FILE: TMAN2

Compliments of FRESS

A File Retrieval and Editing SyStem

Release 9.1 2 MAY 79

A FRESS Guide to Large Document and Thesis Preparation

Originated by

George M. Stabler

April, 1975

Revised by

Josiah R.W. Strandberg

and

Andries van Dam

January 25, 1919

(Unofficial Version)

1 INTRODUCTION

This manual summarizes the experience gained during the preparation of Ph.D. theses and other large, frequently chanaged documents. Much of what is contained herein is already mentioned in other documents[1] and will be obvious to an experienced user of the system. The aims of this manual are twofold:

(1) To acquaint a less experienced user of the system with some of the more esoteric facilities which are useful in the preparation of large documents;

(2) To illustrate the codes which will produce a Ph.D. thesis format satisfying the standards of Brown's Graduate School[1].

In the next section are described several ways in which the input and editing of a large file can be simplified. The decimal block facility is emphasized as being a convenient means of organizing a file. Section 3 gives the exact format codes needed to format a thesis. The References describe all currently available FRESS manuals.

---

[1]See a list of available documents in the reference section at the end of this guide.
[1]The Masters thesis format varies slightly.

2 USEFUL USE OF FRESS

In general, the way in which FRESS is used to create and edit a file will be very much a function of personal preference. For example, some users will input a large block of text, then add structure to it; others will create the structure and afterwards add the text. Most of the time, individual differences in how FRESS is used do not significantly impact the efficiency or cost of using the system. There are, however, several techniques which have been found to significantly cut down on the time required to create a finished document. Some of these techniques will now be described.

2.1 FILE CREATION AND INPUTING

2.1.1 FILE CREATION

When creating a completely new document, the standard procedure is to issue a Make File command and start inputing text. In many cases, however, the new document will be a considerably revised version of an existing one or will include large sections extracted from an existing document. In these cases, do not overlook the Move and Copy commands, which can be used to move or copy blocks of text from one file to another.

If several files of similar format are to be created, a nice technique is to create a file which consists of a standard set of format macros (see Section 2.5) that will be used to format all the new files. The Copy File command can then be used to simultaneously create a new file and copy into the new file the standard set of format macro definitions. This should be done with a command macro (see section 2.2.2 below). This technique insures a uniform appearance of files, and obviates re-invention and retyping of commonly used formating effects.

2.1.2 TEXT INPUT

When inputing text, the Swift input mode (the SI command) should practically always be used. This mode is considerably faster and cheaper that the normal input mode. The differences in the two input modes are as follows:

(1) In Swift input, no format code verification is done. Thus, if you are inputing many format codes that you want checked immediately, use normal input.

(2) Swift input buffers more text before writing to the file. If CP were to crash in the middle of input, it is possible to lose approximately three times as much input as in normal input mode (i.e., up to three 130 character lines of input as opposed to just one line).

(3) When editing literal text which is part of currently existing structure (e.g., a long explainer for a jump), normal input mode must be used.

(4) Swift input mode cannot be entered by using a null parameter on, for example, the Substitute command. (But command macros can be created which have the same effect. See Section 2.2.2 below.)

(5) &Asis mode cannot be used with swift input (see User's Guide, p. 145).

2.2 EDITING TECHNIQUES

2.2.1 THE UNIFORM SUBSTITUTE COMMAND (US)

A FRESS command of great utility is the Uniform Substitute (US) command. This allows a single editing change to be made throughout a file with one command. Options associated with the command allow selective acceptance or rejection of particular changes and exiting from the command at any point. The uniform substitute command should be used to correct an error discovered after inputing. If you want to delay choosing the exact spelling of a word or the construction of an often

repeated acronym or phrase, use format macros (see Section 2.5).

2.2.2 COMMAND MACROS

For sets of editing commands that will be repeated frequently, do not overlook command macros. This facility, which is described in FRESS NEWSLETTER, Vol. 2, No. 1, March 3, 1976, is used to execute a previously defined list of commands. For example, suppose each editing session were started with the following commands (Comments in the right column surrounded by ' [' and ']' explain the function of the commands and are not part of the text of the macro):

```
g gmst1                 [get the file]
cm adec 7               [see Section 2.4.3 below]
sd 75                   [set display width]
l/end of macros         [scroll over all macro
                         defs]
```

With an appropriate command macro, all these commands are consolidated in a command macro, say '.init'. Then the single command:

.init

would result in execution of all the above commands.

As another example, if one wished to scroll through a file making editing changes by scanning the text on-line rather than from marked-up hardcopy, a command macro (called, for example, ".p") could be created to scroll a specified number of lines or 10 if none specififed and print 10 lines:

```
&1|10|  .               [scroll &1 or 10 lines]
p 10                    [print 10 lines]
```

Each time the ".p" command macro is executed the user specified number of lines (or 10 lines if none specified) would be scrolled and the next 10 lines in the file would be printed. This ".p" macro is particularly useful since it has been determined that it is cheaper to "scroll 10, print 10" through a file, than to use the Type command. The function of the ' .' in the first line is to supress the display of that line of the macro, (see User's guide, p. 32.)

The initialization of FRESS when command macros are used requires inclusion of the appropriate command macro library name. You may wish to consider writing an EXEC file which will create a command to initialize FRESS so you don't forget to include your library.[3] If this is done you can include an initial command macro which would execute commands at the begining of every FRESS session. These might include viewspecs (see, e.g. the FN viewspec mentioned in Section 2.4.1) or display keywords (see Section 2.4.5). The following is a sample of such an EXEC file:

```
&TYPE OFF              [Prevents printing of
                        EXEC lines on execution]
&STACK .START          [Causes ".start" command
                        to be executed after
                        FRESS is initialized]
EXEC FRESS &1 USRLIB   [Enters FRESS (version
                        &1) with macro library
                        "USERLIB"]
```

In this example, the ".start" command macro would include the initial FRESS commands; "USRLIB" would be the name of a command macro library.

2.3 LABELS

A frequently overlooked facility, particularly by new users of FRESS, is that of labels. A label is a means of giving a name to a particular point in a file. Once a point has been "labeled" (with the Make Label command), it can be retrieved with the Get Label command. The value of labels is that they obviate the need for time (and money) consuming scrolling or locating to reach a certain position in a file. For example, in the absence of labels in this file, to get to just after the first sentence of this paragraph from the top of the file one would have to either scroll or do a Locate Long on the string "labels.". However, given a label, created by specifying

```
ml/example/labels.
```

the position could be reached almost instantly with the command:

```
gl/example
```

---

[3] For more on EXEC files see CP-67/CMS User's Guide, available in the manual rack at the computer center.

Labels may be up to 16 characters so make them meaningful; only as many initial characters to uniquely identify the desired label need be specified in the getlabel command.

2.4 USE OF DECIMAL BLOCK STRUCTURE

One of the more useful facilities of FRESS, especially when working on a large file, is the decimal label (and/or block) facility. Use of this facility allows a chunk of text to be subdivided into blocks and subblocks, decimalized labels being automatically assigned and managed by the system. The chief advantages of decimal blocks are as follows:

(1) A natural mechanism is provided for the organization of text into logical sections and subsections.

(2) The movement and reorganization of these blocks is greatly simplified, since the system automatically renumbers the blocks.

(3) The margins and heading levels to be associated with the blocks can easily be redefined.

(4) The "decimal reference" facility (see Section 2.4.4) allows references to a block of text, which are automatically updated so as always to reflect the current decimal number of the block.

We shall now outline these facilities and show some ways in which they can be used.

2.4.1 OVERVIEW OF DECIMAL BLOCK COMMANDS

A complete description of the decimal block facility is given in FRESS Reference Manual [FRM], p. 11. Here we shall briefly note the principal commands used to create and manipulate decimal blocks.

Make Decimal Block (MD). This command is used to create a decimal block when the text to appear in the block already exists. For example, if this clause were to be made into a block, the command

md/if this...block

would result in

%< '2.4.1.1' if this clause were to be made into a block%>

Note that the MD command also allows one to assign normal labels to blocks. This is useful when scanning the structure space (See FRM, p. 25, 28) since the label appears next to the decimal block number to which it is assigned and can be used to "explain" the contents of the block. Furthermore, such normal labels will stay with the block while the decimal labels change with the block's position in the hierarchy.

If decimal blocks are used, decimal labels are, of course, automatically generated. Even so, in a long block it may be desirable to insert further labels. For example, in a large bibliography, it is convenient to label each alphabetic section with, say, labels 'biba', 'bibb', ..., 'bibz'.

Insert Decimal Block (ID). ID is used to LP a particular point at which a new decimal block is to be inserted when the text to be in the block does not already exist. Since the ID command places the LP on the implied insert stack, care must be taken when building a decimal structure using the implied insert point (e.g., with the command "id/"). See the description of the &P command in FRM, p. 14.

Get Decimal Label (GDL). This is the principal means of retrieving a particular decimal block. For example,

gdl/2.4.1

would retrieve and display the top of this block.

The Block Qualifier (-b). When working with decimal blocks, use of the block qualifier on editing commands is occasionally helpful. For example, a complete block (text included) can be moved by specifying:

m-b/%/?

where the "%" is the start of the block and the "?" is a deferred location pointer of the move-to-place. Note that s-b (substitute for block) and d-b (delete block) will get rid of the block as well as the text inside the block.[4]

The FN Viewspec. When working in a decimal block structure, one is frequently presented with a string such

[4]For more on command qualifiers, see User's Guide, p. 41.

as "%>%>%>" which appears at the end of a series of nested blocks. By specifying the Set Viewspec command:

sv/*+fn

this string would appear as "%> '2.4.5' %> '2.4' %> '2'", thus clarifying the display of the block structure for the purpose of LPing a particular block level (e.g., for the insert decimal block (ID) command). Moves and deletes of blocks in the structure space are especially easy with this viewspec on.[5] Note that the setting of viewspecs could be made part of the ".init" command macro illustrated in Section 2.2.2 above. One should consider using command macros to incorporate these FRESS commands into likely contexts. For example, one frequently moves one block to after another. The following command macro could be used:

```
gdl &1                  [gets block to be moved]
m-b/&1/  .              [moves block to deferred
                         lp]
gdl &2                  [gets block after which
                         &1 is to be moved]
j/%  .                  [jumps to block end]
?/>>  .                 [Selects the block end as
                         the deferred lp.]
p 3                     [displays result][6]
```

2.4.2 REDEFINING DECIMAL BLOCK FORMATING

One of the conveniences of decimal blocks is that the system takes over many of the formating tasks involved in the indentation, etc. of the various levels of subblocks. As is noted in FRM, p. 15, there are certain defaults which are used to determine the heading and indent values to be used for each decimal level. Also to be noted is that it is possible to redefine these formating conventions. This is almost always desirable for the following reason: the default for each decimal level is to indent three more spaces than the previous level. Since, in many cases, the top level will be in the form of a chapter heading, it makes little sense to indent the rest of the chapter because it appears on decimal level 2. (e.g., in Sections 2.1, 2.2, etc.).

---

[5]For a description of the Set viewspec command see the FRESS User's Guide, p. 81, and FRM, pp. 29, 30.

[6]The " ." after some of the commands supresses display. See User's Guide, p. 32.

Redefinition of decimal block headings is described in FRM, p. 15. Below is given a sample of one possible set of macros to redefine the indent and decimal heading levels.[7]

```
!.dm0=!+margin0+.
!.dm1=!+margin0+.
!.h1=!-h1-.
!.dm2=!+margin0+.
!.h2=!-h2-.
!.dm3=!+margin0+.
!.h3=!-h3-.
!.dm4=!+margin3+.
!.h4=!-s1-.
!.dm5=!+margin6+.
!.h5=!-s1-.
!.dm6=!+margin9+.
!.h6=!-s1-.
```

Note, that the "m" option of the fullprint command must be used for these redefined headings to take effect. If the paper does not have several chapters then you will probably wish to substitute "!.h1=!-h2-." for "!.h1=!-h1-." to prevent level one blocks from starting a new page.

2.4.3 ALPHABETIC DECIMAL LABELS

A desirable facility not supported in the current FRESS system is the ability to switch from purely numeric decimal labels to labels which are partially alphabetic and partially numeric. For example, the main sections (chapters) of a thesis are typically numbered, while the appendices are lettered. For this reason, a "stopgap" measure is available by which the system can be told to switch from numbers to letters at the top (chapter) level of the hierarchy, starting with a specified decimal number. To do this, the user should enter the FRESS command:

```
cm adec x
```

where 'x' is between 1 and 9 and specifies the top decimal block number at which lettering is to start. For example, the command

---

[7] N.B.: no definition for h0 is possible or needed.

```
cm adec 4
```

would cause the top decimal labels to appear as 1, 2, 3, A, B, C, etc.

Note that this facility is far from being general. Only the top decimal level can be altered, and once in effect, a return to numbers can not be effected. Also, note that the command takes effect only during printouts; while editing a file, the numeric equivalents of the lettered blocks must always be used. (So you don't forget, a fullprint command macro can handle this command.)

2.4.4 REFERENCES TO DECIMAL BLOCKS

Once a decimal block structure has been created, it is possible to insert references to these blocks into the text. This is done with the Make Decimal Reference command which is fully described in FRM, p. 16. For example, the decimal reference to this section in Section 2.4, "(see Section 2.4.4)" was created by the command:

```
mdr/Section /2.4.4
```

where "(see Section)" had been previously typed in. Once a reference is created, it will always refer to the same block, regardless of whether or not the actual number of the block changes. For example, if another block were to be inserted between the blocks currently numbered 2.4.3 and 2.4.4, the decimal reference in 2.4 to this block would automatically be changed to point to 2.4.5.

Two other notes should be made about the behavior of decimal references. First, a decimal reference will reflect the alphabetic numbering described in Section 2.4.3 on a printout, but not during on-line display. Second, if a printout contains a decimal reference to a block which is not printed (as, for example, could occur with the file described in the next section), the decimal reference will be printed as '1.999'. Once the block referred to can be "seen" again by the reference, the reference number will revert back to the correct format.

2.4.5 A SAMPLE FILE STRUCTURE

For many large files, it would obviously be convenient to separate the major sections into separate files. With limited disk or segment space, this allows all but the section currently being edited to be stored on some secondary storage device. When the whole document is to be printed, the separate files can be either combined into one, using move or copy, or they can be spliced together (see the description of splices in Section 2.5 of FRM, p. 6).

There are at least four problems with keeping a large document in separate files (how important these are is a function of personal preferences in structuring the document):

(1) Structure (blocks, labels, decimal references, etc.) cannot be moved between files. Thus, if two or more files are to be combined, the structure in all but one of the files must be recreated.

(2) Interfile decimal references are not supported. This is an outgrowth of the previous problem.

(3) If a common set of macros is used for the various sections of a document, the macros must be included in each file to avoid multitudinous "UNDEFINED MACRO" messages. This can, however, be circumvented by using the command "fmsg off".

(4) Printing the entire file for each revision even though the revision is only of a part of the document is wasteful both of computer time (money) and paper. However, when printing a section, it is nice to have the section (and subsections) numbered as they will be in the final copy when, presumably, the section will appear in the midst of several other sections.

For these reasons, the following file structure is recommended as being a convenient means to circumvent all of the above problems. It takes a few minutes to set up initially, but this initial investment pays off greatly once serious work starts.

The file illustrated uses display keywords (See FRESS NEWSLETTER, Vol. 2, No. 2, November 8, 1976, "FRESS Display Keywords").

```
[standard format macros]
!.h1=!-s1-.
```

```
%<$1;all$!.h1=!-h1-.%>
%< '1'
%<$1;all$text  in  block 1,  including nested blocks
%>%> '1'
!.h1=!-s1-.
%<$2;all$!.h1=!-h1-.%>
%< '2'
%<$2;all$text  in  block 2,  including nested blocks
%>%> '2'
     .
     .
     .
```

The outer blocks are decimal blocks which, having no keywords, will always be included in a printout. Inside these are display keyworded blocks that control whether the text in the decimal block will be printed. The keywords strings are surrounded by "$", each keyword is spearated by ";". Thus, specifying:

```
SKD/2
```

will cause only Section 2 text to be printed when the Fullprint command is issued. However, since the decimal block 1 will be printed, without the text inside it, Section 2 will still be numbered 2. Similarly, to print sections 2 and 4, the command would be:

```
SKD/2|4
```

To print all blocks and text, you would specify:

```
SKD/all
```

Although this example showed the method for printing only specified sections at the highest "chapter" level, it can be extended to control the printing and numbering of subsections as well.

In this example, there is a macro which temporarily redefines the heading level given the first decimal level. Heading level 1 is redefined to be a !-s1-. The purpose of this is to avoid many blank pages when printing just one section of the file. This format macro is followed by another which redefines heading level 1 as !-h1-. This latter macro is inside a block that is keyworded the same as the text of the block that follows it. Thus when that block is printed, level 1 blocks will start a new page. The "m" option of the fullprint command must be used for these to take effect.

2.5 FORMAT MACROS

As with labels, a new user may overlook the format macro facility (User's Guide, p. 56). This is particularly unfortunate since in a large file practically all formating should be done using macros. In brief, a macro lets a user give a "name" to an arbitrary string of characters. Once the name has been defined, anywhere it appears in a file it will be replaced by the string of characters. (The replacement does not actually occur until the file is printed.) The value of this can be shown by an example.

Suppose a file contains several hundred formulae, and each formula is to appear indented 10 on a new line as follows:

```
!-i10-X = Y
```

Now suppose that, after the entire file had been input, it was found that the formulae were required to be centered, rather than just indented. This would require changing every one of the several hundred format codes. Had macros been used, however, only one change would be required. In this case, a macro "form" would have first been defined as follows:

```
!.form=!-i10-.
```

Each formula would use this macro rather than the format code:

```
!.form.X = Y
```

To then center all the formulae, all that is required is to redefine 'form':

```
!.form=!-c-.
```

This would result in the substitution of a center code, rather than the indent code, in front of all formulae. Several hundred edits have been replaced by just one.

Note that the use of macros is not limited to format codes. Any string which occurs frequently or is likely to change can be represented by a macro. For example, if the implementers of FRESS decided to change the name of the macro facility to "troll", this section could reflect the update with a single change, since each time the word "macro" is used, it is represented by a macro, !.macro., which was defined as:

```
!.macro=macro.
```

To change "macro" to "troll" all that is required is to redefine the "macro" macro as follows:

```
!.macro=troll.
```

As another example, if a user frequently makes numbered points, as in Section 2.1.2 above, formating macros similar to the following should be included in each file:

```
!.n1=!-s1;i2;j6-(1)¬.
!.n2=!-s1;i2;j6-(2)¬.
!.n3=!-s1;i2;j6-(3)¬.
!.n4=!-s1;i2;j6-(4)¬.
!.n5=!-s1;i2;j6-(5)¬.
!.n6=!-s1;i2;j6-(6)¬.
!.n7=!-s1;i2;j6-(7)¬.
```

Some further notes on the use of macros: first, macro names should be short and mnemonic. Second, all macro definitions should be copied from a standard file to avoid retyping each time a new file is created. Third, they should be kept together at the top of the file so they will be "seen" when the file is retrieved. (If a macro definition has not been scrolled over, its use will result in "UNDEFINED MACRO" messages. One way to avoid this problem is illustrated in the ".init" command macro in Section 2.2.2 above. That command macro includes a scroll over all the format macros.)

3 THESIS FORMAT

General instructions for formating a thesis are given in the document "Instructions for the Preparation and Presentation of Theses for Advanced Degrees" (available from the Graduate School). In this section we will show how the guidelines can be implemented in FRESS. Note that small variations in the EDIT and ALTER codes shown may be possible. Also note that the requirements for Master's theses vary slightly from those for Ph.D. theses.

3.1 THESIS SPACING AND MARGINS

The final printout of the thesis will be xeroxed onto regulation thesis paper, and the margins must be set so as to position all text (and page numbers) within the red borders of the thesis paper. In FRESS, the appropriate margins can be set up by the following code

```
!+margin,,7+
```

which should precede the title page.

The page numbers for the thesis must appear at the top center or right hand corner of the page. This is achieved by the ALTER code:

```
!+titlep=5++
```

to center the pages numbers, or:

```
!+titlep=5+**:+
```

to place the page numbers in the right hand corner.

The title page, approval page, vita, acknowledgments, etc. are numbered with Roman numerals i, ii, etc. (The title page is 'i', but the page number should not appear.) To force Roman numbering, place a:

```
!-n2r-
```

at the bottom (or end of text) of the title page. To revert to normal numbering, place a

```
!-n1-
```

just before the first page of the thesis proper.

3.2 FOOTNOTES

Footnotes in FRESS are automatically numbered and placed at the bottom of the page on which they occur. Note that separate numbering for each section or chapter can be forced by specifying

!-f1-

on the first footnote of each new section.

There are two formating errors which FRESS may occasionally introduce when trying to position footnotes on a printout. First, if reference to a footnote appears on the last line of a page, the footnote may be forced to appear on the following page. Also, a footnote reference on the top line of a page may have the footnote text appear at the bottom of the previous page. Since these two idiosyncracies are completely dependent on the relationship between the text and where it appears on the physical page when printed, attempts to fix them should be put off until the final copy of the file is to be printed. At that time, new-page codes (!-n-) should be inserted a line or so before the footnote reference to force the footnote onto the correct page. (This problem can be solved in some cases with the WIDOW code -- see User's Guide, p. 56.)

3.3 TABLE OF CONTENTS NUMBERING

It will be noted that on a normal printout, the table of contents is numbered starting with '-i-'. In many cases, however, the table of contents will not directly follow the title page, and it is desirable to start the the table of contents with another roman numeral. This capability is not supported by the current version of FRESS, but the following command can be used to produce the same effect:

cm tofc x

This command should be entered just prior to the full-print in which the table of contents is to be renumbered. (So you don't forget, a fullprint command macro can handle this, as well as the "adec" command (see Section 2.4.3).) 'x' is a number between 1 and 9 which specifies the roman numeral at which numbering is to start. For example, '6' would cause

the first page of the table of contents to be numbered "-vi-".

To single space the table of contents, a !+space1+ should be at the bottom of the file.

3.4 REFERENCES AND BIBLIOGRAPHY

Entries in either of these sections are usually single-spaced, so remember to include a:

!+space1+

before the section (and a !+space2+ after the section if more text is to follow). It is convenient to precede each entry by a standard macro so that the format can be changed easily. A possible macro which "looks good" is:

!.ref=!-s1;j4-.

It is also convenient (for moving and inserting references) to end each reference with a macro. For example, the macro:

!.annot=.

will not generate anything, but provides a convenient string for LPs and simplifies the later insertion of subsequent entries or an annotation into an annoted bibliography. (Annotations may, of course, be included in the thesis to begin with.)

3.5 GRAPHS AND DRAWINGS

Note the restrictions noted in the Graduate School's guidelines. In particular, remember to leave a completely blank (numbered) page in the text for full-page insertions. A blank page can be left by:

!-n-¬!-n-

The '¬' is a special blank to prevent FRESS from just skipping to the top of the next page, thus not leaving a completely blank page.

For geometric diagrams and graphs, consider using the BOX and GRID codes. User's Guide, pp. 68-78, shows how these can be used to construct diagrams in FRESS, thus not requiring

later hand-drawn insertions. The advantage of making them with FRESS codes is that they can be moved around with their surrounding text and need no separate reproduction and pasting in; they do take some time to master and aren't worth it if you're only doing a couple of drawings.

3.6 SAMPLE PAGES I AND II

The following illustrates one way in which the first two pages of a thesis can be formated. Also shown are all the ALTER codes necessary to format a thesis correctly. When an EDIT code itself will not fit on the line that it effects in this document, it is shown on the immediately preceding line. Comments surrounded by ' [' and ']' explain the function of the preceding code(s) and are not part of the text to be specified by the user.

```
*START OF TEXT AREA*
!+tofc1=0;2=3;3=5+      [Indents and heading levels for the
                         table of contents.]
!+titlep=5c++           [Center page numbers 5 lines from top
                         of page.]
!+bu+                   [Underline blanks between underscore
                         codes: !(0...!).]
!+margin,,7+            [Set up top margin]
!+width63+              [Set up correct width for thesis
                         paper.]
!+para0,1+              [Define '!-p-' function -- skip 1
                         line, indent 0 spaces (personal
                         preference).]
!+space1+               [Single space for first two pages
                         (and vita).]
```

!-n0;s8;c-A System for Interconnected Processing

!-s8;c-by

!-s3;c-George Merritt Stabler

!-s1;c-A.B., Harvard College, 1965
!-s0;c-M.Sc., University of Manchester, 1967

!-s4;c-Thesis

!-s2;c-
Submitted in partial fulfillment of the requirements for the

!-s2;c-Degree of Doctor of Philosophy

!-s2;c-
in the Division of Applied Mathematics at Brown University

!-s2;c-October, 1974

!-s5;c-This thesis by George Merritt Stabler

!-s1;c-is accepted in its present form by the

!-s1;c-Division of Applied Mathematics

!-s1;c-as satisfying the thesis requirement for the degree of

!-s1;c-Doctor of Philosophy

!-s2;i7-Date:¬!-t*32r;75- !-t*37-!-t*63r;75-!75
Date: ..... .

!-s5;c-Recommended to the Graduate Council

!-s2;i7-Date:¬!-t*32r;75-.....!-t*37-!-t*63r;75-!75
Date: ..... .

.....!-t*37-!-t*63r;75-!75
.

!-s2;i7-Date:¬!-t*32r;75-.....!-t*37-!-t*63r;75-!75
Date: ..... .
.....!-t*37-!-t*63r;75-!75
.

!-s4;c-Approved by the Graduate Council

!-s2;i7-Date:¬!-t*32r;75-.....!-t*37-!-t*63r;75-!75
Date: ..... .

4 REFERENCES

FRESS Concepts and Facilities for the Layman (revised November 19, 1976)

An introduction to FRESS. It should be read before other FRESS manuals by those unfamiliar with the system (or unacquainted with text-editing systems in general).

FRESS User's Guide (August 6, 1975)

Describes the basic use of FRESS, including commmand specification, inputing, editing, and formating. A complete list of format codes and special characters is given; no structure is discussed.

FRESS Reference Manual -- Structure and Commands (2nd printing, January 15, 1976) [FRM]

Contains a complete list of FRESS commands and describes the creation and use of structure (labels, jumps, decimal blocks, etc.).

FRESS NEWSLETTER, Vol. 2, No. 1, March 3, 1976, FRESS Command Macros

Contains a complete description of the Command Macro facility.

FRESS NEWSLETTER, Vol. 2, No. 2, November 8, 1976, FRESS Display Keywords

Contains a complete description of the Display Keyword facility.

All of these documents are available in Computer Center document racks. The Reference Manual, and the User's Guide can be purchased in the Brown Bookstore. The Newsletters are available for printing online. For information type the FRESS command ".frsnews".

TABLE OF CONTENTS