FILE: CMAC

Compliments of FRESS

A File Retrieval and Editing SyStem

Release 9.1 2 MAY 79

1 OVERVIEW

This explanation assumes that the reader is familiar with the FRESS NEWSLETTER of March 3, 1976 Vol. 2 No. 1, which initially introduced FRESS command macros. The original document may be retrieved by typing ".frsnews" while in FRESS.

This update explains new features and capabilities of command macros and also corrects errors in the previous documentation, particularly on the subject of macro expansion errors, contents of defaults, and restrictions for '<prompt>'s. Note that this new version of command macros is upward compatible with the previous one. This means that macros created previously will still work as they did before.

Also included is a note on the use of comments in macros.

2 ADDITIONAL FEATURES

2.1 CONTINUATION COLUMN

Ordinarily each line of the memo file defining the macro is considered as a separate command line to FRESS. These lines are limited to 80 characters in length by the editor used to create them. In some cases 80 characters is not enough to contain the whole command line. For this reason the user may specify concatenation of one line in the memo file to the next by placing a '*' in column 80. The next line will then be concatenated to the 79th column of the first line. There may be two such continuation lines in any one command line, that is, a total of three physical lines. Keep in mind that although the macro definition may be 3 lines long (including macro comments, prompts, etc.) the expanded FRESS command line that comes from this must be less than 130 characters in length. If the FRESS command line expanded from the macro definition is more than 130 characters, the error message "MACRO EXPANSION TOO LONG" will be printed and macro expansion and execution will be terminated.

Symbolic parameters (&1, &2, etc.), special function indicators (&R, &C, &I) and literal ampersands (&&) all consist of 2 characters which must not be split over card boundaries.

2.2 NEW SPECIAL FUNCTION, &R<N>

Format

```
&R<n>'<prompt>'Definition (or redefinition) and use
&R<n>          use
          <n> is a number between 1 and 9
```

Many users have used the &R special function to read a line from the terminal and then have found that they wanted to use the same line again later in the macro. &R<n> allows the macro to read a line from the user's terminal (as does &R) and then reuse that line later in the macro.

&R<n>'<prompt>' works just like &R'<prompt>' except that, in addition to placing the line from the terminal in the macro expansion, it also saves the line for future use. When &R<n>'<prompt>' is reached in a macro definition the <prompt> will be printed at the terminal, and a line will be read from the terminal. That line will be placed in the macro expansion in place of the &R<n>'<prompt>', and will be saved for future use. When &R<n> is subsequently included in the macro definition (where <n> has the same value used previously and the <prompt> is omitted), the saved line will be placed in the macro expansion in place of the &R<n>.

An &R<n> must be defined before it is used or an expansion error will occur and macro execution will terminate. &R<n> cannot itself take a default, but it can be included in a default for a symbolic parameter (&<n>). Any &R<n> may be redefined at any time by another &R<n>'<prompt>'.

Caution: If the &R<n>'<prompt>' occurs as part of a default the &R<n> is not defined unless the default is used. A null prompt may be specified by &R1''. The definitions do not have to proceed in numerical order, that is, &R2 may be defined and used before &R1. This new function may be used anywhere in a macro except inside a '<prompt>'.

Error Message

&R<n> HAS NOT BEEN DEFINED
: The stated &R<n> was used before initial definition was encountered. Macro expansion and execution are terminated.

Example

```
l/&R1'What to change:'
s/&R1/&R2'Change to:'
l/&R2
```

This macro locates a pattern, changes it to a new pattern, and then makes the pattern the first text in the buffer. The macro first prompts for the locate pattern. In the substitute command the same pattern is then used as the <scope> parameter and the macro prompts for the text string. The text string is used again in the third line as a locate pattern.

Example

(the line numbers are included for easy reference in the explanation given below)

```
(1)  mc off
(2)  g/&1|&R1'filename?'|
(3)  l/&1|&R1|>      &C'to find end of macros'
(4)  ib/&1|&R1|/&R2'new macro?'
(5)  s/&1|&R1|/&R1'new end of macro delimiter?'
(6)  g/macfile>      &C'GET universal macro file'
(7)  l/&R1>          &C'Find end of macro delimiter'
(8)  ib/&R1/&R2>     &C'put new macro in universal'
(9)  mc on
```

In this macro the comments (&C) are just internal documentation so they are turned off at the top and back on at the bottom (mc off, mc on). The second line will define &R1 to be the file name if the first symbolic parameter is omitted or left null. This same line will then get the file. The next line locates the file name in the file. The file name was used as a delimiter the end of the format code macro definitions in the file. The fourth line will define &R2 to be a new format code macro and insert it before the file name. Note that if the user typed a null line in response to "new macro?" this would define &R2 as null and the command would put the user into input mode. The subsequent input would not be saved as &R2.

The fifth line will use the saved file name in place of &R1 in its first instance and then redefine &R1 to be the new end of macros delimiter. The command will then replace the old delimiter (the file name) with the new delimiter. The sixth line gets a file that this user set up to store all format code macro definitions and the next two lines locate the new

end of format macros delimiter and inserts the new format code macro defined in line four before this delimiter.

2.3 NEW ERROR MESSAGE FROM FMACADD

If there is a bad character in the memo file, or an error occurs while reading the memo file, or an invalid character follows an ampersand (i.e. not &, I, R, C, or 1 to 9) then the following error message is printed:

FMAC: ERROR IN LINE <n> OF MEMO FILE

<n> is the line number of the line that is in error.

3 CORRECTIONS TO PREVIOUS DOCUMENTATION

3.1 RESTRICTIONS ON PROMPTS

Symbolic parameters (&<n>) and the special functions (&R, &I, &C, &R<n>) cannot be used inside a prompt. Literal ampersands should be specified as two ampersands (&&). If the user does inadvertently specify a symbolic parameter or special function inside a prompt or comment the offending singular ampersand and the character following it will be an unprintable character when the prompt is printed on the terminal. There are no error messages generated when this happens and it does not affect the operation of the special function or the macro expansion.

Prompts may now be left off of the &R special function if the user desires a null prompt. &R (not followed by a digit or an apostrophe) is the same as &R''.

3.2 CONTENTS OF DEFAULTS

Contrary to the description contained in the previous newsletter, defaults for symbolic parameters (&<n>) may contain any of the symbolic parameters, any of the special functions, or regular text as well as any combination of these. The only thing a default cannot contain is another default.

Example

```
i/&1/&3|&2 and &R'second half of inserted text:'|
```

In this macro, the default will be used if &3 is either omitted or left null. If the default is used, &2 must have been specified since it can have no default.

3.3 CLARIFICATION OF ERRORS

The following error messages indicate macro expansion errors:

```
MISSING SYMBOLIC PARAMETER
MACRO NOT FOUND
MACRO EXPANSION TOO LONG
MACRO NAME TOO LONG
INVALID DEFAULT LITERAL
&R<n> HAS NOT BEEN DEFINED
```

If these errors occur macro expansion and execution are both terminated, the error message is printed along with the last line it was trying to expand, and control is returned to the user in the FRESS environment. If errors occur during execution, (e.g. "PATTERN NOT FOUND"), the error message will be printed along with the command line causing the error. The user will then be asked to accept (A) or reject (R) the remaining functions. Remaining functions include the rest of the command macro and any commands that were on the same line as the macro invocation (after a command seperator).

4 NOTES ON THE USE OF COMMAND MACRO COMMENTS (&C)

For clear internal documentation of a command macro it is often desirable to include a comment on the end of a command line. To make sure that intervening blanks between the end of the command and the &C'<comment>' are not included in the command line the FRESS command seperator (">") should be used. To suppress the printing of the comments on the terminal the MCOMMENT command (mc off, mc on) should be used in the macro.

Example

```
     i/&1/NOT     &C'negate it'
```

will insert the text string "NOT   ", whereas,

```
     i/&1/NOT>     &C'negate it'
```

will insert "NOT".

Temporary overriding of display modes as described in Section 4.2 of the FRESS User's Guide can be accomplished in a command macro if the user remembers the following. Macro comments must immediately follow the period (or modifier, if present) with no intervening blanks. This is necessary because the period (or modifier) must be the last thing on the command line and the comment ends the command line if nothing follows the comment. Therefore blanks between the period and the comment are included in the command line and overriding will not occur.

Example

```
     10>p 10  .*&C'print next 10 lines'
```

is correct, whereas

```
     10>p 10  .*  &C'print next 10 lines'
```

and

```
     10>p 10  .*>  &C'print next 10 lines'
```

are not correct, since " .*" are not the last characters on the command line.

TABLE OF CONTENTS

FILE: CMACDOC

Compliments of FRESS

A File Retrieval and Editing SyStem

Release 9.1 2 MAY 79

A command macro is defined by the contents of a CMS MEMO file with the same name as that desired for the macro. Each line of the MEMO file is considered to be a FRESS command line and may contain multiple commands separated by the FRESS logical command separator ">", but may not contain nested macro calls. Trailing blanks are removed from lines during processing; therefore if the user desires to have trailing blanks on a command line the line must be ended with a command separator (see example in Section 7). Each command in the macro consists of literal character strings, special functions (described below in Section 2.2) and symbolic parameter indicators which represent values to be filled in when the user invokes the macro.

There may be problems if an upper-case only terminal is used because the standard CMS editor translates all input typed on such a terminal into uppercase. Thus it would be impossible to specify a lower-case character string as part of a MEMO file. This can be avoided by creating macros on uperr/lower case terminals or by using NEWEDIT. Documentation on NEWEDIT is available from the Computer Center consultant.

1 SYMBOLIC PARAMETERS

The symbolic parameter indicators are of the form "&N", representing the "Nth" parameter, where N may be any number from 1 to 9. A symbolic parameter indicator may optionally be followed by a string of the form "|<default>|" where "<default>" is a character string and the or-bars are included literally as shown. If a parameter represented by a symbolic parameter indicator is omitted or left null[1] when the macro is invoked, the <default>, if present, will be substituted in its place. The <default> string may be the null string or any text but may not itself contain any symbolic parameter indicators. If a parameter is omitted when no default is present an error message will be typed and the command macro will not be executed. Parameters may, however, be left null when no default is present; in this case, the null string will be substituted in its place. Note that each separate occurrence of a symbolic parameter indicator may have its own particular associated default string.

Example 1:

Consider the macro:

```
SA
T &1|15|
RET
```

This macro will type a specified number of lines and then return the display to the position before the Type was performed. If no parameter is specified in invoking this macro or if the parameter is left null the number of lines typed defaults to 15.

Example 2:

If, instead, the macro had been specified as:

```
SA
T &1
RET
```

---

[1]To leave a parameter null you would include a key delimiter for the parameter but omit the value for it. For example, to omit the parameter in a Locate command you would type "l"; to leave it null you would type "l/".

then leaving out the parameter would result in an error message and none of the three commands in the macro would be executed. If the parameter were left null the commands would be executed, but an error would occur because Type has a required parameter.

2 SPECIAL FUNCTIONS

Three special functions may also be embedded insde macro definitions. Two new FRESS commands have been added to manipulate these functions.

2.1 &COMMENT

Format: &C'<text string>'

Embedding an &C function inside a macro will cause the character string enclosed in apostrophes to be printed at the user's terminal. This facility can be used to monitor progress through a command macro without expanding the entire macro (see Section 3) or to warn the user of the macro about its effects. All text between the apostrophes is considered literal text; that is, no parameter substitution can occur. No apostrophes are allowed in the text string.

A new FRESS command has been added to allow the user to determine whether these "macro comments" will be printed. The command is of the form:

MCOMMENT <option>

where <option> can be "ON" or "OFF"; "ON" is the default.

2.2 &READ

Format: &R'<text string>'

Embedding an &R function inside a macro will cause the character string enclosed in apostrophes to be printed and a read to be done at the user's terminal. If the character string is null, nothing will be printed but a read will be done. The string typed by the user will replace the &R function in the macro expansion. Both the <text string> and the input from the user are considered literal text; no parameter substitution is done.

This function can be used to prompt the user for required parameters which are not specified initially and for which no standard default can be used. For example, consider a macro which retrieves a file, locates a pattern at the end of the format code macro definitions (to make sure they are all defined) and then travels to a particular decimal block. This might be written as:

```
g/&1|&r'Filename?'|
l/!.end of macros
gdl/&2|&r'Dec block?'|
```

If &1 or &2 are not specified when the macro is invoked, the message "Filename?" or "Dec block?" will be printed when that line in the macro is reached. Notice that &R may be used as the default value for a symbolic parameter. However, there can be no default value for an &R function.

2.3 &IDENTIFIER

Format: &I

Embedding an &I function inside a macro will cause the function to be replaced by a 4-character identifier previously defined. This facility provides a (limited) means of setting global variables. The macro identifier is set using the FRESS command:

MIDENTIFIER <4-character string>

If the string is shorter than 4 characters it will be padded on the right with blanks.

3 USING COMMAND MACROS

Command macros are invoked or "called" much as ordinary commands are. The differences between the two methods are:

1) A command macro name must be preceded by a period or a comma to indicate it is a macro. If a comma is used, each expanded command line (preceded by a colon) is printed at the terminal before execution. If a period is used the expanded lines are not typed.

2) A command macro must be the first command on a line and must not be preceded by any blanks.

3) If a command macro is followed by a command separator and an ordinary FRESS command, the entire expanded macro will be executed before the ordinary FRESS command is parsed and executed.

As in normal command lines, key delimiters are used to separate the parameters for the macro. The macro itself will also make use of key delimiters to separate the parameters for the FRESS commands. These two sets of key delimiters bear no direct relation and need not use the same character. For instance, if the macro in Example 1 were called MAC it could be invoked by typing:

```
.MAC 20
```

which uses the same key delimiter as the macro itself (a blank) or by typing:

```
.MAC/20
```

or using any other key delimiter.

A parameter is left null by specifying a key delimiter but no value for it. Thus the parameter in Example 1 would be left null by typing:

```
.MAC/
```

Parameters in excess of those required by the macro are considered to be part of the last parameter.

Note that there are no optional parameters to macros. Parameters may be omitted only if an explicit default is given for each occurrence of it in the macro.

If any error occurs during expansion or execution of a command macro, the relevant error message is typed, followed by the expansion of the line causing the error, followed by the message:

COMMAND MACRO: ACCEPT (A) OR REJECT (R) REMAINING FCNS

If the user accepts by typing A, the execution of the macro will proceed but the line containing the error will be skipped. If the user rejects, expansion and execution of the macro immediately halts.

Each line of the MEMO file is expanded separately. That is, all parameter and default substitutions are made and the resulting line is then executed as a FRESS command line. This expanded line must be less than 130 characters in length. The unexpanded line must be less than 80 characters in length because longer lines are truncated to 80 characters by the CMS editor.

4 MACRO LIBRARIES

In order to be invoked from FRESS, command macros must be contained in a macro library. A FRESS command macro library may contain any number of command macro definitions. These libraries should never be edited directly. A set of utility functions has been provided which must be used to do the necessary manipulations such as adding to and deleting from these libraries.[1] In the following descriptions <libename> represents a macro library name and <macname> represents a command macro title and MEMO file name. Filenames may be up to 8 characters in length. The only restriction is that no macro library may have the name "D" or "FMAC", which is the name of the system command macro library. Each description is folowed by a list of possible error messages. These commands must be executed in CMS; they may not be executed from FRESS.

4.1 LIBRARY COMMANDSFMACADD <LIBENAME> <MACNAME1> ... <MACNAME10>THE SPECIFIED MEMO FILES ARE ADDED TO THE SPECIFIED MACRO LIBRARY. IF <LIBENAME> DID NOT PREVIOUSLY EXIST IT WILL BE CREATED. IF <LIBENAME> ALREADY CONTAINED a copy of any of the specified macros those particular "adds" will be ignored. All MEMO files which are successfully added to <libename> will be erased.<libename> MACLIB BEING GEN'ED<libename> did not exist and is being created.FATAL MACLIB GEN ERROR <n>FMACDEL <libename> <macname1> ... <macname10>The specified macros are deleted from the specified macro library. Unlike FMACSPLT, no MEMO file is created.FMACLIST <libename>The names of the macros contained in the specified macro library will be printed at the user's terminal.FMAC: FATAL MACLIB LIST ERROR <n>FMACSPLT <libename> <macname1> ... <macname10>The specified macros are recreated as MEMO files but not deleted from the macro library. They can then be edited.FMAC: FATAL LSPLIT ERROR <n>FMAC: <macname> NOT FOUNDThe specified macro was not found in the specified macro library.FMACCVTM ERROR <n> ON <macname> COPYFMACREP <libename> <macname1> ... <macname10>The specified macros in the specified library will be replaced by their current MEMO file representation. The MEMO files will then be erased.FMACPRNT <libename>The contents of the specified macro library will be printed on the high-speed printer on STD TN forms. The output is formatted with

---

[1]The ordinary CMS macro library commands may not be used because the FRESS command macro libraries use a special internal format.

separators between individual macros and each macro is preceded by its name. This function tends to be expensive. It will erase MEMO files on the P-disk that have the same names as members of the macro library. The function requires no more than twice the macro liberary size in available space on the P-disk.

4.2 USE OF LIBRARY COMMANDS

FMACADD is used to add new macro definitions to existing libraries or to create new macro libraries. Thus to add the macros MAC1 and MAC2 to library XLIB the command would be:

```
FMACADD XLIB MAC1 MAC2
```

MAC1 MEMO and MAC2 MEMO will be erased. If the user desired to change the contents of MAC2 he would use FMACSPLT to make a new copy as a MEMO file:

```
FMACSPLT XLIB MAC2
```

Note that the macro MAC2 still exists in XLIB. When finished editing the MEMO file he would replace the old copy in XLIB by the new copy in the MEMO file by typing:

```
FMACREP XLIB MAC2
```

If he wished to delete MAC1 from the library he would type:

```
FMACDEL XLIB MAC1
```

To obtain a list of the macros currently in XLIB he would type:

```
FMACLIST XLIB
```

To obtain a hardcopy printout of the contents of the library he would type:

```
FMACPRNT XLIB
```

5 MACROS AND ENTERING FRESS

When entering FRESS the user must specify the macro libraries to be searched for macro definitions. These are specified when FRESS is invoked:

FRESS [NOS] <option> <libename1> ... <libename9>

(NOS is specified if the user wishes to avoid the automatic "spool e as std tn" which is otherwise executed when FRESS is entered. The <option> corresponds to the version of FRESS being used; see Section B.1 of the User's Guide for more details)

The libraries will be searched in the specified order; that is, if duplicate macro definitions exist the one in the library specified earliest will be used. The system macro library will be searched last. If any specified library does not exist the message

MACLIB ERROR

will be typed and the user will be returned to CMS.

6 MACRO EXPANSION ERROR MESSAGES

The following errors may occur during expansion of a macro when it is invoked from FRESS. If any of these messages occur, the user is given the option to halt or continue the macro execution (see Section 3).

MISSING SYMBOLIC PARAMETER
: A macro call must specify a value for each symbolic parameter in the macro which does not have a default string associated with it. This value may be the null string, which is specified by including a key delimiter but no explicit value for the parameter. This message indicates that the macro call did not specify values for one or more symbolic parameters without such defaults.

MACRO NOT FOUND
: The indicated macro was not found in any of the libraries specified on the CMS command line used to invoke FRESS.

MACRO EXPANSION TOO LONG
: An expanded line exceeded 130 characters.

MACRO NAME TOO LONG
: Macro names may not exceed eight characters.

INVALID DEFAULT LITERAL
: The ending or-bar (|) was not found for a default literal within a given line of the macro.

7 EXAMPLES OF COMMAND MACRO USAGE

The following examples further illustrate the use of command macros. Consider a FRESS command macro in a MEMO file called MAC:

```
gf/&1
gl/&2
&3 .
p &3|10|
```

To add this macro to a macro library called MYLIB the user would type, in CMS:

```
FMACADD MYLIB MAC
```

FRESS may now be entered by specifying:

```
fress t mylib
```

THE MACRO may be invoked by typing:

```
,mac$myfile$mylab$4
```

This would expand into:

```
:gf/myfile
:gl/mylab
:4 .
:p 4
```

If the third parameter had been left null by typing:

```
,mac$myfile$mylab$
```

the expansion of the macro would have produced:

```
:gf/myfile
:gl/mylab
:  .
:p 10
```

The third line will be ignored when executed.

If, however, the third parameter had been omitted, by typing:

```
,mac$myfile$mylab
```

an error message would be typed and the macro would not be executed. If the user wishes to be able to omit parameters, a default string must be associated with each occurrence of these parameters. The user would then specify the last lines of the macro as:

```
&3||
p &3|10|
```

As mentioned previously, trailing blanks may be used in a macro definition only if followed by a command separator. Thus to include the command:

```
l/textØØØ
```

in a macro the line in the MEMO file should be:

```
l/textØØØ>
```

8 SYSTEM MACRO LIBRARY

The system macro library currently contains the .FRSCOMM, .FRSNEWS, .FRSMISC, and .FRSMONTH macros (described in the new User's Guide and in a handout available from the Computer Center consultant) and the functions described below. Future additions to the library will be described in future newsletters.

```
.FU <spool class>° <FU options>°
```

<spool class> is the spooling class (NW, NC, 340, etc.); TN is the default

<FU options> are options for Fullprint; o9 is the default

Purpose:

To simplify the procedure for Fullprinting a FRESS file on narrow white special forms with the TN print chain. If no parameters are specified, a file of width 65 will be printed, centered, on narrow white paper.

Expected environment:

The file to be printed must be the current file.

.REMOTEFU

Purpose:

To make it easy for non-FRESS users to Fullprint a file on a terminal.

Expected environment:

No special requirements.

Macro messages:

- "Enter filename:"

  Type in the name of the file to be printed.

- "Enter options (if any):"

  Type in any Fullprint options desired or a null line if no special options are required. The "2741" option for Fullprint to the terinal will be used in either case.

- "Position paper; hit CR"

  The Fullprint is ready to begin. Position the top edge of the paper below the printing position of the terminal, then hit Carriage Return.

.GET

Purpose:

To simulate, to a limited extent, the GET facility of the CMS editor by inserting a Fress file into the current file.

Expected environment:

There must be a current (open) file; the file to be gotten must exist as a Fress file.

Macro messages:

- "Where should the file be inserted?"

  Type in the unique character string that identifies the location after which the file is to be inserted.

- "Enter name of file to be inserted:"

  Type in a Fress file name. That file will be inserted in the current file at the point you specified.

Notes:

1) Once the macro has begun execution you cannot scroll to a particular location, so be sure you know beforehand just where the copied file is to go.

.PUT

Purpose:

To write part of a Fress file into a new file.

Expected environment:

There must be a current file; the file to be created must not already exist.

Macro messages:

- "Enter name of file to be created:"

  Type in the name of the new file that is to contain part of the current file.

- "Enter starting point:"

  Type in the unique character string that identifies the beginning of the section of text to be written to the new file.

- "Enter travel command or null line:"

  Type in a travel command if you must travel to find the endpoint for the copy. If no traveling is necessary, type a null line.

- "Enter ending point:"

  Type in the unique character string that identifies the end of the section of text to be written to the new file.

Notes:

1) To copy an entire file, use the Fress CF (copy file) command.

2) Care should be taken to ensure that the definitions of any edit macros used in the copied text are also copied.

3) If the macro is aborted after a file name has been entered, the new (empty) file will have been created and must be scratched for the .PUT macro to work.

.ICARDS

Purpose:

To insert a card-image file into the current Fress files and perform any conversion necessary to give the file its original appearance when it is printed by the Fress FULLPRINT program.

Expected environment:

There must be a current file; the file to be inserted must be fixed-length, 80 byte format.

Macro messages:

- "Enter card file name:"

  Type in the name of the file to be inserted.

- "Enter card file type:"

  Type in the type of the file to be included.

- "Where should the file be inserted?"

  Type in the unique character string that identifies the location after which the file is to be inserted.

Notes:

1) The .ICARDS macro creates temporary files named $TEMP $FILE and $TEMP FRESS. Make sure you don't have files with the same names.

2) Conversion for files that are not MEMO files proceeds as follows:

   a) The first line inserted in the current file is the definition of an edit macro named !.NL. (for new line). This macro is inserted at the start of every card-image line put in your file. This scheme has been chosen to make the format of the inserted file easy to change.

   b) All blanks in the card image file are translated to literal blanks (C'¬') so that the spacing on the inserted lines is not altered.

   c) Trailing blanks on the inserted cards are ignored; serial numbers are also ignored.

   d) The Fress converter used by the macro converts all ampersands in your file to the string "!80*". Such strings will still appear as ampersands on printed output.

3) MEMO files are converted as follows:

   a) The first line inserted in the current file is the definition of an edit macro named !.SK. (for "skip"). The macro is inserted as a replacement for any blank cards found in the memo file.

   b) Blanks are <u>not</u> translated to literal blanks but left as is. This makes editing easier but also means that MEMO files, when printed, will not look the same as they originally did.

TABLE OF CONTENTS

FILE: CMACDOC

Compliments of FRESS

A File Retrieval and Editing SyStem

Release 9.1 2 MAY 79

FRESS NEWSLETTER

Vol. 2, No. 1
January 25, 1919

FRESS COMMAND MACROS

1 OVERVIEW

A command macro is a series of FRESS commands grouped so that they may be invoked with a single command name. Parameters may be passed to the macro in the same way parameters are specified in an ordinary FRESS command. Command macros are designed to reduce the number of keystrokes needed to peform standard operations or to make a complicated operation easier. In general only relatively sophisticated FRESS users will construct their own macros; others can use the system defined macros or seek help from the FRESS staff in building their own.

The command macro facility was designed and implemented by R.J. Harrington, C.J. Mathias, and D.L. Irvine. Questions about macros should be directed to the FRESS staff (Philip Wisoff) at 863-2286. Suggstions as to functions to be included in the system library are welcome.

2 CREATING COMMAND MACROS

2.1 COMMAND MACRO FORMAT

FRESS command macros are created and changed using the CMS editor. Information concerning the use of Brown's version of the CMS editor may be found in the CP-67/CMS User's Guide (GH20-0859-2) and the Brown University Interactive User's Guide.

A command macro is defined by the contents of a CMS MEMO file with the same name as that desired for the macro. Each line of the MEMO file is considered to be a FRESS command line and may contain multiple commands separated by the FRESS logical command

separator ">", but may not contain nested macro calls. Trailing blanks are removed from lines during processing; therefore if the user desires to have trailing blanks on a command line the line must be ended with a command separator (see example in Section 7). Each command in the macro consists of literal character strings, special functions (described below in Section 2.2) and symbolic parameter indicators which represent values to be filled in when the user invokes the macro.

There may be problems if an upper-case only terminal is used because the standard CMS editor translates all input typed on such a terminal into uppercase. Thus it would be impossible to specify a lower-case character string as part of a MEMO file. This can be avoided by creating macros on uperr/lower case terminals or by using NEWEDIT. Documentation on NEWEDIT is available from the Computer Center consultant.

2.1.1 SYMBOLIC PARAMETERS

The symbolic parameter indicators are of the form "&N", representing the "Nth" parameter, where N may be any number from 1 to 9. A symbolic parameter indicator may optionally be followed by a string of the form "|<default>|" where "<default>" is a character string and the or-bars are included literally as shown. If a parameter represented by a symbolic parameter indicator is omitted or left null[1] when the macro is invoked, the <default>, if present, will be substituted in its place. The <default> string may be the null string or any text but may not itself contain any symbolic parameter indicators. If a parameter is omitted when no default is present an error message will be typed and the command macro will not be executed. Parameters may, however, be left null when no default is present; in this case, the null string will be substituted in its place. Note that each separate occurrence of a symbolic parameter indicator may have its own particular associated default string.

Example 1:

Consider the macro:

```
SA
T &1|15|
RET
```

---

[1]To leave a parameter null you would include a key delimiter for the parameter but omit the value for it. For example, to omit the parameter in a Locate command you would type "1"; to leave it null you would type "1/".

This macro will type a specified number of lines and then return the display to the position before the Type was performed. If no parameter is specified in invoking this macro or if the parameter is left null the number of lines typed defaults to 15.

Example 2:

If, instead, the macro had been specified as:

```
SA
T &1
RET
```

then leaving out the parameter would result in an error message and none of the three commands in the macro would be executed. If the parameter were left null the commands would be executed, but an error would occur because Type has a required parameter.

2.2 SPECIAL FUNCTIONS

Three special functions may also be embedded insde macro definitions. Two new FRESS commands have been added to manipulate these functions.

2.2.1 &COMMENT

Format: &C'<text string>'

Embedding an &C function inside a macro will cause the character string enclosed in apostrophes to be printed at the user's terminal. This facility can be used to monitor progress through a command macro without expanding the entire macro (see Section 3) or to warn the user of the macro about its effects. All text between the apostrophes is considered literal text; that is, no parameter substitution can occur. No apostrophes are allowed in the text string.

A new FRESS command has been added to allow the user to determine whether these "macro comments" will be printed. The command is of the form:

MCOMMENT <option>

where <option> can be "ON" or "OFF"; "ON" is the default.

2.2.2 &READ

Format: &R'<text string>'

Embedding an &R function inside a macro will cause the character string enclosed in apostrophes to be printed and a read to be done at the user's terminal. If the character string is null, nothing will be printed but a read will be done. The string typed by the user will replace the &R function in the macro expansion. Both the <text string> and the input from the user are considered literal text; no parameter substitution is done.

This function can be used to prompt the user for required parameters which are not specified initially and for which no standard default can be used. For example, consider a macro which retrieves a file, locates a pattern at the end of the format code macro definitions (to make sure they are all defined) and then travels to a particular decimal block. This might be written as:

```
g/&1|&r'Filename?'|
1/!.end of macros
gdl/&2|&r'Dec block?'|
```

If &1 or &2 are not specified when the macro is invoked, the message "Filename?" or "Dec block?" will be printed when that line in the macro is reached. Notice that &R may be used as the default value for a symbolic parameter. However, there can be no default value for an &R function.

2.2.3 &IDENTIFIER

Format: &I

Embedding an &I function inside a macro will cause the function to be replaced by a 4-character identifier previously defined. This facility provides a (limited) means of setting global variables. The macro identifier is set using the FRESS command:

MIDENTIFIER <4-character string>

If the string is shorter than 4 characters it will be padded on the right with blanks.

Command macros are invoked or "called" much as ordinary commands are. The differences between the two methods are:

1) A command macro name must be preceded by a period or a comma to indicate it is a macro. If a comma is used, each expanded command line (preceded by a colon) is printed at the terminal before execution. If a period is used the expanded lines are not typed.

2) A command macro must be the first command on a line and must not be preceded by any blanks.

3) If a command macro is followed by a command separator and an ordinary FRESS command, the entire expanded macro will be executed before the ordinary FRESS command is parsed and executed.

As in normal command lines, key delimiters are used to separate the parameters for the macro. The macro itself will also make use of key delimiters to separate the parameters for the FRESS commands. These two sets of key delimiters bear no direct relation and need not use the same character. For instance, if the macro in Example 1 were called MAC it could be invoked by typing:

```
.MAC 20
```

which uses the same key delimiter as the macro itself (a blank) or by typing:

```
.MAC/20
```

or using any other key delimiter.

A parameter is left null by specifying a key delimiter but no value for it. Thus the parameter in Example 1 would be left null by typing:

```
.MAC/
```

Parameters in excess of those required by the macro are considered to be part of the last parameter.

Note that there are no optional parameters to macros. Parameters may be omitted only if an explicit default is given for each occurrence of it in the macro.

If any error occurs during expansion or execution of a command macro, the relevant error message is typed, followed by

the expansion of the line causing the error, followed by the message:

COMMAND MACRO: ACCEPT (A) OR REJECT (R) REMAINING FCNS

If the user accepts by typing A, the execution of the macro will proceed but the line containing the error will be skipped. If the user rejects, expansion and execution of the macro immediately halts.

Each line of the MEMO file is expanded separately. That is, all parameter and default substitutions are made and the resulting line is then executed as a FRESS command line. This expanded line must be less than 130 characters in length. The unexpanded line must be less than 80 characters in length because longer lines are truncated to 80 characters by the CMS editor.

4 MACRO LIBRARIES

In order to be invoked from FRESS, command macros must be contained in a macro library. A FRESS command macro library may contain any number of command macro definitions. These libraries should never be edited directly. A set of utility functions has been provided which must be used to do the necessary manipulations such as adding to and deleting from these libraries.[1] In the following descriptions <libename> represents a macro library name and <macname> represents a command macro title and MEMO file name. Filenames may be up to 8 characters in length. The only restriction is that no macro library may have the name "D" or "FMAC", which is the name of the system command macro library. Each description is folowed by a list of possible error messages. These commands must be executed in CMS; they may not be executed from FRESS.

4.1 LIBRARY COMMANDS

- FMACADD <libename> <macname1> ... <macname10>

The specified MEMO files are added to the specified macro library. If <libename> did not previously exist it will be created. If <libename> already contained a copy of any of the specified macros those particular "adds" will be ignored. All

[1]The ordinary CMS macro library commands may not be used because the FRESS command macro libraries use a special internal format.

MEMO files which are successfully added to <libename> will be erased.

Messages:

FMAC: NO MACLIB TYPED
No <libename> was specified.

FMAC: NO MACROS TYPED
No <macname>s were specified.

FMAC: MORE THAN 10 MACROS TYPED; FIRST 10 USED
The limit of 10 <macname>s has been exceeded; all but the first 10 have been ignored.

FMAC: <macname> MEMO NOT FOUND
The specified macro does not exist as a MEMO file. Execution of the ADD continues for the other macros.

FMACCVTC ERROR <n> ON <macname> MEMO
A serious error has occurred. Contact a member of the FRESS staff.

<libename> MACLIB BEING GEN'ED
<libename> did not exist and is being created.

FATAL MACLIB ADD ERROR <n>
A serious error has occurred. Contact a member of the FRESS staff.

FATAL MACLIB GEN ERROR <n>
A serious error has occurred. Contact a member of the FRESS staff.

- FMACDEL <libename> <macname1> ... <macname10>

The specified macros are deleted from the specified macro library. Unlike FMACSPLT, no MEMO file is created.

Messages:

FMAC: NO MACLIB TYPED
No <libename> was specified.

FMAC: NO MACROS TYPED
No <macname>s were specified.

FMAC: MORE THAN 10 MACROS TYPED; FIRST 10 USED
The limit of 10 <macname>s has been exceeded; all but the first 10 have been ignored.

FMAC: MACLIB NOT FOUND ON R/W P-DISK
The specified macro library could not be updated because it is not on a Read-Write P-disk. The command has been ignored.

FMAC: ONE OR MORE MACROS NOT IN MACLIB; NOTHING DONE
One of the specified <macname>s is not in the specified library. The command has been ignored.

FMAC: FATAL MACLIB DEL ERROR <n>
A serious error has occurred. Contact a member of the FRESS staff.

FMAC: FATAL MACLIB COMP ERROR <n>
A serious error has occurred. Contact a member of the FRESS staff.

• FMACLIST <libename>
The names of the macros contained in the specified macro library will be printed at the user's terminal.

Messages:
FMAC: NO MACLIB TYPED
No <libename> was specified.
FMAC: MACLIB NOT FOUND
The specified library does not exist.
FMAC: FATAL MACLIB LIST ERROR <n>
A serious error has occurred. Contact a member of the FRESS staff.

• FMACSPLT <libename> <macname1> ... <macname10>
The specified macros are recreated as MEMO files but not deleted from the macro library. They can then be edited.

Messages:
FMAC: NO MACLIB TYPED
No <libename> was specified.
FMAC: NO MACROS TYPED
No <macname>s were specified.
FMAC: MACLIB NOT FOUND
The specified library does not exist.
FMAC: MORE THAN 10 MACROS TYPED; FIRST 10 USED
The limit of 10 <macname>s has been exceeded; all but the first 10 have been ignored.
FMAC: FATAL LSPLIT ERROR <n>
A serious error has occurred. Contact a member of the FRESS staff.
FMAC: <macname> NOT FOUND
The specified macro was not found in the specified macro library.
FMACCVTM ERROR <n> ON <macname> COPY
A serious error has occurred. Contact a member of the FRESS staff.

• FMACREP <libename> <macname1> ... <macname10>
The specified macros in the specified library will be replaced by their current MEMO file representation. The MEMO files will then be erased.

Messages:
FMAC: NO MACLIB TYPED
No <libename> was specified.
FMAC: NO MACROS TYPED
No <macname>s were specified.
FMAC: MORE THAN 10 MACROS TYPED; FIRST 10 USED
The limit of 10 <macname>s has been exceeded; all but the first 10 have been ignored.
FMAC: <macname> MEMO NOT FOUND

The specified macro does not exist as a MEMO file. Execution of the ADD continues for the other macros.

FMACCVTC ERROR <n> ON <macname> MEMO
A serious error has occurred. Contact a member of the FRESS staff.

FMAC: MACLIB NOT FOUND ON R/W P-DISK
The specified macro library could not be updated because it is not on a Read-Write P-disk. The command has been ignored.

FMAC: ONE OR MORE MACROS NOT IN MACLIB; NOTHING DONE
One of the specified <macname>s is not in the specified library. The command has been ignored.

FMAC: FATAL MACLIB DEL ERROR <n>
A serious error has occurred. Contact a member of the FRESS staff.

FMAC: FATAL MACLIB COMP ERROR <n>
A serious error has occurred. Contact a member of the FRESS staff.

FATAL MACLIB ADD ERROR <n>
A serious error has occurred. Contact a member of the FRESS staff.

- FMACPRNT <libename>

The contents of the specified macro library will be printed on the high-speed printer on STD TN forms. The output is formatted with separators between individual macros and each macro is preceded by its name. This function tends to be expensive. It will erase MEMO files on the P-disk that have the same names as members of the macro library. The function requires no more than twice the macro liberary size in available space on the P-disk.

4.2 USE OF LIBRARY COMMANDS

FMACADD is used to add new macro definitions to existing libraries or to create new macro libraries. Thus to add the macros MAC1 and MAC2 to library XLIB the command would be:

```
FMACADD XLIB MAC1 MAC2
```

MAC1 MEMO and MAC2 MEMO will be erased. If the user desired to change the contents of MAC2 he would use FMACSPLT to make a new copy as a MEMO file:

```
FMACSPLT XLIB MAC2
```

Note that the macro MAC2 still exists in XLIB. When finished editing the MEMO file he would replace the old copy in XLIB by the new copy in the MEMO file by typing:

```
FMACREP XLIB MAC2
```

If he wished to delete MAC1 from the library he would type:

```
FMACDEL XLIB MAC1
```

To obtain a list of the macros currently in XLIB he would type:

```
FMACLIST XLIB
```

To obtain a hardcopy printout of the contents of the library he would type:

```
FMACPRNT XLIB
```

5 MACROS AND ENTERING FRESS

When entering FRESS the user must specify the macro libraries to be searched for macro definitions. These are specified when FRESS is invoked:

```
FRESS [NOS] <option> <libename1> ... <libename9>
```

(NOS is specified if the user wishes to avoid the automatic "spool e as std tn" which is otherwise executed when FRESS is entered. The <option> corresponds to the version of FRESS being used; see Section B.1 of the User's Guide for more details)

The libraries will be searched in the specified order; that is, if duplicate macro definitions exist the one in the library specified earliest will be used. The system macro library will be searched last. If any specified library does not exist the message

```
MACLIB ERROR
```

will be typed and the user will be returned to CMS.

6 MACRO EXPANSION ERROR MESSAGES

The following errors may occur during expansion of a macro when it is invoked from FRESS. If any of these messages occur, the user is given the option to halt or continue the macro execution (see Section 3).

MISSING SYMBOLIC PARAMETER
: A macro call must specify a value for each symbolic parameter in the macro which does not have a default string associated with it. This value may be the null string, which is specified by including a key delimiter but no explicit value for the parameter. This message indicates that the macro call did not specify values for one or more symbolic parameters without such defaults.

MACRO NOT FOUND
: The indicated macro was not found in any of the libraries specified on the CMS command line used to invoke FRESS.

MACRO EXPANSION TOO LONG
: An expanded line exceeded 130 characters.

MACRO NAME TOO LONG
: Macro names may not exceed eight characters.

INVALID DEFAULT LITERAL
: The ending or-bar (|) was not found for a default literal within a given line of the macro.

7 EXAMPLES OF COMMAND MACRO USAGE

The following examples further illustrate the use of command macros. Consider a FRESS command macro in a MEMO file called MAC:

```
gf/&1
gl/&2
&3 .
p &3|10|
```

To add this macro to a macro library called MYLIB the user would type, in CMS:

```
FMACADD MYLIB MAC
```

FRESS may now be entered by specifying:

```
fress t mylib
```

THE MACRO may be invoked by typing:

```
,mac$myfile$mylab$4
```

This would expand into:

```
:gf/myfile
:gl/mylab
:4 .
:p 4
```

If the third parameter had been left null by typing:

```
,mac$myfile$mylab$
```

the expansion of the macro would have produced:

```
:gf/myfile
:gl/mylab
:  .
:p 10
```

The third line will be ignored when executed.

If, however, the third parameter had been omitted, by typing:

```
,mac$myfile$mylab
```

an error message would be typed and the macro would not be executed. If the user wishes to be able to omit parameters, a default string must be associated with each occurrence of these parameters. The user would then specify the last lines of the macro as:

```
&3||
p &3|10|
```

As mentioned previously, trailing blanks may be used in a macro definition only if followed by a command separator. Thus to include the command:

```
l/textØØØ
```

in a macro the line in the MEMO file should be:

```
l/textØØØ>
```

8 SYSTEM MACRO LIBRARY

The system macro library currently contains the .FRSCOMM, .FRSNEWS, .FRSMISC, and .FRSMONTH macros (described in the new User's Guide and in a handout available from the Computer Center consultant) and the functions described below. Future additions to the library will be described in future newsletters.

```
.FU <spool class>° <FU options>°
```

<spool class> is the spooling class (NW, NC, 340, etc.); TN is the default

<FU options> are options for Fullprint; o9 is the default

Purpose:

To simplify the procedure for Fullprinting a FRESS file on narrow white special forms with the TN print chain. If no parameters are specified, a file of width 65 will be printed, centered, on narrow white paper.

Expected environment:

The file to be printed must be the current file.

.REMOTEFU

Purpose:

To make it easy for non-FRESS users to Fullprint a file on a terminal.

Expected environment:

No special requirements.

Macro messages:

- "Enter filename:"

  Type in the name of the file to be printed.

- "Enter options (if any):"

  Type in any Fullprint options desired or a null line if no special options are required. The "2741" option for Fullprint to the terinal will be used in either case.

- "Position paper; hit CR"

  The Fullprint is ready to begin. Position the top edge of the paper below the printing position of the terminal, then hit Carriage Return.

.GET

Purpose:

To simulate, to a limited extent, the GET facility of the CMS editor by inserting a Fress file into the current file.

Expected environment:

There must be a current (open) file; the file to be gotten must exist as a Fress file.

Macro messages:

- "Where should the file be inserted?"

  Type in the unique character string that identifies the location after which the file is to be inserted.

- "Enter name of file to be inserted:"

  Type in a Fress file name. That file will be inserted in the current file at the point you specified.

Notes:

1) Once the macro has begun execution you cannot scroll to a particular location, so be sure you know beforehand just where the copied file is to go.

.PUT

Purpose:

To write part of a Fress file into a new file.

Expected environment:

There must be a current file; the file to be created must not already exist.

Macro messages:

- "Enter name of file to be created:"

  Type in the name of the new file that is to contain part of the current file.

- "Enter starting point:"

  Type in the unique character string that identifies the beginning of the section of text to be written to the new file.

- "Enter travel command or null line:"

  Type in a travel command if you must travel to find the endpoint for the copy. If no traveling is necessary, type a null line.

- "Enter ending point:"

  Type in the unique character string that identifies the end of the section of text to be written to the new file.

Notes:

1) To copy an entire file, use the Fress CF (copy file) command.

2) Care should be taken to ensure that the definitions of any edit macros used in the copied text are also copied.

3) If the macro is aborted after a file name has been entered, the new (empty) file will have been created and must be scratched for the .PUT macro to work.

.ICARDS

Purpose:

To insert a card-image file into the current Fress files and perform any conversion necessary to give the file its original appearance when it is printed by the Fress FULLPRINT program.

Expected environment:

There must be a current file; the file to be inserted must be fixed-length, 80 byte format.

Macro messages:

- "Enter card file name:"

  Type in the name of the file to be inserted.

- "Enter card file type:"

  Type in the type of the file to be included.

- "Where should the file be inserted?"

  Type in the unique character string that identifies the location after which the file is to be inserted.

Notes:

1) The .ICARDS macro creates temporary files named $TEMP $FILE and $TEMP FRESS. Make sure you don't have files with the same names.

2) Conversion for files that are not MEMO files proceeds as follows:

   a) The first line inserted in the current file is the definition of an edit macro named !.NL. (for new line). This macro is inserted at the start of every card-image line put in your file. This scheme has been chosen to make the format of the inserted file easy to change.

   b) All blanks in the card image file are translated to literal blanks (C'¬') so that the spacing on the inserted lines is not altered.

c) Trailing blanks on the inserted cards are ignored; serial numbers are also ignored.

d) The Fress converter used by the macro converts all ampersands in your file to the string "!80*". Such strings will still appear as ampersands on printed output.

3) MEMO files are converted as follows:

a) The first line inserted in the current file is the definition of an edit macro named !.SK. (for "skip"). The macro is inserted as a replacement for any blank cards found in the memo file.

b) Blanks are <u>not</u> translated to literal blanks but left as is. This makes editing easier but also means that MEMO files, when printed, will not look the same as they originally did.

TABLE OF CONTENTS