWAPPING AREA

            MARK OFF SWAPPING CLUSTER IN SWAPPING SPACE IN THIS SAT
                        [MAX (BEGSAT,((UNISLB-1)/UNYBPC+1)) UP TO
                        MIN (BEGSAT+SVSBTL-1,((UNISLB+UNYKRS*2*BLKSPK-2)/UNYBPC+1))]

            IF THIS IS FIRST SAT ON THIS UNIT [BEGSAT=0]

;ALLOCATE HOME BLOCK GROUP

              TAKE BLOCK LBNHOM FOR THE HOME BLOCK [CALL TAKBLK]
              IF CANT GET IT, TO ERR
              STORE RETRIEVAL POINTER IN HOME,SYS RIB (IN CORE)
              SET PREVIOUS BLOCK=LBNHOM FOR FOLLOWING LOOP
              DO GETBLK, FOR BLOCKS LBNBAT,LBNISW,LB2HOM,LB2BAT,LB2ISW
                TRY TO GET CURRENT BLOCK
                IF CANT GET BLOCK
                  IF CURRENT BLOCK IS NOT IN SAME CLUSTER AS PREVIOUS, TO ERR
                ELSE IF CLUSTER JUST ALLOCATED IMMEDIATELY FOLLOWS PREVIOUS CLUSTER
                  INCREMENT CLUSTER COUNT IN TRERIEVAL POINTER
                    ELSE STORE NEW RETRIEVAL POINTER IN HOME,SYS RIB IN CORE
                  END
GETBLK:       CONTINUE
            END
```

```
;ALLOCATE A BLOCK FOR THIS SAT IN THE REGION IT REPRESENTS

            TAKE 1 BLOCK FOR THIS SAT BLOCK
            STORE RETIEVAL POINTER IN SAT,SYS RIB
            WRITE OUT SAT BLOCK IN FIRST BLOCK OF CLUSTER [CALL OMNWRT]
            WRITE ALL OTHER BLOCKS IN THIS CLUSTER WITH ONES [CALL OMNWRT]
            IF INDEX OF THIS SAT LT SATS IN CORE FOR THIS UNIT [UNYSIC],
              READ THIS SAT INTO MONITOR CORE [CALL REDSAT]
MRKSAT:   CONTINUE
REDUNI: CONTINUE

        ALLOCATE SPACE FOR SYS UFD ON 1ST UNIT (LENGTH FROM HOME BLOCK,
          +2 FOR RIBS - NEED LOG BLK NUM OF 1ST UFD BLOCK FOR RIB OF FILES WE CREATE)

;ALLOCATE RIBS FOR SAT,SYS AND HOME,SYS

        TAKE BLOCK FOR 1ST SAT,SYS RIB ON 1ST UNIT [CALL TAKBLK]
        STORE RETRIEVAL POINTER IN SAT,SYS RIB IN CORE
        REMEMBER LOGICAL BLOCK NUMBER OF 1ST SAT,SYS RIB
        TAKE 1 BLOCK FOR 2ND SAT,SYS RIB ON LAST UNIT
        STORE RETRIEVAL POINTER IN SAT,SYS RIB IN CORE
        WRITE FIRST AND LAST RIB [CALL OMNWRT]

        TAKE 1 BLOCK FOR 1ST HOME,SYS RIB ON 1ST UNIT [CALL TAKBLK]
        STORE RETRIEVAL POINTER IN HOME,SYS RIB IN CORE
        REMEMBER LOGICAL BLOCK NUMBER OF 1ST HOME,SYS RIB
        TAKE 1 BLOCK FOR 2ND HOME,SYS RIB ON LAST UNIT [CALL TAKBLK]
        WRITE 1ST AND 2ND RIB FOR HOME,SYS [CALL OMNWRT]

        ZERO OUT ALL BLOCKS IN HOME,SYS WHICH ARE NOT DATA (DUE TO LARGE CLUSTER SIZE)
```

```
        ALLOCATE RIB FOR MAINT.SYS
        DO MANLOP, FOR ALL UNITS IN THIS STR
          ALLOCATE SPACE FOR MAINT.SYS (DEVICE DEPENDENT) ON THIS UNIT
MANLOP: CONTINUE
        ALLOCATE 2ND RIB FOR MAINT.SYS
        STORE CFP, AND WRITE RIBS [SET, RIPNDL & RIPNCN]

        ALLOCATE RIB + SPACE (CRSBKN) + RIB FOR CRASH.SAV(CONSECUTIVE)
        STORE NEW LOC OF CRASH.SAV (FIRST DATA BLOCK RATHER THAN RIB) FOR HOME BLOCK[HOMCRS]
        WRITE ZEROS IN ALL DATA BLOCKS

        ALLOCATE RIB + SPACE (SNPBKN) + RIB FOR SNAP.SYS (CONSECUTIVE)
        STORE NEW LOC OF SNAP.SPS FOR HOME BLOCK [HOMSNP]
        WRITE ZEROS IN ALL DATA BLOCKS
        STORE CFP AND WRITE RIBS

        ALLOCATE RIB + SPACE (RCVBKN) + RIB (CONSECUTIVE)
        STORE NEW LOC OF RECOV.SYS FOR HOME BLOCK [HOMRCV]
        WRITE ZEROS IN ALL DATA BLOCKS
        STORE CFP AND WRITE RIBS

        ALLOCATE SPACE FOR 1ST SWAP.SYS RIB
        WRITE ZEROS IN UNUSED BLOCKS IN FIRST CLUSTER
        DO SCNUNI, FOR ALL UNITS IN STR
          IF THIS UNIT HAS SWAP SPACE
            STORE NEW UNIT RETRIEVAL POINTER IN RIB
            STORE NEW RETRIEVAL POINTER(S)
          END
SCNUNI: CONTINUE
        ALLOCATE SPACE FOR 2ND SWAP.SYS RIB
        WRITE RIBS AND STORE CFP (MARK RIB NOT DEL OR CHNG NAM [RIPNDL,RIPNCN])

        ALLOCATE SPACE FOR 1ST BADBLK.SYS RIB
        WRIT ZEROS IN UNUSED BLOCKS IN FIRST CLUSTER
        DO SCNSTK, FOR ALL UNITS IN THIS STR [STRUNI-UNISTR]
          CONVERT BAD REGION TO ONE OR MORE RETRIEVAL POINTERS AND STORE IN RIB
SCNSTK: CONTINUE
        ALLOCATE SPACE FOR 2ND BADBLK.SYS RIB
        WRITE RIBS AND STORE CFP (SET RIPNDL AND RIPNCN BITS)
```

```
        ALLOCATE SPACE FOR RIB + MFD + RIB (NEED LOG BLK NUM OF
          1ST DATA BLOCK OF MFD TO STORE IN RIB OF UFD'S)
        WRITE RIBS AND DATA FOR SYS UFD [ENTRIES FOR FILES JUST GENERATED, Ø FILL]

        ALLOCATE SPACE FOR RIB + 3,3,UFD [PRUFSZ BLOCKS] + RIB
        STORE NAME, EXT, AND CFP IN MFD
        WRITE RIBS AND ZEROS FOR DATA

        WRITE RIBS AND DATA FOR MFD [ENTRIES FOR UFD'S JUST CREATED]
        REMEMBER 1ST RETRIEVAL POINTER OF MFD FOR HOME BLOCK

        DO SCNUNI, FOR ALL UNITS IN STR
          READ HOME BLOCKS (CALL REDRUN)
          UPDATE LOGICAL BLOCK OF 1ST RIB OF SYSTEM FILES [HOMSAT,HOMSWP,HOMMNT,        ]
          WRITE HOME BLOCKS
SCNUN7: CONTINUE

        STORE SYSPPN IN REFLAG
        REWRITE SAT BLOCKS WHICH HAVE CHANGED [CALL WTSATS]

        EXIT
```

100-221-023-00

THE FILTST LANGUAGE

D. PLUMER 6 MARCH 70

1. TABLE OF CONTENTS

2. OVERVIEW

FILTST IS A MACRO SOURCE PROGRAM WHICH DEFINES A MACRO LEVEL LANGUAGE INTENDED AS A CONVENIENT MEDIUM IN WHICH TO WRITE PROCEDURES WHICH EXERCISE AND TEST THE FILE HANDLING CAPABILITIES OF THE PDP-10 TIME-SHARING MONITOR. IN ORDER THAT THIS MACRO LANGUAGE POSSESS THE GENERALITY AND FLEXIBILITY OF THE STANDARD USER MODE TECHNIQUES FOR DOING FILE I/O AT ASSEMBLY LEVEL, THE LANGUAGE IS NECESSARILY QUITE CLOSE TO STANDARD USER-MODE I/O, AND IS INTENDED TO BE USED BY SOMEONE THOROUGHLY FAMILIAR WITH THE STRUCTURE AND CALLING SEQUENCES OF THE STANDARD UUOS. HOWEVER, IT IS INTENDED TO RELIEVE THE PROGRAMMER OF ATTENTION TO BIT AND WORD LEVEL DETAIL AND TO PERMIT HIM TO SPECIFY REASONABLY COMPLEX DISK OPERATIONS WITH SIMPLICITY AND COMPACTNESS AT BOTH SOURCE AND OBJECT LEVELS.

FILTST IS NOT A "CONDITIONAL LANGUAGE. THAT IS, IT IS NOT INTENDED THAT FILE OPERATIONS BE "TRIED TO SEE WHAT HAPPENS", WITH SUBSEQUENT ACTION DEPENDING UPON A PREVIOUS RESULT. RATHER, IT IS INTENDED THAT THE USER WHO PREPARES ANY TEST PROCEDURE HAVE AN UNDERSTANDING OF EXACTLY WHAT SHOULD HAPPEN WHEN THE PROCEDURE IS EXECUTED. FILTST WILL ATTEMPT TO PROVIDE DIAGNOSTIC INFORMATION WHENEVER A RESULT IS CONTRARY TO AN INDICATED EXPECTATION.

THE USE OF FILTST TYPICALLY INVOLVES THE PREPARATION OF AN AUXILIARY TEXT FILE IN THE FILTST LANGUAGE CONSISTING OF A NUMBER OF INDEPENDENT DEFINITIONS OF TEST PROCEDURES (WHICH MAY CALL EACH OTHER RECURSIVELY TO ANY DEPTH) AND A LIST OF WHICH OF THESE PROCEDURES ARE TO BE EXECUTED AND IN WHAT ORDER. THIS AUXILIARY FILE IS THEN ASSEMBLED WITH FILTST AND THE RESULTING SINGLY BINARY FILE IS LOADED AND EXECUTED. THE RUNNING TO COMPLETION OF THIS EXECUTION WITH NO UNUSUAL OUTPUT, INDICATES THAT ALL OF THE TESTS SPECIFIED IN THE AUXILIARY FILE RAN WITH SPECIFIED EXPECTATIONS BEING MET. ANY ERROR CONDITION, UNEXPECTED RETURN FROM A UUO, OR UNEXPECTED VALUE RETURNED BY THE MONITOR WILL RESULT IN AN ERROR MESSAGE AT THE CONSOLE. THE USER IS THEN FREE TO PROCEDE FROM THE ERROR CONDITION, IGNORING IT, OR ATTEMPT TO ANALYZE THE PROBLEM BEFORE CONTINUING.

(2. CONT'D)

WHEN SOME ERROR DOES OCCUR, USUALLY BECAUSE SOME EXPECTATION HAS NOT BEEN MET, THE USER IS FACED WITH THE PROBLEM OF DETERMINING WHAT WENT WRONG. UNLESS THE TEST HAS RUN SUCCESSFULLY WITH THE SAME MONITOR BEFORE, THE FIRST QUESTION IS LIKELY TO BE: "HAVE I DISCOVERED A BUG IN THE DISK SERVICE, OR IS THERE A BUG IN BY TEST PROCEDURE?" UNFORTUNATELY, FILTST IS HARD PRESSED TO DETERMINE THE DIFFERENCE AND THE USER MUST RELY UPON ANALYSIS OF HIS PROCEDURES IN THE LIGHT OF WHAT HAPPENED. IT IS FILTST'S MAIN ORDER OF BUSINESS TO SHOW THE USER WHAT DID HAPPEN. THE ERROR TRACE PRINTED BY FILTST FOLLOWING ANY ERROR MESSAGE WILL DISCLOSE THE EXACT PATH TAKEN THROUGH THE USERS NESTED PROCEDURES AND AT JUST WHICH STEP OF WHICH PROCEDURE THE ERROR OCCURRED. IT ALSO SHOWS THE NUMBER OF TIMES EACH PROCEDURE HAS BEEN CALLED SUCCESSFULLY (MORE DETAIL BELOW). IT IS HOPED THAT THIS INFORMATION AND A SOURCE LISTING OF THE AUXILIARY FILE OF PROCEDURE DEFINITIONS WILL BE SUFFICIENT IN MANY CASES FOR COMPLETE ANALYSIS OF THE PROBLEM. FOR THOSE CASES IN WHICH IT IS NOT SUFFICIENT, A COPY OF DDT IS LOADED WITH FILTST. ENOUGH OF FILTST'S INTERNAL STRUCTURE, USE OF ACS ETC. IS PRESENTED BELOW SO THAT DDT CAN BE USED EFFECTIVELY TO DISCOVER MORE ABOUT THE CAUSE OF ANY ERROR MESSAGE.

MORE GENERALLY, THE OPERATION OF "PREPARING THE AUXILIARY FILE OF TEST PROCEDURES" IS LIKELY TO BE THE EXPANDING OF ONE OR MORE GROWING PROCEDURE FILES. IT IS INTENDED THAT VARIOUS OPERATING VERSIONS OF FILTST, EACH CONTAINING ITS OWN SUB SET OF PROCEDURES, BE SAVED IN CORE IMAGE FORM AFTER BEING DEBUGGED, TO BE USED REPEATEDLY TO TEST THE CONTINUED INTEGRITY OF THE MONITOR'S DISK SERVICE WHENEVER CHANGES ARE MADE TO IT.

3. WRITING TEST PROCEDURES

TEST PROCEDURES IN THE FILTST LANGUAGE ARE USUALLY SHORT AND QUITE COMPACT. THEY CONSIST MAINLY OF MACRO CALLS, SOME OF WHICH TAKE ONE OR MORE SIMPLE ARGUMENTS. UPON RARE OCCASIONS IT IS ALSO CONVENIENT TO INCLUDE A FEW BASIC PDP-10 INSTRUCTIONS WHICH REFERENCE A SMALL SET OF ACCUMULATORS KNOWN TO BE AVAILABLE AND A FEW ADDRESSES WITH PREDICTABLE MNEMONICS.

TO THE EXTENT THAT PROCEDURES ARE COMPACT, THEY ARE VERY EASY TO WRITE; UNFORTUNATELY, THEY ARE CORRESPONDINGLY DIFFICULT TO UNDERSTAND WITH RESPECT TO OVERALL INTENT UNLESS GENEROUSLY COMMENTED. A RECOMMENDED CONVENTION WHICH RESULTS IN A CLEAN CREF LISTING FOR LATER USE DURING ERROR ANALYSIS, IS THAT COMMENTS APPEAR IN BLOCKS PRECEDING PROCEDURE DEFINITIONS RATHER THAN ON INDIVIDUAL SOURCE LINES WITHIN THE PROCEDURES. (COMMENTS ON LINES WITH MACRO CALLS ARE DELETED DURING ASSEMBLY. THIS IS TRUE BECAUSE IT IS FELT THAT A SOURCE-LIKE LISTING CONTAINING ONLY MACRO CALLS AND ARGUMENTS IS A MUCH MORE CONVENIENT TOOL FOR DEBUGGING THAN A LISTING IN WHICH MACRO EXPANSIONS ARE SHOWN. IN FACT SOME AMOUNT OF TROUBLE IS TAKEN TO AVOID ALL UNNECESSARY OUTPUT IN CREF LISTINGS.)

3.1 FILTST DATA BASE

SOME MONITOR UUO'S (EG, OPEN, LOOKUP, SEARCH, ENTER, RENAME) TAKE ARGUMENTS AND RETURN VALUES IN DATA BLOCKS WITHIN A JOBS LOW SEGMENT. A LARGE PART OF THE INTRICACY OF DOING DISK I/O IN USER MODE INVOLVES ESTABLISHING, SETTING, TESTING AND CLEARING THE RIGHT PARTS OF THESE UUO DATA BLOCKS. MANY OF THE FILTST MACROS EXIST FOR THE PURPOSE OF MAKING THIS JOB EASY. AS A COMPROMISE BETWEEN SIMPLICITY AND SIZE, THE FOLLOWING IMPLEMENTATION DECISIONS WERE REACHED:

1. THERE WILL BE A MAXIMUM NUMBER OF SOFTWARE CHANNELS WHICH ANY SINGLE JOB RUNNING FILTST CAN USE SIMULTANEOUSLY; THIS PARAMETER HAS BEEN ARBITRARILY SET AT FIVE (5) INITIALLY.

2. ALLOCATION OF CORE TO UUO DATA BLOCKS WILL NOT BE DYNAMIC; RATHER, SUFFICIENT DATA SPACE FOR THE MAXIMUM NUMBER OF SIMULTANEOUS CHANNELS WILL BE GENERATED AT ASSEMBLY TIME.

FOR EACH POTENTIALLY ACTIVE CHANNEL, THERE EXISTS IN FILTST'S LOW SEGMENT A BLOCK OF LOCATION CONSISTING OF THREE WORDS FOR USE WITH THE OPEN UUO, A SINGLE EXPANDABLE BLOCK (CURRENTLY SET AT TEN WORDS) FOR USE WITH ALL EXTENDED LOOKUP - SEARCH - ENTER - RENAME, UUOS, AND THREE OR MORE ADDITIONAL PARAMETER WORDS (TO BE EXPLAINED BELOW). THE GENERATION OF MNEMONICS AND MACROS FOR EACH OF THE WORDS IN THIS "PER-CHANNEL" DATA BASE IS CONTROLLED BY THE DEFINITION OF THE ARGS MACRO TO BE FOUND ON PAGE 4 OF A LISTING OF FILTST. IT IS INTENDED THAT ONLY THE DEFINITION OF THIS ARGS MACRO NEED BE CHANGED IN ORDER TO REARRANGE OR EXPAND THE ARRAYS OF ARGUMENTS AND VALUES WHICH ACCOMPANY THE UUOS. IN FACT, THE ONLY SOURCE LEVEL DIFFERENCE BETWEEN THE VERSION OF FILTST FOR LEVEL C DISK SERVICE AND THAT FOR LEVEL D DISK SERVICE, IS A REARRANGEMENT OF FIVE ENTRIES (ENTRIES FOUR THROUGH EIGHT) IN THE ARGS MACRO.

3.2 THE SETXXX AND SELXXX MACROS

FOR EACH WORD IN THE "PER-CHANNEL" DATA BASE, THERE IS A MACRO OF THE FORM SETXXX WHERE THE LAST THREE CHARACTERS MATCH SOME THREE CHARACTER MNEMONIC WITHIN THE DEFIITION OF ARGS. BY MEANS OF EACH OF THESE MACROS, THE CORRESPONDING LOCATION IN THE DATA BASE CAN BE "SET". IN ADDITION TO ONE DATA BLOCK FOR EACH CHANNEL, THERE IS ONE "JOB STANDARD" DATA BLOCK WHICH IS INTENDED TO CONTAIN DEFAULT ARGUMENTS TO BE USED ON ANY USER CHANNEL WHENEVER A REQUIRED ARGUMENT HAS NOT BEEN PREVIOUSLY SET. THE VARIOUS ENTRIES IN THIS "JOB STANDARD" DATA BLOCK ARE SET BY MEANS OF SELXXX MACROS. FOR EVERY SETXXX MACRO, THERE IS A CORRESPONDING SELXXX "SELECT" MACRO. SELXXX MACROS ARE CHANNEL INDEPENDENT, WHILE SETXXX MACROS ALWAYS APPLY TO SOME SPECIFIC "CURRENT" CHANNEL (AS DESCRIBED BELOW).

3.2.1 MACRO ARGUMENTS

WITH CERTAIN SPECIAL EXCEPTIONS, MACRO ARGUMENTS ARE TYPICALLY EITHER DECIMAL NUMBERS (IMMEDIATE MODE) OR CHARACTERS WHICH SPECIFY AN ADDRESS WHOSE CONTENTS COMPRISES THE ARGUMENT. IN THE CASE OF SETXXX AND SELXXX MACROS, OBSERVATION OF THE ARGS DEFINITION AND THE COMMENTS WHICH PRECEDE IT WILL REVEAL WHICH OF THE MACROS ARE IMMEDIATE AND WHICH TAKE ADDRESS, OR SPECIAL ARGUMENTS. AS A COMPROMISE BETWEEN GENERALITY AND SIMPLE USEFULNESS, THOSE DATA BASE ENTRIES WHICH REQUIRE ADDRESS ARGUMENTS (I.E., THOSE OF MORE THAN 18 BITS) ARE RESTRICTED TO A SET OF CHOICES GIVEN IN TABLES ON PAGE 8 OF FILTST. BY CONVENTION, A CHOICE IS MADE BY GIVING AS AN ARGUMENT TO A SETXXX OR SELXXX MACRO, JUST THE CHARACTERS FOLLOWING THE PERIOD IN LOCATION NAMES SHOWN IN THESE TABLES. THE TABLES CAN CERTAINLY BE EXPANDED, MODIFIED WITH DDT, OR PERHAPS (AS A POTENTIAL FUTURE FEATURE) SET CONVERSATIONALLY IN AN INITIALIZATION DIALOGUE.

3.3 THE SELCHN MACRO AND EXAMPLES

WITH THE EXCEPTION OF THE SELXXX MACROS WHICH MODIFY THE SINGLE CHANNEL INDEPENDENT DATA BASE, AND THE EXPECT MACRO (TO BE DESCRIBED BELOW) ALL MACROS PERTAIN TO SOME SOFTWARE CHANNEL. WHICH CHANNEL A MACRO PERTAINS TO IS NOT SPECIFIED IN THE MACRO CALL, BUT RATHER IS DETERMINED BY THE MOST RECENTLY EXECUTED SELCHN ("SELECT CHANNEL") MACRO. THIS IS SO THAT WHOLE PROCEDURES MAY BE WRITTEN INDEPENDENT OF A SOFTWARE CHANNEL NUMBER AND THEN CALLED AFTER AN APPROPRIATE SELCHN MACRO. SELCHN IS A SPECIAL MACRO WHICH "SELECTS" THE CHANNEL TO WHICH ALL SUBSEQUENT MACROS (AND THE UUOS THEY STIMULATE) WILL APPLY UNTIL THE EXECUTION OF ANOTHER SELCHN MACRO. SOME EXAMPLES MAY CLARIFY THE ABOVE DISCUSSION.

MACRO SEQUENCE

```
SELNAM A        SELECT FILENAME FILTSA AS THE CHANNEL
                INDEPENDENT DEFAULT FILENAME IN THE
                "JOB STANDARD" DATA BASE (6TH WORD)
SELCHN2         SET STRUCTURE NAME TO DPAØ IN DATA
SETSTR AØ       BASE FOR CHANNEL 2 (2ND WORD)
SETDIR Ø        SET THE DIRECTORY NAME TO ZERO (TO
                USE CURRENT PJ, PG) STILL IN DATA
                BASE FOR CHANNEL 2,
                (2ND WORD IN LOOKUP ETC, BLOCK = OR
                5TH WORD IN CHANNEL DATA BASE)
SELCHN Ø        SET THE NUMBER OF ARGUMENTS PASSED
SETARG 4        IN SUBSEQUENT LOOKUP ETC, UUOS TO 4
                IN CHANNEL Ø DATA BASE (4TH WORD)
```

IT CAN BE SEEN THAT BY MEANS OF SETXXX MACROS FOLLOWING SOME SELCHN MACRO, ANY ENTRY IN THE UUO DATA ARRAYS CAN BE SET FOR ANY CHANNEL. NOT ALL OF THE ENTRIES DEFINED IN THE ARGS MACRO HAVE BEEN EXPLAINED SUFFICIENTLY AT THIS POINT; A COMPLETE DESCRIPTION OF EACH APPEARS IN THE LIST OF MACROS IN APPENDIX I. ALSO, THE PURPOSE OF THE CHANNEL INDEPENDENT SELXXX MACROS HAS NOT YET BEEN SUFFICIENTLY EXPLAINED. READ ON.

3.4 OTHER DATA BASE MACROS

IN ADDITION TO THE SETXXX MACROS FOR SETTING ENTRIES IN THE "PER CHANNEL" DATA BASE, AND SELXXX MACROS FOR "SELECTING" ENTRIES IN THE "JOB STANDARD" DATA BASE, FIVE OTHER MACROS ARE USEFUL IN SETTING DATA BASE ENTRIES:

CLRVAL WILL CLEAR THE ENTIRE DATA BASE FOR A PARTICULAR CHANNEL (DEPENDING AS USUAL UPON THE MOST RECENT SELCHN MACRO). IF AN ARGUMENT IS SUPPLIED TO CLRVAL, IT IS A DECIMAL NUMBER SPECIFYING WHERE THE CLEARING OPERATION IS TO BEGIN. FOR EXAMPLE, CLRVAL 7 WILL CLEAR ALL OF THE EXTENDED ARGUMENTS IN THE CURRENT CHANNEL LKP-ENT-REN BLOCK. (I.E., IT WILL NOT CLEAR THE FIRST 7 WORDS - THREE IN THE OPEN BLOCK AND FOUR IN THE LKP-ENT-REN BLOCK. ANY SINGLE DATA BASE WORD CAN BE CLEARED BY GIVING Ø AS AN ARGUMENT TO THE APPROPRIATE SETXXX MACRO, NO MATTER WHAT TYPE OF ARGUMENT THE PARTICULAR MACRO USUALLY TAKES.

RSTEXT AND RSTPRT WILL CLEAR ALL BUT THE EXTENSION AND PROTECTION FIELDS RESPECTIVELY IN THE CORRESPONDING WORDS IN THE CURRENT CHANNEL DATA BASE. THAT IS, THEY WILL CLEAR ANY VALUES THE MONITOR MAY HAVE RETURNED IN OTHER FIELDS IN THESE WORDS.

THE FORSET MACRO IS USED TO "FORCE THE SETTING" OF SPECIFIC WORDS IN THE CURRENT CHANNEL DATA BASE USING THE CORRESPONDING WORDS IN THE JOB STANDARD DATA BASE AS A SOURCE. THE MACRO TAKES AN ARBITRARY LENGTH STRING OF ARGUMENTS ENCLOSED IN ANGLE BRACKETS AND SEPARATED BY COMMAS; EACH ARGUMENT IS ONE OF THE THREE CHARACTER MNEMONICS FOUND IN THE ARGS MACRO. FOR EXAMPLE, FORSET <ARG, DIR> WILL FORCE THE FIRST TWO WORDS IN THE LKP-ENT-REN BLOCK FOR THE CURRENT CHANNEL TO BE SET FROM THE JOB STANDARD DATA BASE WHICH WOULD PRESUMABLY HAVE BEEN SET PREVIOUSLY BY SELARG AND SELDIR MACROS. THE INTENTION OF THIS MECHANISM IS THAT PROCEDURES MAY BE WRITTEN INDEPENDENT OF PARTICULAR UFDS, FILENAMES, STRUCTURES, ETC., AND THEN CALLED (FROM WITHIN SOME OTHER PROCEDURE) AFTER APPROPRIATE SELXXX MACROS.

(3.4 CONT'D)

FINALLY (FOR THIS SECTION), THE INSURE MACRO IS IDENTICAL TO FORSET, EXCEPT THAT THE SPECIFIED DATA BASE ARGUMENTS ARE SET ONLY IF THEY HAVE NOT ALREADY BEEN SET (BY A SETXXX, FORSET OR INSURE MACRO) SINCE THE MOST RECENT RELEASE UUO ON THE CURRENT CHANNEL. A RANDOM FACT WHICH BELONGS FURTHER ALONG IN THIS DOCUMENTATION BUT MIGHT CLARIFY THE USE OF THE INSURE MACRO IS THAT WHENEVER AN OPEN UUO IS EXECUTED FOR A PARTICULAR CHANNEL, FILTST DOES AN AUTOMATIC

```
INSURE  <MOD, STR, BUF>
```

AND BEFORE ANY UUO WHICH REFERENCES THE LKP-ENT-REN BLOCK, FILTST DOES AN AUTOMATIC

```
INSURE  <ARG, DIR, NAM, EXT, PRT>
```

(THIS DOES NOT IMPLY THAT THE USER CANNOT SET THE NUMBER OF ARGUMENTS FOR LKP-ENT-REN UUOS LESS THAN FOUR).

3.5 SPECIFYING EXPECTATIONS

SO FAR, THE ONLY MACROS DISCUSSED HAVE BEEN FOR MODIFYING AN IN CORE DATA BASE; NONE HAVE RESULTED IN INVOLVING THE MONITOR. FILTST PROCEDURES ARE INTENDED TO TAKE TWO PREPARATORY ACTIONS BEFORE EXECUTING ANY MACROS WHICH RESULT IN UUOS. THE FIRST OF THESE ACTIONS IS THE PREPARATION OF THE CHANNEL DATA BASE AS DESCRIBED ABOVE; THE SECOND IS THE PREDICTING OF EXPECTED RESULTS. A SINGLE MACRO, CALLED EXPECT, EXISTS FOR THIS PURPOSE.

EXPECT TAKES TWO ARGUMENTS EITHER OF WHICH MAY BE MISSING. THE FIRST OF THESE SPECIFIES RETURN EXPECTATIONS FOR ANY SUBSEQUENT UUOS WHICH HAVE MORE THAN ONE RETURN. THE ARGUMENT MAY BE A SINGLE CHARACTER FROM THE SET GIVEN BELOW, OR A DECIMAL NUMBER. IT IS INTERPRETED AS FOLLOWS:

ARGUMENT	INTERPRETATION
X	EITHER UUO RETURN ACCEPTABLE
N	NORMAL (SKIP) RETURN EXPECTED
E	ERROR (FIRST) RETURN EXPECTED BUT ANY ERROR CODE IS ACCEPTABLE
<NUMBER>	ERROR (FIRST) RETURN EXPECTED AND ERROR CODE MUST BE <NUMBER>

THIS FIRST ARGUMENT TO EXPECT IS RELEVANT ONLY FOR SUBSEQUENT UUOS OF TWO RETURNS. SINGLE RETURN UUOS DO NOT REQUIRE THIS FORM OF EXPECTATION PREDICTION.

(3.5 CONT'D)

THE SECOND ARGUMENT TO EXPECT IS RELEVANT FOR ALL SUBSEQUENT UUOS. IT SPECIFIES THE EXPECTED STATES OF THE FIVE STATUS BITS RETURNED IN BIT POSITIONS 18 - 22 OF THE GETSTS UUO. UNLESS ERROR CHECKING IS SUPPRESSED FOR SOME REASON (AS IT USUALLY IS IN CASES WHERE A SINGLE PROCEDURE MACRO REQUIRES THE EXECUTION OF MORE THAN ONE UUO), A GETSTS IS DONE AFTER EVERY OTHER UUO, AND THE FOUR ERROR BITS PLUS THE END-OF-FILE BIT ARE CHECKED AGAINST EXPECTATIONS.

THE MACRO ARGUMENT MUST CONSIST OF STRING OF FIVE CHARACTERS, EACH OF WHICH MUST BE EITHER Ø, 1 OR X. EACH CHARACTER CORRESPONDS TO ONE OF THE FIVE STATUS BITS (FROM LEFT TO RIGHT) WHERE Ø AND 1 SPECIFY THAT THE BIT MUST BE EITHER A ZERO OR A ONE, AND X SPECIFIES THAT EITHER ZERO OR ONE IS ACCEPTABLE FOR THAT BIT.

NOTICE THAT THE EXPECT MACRO IS CHANNEL INDEPENDENT, AND SPECIFIES EXPECTATION FOR ALL SUBSEQUENT UUOS, NO MATTER WHAT CHANNEL THEY ARE EXECUTED ON. IT IS INTENDED THAT EXPECT MACROS BE EXECUTED AS OFTEN AS ECPECTATIONS CHANGE. DURING MOST PARTS OF MOST PROCEDURES, THE EXPECTATIONS ARE LIKLY TO BE

```
EXPECT          N,ØØØØØ
```

ONE SUCH MACRO IS SUFFICIENT UNTIL SOME ERROR RETURN OR ERROR BIT IS TO BE EXPECTED. IF EITHER ARGUMENT TO AN EXPECT MACRO IS MISSING (I.E. BLANK), THE PREVIOUS EXPECTATIONS FOR THAT ARGUMENT ARE NOT ALTERED.

3.6 THE EXECUTE MACROS

ONCE THE DATA BASE FOR A PARTICULAR CHANNEL AND THE APPROPRIATE EXPECTATIONS HAVE BEEN ESTABLISHED BY MEANS OF SETXXX (OR SELXXX AND FORSET) AND EXPECT MACROS, IT IS THEN REASONABLE TO PROVOKE ACTION BY MEANS OF THE XCTXXX MACROS, WITH THE EXCEPTION OF INPUT AND OUTPUT WHICH ARE DESCRIBED IN DETAIL BELOW, THERE IS ONE SIMPLE XCT MACRO FOR EACH OF THE POSSIBLE UUOS, FOR THE UUOS OPEN (INIT IS NEVER USED), LOOKUP ENTER, RENAME, SEARCH, SEEK AND RELEASE, THE CORRESPONDING MACROS ARE XCTOPN, XCTLKP, XCTENT, XCTREN, XCTSRC, XCTSEK, AND XCTRLS. NONE OF THESE MACROS TAKE ARGUMENTS SINCE THE PROPER CHANNEL NUMBER IS DETERMINED BY THE MOST RECENT SELCHN MACRO, AND ALL OTHER PARAMETERS HAVE BEEN DETERMINED BY PREVIOUS PREPARATORY MACROS (AS DESCRIBED IN THE PARAGRAPHS ABOVE).

THE UUOS USETO, USETI AND CLOSE ARE ACCOMPLISHED BY XCTSTO, XCTSTI AND XCTCLS MACROS. THESE THREE MACROS MAY EACH TAKE A SINGLE NUMERIC ARGUMENT (INTERPRETED AS DECIMAL); TO SPECIFY THE RELATIVE BLOCK NUMBERS IN XCTSTO AND XCTSTI MACROS, AND TO OPTIONALLY SPECIFY THE ADDRESS FIELD OF THE CLOSE UUO (WHEREBY ONE SIDE OR THE OTHER OF A BI-DIRECTIONAL OPERATION CAN BE CLOSED).

EACH OF THESE XCTXXX MACROS GENERATES A PUSHT CALL TO A ROUTINE WHICH SETS UP AND EXECUTES FROM WITHIN THE ACS THE PROPER UUO FOR THE CURRENT CHANNEL, THEN PERFORMS ALL EXPECTATION CHECKING WITH ERROR MESSAGES AND DIAGNOSTIC OPERATIONS INVOKED AS APPROPRIATE AND FINALLY, TURNS OFF ANY STATUS BITS WHICH COME ON FOR THAT CHANNEL, AND RETURNS TO THE NEXT MACRO IN THE CURRENT PROCEDURE.

3.7 READ AND WRITE OPERATIONS

FOUR MACROS EXIST TO ALLOW EASY SPECIFICATION OF INPUT AND OUTPUT OPERATIONS. EACH OF THESE MACROS CAN RESULT IN AN ARBITRARY NUMBER OF INPUT OR OUTPUT UUOS BEING EXECUTED. THE FOUR MACROS ARE

```
REDNXT  N, P - "READ NEXT" N WORDS; EXPECT DATA
                PATTERN P.
WRTNXT  N, P - "WRITE NEXT" N WORDS IN DATA PATTERN P.
REDSPC N, B, P - "READ SPECIFIC" N WORDS STARTING WITH
                 FIRST WORD OF RELATIVE BLOCK B;
                 EXPECT DATA PATTERN P.
WRTSPC N, B, P - "WRITE SPECIFIC" N WORDS STARTING
                 WITH FIRST WORD OF RELATIVE BLOCK B,
                 IN DATA PATTERN P.
```

THE ARGUMENT N IS GIVEN IN WORDS RATHER THAN BLOCKS SO THAT PARTIAL BLOCKS MAY BE READ AND WRITTEN. THE ARGUMENT B IS A (DECIMAL AS USUAL) RELATIVE BLOCK NUMBER IN THE CURRENT FILE. THE DATA PATTERN IS ALSO A DECIMAL ARGUMENT WHICH SHOULD BE A NUMBER BETWEEN Ø AND 255. THIS IS BECAUSE THE ARGUMENT SUPPLIED IS TRUNCATED TO 8 BITS, AND THEN REPRODUCED FOUR TIMES IN A 36 BIT WORD. THE RESULTING WORD IS WRITTEN (OR EXPECTED) IN A PARTICULAR POSITION IN EACH BLOCK OF THE FILE BEING WRITTEN OR READ. A DATA PATTERN ARGUMENT OF Ø IS A SPECIAL CASE SPECIFYING THAT DATA PATTERNS ARE NOT TO BE CHECKED UPON INPUT.

IT IS SOMETIMES USEFUL FOR SOME OR ALL OF THE ARGUMENTS TO RED OR WRT MACROS TO BE MISSING. TO ALLOW FOR THIS POSSIBILITY, THREE ADDITIONAL WORDS ARE PART OF EACH CHANNEL DATA BASE (AND THE JOB STANDARD DATA BASE). THEY ARE THE LAST THREE ENTRIES IN THE ARGS MACRO (NAMELY: NRB, XFW, AND PAT) AND STAND FOR "NEXT RELATIVE BLOCK" (TO READ OR WRITE), "TRANSFER WORDS" (I.E. NUMBER OF WORDS TO READ OR WRITE), AND "DATA PATTERN." THESE LOCATIONS ARE SET IN THE USUAL MANNER WITH SETXXX (AND SELXXX) MACROS AND ONE REFERENCED BY THE RED AND WRT MACROS WHENEVER (ONLY IF) THE CORRESPONDING ARGUMENTS WERE NOT SUPPLIED. FOR EXAMPLES, SOME OF THE MANY EQUIVALENT MACRO SEQUENCES TO ACCOMPLISH THE WRITING OF BLOCKS 3, 4, AND 5 OF A FILE IN DATA PATTERN 2 MIGHT BE:

```
WRTSPC          3*BL, 3, 2
        (OR)
SETNRB          3
SETPAT          2
WRTSPC          3*BL
        (OR)
```

(3.7 CONT'D)

```
SELNRB          3
SELXFW          3*BL
SELPAT          2
FORSET          <NRB, XFW, PAT>
WRTSPC
        (OR)
SETPAT          2
XCTSTO          3
WRTNXT          3*BL
        (ETC.)
```

THE MECHANISM BY WHICH I/O IS ACCOMPISHED DEPENDS UPON THE DATA MODE. PRIOR TO ANY XCTOPN, THE DATA MODE MUST HAVE BEEN SELECTED BY A SETMOD (OR SELMOD) MACRO. MOD IS ONE OF THE ENTRIES DECLARED IN THE ARGS MACRO TO HAVE UNUSUAL ARGUMENTS. BASICALLY, ALL I/O IS DONE IN EITHER BINARY (14) OR DUMP (17) MODE; HOWEVER, 3 DIFFERENT BINARY MODES ARE AVAILABLE: BINARY INPUT ONLY (BI), BINARY OUTPUT ONLY (BO), AND BINARY BOTH (BB). THE SYMBOLS BI, BO, BB AND DM (WHICH STANDS FOR DUMP MODE) ARE DEFINED IN FILTST TO BE USED AS THE SET OF ARGUMENTS TO CHOOSE AMONG FOR SETMOD AND SELMOD MACROS. ACTUALLY BB MODE CAN ALWAYS BE USED IN PLACE OF BI OR BO; THE ONLY DIFFERENCE IS THAT TWO BUFFER HEADERS ARE ESTABLISHED IN BB MODE (I.E. BOTH SIDES OF THE BUF ENTRY IN THE DATA BASE ARE SET).

IN BINARY MODE, DOUBLE BUFFERING IS ALWAYS USED (BUFFERS ARE ASSEMBLED INTO EACH PER-CHANNEL DATA BASE), AND THE USUAL OVERLAPPING OF I/O WITH COMPUTATION IS MAINTAINED.

IN DUMP MODE, WITH SPACE FOR TWO INPUT AND TWO OUTPUT BUFFERS AVAILABLE, EACH OPERATION IS AUTOMATICALLY BROKEN INTO SOME NUMBER (POSSIBLY ZERO) OF 512 WORD TRANSFERS FOLLOWED (POSSIBLY) BY ONE LAST TRANSFER OF LESS THAN 512 WORDS.

3.8 DEFINING PROCEDURES

THE REMAINING MACROS (IMPLEMENTED TO DATE) IN FILTST ARE USED FOR DEFINING AND CALLING TEST PROCEDURES.

ALL TEST PROCEDURES HAVE NAMES. THESE NAMES HAVE TWO PARTS EACH OF WHICH MAY BE ONE OR TWO CHARACTERS LONG; THE TWO PARTS ARE SEPARATED BY A COMMA. THIS CONVENTION OF TWO PART NAMES, CONTAINIING NO MORE THAN FOUR CHARACTERS TOTAL, SEEMED A GOOD COMPROMISE BETWEEN THE NEED TO RECOGNIZE PROCEDURE FUNCTIONS EASILY FROM THEIR NAMES, AND THE NEED TO GENERATE UNIQUE SYMBOLS FOR EACH PROCEDURE. NO PARTICULAR CONVENTIONS ARE RECOMMENDED FOR NAMING PROCEDURES EXCEPT THAT NAMES SHOULD SUGGEST THE LIKELY "LEVEL" OF THE TEST PROCEDURE; THAT IS, SINCE PROCEDURES CAN CALL EACH OTHER RECURSIVELY, IT IS QUITE CONVENIENT TO WRITE AN EXPANDING SET OF GENERAL PURPOSE "INNER" PROCEDURES AND TO BUILD LEVELS OF "OUTER" PROCEDURES WHICH CALL THEM; NAMES SHOULD REVEAL AT LEAST WHICH OF THE PROCEDURES ARE OUTERMOST.

THE MACRO PROCED MUST BE THE FIRST IN ANY PROCEDURE, AND DEFINES ITS NAME; THE TWO ARGUMENTS TO PROCED ARE THE TWO PARTS OF THE PROCEDURE NAME. FOR EXAMPLE, AMONG THE SAMPLE PROCEDURES WRITTEN SO FAR FOR FILTST IS ONE WHICH BEGINS

```
PROCED  MT, UB
```

THE MNEMONIC IS INTENDED TO SUGGEST "MAIN TEST" (MEANING OUTERMOST LEVEL), "UPDATE BACKWARDS." AS THE COMMENTS REVEAL, THIS PROCEDURE UPDATES MULTIPLE BLOCKS OF DECREASING RELATIVE BLOCK NUMBER. THE PROCED MACRO GENERATES THE CODE WHICH MAINTAINS THE ERROR TRACE PUSH DOWN STACK AND ENTRY COUNTS FOR THE PROCEDURE, AND GENERATES TWO SYMBOLS OF THE FORM

```
X.N1N2  AND     C.N1N2
```

WHERE N1 AND N2 ARE THE TWO PARTS OF THE PROCEDURE NAME. THE FIRST SYMBOL IS THE PROCEDURE ENTRY POINT AND THE SECOND LABELS THE REGISTER WHERE CALLS TO THIS PROCEDURE ARE COUNTED. ALL PROCEDURES MUST END WITH THE MACRO ENDPR WHICH TAKES NO ARGUMENTS. BLOCKS WHICH BEGIN WITH PROCED AND END WITH ENDPR MAY BE INSERTED IN ANY ORDER WITHIN THE AUXILLIARY FILE (ASSEMBLED WITH FILTST) SINCE CONTROL NEVER FALLS FROM ONE INTO THE NEXT.

3.9 CALLING PROCEDURES

TWO MACROS EXIST FOR CALLING PROCEDURES FROM WITHIN OTHER PROCEDURES. (A SEPARATE MECHANISM EXISTS FOR CALLING OUTERMOST PROCEDURES INITIALLY AS EXPLAINED BELOW.) THE MACRO CALLPR TAKES TWO ARGUMENTS WHICH, AS IN THE CASE OF PROCED, ARE THE TWO PARTS OF A PROCEDURE NAME. CALLLPR SIMPLY GENERATES A PUSHJ TO THE ENTRY POINT X.N1N2.

A RELATED MACRO, RPCALL ("REPEAT CALL"), IS USED TO CALL PROCEDURES MORE THAN ONCE. IT TAKES A THIRD ARGUMENT WHICH IS A DECIMAL COUNT OF THE NUMBER OF TIMES TO CALL THE PROCEDURE. IF THIS ARGUMENT IS MISSING, THE VALUE IS TAKEN INSTEAD FROM THE CONTENTS OF YET ANOTHER WORD IN THE CURRENT CHANNEL DATA BASE; THIS WORD IS DEFINED BY THE MNEMONIC REP IN THE ARGS MACRO, AND IS SET AS USUAL BY MEANS OF THE SETREP (OR SELREP AND FORSET) MACROS.

RPCALL IS THE MEANS BY WHICH "LOOPS" ARE ACCOMPLISHED. ANY SEQUENCE OF MACROS TO BE INCLUDED IN A LOOP IS DECLARED TO BE A SEPARATE PROCEDURE AND CALLED THE APPRPRIATE NUMBER OF TIMES BY A RPCALL MACRO WITHIN SOME OTHER PROCEDURE. THE "TALLY" CONCEPT USUALLY ASSOCIATED WITH LOOPS IS OCCASIONALLY USEFUL WITHIN FILTST PROCEDURES; INCREMENT AND DECREMENT OPERATIONS WHEN NEEDED ARE ACCOMPLISHED BY MEANS OF DIRECT PDP-10 INSTRUCTIONS AS FOLLOWS:

3.10 PROCEDURE STEPS WHICH ARE NOT FILTST MACROS:

SO FAR ALL DISCUSSION OF THE FILTST DATA BASE HAS INVOLVED SETTING LOCATIONS WITHIN IT. THESE LOCATIONS CAN BE REFERENCED, CHECKED, INCREMENTED, ETC., BY MEANS OF DIRECT PDP-10 INSTRUCTIONS. FIVE ACCUMULATORS MAY BE OF INTEREST TO USERS WRITING TEST PROCEDURES:

ACCUMULATORS "A", "B" AND "C" ARE GUARANTEED TO BE PRESERVED BY ALL CODE GENERATED OR CALLED BY FILTST MACROS AND ARE THEREFORE AVAILABLE TO BE USED WITHIN TEST PROCEDURES. IF THE USER FEELS IT NECESSARY TO PRESERVE THESE ACS IN "INNER" PROCEDURES WHICH MIGHT USE THEM, PUSH AND POP INSTRUCTIONS CAN BE USED WITH ACCUMULATOR "P". FINALLY ACCUMULATOR "D" ALWAYS POINTS TO THE DATA BASE FOR THE CURRENT USER CHANNEL (SET BY THE MOST RECENT SELCHN MACRO); EACH OF THE MNEMONICS IN THE ARGS MACRO GENERATES A SYMBOL OF THE FORM RIBXXX WHICH WHEN INDEXED BY D WILL REFERENCE THAT PARTICULAR ENTRY IN THE CURRENT CHANNEL DATA BASE. ("RIB" WAS CHOSEN BECAUSE MANY, BUT NOT ALL, OF THE ENTRIES IN EACH DATA BASE CORRESPOND TO ENTRIES IN A FILES RETRIEVAL INFORMATION BLOCK.)

(3.10 CONT'D)

THUS, FOR EXAMPLE, THE INSTRUCTION SEQUENCE

```
MOVEI   A,5
ADDM    A,RIBNRB (D)
```

WILL ADD 5 TO THE "NEXT RELATIVE BLOCK" NUMBER IN THE CURRENT CHANNEL DATA BASE; THE NEW VALUE WILL BE USED WHEN THE NEXT REDSPC OR WRTSPC MACRO IS EXECUTED (PROVIDED ITS SECOND ARGUMENT IS MISSING IN THE CALL).

REGISTERS WITHIN THE "JOB STANDARD" DATA BASE MAY BE REFERENCED AS SYMBOLIC LOCATIONS USRXXX WHERE XXX MATCHES ANY MNEMONIC IN THE ARGS MACRO; (NO INDEX REGISTER IS USED IN REFERENCES TO LOCATIONS USRXXX).

3.11 THE TEST TABLE:

IN ADDITION TO A COLLECTION OF SEPARATE PROCEDURE DEFINITIONS IN NO PARTICULAR ORDER, THE AUXILLIARY FILE ASSEMBLED WITH FILTST MUST CONTAIN A TABLE WHOSE STARTING LOCATION IS TESTS AND WHOSE CONTENTS DEFINE THE ORDER IN WHICH THE OUTERMOST LEVEL OF TEST PROCEDURES ARE TO BE CALLED WHEN FILTST IS RUN.

ONE MACRO, CALLED DO, IS USEFUL FOR SPECIFYING ENTRIES IN THE TESTS TABLE. ITS FIRST TWO ARGUMENTS ARE THE TWO PARTS OF THE NAME OF SOME PROCEDURE TO BE EXECUTED, AND ITS THIRD ARGUMENT, IF PRESENT, IS A DECIMAL COUNT OF THE NUMBER OF TIMES THAT PROCEDURE IS TO BE EXECUTED BEFORE THE NEXT ENTRY IN THE TESTS TABLE IS SEEN. THE TESTS TABLE IS THERFORE A LIST OF DO MACROS AND MUST BE FOLLOWED BY A WORD CONTAINING ZERO. WHEN THE ZERO WORD IS ENCOUNTERED DURING EXECUTION, THE LIST OF TESTS IS REPEATED FROM THE BEGINNING. MORE INFORMATION ABOUT THE STRUCTURE AND USE OF THE TESTS TABLE CAN BE FOUND IN THE FILTST LISTING ON THE PAGE WHEREON THE SYMBOL DOTEST IS DEFINED. USE OF THE DO MACRO RESULTS IN THE GENERATION OF COMMENTS IN THE LISTING WHICH NUMBER THE ENTRIES IN THE TEST TABLE FOR EAST IDENTIFICATION DURING DEBUGGING.

THE ONLY OTHER CONSTRUCTION WHICH MUST BE PART OF THE AUXILLIARY FILE ASSEMBLED WITH FILTST ARE THE TWO LINES:

```
LSTECT : XWD ZCHAIN,0
         AND
END START
```

THE FIRST OF THESE SPECIFIES THE END OF A CHAIN LIST THROUGH WHICH THE ENTRY COUNTS OF ALL PROCEDURES ARE LINKED (SO THAT THEY MAY BE CLEARED AT APPROPRIATE TIMES). THE FILTST STARTING ADDRESS IS START, DEFINED NEAR THE END OF THE MAIN FILTST FILE.

4. FILTST OPERATION

4.1 THE ERROR TRACE

UPON ANY ERROR MESSAGE (AND WHENEVER THE W COMMAND IS TYPED - SEE BELOW) A PRINTOUT OF THE CURRENT TRACE STACK IS GENERATED. THE TRACE SHOWS WHAT STEP OF WHAT PROCEDURE WAS BEING EXECUTED WHEN THE ERROR OCCURRED, AND BY EXACTLY WHAT PATH CONTROL HAD REACHED THAT POINT. THE TRACE BEGINS WITH THE CURRENT INDEX INTO THE TESTS TABLE (PRINTED IN OCTAL SINCE THE DO MACRO NUMBERS THE ENTRIES IN OCTAL IN THE LISTING). A DECIMAL NUMBER IS GIVEN IN PARENTHESIS WITH EACH PROCEDURE NAMED IN THE TRACE; IT IS THE NUMBER OF TIMES THAT PROCEDURE HAS BEEN CALLED SINCE THE TESTS TABLE INDEX WAS INCREMENTED TO ITS PRESENT VALUE. THAT IS, THE ENTRY COUNTS FOR ALL PROCEDURES ARE CLEARED WHENEVER ANY ENTRY IN THE TESTS TABLE HAS BEEN COMPLETED. (SINCE ENTRY COUNTS ARE PART OF THE "PURE" PROCEDURE DEFINITIONS IN FILTST'S HIGH SEGMENT, THIS FEATURE IS MOST USEFUL WHEN ONLY A SINGLE JOB IS USING THE HIGH SEGMENT.) AS AN EXAMPLE, THE TRACE:

```
5 RD,DL -- 9 (1) / OP,RD -- 3 (2)
```

INDICATES THAT CONTROL IS CURRENTLY AT STEP 3 OF PROCEDURE OP,RD (AND THAT THIS IS THE SECOND CALL TO OP,RD). IT SAYS THAT OP,RD WAS CALLED BY STEP 9 OF PROCEDURE RD,DL (DURING THE FIRST CALL TO RD,DL), AND THAT RD,DL WAS CALLED DIRECTLY AS A RESULT OF THE DO MACRO COMPRISING THE FIFTH ENTRY IN THE TESTS TABLE. NOTE THAT ONLY THOSE PROCEDURE STEPS WHICH ARE FILTST MACROS ARE COUNTED AS STEPS; IN DETERMINING WHAT LINE OF A PROCEDURE CORRESPONDS TO WHAT STEP, ONE MUST NOT COUNT ANY LINES COMPRISING BASIC PDP-10 INSTRUCTIONS.

ANY FURTHER INFORMATION REQUIRED FOR ANALYSIS OF THE PROBLEM (IF THE TRACE WAS PRODUCED AS THE RESULT OF SOME PROBLEM) MUST BE OBTAINED THROUGH DDT WHICH IS TYPICALLY LOADED WITH FILTST.

4.2 FILTST COMMANDS

WHEN FILTST IS RUNNING, IT ACCEPTS A SET OF SINGLE CHARACTER COMMANDS WHICH MAY BE TYPED AT ANY TIME. THE CONSOLE INPUT BUFFER IS CHECKED FOR THE PRESENCE OF COMMAND CHARACTERS AFTER EVERY UUO IS EXECUTED. SOME COMMANDS ARE PROCESSED WITHOUT STOPPING THE TEST SEQUENCE IN PROGRESS; OTHERS INTERRUPT THE RUNNING OF FILTST IN CONVENIENT WAYS. THE PAGE OF THE FILTST LISTING ON WHICH THE SYMBOL COMTAB IS DEFINED DESCRIBES HOW NEW COMMANDS MAY BE ADDED; THOSE WHICH HAVE BEEN IMPLEMENTED TO DATE ARE DESCRIBED IN THE PARAGRAPHS BELOW, AND SUMMARIZED IN APPENDIX LL.

AS MENTIONED ABOVE, THE COMMAND W (MNEMONICS FOR "WHERE" ARE WE?) MAY BE TYPED AT ANY TIME; WHEN CONTROL RETURNS FROM THE NEXT UUO, THE TRACE STACK WILL BE PRINTED SO THAT, BY REFERENCE TO THE AUXILLIARY FILE LISTING, THE USER CAN DETERMINE HOW FAR THE TEST SEQUENCE HAS PROGRESSED. ERROR MESSAGES CONSIST OF AN ERROR NUMBER OF THE FORM EXX AND THE TEXT OF THE MESSAGE. OUTPUT CAN BE ABBREVIATED TO THE ERROR NUMBER ALONE BY MEANS OF THE A COMMAND; THE "LONG" FORM CAN BE RE-ENABLED BY MEANS OF THE L COMMAND.

4.3 FILTST AND DDT

IT IS INTENDED THAT DDT BE LOADED WITH FILTST FOR INTERACTIVE INVESTIGATION OF OBSCURE ERRORS. IT IS FURTHER INTENDED THAT A DDT BREAKPOINT BE INSERTED AT LOCATION BREAK IN FILTST'S LOW SEGMENT. INSERTING A BREAKPOINT AT THIS LOCATION, WHICH IS OTHERWISE INITIALIZED TO CONTAIN A POPJ INSTRUCTION, ENABLES TWO OTHER USEFUL FILTST COMMANDS. FILTST CAN BE MADE TO EXECUTE A PUSHJ TO BREAK, WHICH WILL THEN ENTER DDT VIA THE BREAKPOINT, AT TWO PREDICTABLE TIMES. THE "SPACE" COMMAND (THE CHARACTER "SPACE") WILL CAUSE THE PUSHJ TO BE EXECUTED AS SOON AS THE COMMAND IS PROCESSED - THAT IS, AS SOON AS THE NEXT UUO IS EXECUTED. THE S (MNEMONIC FOR "STOP") COMMAND WILL CAUSE THE PUSHJ TO BREAK WHEN THE TEST PROCEDURE INDICATED IN THE CURRENT POSITION OF THE TESTS TABLE IS COMPLETE. THAT IS, S WILL STOP FILTST WITH NO PROCEDURE IN PROGRESS AS IT IS JUST ABOUT TO BEGIN ANOTHER "OUTERMOST" PROCEDURE. NEITHER THE "SPACE" NOR THE S COMMAND HAVE ANY EFFECT IF A BREAKPOINT HAS NOT BEEN PLACED AT LOCATION BREAK.

4.4 AFTER AN ERROR MESSAGE

EXCEPT AFTER AN ERROR MESSAGE, FILTST COMMANDS ARE PROCESSED WHENEVER THEY ARE TYPED, BUT NONE ARE EVER REQUIRED. AFTER AN ERROR MESSAGE HOWEVER, AT LEAST ONE OF THE "ACTION TYPE" COMMANDS IS REQUIRED AND IS WAITED FOR. IN THE TABLE OF COMMANDS (COMTAB IN FILTST LISTING), TWO COMMANDS ARE CURRENTLY FLAGGED AS ACTION COMMANDS; ONE OR THE OTHER OF THESE MUST BE TYPED THE USER AFTER ANY ERROR. ONE IS THE "SPACE" COMMAND WHICH CAUSES DDT TO BE ENTERED (AS DESCRIBED ABOVE) SO THAT THE ERROR MAY BE INVESTIGATED; THE OTHER IS THE "C" (FOR "CONTINUE") COMMAND WHICH CAUSES FILTST TO PROCEDE AND IGNORE THE ERROR. IT IS INTENDED TO BE POSSIBLE TO CONTINUE FROM ANY ERROR THAT MIGHT OCCUR WITH INTERNAL CONDITIONS SET AS THEY WOULD HAVE BEEN HAD THE ERROR NOT OCCURRED. TYPING C MAY WELL RESULT IN OTHER RELATED ERRORS BEING DETECTED, BUT AT LEAST WILL NOT LOOP ON A SINGLE ERROR. IT IS HOPED THAT WHEN AN ERROR DOES OCCUR, THE ERROR TRACE STACK (PRINTED AUTOMATICALLY), WILL BE SUFFICIENT INFORMATION FOR ANALYSIS OF THE PROBLEM, (IN CONJUNCTION WITH THE FILTST LISTING), AND THAT TYPING "C" WILL RESULT IN A SUCCESSFUL "CONTINUE" SUCH THAT FURTHER ERRORS WHICH OCCUR WILL BE SEPARATE SITUATIONS FOR LATER ANALYSIS. THE "C" COMMAND HAS NO EFFECT IF TYPED WHILE FILTST IS RUNNING. ALL CHARACTERS WHICH ARE NOT COMMANDS (INCLUDING CARRIAGE RETURN AND LINE FEED) ARE IGNORED.

4.5 DDT TECHNIQUES

THE FILTST LISTING SHOULD BE CONSULTED FOR COMPLETE INFORMATION ON INTERNAL STRUCTURE, ESPECIALLY PAGE 3 WHEREON THE IMPURE DATA BASE IS DEFINED. A FEW REGISTERS IN THE DATA BASE WERE INCLUDED SPECIFICALLY FOR DEBUGGING PURPOSES; THEIR SPECIAL FUNCTIONS ARE EXPLAINED MORE FULLY HERE THAN IN THE LISTING COMMENTS.

ACCUMULATORS OF SPECIAL INTEREST DURING DEBUGGING ARE:

- P - THE CONTROL PUSH-DOWN POINTER
- TR - THE TRACE STACK PUSH-DOWN POINTER
- D - POINTER TO THE DATA BASE FOR THE CURRENT USER CHANNEL
- U - ALWAYS CONTAINS THE LAST UUO EXECUTED

LOCATION TRLAST CONTAINS THE NAME OF THE LAST PROCEDURE SUCCESSFULLY COMPLETED (LOADED BY EVERY ENDPR MACRO). THE TWO HALVES OF THE PROCEDURE NAME ARE IN LEFT JUSTIFIED 7-BIT ASCII IN THE TWO HALVES OF THE WORD - A RATHER INCONVENIENT FORMAT FOR DDT, UNFORTUNATELY.

(4.5 CONT'D)

THREE LOCATIONS IN EACH CHANNEL DATA BASE (NO NEED FOR THEM TO BE PART OF "JOB STANDARD" DATA BASE) ARE OF SPECIAL INTEREST FOR DEBUGGING:

LOCATION CHNCNT CONTAINS IN ITS LEFT HALF A SET OF BITS, ONE BIT FOR EACH OF THE POSSIBLE UUOS, SHOWING WHICH UUOS HAVE BEEN EXECUTED ON THIS CHANNEL. IN THE RIGHT HALF, THERE IS A COUNT OF THE NUMBER OF UUOS WHICH HAVE BEEN EXECUTED ON THIS CHANNEL (TO ACCOUNT FOR THE FACT THAT SOME UUOS MAY HAVE BEEN EXECUTED SEVERAL TIMES). CHNCNT IS INITIALIZED TO ZERO JUST BEFORE ANY OPEN UUO IS EXECUTED ON THAT CHANNEL SO THAT THE CHANNEL HISTORY REMAINS EVEN AFTER A RELEASE UUO.

LOCATION CHNARG HAS A BIT POSITION FOR EACH OF THE ENTRIES IN THE ARGS MACRO; THAT IS, FOR EVER DATA BASE ENTRY FOR WHICH THERE IS A SETXXX MACRO. IT IS INITIALIZED TO ZERO UPON ANY RELEASE UUO, AND IS USED TO REMEMBER WHICH ENTRIES HAVE BEEN SET (BY SETXXX OR FORSET OR INSURE) SINCE THE RELEASE. CHNARG IS USED BY THE INSURE MACRO TO DETERMINE WHICH ENTRIES MUST BE SET FROM THE JOB STANDARD DATA BASE.

LOCATION CHNCRB ALWAYS CONTAINS THE "CURRENT RELATIVE BLOCK" NUMBER WHICH FILTST EXPECTS TO READ OR WRITE NEXT. IT IS SET UP BY ANY XCTSTI ON XCTSTO MACRO, BUT MORE IMPORTANTLY, IT IS INCREMENTED ON EVERY INPUT OR OUTPUT UUO DONE BY FILTST. ITS CONTENTS ARE WRITTEN INTO ANY OUTPUT BLOCK AND CHECKED UPON READING ANY INPUT BLOCK. THAT IS, AMONG OTHER THINGS EVERY BLOCK OF ANY FILE CREATED BY FILTST CONTAINS ITS OWN RELATIVE BLOCK NUMBER. IF THE ERROR MESSAGE "INCORRECT RELATIVE BLOCK NUMBER IN INPUT DATA" APPEARS, THE USER CAN FIND THE EXPECTED NUMBER IN CHNCRB AND THE ACTUAL NUMBER IN THE CURRENT INPUT BUFFER (AS THE IMPURE DATA BASE DEFINITION IN FILTST LISTING WOULD REVEAL, LOCATION IBHBLK IN THE CHANNEL DATA BASE - FOUND THRU ACCUMULATOR D - IS THE INPUT BUFFER HEADER POINTER; THE RELATIVE BLOCK NUMBER IS THE FIRST DATA WORD IN ANY BUFFER AS THE TABLE AT LOCATION ITMLST WOULD REVEAL).

NOTE THAT ALL OF THE ABOVE SYMBOLIC LOCATIONS ARE RELATIVE TO THE CURRENT CHANNEL DATA BASE. USING DDT, ONE FINDS THE CURRENT DATA BASE BY TYPING D/ AND THEN PERHAPS $Q<DB: SUCH THAT FROM THEN ON ONE CAN TYPE DB+CHNCRB/ ETC.

5. FEATURES YET TO BE IMPLEMENTED

A NUMBER OF FEATURES PLANNED FOR FILTST ARE NOT IMPLEMENTED AS OF THIS WRITING. THEIR ABSENCE DOES NOT PRECLUDE THE USE OF FILTST AS AN EFFECTIVE TEST VEHICLE IN ITS PRECENT STATE; HOWEVER, SINCE THE STRUCTURE OF THE PRESENT VERSION IN SOME WAYS REFLECTS THE ANTICIPATED INCLUSION OF THESE FEATURES IT IS WELL FOR ANYONE WHO MIGHT MODIFY THE PRESENT VERSION TO HAVE CONSIDERED THE FOLLOWING POINTS:

1. MULTI-JOB FEATURES: SOME COORDINATED TESTS CAN BE PERFORMED ONLY WITH TWO (OR MORE) JOBS ACTING IN A SYNCHRONOUS FASHION ON THE SAME FILES. IT IS PLANNED THAT FILTST INCLUDE AN OPTION TO ESTABLISH ONE (OR MORE) ADDITIONAL JOBS THROUGH THE PSEUDO-TELETYPE TO SYNCHRONIZE THE ACTIVITY OF THIS SLAVE JOB WITH THE MAIN JOB.

   A SIMPLE MECHANISM TO ACCOMPLISH THIS SYNCRONIZATION MIGHT INVOLVE TWO SEPARATE TESTS TABLES, ONE TO DRIVE EACH OF THE JOBS, WITH ALL PROCEDURES BEING SHARED AS USUAL. A SINGLE MACRO MIGHT BE DEFINED FOR INCLUSION IN EITHER TESTS TABLES WHICH SAYS IN EFFECT: "PUT THIS JOB INTO IDLE (SLEEP LOOP) AND START THE OTHER JOB AGAIN". THE TWO JOBS WOULD NOT RUN IN PARALLET, BUT RATHER IN SEE-SAW FASHION TO ACCOMPLISH SYNCHRONIZED ACTIONS. (THE EFFECT OF MULTIPLE COPIES OF FILTST RUNNING TO CREATE AN INTERESTINGLY HEAVY LOAD IS EASILY ACHIEVED WITH THE PRESENT VERSION OF FILTST BY RUNNING MULTIPLE JOBS, SHOWING THE HIGH SEGMENT, AND STARTING EACH WITH A DIFFERENT INITIAL OFFSET INTO THE TESTS TABLE.)

   THE MAJOR REASON WHY MULTI-JOB FEATURES HAVE NOT YET BEEN IMPLEMENTED IS THE TIME REQUIRED TO DESIGN AND IMPLEMENT THE CODE WHICH DEALS WITH THE PSEUOD-TELETYPE AND WHICH HANDLES THE PRINTING OF ERROR MESSAGES (AND THE SUBSEQUENT COMMAND INPUT) FOR THE JOB(S) BEING CONTROLLED BY THE PTY(S).

2. CURRENTLY THE TWO MACROS XCTSRC (SEARCH UUO) AND XCTSEK (SEEK UUO) ARE NOT CODED. THEY PRESENT NO DIFFICULTY BUT WERE NOT USEFUL IN DEBUGGING FILTST UNDER LEVEL C DISK SOFTWARE.

(5. CONT'D)

3. THERE SHOULD BE A PRIVILEGED UUO IMPLEMENTED IN THE MONITOR WHEREBY A PRIVILEGED JOB COULD CHANGE THE PROJECT PROGRAMMER NUMBERS IT WAS "LOGGED IN UNDER". THEN A FILTST MACRO SHOULD BE DEFINED TO EXECUTE THIS UUO (E.G., CNGPPN PJ, PG) SO THAT FILTST PROCEDURES COULD BE WRITTEN TO TEST THE VARIOUS FILE PROTECTION FEATURES OF THE MONITOR.

4. AS PART OF ITS INITIALIZATION, FILTST COULD, AND PROABABLY SHOULD, CREATE UFDS FOR ALL OF THE ENTRIES IN DIR.XX TABLE IN EACH OF THE STRUCTURES IN THE STR.XX TABLE, IF SUCH UFDS DID NOT ALREADY EXIST. THIS ACTION WOULD REQUIRE THAT FILTST BE ABLE TO TEMPORARILY ASSUME MASTER FILE PRIVILEGES AS IT COULD DO WITH THE CNGPPN MACRO IN 3 ABOVE.

5. RATHER THAN ASSEMBLING INTO FILTST THE VARIOUS OPTIONS AVAILABLE TO TEST PROCEDURES FOR THOSE DATA BASE ARGUMENTS OF MORE THAN 18 BITS, THESE TABLES (XXX.YY) COULD BE SET CONVERSATIONALLY BY MEANS OF AN INITIALIZATION DIALOGUE.

6. FINALLY, NO REASON HAS BEEN FOUND SO FAR FOR FILTST TO DO ANY OF THE FOLLOWING; IT SEEMS HOWEVER, THAT THE FACT THAT FILTST DOES NOT FIND ANY REASON TO DO THESE THINGS SHOULD BE MENTIONED.

   A. USE THE UUO WHICH ALLOWS SETTING AND/OR INTERROGATING A JOBS FILE SEARCHLIST.

   B. USE THE UUOS WHICH ALLOW INTERROGATING WHICH FILESTRUCTURES EXIST OR OBTAINING THEIR CHARACTERISTICS.

   C. ATTEMPT TO REMEMBER THE STATES (CONTENTS) OF THE VARIOUS UFDS IN THE DIR.XX TABLE.

   D. ALLOW ANY ABILITY TO INTRODUCE RANDOMNEOS INTO TEST PROCEDURES.

6. APPENDIX I FILTST MACROS AND THEIR ARGUMENTS

MACRO ARGUMENTS SHOWN BELOW ARE TO BE INTERPRETED AS FOLLOWS:

N - ANY DECIMAL INTEGER

A - ONE OR TWO CHARACTERS CORRESPONDING TO THOSE CHARACTERS FOLLOWING THE PERIOD IN A TABLE LOCATION OF THE FORM XXX.XX WITHIN FILTST

O - ANY OCTAL INTEGER

M - ANY ONE OF THE SYMBOLS BI, BO, BB OR DM

IST - A STRING OF THREE CHARACTER MNEMONICS (FOUND WITHIN THE ARGS MACRO IN FILTST) SEPARATED BY COMMAS AND ENCLOSED IN ANGLE BRACKETS

R - ONE OF THE CHARACTERS N, E, X OR ANY DECIMAL INTEGER

S - A STRING OF FIVE CHARACTERS FROM THE SET X, Ø OR 1

L - ANY SINGLE CHARACTER OR ANY TWO CHARACTERS

. - A PERIOD AS PART OF ANY ARGUMENT INDICATES THAT THE ARGUMENT MAY BE MISSING, THE ABSENCE OF A PERIOD INDICATES THAT THE ARGUMENT MUST BE PRESENT

* - AN ASTERISK FOLLOWING A MACRO INDICATES THAT THE MACRO IS CHANNEL INDEPENDENT, THE ABSENCE OF AN ASTERISK INDICATES THAT THE MACRO PERTAINS TO THE CHANNEL DECLARED BY THE MOST RECENTLY EXECUTED SELCHN MACRO

A. SELECTING A CHANNEL

```
SELCHN  N   *
```

B. SETTING CHANNEL DATA BASE (NOTE: FOR EVERY SETXXX MACRO BELOW, THERE IS A CORRESPONDING SELXXX MACRO WITH IDENTICAL FORMAT ARGUMENTS FOR MODIFYING THE JOB STANDARD, CHANNEL INDEPENDENT DATE BASE.)

```
SETMOD  M       -   DATA MODE
SETSTR  A       -   FILE STRUCTURE
SETARG  N       -   NO. OF EXTENDED LKP-ENT-REN ARGUMENTS
SETDIR  A       -   DIRECTORY (PJ, PG)
SETNAM  A       -   FILE NAME
SETEXT  A       -   FILE EXTENSION
SETPRT  O       -   FILE PROTECTION
SETEST  N       -   ESTIMATED LENGTH
SETALC  N       -   HIGHEST BLOCK TO ALLOCATE
SETPOS  N       -   WHERE TO ALLOCATE
SETVER  N       -   VERSION NUMBER
SETREP  N       -   REPEAT COUNT FOR RPCALL
SETNRB  N       -   NEXT RELATIVE BLOCK FOR RED & WRTSPC
SETXFW  N       -   WORDS TO TRANSFER FOR ALL RED & WRT
SETPAT  N       -   DATA PATTERN FOR ALL RED & WRT
RSTEXT          -   CLEAR RIBEXT RH
RSTPRT          -   CLEAR RIBPRT BITS 9 - 35
CLRVAL  N       -   CLEAR DATA BASE, START AT WORD N
FORSET 1ST      -   FORCE SETTING FROM STANDARD DATA BASE
INSURE 1ST      -   SET FROM STANDARD DATA BASE IF
                    NOT ALREADY SET
```

C. DECLARING EXPECTATIONS

```
EXPECT R.,S.
```

D. EXECUTING UUO'S

```
XCTOPN          OPEN
XCTLKP          LOOKUP
XCTENT          ENTER
XCTREN          RENAME
XCTSTO N        USETO
XCTSTI N        USETI
XCTCLS N.       CLOSE
XCTRLS          RELEASE
REDNXT N.,N.)   WORDS, PATTERN
         )
WRTNXT N.,N.)
REDSPC N.,N.,N.)WORDS, BLOCK #, PATTERN
         )
WRTSPC N.,N.,N.)
```

E. DEFINING PROCEDURES

```
PROCED          1,1
ENDPR
CALLPR          1,1
RPCALL          1,1,N.
```

F. THE TESTS TABLE

```
DO      1,1,N.
```

7. APPENDIX II FILTST COMMANDS

CHARACTER	ACTION
SPACE	HIT DDT BREAKPOINT IMMEDIATELY
S	HIT DDT BREAKPOINT WHEN CURRENT STEP OF TEST TABLE COMPLETE
C	CONTINUE AFTER AN ERROR
W	PRINT CURRENT TRACE STACK WITHOUT STOPPING
A	ABBREVIATE ERROR MESSAGES TO ERROR NUMBER ONLY
L	TYPE COMPLETE ERROR MESSAGES

SCRIPT EXTERNAL DOCUMENTATION
100-XXX-XXX-00
TONY LAUCK
2 MAY 70

THE PROGRAM DESCRIBED IN THIS MEMORANDUM WAS DEVELOPED TO AID THE STUDY OF THE PDP-10 MONITOR. IT IS NOT A SUPPORTED PRODUCT, BUT IS AVAILABLE TO CUSTOMERS ON A FOR INFORMATION PURPOSES ONLY BASIS.

1.0 PURPOSE

THE SCRIPT PROGRAM ALLOWS PREDETERMINED SEQUENCES OF CHARACTERS TO BE SENT OVER MULTIPLE PSEUDO-TELETYPES AND THEREBY ALLOW THE SIMULATION OF A LOAD ON THE TIME-SHARING SYSTEM. THE RESULTS OF A RUN CAN BE WATCHED ON-LINE AND STATISTICS OF RESPONSE TIME, ETC. CAN BE RECORDED ON THE SYSTEM DISK.

2.0 JOB CAPABILITY

THE SCRIPT PROGRAM HAS BEEN WRITTEN AS A REENTRANT PROGRAM. THE CODE AND TEXT TO BE SENT ARE IN THE HIGH SEGMENT AND CAN BE SHARED BY SEVERAL JOBS, EACH WITH A 1K LOW SEGMENT. EACH JOB CAN CONTROL UP TO 14 JOBS, SO IT BECOMES POSSIBLE TO SIMULATE A LARGE NUMBER OF PDP-10 TIME-SHARING USERS.

3.0 CONTROL FEATURES

THE SCRIPT IS LOADED INTO THE HIGH SEGMENT FROM ANY PDP-10 INPUT DEVICE. IN ADDITION TO THE TEXT TO BE SENT, CONTROL COMMANDS CAN BE INCLUDED TO DETERMINE THE SPEED AT WHICH THE SIMULATED USERS WILL OPERATE.

3.1 TIMING PARAMETERS

PARAMETERS SUBJECT TO VARIATION INCLUDE TYPE-IN TIME, TYPE-OUT TIME, AND USER "THINK" TIME. THE TYPE-IN AND TYPE-OUT TIME CAN BE SPECIFIED AS A CONSTANT OR AS A RATE, IN WHICH CASE THE TIME WOULD DEPEND ON THE NUMBER OF CHARACTERS SENT OR RECEIVED. THE "THINK" TIME IS BROKEN UP INTO TWO PARTS: 1) THE ALLOWED RESPONSE TIME, AND 2) "FREE" TIME. IF THE COMPUTER GIVES INSTANTANEOUS RESPONSE TO A COMMAND, THE TOTAL DELAY IS SIMPLY THE SUM OF THE TWO QUANTITIES. IF RESPONSE IS GREATER THAN ALLOWED, "THINK" TIME IS EQUAL TO THE "FREE" TIME. IN BETWEEN, "THINK" TIME IS THE SUM OF ALLOWED RESPONSE PLUS "FREE" TIME MINUS THE ACTUAL COMPUTER RESPONSE TO THE COMMAND. ONE ADDITIONAL TIME FACTOR ALLOWED IS THE ABILITY TO SET A MAXIMUM RESPONSE PLUS TYPE-OUT DELAY FOR A GIVEN COMMAND. IF THE COMMAND EXCEEDS THIS LIMIT, THEN THE JOB IS INTERRUPTED BY SENDING TWO CONTROL-C CHARACTERS OVER THE PSEUDO TELETYPE. THIS ALLOWS A SCRIPT TO INCLUDE PROGRAM LOOPS WHICH ARE INTERRUPTED AFTER A SPECIFIED TIME INTERVAL.

3.2 REPETITION OF SCRIPTS

AN ADDITIONAL SCRIPT LANGUAGE FEATURE ALLOWS THE SCRIPT TO SPECIFY HOW MANY TIMES IT WILL BE EXECUTED, SO THAT STEADY STATE LOADING CONDITIONS CAN BE MEASURED.

3.3 STAGGERING LOAD BUILDUP

WHEN MULTIPLE JOBS ARE TO FOLLOW THE SCRIPT, THE SCRIPT MAY SPECIFY A STAGGERING INTERVAL SO THAT LOAD BUILDUP CAN BE GRADUAL.

3.4 MULTIPLE USER NUMBERS

AS AN OPTIONAL FEATURE, THE SCRIPT PROGRAM WILL CONVERT THE "#" CHARACTER INTO AN OCTAL STRING EQUAL TO THE PTY UNIT NUMBER. THIS WILL ALLOW ONE SCRIPT TO LOG MANY JOBS IN UNDER DIFFERENT NUMBERS OR FOR MULTIPLE JOBS TO USE SEPERATE FILES.

4.0 LOGGING OF RESULTS

THE SCRIPT PROGRAM MAKES AN ENTRY IN THE LOG FILE FOR EACH LINE OF PTY DATA SENT TO THE TIME-SHARING SYSTEM. THE TIME, IDENTITY OF A PARTICULAR JOB, NUMBER OF CHARACTERS SENT, NUMBER OF CHARACTERS RECEIVED, NUMBER OF BUFFERS RECEIVED, AND TOTAL RESPONSE TIME ARE ALL RECORDED. IN ADDITION, DELAYS IN SENDING OR SIMULATING TELETYPE OUTPUT DUE TO POOR SCRIPT PROGRAM RESPONSE ARE ALSO RECORDED. THIS PROVIDES SOME IDEA OF HOW THE RESPONSE TO THE CONTROLLING JOB AFFECTED SYSTEM LOAD AND RESPONSE TIME MEASUREMENTS. IDEALLY THE CONTROLLLING JOB WOULD BE REAL-TIME, E.G. HIGH PRIORITY AND LOCKED IN CORE. DELAYS TO THE SCRIPT INPUT/OUTPUT LIGHTEN THE OVERAL SYSTEM LOAD, WHILE DELAYS IN RECEIVING REQUESTS FOR MORE INPUT ARE MEASURED AS IF THE USER JOB HAD EXPERIENCED WORSE RESPONSE THAN IT ACTUALLY DID.

AS A FURTHER CHECK ON ERRORS IN RESPONSE TIME MEASUREMENTS DUE TO POOR SCRIPT PROGRAM RESPONSE, A "+" CHARACTER IS OUTPUT WHEN POOR SCRIPT PROGRAM RESPONSE MAY HAVE CAUSED THE PT WAKE FUNCTION IN THE MONITOR TO FAIL. WHEN THIS HAPPENS, THE SCRIPT PROGRAM WILL WAKE UP BASED ON ITS SLEEP INTERVAL AND NOT ON OBJECT JOB RESPONSE. NORMALLY, THE MONITOR RESETS THE SLEEP COUNT WHEN A JOB RUNNING ON A PSEUDO-TELETYPE NEEDS SERVICE. THE SCRIPT PROGRAM TESTS THE PTY FLAGS, AND IF THE PTY DOESN'T NEED SERVICE SLEEPS. IDEALLY, THIS SLEEP WILL BE TERMINATED IMMEDIATELY BY THE MONITOR. HOWEVER, SHOULD THE SCRIPT PROGRAM BE RESCHEDULED BETWEEN TESTING THE PTY AND DOING THE SLEEP, THIS CAN'T HAPPEN. IN THIS CASE, THE SCRIPT PROGRAM COULD SLEEP FOR ITS MAXIMUM INTERVAL. THIS COULD BE UP TO 5 SECONDS. UNFORTUNATELY, THERE IS NO WAY TO TELL IF THIS HAPPENED. THE "+" IS SET WHENEVER A JOB GETS RESPONSE AND THE TIME OF DAY EXCEEDED THE TIME OF DAY THE SLEEP ENTRY WOULD HAVE TERMINATED. THIS CAN OCCUR ALSO DUE TO POOR RESPONSE ONCE THE SLEEP INTERVAL HAS TERMINATED.

4.1 DEFINITION OF RESPONSE TIME

THE RESPONSE TIME MEASURED IS THE TOTAL TIME FROM OUTPUTTING ("TYPING") A LINE OVER THE PTY UNTIL THE PROGRAM REQUESTS THE NEXT LINE OR TIMES OUT TO TWO CONTROL-C CHARACTERS, LESS TIME SPENT SIMULATING TTY OUTPUT. THUS, IF ONE LINE TO THE SYSTEM PRODUCES 50 LINES OF OUTPUT, ONE RESPONSE TIME WILL BE RECORDED, WHICH IS THE TOTAL TIME THE USER WAS WAITING FOR THE COMPUTER.

4.2 OVERLAP AND ITS EFFECT ON RESPONSE TIME MEASUREMENTS

THERE IS ONE POSSIBLE DIFFICULTY: OVERLAP BETWEEN OUTPUT AND THE NEXT INPUT. NORMALLY AN OUTPUT-BOUND PROGRAM IS WOKEN UP WHEN 8 CHARACTERS OF SPACE REMAIN IN THE MONITOR BUFFER, THUS A 0.8 SECOND DELAY (AT 10 CHARACTERS/SEC) WOULD NOT CAUSE A PAUSE IN TYPE-OUT. UNFORTUNATELY, THE PTY TRANSFERS A LINE AT A TIME, SO THIS OVERLAPPED TIME IS LOST. THE NET AFFECT IS TO MAKE THE MEASURED RESPONSE TIMES LOOK A LITTLE WORSE THAN RESPONSE TIMES AN ACTUAL TTY USER WOULD SEE. INCIDENTALLY, SHOULD THE PROGRAM REQUEST INPUT WHILE TYPEOUT IS IN PROGRESS, ALL TYPEOUT DELAY WILL BE COMPLETED BEFORE ACTING ON THE INPUT REQUEST.

4.3 DIFFERENT BUFFERING MODES

IF THE SCRIPT SPECIFIED OUTPUT AS A FUNCTION OF TIME, THEN THE SCRIPT PROGRAM WAITS UNTIL THE DELAY IS UP, THEN DATA IS TRANSFERRED OVER THE PTY TO THE SCRIPT PROGRAM. WHEN THE SCRIPT SPECIFIES AN OUTPUT RATE, THE SCRIPT PROGRAM MUST FIRST READ THE DATA FROM THE PTY, DETERMINE THE DELAY, AND THEN SLEEP. SPECIFYING ONE MODE OR THE OTHER MAY BE APPROPRIATE TO PARTICULAR PROGRAMS DUE TO DIFFERENT BUFFERING MODES.

4.4 ANALYSIS OF RESPONSE TIME RESULTS

IN ANY EVENT, THE USER MUST ANALYZE RESPONSE-TIME RESULTS WITH A KNOWLEDGE OF THESE FACTORS, AND AN UNDERSTANDING OF THE OVERLAP BETWEEN USER PROGRAM BUFFERS, MONITOR BUFFERS, AND SCRIPT BUFFERS. ALTERNATELY, IF PSYCHOLOGICAL SYSTEM PERFORMANCE IS IMPORTANT, THE USER CAN USE THE SYSTEM MANUALLY AND SEE HOW CLEVERLY THE SYSTEM IS CONCEALING ITS RESPONSE. THE MAIN PURPOSE OF THE SCRIPT PROGRAM IS TO OBTAIN CONSISTENT RELATIVE RESPONSE FIGURES TO ASSESS THE EFFECT OF HARDWARE AND SOFTWARE CHANGES IN THE TIME-SHARING SYSTEM.

5.0 FORMAT OF SCRIPT FILES

5.1 COMMAND LINES

A SCRIPT FILE CONSISTS OF SCRIPT LINES WHICH END WITH A <LINE FEED> CHARACTER. SCRIPT LINES ARE EITHER COMMAND LINES OR TEXT LINES. COMMAND LINES ARE USED TO CONTROL TYPING AND OPERATION OF THE SCRIPT PROGRAM. TEXT LINES ARE SENT OVER THE PSEUDO TELETYPE TO RUN THE OBJECT JOBS.

5.2 TEXT LINES

A COMMAND LINE BEGINS WITH ONE EXCLAMATION POINT WHICH IS FOLLOWED BY A NON-EXCLAMATION POINT CHARACTER.

A TEXT LINE BEGINS WITH NO EXCLAMATION POINT, OR WITH TWO EXCLAMATION POINTS. IN THIS CASE, ONLY THE SECOND IS SENT TO THE OBJECT JOB. IF A SCRIPT LINE BEGINS WITH ONE UP-ARROW, THEN THE UP-ARROW IS NOT SENT; INSTEAD THE NEXT CHARACTER IS CONVERTED TO A CONTROL CHARACTER BY COMPLEMENTING BIT 100. TWO UP-ARROWS RESULT IN ONE UP-ARROW BEING SENT. (THE CR/LF AT THE END OF ANY LINE STARTING WITH A SINGLE UP-ARROW ARE NOT SENT.)

NOTE THAT "!" AND "↑" HAVE NO SIGNIFICANCE EXCEPT AT THE BEGINNING OF A TEXT OR COMMAND LINE.

5.3 COMMAND LINE SYNTAX

COMMAND LINES CONSIST OF NUMBERS AND LETTER SWITCHES. SWITCHES THAT TAKE NUMERICAL VALUES MAY BE PRECEEDED BY A NUMBER. A NUMBER CONSISTS OF POSSIBLY ONE MINUS SIGN FOLLOWED BY A STRING OF DECIMAL DIGITS. THERE CAN BE NO SPACES OR TABS BETWEEN THE START AND END OF A NUMBER. AT OTHER POINTS, SPACES AND TABS ARE IGNORED IN COMMAND LINES.

SHOULD A NUMBER NOT BE SPECIFIED, THE LAST NUMBER SUPPLIED IS ASSUMED. IF NO VALUE HAS BEEN SUPPLIED, A ZERO IS ASSUMED AT THE START OF EACH COMMAND LINE.

CERTAIN LETTER SWITCHES MAY NOT HAVE NUMBERS ASSOCIATED WITH THEM. THESE SWITCHES MAY NOT BE PRECEEDED BY A NUMBER UNLESS THERE IS AN INTERVENING LETTER SWITCH THAT ALLOWS A NUMBER.

ILLEGAL CHARACTERS OR BAD SYNTAX RESULT IN AN ERROR WHEN PROCESSING JOBS. THESE ERRORS ARE NOT DETECTED WHEN THE SCRIPT IS LOADED INTO THE HIGH SEGMENT.

A COMMAND LINE MAY INCLUDE A COMMENT BY USING A SEMICOLON. THE SEMICOLON AND ANY CHARACTERS REMAINING ON THE COMMAND LINE ARE DELETED WHEN THE SCRIPT IS LOADED AND SO USE NO CORE AT RUN-TIME.

A COMMAND LINE MUST NOT END WITH A NUMBER. THUS A LETTER SWITCH MUST FOLLOW ANY NUMBER IN A COMMAND LINE.

5.4 LETTER SWITCHES

I	IF NON-NEGATIVE ARGUMENT SUPPLIED DELAY	-- SETS TYPE-IN
	IF NEGATIVE ARGUMENT SUPPLIED RATE.	-- SETS TYPE-IN
O	SAME AS I EXCEPT THAT IT SETS TYPE-OUT RATE.	
R	IF NON-NEGATIVE ARGUMENT SUPPLIED RESPONSE TIME	-- SETS ALLOWED
F	IF NON-NEGATIVE ARGUMENT SUPPLIED	-- SETS FREE TIME
S	IF NON-NEGATIVE ARGUMENT SUPPLIED	-- SETS STAGGER TIME
T	IF POSITIVE ARGUMENT SUPPLIED TIMES TO DO SCRIPT.	-- SETS NUMBER OF
C	IF A POSITIVE ARGUMENT SUPPLIED DELAY BEFORE PROGRAM SENDS ↑C↑C	-- SETS MAXIMUM
	IF ZERO ARGUMENT SUPPLIED ↑C↑C TIME-OUT	-- INHIBITS SENDING
L	NO ARGUMENTS ALLOWED	-- SETS L MODE FLAG
N	NO ARGUMENTS ALLOWED FLAG	-- CLEARS L MODE
U	NO ARGUMENTS ALLOWED	-- SETS U MODE FLAG
V	NO ARGUMENTS ALLOWED FLAG	-- CLEARS U MODE

5.5 TIMING UNITS

TIMES ARE IN MILLISECONDS. RATES ARE IN MILLISECONDS PER CHARACTER.

5.6 L MODE

WHEN L MODE IS SET <CR> AND <LF> WILL NOT BE SENT OVER THE PTY. THUS A LINE ENDING WITH <ALTMODE> CAN BE SENT. ANY LINE MUST END HOWEVER WITH A FULL CHARACTER SET BREAK CHARACTER DUE TO A LIMITATION IN THE PTY.

5.7 U MODE.

WHEN SET U MODE CONVERTS "#" IN TEXT LINES INTO A STRING OF TWO OCTAL DIGITS EQUAL TO THE PTY UNIT NUMBER. (LEADING ZEROS WILL BE INCLUDED.) ALL DEC SCRIPTS USE THIS FEATURE TO LOG IN THE JOB UNDER PPN 4,77700 THROUGH 4,77777 (I.E., 4,777#).

6.0 OPERATING INSTRUCTIONS

6.1

BUILD A TIME-SHARING SYSTEM WITH SUFFICIENT JOBS AND PSEUDO-TELETYPES. ONE CONTROL JOB WILL BE NEEDED FOR EACH 14 JOBS FOLLOWING THE SCRIPT.

6.2

CREATE AN APPROPRIATE SCRIPT.

6.3

START THE SCRIPT PROGRAM.

6.4

WHEN IT ASKS IF A SCRIPT IS TO BE LOADED, TYPE "YES <CR>".

6.5

TYPE THE DEVICE, FILE NAME, AND EXTENSION. IF ARGUMENTS ARE NOT SUPPLIED, THEY DEFAULT TO DSK:SCRIPT.

6.6

THE PROGRAM WILL RESPOND WITH "LOADED!". IF THERE WERE ERRORS IT WILL ASK INSTEAD IF A SCRIPT IS TO BE LOADED. GO BACK TO STEP 6.4.

6.7

IF THE SCRIPT IS NOT GOING TO BE SHARED GO TO STEP 6.10.

6.8

RETURN TO MONITOR MODE AND SAVE THE LOADED SCRIPT WITH A SSAVE COMMAND. EXAMPLE: "SSAVE DSK SCRIPT <CR>".

6.9

START N COPIES OF THE PROGRAM BY LOGGING IN N JOBS AND GIVING THE COMMAND "RUN DSK SCRIPT" OR WHAT EVER IS NEEDED TO LOAD THE SAVED VERSION FROM STEP 6.8. EACH JOB WILL ASK IF A SCRIPT IS TO BE LOADED, ANSWER "NO<CR>". THE INSTRUCTIONS IN THE REMAINING STEPS SHOULD BE FOLLOWED FOR EACH OF THE N JOBS.

6.10

THE PROGRAM WILL NOW ASK HOW MANY JOBS ARE TO BE RUN. ENTER THE NUMBER FOR EACH CONTROL JOB ON ITS TTY. THE MAXMIMUM NUMBER IS 14 JOBS PER CONTROL JOB. EACH OBJECT JOB NEEDS A PTY.

6.11

THE PTY NAMES WILL BE TYPED OUT AS THEY ARE INITED BY EACH SCRIPT JOB. SHOULD THERE BE TOO FEW, THEN THE SCRIPT PROGRAM WILL RELEASE ALL GOTTEN SO FAR AND ASK OVER AGAIN HOW MANY JOBS ARE TO BE RUN.

6.12

THE FIRST JOB RUN BY A GIVEN SCRIPT JOB CAN BE MONITORED ON DEVICE TTY. THIS IS USEFUL FOR DEBUGGING A NEW SCRIPT. ANSWER THE QUESTION "DO YOU WANT TO WATCH FIRST JOB ON TTY?" APPROPRIATELY. INCIDENTALLY, IF DEVICE TTY HAS BEEN ASSIGNED TO A DIRECTORY DEVICE, THIS OUTPUT WILL BE GIVEN THE FILE NAME "MONITR". SHOULD THE DEVICE BE UNAVAILABLE, OR THE FILE UNENTERABLE, AN ERROR MESSAGE WILL APPEAR ON THE TELETYPE AND THE QUESTION WILL BE ASKED AGAIN.

SHOULD AN ERROR OCCUR ON DEVICE TTY WHILE RUNNING, THE RUN WILL NOT BE SUSPENDED. HOWEVER, SUBSEQUENT MONITORING WILL BE INHIBITED.

6.13

IF THE SCRIPT CALLS FOR FASTER OUTPUT THAN DEVICE TTY CAN HANDLE, THEN THE SCRIPT JOB WILL GO INTO I/O WAIT. THIS WILL RESULT IN VERY LONG DELAY TIMES ATTRIBUTED TO THE SCRIPT PROGRAM. THIS CONDITION SHOULD BE AVOIDED BY CHANGING THE SCRIPT PARAMETERS, USING A FASTER DEVICE TTY OR NOT MONITORING AT ALL.

6.14

THE PROGRAM WILL NOW ASK, "DO YOU WANT RESPONSE TIMES LOGGED?" A "NO<CR>" ANSWER WILL CAUSE THE SCRIPT TO BEGIN. IN THIS CASE, GO TO STEP 6.16.

6.15

IF THE QUESTION WAS ANSWERED "YES<CR>", THE PROGRAM WILL ASKED FOR A DEVICE, FILE NAME AND EXTENSION. ARGUMENTS NOT SUPPLIED WILL DEFAULT TO DSK:LOGFIL.

IF THE DEVICE IS NOT AVAILABLE, OR IF THE FILE CAN NOT BE ENTERED, GO BACK TO STEP 6.14.

SHOULD A DATA ERROR OCCUR WHILE RUNNING, THE RUN WILL PROCEED WITH SUBSEQUENT LOGGING INHIBITED. AN ERROR MESSAGE WILL APPEAR ON THE TELETYPE.

6.16

JOBS UNDER CONTROL OF A GIVEN CONTROL PROGRAM WILL BE STAGGERED BY AN INTERVAL SPECIFIED IN THE SCRIPT. IF MULTIPLE SCRIPT JOBS ARE RUNNING, THE USER CAN STAGGER THEIR TIMING BY ENDING STEP 6.14 OR 6.15 AT THE APPROPRIATE TIME OF DAY.

6.17

WHEN THE LAST JOB UNDER A GIVEN SCRIPT PROGRAM HAS FINISHED, THE MESSAGE "ALL JOBS DONE!" WILL APPEAR ON THE USERS TELETYPE AND THE LOG FILE AND MONITOR FILE WILL BE CLOSED, THE PROGRAM WILL THEN EXIT.

NOTE THAT IF MULTIPLE JOBS ARE RUNNING THE FIRST JOB MAY WELL BE OVER A LONG TIME BEFORE THE LAST. THIS IS THE JOB MONITORED ON DEVICE TTY.

6.18

SHOULD IT BE NECESSARY TO TERMINATE A RUN, STOP THE CONTROL PROGRAM BY TYPING TWO <CONTROL C> CHARACTERS, TO CLOSE OUT THE LOG FILE REENTER THE PROGRAM BY A REENTER COMMAND. A MESSAGE "JOBS ABORTED!" WILL APPEAR ON THE TELETYPE.

IN CASE THIS IS DONE, BE AWARE THAT ANY JOBS RUNNING UNDER SCRIPT PROGRAM CONTROL WILL NOW BE DETACHED. THE USER SHOULD KILL THESE JOBS OFF OR DO A 143 MONITOR RESTART TO FLUSH ALL JOBS.

6.19

THE FORTRAN PROGRAM TOTAL CAN BE USED TO SUMMARIZE THE RESPONSE LOG FILE. IT TAKES INPUT FROM 1:FOR01.DAT AND PLACES A SHORT LISTING ON 6:FOR06.DAT.

DMPFIL IS A FILE DUMP PROGRAM; IT READS A FILE AND PREPARES A PRINTABLE VERSION OF AN OCTAL DUMP OF THE FILE. IT WILL ALSO DUMP A DECTAPE BLOCK BY BLOCK, OR DUMP A SAVE FILE AS A CORE DUMP, OR A DISK FILE STRUCTURE BLOCK BY BLOCK.

ASSEMBLY INSTRUCTIONS:

```
        .COMPILE  DMPFIL

        THERE ARE NO ASSEMBLY OPTIONS
```

RUN INSTRUCTIONS:

```
        .R DMPFIL
        *<OUTPUT FILE>=<INPUT FILE>/SWITCHES
```

SWITCHES:

```
        NNNNNND = DUMP DECTAPE, BEGINNING AT BLOCK NNNNNN (OCTAL)
                  INCLUDES LISTING OF DIRECTORY

        NNNNNNK = ASSUME FILE IS SAVE FILE, DUMP AS CORE DUMP, BEGINNING AT LOC NNNNNN
        NNNNNNH = ASSUME FILE IS HIGH SEGMENT SAVE FILE, DUMP AS CORE DUMP

        NNNNNNS = DUMP DISK, BEGINNING AT BLOCK NNNNNN(OCTAL), USING SUPER USETI

        NNNNNNT = STOP DUMP AT NNNNNN(OCTAL)
```

DEFAULT OUTPUT FILE IS LPT: .LST

```
DEFAULT INPUT  FILE IS  DSK:   .
                DEFAULT IS THE SAME FILE ON BOTH SIDES
                IF NO "=", INPUT FILE IS SPECIFIED
                "=" CAN BE USED FOR "="
```

REENTER WILL CLOSE THE FILES.