MASTER COPY

G. E. Friend

MOLD

Multiple On - Line Debugging System
Prepared by the Dartmouth Team

for the

Dartmouth - GE Time Sharing System

Users' Manual
August 23, 1966

1. Introduction

The supervisory system MOLD is designed to enable a number of systems-programmers (perhaps a dozen) to debug their programs simultaneously. Each user will have the freedom of hands-on debugging, and the convenience of teletype control from remote locations.

MOLD uses a GE 625/635 configuration, with communication controlled by a local Datanet-30. Input from teletypes is either directly to this D-30, or chanelled via a remote D-30.

To connect to MOLD, one calls the D-30 from a teletype. When the phone is answered, an identification may be obtained by a WRU (control-E). Communication beyond this point is by a one-line-at-a-time command system (except when building files, see EDIT), waiting for system response before the next command. Each of these must be addressed to one of the four operating systems: GMAP (for compiling), LOADER (for obtaining a core-image of a compiled program), EDIT (for file-manipulation), and DDT (for debugging). These are explained in detail below.

The following information is generally useful. One must start by typing

RUN XXXX

where XXXX is the name of one of the four systems. GMAP and LOADER accept only a single command. (To reuse them, a new RUN is needed.) However, EDIT and DDT permit conversations, hence they are available to the user until he types EXIT.

File names and passwords (which are optional) have up to eight characters (letters and digits only), and they are recorded left-justified, filled with blanks.

Certain teletype keys play special roles: Back-arrow or control-H erases a character, control-X deletes a line, and the break key may be used to abort a job (e.g. stop output). In the initial phase of the system it would be a good idea for the user to push the break key before typing RUN for the first time, to wipe out any incomplete jobs associated with his teletype.

When finished, the user should type TERMIN.

2. GMAP

After the user has typed RUN GMAP, the system will return GMAP HERE when ready. The user will then type an input line of the following form:

SOURCEFILE (, password); LISTFILE (, password); BINFILE (,p); ERRFILE (,p) (cr)

LISTFILE, BINFILE, and ERRFILE stand for the three output files - listing file, binary deck file, and error file. If these are not to be outputted, nothing should be inputted in their place in the input line. Thus

SOURCE;;;ERRFILE, password

would output only the error file. Files used in GMAP must have been previously saved by the user through EDIT. The error file will contain the line numbers of the incorrect "cards" with the error flags, the cross-reference table, and the undefined symbol list. To use GMAP again after the line has been inputted, the user must again type RUN GMAP. The system automatically exits from GMAP after each operation.

GMAP assumes that the file to be compiled has line numbers. These may be supplied by means of EDIT.

LOADER

The purpose of the LOADER is to convert a card image file to a core image file. After typing

RUN LOADER

the system will return

LOADER HERE. YOUR MOVE.

Then, as in GMAP, the user inputs one line

FILEOUT; SYMFILE; MAPFILE; FILE1;--; FILEn

where SYMFILE and MAPFILE are optional and may be omitted by two semi-colons in a row as in GMAP. SYMFILE stands for symbol table and MAPFILE for memory map. Thus the LOADER takes FILE1,--, FILEn and converts them to binary images and puts

them into a single FILEOUT. The output then is the core image, FILEOUT, and the optional reshuffled symbol table and memory map. Passwords may be used, as in GMAP, and all files must have been previously saved. The system automatically exits from LOADER after the completion of the job. GMAP and LOADER also automatically SAVE and CLOSE the files used in their operations.

EDIT

The object of EDIT is to maintain and make changes in the source files. After the typing of

RUN EDIT

the system will return

EDIT HERE

when EDIT has been loaded. The user can then give any one of several commands. The system will remain in EDIT until the user types EXIT as a command. EXIT automatically releases all active files.

Only one file with a given file name can be active at once. Thus, two files with identical names but different passwords cannot be active at the same time. Passwords, if used, are needed only with the commands OLD, SAVE, and UNSAVE. The EDIT commands and their descriptions follow. Several files can be active at once, so each command requires a file name.

NEW FILENAME1; FN2; ...
creates a new scratch file for each name given.

OLD FILENAME, password; ....
recalls previously used and saved files, needs the password used when last saved.

UNSAVE FILENAME, password; ....
destroys permanent files. Files must be inactive at the time.

RENAME OLDNAME; NEWNAME; OLD; NEW; ....
renames OLD file with NEWNAME, inactivates OLD file.

RELEASE FILENAME; FN; ....
inactivates as many files as you name. Passwords not necessary.

SCRATCH FILENAME; ....
destroys temporary copies of files named, name remains available for use.

SAVE FILENAME, password; ....
catalogs files, password needed to get file back. It is not possible to SAVE a file if there is already a filename with that password SAVED. It must first be UNSAVED, if you wish to replace it.

BUILD FILENAME

Normally the user is allowed to type only one line at a time, and must then wait for the system to respond. This is clearly inconvenient for building files. The command BUILD, issued to EDIT, allows the user to type in a large number of lines, without waiting for system response. These lines are added to the file he is building. Clearly only one file can be built at a time, hence BUILD has a single filename as argument. When the building operation is completed, the user types a blank line (i.e. hits the carriage return an extra time). The system will respond READY. Then he can issue a new command to EDIT.

There is a limit of 2000 characters in one BUILD. Since this represents over 3 minutes of typing, at maximum teletype speed, this is not a serious restriction. In any case, though a vacuous carriage return must occur no later than the 2000th character, the user may then type BUILD, and continue building his file.

To make changes in a file the following procedure must be used.

```
OLD A
BUILD A
RENAME A;B
UNSAVE A
RENAME B;A
SAVE A
```

This procudure will be simplified later.

The following EDIT commands are similiar to the EDIT commands in the present Time-Sharing System.

APPEND FILE#1 ; FILEJ; ....
All files are added to the end of the previous file and the resulting file is called FILE#1.

SORT FILENAME
Only one file name is given. The line numbers are sorted and only the last one appears in the file. Thus if there are two lines with number 340, only the last one will appear in the resulting file. A file can contain no more than 4094 lines, and if no line number is given it will be considered as line 0.

SEQUENCE FILE; A,B
Puts line numbers in the file. The original file need not have line numbers. The first line number will be A with the increment equal to B. If A and B are not given, A will be 100 and B will be 10. The only legal line numbers are $0 - 2^{36}-1$.

LIST FILE; A,B, C-D, ...
will list file with lines in the order given. If no parameters are given, a forward list will be given. In a select list, the lines must be in the first 4094 lines of the file. There is no restriction on a non-select list.

DELETE and EXTRACT
are the same as in the present EDIT system. The format is the same as LIST. The parameters must be given in increasing order.

DDT

DDT, like EDIT, can do several operations and depends on the instructions given to it. Like EDIT, you must type "EXIT" as a command to get out of DDT. The purpose of DDT is three-fold.

1) Trace a program by printing out the contents of selected registers or memory locations, at selected times during a RUN of the program.

2) Patch any desired corrections or changes into any location in the assembled program. This can be done without reassembling the program and can be written in the source language of the program.

3) Embed "breakpoints" in the program. The message "BREAKPOINT AT LOCATION XXXXX" is then printed out. The user may then make changes, have memory or registers printed out, or may simply resume execution.

In DDT, the format of a number specifies its type. Thus, 64 octal, 64. is decimal, and 64.0 is floating point.

To allow more than one DDT command on a teletype line, ✲ is used as a substitute for c.r. Thus several commands separated by ✲'s may be typed on a line. A description of the DDT commands follows.

LOAD Filename, Password; SYMFILE, Password✲

A file and its symbol table file created by the loader are loaded into memory. The loader must always be used first to create a core image file of all the programs you want to run under DDT.

FORMAT Mode ✲

All output will be given in that output mode, with one exception. If a command is preceded by an output mode symbol, the output for that single command will be in the specified mode. The available output modes are:

SYM Symbolic Mode. The contents of the location are printed as a mnemonic operation code followed by a symbolic address and as address modification field.

DEC Decimal Mode. The contents of the location are printed as a fixed-point decimal constant.

OCT Octal Mode. The contents of the location are printed as a fixed-point octal constant.

FLO Floating point Mode. The contents of the location are printed as a floating point constant.

BCD Binary Coded Decimal Mode. The contents of the location are printed as 6 alpha-numeric characters of 6 bits each.

ASC ASCII Mode. The contents of the location are printed as 4 alpha-numeric characters of 9 bits each.

HAL Half-word Mode. The contents of the location are printed as two symbolic address expressions (18 bits each).

TAL Tally Mode . The contents of the location are printed as a symbolic address, a tally count, and 6 modification bits which will be printed as 2 octal digits.

DUMP Address List

Each element of the address list is either a single address or a pair of address expressions separated by a comma. The elements of the address list are separated by semi-colons. The contents of these locations are printed out.

PATCH Single address Expression; List of Word Expressions

The word-expressions, which are separated from each other by semi-colons, replace the contents of sequential memory locations at the address given in the first argument.

BREAK A(1),I(1),L(1)U(1); A(2),.....

A single BREAK command can embed several breakpoints. Each breakpoint has from 1 to 4 arguments. Different breakpoints are separated by semi-colons.
The first argument, A(N), is the address at which the the breakpoint is inserted. If A(N) is the only argument supplied, the program halts, the breakpoint message is printed out, and further instructions are awaited. The instruction in the location where the breakpoint is inserted is not executed before the BREAK takes place.
If I(N) is also supplied, the breakpoint will be ignored I(N) times. If it is not specified, it will automatically be 0.
If L(N) is supplied also, the contents of location L(N) will be printed out along with the breakpoint message.
If U(N) is also supplied, L(N) specifies the lower bound of the block of memory to be printed out. U(N) is the upper bound.

Breakpoints cannot be placed in the following locations:
1) Program modified locations.
2) Instructions to be executed by a XEC or XED.

UNBREAK Address List #

The address list may be omitted or may contain any number of single addresses separated by semi-colons. The breakpoints at these addresses are removed. If no list is given all the breakpoints are removed.

CONTINUE I(N) #

Execution of program resumes. The next instruction executed is the instruction at the breakpoint location. If I(N) is supplied, it replaces the I(N) for this breakpoint.

TRANSFER Single Address Expression #

Execution of program commences, or resumes at location given. This must be given to start a program. After a breakpoint, control can be transferred to any location desired.

REGISTER Optional List of Register Identifiers #

The argument list, if supplied, is a list of register identifiers separated by semi-colons. The contents of these registers will be printed out. If no arguments are given, all registers will be printed out. The index registers are specified by a number only, not XN.

ALTER Register Identifier, Expression;... #

The contents of each register specified are replaced by the corresponding expression.

SAVE Program Name #

The program, or programs, currently loaded are saved under the name supplied. All breakpoints, patches, ect, are preserved as they stand.

D.E. Knuth

MØLD

USER MANUAL

SUPPLEMENT NØ. 2 -- ØCT. 8,1966

MØLD HAS NØW RUN SUCCESSFULLY IN BØTH RØME AND PHEØNIX. IN FACT, BILL ADAMS REPØRTED HE GØT 4 DAYS WØRK DØNE IN SEVERAL HØURS USING MØLD. HE SAYS TØ CØUNT HIM AS AN ENTHUSIASTIC MØLDY SUPPØRTER.

.S SUPPLEMENT CØNTAINS SØME WARNINGS CØNCERNING CØMMANDS WHICH WERE EXPLAINED IN PREVIØUS MØLDY CØMMUNICATIØNS. ALSØ SEVERAL NEW CØMMANDS HAVE BEEN ADDED TØ MØLD SYSTEMS. THESE NEW CØMMANDS ARE EXPLAINED BELØW.

MØLD EDIT PACKAGE

BUILD F

WARNING: A MAXIMUM ØF 1024 CHARACTERS CAN BE APPENDED TØ FILE F WITH ØNE USE ØF THE BUILD CØMMAND. IF YØU NEED TØ APPEND MØRE THAN 1024 CHARACTERS YØU MUST EXIT BUILD MØDE PRIØR TØ THE TYPING ØF THE 1024TH CHARACTER AND THEN ISSUE ANØTHER BUILD CØMMAND.

APPEND F1;F2; ... ;FN

NØTE: IN AN APPEND CØMMAND THE SECØND ØCCURRENCE ØF THE FIRST FILE F1 REFERS TØ THE FILE AS MØDIFIED BY THE APPEND CØMMAND UP TØ THAT PØINT. FØR EXAMPLE, THE CØMMAND

```
APPEND A,B,C,A
```

RESULTS IN THE FØLLØWING SEQUENCE ØF ØPERATIØNS

1 FILE B IS APPENDED TØ FILE A GIVING A+B
2 FILE C IS APPENDED TØ FILE A+B GIVING A+B+C
3 FILE A+B+C IS APPENDED TØ A+B+C GIVING A+B+C+A+B+C

AS THE FINAL RESULT. THE RESULTANT FILE HAS THE NAME A AND REPLACES THE PREVIØUS CØNTENTS ØF A.

MØLD DEBUGGING PACKAGE

EXTEND E#

THE EXTEND CØMMAND IS USED FØR SETTING THE MEMØRY BØUND ØF THE PRØGRAM BEING DEBUGGED. EXTEND TAKES ØNE ARGUMENT WHICH IS AN EXPRESSIØN REPRESENTING THE FIRST CØRE LØCATIØN NØT REFERENCED BY THE PRØGRAM. THE DEBUGGING SYSTEM WILL REJECT ANY EXECUTIVE CØMMAND TØ THE MØLD FILE SYSTEM THAT REFERENCES MEMØRY NØT IN THE PRØGRAM'S BØUND. ALSØ THE USER MAY NØT PATCH ØR DUMP MEMØRY NØT IN HIS PRØGRAM'S BØUND. THE MEMØRY BØUND IS ALSØ CHANGED BY THE LØAD CØMMAND TØ PØINT TØ THE FIRST LØCATIØN NØT USED BY THE CØRE IMAGE PRØGRAM. THE SAVE CØMMAND ALSØ MAKES USE ØF THE MEMØRY BØUND IN THAT IT ØNLY SAVES THAT SECTIØN ØF CØRE WITHIN THE USER'S BØUNDS.

ZERØ M1,N1,E1;M2,N2,E2;...;MJ,NJ,EJ#

ZERØ IS USED FØR SETTING A BLØCK ØF MEMØRY TØ A CØNSTANT. WHAT ZERØ DØES IS SET THE BLØCK WHØSE LØWER ADDRESS IS M AND WHØSE UPPER ADDRES IS N TØ E. IF E IS MISSING THEN THE BLØCK BETWEEN M AND N IS SET TØ ZERØ. IF BØTH N AND E ARE MISSING THEN JUST THE SINGLE LØCATIØN M IS SET TØ ZERØ.

MERGE F1;F2; ... ;FN

THIS CØMMAND MERGES FILES F2 THRØUGH FN INTØ FILE F1. THE KEYS USED IN THE SØRT ARE 1) THE ØRDER IN WHICH THE FILE NAME APPEARED IN THE MERGE CØMMAND, AND 2) THE LINE NUMBER ØF THE LINE IN THE FILE. SØ A LINE (L,F) , WHERE L IS THE LINE NUMBER AND F IS THE FILE NAME, WILL PRECEDE A LINE (M,A) IF L<M ØR IF L=M AND F ØCCURRED BEFØRE A IN THE MERGE CØMMAND.

WEAVE F1: F2; ... ;FN

THIS CØMMAND IS EQUIVALENT TØ THE EXECUTIØN ØF THE FØLLØWING TWØ CØMMANDS:

1 APPEND F1;F2; ... ;FN
2 SØRT F1

ADDRESS MØDE#

ADDRESS IS USED TØ SET THE ADDRESS MØDE. IT TAKES ØNE ARGUMENT WHICH MUST BE ØNE ØF:

1) ØCTAL ADDRESSES(PRINT AS 6 DIGIT ØCTAL NUMBERS)
2) DECIMAL ADDRESSES(PRINT AS DECIMAL NUMBERS)
3) SYMBØLIC ADDRESSES(PRINT AS SYMBØL + A CØNSTANT)

SYMBØLIC, WHICH ISN'T IMPLEMENTED YET, WILL BE NØRMAL MØDE. HØWEVER, UNTIL SYMBØLIC MØDE IS WØRKING, ØCTAL MØDE WILL BE THE NØRMAL MØDE.

ØPTIØN ØPT1;ØPT2;...;ØPTJ#

ØPTIØN IS USED TØ SET THE DEBUGGER'S METHØD ØF HANDLING FAULTS. THE FØLLØWING ØPTIØNS ARE AVAILABLE

MEMØRY;NØMEMØRY	MEMØRY FAULTS RETURNED.
MME;NØMME	MME'S AND FILE CALLS.
FAULT;NØFAULT	FAULT TAG FAULTS RETURNED.
TIMER;NØTIMER	TIMER RUNØUT FAULTS RETURNED.
CØMMAND;NØCØMMAND	CØMMAND FAULTS RETURNED.
DERAIL;NØDERAIL	DERAIL FAULTS RETURNED.
LØCKUP;NØLØCKUP	LØCKUP FAULTS RETURNED.
CØNNECT;NØCØNNECT	CØNNECT FAULTS RETURNED.
PARITY;NØPARITY	PARITY FAULTS RETURNED.
ZERØ;NØZERØ	ZERØ ØP FAULTS RETURNED.
ØNC:NØØNC	ØP NØT CØMP. FAULTS RETURNED.
STARTUP;NØSTARTUP	STARTUP FAULTS RETURNED.
ØVERFLØW;NØØVERFLØW	ØVERFLØW FAULTS RETURNED.
DIVIDE;NØDIVIDE	DIVIDE CHECK FAULTS RETURNED.
EXECUTE;NØEXECUTE	EXECUTE FAULTS RETURNED.
MØLD;NØMØLD	MME'S GIVEN TØ THE MØLD EXEC.

THE NØ ØPTIØN ASKS THAT THE SPECIAL FAULT HANDLING ØPTIØN BE TURNED ØFF. NØRMAL MØDE IS ALL ØPTIØNS ØFF. SHØULD ALL ØPTIONS BE TURNED ØN THEN ANY PRØGRAM RUN UNDER THE DEBUGGING SYSTEM WILL RECEIVE FAULTS AND HANDLE FILE CALLS EXACTLY AS IF UNDER THE MØLD EXECUTIVE DIRECTLY. THE ØNLY FILE CALL THAT IS NØT SIMULATED IS THE "GETCAT" CALL. FAULTS ARE RETURNED IN THE SAME MANNER AS THE MØLD EXECUTIVE.

FILE CALLS---

THE DEBUGGING PACKAGE ALSØ ALLØWS THE USER TØ TYPE IN AS CØMMANDS FILE CALLS TØ THE MØLD EXECUTIVE. THESE CALLS WILL BE EXECUTED EXACTLY AS IF THE USER'S PRØGRAM MADE THE CALL ITSELF. THE AVAILABLE CØMMANDS ARE

```
READ FR1,M1,N1;FR2,M2,N2 ...
APPEND FR1,M1,N1;FR2,M2,N2 ...
REWIND FR1;FR2 ...
SCRATCH FR1;FR2 ...
CLØSE FR1;FR2 ...
DESTRØY FR1,NAME,PSW;...
CATLØG FR1,FR2,NAME,PSW;...
ØPEN FR1,NAME,PSW;...      ØR   ØPEN ;...
```

IN THØSE CØMMANDS REQUIRING A PASSWØRD, IF THE PASSWØRD IS MISSING THEN NØ PASSWØRD (A PASSWØRD ØF ZERØ) IS ASSUMED. IF THE FIRST CHARACTER ØF THE NAME ØR PASSWØRD IS A SLASH, THEN THE REMAINING CHARACTERS ØF THE NAME ØR PASSWØRD ARE ASSUMED TØ BE ØCTAL DIGITS WHØSE VALUE WILL BE USED IN PLACE ØF THE ASCII NAME ØR PASSWØRD. A SPECIAL CASE IS THE ØPEN CØMMAND. IF ITS ARGUMENTS ARE NULL THEN IT ØPENS A SCRATCH FILE. ØTHERWISE IT ØPENS THE FILE WITH SPECIFIED NAME AND PASSWØRD FRØM THE CATLØG SPECIFIED IN FILE REFERENCE NUMBER. ØN THE INITIAL MØLD SYSTEM THERE IS ØNLY ØNE CATALØG AND ITS FILE REFERENCE NUMBER SHØULD BE 1.

```
LIST

TABLE     10:07   JULY 1,1966

100  LAST CHANGE:  11:40 JUNE 30, 1966 (KML)
110
120 JUNE 21,  1966 (KML)
130
140
150
160  DOCUMENTATION OF THE MOLD SYSTEM
170
180
190  INTRODUCTORY MATERIAL
200
210    MMNT1
220
230
240  FILE SYSTEM
250
260    MFINIT
270    MFC
280    MF1
290    MFC1
300    MF2
310    MFC2
320    MFC4
330
340  EDITOR
350
360    MEDITS
370    MEDIT1
380    MEDIT2
390
400
410  DDT
420
430    MDDTS
440    M1DDT
450    M2DDT
460    M3DDT
470    M4DDT
480    M5DDT
490
500
510  GMAP
520
530    MGMAP1
540
550
560  LOADER
570
580    LDINIT
590    MLOADS
600    MLOAD1
```

SYSTEM CONVENTIONS

MCHAR
MISC
FAULTV

SYSTEM SERVICES

SETRAP
STIMER
SWAPPE
SPAWN
STOP

IN PREPARATION

MSERV	SERVICES SUPPLIED BY MOLD TO A JOB
LISTEN	CONVENTIONS FOR LISTEN(TOP PROGRAM IN THE SPAWN TREE)
LOADPR	LOADING PROCEDURES
OPERAT	OPERATIONAL PROCEDURES
D-30	D-30 DOCUMENTATION
ACCESS	INTERPRETATION OF ACCESS LOCKS FOR CATALOG AND DATA FILES; AND HOW THE LOCKS CAN BE SET AND RESET.

MINT1 18:15 JUNE 28, 1966

```
100  MØLD INTRØDUCTIØN
120
140
142     MØLD WILL BE AVAILABLE ØVER DIAL-CØMM AND INDIRECTLY
144   THRØUGH THE DARTMØUTH DATANET-30.  THE ANSWER BACK
146   DRUM WILL NØT BE INTERRØGATED.  IF MØLD IS AVAILABLE
150   THE PHRASE "MØLDY HERE" WILL BE TYPED ØN THE REMØTE CØNSØLE.
152   THE USER WILL THEN HAVE A FEW MINUTES TØ TYPE "HELLØ".
154   IF HE DØES NØT TYPE "HELLØ" HIS CØNSØLE WILL
156   BE DISCØNNECTED AND THE PHØNE WILL BE HUNG UP.  AFTER THE
158   USER HAS TYPED IN "HELLØ" MØLD WILL TYPE ØUT
160   "USER NUMBER  " .  THE USER WILL AGAIN BE GIVEN
162   A GRACE PERIØD TØ CØMPØSE A RESPØNSE.  IF THE RESPØNSE IS
164   A VALID USER NUMBER THE WØRD "READY" WILL BE TYPED BY MØLD.
166   AT THIS PØINT THE USER CAN REQUEST THAT ANY SYSTEM
168   AVAILABLE UNDER MØLD BE RUN.  THE REQUEST IS MADE BY TYPING
170   "RUN" FØLLØWED BY THE NAME ØF THE SYSTEM WHICH IS TØ BE RUN.
172   FØR EXAMPLE, A REQUEST TØ RUN GMAP WØULD BE ENTERED AS
174   FØLLØWS:  "RUN GMAP".  THE ØTHER STANDARD CØMMAND
176   WHICH MØLD WILL RECØGNIZE IS "GØØDBYE" WHICH WILL HAVE THE
178   SAME EFFECT AS IT DØES ØN THE DARTMØUTH TIME SHARING
180   SYSTEM ... YØU WILL BE LØGGED ØUT.  THE MØLD EXECUTIVE NØRMALLY
182   RECØGNIZES ØNLY THREE CØMMANDS "HELLØ", "RUN X" AND "GØØDBYE".
184   THIS MEANS THAT ANY USEFUL WØRK MUST BE DØNE AFTER A CØNVERSATIØN
186   WITH ØNE ØF THE SYSTEMS AVAILABLE UNDER MØLD:  INITIALLY ØNLY
188   GMAP, GLØAD, DDT AND EDITØR WILL BE AVAILABLE.  THE MØLD
190   EXECUTIVE WILL ALSØ RECØGNIZE A BREAK CHARACTER AS A PANIC
192   AT THE REMØTE TERMINAL;  IF ANY SYSTEM IS RUNNING, PUSHING THE
194   "BREAK" KEY WILL ABØRT IT.  NØRMALLY THE SYSTEM WILL
200   PRØVIDE LØCAL EDITING VIA A BACKSPACE CHARACTER AND A
210   LINE DELETE CHARACTER.  THESE 2 CHARACTERS WILL BEHAVE EXACTLY
230   AS THEY DØ IN THE PRESENT DARTMØUTH TIME SHARING SYSTEM.
240   HØWEVER,  THE CHARACTER ØN THE KEYBØARD WILL BE DEVICE
255   DEPENDENT ... MØDEL 35 TELETYPES USE "BACK ARRØW" AND
256   THE "ALT MØDE" AS THE TWØ LØCAL EDITING CØNTRØL CHARACTERS.
260   EVENTUALLY SYSTEMS SHØULD BE ABLE TØ SUPPRESS THE LØCAL
270   EDITING ØPTIØN.
272
274     IN SUMMARY, MØLD WILL HAVE THE FØLLØWING FEATURES:
276
278      CØMMANDS
280               HELLØ
282               RUN XXXXX
284               GØØDBYE
286
```

```
LOCAL EDITING
          "BACK ARROW" DELETES ONE CHARACTER
          "ALT MODE" DELETES THE LINE

PANIC
          BREAK CHARACTER WILL ABORT A JOB

SYSTEMS AVAILABLE

          GMAP
          GLOAD
          DDT
          EDITOR


  SEE THE FILE WITH THE SAME NAME AS THE SYSTEM FOR
  A DESCRIPTION OF THE SYSTEM.
```

```
LIST


MFINIT     16:48   JUNE 30,1966

10  LAST CHANGE 1110, 24/6/66 (KML)
20
30
100 PHASE 4 FILE IMPLEMENTATIØN
110
120    INITIALLY MØLD WILL HAVE ØNLY ØNE MASTER CATALØG AND
130   AND ALL CATALØG ACCESSES WILL BE DØNE IN THIS CATALØG.
140   NAMES ØF FILES WILL CØNSIST ØF 2 36-BIT WØRDS AND PASSWØRDS
150   MUST BE A HALF WØRD QUANTITY SINCE THEY ARE TRANSMITTED TØ THE
160   EXECUTIVE THRØUGH AN INDEX REGISTER.  INITIALLY ØNLY
170   THE FØLLØWING FILE CØMMANDS WILL BE AVAILABLE:
180
190      ØPEN T,FR,NAME,PASSWØRD
200
210            IF A SCRATCH FILE IS DESIRED USE THE NAME (0,0)
220            I.E. A DØUBLE LENGTH ZERØ.  SINCE THERE IS ØNLY
230            ØNE CATALØG THE FILE REFERENCE NUMBER FR IS
240            REDUNDANT(IN FACT MEANINGLESS) BUT IT SHØULD
250            BE USED IF ØNLY TØ REMIND YOU THAT IT WØNT BE LØNG
260            BEFØRE YØU HAVE TØ SUPPLY THE FILE REFERENCE
270            NUMBER ØF A CATALØG FILE.
280
290
300
310      CATALØG T,FR,FR,F,A
320
330            THIS CØMMAND WØRKS AS DESCRIBED EXCEPT THAT THE FIRST
340            FILE REFERENCE NUMBER (WHICH SHØULD REFERENCE A CATALØG)
350            IS MEANINGLESS AND THAT THE ACCESS LØCKS ARE ALSØ NØT
360            USED BY THE EXECUTIVE.
370
380
390
400      DESTRØY T,FR,F
410
420
430            THIS CØMMAND ALSØ WØRKS AS DESCRIBED EXCEPT
440            THAT THE FILE REFERENCE NUMBER IS NØT USED.
450
460
470
480            THE REMAINING CØMMANDS WHICH WILL BE AVAILABLE
490            INITIALLY WILL WØRK AS DESCRIBED AND THEREFØRE ARE
500            ØNLY LISTED HERE.
510
520
530      READ T,FR,N,M
540
550      APPEND T,FR,N,M
560
570      REWIND T,FR
580
590      SCRATCH T,FR
600
```

```
OLD
OLD PROBLEM NAME--MFCS

READY.

LIST


MFCS       11:34   JUNE 24,1966

100  LAST CHANGE HERE:  1523,  23/6/66  [IRJ]
110
120
130
140
150  >>>CLOSE  T,FR
160
170  >>>>>OPEN  T,FR,F
180
190  >>>>>>>OPEN  T
200
210  >>>>>>>>>READ  T,FR,N,M
220
230  >>>>>>>>>>>REWIND  T,FR
240
250  >>>>>>>>>>>>>SKIP  T;FR,N
260
270  >>>>>>>>>>CATLOG T,FR,FR,F,A
280
290  >>>>>>>>>REPLAC T,FR,FR,F,A
300
310  >>>>>>>DESTRO  T,FR,F
320
330  >>>>>APPEND  T,FR,N,M
340
350  >>>SCRATC  T,FR
360
370
380
390
400                NOTATION:
410
420
430          T:     TRAP PROGRAM LOCATION
440
450          FR:    FILE REFERENCE NUMBER
460
470          F:     FILE IDENTIFIER (OF FORM: <FILENAME,PASSWORD>)
480
490          N:     NUMBER OF WORDS (OR ENTRIES)
500
510          M:     MEMORY LOCATION
520
530          A:     ACCESS PROHIBITIONS
```

```
LIST


MFT        17:05   JUNE 24,1966

100 NØLD FILE SYSTEM
102
104 TERMINØLØGY
106
108 SECØNDARY STØRAGE
110  ANY ØNE ØF THE FØLLØWING PHYSICAL DEVICES:
112  1 DRUM
114  2 DS-250
116  3 DS- 200
118  4 RACE
120  5 MAGNETIC TAPE
122  7 CARDS
124  8 REMØTE CØNSØLES
126  9 TYPEWRITER
127.BRE
128 10 CARD READER
130
136 SECØNDARY STØRAGE IS CØMPØSED ØF TWØ CLASSES ØF WØRDS:
138  CLASS 1  WØRDS ASSIGNED FØR USE BY A FILE, AND
140  CLASS 2  WØRDS WHICH ARE AVAILABLE FØR ASSIGNMENT , I.E.
142           FREE STØRAGE.
144
146 FILE
148   A FILE IS ANY SEQUENCE ØF N (N MAY BE 0) WØRDS ØF SECØNDARY
150 STØRAGE ASSIGNED BY THE EXECUTIVE FØR USE BY A PRØGRAM.
152
154 TYPES ØF FILES
156
158  CATALØG FILE
160   A CATALØG FILE IS A FILE WHICH CØNSISTS SØLELY ØF
162 ENTRIES WHICH CATALØG ØTHER FILES. THAT IS EACH ENTRY
164 NAMES A FILE ØF THE SYSTEM.  CATALØGS HAVE A RIGID FØRMAT
166 WHICH IS DETERMINED BY THE EXECUTIVE.  ALSØ THE USER CAN ØNLY
168 MØDIFY CATALØG ENTRIES BY CALLING UPØN EXECUTIVE RØUTINES WHICH
170 CØNVERT THE PARAMETERS ØF THE CALL TØ THE CØRRECT FØRMAT.
178
180 LØGICAL CØNTENTS ØF A CATALØG ENTRY
181
182   NAME ØF FILE
184   ACCESS LØCKS(READ,APPEND,WRITE,EXECUTE,RETIRE,TRAP)
186   PASSWØRD ØR NAME ØF TRAP FILE
188   STATISTICAL INFØRMATIØN (DATE ØF CREATIØN, ØPENS SINCE CREATIØN)
189   NUMBER ØF DATA WØRDS IN FILE
190   BEGINNING AND ENDING DEVICE ADDRESS, PLUS FØRMAT
192   TYPE ØF FILE (CATALØG ØR DATA FILE)
194   ATTACHMENT LEVEL(CØUNT ØF USERS ATTACHED TØ THE FILE)
195   APPEND FLAG (ØNLY ØNE ØF THE USERS ATTACHED TØ THE FILE
196                CAN BE APPENDING)
197
```

DATA FILE
  ANY FILE WHICH IS NØT A CATALØG. THE FØRMAT AND CØNTENT ØF DATA FILES ARE THE SØLE CØNCERN ØF THE USER AND ARE NØT RESTRICTED IN ANY WAY BY THE EXECUTIVE.

ALL FILES CØNTAIN HEADERS AND TRAILERS WHICH ARE USED FØR LINKING PHYSICAL BLØCKS AND FØR REDUNDANCY CHECKING. HØWEVER, HEADERS(AND TRAILERS) ARE INVISIBLE TØ THE PRØGRAM USING THE FILE.

ATTACHED VS. UNATTACHED FILES

EACH FILE HAS AN ATTACHMENT LEVEL WHICH IS INCREMENTED BY ØNE EACH TIME THE FILE IS ØPENED AND IS DECREMENTED EACH TIME THE FILE IS CLØSED. THUS THE ATTACHMENT LEVEL CØUNTS THE NUMBER ØF FILE REFERENCE NUMBERS ASSIGNED TØ THE FILE.

UNATTACHED FILE
 AN UNATTACHED FILE IS A CATALØGUED FILE WITH AN ATTACHMENT LEVEL ØF 0. THUS AN UNATTACHED FILE IS NØT IN USE.

ATTACHED FILE
AN ATTACHED FILE IS ANY FILE WITH AN ATTACHMENT LEVEL WHICH IS GREATER THAN 0. AN ATTACHED FILE HAS THE FØLLØWING PRØPERTIES:

1. IT IS IN USE BY ØNE ØR MØRE JØBS,
2. FØR EACH JØB WHICH IS USING THE FILE THE EXECUTIVE SYSTEM HAS SUPPLIED A "FILE REFERENCE NUMBER" WHICH MUST BE USED WHEN REFERENCING THE FILE.(THE FILE REFERENCE NUMBER IS RETURNED AFTER A SUCCESSFUL ØPEN.)
3. FILE MAY BE UNCATALØGUED .... AN UNCATALØGUED FILE IS ØNLY TEMPØRARY IN THE SENSE THAT IF IT IS CLØSED BEFØRE BEING CATALØGUED ITS CØNTENTS ARE LØST.

ØNLY ATTACHED FILES MAY BE READ, APPENDED TØ, ØR USED AS A CATALØG. AN ATTACHED FILE MAY BE IN ØNE ØF THE FØLLØWING THREE SUBSTATES:

1. NEUTRAL — THAT STATE WHICH EXISTS AFTER A SUCCESSFUL ØPEN BUT PRIØR TØ THE FIRST READ ØR APPEND REQUEST.
2. APPENDABLE — ENTERED ØN EACH APPEND REQUEST.
3. READABLE — ENTERED ØN EACH READ, SKIP ØR REWIND REQUEST.

.PAG

CATALØGUED VS UNCATALØGUED FILES

CATALØGUED FILE
 ANY FILE WHICH IS REFERENCED BY AN ENTRY IN A CATALØG.

UNCATALØGUED FILE
 ANY FILE WHICH HAS BEEN GIVEN A FILE REFERENCE NUMBER (IT IS IN USE) BUT HAS NØT YET BEEN CATALØGUED.

```
LIST

MFC1      13:31   JUNE 22,1966

10   JUNE 22 (KML)
100 MØLD FILE CØMMANDS
102
104  EACH CØMMAND IS IN THE FØRM ØF A SYSTEM MACRØ.  THE MACRØ
106 PRØTØTYPE WILL BE SUPPLIED TØ EACH SYSTEM PRØGRAMMER.  EVENTUALLY
108 THESE MACRØS WILL BE A PART ØF MØLD'S SYSTEM LIBRARY.
110
112  WHEN A FILE CØMMAND IS ISSUED TØ THE EXECUTIVE THE USER MUST
114 REALIZE THAT:
116           (1) ALL REGISTERS (A,Q,I AND INDEX REGISTERS) WILL
118               HAVE BEEN DESTRØYED,
120           (2) THE FUNCTIØN REQUESTED BY THE CØMMAND WILL NØT HAVE
122               BEEN CARRIED ØUT,
124           (3) INDEX REGISTER 0 WILL BE ZERØ IF THE CØMMAND WAS
126               ACCEPTED AND NØN ZERØ IF THE CØMMAND IS
128               SYNTACTICALLY INCØRRECT, AND
130           (4) THAT EACH CØMMAND MUST SPECIFY A TRAP PRØGRAM
132               WHICH WILL BE INVØKED WHEN THE REQUEST
134               HAS BEEN CARRIED TØ CØMPLETIØN ØR ABØRTED DUE TØ
136               AN ERRØR ØF SØME SØRT.
138 THE PRØGRAM CAN THEN CØNTINUE EXECUTIØN IN PARALLEL WITH THE
140 EXECUTIØN ØF THE FILE ØPERATIØN.  WHEN THE EXECUTIVE HAS
142 CØMPLETED THE TASK (ØR TASKS) REQUESTED BY THE CØMMAND THE
144 PRØGRAM WILL BE INTERUPTED AND CØNTRØL WILL BE TRANSFERRED
146 TØ THE TRAP PRØGRAM.  THERE MUST BE A SEPARATE TRAP PRØGRAM
148 FØR EACH FILE CØMMAND ØUTSTANDING.  IF THE PRØGRAMMER IS
150 CLEVER HE WILL SEE THAT HE CØULD USE THE SAME TRAP RØUTINE
152 IF HE PRØVIDES THE EXECUTIVE WITH 4 WØRDS WHICH PURPØRT TØ BE
154 THE FIRST FØUR WØRDS ØF A TRAP PRØGRAM.  THE EXECUTIVE ASSUMES
156 THAT THE FIRST FØUR WØRDS ØF THE TRAP PRØGRAM CAN BE USED FØR
158 ØR ALWAYS CØNTAIN THE FØLLØWING:
160     WØRD 1      STATUS RETURN WØRD 1
162                 SRW1(18,35) CØNTAINS THE TYPE ØF RETURN
164     WØRD 2      STATUS RETURN WØRD 2
166     WØRD 3      THE EXECUTIVE STØRES THE INTERUPT LØCATIØN
168                 PLUS 1 AND THE INDICATØRS IN THIS LØCATIØN.
169                 THE EXECUTIVE DØES THE EQUIVALENT ØF A
170                 "STC1 WØRD3".  THE TRAP PRØGRAM MUST
171                 EXIT BY EXECUTING "RET WØRD3 ".
172     WØRD 4      FIRST LØCATIØN ØF THE TRAP PRØGRAM
174 WARNING:  THE ØRDER IN WHICH THE PRØGRAM EXECUTES FILE
176           CØMMANDS WILL PRØBABLY NØT BEAR ANY RELATIØN TØ THE
178           ØRDER IN WHICH THE ASSØCIATED TRAP PRØGRAMS ARE
180           ENABLED.  DØN'T TRY TØ DISCERN ØRDER WHERE NØNE
182           EXISTS.
184
186
188 ØPENING A FILE
190  IN ØRDER TØ USE ANY FILE IT MUST FIRST BE ASSIGNED A
192 FILE REFERENCE NUMBER.  ALL SUBSEQUENT CØMMANDS WHICH AFFECT
194 THIS FILE MUST SUPPLY THE FILE REFERENCE NUMBER AS A PART ØF
196 THE CØMMAND. THUS THE FILE REFERENCE NUMBER IS THE NAME WHICH
198 THE EXECUTIVE AND THE PRØGRAM USE TØ DESCRIBE THE FILE WHILE
200 IT IS ATTACHED TØ THE PRØGRAM.
202
```

OPENING A CATALOGUED FILE

ØPEN T,FR,NAME,PASSWØRD,(ACCESS LIST)

T	LØCATIØN ØF TRAP PRØGRAM
FR	FILE REFERENCE NUMBER ØF CATALØG FILE TØ SEARCH
NAME	NAME ØF THE FILE
PASSWØRD	MUST BE EQUAL TØ PASSWØRD IN CATALØG ENTRY
(ACCESS LIST)	LIST ØF THE ØPERATIØNS WHICH THE PRØGRAM WILL BE CARRYING ØUT ØN THE FILE. MUST BE ØNE ØF THE FØLLØWING: APPEND,READ,CØPY, EXECUTE ØR LINK.

EXAMPLE ØPEN TRAPL,FRCAT, BASBAL, PL, (READ,EXECUTE)

THIS CØMMAND REQUESTS THAT A CATALØGUED FILE BE ASSIGNED A FILE REFERENCE NUMBER.

TYPE ØF RETURN	MEANING
0	NØRMAL RETURN SRW1(0,17) CØNTAINS THE FILE REFERENCE NUMBER ASSIGNED BY THE EXECUTIVE SRW2(0,11) FILE ACCESS FLIP FLØPS 0 0 INDICATES READ PERMISSIØN 1 0 INDICATES APPEND PERMISSIØN 2 0 INDICATES WRITE PERMISSIØN 3 0 INDICATES EXECUTE PERMISSIØN 4-11 THESE BITS ARE NØT YET USED SRW2(12,35) CØNTAINS THE CØUNT ØF THE NUMBER ØF WØRDS IN THE FILE
1	NØT APPLICABLE
2	NØT APPLICABLE
3	UNRECØVERABLE ERRØR
4	FILE REFERENCE NUMBER (FR) HAS NØT BEEN ASSIGNED
5	PRØTECTIØN VIØLATIØN I.E. THE TYPE ØF ACCESS REQUESTED IS NØT ALLØWED ØR THE PASSWØRD WAS NØT EQUAL TØ THAT IN THE CATALØG
6	CATALØGUE FR HAS NØ ENTRY WITH SAME FILE NAME EQUIVALENT TØ "PRØGRAM NØT SAVED"
7	BUSY

ØPENING AN UNCATALØGUED FILE(TEMPØRARY)

ØPEN T

EXAMPLE ØPEN TRAPL

THIS CØMMAND REQUESTS THE ASSIGNMENT ØF A FILE REFERENCE NUMBER FØR A FILE WHICH INITIALLY CØNTAINS 0-WØRDS AND WHICH MAY BE ACCESSED IN ANY ALLØWED MANNER.

TYPE ØF RETURN	MEANING
0	NØRMAL RETURN SRW1 AND SRW2 HAVE THE SAME MEANING AS ØPEN FØR A CATALØGUED FILE
3	UNRECØVERABLE ERRØR
6	NØ SPACE AVAILABLE FØR TEMPØRARY STØRAGE THIS SHØULD NØT ØCCUR
7	BUSY

```
LIST


MFC2      13:32   JUNE 22,1966

100 MØLD FILE CØMMANDS(CØNTINUED)
110
120 CØMMANDS WHICH APPLY ØNLY TØ ATTACHED FILES
130
140  ØNLY ATTACHED FILES CAN USE THE READ, APPEND,
150 REWIND, SKIP ØR CLØSE CØMMANDS.  ØNLY ATTACHED
160 UNCATALØGUED FILES CAN BE REFERENCED WHEN USING
170 THE CATALØGUE CØMMAND.  THE DESTRØY CØMMAND
180 (WHICH DESTRØYS A CATALØGUE ENTRY) CANNØT BE
182 USED WITH ATTACHED FILES.
190
200  READ T,FR,N,M
210       T          TRAP LØCATIØN
220       FR         FILE REFERENCE NUMBER
230       N          NUMBER ØF WØRDS TØ BE READ
240       M          MEMØRY LØCATIØN INTØ WHICH FIRST WØRD GØES
250
260        THIS CØMMAND REQUESTS THE NEXT N WØRDS FRØM
270        FILE FR BE PLACED IN LØCATIØNS M, M+1, ... M+(N-1).
271
272    TYPE ØF RETURN      MEANING
273         0              NØRMAL RETURN
274         1              END ØF RECØRD (SRW2 CØNTAINS THE RESIDUE)
275         2              END ØF FILE   (SRW2 CØNTAINS THE RESIDUE)
276         3              UNRECØVERABLE ERRØR
277         4              FR HAS NØT BEEN ASSIGNED
278         5              PRØTECTIØN VIØLATIØN
279         7              FILE BUSY
280
300  REWIND T,FR
310       T          TRAP LØCATIØN
320       FR         FILE REFERENCE NUMBER
330
340        THIS CØMMAND RESETS THE READ PØINTER SØ THAT A
350        SUBSEQUENT READ CØMMAND WILL GET THE FIRST N
360        WØRDS ØF THE FILE.
```

```
371    TYPE ØF RETURN      MEANING
372         0              NØRMAL RETURN
373         4              FR HAS NØT BEEN ASSIGNED
374         5              PRØTECTIØN VIØLATIØN
375         7              FILE BUSY
376
400  SKIP T,FR,N
410       T          TRAP LØCATIØN
420       FR         FILE REFERENCE NUMBER
430       N          NUMBER ØF WØRDS TØ SKIP ØVER
440
450       THIS CØMMAND ADDS N TØ THE READ PØINTER.  IT IS
460       EQUIVALENT TØ READING N WØRDS INTØ A SCRATCH AREA.
470
471    TYPE ØF RETURN      MEANING
472         1              END ØF RECØRD (SRW2 CØNTAINS THE RESIDUE)
473                          THE NEXT WØRD READ WILL BE THE ØNE
474                          FØLLØWING THE END ØF RECØRD WØRD
475         2              END ØF FILE  (SRW2 CØNTAINS THE RESIDUE)
476         3              UNRECØVERABLE ERRØR
477         4              FR NØT ASSIGNED
478         5              PRØTECTIØN VIØLATIØN
479         7              FILE BUSY
480
500  APPEND T,FR,N,M
510       T          TRAP LØCATIØN
520       FR         FILE REFERENCE NUMBER
530       N          NUMBER ØF WØRDS TØ APPEND TØ THE FILE
540       M          MEMØRY LØCATIØN ØF THE N WØRDS
550
560       THIS CØMMAND REQUESTS THAT THE N WØRDS IN MEMØRY
570       LØCATIØNS M, M-1,... M-(N-1) BE ADDED TØ THE
580       END ØF FILE FR.
581
582    TYPE ØF RETURN      MEANING
583         0              NØRMAL RETURN
584         1              END ØF RECØRD (SRW2 CØNTAINS THE RESIDUE)
585         2              END ØF FILE (SRW2 CØNTAINS THE RESIDUE)
586         3              UNRECØVERABLE ERRØR
587         4              FR NØT ASSIGNED
588         5              PRØTECTIØN VIØLATIØN
589         7              FILE BUSY
590
```

```
   APPENDING, ETC. ; IN Ø.

STØP.
READY.

LIST


MFC3       11:41   JUNE 24,1966

0   LAST MØDIFIED  1137, 24/6/66  (KML)
1
2
100 MØLD FILE CØMMANDS(CØNTINUED)
104
108 CØMMANDS WHICH APPLY ØNLY TØ ATTACHED UNCATALØGUED FILES
112
116  TEMPØRARY (ATTACHED UNCATALØGUED) FILES CAN BE USED AS
120 A SCRATCH PAD.  THAT IS THE CØNTENT ØF THE FILE CAN BE
124 SCRATCHED(FILE IS THEN EMPTY) WITHØUT LØSING THE FILE REFERENCE
128 NUMBER.  ALSØ TEMPØRARY FILES CAN BE CATALØGUED ØR USED
132 TØ REPLACE THE FILE CØNTENTS ØF A CATALØGED FILE.
136
140  SCRATC T,FR
144       T          TRAP LØCATIØN
148       FR         FILE REFERENCE NUMBER
152
156         CØMMAND RESETS THE FILE WØRD CØUNT TØ ZERØ.  THE
160         FILE REFERENCE NUMBER MAY STILL BE USED FØR READING,
164         APPENDING, ETC. ; IN ØTHER WØRDS, SCRATCH PUTS THE FILE
168         BACK IN THE CØNDITIØN IMMEDIATELY FØLLØWING AN ØPEN.
172         THE CØMMAND IS FULLY EQUIVALENT TØ A CLØSE FØLLØWED
176         BY AN ØPEN WITH A GUARANTEE THAT THE FILE REFERENCE
180         NUMBER RETURNED AFTER THE ØPEN IS THE SAME AS THAT USED
184         IN THE CLØSE.
188
192    TYPE ØF RETURN      MEANING
196         0              NØRMAL RETURN
200         3              UNRECØVERABLE ERRØR
204         4              FR NØT ASSIGNED
208         5              PRØTECTIØN VIØLATIØN
212         7              FILE BUSY
```

```
216  CATLØG T,FR1,FR2,NAME,PASSWØRD,(ACCESS PRØHIBITIØNS)
220      T           TRAP LØCATIØN
224      FR1         CATALØG FILE REFERENCE NUMBER
228      FR2         UNCATALØGUED FILE
232      NAME        FILE NAME
236      PASSWØRD    PASSWØRD TØ BE PUT IN ENTRY
240      (A P)       TYPES ØF ACCESS WHICH WILL BE PRØHIBITED
244
248      CATLØG CAUSES THE FILE WITH FILE REFERENCE NUMBER FR2
252      TØ BE ENTERED INTØ THE CATALØG FILE WITH FILE REFERENCE
256      NUMBER FR1.  THE ENTRY WILL BE CREATED USING THE NAME
260      PASSWØRD AND ACCESS PRØHIBITIØNS LISTED IN THE MACRØ.
264      THE PRØGRAM RETAINS THE FILE AS AN ATTACHED CATALØGUED
268      FILE ØPENED FØR ALL TYPES ØF ACCESS.
272
276    TYPE ØF RETURN      MEANING
280        0               NØRMAL RETURN
284        3               UNRECØVERABLE ERRØR
288        4               FR1 ØR FR2 NØT ASSIGNED
292        5               PRØTECTIØN VIØLATIØN
296        6               ENTRY WITH SAME NAME IN FR1
300        7               FILE BUSY
304
308  REPLAC T,FR1,FR2,NAME,PASSWØRD,(ACCESS PRØHIBITIØNS)
312      T           TRAP LØCATIØN
316      FR1         CATALØG FILE REFERENCE NUMBER
320      FR2         UNCATALØGUED FILE
324      NAME        FILE NAME
328      PASSWØRD    PASSWØRD TØ BE PUT IN ENTRY
332      (A P)       TYPES ØF ACCESS WHICH WILL BE PRØHIBITED
336
340      REPLAC CAUSES FILE FR2 TØ REPLACE THE FILE PØINTED TØ
344      BY THE ENTRY(WITH THE SAME NAME) IN CATALØG FILE FR1.
348      THE CØMMAND IS EQUIVALENT TØ A DESTRØY FØLLØWED BY A
352      CATLØG.
356
360    TYPE ØF RETURN      MEANING
364        0               NØRMAL RETURN
368        3               UNRECØVERABLE ERRØR
372        4               FR1 ØR FR2 NØT ASSIGNED
376        5               PRØTECTIØN VIØLATIØN
380        7               FILE BUSY
```

```
LIST


MFC4       13:44   JUNE 22,1966

100 MØLD FILE CØMMANDS(CØNTINUED)
104
108 CLØSING AN ATTACHED FILE
112
116  ANY PRØGRAM WHICH ØPENS A FILE SHØULD CLØSE THAT FILE
120  PRIØR TØ TERMINATING.  IF THE PRØGRAM DØES NØT CLØSE
124  THE FILE THE SUBSEQUENT ACTIØN ØF THE EXECUTIVE IS
128  UNDEFINED AND WILL PRØBABLY BE UNPREDICTABLE.
132
136    THØSE FILES WHICH HAVE BEEN PASSED ALØNG VIA A SPAWN
140  REQUEST WILL NØT BE CLØSED BUT WILL BE PASSED BACK TØ
144  THE JØB WHICH REQUESTED THE SPAWN.  IF THE JØB WHICH
148  PASSED ALØNG THE FILE THRØUGH THE SPAWN CØMMAND DID
152  NØT REQUEST THAT THE FILE BE PASSED BACK THE FILE WILL BE
156  CLØSED IN A NØRMAL FASHIØN.  THUS THE PRØGRAM SETTING UP
160  THE SPAWN IS PARTIALLY PRØTECTED FRØM THE VAGARIES ØF THE
164  PRØGRAM AT THE LØWER LEVEL ØR LEVELS.
168
172
176  CLØSE T,FR
180         T          TRAP LØCATIØN
184         FR         FILE REFERENCE NUMBER
188
192        IF FR REFERS TØ AN UNCATALØGUED FILE THEN THE
196        SPACE ALLØTED IS RETURNED TØ FREE STØRAGE.  IF
200        FR REFERS TØ A CATALØGUED FILE THAN THE CATALØG
204        ENTRY CØRRESPØNDING TØ THIS FILE WILL BE UPDATED
208        (ATTACHMENT LEVEL DECREASED AND IF THE PRØGRAM APPENDED
212        TØ THE FILE THE APPEND FF IS RESET) AND IF THE
216        FILE HAS ACTUALLY BEEN MØDIFIED IT IS WRITTEN ØN THE
220        INCREMENTAL DUMP TAPE.  IN EITHER CASE THE FILE
224        REFERENCE NUMBER IS NØ LØNGER ASSIGNED AND MAY BE
228        REASSIGNED BY THE EXECUTIVE.
232
236   TYPE ØF RETURN   MEANING
240         0          NØRMAL
244         3          UNRECØVERABLE ERRØR
248         4          FR NØT ASSIGNED
252         5          PRØTECTIØN VIØLATIØN
253         7          FILE BUSY
256
260
264
```

```
CATALØG MAINTENANCE

 READ T,FR,N,M
        T           TRAP LØCATIØN
        FR          FILE REFERENCE NUMBER ØF CATALØG FILE
        N           NUMBER ØF CATALØG ENTRIES TØ BE READ
        M           MEMØRY LØCATIØN

        PRØVIDES AN EDITED LØØK AT THE CØNTENTS ØF THE
        NEXT N ENTRIES ØF THE CATALØG WITH FILE REFERENCE
        NUMBER FR.  THE INITIAL IMPLEMENTATIØN WILL RETURN
        TWØ WØRDS PER ENTRY. THE ASCII CHARACTERS
        USED AS A NAME FØR THE FILE WILL BE LEFT JUSTIFIED IN
        THE 2 WØRDS.  ADDITIØNAL INFØRMATIØN WILL BE PRØVIDED
        AD HØC.

  TYPE ØF RETURN      MEANING
        0             NØRMAL RETURN
        1             END ØF RECØRD (SRW2 CØNTAINS THE RESIDUE)
        2             END ØF FILE   (SRW2 CØNTAINS THE RESIDUE)
        3             UNRECØVERABLE ERRØR
        4             FR HAS NØT BEEN ASSIGNED
        5             PRØTECTIØN VIØLATIØN
        7             FILE BUSY

 REWIND T,FR
       T          TRAP LØCATIØN
       FR         FILE REFERENCE NUMBER

        THIS CØMMAND RESETS THE READ PØINTER SØ THAT A
        SUBSEQUENT READ CØMMAND WILL GET THE FIRST N
        CATALØG ENTRIES ØF THE FILE.

  TYPE ØF RETURN      MEANING
        0             NØRMAL RETURN
        4             FR HAS NØT BEEN ASSIGNED
        5             PRØTECTIØN VIØLATIØN

 SKIP T,FR,N
      T          TRAP LØCATIØN
      FR         FILE REFERENCE NUMBER
      N          NUMBER ØF ENTRIES TØ SKIP ØVER

        THIS CØMMAND ADDS N TØ THE READ PØINTER.  IT IS
        EQUIVALENT TØ READING N ENTRIES INTØ A SCRATCH AREA.

  TYPE ØF RETURN      MEANING
        1             END ØF RECØRD (SRW2 CØNTAINS THE RESIDUE)
        2             END ØF FILE  (SRW2 CØNTAINS THE RESIDUE)
        3             UNRECØVERABLE ERRØR
        4             FR NØT ASSIGNED
        5             PRØTECTIØN VIØLATIØN
        6             NAME NØT IN CATALØG
        7             FILE BUSY
```

```
488   DESTRØ T,FR,NAME,PASSWØRD
492         T              TRAP LØCATIØN
496         FR             FILE REFERENCE NUMBER ØF CATALØG
500         NAME           NAME ØF ENTRY TØ BE DESTRØYED
504
508         THIS CØMMAND WILL DESTRØY THE ENTRY WITH THE NAME
512         SPECIFIED IN THE CATALØG FILE FR IF THE USER
516         HAS ØPENED CATALØG FR FØR WRITING AND THE PASSWØRD
520         SUPPLIED MATCHES THAT IN THE ENTRY.
524
528   TYPE ØF RETURN       MEANING
532         1              NØRMAL RETURN
536         3              UNRECØVERABLE ERRØR
540         4              FR NØT ASSIGNED
544         5              PRØTECTIØN VIØLATIØN
548         6              NØ ENTRY WITH NAME SPECIFIED
549         7              FILE BUSY
552
556
560   HANDLING SPASTIC TYPE FILES ... THAT IS FILES WHICH
564   PRØDUCE DATA AT THEIR DISCRETIØN AT NØT NECESSARILY AT
568   "SPASTICS".
572
576
580   UDFC "USER DEFINED FILE CØMMANDS"
```

```
ØLD
ØLD PRØBLEM NAME--MEDITS

READY.

LIST


MEDITS     13:39   JUNE 22,1966

100  LAST CHANGE HERE:  2259,  20/6/66 [TRJ]
110
120
130
140
150  JUNE 16,1966  KML
160
170
180
190        SUMMARY ØF CØMMANDS FØR MØLDY EDITØR
200
210
220         ØLD   F;F; ..... ;F
230
240         NEW   N;N; ..... ;N
250
260         RENAME  N;N;N;N; ..... ;N;N
270
280         SCRATCH  N;N; ... ;N
290
300         LIST  N;B,B, ... ,B
310
320         SØRT    N
330
340         DELETE N;B,B, ... ,B
350
360         EXTRACT  N;B,B, ... ,B
370
380         RESEQUENCE  N;L,L,I
390
400         SAVE  F;F; ..... ;F
410
420         UNSAVE  F;F; .....;F
430
440         APPEND  N;N; ... ;N
450
460         BUILD  N
470
480         RELEASE  N;N; ... ;N
490
500         EXIT
510
```

```
520
530                 NØTATIØN
540
550         N:    NAME ØF A FILE
560
570         P:    PASSWØRD(ØPTIØNAL)
580
590         B:    BLØCK ØF LINES(TWØ LINE NUMBERS SEPARATED BY "-")
600
610         L:    LINE NUMBER
620
630         I:    INTEGER
640
650         F:    FILE IDENTIFIER(ØF FØRM: <FILENAME,P>)
660
670
680
```

```
)LD
)LD PRØBLEM NAME--MEDIT1

READY.

LIST


MEDIT1     13:41   JUNE 22,1966

100  LAST CHANGE HERE:  2305,  20/6/66 [TRJ]
105
110
115
120 JUNE 15,1966 (KML)
125
130
135
140  * * *   A LITTLE STØRY ABØUT THE MØLDY EDITØR   * * *
145
150
155
160  THE MAINTENANCE ØF ALL SØURCE FILES IN MØLD WILL NØRMALLY
165 BE DØNE BY USING THE SYSTEM CALLED "EDITØR".  EDITØR
170 CAN BE CALLED BY TYPING "RUN EDITØR" IF YØU ARE NØT IN A
175 CØNVERSATIØN WITH SØME ØTHER SYSTEM PRØGRAM.  IF YØU ARE
180 IN A CØNVERSATIØN WITH ANØTHER SYSTEM PRØGRAM YØU MUST
185 FIRST EXIT FRØM THAT PRØGRAM BY TYPING "EXIT".
190
195  AFTER EDITØR IS LØADED IT WILL TYPE EDITØR AND WAIT FØR
200  A CØMMAND TØ BE ISSUED.  AFTER A LEGITIMATE CØMMAND HAS BEEN
205  ISSUED EDITØR CARRIES ØUT THE FUNCTIØN REQUESTED BY THE CØMMAND.
210  WHEN FINISHED WITH THE TASK EDITØR TYPES ØUT "READY" AND
215  WAITS FØR ANØTHER CØMMAND.  IF THE CØMMAND HAPPENS TØ BE
220  UNRECØGNIZABLE ØR INVALID FØR SØME REASØN "INVALID EDITØR
225  CØMMAND" WILL BE TYPED.  AFTER EITHER RESPØNSE TØ A CØMMAND
230  EDITØR IS READY TØ ACCEPT ANØTHER CØMMAND.  DURING THE
235  EXECUTIØN ØF THE CØMMAND ØTHER CØMMANDS WILL NØT BE
240  RECØGNIZED.
245
250   IN CØNTRAST TØ THE DARTMØUTH TIME SHARING SYSTEM MØLD
255  ALLØWS A USER TØ BE WØRKING WITH MØRE THAN ØNE FILE AT A
260  TIME.  THIS MEANS THAT EDITØR CØMMANDS SUCH AS LIST,DEL, ETC.
265  MUST NØW GIVE THE NAME ØF THE FILE WHICH IS TØ BE USED WHEN
270  EDITØR CARRIES ØUT THE CØMMAND.  ALSØ SØME CØMMANDS SPECIFY A
275  PASSWØRD WHICH CAN BE USED AT THE USERS ØPTIØN.
280
285  THERE IS A RESTRICTIØN ØN FILENAMES AND PASSWØRDS, NAMELY THAT
290  THEY MUST NØT CØNTAIN:
295
300          1)   LEADING BLANKS
305          2)   CØLØNS  [:]
310          3)   SEMI-CØLØNS  [;]
315          4)   CØMMAS  [-,]
320
325  IN THE CØMMAND DESCRIPTIØNS BELØW, A "FILE IDENTIFIER" IS ØF THE
330  FØLLØWING FØRM:  <FILENAME>,<ØPTIØNAL PASSWØRD>
```

```
340
345  ØLD   FILE IDENTIFIER;FILE IDENTIFIER; ..... ;FILE IDENTIFIER
350
355   GETS CATALØGUED (SAVED) FILES FØR USE DURING THE RUNNING
360   ØF EDITØR.  IF A PASSWØRD IS INVALID ØR THE USER'S FILE
365   DIRECTØRY DØES NØT CØNTAIN AN ENTRY WITH A FILE NAME
370   GIVEN IN THE CØMMAND, EDITØR WILL TYPE "INVALID EDIT CØMMAND".
375   ALL SUBSEQUENT CØMMANDS TØ EDITØR RELATING TØ THE FILE MUST USE
380   THE FILENAME (WHICH MAY BE CHANGED BY RENAME).
385
390  NEW   FILENAME;FILENAME; ..... ;FILENAME
395
400   ØPENS TEMPØRARY FILES WHICH WILL BE GIVEN THE NAMES WHICH
405   APPEAR IN THE CØMMAND.  ALL SUBSEQUENT CØMMANDS RELATING
410   TØ ØNE ØF THESE FILES MUST USE THE NAME ØF THE FILE.
415   TEMPØRARY FILES MUST BE SAVED DURING AN EDITØR RUN ØR THEY
420   WILL BE LØST ØN EXIT FRØM EDITØR.
425
430  RENAME  NAME1A;NAME1B;NAME2A;NAME2B; ..... ;NAMENA;NAMENB
435
440   RENAMES FILE NAME1A TØ NAME1B, FILE NAME2A TØ NAME2B, ETC.,
445   THE NAMES NAME1A, NAME2A, ETC., MUST HAVE BEEN USED
450   PREVIØUSLY IN AN ØLD, NEW, ØR RENAME CØMMAND.
455
460  SCRATCH  NAME1;NAME2; ..... ;NAMEN
465
470   THE CØNTENTS ØF THE FILES NAMED ARE LØST BUT THE NAMES ARE
475   STILL AVAILABLE FØR USE.  IN EFFECT THIS ERASES THE CØNTENTS
480   ØF A FILE BEING USED AS A SCRATCH PAD.
485
490  RELEASE  NAME1;NAME2; ..... ;NAMEN
495
500   THE FILES NAMED ARE RELEASED.  THAT IS TØ SAY, THE CØNTENTS
505   ARE LØST AND THE NAMES ARE NØ LØNGER AVAILBLE FØR USE
510   IN EDITØR CØMMANDS.  IF YØU HAVE A SAVED FILE ØF THE SAME NAME
515   IT IS UNALTERED.
520
525  UNSAVE  FILE IDENTIFIER;FILE IDENTIFIER; ..... ;FILE IDENTIFIER
530
535   IF THE PASSWØRD(S) MATCH THØSE IN THE CATALØG ENTRY(S)
540   WITH THE NAME(S) GIVEN, THE ENTRY(S) WILL BE EXPUNGED FRØM
545   THE FILE DIRECTØRY(CATALØG).
550
555  SAVE  FILE IDENTIFIER;FILE IDENTIFIER; ..... ;FILE IDENTIFIER
560
565   FØR EACH FILE IDENTIFIER GIVEN:
570   IF NØ ENTRY ØF THE SAME NAME APPEARS IN THE FILE
575   DIRECTØRY, AN ENTRY WITH THE NAME AND PASSWØRD
580   GIVEN IN THE CØMMAND WILL BE CREATED.  THE CØNTENT
585   ØF THE FILE WILL BE THE CØNTENT ØF THE FILE WITH THE
590   SAME NAME BEING USED IN THE EDITØR RUN.  IF THE CATALØG
595   CØNTAINS A ENTRY ØF THE SAME NAME, EDITØR WILL TYPE
600   "FILE ØF SAME NAME SHALL I REPLACE IT?"... ANSWER YES ØR NØ.
605
610
615
620
625  CØNTINUED IN MEDIT2
```

```
LIST


MEDIT2    13:42   JUNE 22,1966

100  LAST CHANGE HERE:  2142,  20/6/66 [TRJ]
110
120
130
140
150  CATALØG
160
170   LISTS THE NAMES ØF ALL ENTRIES IN THE CURRENT FILE DIRECTØRY.
180   THIS CØMMAND HAS EXACTLY THE SAME EFFECT AS THE CØMMAND
190   ØF THE SAME NAME IN THE DARTMØUTH TIME SHARING SYSTEM.
200
210  APPEND  NAME;NAME1;NAME2; ..... ;NAMEN
220
230   THE CØNTENTS ØF FILES WITH NAMES NAME1 THRØUGH NAMEN ARE
240   APPENDED TØ THE FILE NAME.  THE FILES NAME1, NAME2, ...
250   ARE NØT ALTERED BY THIS CØMMAND.
260
270  BUILD NAME
280
290   THIS CØMMAND IS USED TØ ADD TELETYPED INPUT TØ A FILE.
300   AFTER RECEIPT ØF THE CØMMAND EDITØR WILL TYPE GØ AHEAD.
310   THE EDITØR WILL APPEND ALL SUBSEQUENT CHARACTERS TYPED IN
320   AT THE REMØTE CØNSØLE UNTIL FILE BUILDING MØDE IS EXITED.
330   THE EXACT METHØD ØF EXITING FILE BUILDING MØDE HAS
340   NØT YET BEEN SPECIFIED BUT WILL PRØBABLY BE EITHER A VACUØUS
350   LINE ØR A LINE WHICH DØES NØT BEGIN WITH A DIGIT.
360
370  LIST NAME;<BLØCK ØF LINES>,<BLØCK ØF LINES>, ... ,<BLØCK ØF LINES>
380
390   LIST THE CØNTENTS ØF THE FILE "NAME".  THE FILE
400   ITEMS ARE ØMITTED THEN THE ENTIRE FILE WILL BE TYPED
410   ØUT ØN THE REMØTE PRINTER.  IF THE ØPTIØNAL ITEMS ARE
420   INCLUDED THEN ØNLY THE BLØCKS SPECIFIED WILL BE
430   TYPED ØUT.  A BLØCK ØF LINES CAN BE SPECIFIED BY GIVING
440   A SINGLE LINE NUMBER ØR TWØ LINE NUMBERS SEPARATED
450   BY A DASH.
460
470  SØRT NAME
480
490   SØRTS THE FILE ACCØRDING TØ LINE NUMBERS WHICH PRECEDE EACH
500   LINE ØF THE FILE.  IF TWØ LINES HAVE THE SAME NUMBER THE
510   LINE WHICH IS ENCØUNTERED LAST WHEN READING THE FILE FRØM
520   BEGINNING TØ END IS RETAINED.  VACUØUS LINES ARE ØMITTED. THE
530   REMAINING LINES ARE SØRTED AND WRITTEN BACK INTØ THE FILE.
540
550  RESEQUENCE  NAME;N1,N2,N3
560
```

```
570   CAUSES THE LINE NUMBERED FILE WITH THE NAME GIVEN TO BE
580  RESEQUENCED STARTING AT LINE NUMBER N2 WITH THE INTEGER
590  N3 USED AS THE INCREMENT.
600
610  DELETE NAME;<BLOCK OF LINES>,<BLOCK OF LINES>, ... ,<BLOCK OF LINES>
620
630   DELETES THE BLOCKS OF LINES IN THE FILE "NAME".  A BLOCK
640   OF LINES IS EITHER A SINGLE LINE NUMBER OR TWO LINE
650   NUMBERS SEPARATED BY A "-".
660
670  EXIT
680
690   THIS COMMAND IS USED TO EXIT FROM THE EDITOR.
700
995
996
997
998
999   * * * * *       END MOLDY EDIT      * * * * *
```

```
MDDTS     13:43   JUNE 22,1966

1   LAST CHANGE HERE:  0309,  20/6/66
3
6
7
8
9
10      * * * * *   SUMMARY OF COMMANDS FOR MOLDY DDT   * * * * *
12
14
15                  LOAD  N,P; N,P; .....; N,P  #
16
20                  FORMAT   M   #
25
30                  DUMP  A; A; .....; A  #
35
40                  PATCH A1; W; W; .....; W  #
45
60                  BREAK A1,I,A; A1,I,A; .....; A1,I,A  #
61
62                  UNBREAK  A1; A1; .....; A1  #
65
70                  CONTINUE  I  #
75
80                  TRANSFER A1   #
85
90                  REGISTERS  R; R; .....; R  #
95
100                 ALTER  R,W; R,W; .....; R,W  #
102
103                 SAVE  N   #
104
105
110
115
200                       NOTATION:
210
220         A:  SINGLE ADDRESS, OR PAIR OF ADDRESSES SEPARATED BY COMMAS
225
230         A1: SINGLE ADDRESS
235
240         I:  ITERATION COUNT
245
250         R:  REGISTER IDENTIFIER
255
260         M:  OUTPUT MODE
262
263         N:  NAME
264
265         P:  PASSWORD
266
270         W:  EXPRESSION
275
```

PRØBLEM NAME--M1ØDT

READY.

LIST

M1 ØDT 13:44 JUNE 22,1966

100 LAST CHANGE HERE: 1853, 20/6/66
105
110
115
120
125 * * * * * * INTRØDUCTIØN TØ MØLDY ØDT * * * * * *
130
135
140 THE FØLLØWING FILE CONTAINS A DESCRIPTIØN ØF THE INITIAL ØDT PACKAGE
145 TØ BE PRØVIDED ØN THE GE/625. THE PURPØSE ØF THE SYSTEN IS TØ
150 PRØVIDE THE USER WITH CØNVENIENT MEANS TØ:
155
160 A) TRACE A PRØGRAM BY PRINTING ØUT THE CØNTENTS ØF SELECTED
165 REGISTERS ØR MEMØRY LØCATIØNS, AT SELECTED TIMES DURING A RUN
170 ØF THE PRØGRAM. PRINTØUT ØF ØPERATIØN CØDES AND ADDRESSES
     MAY BE IN SYMBØLIC FØRM, AS IN THE ØRIGINAL SØURCE PRØGRAM,
     IF DESIRED.
185
190 B) PATCH ANY DESIRED CØRRECTIØNS ØR CHANGES INTØ ANY LØCATIØN
195 IN THE ASSEMBLED PRØGRAM. THIS CAN BE DØNE WITHØUT
200 REASSEMBLING THE PRØGRAM. THE CHANGES CAN BE WRITTEN IN
205 THE SAME SØURCE LANGUAGE AS THE ØRIGINAL PRØGRAM; NEITHER
210 ØCTAL NØR BINARY PATCHES ARE REQUIRED.
215
220 C) EMBED "BREAKPØINTS" IN THE PRØGRAM. WHEN A BREAKPØINT HAS
225 BEEN REACHED A SPECIFIED NUMBER ØF TIMES IN THE RUNNING ØF
230 A PRØGRAM, EXECUTIØN CEASES, THE BREAKPØINT MESSAGE:
235 "BREAKPØINT AT LØCATIØN XXXXXX" IS PRINTED, SELECTED
240 MEMØRY LØCATIØNS ARE PRINTED, AND ØDT REQUESTS FURTHER
245 CØMMANDS FRØM THE USER'S CØNSØLE. THE USER MAY AT THAT
250 PØINT ADD ØR DELETE BREAKPØINTS, MAKE ADDITIØNS ØR
255 CØRRECTIØNS, TRANSFER TØ A DIFFERENT PØINT IN THE PRØGRAM,
260 PRINT ØUT MEMØRY ØR REGISTERS, ØR SIMPLY RESUME EXECUTIØN.
265
270 IN LIEU ØF THE ABØVE HANDLING ØF A BREAKPØINT, THE USER MAY
275 SPECIFY THAT THE BREAKPØINT MESSAGE AND THE SELECTED
280 LØCATIØNS ARE TØ BE PRINTED AT EACH ENTRANCE TØ THE BREAK-
285 PØINT, BUT THAT EXECUTIØN ØF THE PRØGRAM IS NØT TØ BE
290 INTERRUPTED UNTIL A SPECIFIED NUMBER ØF PASSES THRU THE
295 BREAKPØINT HAVE ØCCURED.
300
305 D) EVENTUALLY IT WILL BE PØSSIBLE, BY TYPING A SINGLE SPECIAL
310 CHARACTER YET TØ BE SELECTED, TØ HALT EXECUTIØN ØF A RUNNING
315 PRØGRAM AND GIVE CØNTRØL TØ ØDT, WHICH WILL BEHAVE JUST AS
320 IT WØULD HAD IT ENCØUNTERED A BREAKPØINT.
325

THE ONLY RESTRICTION ON A USER'S PROGRAM TO INSURE COMPATIBILITY WITH DDT IS THAT ALL SYMBOLS SHOULD BEGIN WITH A LETTER. A SYMBOL THAT BEGINS WITH A NON-LETTER CHARACTER MAY NOT BE REFERENCED IN A DDT COMMAND.

TO RUN A PROGRAM (WHICH MUST BE STORED IN ASSEMBLED, BINARY FORM) UNDER DDT, THE USER MUST GIVE THE FOLLOWING COMMAND TO THE MOLDY EXECUTIVE. "RUN DDT". WHEN DDT RESPONDS WITH A "DDT READY", IT WILL ACCEPT COMMANDS - A "LOAD" BEING THE OBVIOUS FIRST CHOICE.

* * * * * FORMAT OF DDT COMMANDS * * * * *

ALL COMMANDS TO DDT WILL TAKE THE FOLLOWING FORMAT:

<OPTIONAL TEMPORARY OUTPUT MODE MODIFIER> <COMMAND NAME>
<A STRING OF ARGUMENTS SEPARATED BY SEMI-COLONS> <META-CHARACTER>

THE ONE EXCEPTION TO THIS FORMAT WILL BE THE SINGLE-CHARACTER COMMAND REFERRED TO IN D) ABOVE.

FOR THE COMMANDS "CONTINUE" AND "REGISTERS", THE ARGUMENT STRING IS OPTIONAL. FOR ALL CONMANDS EXCEPT "DUMP" AND "REGISTERS", THE TEMPORARY OUTPUT MODE MODIFIER WOULD BE MEANINGLESS.

MORE THAN ONE COMMAND MAY BE TYPED ON A LINE, AND A COMMAND MAY EXTEND OVER MORE THAN ONE LINE.

FOR THE EXAMPLES IN THIS FILE, THE NUMBER SIGN [#] WILL BE USED AS THE META-CHARACTER.

* * * * * A NOTE ABOUT NUMBERS * * * * *

ALL NUMBERS APPEARING IN DDT COMMANDS ARE TAKEN TO BE OCTAL UNLESS SPECIALLY INDICATED OTHERWISE. FOR A FEW INSTRUCTIONS, SUCH AS "DEC", THE ACCOMPANYING NUMBERS ARE AUTOMATICALLY ASSUMED TO BE DECIMAL. TO OTHERWISE ENTER NON-OCTAL NUMBERS, OR OCTAL NUMBERS WHERE ANOTHER TYPE IS EXPECTED, USE ONE OF THE FOLLOWING FORMS:

NNNNNN	NUMBER TAKEN AS OCTAL
NNNNNN.	NUMBER TAKEN AS DECIMAL
NNNNNN.MM	NUMBER TAKEN AS FLOATING POINT

MOLDY DDT HAS A LIMITED COMPETENCE IN ARITHMETIC. IT RECOGNIZES "+" AND "-" IN CALCULATING ADDRESSES AND CONSTANTS. EXPRESSIONS SUCH AS: (ADR1+P0SN7-SIZE) ARE ACCEPTABLE.

CONTINUED IN M2DDT

```
OLD
OLD PROBLEM NAME--M2DDT

READY.

LIST


M2 DDT     13:53   JUNE 22,1966

100  LAST CHANGE HERE;  1934,  20/6/66 .
105
110
115
120
125   * * * * * *    DETAILS OF COMMANDS AVAILABLE    * * * * * *
130
135
140
145   LOAD <PROGRAM NAME>,<PASSWORD>;<PROGRAM NAME>,<PASSWORD> ..... #
150
155          IF THE PASSWORD(S) ARE CORRECT, THE NAMED PROGRAM(S)
160          - IN ASSEMBLED, BINARY FORM - ARE LOADED INTO CORE FROM
165          SECONDARY STORAGE.  THE PROGRAMS MUST ALL BE OF THE SAME
170          TYPE -  ABSOLUTE OR RELOCATABLE.  ONLY ONE CORE IMAGE
175          MAY BE LOADED.
180
185  EXAMPLE:    LOAD PROG1,SHAZAM;SBILLS,IRONM;PUNCHR,CUFFS#
190
195               THIS WOULD LOAD THE THREE PROGRAMS "PROG1", "SBILLS",
               AND "PUNCHR" IF "SHAZAM", "IRONM", AND "CUFFS" WERE
               THE CORRECT PASSWORDS RESPECTIVELY.
210
215
220
225
230   FORMAT <MODE> #
235
240          THE SINGLE ARGUMENT IS ONE OF THE 8 OUTPUT MODE SYMBOLS
245          LISTED BELOW.  WITH ONE EXCEPTION, ALL OUTPUT WILL BE IN
250          THIS MODE UNTIL A NEW FORMAT COMMAND IS GIVEN.
255          THE EXCEPTION IS:  IF A COMMAND IS PRECEDED BY AN OUTPUT
260          MODE SYMBOL, THE OUTPUT PRODUCED BY THAT SINGLE COMMAND
265          WILL BE IN THAT MODE.  WHEN THE OUTPUT FROM THE SINGLE
270          COMMAND IS COMPLETED, THE OUTPUT MODE WILL REVERT TO THE
275          MODE SPECIFIED IN THE LAST FORMAT COMMAND.
280
285  EXAMPLES:     FORMAT SYM#      OR:      FORMAT ASC#
290
295
300
305
310   DUMP <ADDRESS LIST>  #
315
320          EACH ELEMENT OF THE ADDRESS LIST IS EITHER A SINGLE ADDRESS
               EXPRESSION, OR A PAIR OF ADDRESS EXPRESSIONS SEPARATED BY
330          COMMAS.  THE ELEMENTS OF THE ADDRESS LIST ARE SEPARATED BY
335          SEMI-COLONS.  THE CONTENTS OF EACH SINGLE LOCATION, AND OF
340          EVERY LOCATION BOUNDED [INCLUSIVELY] BY THE ADDRESS-PAIRS,
345          ARE PRINTED IN THE CURRENT OUTPUT MODE.  ANY NUMBER OF
350          ADDRESSES OR ADDRESS-PAIRS MAY BE USED AS ARGUMENTS TO A
355          DUMP COMMAND.
360
```

THIS WØULD PRINT THE CØNTENTS ØF THE FIRST 20 LØCATIØNS ØF THE RØUTINE GETPT, PLUS LØCATIØNS PT1 AND PT2, ALL IN THE ØUTPUT MØDE SELECTED BY THE LAST "FØRMAT" CØMMAND.

FØRMAT SYM# DUMP GETPT,GETPT+19.#TAL DUMP PT1; PT2# DUMP STØPT,STØP1;STUFF#

THIS WØULD PRINT THE CØNTENTS ØF THE FIRST 20 LØCATIØNS ØF THE RØUTINE GETPT IN THE SYMBØLIC ØUTPUT MØDE, THEN THE LØCATIØNS PT1 AND PT2 IN THE TALLY MØDE, THEN THE LØCATIØNS FRØM STØPT TØ AND INCLUDING STØPT1 IN THE SYMBØLIC MØDE, THEN THE LØCATIØN STUFF IN THE SYMBØLIC MØDE.

PATCH <SINGLE ADDRESS EXPRESSIØN> ; <A LIST ØF WØRD EXPRESSIØNS> #

THE WØRD EXPRESSIØNS, WHICH ARE SEPARATED FRØM EACH ØTHER BY SEMI-CØLØNS, REPLACE THE CØNTENTS ØF SEQUENTIAL MEMØRY LØCATIØNS STARTING AT THE ADDRESS GIVEN IN THE FIRST ARGUMENT. EVENTUALLY, CERTAIN GMAP PSEUDØ-ØPERATIØNS MAY BE ALLØWED TØ REPLACE A WØRD EXPRESSIØN AS AN ARGUMENT.

EXAMPLES: PATCH MASK+3;777777777000#

THIS WØULD REPLACE WHATEVER HAD BEEN IN LØCATIØN MASK+3 WITH THE ØCTAL CØNSTANT 777777777000.

PATCH JESTER-3;LDA GET,ID;CMPA QUØTE;TZE TF4#

THIS WØULD REPLACE WHATEVER HAD BEEN IN LØCATIØNS:

JESTER-3	WITH	LDA GET,ID
JESTER-2	WITH	CMPA QUØTE
JESTER-1	WITH	TZE TF4

PATCH BARF+11;TRA PAT# PATCH PAT;LDX A,FØØ;ADX A,5,DU; TRA BARF+12#

THE ØRIGINAL INSTRUCTIØN IN LØCATIØN BARF+9(DECIMAL) [REMEMBER THAT THE BARF+11 IN THE PATCH CØMMAND IS ØCTAL] WØULD BE EXPUNGED. THE PRØGRAM WØULD TRANSFER TØ THE INSTRUCTIØN IN LØCATIØN PAT. "PAT" MUST HAVE BEEN DEFINED SØMEWHERE, AND THE THREE LØCATIØNS PAT, PAT+1, AND PAT+2 MUST BE AVAILABLE FØR ØVERWRITING, AS ANY CØNTENTS THEY MAY HAVE HAD WØULD BE DESTRØYED BY THE SECØND PATCH CØMMAND.

CØNTINUED IN MBØDT

```
ØLD
ØLD PRØBLEM NAME--M3DDT

READY.

LIST


M3DDT     13:49   JUNE 22,1966

100  LAST CHANGE HERE:  1831,  20/6/66
105
110
115
120
125   BREAK  <A[1],I[1],L[1],U[1]>  ;  <A[2],I[2],L[2],U[2]>  ;  ....  ;
130                                          <A[N],I[N],L[N],U[N]>  #
135
140          THERE ARE RESTRICTIØNS ØN THE LØCATING ØF BREAKPØINTS;
145          THESE ARE GIVEN AT THE END ØF THE SECTIØN DEALING WITH
150          THE BREAK CØMMANDS.
155
160          A SINGLE BREAK CØMMAND CAN EMBED ANY NUMBER ØF BREAKPØINTS.
165          EACH BREAKPØINT SPECIFIED TAKES A LIST ØF 1 TØ 4 ARGUMENTS,
170          SEPARATED BY CØMMAS.  THE ARGUMENT LISTS FØR DIFFERENT
175          BREAKPØINTS ARE SEPARATED BY SEMI-CØLØNS.  AS THE ACTIØN
180          DEPENDS ØN THE NUMBER ØF ARGUMENTS SUPPLIED, FØUR CASES
185          ARE CØNSIDERED.
190
195          IN EACH CASE THE FIRST ARGUMENT, A[N], IS THE ADDRESS AT
200          WHICH THE BREAKPØINT IS INSERTED.  IF A[N] IS THE ØNLY
205          ARGUMENT SUPPLIED, THE PRØGRAM HALTS, DDT PRINTS THE
210          BREAKPØINT MESSAGE: "BREAKPØINT AT LØCATIØN A[N]", AND
215          AWAITS A CØMMAND FRØM THE CØNSØLE.  THE INSTRUCTIØN IN THE
220          LØCATIØN WHERE THE BREAKPØINT IS INSERTED IS NØT EXECUTED
225          BEFØRE THE BREAK TAKES PLACE.
230
235  EXAMPLES:  BREAK CRUNCH#
240
245                  WHEN THE PRØGRAM REACHES LØCATIØN CRUNCH, THE
250                  INSTRUCTIØN THEREIN IS NØT EXECUTED.  THE MESSAGE
255                  "BREAKPØINT AT LØCATIØN CRUNCH" IS PRINTED.  THE
260                  SYSTEM AWAITS A TYPED CØMMAND, WHICH DETERMINES THE
265                  NEXT ACTIØN.  IF, FØR EXAMPLE, THE USER TYPES IN THE
270                  CØMMAND "CØNTINUE"(WITHØUT ARGUMENT), THE INSTRUCTIØN
275                  IN CRUNCH IS EXECUTED, AND THE PRØGRAM CØNTINUES.
280                  THE BREAKPØINT REMAINS SET, SØ THAT THE PRØGRAM IS
285                  AGAIN INTERRUPTED IF AND WHEN CRUNCH IS REACHED AGAIN.
290
```

THE ACTIØN IS IDENTICAL TØ THE PRECEDING EXAMPLE, EXCEPTING THE APPRØPRIATE CHANGE ØF LØCATIØN IN THE MESSAGE; IT ØCCURS ØN EVERY ARRIVAL AT SNAP, CRACKL ØR PØP.

BREAK CØMMAND WITH 2 ARGUMENTS:

IF TWØ ARGUMENTS, A[N] AND I[N], ARE SUPPLIED TØ THE BREAK CØMMAND, A[N] IS AGAIN THE ADDRESS WHERE THE BREAKPØINT IS INSERTED, AND I[N] IS THE "ITERATIØN CØUNT".

IF I[N]>0, THE BREAKPØINT IS IGNØRED EXCEPT THAT I[N] IS DECREMENTED BY ØNE EACH TIME THE PRØGRAM REACHES LØCATIØN A[N]. WHEN I[N] REACHES ZERØ, ØR IF THE PRØGRAMMER HAS SPECIFIED I[N]=0, BREAK ACTIØN TAKES PLACE AS IF NØ I[N] HAD BEEN SPECIFIED.

IF I[N] HAS BEEN SPECIFIED <0, ØN EACH PASSAGE THRU LØCATIØN A[N] THE BREAKPØINT MESSAGE IS PRINTED AND I[N] IS INCREMENTED BY 1. WHEN I[N] REACHES ZERØ, BREAK ACTIØN TAKES PLACE AS IF NØ I[N] HAD BEEN SPECIFIED.

IF I[N]=0, IT IS NØT FURTHER INCREMENTED ØR DECREMENTED.

EXAMPLE: BREAK VIEWER,-39.;NØW333,0;SEEER,SIGHT÷77Ø

THE ACTIØN: ØN EVERY ARRIVAL AT LØCATIØN VIEWER, THE BREAKPØINT MESSAGE: "BREAKPØINT AT LØCATIØN VIEWER" IS PRINTED; ØN THE 40TH AND SUBSEQUENT ARRIVALS, CØNTRØL IS ALSØ TRANSFERRED TØ THE CØNSØLE. ØN EVERY ARRIVAL AT LØCATIØN NØW333, THE BREAKPØINT MESSAGE: "BREAKPØINT AT LØCATIØN NØW333" IS PRINTED, AND CØNTRØL IS TRANSFERRED TØ THE CØNSØLE. ØN THE (SIGHT÷64)TH AND SUBSEQUENT ARRIVALS AT LØCATIØN SEEER, THE BREAKPØINT MESSAGE: "BREAKPØINT AT LØCATIØN SEEER" IS PRINTED, AND CØNTRØL IS TRANSFERRED TØ THE CØNSØLE.

THE ABØVE EXAMPLE ALSØ ILLUSTRATES THE DECIMAL AND ØCTAL NUMBER CØNVENTIØNS. THE '0' IN 'NØW333;' IS ACCEPTABLE, BUT WITHØUT EFFECT, AND CØULD BE ØMITTED.

BREAK CØMMANDS WITH 3 AND 4 ARGUMENTS: SEE NEXT FILE

CØNTINUED IN M4DDT

```
ØLD
ØLD PRØBLEM NAME--M4DDT

READY.

LIST


M4DDT     13:50   JUNE 22,1966

100  LAST CHANGE HERE:  1833,  20/6/66
105
110
115
120
125   BREAK CØMMAND WITH 3 ARGUMENTS:
130
135         IF THREE ARGUMENTS ARE SUPPLIED TØ THE BREAK CØMMAND, A[N]
140         AND I[N] ARE AGAIN THE ADDRESS WHERE THE BREAKPØINT IS TØ
145         BE INSERTED AND THE ITERATIØN CØUNT RESPECTIVELY.  THE
150         THIRD ARGUMENT, L[N], IS THE ADDRESS ØF A LØCATIØN WHØSE
155         CØNTENTS WILL BE PRINTED ØUT ALØNG WITH THE BREAKPØINT
160         MESSAGE WHENEVER THAT IS PRINTED.  SEE THE 2-ARGUMENT
165         BREAK CØMMAND FØR DETAILS ØF THE SIGNIFICANCE ØF I[N].
170
175  EXAMPLE:    BREAK SMASH,17,JUNKIE;CRASH,0,MUSH;EXAM;LØS,-3,FØUN#
180
185               THE ACTIØN:  ØN THE 16TH AND SUBSEQUENT ARRIVALS AT
190               LØCATIØN SMASH, THE BREAKPØINT MESSAGE: "BREAK-
195               PØINT AT LØCATIØN SMASH" AND THE CØNTENTS ØF
200               LØCATIØN JUNKIE ARE PRINTED, AND CØNTRØL IS
205               TRANSFERRED TØ THE CØNSØLE;
210               ØN EVERY ARRIVAL AT LØCATIØN CRASH, THE BREAKPØINT
215               MESSAGE: "BREAKPØINT AT LØCATIØN CRASH" AND THE
220               CØNTENTS ØF LØCATIØN MUSH ARE PRINTED, AND CØNTRØL
225               IS TRANSFERRED TØ THE CØNSØLE;
230               ØN EVERY ARRIVAL AT LØCATIØN EXAM, THE BREAKPØINT
235               MESSAGE: "BREAKPØINT AT LØCATIØN EXAM" IS PRINTED,
240               AND CØNTRØL IS TRANSFERRED TØ THE CØNSØLE;
245               ØN EVERY ARRIVAL AT LØCATIØN LØS, THE BREAKPØINT
250               MESSAGE: "BREAKPØINT AT LØCATIØN LØS" AND THE
255               CØNTENTS ØF LØCATIØN FØUN ARE PRINTED;
260               ØN THE FØURTH AND SUBSEQUENT ARRIVALS AT LØCATIØN
265               LØS, CØNTRØL IS TRANSFERRED TØ THE CØNSØLE.
270
275
280
285   BREAK CØMMAND WITH 4 ARGUMENTS:
290
295         IF FØUR ARGUMENTS ARE SUPPLIED TØ THE BREAK CØMMAND, A[N]
300         AND I[N] ARE AGAIN THE ADDRESS WHERE THE BREAKPØINT IS TØ
305         BE INSERTED AND THE ITERATIØN CØUNT RESPECTIVELY.  THE
310         THIRD AND FØURTH ARGUMENTS, L[N] AND U[N], ARE THE LØWER
315         AND UPPER BØUNDS ØF THE BLØCK ØF MEMØRY TØ BE PRINTED ØUT
320         ALØNG WITH THE BREAKPØINT MESSAGE WHENEVER THAT IS PRINTED.
325         ALL LØCATIØNS FRØM L[N] TØ U[N] INCLUSIVE ARE PRINTED.
330         SEE THE 2-ARGUMENT BREAK CØMMAND FØR DETAILS ØF
335         THE SIGNIFICANCE ØF I[N].
```

```
EXAMPLE:  BREAK CHAOS,0,KEG,KEG+14,QUEST,0,PANDOR,SAFETY,
                                                -98.,SHELTR,HOME#

                THE ACTION:  ON THE FOURTH AND SUBSEQUENT ARRIVALS
                AT LOCATION CHAOS, THE BREAKPOINT MESSAGE:
                "BREAKPOINT AT LOCATION CHAOS" AND THE CONTENTS
                OF LOCATIONS KEG THRU KEG+14(DECIMAL) ARE PRINTED,
                AND CONTROL IS TRANSFERRED TO THE CONSOLE;
                ON ,EVERY ARRIVAL AT LOCATION QUEST, THE BREAKPOINT
                MESSAGE: "BREAKPOINT AT LOCATION QUEST"" AND THE
                CONTENTS OF LOCATION PANDOR ARE PRINTED, AND CONTROL
                IS TRANSFERRED TO THE CONSOLE;
                ON EVERY ARRIVAL AT LOCATION SAFETY (IF YOU EVER
                MAKE IT) THE BREAKPOINT MESSAGE: "BREAKPOINT AT
                LOCATION SAFETY" AND THE CONTENTS OF ALL LOCATIONS
                SHELTR THRU HOME ARE PRINTED;
                ON THE 99TH AND SUBSEQUENT ARRIVALS AT LOCATION
                SAFETY, CONTROL IS ALSO TRANSFERRED TO THE CONSOLE.



* * * * *    RESTRICTIONS ON BREAKPOINT LOCATION    * * * * *

IN ORDER TO ESTABLISH THE BREAKPOINT, DDT MOVES THE INSTRUCTION
ORIGINALLY FOUND IN THE BREAKPOINT LOCATION TO AN UNSPECIFIED
LOCATION, AND REPLACES IT WITH A TRANSFER TO THE DDT ROUTINE
WHICH EXECUTES THE BREAK.  THIS DOES NOT APPEAR TO THE USER;
DUMPS, PATCHS, ETC., WORK AS IF THE ORIGINAL INSTRUCTION WAS WHERE
IT STARTED.  UNBREAKING RESTORES THE ORIGINAL INSTRUCTION TO ITS
RIGHTFUL PLACE.

DDT, THEREFORE, HAS DIFFICULTY COPING WITH BREAKPOINTS PLACED
AT INSTRUCTIONS WHICH DEPEND ON THEIR LOCATION FOR PROPER
EXECUTION.  IC MODIFIED INSTRUCTIONS CAN BE ACCEPTED AS LOCATIONS
FOR BREAKPOINTS, BUT THE FOLLOWING ARE PROHIBITED PLACES:

        1)  PROGRAM MODIFIED LOCATIONS

        2)  INSTRUCTIONS TO BE EXECUTED BY AN XEC OR XED



UNBREAK <ADDRESS LIST>  #

        THE ADDRESS LIST MAY BE OMITTED, OR MAY CONTAIN ANY NUMBER
        OF SINGLE ADDRESSES SEPARATED BY SEMI-COLONS.  IF NO ADDRESS
        LIST IS SUPPLIED, ALL BREAKPOINTS ARE REMOVED; OTHERWISE
        THE BREAKPOINTS ARE REMOVED FROM THE ADDRESSES SPECIFIED.

EXAMPLES:   UNBREAK  CHAOS;SAFETY#

                THIS WOULD EXPUNGE THE FIRST AND LAST BREAKPOINTS
                INSERTED BY THE PRECEEDING EXAMPLE;  THE BREAKPOINT
                AT 'QUEST' WOULD REMAIN.

            UNBREAK #

                THIS WOULD EXPUNGE ALL BREAKPOINTS CURRENTLY SET.



L70 - CONT'D  IN  M500T
```

```
ØLD
ØLD PRØBLEM NAME--M5DDT

READY.

LIST


M5DDT      13:53   JUNE 22,1966

100  LAST CHANGE HERE:  190C, 20/6/66
105
110
115
120
125   CØNTINUE <ØPTIØNAL ITERATIØN CØUNT>  #
130
135          EXECUTIØN ØF THE PRØGRAM RESUMES.  THE NEXT INSTRUCTIØN
140          EXECUTED IS THE INSTRUCTIØN AT THE BREAKPØINT LØCATIØN.
145          IF THE ØPTIØNAL SINGLE ARGUMENT IS SUPPLIED, THIS REPLACES
150          THE ITERATIØN CØUNT, I[N], FØR THIS BREAKPØINT.
155
160  EXAMPLES:   CØNTINUE  #
165
170                EXECUTIØN ØF THE USER'S PRØGRAM CØNTINUES FRØM THE
175                BREAKPØINT INSTRUCTIØN, WITHØUT FURTHER CHANGES.
180
185              CØNTINUE -9. #
190
195                EXECUTIØN ØF THE USER'S PRØGRAM CØNTINUES FRØM THE
200                BREAKPØINT INSTRUCTIØN; THE ITERATIØN CØUNT, I[N],
205                FØR THIS BREAKPØINT IS RESET TØ -9.
210
215
220
225
230   TRANSFER <SINGLE ADDRESS EXPRESSIØN>  #
235
240          EXECUTIØN ØF THE PRØGRAM CØMMENCES, ØR RESUMES, AT THE
245          INSTRUCTIØN IN THE LØCATIØN WHØSE ADDRESS IS SUPPLIED AS
250          THE SINGLE ARGUMENT.  THUS, AFTER A BREAK HAS BEEN
255          EXECUTED, CØNTRØL MAY BE TRANSFERRED TØ ANY LØCATIØN
260          DESIRED.
265
270  EXAMPLE:     TRANSFER TABEND-TABLEN+3  #
275
280                IF ISSUED AFTER THE INITIAL LØAD, EXECUTIØN ØF THE
285                PRØGRAM CØMMENCES AT (TABEND-TABLEN+3) ; IF ISSUED
290                AFTER CØNTRØL HAS BEEN TRANSFERRED TØ THE CØNSØLE
295                AT A BREAKPØINT, EXECUTIØN ØF THE PRØGRAM RESUMES
300                AT (TABEND-TABLEN+3), RATHER THAN AT THE INSTRUCTIØN
305                AT THE BREAKPØINT, AS WØULD BE THE CASE WITH A
310                "CØNTINUE".
315
320
325
```

REGISTERS <ØPTIØNAL LIST ØF REGISTER IDENTIFIERS> #

THE ARGUMENT LIST, IF SUPPLIED, IS A LIST ØF REGISTER IDENTIFIERS (A, Q, 4, ETC) SEPARATED BY SEMICØLØNS. THE CØNTENTS ØF THE REGISTERS SPECIFIED ARE PRINTED. IF NØ ARGUMENT IS SUPPLIED, THE CØNTENTS ØF ALL REGISTERS [A, Q, E, BAR, IR, TR, IC, XØ, X1, ... , X7] ARE PRINTED. THE INDEX REGISTERS XO, X1, ETC., ARE SPECIFIED IN THE ARGUMENT LIST BY NUMBER ØNLY, NØT AS XN.

EXAMPLES: REGISTERS E:Q:2 #

THE CØNTENTS ØF THE THREE REGISTERS: E, Q, AND X2 ARE PRINTED.

REGISTERS #

THE CØNTENTS ØF ALL REGISTERS ARE PRINTED.

ALTER <REGISTER IDENTIFIER,EXPRESSIØN>; ..... ;
<REGISTER IDENTIFIER,EXPRESSIØN> #

THE CØNTENTS ØF EACH REGISTER SPECIFIED ARE REPLACED BY THE CØRRESPØNDING EXPRESSIØN.

EXAMPLE: ALTER Q,LØWBND;5,037741;IC,TESTP;E,22. #

THE CØNTENTS ØF THE:	
Q REGISTER ARE REPLACED BY:	LØWBND
X5	037741
IC	TESTP
E	22(DECIMAL)

SAVE <PRØGRAM NAME> #

THE PRØGRAM(S) CURRENTLY LØADED ARE SAVED UNDER THE NAME SUPPLIED. ALL BREAKPØINTS, PATCHES, ETC., ARE PRESERVED AS THEY STAND.

EXAMPLE: SAVE TRASH #

/ / / / / ØUTPUT MØDES AVAILABLE \ \ \ \ \

ANY ØF THE FØLLØWING 8 SYMBØLS MAY BE SUPPLIED AS THE ARGUMENT TØ THE "FØRMAT" CØMMAND:

SYM SYMBØLIC MØDE. THE CØNTENTS ØF THE LØCATIØN ARE PRINTED AS A MNEMØNIC ØPERATIØN CØDE FØLLØWED BY A SYMBØLIC ADDRESS AND AN ADDRESS MØDIFICATIØN FIELD. THE ADDRESS WILL BE PRINTED AS (NEAREST SYMBØL<ADDRESS+ØCTAL CØNSTANT) IF NØ SYMBØL ALØNE APPLIES. IF THERE IS NØ MNEMØNIC WHICH CØRRESPØNDS TØ THE ØPCØDE PØRTIØN ØF THE WØRD, THE ØUTPUT MØDE IS CHANGED TØ ØCT FØR THE ØNE WØRD ØNLY, AFTER WHICH IT REVERTS TØ SYM.

DEC DECIMAL MØDE. THE CØNTENTS ØF THE LØCATIØN ARE PRINTED AS A FIXED-PØINT DECIMAL CØNSTANT

ØCT ØCTAL MØDE. THE CØNTENTS ØF THE LØCATIØN ARE PRINTED AS A FIXED-PØINT ØCTAL CØNSTANT.

FLØ FLØATING PØINT MØDE. THE CØNTENTS ØF THE LØCATIØN ARE PRINTED AS A FLØATING PØINT CØNSTANT.

BCD BINARY CØDED DECIMAL MØDE. THE CØNTENTS ØF THE LØCATIØN ARE PRINTED AS 6 ALPHA-NUMERIC CHARACTERS (6 BITS EACH).

ASC ASCII MØDE. THE CØNTENTS ØF THE LØCATIØN ARE PRINTED AS 4 ALPHA-NUMERIC CHARACTERS (9 BITS EACH).

HAL HALF-WØRD MØDE. THE CØNTENTS ØF THE LØCATIØN ARE PRINTED AS TWØ SYMBØLIC ADDRESS EXPRESSIØNS (18 BITS EACH).

TAL TALLY MØDE. THE CØNTENTS ØF THE LØCATIØN ARE PRINTED AS A SYMBØLIC ADDRESS, A TALLY CØUNT, AND 6 MØDIFICATIØN BITS WHICH WILL BE PRINTED AS 2 ØCTAL DIGITS.

IF YØU HAVEN'T FIGURED IT ØUT BY NØW, RETURN TØ GØ AS THERE AINT NØ MØ

**********END MØLDY DDT**********

GMAP--PHASE 0
APRIL 20, 1966

THIS DØCUMENT CØNTAINS A USER DESCRIPTIØN ØF THE GMAP TØ BE PRØVIDED ØN THE PHASE 0 MØLD SYSTEM.

UPØN RECEIVING CØNTRØL GMAP WILL REQUEST INPUT FROM THE TERMINAL, IN ØRDER TØ GET ITS ØPTIØN PARAMETERS (THESE MAY BE TYPED WITH THE "RUN GMAP" CØMMAND IF LISTEN ØR ØTHER INITIATING MØDULES PRESERVE AND TRANSMIT THE REMAINDER ØF THE INPUT LINE). AT LEAST THREE FIELDS, CØMMA SEPARATED, WILL BE EXPECTED. THE FIRST FIELD WILL BE THE NAME ØF THE SØURCE FILE. THE SECØND FIELD WILL BE THE NAME ØF THE BINARY ØUTPUT FILE, UNLESS IT IS "NBIN" IN WHICH CASE NØ BINARY ØUTPUT WILL BE PRØDUCED. THE THIRD FIELD WILL BE THE NAME ØF THE LISTING FILE UNLESS IT IS "NLSTØU" IN WHICH CASE NØ ØUTPUT LISTING WILL BE PRØDUCED. AN ØPTIØNAL FØURTH FIELD MAY CØNTAIN EITHER "SYMTAB" ØR "NSYMTAB". THESE WILL FØRCE ØR INHIBIT RESPECTIVELY THE ØUTPUTING ØF THE SYMBØL TABLE FØR DDT. IF IT IS NØT PRESENT NSYMTAB WILL BE ASSUMED.

SEVERAL FEATURES FØR GENERATING ASCII CØNSTANTS WILL BE PRØVIDED. THE PSEUDØ-ØP "ACI" CØRRESPØNDS TØ THE PSEUDØ-ØP "BCI" BUT PRØDUCES ASCII RATHER THAN BCD CHARACTERS. THE "A" SPECCIFICATIØN IN LITERALS ØR "VFD" PSEUDØ-ØPS ALSØ GENERATES ASCII.

CERTAIN FEATURES ØF GECØS GMAP WILL NØT BE PRØVIDED. BECAUSE ØF THE EXISTENCE ØF A CØMPREHENSIVE EDITING SYSTEM WITHIN THE ØPERATING ENVIRØNMENT THE FILE MAINTENANCE (I.E. ALTER) FUNCTIØNS WILL BE DELETED. SINCE GMAP WILL BE INDEPENDENT ØF THE DEVICE ØN WHICH THE FILE IS MAINTAINED, THE CØMPRESSED DECK FEATURE WILL ØF CØURSE ALSØ BE REMØVED. IN ADDITIØN, SYSTEM MACRØS WILL NØT, AT LEAST AT FIRST, BE PRØVIDED.

INPUT WILL BE FREE FØRMAT, FØLLØWING RØUGHLY TSAP CØNVENTIØNS:

<LABEL>:<ØPCØDE><SPACE><VARIABLE FIELD>"<CØMMENT><; ØR <CR>>

GMAP WILL DEPEND ØN THE FILE SYSTEM FØR CØMMUNICATIØN WITH THE TERMINAL, FØR LØCATING USER FILES, FØR CREATING AND DELETING ITS SCRATCH FILE, AND FØR READING AND WRITING ALL FILES. ALL ØTHER ØPERATIØNS WILL BE INTERNAL. THE PHASE 0 SYSTEM WILL REQUIRE AN INITIAL MEMØRY ALLØCATIØN AND WILL NEITHER REQUEST NØR RELEASE MEMØRY UNTIL TERMINATIØN.

```
ØLD
ØLD PRØBLEM NAME--LDINIT

READY.

LIST


LDINIT    14:00   JUNE 22,1966

0   6/21/66  (RPL)
100 INITIAL VERSIØN ØF THE ØLD LØADER
110
120
130 THE FIRST VERSIØN ØF THE ØLD LØADER WILL BE MUCH SIMPLER THAN
140 THAT DESCRIBED IN ØLØAD1 AND ØLØAD2.  THE CØMMANDS WILL BE
150 SIMILAR BUT SIMPLER.  THERE ARE THREE BASIC CØMMANDS:
160
170    LØAD NAME,PASSWØRD
180
185    START
186
190    ØPTIØN XXX;YYY;ZZZ
200      XXX,YYY, AND ZZZ ARE ØPTIØNS FRØM THE LIST THAT FØLLØWS.
210
220  THE ØPTIØNS ARE:
230
240    MAP  THIS WILL CAUSE A MEMØRY MAP TØ BE PRINTED ØN THE TELETYPE.
250
260    CØMMØN N  THIS WILL CAUSE A BLØCK ØF N WØRDS TØ BE RESERVED AS
270              CØMMØN AT THE BEGINNING ØF THE PRØGRAM.  THIS MUST
275            ØN THE TELETYPE.
280              APPEAR BEFØRE THE FIRST PRØGRAM IS LØADED IF IT IS
290              TØ BE USED.
300
310    USE NAME (N)  THIS WILL CAUSE THE SYMBØL NAME TØ BE ENTERED INTØ
320              THE LØAD TABLE AS A LABELED CØMMØN REGIØN ØF SIZE N.  A
330            BLØCK ØF N WØRDS WILL BE RESERVED AT THIS PØINT IN THE
340            PRØGRAM FØR THE LABELED CØMMØN REGIØN.
350
360   BEGIN SYMDF   THIS WILL CAUSE THE LØCATIØN ØF THE SYMBLØ SYMDF
370            TØ BE TAKEN AS THE TRANSFER ADDRESS INTØ THE PRØGRAM.  THE
380          SYMBØL SYMDF MUST BE DEFINED AS A SYMDEF IN ØNE ØF THE
390          PRØCEDURES BEING LØADED.
400
410   ØRIGIN N  THIS WILL CAUSE THE PRØGRAM TØ BE LØADED WITH A BASE
415            ADDRESS ØF N (ALL PRØGRAMS WILL BE LØADED AS SLAVE
420            PRØGRAMS).  THE FIRST PRØCEDURE WILL BE LØADED AT SLAVE
430            ADDRESS 64, ØR ABSØLUTE ADDRESS N+64.  IF N IS NØT
440            SPECIFIED THE BASE ADDRESS WILL BE AT 1024.
450
460 TØ LØAD A PRØGRAM YØU MUST FIRST SPECIFY THE ØPTIØNS YØU WANT TØ
470 USE BY THE ØPTIØN CØMMAND.  IF YØU DØ NØT NEED ANY FINE.
480 YØU THEN USE THE LØAD CØMMAND.  AFTER THE PRØGRAM HAS BEEN
490 LØADED YØU CAN EITHER RUN IT BY USING THE START CØMMAND ØR LØAD
500 ANØTHER PRØGRAM WITH THE ØPTIØN AND LØAD CØMMANDS.
510
```

```
OLD
OLD PROBLEM NAME--MLOADS

READY.

;←LIS

MLOADS    14:02    JUNE 22,1966

100
110
120
130
140
150    * * * * *       SUMMARY OF COMMANDS FOR MOLDY LOADER      * * * * *
160
170
180
190   * * * LOAD  F;F; ..... ;F
200
210   * * * * * OPTION  O;O; ..... ;O
220
230   * * * * * * * START
240
250   * * * * * * * * * DUMP  F
260
270   * * * * * * * * * * * EXIT
280
290
300
310
320
330                     NOTATION
340
350
360                F:   FILE IDENTIFIER (OF FORM:  <FILENAME,PASSWORD> )
370
380                O:   OPTION
390
400
410
```

```
ØLD
ØLD PRØBLEM NAME--MLØAD1

READY.

LIST


MLØAD1     14:06   JUNE 22,1966

0   6/20/66  (RPL)
100 MØLD LØADER
105
110
115 THE MØLD LØADER WILL BE A MØDIFICATIØN ØF GELØAD, THE LARGEST
120 CHANGE WILL BE THE ELIMINATIØN ØF THE DEBUG FEATURES AND THE
125 ADDITIØN ØF CERTAIN ØPTIØNS TØ CREATE A FILE WHICH IS A CØRE
130 IMAGE ØF SØME PRØGRAM. MØST ØF THE ØTHER GELØAD ØPTIØNS WILL
135 STILL BE AVAILABLE.
140
145
150 LØADER CØMMANDS
155
160    LØAD N1,P1;N2,P2;...;NN,PN
165
170      THIS IS THE CØMMAND TØ LØAD WITH STANDARD ØPTIØNS.  THE
175      N'S ARE FILE NAMES AND THE P'S ARE PASSWØRDS.  THE FILE
180      NAMES AND PASSWØRDS ARE SEPARATED BY CØMMAS AND DIFFERENT
185      FILES ARE SEPARATED BY SEMI-CØLØNS.  PASSWØRDS ARE
190      ØPTIØNAL.  IF THE FILE DØES NØT NEED ØNE USE A SEMI-CØLØN
191      AFTER THE FILE NAME IF MØRE FILES ARE TØ FØLLØW.
195
200    ØPTIØN  XXX;YYY;ZZZ
205
210      IF IT IS NECESSARY TØ USE SØME ØPTIØNS ØTHER THAN THE
215      STANDARD ØNE'S THEY MUST BE SPECIFIED BEFØRE THE LØAD
220      CØMMAND.  THE NEXT LØAD CØMMAND WILL THEN BE EXECUTED WITH
225      SPECIFIED ØPTIØNS.  THE ØPTIØNS SHØULD BE SEPARATED BY
230      SEMI-CØLØNS SINCE SØME ØF THEM WILL HAVE IMBEDDED CØMMAS.
235      A LIST ØF ØPTIØNS FØLLØWS WITH THE STANDARD ØPTIØNS SPECIFIED.
240
245    1. MEMØRY MAP ØPTIØNS
250
255      NØMAP  NØ MEMØRY MAP IS PRØDUCED.  THIS IS STANDARD.
260      MAP XXX  PRØDUCE A MEMØRY MAP AND PLACE IT IN FILE XXX.
261               THIS FILE (EXCEPT FØR TTY) MUST BE ØNE THAT DØES
262               NØT ALREADY EXIST IN THE USERS CATALØG.
265
```

2. EXECUTE OPTIONS

NOGO DOES NOT EXECUTE AFTER LOADING. THIS IS STANDARD. THE LOADER WILL WAIT FOR ANOTHER COMMAND. THIS COULD BE A GO OR ANOTHER LOAD.

CONGO EXECUTES THE JOB UNLESS A FATAL ERROR OCCURS OR THE ERROR COUNT EXCEEDS THAT SET BY ERCNT. DETECTED DURING LOADING

GO EXECUTES THE JOB ONLY IF NO ERRORS OCCURRED DURING LOADING.

3. SET MEMORY OPTIONS

SET N SETS ALLOCATED MEMORY TO OCTAL PATTERN SPECIFIED BY N (MEMORY IS NORMALLY SET TO ZERO).

4. SET MAXIMUM ERROR COUNT

ERCNT N SETS A LIMIT (N) ON THE NUMBER OF FATAL AND NONFATAL ERROR MESSAGES WHICH MAY BE PRINTED BEFORE LOADING IS ABORTED. THIS COUNT IS NORMALLY SET AT 150.

5. SYMREF AND SYMDEF OPTIONS

NOSREF NO SYMREF'S ARE PRINTED. THIS IS STANDARD
SYMREF CAUSES ALL SYMREF'S FOR A LOADED ROUTINE TO BE PRINTED WITH THE MEMORY MAP.
NOSDEF NO SYMDEF'S WILL BE PRINTED. THIS IS STANDARD
SYMDEF SYMDEF'S WILL BE PRINTED WITH THE MEMORY MAP.

6. LOW COMMON OPTION

LOCOMN CAUSES ALL LABELED COMMON TO BE ASSIGNED BELOW BLANK COMMON.

7. SETUP OPTION

?

8. FILE CONTROL BLOCK OPTIONS

?

9. SYMBOL TABLE ADDITIONS

USE NAME/SIZE/,NAME1,NAME2
THIS PERMITS THE USER TO INSTRUCT THE LOADER TO ENTER THE VARIABLE NAME INTO ITS SYMBOL TABLE AS LABELED COMMON REGIONS OR SYMREF'S. SIZE IS GIVEN IF THE VARIABLE IS TO BE USED AS A LABELED COMMON REGION AND REPRESENTS THE AMOUNT OF STORAGE TO BE SET ASIDE AT THAT POINT OF LOADING. IF SIZE IS NOT GIVEN NAME IS CONSIDERED AS A SYMREF. IF SIZE IS TERMINATED BY THE CHARACTER L (I.E. /200L/), THE LABELED COMMON REGION NAME IS HANDLED AS IF UNDER THE LOCOMN OPTION. ALL OTHER LABELED COMMON REGIONS ARE HANDLED NORMALLY.

```
555         THIS PERMITS LØCATIØNS ØF NEW SYMDEF'S TØ BE DEFINED BY
560         SYMDEF'S PREVIØUSLY DEFINED, DEFINING NEW SYMDEF'S
565         RELATIVE TØ PREVIØUSLY DEFINED SYMDEF'S, AND EQUATING
570         LABELED CØMMØN REGIØNS RELATIVE TØ BLANK CØMMØN.
575         DIFFERENT EQUATES ARE SEPARATED BY CØMMAS.
580
585         EX1 EQUATE NAME1/NAME2,NAME3/,NAME4/NAME5/
590         EX2 EQUATE NAME(10)/NAMED/
595         EX3 EQUATE .CMN./LC1/,.CMN.(100)/LC2/
600
605         EX1  DEFINES NAME2 AND NAME3 AS SYMDEF'S WITH THE EQUIVALENT
610              LØCATIØN ØF NAME1.  A FATAL ERRØR RESULTS IF NAME1 IS
615              UNDEFINED.  IF NAME2 ØR NAME3 HAVE BEEN PREVIØUSLY
620              DEFINED, THEY ARE REDEFINED, AND A NØNFATAL ERRØR
625              MESSAGE IS PRINTED.  NAME5 SYMDEF IS DEFINED AS HAVING
630              LØCATIØN EQUIVALENT TØ THAT ØF NAME4.
635
640         EX2  A NEW SYMDEF MAY BE DEFINED RELATIVE TØ A PREVIØUSLY
645              DEFINED SYMDEF BY ENCLØSING THE INCREMENT IN PAREN-
650              THESES.  NAMED IS DEFINED AS THE LØCATIØN NAME+10.
655
660         EX3  .CMN. IS A STANDARD SYMBØL WHICH IS SYNØNYMØUS WITH
665              THE BEGINNING ØF BLANK CØMMØN.  THE LABELED CØMMØN
670              REGIØNS (LC1 AND LC2) ARE EQUATED TØ THE RESPECTIVE
675              PØSITIØNS ØF BLANK CØNMØN.  LABELED CØMMØN REGIØN
680              LC2 IS ASSIGNED AN ADDRESS EQUAL TØ THE BEGINNING ØF
685              BLANK CØMMØN PLUS 100 ØCTAL.  THE LENGTH ØF BLANK
690              CØMMØN IS ADJUSTED ACCØRDINGLY.
695
700 CØNTINUED IN ML0AD2
```

```
LIS


MLØAD2    14:06   JUNE 22,1966

0   6/20/66  (RPL)
100 MØLD LØADER (CØNTINUED)
105
110
115     10. MEMØRY PØSITIØN
120
125       NØRMAL LØADING PRØCEDURE WILL BE THE LØW LØAD ØPTIØN
130       DESCRIBED IN THE GELØAD MANUAL. IN ØRDER TØ SAVE SPACE
135       FØR BLANK CØMMØN IT IS NECESSARY TØ TELL THE LØADER
140       HØW FAR ABØVE THE BASE ADDRESS IT SHØULD PLACE THE
145       FIRST PRØGRAM.
150
155       CØMMØN N  CAUSES THE LØADER TØ SKIP N LØCATIØNS BEFØRE
160                 LØADING THE FIRST PRØGRAM.
165
170       HLØAD  WILL CAUSE PRØGRAMS TØ BE LØADED STARTING AT THE
175              UPPER END ØF ALLØCATED MEMØRY.
180
185     11. LIBRARY ÚSE
190
195       THE USER IS ALLØWED TØ SPECIFY CERTAIN FILES WHICH
200       CØNTAIN LIBRARY SUBRØUTINES.  IF ALL ØF THE SYMREFS HAVE
205       NØT BEEN DEFINED AT THE END ØF LØADING THE USERS PRØGRAMS
210       THE LØADER WILL SEARCH THE LIBRARY FILES THE USER SPECIFIES
215       SEARCHING FØR SUBRØUTINES THAT HAVE PRIMARY SYMDEF SYMBØLS
220       IDENTICAL TØ ANY ØF THE UNDEFINED SYMREF SYMBØLS IN ITS
225       SYMBØL TABLE.  THE FILES WILL BE SEARCHED IN THE ØRDER GIVEN.
230
235       LIB N1,P1;...;NN,PN
240
245       THE N'S ARE NAMES AND THE P'S ARE PASSWØRDS.
250
255     12. ENTRY LØCATIØNS
260
265       ENTER NAME  SYMDEF NAME IS THE DESIRED ENTRY PØINT TØ THE
270                   PRØGRAM.  IF THIS ØPTIØN IS NØT PRESENT ENTRY
275                   IS MADE AT THE FIRST DEFINED PRIMARY SYMDEF
280                   ØF THE FIRST PRØGRAM LØADED.
285
286     START  CØNTRØL WILL BE TRANSFERED TØ THE PRØGRAM JUST LØADED
287            (SEE ENTRY LØACATIØN ØPTIØNS).
288
290     DUMP XXX,PPP
295
300       THIS CØMMAND WILL DUMP A CØRE IMAGE ØF A PRØGRAM THE USER
305       HAS JUST LØADED AND PLACE IT IN FILE XXX.  FILE XXX MUST
310       NØT ALREADY EXIST.  PPP IS A PASSWØRD TØ PUT ØN THE
320       FILE.
330
340     EXIT
350
360       THIS CØMMAND WILL CAUSE THE LØADER TØ EXIT BACK TØ THE
370       EXECUTIVE.
380
```

```
LIST

MCHAR      10:01   JULY 1,1966

100    MØLD CHARACTER SET
110
120
122                MØLD CHARACTER SET
130
200       000      NULL
210       001
220       002
230       003
240       004      EØT
250       005      ENQ
260       006      ACK
270       007      BELL
280       010      BS
290       011      HT
300       012      LF
310       013      VT
320       014      FF
330       015      CR
340       016      SØ
350       017      SI
360       020      DLE
370       021      DC1
380       022      DC2
390       023      DC3
400       024      DC4
410       025      NAK
420       026
430       027
440       030      CAN
450       031
455       032
460       033      ESC
470       034
480       035
```

500	037	
510	040	SPACE
520	041	EXCLAMATIØN PT
530	042	"
540	043	#
550	044	$
560	045	PER CENT
570	046	AMPERSAND
580	047	APØSTRØPHE
590	050	(
600	051	)
620	052	*
630	053	+
640	054	,
650	055	-
660	056	.
670	057	/
680	060	0
690	061	1
700	062	2
710	063	3
720	064	4
730	065	5
740	066	6
750	067	7
760	070	8
770	071	9
780	072	:
790	073	;
800	074	<
810	075	=
820	076	>
830	077	?
840	100	GRAVE
850	101	A
860	102	B
870	103	C
872	104	D
876	105	E
878	106	F
880	107	G
882	110	H
884	111	I
886	112	J
888	113	K
890	114	L
892	115	M
894	116	N

898	120	P
900	121	Q
910	122	R
920	123	S
922	124	T
924	125	U
926	126	V
928	127	W
929	130	X
930	131	Y
932	132	Z
934	133	[
936	134	TILDE
938	135	]
940	136	CIRCUMFLEX
942	137	UNDERSCORE
944	140	AT SIGN
946	141	A (LOWER CASE)
947	142	B "
948	143	C "
949	144	D "
950	145	E "
951	146	F "
952	147	G "
954	150	H "
956	151	I "
957	152	J "
958	153	K "
959	154	L "
960	155	M "
961	156	N "
962	157	O "
963	160	P "
964	161	Q "
965	162	R "
966	163	S "
967	164	T "
968	165	U "
969	166	V "
970	167	W "
971	170	X "
972	171	Y "
973	172	Z "
974	173	LEFT BRACE
975	174	OVERSCORE
976	175	RIGHT BRACE
977	176	VERTICAL LINE
978	177	DELETE

```
ØLD
ØLD PRØBLEM NAME--MISC

READY.

LIST


MISC      13:30   JUNE 30,1966

100  JUNE 27, '66 (KML)
110
120  MISCELLANEØUS SYSTEM CØNVENTIØNS
130
140    FILE NAMES
150
160              AS FAR AS THE EXECUTIVE IS CØNCERNED FILE NAMES ARE
170              CØMPØSED ØF 2 36-BIT WØRDS WHICH MAY HAVE ANY PATTERN
180              ØF 0'S AND 1'S.  BUT EDITØR, DDT, GMAP, AND
190              THE LØADER WILL BE WØRKING PRIMARILY WITH ASC II
200              SØ THE FØLLØWING CØNVENTIØN WILL BE ADHERED TØ BY
210              THØSE 4 SYSTEMS:
220
230              NAMES WILL BE 8 CHARACTERS LØNG.  IF THE NAME SUPPLIED
240              IS LESS THAN 8 CHARACTERS A SUFFICIENT NUMBER ØF
250              BLANKS WILL BE ADDED (ØN THE RIGHT) TØ MAKE THE
260              STRING 8 CHARACTERS IN LENGTH.  IN ØTHER WØRDS
270              NAMES WILL BE LEFT JUSTIFIED AND BLANK FILLED.
272              ALSØ THE NAME MUST NØT HAVE LEADING BLANKS ØR
274              ØNE ØF THE FØLLØWING CHARACTERS IMBEDDED IN THE
276              NAME: "," , ";" ØR ":".
280
290
300    PASSWØRDS
310
320              IN THE INITIAL VERSIØN ØF MØLD, PASSWØRDS WILL
330              BE ALLØTTED ØNLY 18 BITS BUT EDITØR, DDT
340              ETC. WILL ALLØW UP TØ 8 CHARACTERS SØ SØME MAPPING
350              MUST BE APPLIED TØ THE 8 CHARACTERS TØ PRØDUCE THE
360              PASSWØRD.  THE FØLLØWING MAPPING WILL BE USED:
370
380              ASSUMING THE PASSWØRD IS CØMPØSED ØF THE 8 CHARACTERS
390              C1, C2, ... , C8 THEN THE ALGØRITHM IS
400              ((((((C1*N+C2)*N+ C3)*N+ C4)*N+ C4)*N+ C5*N ... + C8)
410              WHERE N IS EQUAL TØ 631463 BASE 8.  IN ØRDER
411              TØ GET AN 18 BIT RESULT USE THE LØW ØRDER 18
412              BITS ØF THE RESULT.
414              IF NØ PASSWØRD IS SUPPLIED THEN THE VALUE 0 IS
416              SUPPLIED AUTØMATICALLY BY EVERY SYSTEM.
420
```

CHARACTER FILES

SINCE THE SMALLEST UNIT THE FILE SYSTEM IS CAPABLE ØF HANDLING IS A WØRD, ALL CHARACTER FILES MUST CØNSIST ØF 4*N CHARACTERS(ASSUMING THERE ARE N WØRDS IN THE FILE). THERE IS NØ END ØF FILE CHARACTER ØN THE SYSTEMS LEVEL. THEREFØRE CHARACTER FILES MUST BE AUGMENTED TØ A MULTIPLE ØF 4 CHARACTERS. THE DELETE CHARACTER(ØCTAL 177) SHØULD BE USED AS THE FILL CHARACTER WHEN THE FILE DØES NØT TURN ØUT TØ HAVE EXACTLY 4*N CHARACTERS. THE FILE CHARACTER WILL ALSØ HAVE TØ BE USED WHEN WRITING N WØRDS ØN A REMØTE CØNSØLE.

TELETYPE I/Ø

INPUT MØDES

1 LINE BY LINE CØNVERSATIØNAL

2 FILE BUILDING(TWØ CARRIAGE RETURNS IN A RØW CAUSES A RETURN TØ LINE BY LINE CØNVERSATIØNAL.

EVENTUALLY SYSTEMS WILL BE ABLE TØ SELECT INPUT SET MØDE EXECUTIVE EXECUTIVE CALL. EVENTUALLY THERE WILL BE A SPECIAL READ CØMMAND WHICH WILL BE USED FØR UNSØLICITED INPUT. UNSØLICITED INPUT IS DEFINED AS ANY INPUT LINE(ØR LINES) WHICH CØMES IN WHEN NØ READ CØMMAND IS ØUTSTANDING FØR THE FILE(REMØTE DEVICE). THE UNSØLICITED INPUT WILL BE GIVEN TØ THE FIRST PRØGRAM IN THE SPAWN TREE WITH A SPECIAL READ ØUTSTANDING. THIS WILL ALLØW THE PRØGRAM TØ BE INTERUPTED WITHØUT BEING ABØRTED.

PANIC

THE BREAK CHARACTER WILL ALWAYS RESULT IN A MØVE ØF ØNE LEVEL UPWARD IN THE SPAWN TREE(SPAWNING CAUSES MØVEMENT DØWN A LEVEL). NØRMALLY THIS MEANS THE CURRENT JØB IS ABØRTED AND THE EXECUTIVE RETURNS CØNTRØL TØ THE ANCESTØRAL JØB.

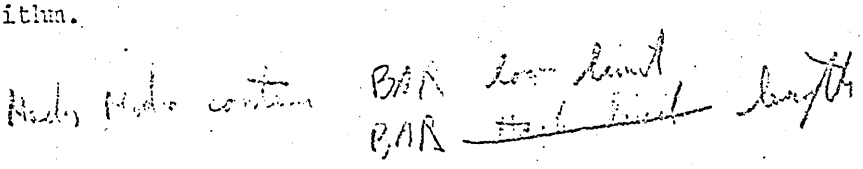
```
ØLD
ØLD PRØBLEM NAME--FAULTV

READY.

LIST


FAULTV    13:15    JUNE 30, 1966

100  JUN 27, 1966 (KML)
110
120
140  EACH SLAVE PRØGRAM TØ BE RUN IN THE MØLDY ENVIRØNMENT
150  MUST DEDICATE 32 WØRDS (LØCATIØNS 0-31) TØ A FAULT VECTØR.
170  IF A FAULT ØCCURS WHICH IS THE SLAVE PRØGRAMS RESPØNSIBILITY
180  THEN THE EXECUTIVE WILL CARRY ØUT THE FØLLØWING:
190      1. SET THE BAR TØ THE MAXIMUM ALLØWABLE LIMITS,
200      2. EXECUTE THE FØLLØWING INSTRUCTIØNS(ØR THE EQUIVALENT)
210           STC1 2*I+B
220           TSS  2*I+1+B  WHERE 0,=I<16 IS THE NUMBER ASSIGNED TØ
230          THE FAULT WHICH ØCCURRED AND B IS THE ABSØLUTE LØCATIØN
232          ØF SLAVE ADDRESS 0.
240
270  IF THE JØB IS NØT USING THE CØNTRØL PRØGRAM ØPTIØN THEN THE
280  FAULT RØUTINE CAN RETURN TØ THE PØINT ØF INTERUPTIØN BY
290  EXECUTING A  RET 2*I.
300
310  WARNING:
320
330           IF THE SAME FAULT ØCCURS IN THE FAULT HANDLING RØUTINE
340           YØU WILL NØT BE ABLE TØ RETURN TØ THE ØRIGINAL PØINT
350           ØF INTERUPTIØN UNLESS THE INSTRUCTIØN CØUNTER (CØNTENTS ØF
360           LØCATIØN 2*I) HAS BEEN PUSHED ØNTØ A STACK.
370
380
390
400           NØTE THAT FAULT 0 IS USED FØR JØB START UP.
410           THAT IS THE EXECUTIVE WILL TRANSFER TØ SLAVE LØCATIØN
420           ØNE WHEN STARTING UP A JØB CALLED THRØUGH THE
422           SPAWN CØNMAND.
430
```

FAULTS AND THEIR SIGNIFICANCE

0	SHUTDØWN	WILL NEVER BE USED EXECPT AS THE STARTUP ØF THE JØB WHEN THE EXECUTIVE WILL TRANSFER CØNTRØL TØ LØCATIØN 1.
1	MEMØRY FAULT	PRØBABLY AN ILLEGAL ADDRESS HAS BEEN USED BY THE PRØGRAM.
2	MME	THIS IS RETURNED TØ THE USER PRØGRAM IF THE LØWER LIMIT ØF THE BAR IS NØT EQUAL TØ THE LØWEST ØRDER MEMØRY LØCATIØN ASSIGNED TØ THE JØB ... THAT IS THE CØNTRØL ØPTIØN ALLØWS THE CØNTRØL PRØGRAM TØ PRE-EMPT MME'S.
3	FAULT TAG	
4	TIMER RUNØUT	THIS FAULT ØCCURS WHEN THE PSEUDØ TIMER REGISTER RUNS DØWN.
5	CØMMAND FAULT	
6	DERAIL	
7	LØCKUP	
8	CØNNECT	THIS ENTRY MAY EVENTUALLY BE USED TØ INDICATE THAT A REMØTE CØNSOLE(FILE) HAS ISSUED A CØNNECT.
9	PARITY	
10	ILLEGAL ØPCØDE	ØPCØDE ØF ZERØ.
11	ØPERATIØN NØT CØMPLETE	
12	STARTUP	
13	ØVERFLØW	
14	DIVIDE CHECK	
15	EXECUTE	

Structure of the job stack

At the bottom of the stack is fixed format information that never varies from job to job. This contains the account number for billing and pointers used to reference file buffers and pointers. It also contains the saved instruction counter plus indicators and the saved registers. This is where the registers and indicators are always saved whenever the program is not currently being executed. On top of the fixed locations are the buffers used by the file system. These are of variable length so the area must be allocated dynamically. In the first implementation the buffers will all be the same size (the maximum) to simplify the design of the file system. There will also be information as to device address, amount of buffer filled, link information, and interrupt location, and data control word storage. If a user calls the exec with a file call that would exceed the limits of his state vector, he is immediately swapped out with the IC decremented by one and the state vector incremented by 1/2 K. When he is swapped back in, the UME will then be executed again and the file call will then be processed. All of this operation is hidden from the user. Memory allocation is done by the trivial scheme of assigning the next available memory to the next program. When this runs out of memory because the next program won't fit, the algorithm waits until the lowest portion of memory is free and the process starts over again. The scheduling algorithm is done by reference to 2 tables, the run flag table and the priority table. One pass is made through the two tables every quantum and the tables are updated. The program with the highest priority that is ready to run is run. If there is no program ready to run, the system waits until one is ready. The scheduling algorithm operates on the principle that if there are n users, each user is entitled to 1/n th of the available machine time. Therefore, a user who does a lot of real time input with very little calculating will get fast response time while a user who requires a great deal of machine time will not delay the system for the other users but will still get his share of the computer. The algorithm is designed so that the priority of a program increases exponentially to one when the program is not being run and decreases exponentially to zero when it is being run. Therefore, the priority represents a weighted average of the past usage of the computer by the program and can be used as an indication of program priority by the scheduling algorithm.



Executive functions

The executive is the section of code that is permanently in core and performs functions such as scheduling and memory allocation. These functions that enable time sharing to function are invisible to the user program and the user can write his routines with the assumption that he is the only one using the machine. Of course, the user must adhere to certain conventions or he cannot take full advantage of the system.. All I/O must be done through the file system. No I/O can refer to a specific device or use a specific device address. The available calls on the file system are given on another page. Other calls are:

TIME — gives the running time in 15.625 usec intervals

CLOCK — gives the time of day in the AQ in edited BCD format

SWAP UNTIL CONDITION — causes the program to resume execution only when the condition specified is satisfied

SET BAR — causes the BAR to be set inside the user program [used for debugging]

SET TIMER — causes a timer runout fault at the end of the given execution time

SPAWN — causes the execution of a spawned process. communication between processes is by interrupts and file operations. a file can also be set up so a write or a read refers to the other process? opened files can also be transfered to the spawned process to use as its own.

TERMINATE — causes immediate termination of the process and the incremental dumping of files closed[?]and the reassignment of special files [teletypes]

note: whenever the user sets the base address register to a value that does not include the fault locations, he immediately gets control of any fault with the base address register reset to the original value and he also gets control of MME faults. It is therefore impossible for the program to get out of control by making calls on the executive. The only faults that the user cannot normally get control of are: shut down, connect, parity, startup, execute. It is to be noticed that the user cannot ignore interrupts with a RET *-1 because the stored IC is with respect to the another base address. The SET BAR call gives the user the option of transfering to any slave location with any machine registers and indicators. Thus a program can be restarted. This feature is mainly used with debugging. The master file directory is keep in memory at all times in [read only storage] with a copy on all random access storage devices. Areas used in swapping are in the master file directory to minimize the time to find a swap address. All other catalogs are on high speed storage. All catalogs with high anticipated usage are duplicated to guard against errors. If a teletype connected to a user program dissappears, this condition is reflected to the user program and all programs related to it. All programs for which the teletype user could be billed must terminate in a short time (they should preserve themselves for restart) or they will be aborted by the executive. This is to prevent accidental clearing out of a teletype from running up a large bill.

Format of a catalog

Each catalog entry contains the following information:

1 name of file or catalog
2 restriction bits (read permit, temporary file etc.)[file or catalog]
3 name of trap file or password
4 date of last modification (or incremental dump tape number or date)
5 statistical information (usage etc.)
6 device address

Links have a different format but since they will not be implemented soon no detailed format information is given.

There is a bit that will prevent the coded date and statistical information from being updated so that often used catalogs will not have to constantly be rewritten back onto the drum or disk. This bit cannot be set by the system? but is initially placed on the master file directory and all other directories that are often used and never dumped because of inactivity.

Status returns of the file system

Unless the call to the file system is in error (wrong format etc.) the file system will return immediately without any information except that the call was accepted. When the file system completes its task, it will interrupt the user program. The status returns of the completed operation cannot be transmitted through the registers because they will contain the user's registers at the time of the interrupt. The status returns cannot be stored in a standard location because of the possibility of simultaneous interrupts. the file system will therefore store the status return of the completed operation by reference to a standard pointer word. The file system will store the status return in the slave location specified by the pointer word. It will then refer to the pointer word with a NOP with an AD modification. Thus the status returns will be stored in a list. If the delta specified in the pointer is zero, no incrementing will take place and all status returns will be stored in the same location. The user program can either scan the table for the new status returns or he can, at the start of each interrupt routine, pick up the status returns with an SD modification. If the user [illegible] will always pick up the status return that caused the interrupt. Think about this for awhile to be convinced that this is true. It is based mainly on the fact that the last interrupt to take place is the first one initiated. Interrupts do not have to be inhibited before the status return is loaded. It is not known yet whether the status return will take up one word or two but user programs should assume two status returns words. These statuses will inform the user whether any unusual conditions were detected on the previous command. It will also contain the information as to the number of words read if the operation was terminated by an end of record or an end of file. Example of a file copying routine:

All files are assumed to be opened.
This routine copies file 1 into file 2.
Error detection and end of file handling is omitted.

File 1 interrupt routine:

```
          TRA   int1
int1:     SREG  reg1
          LDAQ  status,SD
          TMI   endchk
          WRITE 2,i2,m,n        i2 is the 2nd interrupt location
          LREG  reg1
          RET   i1
```

File 2 interrupt routine:

```
          TRA   int2
int2:     SREG  reg2
          LDAQ  status,SD
          TMI   check
          READ  1,i1,m,n
          LREG  reg2
          RET   i2
```

This interrupt routine could be buffered with little additional effort. Notice that the status return is loaded once as a part of both interrupt routines. In this example only one file operation is in progress at any given instant but the presence of other file activities would not affect the operation of these routines. Even with the simultaneous operation of several file operations, it is unnecessary (and useless) to set the inhibit bit on in the interrupt routines. Several interrupts may take place simultaneously and the user must exit the interrupt routine normally (RET) or interrupts will be lost.

Trapping the user and status returns

Whenever the I/O interrupt routine is being executed, all programs currently in memory have all of their registers and indicators stored in a standard place in the program header. When the I/O interrupt routine wants to trap a user program, it loads the saved instruction counter and indicators for the program and stores it in the location trap. It then stores the location of trap+1 in the saved instruction counter location. The effect of this is that when the program is restored it will execute the instruction in location trap+1 with all indicators and registers unchanged. The program can ignore the interrupt by placing a RET *-1 in trap+1. Under no circumstances should a user specify the same trap location for more than one operation. If he does, there is a chance that the program may never leave the interrupt routine as several simulated traps may occur at the same time. If two traps occur at nearly the same time, the second user interrupt routine will be executed first. The word that is stored in location trap contains the instruction counter and the indicators. It ~~also contains in bits 29-35 the status returns of the operation just completed~~ in the status return word. The possible returns are:

Normal - no unusual conditions, operation successfully completed
End of record - an end of record was detected on the last operation. must issue another read command to continue reading the file.
End of file - an end of file was detected on the last operation. cannot continue
Error - unrecoverable error. an attempt will be made to restore the file from the dump tape but you lose for now
Protection violation - incorrect password or priveleged command
File not open - operation requires file specified to be open for read/write etc.
Duplicate name - attempt to create two files of the same name

On a read, the user should wait to be interrupted before attempting to use any of the data read in. On a write, the user must not change the data until he has been interrupted. A copy command can be stopped at any time [somehow]. A user should never execute a DIS instruction. If he must wait for the completion of a file operation he should execute a call "SWAP UNTIL CONDITION" and specify the conditions that must be met before being continuing. The format of this call are unknown but suggestions are welcome.

File System Calls

OPEN FOR READING i, trap, name, password

Search catalog i for name and verify password if any. When complete, trap user to location trap.

OPEN FOR APPENDING i, trap, name, password

Search catalog i for name and verify password if any. When complete, trap user to location trap.

READ i, trap, m, n

Read the next n words into memory location m from file i and trap user to location trap when done.

RESTORE i,trap

WRITE i, trap, m, n

Append the next n words from memory location m to file i and trap user to location trap when done.

CLOSE i, trap

Close file i then trap user to location trap.

CREATE i, trap, name, restrictions

Create an entry of name in catalog i with restrictions. Open the file for writing and trap user to location trap.

DESTROY i,trap, name, password

Delete name in catalog i if password is correct. Then trap user to location trap.

CHANGE i, trap, name, password, newname, restrictions

Search catalog i for name and verify password if any. Then change name to newname and replace old restrictions by restrictions.

COPY i,j,n,trap

Copy file i into file j but not more than n words. Then trap user to location trap. This command is designed for use with teletypes by allowing the teletypes to input directly into a file without explicit read and write commands. It will terminate on n words or on an end of record or end of file.

THE FILE SYSTEM IS A COLLECTION OF EXECUTIVE SUBROUTINES THAT DEAL WITH [illegible] PERIPHERALS. ANY USER PROGRAM THAT WANTS TO USE ANY I/O DEVICE MUST USE THE FILE SYSTEM. NO USER PROGRAM IS EVER PERMITTED TO USE ABSOLUTE DEVICE ADDRESSES. THERE ARE TWO TYPES OF I/O DEVICES: PASSIVE AND ACTIVE. A PASSIVE DEVICE IS ONE IN WHICH INFORMATION IS TRANSMITTED TO IT ONLY BY THE APPROPRIATE USER COMMAND. AN ACTIVE DEVICE IS ONE IN WHICH ANY INFORMATION TRANSMITTED TO IT IS LOST AND ANY INFORMATION OBTAINED FROM IT IS NOT UNDER CONTROL OF THE SOD. EXAMPLES OF EACH OF THE TWO TYPES ARE:

ACTIVE	PASSIVE
D-SD [DISKTYPE]	DRUM
TYPEWRITER	DESK
CARD?	RACE
TAPE	TAPE

THE DISTINCTION BETWEEN THE TWO TYPES OF DEVICES IS NOT CLEAR BUT THE TYPES OF READS AND WRITES ON THE TWO TYPES OF DEVICES ARE QUITE DIFFERENT. THE TYPES OF OPERATIONS PERMITTED WITH A PASSIVE DEVICE ARE:

1. OPEN
2. READ
3. WRITE
4. CLOSE

THE TYPES OF OPERATIONS PERMITTED WITH AN ACTIVE DEVICE ARE:

1. READ
2. WRITE
3. RETURN

NOTICE THAT AN ACTIVE DEVICE CANNOT BE OPENED OR CLOSED. ALL SUCH DEVICES MUST BE OPEN AT THE START OF THE RUN BY A HIGHER LEVEL PROGRAM AND MUST BE RETURNED TO THE HIGHER LEVEL PROGRAM WHEN THE OPERATION IS FINISHED. ACTIVE DEVICES CAN CAUSE USER PROGRAM INTERRUPTS WHEN THEY SPONTANEOUSLY CAUSE INPUT. THESE INTERRUPTS CAN BE MASKED OFF BY THE USER PROGRAM IF DESIRED.

CONVENTIONS FOR USER PROGRAM CALLS TO THE EXECUTIVE

THE ONLY CALL PERMITTED BY A USER PROGRAM TO THE EXECUTIVE IS THE STANDARD MME CALL. ALL SUCH CALLS SHOULD BE DONE THROUGH A PREVIOUSLY DEFINED MACRO BECAUSE THE DETAILS OF THE VARIOUS CALLS ARE SUBJECT TO CONSIDERABLE CHANGE. ALL CALLS ARE OF THE FOLLOWING FORMAT:

```
      LRES ARG          PLACE ARGUMENTS IN A,Q,XR0,XR2-7
      LDX1 MODULE,DU    PLACE THE MODULE NUMBER IN XR1
      MME  0            ENTER MASTER MODE. ADDRESS UNIMPORTANT
     [XXXX]             RETURN IS ALWAYS HERE
```

THE CONTENTS OF XR1 WILL BE USED TO LOAD THE BAR WITH THE CORRECT VALUE AND EXECUTE THE APPROPRIATE ISS INSTRUCTION. IF THE NUMBER OF ARGUMENTS IS TOO LARGE OR FOR ANY REASON IT IS NOT EASY TO TRANSFER THEM THROUGH THE REGISTERS, IT IS POSSIBLE TO LOAD AN INDEX REGISTER WITH THE LOCATION OF THE ARGUMENTS AND LET THE CALLED ROUTINE PICK UP THE ARGUMENTS WITH A SPECIAL CALL ON THE EXECUTIVE. THE FACT THAT THE ARGUMENTS OF A PARTICULAR EXECUTIVE SUBROUTINE ARE TO BE PICKED UP IN THIS MANNER MUST BE PART OF THE CALLING SEQUENCE AS THE CHOICE IS NEVER LEFT UP TO THE USER.

THE EXECUTION OF A MME IN THE USER PROGRAM CAUSES THE IMMEDIATE EXECUTION OF TWO INSTRUCTIONS IN MASTER MODE. THESE TWO INSTRUCTIONS ARE:

```
      STC1 USAVE        SAVE RETURN AND INDICATORS
      TRA  XL           TRANSFER TO LINK CODING

XL:   STBR USAVE+1      SAVE BASE ADDRESS REGISTER
      CMPX MAX,DU       CHECK FOR VALID CALL
      TNC? ERROR        IF NOT BETWEEN 0 AND MAX THEN ERROR
      LDBR BAR,1        LOAD PROPER BASE ADDRESS REGISTER
      ISS  TMS,1        AND TRANSFER TO THE EXEC MODULE
```

LEFT OUT OF THE ABOVE PROGRAM IS THE METHOD FOR INITIALIZING THE FAULT VECTOR. THIS MUST BE DONE BECAUSE THE CALLS OF AN EXECUTIVE SUBROUTINE MAY BE DIFFERENT THAN THAT OF A USER PROGRAM. SINCE A RESTRICTION ON ALL EXECUTIVE ROUTINES IS THAT THEY NEVER FAULT EXCEPT FOR A REGULAR CALL, ONLY THE MME FAULT MAY BE CHANGED IF IT IS FOUND THAT INITIALIZING THE ENTIRE FAULT VECTOR IS TAKING UP TOO MUCH TIME AND THAT THE EXECUTIVE ROUTINES ARE FAIRLY WELL DEBUGGED.

THERE ARE FOUR TYPES OF MODULES CONTAINING INSTRUCTIONS IN THE SYSTEM. THEY ARE.

1. USER PROGRAM. A USER PROGRAM IS A SUITABLE BODY OF CODE WITH A HIDDEN STATE VECTOR LOCATED BELOW THE ACCESSABLE RANGE OF CODE. IT MUST MAKE ALL MASTER MODE ENTRIES TO THE EXECUTIVE THROUGH PREVIOUSLY DEFINED MACROS. THESE MACROS ARE SUBJECT TO CHANGE WITH VERY LITTLE NOTICE.

2. EXECUTIVE SUBROUTINES. AN EXECUTIVE SUBROUTINE IS A NON-SWAPPABLE RE-ENTRANT BODY OF CODE THAT PERFORMS ALMOST ALL PRIVILEGED FUNCTIONS SUCH AS SCHEDULING OR DOING I/O SETUP. ALL SUCH ROUTINES HAVE A NON-ACCESSABLE HEADER THAT POINTS TO TEMPORARY STORAGE THAT MUST BE PRESERVED ON A SWAP AND A TABLE THAT IS USED FOR THE GETTING AND PUTTING OF PARAMETERS THROUGH THE MASTER PROGRAM. IT IS INITIALIZED AT LOAD TIME AND NEVER CHANGED AFTERWARDS. EXECUTIVE SUBROUTINES ARE NOT ALLOWED TO CAUSE ANY FAULTS WHATSOEVER EXCEPT BY USING A PREVIOUSLY DEFINED MACRO TO GET AND PUT PARAMETERS AND CALL OTHER EXECUTIVE SUBROUTINES OR RETURN TO THE USER'S PROGRAM.

3. MASTER PROGRAM. THIS PROGRAM IS THE MASTER MODE PROGRAM THAT LINKS SLAVE MODE ROUTINES TOGETHER AND HANDLES FAULTS. IT DOES ONLY WHAT CANNOT BE DONE EFFICIENTLY IN SLAVE MODE. THERE ARE NO RESTRICTIONS ON THE TYPE OF CODING USED IN THIS PROGRAM EXCEPT THAT THE CODING MUST BE STRAIGHT FORWARD AND VERY FLEXIBLE. THIS ROUTINE IS RESPONSIBLE FOR THE EXECUTION OF ALL MASTER MODE INSTRUCTIONS BUT IT DOES ALMOST NO SETTING UP OF THESE COMMANDS.

4. I/O INTERRUPT PROGRAM. THIS ROUTINE HANDLES ALL I/O INTERRUPTS. IT DOES LIMITED ERROR CORRECTION AND KEEPS A CURRENT TABLE OF THE AVAILABILITY OF I/O DEVICES. ITS MAIN PURPOSE IS TO INITIATE I/O COMMANDS ON THE COMPLETION OF PRECEDING I/O OPERATIONS AND PASS THE STATUS OF THE COMPLETED OPERATION TO THE ROUTINE THAT INITIATED THE OPERATION IN THE FIRST PLACE. THIS ROUTINE GETS CONTROL DIRECTLY FROM THE I/O INTERRUPT VECTOR AND MAY BE PART IN MASTER MODE AND PART IN SLAVE MODE. IT MUST IN ALL CIRCUMSTANCES RETURN TO THE PLACE IT INTERRUPTED FROM.

FILE ROUTINE CALLS FROM USER PROGRAMS

OPEN I,POINTER TO NAME,PASSWORD
 RETURNS A NUMBER IN A INDICATING:
 1. FILE NOT FOUND
 2. PROTECTION VIOLATION
 3. FILE FOUND AND OPENED, FILE NUMBER IN Q
 4. FILE FOUND BUT IT IS ON BACK UP STORAGE
    AND RESTRICTION BITS DO NOT PERMIT ACCESS

IF THE FILE IS OPENED, THEN THE RESTRICTIONS ON THE USE OF THE FILE ARE PLACED IN THE UPPER HALF OF THE Q REGISTER.

READ I,M,N,FLAG
 I IS THE NUMBER OF THE FILE TO BE READ.
 M IS THE MEMORY LOCATION OF THE FIRST WORD READ
 N IS THE NUMBER OF WORDS TO BE READ
 FLAG IS THE LOCATION OF THE END OF OPERATION FLAG.

 RETURNS A NUMBER IN A INDICATING:
 1. NORMAL RETURN
 2.