WRITING PAPERS WITH NROFF USING −ME

Eric P. Allman

Electronics Research Laboratory
University of California, Berkeley
Berkeley, California 94720

This document describes the text processing facilities available on the UNIX† operating system via NROFF† and the −me macro package. It is assumed that the reader already is generally familiar with the UNIX operating system and a text editor such as **ex**. This is intended to be a casual introduction, and as such not all material is covered. In particular, many variations and additional features of the −me macro package are not explained. For a complete discussion of this and other issues, see *The −me Reference Manual* and *The NROFF/TROFF Reference Manual.*

NROFF, a computer program that runs on the UNIX operating system, reads an input file prepared by the user and outputs a formatted paper suitable for publication or framing. The input consists of *text*, or words to be printed, and *requests*, which give instructions to the NROFF program telling how to format the printed copy.

Section 1 describes the basics of text processing. Section 2 describes the basic requests. Section 3 introduces displays. Annotations, such as footnotes, are handled in section 4. The more complex requests which are not discussed in section 2 are covered in section 5. Finally, section 6 discusses things you will need to know if you want to typeset documents. If you are a novice, you probably won't want to read beyond section 4 until you have tried some of the basic features out.

When you have your raw text ready, call the NROFF formatter by typing as a request to the UNIX shell:

nroff −me −T*type files*

where *type* describes the type of terminal you are outputting to. Common values are **dtc** for a DTC 300s (daisy-wheel type) printer and **lpr** for the line printer. If the **−T** flag is omitted, a "lowest common denominator" terminal is assumed; this is good for previewing output on most terminals. A complete description of options to the NROFF command can be found in *The NROFF/TROFF Reference Manual.*

The word *argument* is used in this manual to mean a word or number which appears on the same line as a request which modifies the meaning of that request. For example, the request

```
.sp
```

spaces one line, but

```
.sp 4
```

spaces four lines. The number **4** is an *argument* to the **.sp** request which says to space four lines

†UNIX, NROFF, and TROFF are Trademarks of Bell Laboratories

instead of one. Arguments are separated from the request and from each other by spaces.

1. Basics of Text Processing

The primary function of NROFF is to *collect* words from input lines, *fill* output lines with those words, *justify* the right hand margin by inserting extra spaces in the line, and output the result. For example, the input:

```
Now is the time
for all good men
to come to the aid
of their party.
Four score and seven
years ago,...
```

will be read, packed onto output lines, and justified to produce:

```
Now is the time for all good men to come to the aid of their party. Four
score and seven years ago,...
```

Sometimes you may want to start a new output line even though the line you are on is not yet full; for example, at the end of a paragraph. To do this you can cause a *break*, which starts a new output line. Some requests cause a break automatically, as do blank input lines and input lines beginning with a space.

Not all input lines are text to be formatted. Some of the input lines are *requests* which describe how to format the text. Requests always have a period or an apostrophe ("´") as the first character of the input line.

The text formatter also does more complex things, such as automatically numbering pages, skipping over page folds, putting footnotes in the correct place, and so forth.

I can offer you a few hints for preparing text for input to NROFF. First, keep the input lines short. Short input lines are easier to edit, and NROFF will pack words onto longer lines for you anyhow. In keeping with this, it is helpful to begin a new line after every period, comma, or phrase, since common corrections are to add or delete sentences or phrases. Second, do not put spaces at the end of lines, since this can sometimes confuse the NROFF processor. Third, do not hyphenate words at the end of lines (except words that should have hyphens in them, such as "mother-in-law"); NROFF is smart enough to hyphenate words for you as needed, but is not smart enough to take hyphens out and join a word back together. Also, words such as "mother-in-law" should not be broken over a line, since then you will get a space where not wanted, such as "mother- in-law".

2. Basic Requests

2.1. Paragraphs

Paragraphs are begun by using the **.pp** request. For example, the input:

```
.pp
Now is the time for all good men
to come to the aid of their party.
Four score and seven years ago,...
```

produces a blank line followed by an indented first line. The result is:

```
     Now is the time for all good men to come to the aid of their party.
Four score and seven years ago,...
```

Notice that the sentences of the paragraphs *must not* begin with a space, since blank lines and lines begining with spaces cause a break. For example, if I had typed:

```
.pp
Now is the time for all good men
    to come to the aid of their party.
Four score and seven years ago,...
```

The output would be:

```
 Now is the time for all good men
to come to the aid of their party.  Four score and seven years ago,...
```

A new line begins after the word “men” because the second line began with a space character.

There are many fancier types of paragraphs, which will be described later.

2.2. Headers and Footers

Arbitrary headers and footers can be put at the top and bottom of every page. Two requests of the form **.he** *title* and **.fo** *title* define the titles to put at the head and the foot of every page, respectively. The titles are called *three-part* titles, that is, there is a left-justified part, a centered part, and a right-justified part. To separate these three parts the first character of *title* (whatever it may be) is used as a delimiter. Any character may be used, but backslash and double quote marks should be avoided. The percent sign is replaced by the current page number whenever found in the title. For example, the input:

```
.he ´´%´´
.fo ´Jane Jones´´My Book´
```

results in the page number centered at the top of each page, “Jane Jones” in the lower left corner, and “My Book” in the lower right corner.

2.3. Double Spacing

NROFF will double space output text automatically if you use the request **.ls 2**, as is

done in this section. You can revert to single spaced mode by typing **.ls 1**.

2.4. Page Layout

A number of requests allow you to change the way the printed copy looks, sometimes called the *layout* of the output page. Most of these requests adjust the placing of “white space” (blank lines or spaces). In these explanations, characters in italics should be replaced with values you wish to use; bold characters represent characters which should actually be typed.

The **.bp** request starts a new page.

The request **.sp** *N* leaves *N* lines of blank space. *N* can be omitted (meaning skip a single line) or can be of the form *N***i** (for *N* inches) or *N***c** (for *N* centimeters). For example, the input:

```
.sp 1.5i
My thoughts on the subject
.sp
```

leaves one and a half inches of space, followed by the line “My thoughts on the subject”, followed by a single blank line.

The **.in** *+N* request changes the amount of white space on the left of the page (the *indent*). The argument *N* can be of the form $+N$ (meaning leave *N* spaces more than you are already leaving), $-N$ (meaning leave less than you do now), or just *N* (meaning leave exactly *N* spaces). *N* can be of the form *N***i** or *N***c** also. For example, the input:

```
initial text
.in 5
some text
.in +1i
more text
.in −2c
final text
```

produces “some text” indented exactly five spaces from the left margin, “more text” indented five spaces plus one inch from the left margin (fifteen spaces on a pica typewriter), and “final text” indented five spaces plus one inch minus two centimeters from the margin. That is, the output is:

```
initial text
          some text
                         more text
             final text
```

The **.ti** *+N* (temporary indent) request is used like **.in** *+N* when the indent should apply to one line only, after which it should revert to the previous indent. For example, the input:

```
.in 1i
.ti 0
Ware, James R.  The Best of Confucius,
Halcyon House, 1950.
An excellent book containing translations of
most of Confucius´ most delightful sayings.
A definite must for anyone interested in the early foundations
of Chinese philosophy.
```

produces:

```
Ware, James R.  The Best of Confucius, Halcyon House, 1950.  An excellent book contain-
            ing translations of most of Confucius’ most delightful sayings.  A definite
            must for anyone interested in the early foundations of Chinese philosophy.
```

Text lines can be centered by using the **.ce** request. The line after the **.ce** is centered (horizontally) on the page. To center more than one line, use **.ce** *N* (where *N* is the number of lines to center), followed by the *N* lines. If you want to center many lines but don’t want to count them, type:

```
.ce 1000
lines to center
.ce 0
```

The **.ce 0** request tells NROFF to center zero more lines, in other words, stop centering.

All of these requests cause a break; that is, they always start a new line. If you want to start a new line without performing any other action, use **.br**.

2.5. Underlining

Text can be underlined using the **.ul** request. The **.ul** request causes the next input line to be underlined when output. You can underline multiple lines by stating a count of *input* lines to underline, followed by those lines (as with the **.ce** request). For example, the input:

```
.ul 2
Notice that these two input lines
are underlined.
```

will underline those eight words in NROFF. (In TROFF they will be set in italics.)

3. Displays

Displays are sections of text to be set off from the body of the paper. Major quotes, tables, and figures are types of displays, as are all the examples used in this document. All displays except centered blocks are output single spaced.

3.1. Major Quotes

Major quotes are quotes which are several lines long, and hence are set in from the rest of the text without quote marks around them. These can be generated using the commands **.(q** and **.)q** to surround the quote. For example, the input:

```
As Weizenbaum points out:
.(q
It is said that to explain is to explain away.
This maxim is nowhere so well fulfilled
as in the areas of computer programming,...
.)q
```

generates as output:

As Weizenbaum points out:

> It is said that to explain is to explain away. This maxim is nowhere so well fulfilled as in the areas of computer programming,...

3.2. Lists

A *list* is an indented, single spaced, unfilled display. Lists should be used when the material to be printed should not be filled and justified like normal text, such as columns of figures or the examples used in this paper. Lists are surrounded by the requests **.(l** and **.)l**. For example, type:

```
Alternatives to avoid deadlock are:
.(l
Lock in a specified order
Detect deadlock and back out one process
Lock all resources needed before proceeding
.)l
```

will produce:
Alternatives to avoid deadlock are:

```
Lock in a specified order
Detect deadlock and back out one process
Lock all resources needed before proceeding
```

3.3. Keeps

A *keep* is a display of lines which are kept on a single page if possible. An example of where you would use a keep might be a diagram. Keeps differ from lists in that lists may be broken over a page boundary whereas keeps will not.

Blocks are the basic kind of keep. They begin with the request **.(b** and end with the request **.)b**. If there is not room on the current page for everything in the block, a new page is begun. This has the unpleasant effect of leaving blank space at the bottom of the page. When this is not appropriate, you can use the alternative, called *floating keeps.*

Floating keeps move relative to the text. Hence, they are good for things which will be referred to by name, such as "See figure 3". A floating keep will appear at the bottom of the current page if it will fit; otherwise, it will appear at the top of the next page. Floating keeps begin with the line **.(z** and end with the line **.)z**. For an example of a floating keep, see figure 1. The **.hl** request is used to draw a horizontal line so that the figure stands out from the text.

3.4. Fancier Displays

Keeps and lists are normally collected in *nofill* mode, so that they are good for tables and such. If you want a display in fill mode (for text), type **.(l F** (Throughout this section, comments applied to **.(l** also apply to **.(b** and **.(z**). This kind of display will be indented from both margins. For example, the input:

```
.(l F
And now boys and girls,
a newer, bigger, better toy than ever before!
Be the first on your block to have your own computer!
Yes kids, you too can have one of these modern
data processing devices.
You too can produce beautifully formatted papers
without even batting an eye!
.)l
```

will be output as:

> And now boys and girls, a newer, bigger, better toy than ever before! Be the first on your block to have your own computer! Yes kids, you too can have one of these modern data processing devices. You too can produce beautifully formatted papers without even batting an eye!

Lists and blocks are also normally indented (floating keeps are normally left justified). To get a left-justified list, type **.(l L**. To get a list centered line-for-line, type **.(l C**. For example, to get a filled, left justified list, enter:

```
.(z
.hl
Text of keep to be floated.
.sp
.ce
Figure 1.  Example of a Floating Keep.
.hl
.)z
```

Figure 1. Example of a Floating Keep.

```
.(l L F
text of block
.)l
```

The input:

```
.(l
first line of unfilled display
more lines
.)l
```

produces the indented text:

```
        first line of unfilled display
        more lines
```

Typing the character **L** after the **.(l** request produces the left justified result:

```
first line of unfilled display
more lines
```

Using **C** instead of **L** produces the line-at-a-time centered output:

```
                              first line of unfilled display
                                      more lines
```

Sometimes it may be that you want to center several lines as a group, rather than centering them one line at a time. To do this use centered blocks, which are surrounded by the requests **.(c** and **.)c**. All the lines are centered as a unit, such that the longest line is centered and the rest are lined up around that line. Notice that lines do not move relative to each other using centered blocks, whereas they do using the **C** argument to keeps.

Centered blocks are *not* keeps, and may be used in conjunction with keeps. For example, to center a group of lines as a unit and keep them on one page, use:

```
.(b L
.(c
first line of unfilled display
more lines
.)c
.)b
```

to produce:

```
                         first line of unfilled display
                         more lines
```

If the block requests (**.(b** and **.)b**) had been omitted the result would have been the same, but with no guarantee that the lines of the centered block would have all been on one page. Note the use of the **L** argument to **.(b**; this causes the centered block to center within the entire line rather than within the line minus the indent. Also, the center requests must be nested *inside* the keep requests.

4. Annotations

There are a number of requests to save text for later printing. *Footnotes* are printed at the bottom of the current page. *Delayed text* is intended to be a variant form of footnote; the text is printed only when explicitly called for, such as at the end of each chapter. *Indexes* are a type of delayed text having a tag (usually the page number) attached to each entry after a row of dots. Indexes are also saved until called for explicitly.

4.1. Footnotes

Footnotes begin with the request **.(f** and end with the request **.)f**. The current footnote number is maintained automatically, and can be used by typing **, to produce a footnote number[1]. The number is automatically incremented after every footnote. For example, the input:

```
.(q
A man who is not upright
and at the same time is presumptuous;
one who is not diligent and at the same time is ignorant;
one who is untruthful and at the same time is incompetent;
such men I do not count among acquaintances.\**
.(f
\**James R. Ware,
.ul
The Best of Confucius,
Halcyon House, 1950.
Page 77.
.)f
.)q
```

generates the result:

> A man who is not upright and at the same time is presumptuous; one who is not diligent and at the same time is ignorant; one who is untruthful and at the same time is incompetent; such men I do not count among acquaintances.[2]

It is important that the footnote appears *inside* the quote, so that you can be sure that the footnote will appear on the same page as the quote.

4.2. Delayed Text

Delayed text is very similar to a footnote except that it is printed when called for explicitly. This allows a list of references to appear (for example) at the end of each chapter, as is the convention in some disciplines. Use ***#** on delayed text instead of ****** as on footnotes.

If you are using delayed text as your standard reference mechanism, you can still use footnotes, except that you may want to reference them with special characters* rather than numbers.

4.3. Indexes

An "index" (actually more like a table of contents, since the entries are not sorted alphabetically) resembles delayed text, in that it is saved until called for. However, each entry has the page number (or some other tag) appended to the last line of the index entry after a row of dots.

Index entries begin with the request **.(x** and end with **.)x**. The **.)x** request may have a argument, which is the value to print as the "page number". It defaults to the current page number. If the page number given is an underscore ("_") no page number or line of dots is printed at all. To get the line of dots without a page number, type **.)x ""**, which

[1]Like this.

[2]James R. Ware, *The Best of Confucius,* Halcyon House, 1950. Page 77.

*Such as an asterisk.

specifies an explicitly null page number.

The **.xp** request prints the index.

For example, the input:

```
.(x
Sealing wax
.)x
.(x
Cabbages and kings
.)x _
.(x
Why the sea is boiling hot
.)x 2.5a
.(x
Whether pigs have wings
.)x ""
.(x
This is a terribly long index entry, such as might be used
for a list of illustrations, tables, or figures; I expect it to
take at least two lines.
.)x
.xp
```

generates:

```
Sealing wax ........................................................................ 9
Cabbages and kings
Why the sea is boiling hot ...................................................... 2.5a
Whether pigs have wings .........................................................
This is a terribly long index entry, such as might be used for a list of illustra-
      tions, tables, or figures; I expect it to take at least two lines. .......... 9
```

The **.(x** request may have a single character argument, specifying the “name” of the index; the normal index is **x**. Thus, several “indicies” may be maintained simultaneously (such as a list of tables, table of contents, etc.).

Notice that the index must be printed at the *end* of the paper, rather than at the beginning where it will probably appear (as a table of contents); the pages may have to be physically rearranged after printing.

5. Fancier Features

A large number of fancier requests exist, notably requests to provide other sorts of paragraphs, numbered sections of the form **1.2.3** (such as used in this document), and multicolumn output.

5.1. More Paragraphs

Paragraphs generally start with a blank line and with the first line indented. It is possible to get left-justified block-style paragraphs by using **.lp** instead of **.pp**, as demonstrated by the next paragraph.

Sometimes you want to use paragraphs that have the *body* indented, and the first line exdented (opposite of indented) with a label. This can be done with the **.ip** request. A word specified on the same line as **.ip** is printed in the margin, and the body is lined up at a prespecified position (normally five spaces). For example, the input:

```
.ip one
This is the first paragraph.
Notice how the first line
of the resulting paragraph lines up
with the other lines in the paragraph.
.ip two
And here we are at the second paragraph already.
You may notice that the argument to .ip
appears
in the margin.
.lp
We can continue text...
```

produces as output:

one This is the first paragraph. Notice how the first line of the resulting paragraph lines up with the other lines in the paragraph.

two And here we are at the second paragraph already. You may notice that the argument to **.ip** appears in the margin.

We can continue text without starting a new indented paragraph by using the **.lp** request.

If you have spaces in the label of a **.ip** request, you must use an "unpaddable space" instead of a regular space. This is typed as a backslash character ("\") followed by a space. For example, to print the label "Part 1", enter:

```
.ip "Part\ 1"
```

If a label of an indented paragraph (that is, the argument to **.ip**) is longer than the space allocated for the label, **.ip** will begin a new line after the label. For example, the input:

```
.ip longlabel
This paragraph had a long label.
The first character of text on the first line
will not line up with the text on second and subsequent lines,
although they will line up with each other.
```

will produce:

longlabel
This paragraph had a long label. The first character of text on the first line will not line up with the text on second and subsequent lines, although they will line up with each other.

It is possible to change the size of the label by using a second argument which is the size of the label. For example, the above example could be done correctly by saying:

```
.ip longlabel 10
```

which will make the paragraph indent 10 spaces for this paragraph only. If you have many paragraphs to indent all the same amount, use the *number register* **ii**. For example, to leave one inch of space for the label, type:

```
.nr ii 1i
```

somewhere before the first call to **.ip**. Refer to the reference manual for more information.

If **.ip** is used with no argument at all no hanging tag will be printed. For example, the input:

```
.ip [a]
This is the first paragraph of the example.
We have seen this sort of example before.
.ip
This paragraph is lined up with the previous paragraph,
but it has no tag in the margin.
```

produces as output:

[a] This is the first paragraph of the example. We have seen this sort of example before.

This paragraph is lined up with the previous paragraph, but it has no tag in the margin.

A special case of **.ip** is **.np**, which automatically numbers paragraphs sequentially from 1. The numbering is reset at the next **.pp**, **.lp**, or **.sh** (to be described in the next section) request. For example, the input:

```
.np
This is the first point.
.np
This is the second point.
Points are just regular paragraphs
which are given sequence numbers automatically
by the .np request.
.pp
This paragraph will reset numbering by .np.
.np
For example,
we have reverted to numbering from one now.
```

generates:

(1) This is the first point.

(2) This is the second point. Points are just regular paragraphs which are given sequence numbers automatically by the .np request.

This paragraph will reset numbering by .np.

(1) For example, we have reverted to numbering from one now.

5.2. Section Headings

Section numbers (such as the ones used in this document) can be automatically generated using the **.sh** request. You must tell **.sh** the *depth* of the section number and a section title. The depth specifies how many numbers are to appear (separated by decimal points) in the section number. For example, the section number **4.2.5** has a depth of three.

Section numbers are incremented in a fairly intuitive fashion. If you add a number (increase the depth), the new number starts out at one. If you subtract section numbers (or keep the same number) the final number is incremented. For example, the input:

```
.sh 1 "The Preprocessor"
.sh 2 "Basic Concepts"
.sh 2 "Control Inputs"
.sh 3
.sh 3
.sh 1 "Code Generation"
.sh 3
```

produces as output the result:

1. The Preprocessor
1.1. Basic Concepts
1.2. Control Inputs
1.2.1.
1.2.2.
2. Code Generation
2.1.1.

You can specify the section number to begin by placing the section number after the section title, using spaces instead of dots. For example, the request:

```
.sh 3 "Another section" 7 3 4
```

will begin the section numbered **7.3.4**; all subsequent **.sh** requests will number relative to this number.

There are more complex features which will cause each section to be indented proportionally to the depth of the section. For example, if you enter:

.nr si *N*

each section will be indented by an amount *N*. *N* must have a scaling factor attached, that is, it must be of the form *Nx*, where *x* is a character telling what units *N* is in. Common values for *x* are **i** for inches, **c** for centimeters, and **n** for *ens* (the width of a single character). For example, to indent each section one-half inch, type:

```
.nr si 0.5i
```

After this, sections will be indented by one-half inch per level of depth in the section number. For example, this document was produced using the request

```
.nr si 3n
```

at the beginning of the input file, giving three spaces of indent per section depth.

Section headers without automatically generated numbers can be done using:

```
.uh "Title"
```

which will do a section heading, but will put no number on the section.

5.3. Parts of the Basic Paper

There are some requests which assist in setting up papers. The **.tp** request initializes for a title page. There are no headers or footers on a title page, and unlike other pages you can space down and leave blank space at the top. For example, a typical title page might appear as:

```
.tp
.sp 2i
.(l C
THE GROWTH OF TOENAILS
IN UPPER PRIMATES
.sp
by
.sp
Frank N. Furter
.)l
.bp
```

The request **.th** sets up the environment of the NROFF processor to do a thesis, using the rules established at Berkeley. It defines the correct headers and footers (a page number in the upper right hand corner only), sets the margins correctly, and double spaces.

The **.+c** *T* request can be used to start chapters. Each chapter is automatically numbered from one, and a heading is printed at the top of each chapter with the chapter number and the chapter name *T*. For example, to begin a chapter called "Conclusions", use the request:

```
.+c "CONCLUSIONS"
```

which will produce, on a new page, the lines

```
CHAPTER 5
CONCLUSIONS
```

with appropriate spacing for a thesis. Also, the header is moved to the foot of the page on the first page of a chapter. Although the **.+c** request was not designed to work only with the **.th** request, it is tuned for the format acceptable for a PhD thesis at Berkeley.

If the title parameter *T* is omitted from the **.+c** request, the result is a chapter with no heading. This can also be used at the beginning of a paper; for example, **.+c** was used to generate page one of this document.

Although papers traditionally have the abstract, table of contents, and so forth at the front of the paper, it is more convenient to format and print them last when using NROFF. This is so that index entries can be collected and then printed for the table of contents (or whatever). At the end of the paper, issue the **.++ P** request, which begins the preliminary part of the paper. After issuing this request, the **.+c** request will begin a preliminary section of the paper. Most notably, this prints the page number restarted from one in lower case Roman numbers. **.+c** may be used repeatedly to begin different parts of the front material for example, the abstract, the table of contents, acknowledgments, list of illustrations, etc. The request **.++ B** may also be used to begin the bibliographic section at the end of the paper. For example, the paper might appear as outlined in figure 2. (In this figure, comments begin with the sequence \".)

5.4. Equations and Tables

Two special UNIX programs exist to format special types of material. **Eqn** and **neqn** set equations for the phototypesetter and NROFF respectively. **Tbl** arranges to print extremely pretty tables in a variety of formats. This document will only describe the embellishments to the standard features; consult the reference manuals for those processors for a description of their use.

The **eqn** and **neqn** programs are described fully in the document *Typesetting Mathematics – Users' Guide* by Brian W. Kernighan and Lorinda L. Cherry. Equations are centered, and are kept on one page. They are introduced by the **.EQ** request and terminated

```
.th                         \" set for thesis mode
.fo ´´DRAFT´´               \" define footer for each page
.tp                         \" begin title page
.(l C                       \" center a large block
THE GROWTH OF TOENAILS
IN UPPER PRIMATES
.sp
by
.sp
Frank Furter
.)l                         \" end centered part
.+c INTRODUCTION            \" begin chapter named "INTRODUCTION"
.(x t                       \" make an entry into index ‘t’
Introduction
.)x                         \" end of index entry
text of chapter one
.+c "NEXT CHAPTER"          \" begin another chapter
.(x t                       \" enter into index ‘t’ again
Next Chapter
.)x
text of chapter two
.+c CONCLUSIONS
.(x t
Conclusions
.)x
text of chapter three
.++ B                       \" begin bibliographic information
.+c BIBLIOGRAPHY            \" begin another ‘chapter’
.(x t
Bibliography
.)x
text of bibliography
.++ P                       \" begin preliminary material
.+c "TABLE OF CONTENTS"
.xp t                       \" print index ‘t’ collected above
.+c PREFACE                 \" begin another preliminary section
text of preface
```

Figure 2. Outline of a Sample Paper

by the **.EN** request.

The **.EQ** request may take an equation number as an optional argument, which is printed vertically centered on the right hand side of the equation. If the equation becomes too long it should be split between two lines. To do this, type:

```
.EQ (eq 34)
text of equation 34
.EN C
.EQ
continuation of equation 34
.EN
```

The **C** on the **.EN** request specifies that the equation will be continued.

The **tbl** program produces tables. It is fully described (including numerous examples) in the document *Tbl − A Program to Format Tables* by M. E. Lesk. Tables begin with the **.TS** request and end with the **.TE** request. Tables are normally kept on a single page. If you have a table which is too big to fit on a single page, so that you know it will extend to several pages, begin the table with the request **.TS H** and put the request **.TH** after the part of the table which you want duplicated at the top of every page that the table is printed on. For example, a table definition for a long table might look like:

```
.TS H
c s s
n n n.
THE TABLE TITLE
.TH
text of the table
.TE
```

5.5. Two Column Output

You can get two column output automatically by using the request **.2c**. This causes everything after it to be output in two-column form. The request **.bc** will start a new column; it differs from **.bp** in that **.bp** may leave a totally blank column when it starts a new page. To revert to single column output, use **.1c**.

5.6. Defining Macros

A *macro* is a collection of requests and text which may be used by stating a simple request. Macros begin with the line **.de** *xx* (where *xx* is the name of the macro to be defined) and end with the line consisting of two dots. After defining the macro, stating the line **.***xx* is the same as stating all the other lines. For example, to define a macro that spaces 3 lines and then centers the next input line, enter:

```
.de SS
.sp 3
.ce
..
```

and use it by typing:

```
.SS
Title Line
(beginning of text)
```

Macro names may be one or two characters. In order to avoid conflicts with names in −me, always use upper case letters as names. The only names to avoid are **TS**, **TH**, **TE**, **EQ**, and **EN**.

5.7. Annotations Inside Keeps

Sometimes you may want to put a footnote or index entry inside a keep. For example, if you want to maintain a "list of figures" you will want to do something like:

```
.(z
.(c
text of figure
.)c
.ce
Figure 5.
.(x f
Figure 5
.)x
.)z
```

which you may hope will give you a figure with a label and an entry in the index **f** (presumably a list of figures index). Unfortunately, the index entry is read and interpreted when the keep is read, not when it is printed, so the page number in the index is likely to be wrong. The solution is to use the magic string **\!** at the beginning of all the lines dealing with the index. In other words, you should use:

```
.(z
.(c
Text of figure
.)c
.ce
Figure 5.
\!.(x f
\!Figure 5
\!.)x
.)z
```

which will defer the processing of the index until the figure is output. This will guarantee that the page number in the index is correct. The same comments apply to blocks (with **.(b** and **.)b**) as well.

6. TROFF and the Photosetter

With a little care, you can prepare documents that will print nicely on either a regular terminal or when phototypeset using the TROFF formatting program.

6.1. Fonts

A *font* is a style of type. There are three fonts that are available simultaneously, Times Roman, Times Italic, and Times Bold, plus the special math font. The normal font is Roman. Text which would be underlined in NROFF with the **.ul** request is set in italics in TROFF.

There are ways of switching between fonts. The requests **.r**, **.i**, and **.b** switch to Roman, italic, and bold fonts respectively. You can set a single word in some font by typing (for example):

```
.i word
```

which will set *word* in italics but does not affect the surrounding text. In NROFF, italic and bold text is underlined.

Notice that if you are setting more than one word in whatever font, you must surround that word with double quote marks (‘ " ’) so that it will appear to the NROFF processor as a single word. The quote marks will not appear in the formatted text. If you do want a quote mark to appear, you should quote the entire string (even if a single word), and use *two* quote marks where you want one to appear. For example, if you want to produce the text:

"*Master Control*"

in italics, you must type:

```
.i """Master Control\|"""
```

The \| produces a very narrow space so that the “l” does not overlap the quote sign in TROFF, like this:

"*Master Control*"

There are also several “pseudo-fonts” available. The input:

```
.(b
.u underlined
.bi "bold italics"
.bx "words in a box"
.)b
```

generates

underlined
bold italics
$\boxed{\text{words in a box}}$

In NROFF these all just underline the text. Notice that pseudo font requests set only the single parameter in the pseudo font; ordinary font requests will begin setting all text in the special font if you do not provide a parameter. No more than one word should appear with these three font requests in the middle of lines. This is because of the way TROFF justifies text. For example, if you were to issue the requests:

```
.bi "some bold italics"
and
.bx "words in a box"
```

in the middle of a line TROFF would produce ***some bold italics*** and $\boxed{\text{words in a box}}$, which I think you will agree does not look good.

The second parameter of all font requests is set in the original font. For example, the font request:

```
.b bold face
```

generates “bold” in bold font, but sets “face” in the font of the surrounding text, resulting in:

boldface.

To set the two words **bold** and **face** both in **bold face**, type:

```
.b "bold face"
```

You can mix fonts in a word by using the special sequence **c** at the end of a line to indicate “continue text processing”; this allows input lines to be joined together without a space inbetween them. For example, the input:

```
.u under \c
.i italics
```

generates under*italics*, but if we had typed:

```
.u under
.i italics
```

the result would have been under *italics* as two words.

6.2. Point Sizes

The phototypesetter supports different sizes of type, measured in points. The default point size is 10 points for most text, 8 points for footnotes. To change the pointsize, type:

> .sz *+N*

where *N* is the size wanted in points. The *vertical spacing* (distance between the bottom of most letters (the *baseline*) between adjacent lines) is set to be proportional to the type size.

Warning: changing point sizes on the phototypesetter is a slow mechanical operation. Size changes should be considered carefully.

6.3. Quotes

It is conventional when using the typesetter to use pairs of grave and acute accents to generate double quotes, rather than the double quote character (‘ " ’). This is because it looks better to use grave and acute accents; for example, compare "quote" to “quote”.

In order to make quotes compatible between the typesetter and terminals, you may use the sequences ***(lq** and ***(rq** to stand for the left and right quote respectively. These both appear as " on most terminals, but are typeset as **“** and **”** respectively. For example, use:

```
\*(lqSome things aren´t true
even if they did happen.\*(rq
```

to generate the result:

> “Some things aren’t true even if they did happen.”

As a shorthand, the special font request:

> .q "quoted text"

will generate “quoted text”. Notice that you must surround the material to be quoted with double quote marks if it is more than one word.

Acknowledgments

I would like to thank Bob Epstein, Bill Joy, and Larry Rowe for having the courage to use the −me macros to produce non-trivial papers during the development stages; Ricki Blau, Pamela Humphrey, and Jim Joyce for their help with the documentation phase; and the plethora of people who have contributed ideas and have given support for the project.

This document was TROFF’ed on April 8, 1993 and applies to version 1.1 of the −me macros.

−ME REFERENCE MANUAL

Release 1.1/25

Eric P. Allman

Electronics Research Laboratory
University of California, Berkeley
Berkeley, California 94720

This document describes in extremely terse form the features of the −**me** macro package for version seven NROFF/TROFF. Some familiarity is assumed with those programs, specifically, the reader should understand breaks, fonts, pointsizes, the use and definition of number registers and strings, how to define macros, and scaling factors for ens, points, **v**'s (vertical line spaces), etc.

For a more casual introduction to text processing using NROFF, refer to the document *Writing Papers with NROFF using −me.*

There are a number of macro parameters that may be adjusted. Fonts may be set to a font number only. In NROFF font 8 is underlined, and is set in bold font in TROFF (although font 3, bold in TROFF, is not underlined in NROFF). Font 0 is no font change; the font of the surrounding text is used instead. Notice that fonts 0 and 8 are "pseudo-fonts"; that is, they are simulated by the macros. This means that although it is legal to set a font register to zero or eight, it is not legal to use the escape character form, such as:

```
\f8
```

All distances are in basic units, so it is nearly always necessary to use a scaling factor. For example, the request to set the paragraph indent to eight one-en spaces is:

```
.nr pi 8n
```

and not

```
.nr pi 8
```

which would set the paragraph indent to eight basic units, or about 0.02 inch. Default parameter values are given in brackets in the remainder of this document.

Registers and strings of the form **$***x* may be used in expressions but should not be changed. Macros of the form **$***x* perform some function (as described) and may be redefined to change this function. This may be a sensitive operation; look at the body of the original macro before changing it.

All names in −me follow a rigid naming convention. The user may define number registers, strings, and macros, provided that s/he uses single character upper case names or double character names consisting of letters and digits, with at least one upper case letter. In no case should special characters be used in user-defined names.

†NROFF and TROFF are Trademarks of Bell Laboratories.

On daisy wheel type printers in twelve pitch, the **−rx1** flag can be stated to make lines default to one eighth inch (the normal spacing for a newline in twelve-pitch). This is normally too small for easy readability, so the default is to space one sixth inch.

This documentation was TROFF'ed on April 8, 1993 and applies to version 1.1/25 of the −me macros.

1. Paragraphing

These macros are used to begin paragraphs. The standard paragraph macro is **.pp**; the others are all variants to be used for special purposes.

The first call to one of the paragraphing macros defined in this section or the **.sh** macro (defined in the next session) *initializes* the macro processor. After initialization it is not possible to use any of the following requests: **.sc**, **.lo**, **.th**, or **.ac**. Also, the effects of changing parameters which will have a global effect on the format of the page (notably page length and header and footer margins) are not well defined and should be avoided.

.lp \h'|86724u'\c Begin left-justified paragraph. Centering and underlining are turned off if they were on, the font is set to **n(pf** [1] the type size is set to **n(pp** [10p], and a **n(ps** space is inserted before the paragraph [0.35v in TROFF, 1v or 0.5v in NROFF depending on device resolution]. The indent is reset to **n($i** [0] plus **n(po** [0] unless the paragraph is inside a display. (see **.ba**). At least the first two lines of the paragraph are kept together on a page.

.pp \h'|86724u'\c Like **.lp**, except that it puts **n(pi** [5n] units of indent. This is the standard paragraph macro.

.ip *T I* \h'|86724u'\c Indented paragraph with hanging tag. The body of the following paragraph is indented *I* spaces (or **n(ii** [5n] spaces if *I* is not specified) more than a non-indented paragraph (such as with **.pp**) is. The title *T* is exdented (opposite of indented). The result is a paragraph with an even left edge and *T* printed in the margin. Any spaces in *T* must be unpaddable. If *T* will not fit in the space provided, **.ip** will start a new line.

.np \h'|86724u'\c A variant of .ip which numbers paragraphs. Numbering is reset after a **.lp**, **.pp**, or **.sh**. The current paragraph number is in **n($p**.

2. Section Headings

Numbered sections are similiar to paragraphs except that a section number is automatically generated for each one. The section numbers are of the form **1.2.3**. The *depth* of the section is the count of numbers (separated by decimal points) in the section number.

Unnumbered section headings are similar, except that no number is attached to the heading.

.sh *+N T a b c d e f* \h'|86724u'\c Begin numbered section of depth *N*. If *N* is missing the current depth (maintained in the number register **n($0**) is used. The values of the individual parts of the section number are maintained in **n($1** through **n($6**. There is a **n(ss** [1v] space before the section. *T* is printed as a section title in font **n(sf** [8] and size **n(sp** [10p]. The "name" of the section may be accessed via ***($n**. If **n(si** is non-zero, the base indent is set to **n(si** times the section depth, and the section title is exdented. (See **.ba**.) Also, an additional indent of **n(so** [0] is added to the section title (but not to the body of the section). The font is then set to the paragraph font, so that more information may occur on the line with the section number and title. **.sh** insures that there is enough room to print the section head plus the beginning of a paragraph (about 3 lines total). If *a* through *f* are specified, the section number is set to that number rather than incremented automatically. If any of *a* through *f* are a hyphen that number is not reset. If *T* is a single underscore ("_") then the section depth and numbering is reset, but the base indent is not reset and nothing is printed out. This is useful to automatically coordinate section numbers with chapter numbers.

.sx *+N* \h'|86724u'\c Go to section depth *N* [−**1**], but do not print the number and title, and do not increment the section number at level *N*. This has the effect of starting a new paragraph at level *N*.

.uh *T* \h'|86724u'\c Unnumbered section heading. The title *T* is printed with the same rules for spacing, font, etc., as for **.sh**.

.$p *T B N* \h'|86724u'\c Print section heading. May be redefined to get fancier headings. *T* is the title passed on the **.sh** or **.uh** line; *B* is the section number for this section, and *N* is the depth of this section. These parameters are not always present; in particular, **.sh** passes all three, **.uh** passes only the first, and **.sx** passes three, but the first two are null strings. Care should be taken if this macro is redefined; it is quite complex and subtle.

.$0** *T B N* \h'|86724u'\c This macro is called automatically after every call to **.$p. It is normally undefined, but may be used to automatically put every section title into the table of contents or for some similiar function. *T* is the section title for the section title which was just printed, *B* is the section number, and *N* is the section depth.

.$1 – **.$6** \h'|86724u'\c Traps called just before printing that depth section. May be defined to (for example) give variable spacing before sections. These macros are called from **.$p**, so if you redefine that macro you may lose this feature.

3. Headers and Footers

Headers and footers are put at the top and bottom of every page automatically. They are set in font **\n(tf** [3] and size **\n(tp** [10p]. Each of the definitions apply as of the *next* page. Three-part titles must be quoted if there are two blanks adjacent anywhere in the title or more than eight blanks total.

The spacing of headers and footers are controlled by three number registers. **\n(hm** [4v] is the distance from the top of the page to the top of the header, **\n(fm** [3v] is the distance from the bottom of the page to the bottom of the footer, **\n(tm** [7v] is the distance from the top of the page to the top of the text, and **\n(bm** [6v] is the distance from the bottom of the page to the bottom of the text (nominal). The macros **.m1**, **.m2**, **.m3**, and **.m4** are also supplied for compatibility with ROFF documents.

.he *´l´m´r´* \h'|86724u'\c Define three-part header, to be printed on the top of every page.

.fo *´l´m´r´* \h'|86724u'\c Define footer, to be printed at the bottom of every page.

.eh *´l´m´r´* \h'|86724u'\c Define header, to be printed at the top of every even-numbered page.

.oh *´l´m´r´* \h'|86724u'\c Define header, to be printed at the top of every odd-numbered page.

.ef *´l´m´r´* \h'|86724u'\c Define footer, to be printed at the bottom of every even-numbered page.

.of *´l ´m ´r ´* \h'|86724u'\c Define footer, to be printed at the bottom of every odd-numbered page.

.hx \h'|86724u'\c Suppress headers and footers on the next page.

.m1 *+N* \h'|86724u'\c Set the space between the top of the page and the header [4v].

.m2 *+N* \h'|86724u'\c Set the space between the header and the first line of text [2v].

.m3 *+N* \h'|86724u'\c Set the space between the bottom of the text and the footer [2v].

.m4 *+N* \h'|86724u'\c Set the space between the footer and the bottom of the page [4v].

.ep \h'|86724u'\c End this page, but do not begin the next page. Useful for forcing out footnotes, but other than that hardly every used. Must be followed by a **.bp** or the end of input.

.$h \h'|86724u'\c Called at every page to print the header. May be redefined to provide fancy (e.g., multi-line) headers, but doing so loses the function of the **.he**, **.fo**, **.eh**, **.oh**, **.ef**, and **.of** requests, as well as the chapter-style title feature of **.+c**.

.$f** \h'|86724u'\c Print footer; same comments apply as in **.$h.

.$H \h'|86724u'\c A normally undefined macro which is called at the top of each page (after outputing the header, initial saved floating keeps, etc.); in other words, this macro is called immediately before printing text on a page. It can be used for column headings and the like.

4. Displays

All displays except centered blocks and block quotes are preceeded and followed by an extra **n(bs** [same as **n(ps**] space. Quote spacing is stored in a separate register; centered blocks have no default initial or trailing space. The vertical spacing of all displays except quotes and centered blocks is stored in register **n($R** instead of **n($r**.

.(l *m f* \h'|86724u'\c Begin list. Lists are single spaced, unfilled text. If *f* is **F**, the list will be filled. If *m* [**I**] is **I** the list is indented by **n(bi** [4n]; if **M** the list is indented to the left margin; if **L** the list is left justified with respect to the text (different from **M** only if the base indent (stored in **n($i** and set with **.ba**) is not zero); and if **C** the list is centered on a line-by-line basis. The list is set in font **n(df** [0]. Must be matched by a **.)l**. This macro is almost like **.(b** except that no attempt is made to keep the display on one page.

.)l \h'|86724u'\c End list.

.(q \h'|86724u'\c Begin major quote. These are single spaced, filled, moved in from the text on both sides by **n(qi** [4n], preceeded and followed by **n(qs** [same as **n(bs**] space, and are set in point size **n(qp** [one point smaller than surrounding text].

.)q \h'|86724u'\c End major quote.

.(b *m f* \h'|86724u'\c Begin block. Blocks are a form of *keep*, where the text of a keep is kept together on one page if possible (keeps are useful for tables and figures which should not be broken over a page). If the block will not fit on the current page a new page is begun, *unless* that would leave more than **n(bt** [0] white space at the bottom of the text. If **n(bt** is zero, the threshold feature is turned off. Blocks are not filled unless *f* is **F**, when they are filled. The block will be left-justified if *m* is **L**, indented by **n(bi** [4n] if *m* is **I** or absent, centered (line-for-line) if *m* is **C**, and left justified to the margin (not to the base indent) if *m* is

M. The block is set in font **n(df** [0].

.)b \h'|86724u'\c End block.

.(z *m f* \h'|86724u'\c Begin floating keep. Like **.(b** except that the keep is *floated* to the bottom of the page or the top of the next page. Therefore, its position relative to the text changes. The floating keep is preceeded and followed by **n(zs** [1v] space. Also, it defaults to mode **M**.

.)z \h'|86724u'\c End floating keep.

.(c \h'|86724u'\c Begin centered block. The next keep is centered as a block, rather than on a line-by-line basis as with **.(b C**. This call may be nested inside keeps.

.)c \h'|86724u'\c End centered block.

5. Annotations

.(d \h'|86724u'\c Begin delayed text. Everything in the next keep is saved for output later with **.pd**, in a manner similar to footnotes.

.)d *n* \h'|86724u'\c End delayed text. The delayed text number register **n($d** and the associated string *# are incremented if *# has been referenced.

.pd \h'|86724u'\c Print delayed text. Everything diverted via **.(d** is printed and truncated. This might be used at the end of each chapter.

.(f \h'|86724u'\c Begin footnote. The text of the footnote is floated to the bottom of the page and set in font **n(ff** [1] and size **n(fp** [8p]. Each entry is preceeded by **n(fs** [0.2v] space, is indented **n(fi** [3n] on the first line, and is indented **n(fu** [0] from the right margin. Footnotes line up underneath two columned output. If the text of the footnote will not all fit on one page it will be carried over to the next page.

.)f *n* \h'|86724u'\c End footnote. The number register **n($f** and the associated string ** are incremented if they have been referenced.

.$s \h'|86724u'\c The macro to output the footnote seperator. This macro may be redefined to give other size lines or other types of separators. Currently it draws a 1.5i line.

.(x *x* \h'|86724u'\c Begin index entry. Index entries are saved in the index *x* [**x**] until called up with **.xp.** Each entry is preceeded by a **n(xs** [0.2v] space. Each entry is "undented" by **n(xu** [0.5i]; this register tells how far the page number extends into the right margin.

.)x *P A* \h'|86724u'\c End index entry. The index entry is finished with a row of dots with *A* [null] right justified on the last line (such as for an author's name), followed by P [**n%**]. If *A* is specified, *P* must be specified; **n%** can be used to print the current page number. If *P* is an underscore, no page number and no row of dots are printed.

.xp *x* \h'|86724u'\c Print index *x* [**x**]. The index is formated in the font, size, and so forth in effect at the time it is printed, rather than at the time it is collected.

6. Columned Output

.2c *+S N* \h'|86724u'\c Enter two-column mode. The column separation is set to *+S* [4n, 0.5i in ACM mode] (saved in **n($s**). The column width, calculated to fill the single column line length with both columns, is stored in **n($l**. The current column is in **n($c**. You can test register **n($m** [1] to see if you are in single column or double column mode. Actually, the request enters *N* [2] columned output.

.1c \h'|86724u'\c Revert to single-column mode.

.bc \h'|86724u'\c Begin column. This is like **.bp** except that it begins a new column on a new page only if necessary, rather than forcing a whole new page if there is another column left on the current page.

7. Fonts and Sizes

.sz *+P* \h'|86724u'\c The pointsize is set to *P* [10p], and the line spacing is set proportionally. The ratio of line spacing to pointsize is stored in **\n($r**. The ratio used internally by displays and annotations is stored in **\n($R** (although this is not used by **.sz**).

.r *W X* \h'|86724u'\c Set *W* in roman font, appending *X* in the previous font. To append different font requests, use *X* = **\c**. If no parameters, change to roman font.

.i *W X* \h'|86724u'\c Set *W* in italics, appending *X* in the previous font. If no parameters, change to italic font. Underlines in NROFF.

.b *W X* \h'|86724u'\c Set *W* in bold font and append *X* in the previous font. If no parameters, switch to bold font. In NROFF, underlines.

.rb *W X* \h'|86724u'\c Set *W* in bold font and append *X* in the previous font. If no parameters, switch to bold font. **.rb** differs from **.b** in that **.rb** does not underline in NROFF.

.u *W X* \h'|86724u'\c Underline *W* and append *X*. This is a true underlining, as opposed to the **.ul** request, which changes to "underline font" (usually italics in TROFF). It won't work right if *W* is spread or broken (including hyphenated). In other words, it is safe in nofill mode only.

.q *W X* \h'|86724u'\c Quote *W* and append *X*. In NROFF this just surrounds *W* with double quote marks (' " '), but in TROFF uses directed quotes.

.bi *W X* \h'|86724u'\c Set *W* in bold italics and append *X*. Actually, sets *W* in italic and overstrikes once. Underlines in NROFF. It won't work right if *W* is spread or broken (including hyphenated). In other words, it is safe in nofill mode only.

.bx *W X* \h'|86724u'\c Sets *W* in a box, with *X* appended. Underlines in NROFF. It won't work right if *W* is spread or broken (including hyphenated). In other words, it is safe in nofill mode only.

8. Roff Support

.ix *+N* \h'|86724u'\c Indent, no break. Equivalent to **´in** *N*.

.bl *N* \h'|86724u'\c Leave *N* contiguous white space, on the next page if not enough room on this page. Equivalent to a **.sp** *N* inside a block.

.pa *+N* \h'|86724u'\c Equivalent to **.bp**.

.ro \h'|86724u'\c Set page number in roman numerals. Equivalent to **.af % i**.

.ar \h'|86724u'\c Set page number in arabic. Equivalent to **.af % 1**.

.n1 \h'|86724u'\c Number lines in margin from one on each page.

.n2 *N* \h'|86724u'\c Number lines from *N*, stop if $N = 0$.

.sk \h'|86724u'\c Leave the next output page blank, except for headers and footers. This is used to leave space for a full-page diagram which is produced externally and pasted in later. To get a partial-page paste-in display, say **.sv** *N*, where *N* is the amount of space to leave; this space will be output immediately if there is room, and will otherwise be output at the top of the next page. However, be warned: if *N* is greater than the amount of available space on an empty page, no space will ever be output.

9. Preprocessor Support

.EQ *m T* \h'|86724u'\c Begin equation. The equation is centered if *m* is **C** or omitted, indented **\n(bi** [4n] if *m* is **I**, and left justified if *m* is **L**. *T* is a title printed on the right margin next to the equation. See *Typesetting Mathematics − User's Guide* by Brian W. Kernighan and Lorinda L. Cherry.

.EN *c* \h'|86724u'\c End equation. If *c* is **C** the equation must be continued by immediately following with another **.EQ**, the text of which can be centered along with this one. Otherwise, the equation is printed, always on one page, with **\n(es** [0.5v in TROFF, 1v in NROFF] space above and below it.

.TS *h* \h'|86724u'\c Table start. Tables are single spaced and kept on one page if possible. If you have a large table which will not fit on one page, use h = **H** and follow the header part (to be printed on every page of the table) with a **.TH**. See *Tbl − A Program to Format Tables* by M. E. Lesk.

.TH \h'|86724u'\c With **.TS H**, ends the header portion of the table.

.TE \h'|86724u'\c Table end. Note that this table does not float, in fact, it is not even guaranteed to stay on one page if you use requests such as **.sp** intermixed with the text of the table. If you want it to float (or if you use requests inside the table), surround the entire table (including the **.TS** and **.TE** requests) with the requests **.(z** and **.)z**.

10. Miscellaneous

.re \h'|86724u'\c Reset tabs. Set to every 0.5i in TROFF and every 0.8i in NROFF.

.ba *+N* \h'|86724u'\c Set the base indent to *+N* [0] (saved in **\n($i**). All paragraphs, sections, and displays come out indented by this amount. Titles and footnotes are unaffected. The **.sh** request performs a **.ba** request if **\n(si** [0] is not zero, and sets the base indent to **\n(si*\n($0**.

.xl *+N* \h'|86724u'\c Set the line length to *N* [6.0i]. This differs from **.ll** because it only affects the current environment.

.ll *+N* \h'|86724u'\c Set line length in all environments to *N* [6.0i]. This should not be used after output has begun, and particularly not in two-columned output. The current line length is stored in **\n($l**.

.hl \h'|86724u'\c Draws a horizontal line the length of the page. This is useful inside floating keeps to differentiate between the text and the figure.

.lo \h'|86724u'\c This macro loads another set of macros (in **/usr/lib/me/local.me**) which is intended to be a set of locally defined macros. These macros should all be of the form **.****X*, where *X* is any letter (upper or lower case) or digit.

11. Standard Papers

.tp \h'|86724u'\c Begin title page. Spacing at the top of the page can occur, and headers and footers are supressed. Also, the page number is not incremented for this page.

.th \h'|86724u'\c Set thesis mode. This defines the modes acceptable for a doctoral dissertation at Berkeley. It double spaces, defines the header to be a single page number, and changes the margins to be 1.5 inch on the left and one inch on the top. **.++** and **.+c** should be used with it. This macro must be stated before initialization, that is, before the first call of a paragraphing macro or **.sh**.

.++ *m H* \h'|86724u'\c This request defines the section of the paper which we are entering. The section type is defined by *m*. **C** means that we are entering the chapter portion of the paper, **A** means that we are entering the appendix portion of the paper, **P** means that the material following should be the preliminary portion (abstract, table of contents, etc.) portion of the paper, **AB** means that we are entering the abstract (numbered independently from 1 in Arabic numerals), and **B** means that we are entering the bibliographic portion at the end of the paper. Also, the variants **RC** and **RA** are allowed, which specify renumbering of pages from one at the beginning of each chapter or appendix, respectively. The *H* parameter defines the new header. If there are any spaces in it, the entire header must be quoted. If you want the header to have the chapter number in it, Use the string \\\\n(ch. For example, to number appendixes **A.1** etc., type **.++ RA** ´´´\\\\n(ch.%´. Each section (chapter, appendix, etc.) should be preceeded by the **.+c** request. It should be mentioned that it is easier when using TROFF to put the front material at the end of the paper, so that the table of contents can be collected and output; this material can then be physically moved to the beginning of the paper.

.+c *T* \h'|86724u'\c Begin chapter with title *T*. The chapter number is maintained in **n(ch**. This register is incremented every time **.+c** is called with a parameter. The title and chapter number are printed by **.$c**. The header is moved to the footer on the first page of each chapter. If *T* is omitted, **.$c** is not called; this is useful for doing your own "title page" at the beginning of papers without a title page proper. **.$c** calls **.$C** as a hook so that chapter titles can be inserted into a table of contents automatically. The footnote numbering is reset to one.

.$c** *T* \h'|86724u'\c Print chapter number (from **n(ch**) and *T*. This macro can be redefined to your liking. It is defined by default to be acceptable for a PhD thesis at Berkeley. This macro calls **$C, which can be defined to make index entries, or whatever.

.$C** *K N T* \h'|86724u'\c This macro is called by **.$c. It is normally undefined, but can be used to automatically insert index entries, or whatever. *K* is a keyword, either "Chapter" or "Appendix" (depending on the **.++** mode); *N* is the chapter or appendix number, and *T* is the chapter or appendix title.

.ac *A N* \h'|86724u'\c This macro (short for **.acm**) sets up the NROFF environment for photo-ready papers as used by the ACM. This format is 25% larger, and has no headers or footers. The author's name *A* is printed at the bottom of the page (but off the part which will be printed in the conference proceedings), together with the current page number and the total number of pages *N*. Additionally, this macro loads the file **/usr/lib/me/acm.me**, which may later be augmented with other macros useful for printing papers for ACM conferences. It should be noted that this macro will not work correctly in TROFF, since it sets the page length wider than the physical width of the phototypesetter roll.

12. Predefined Strings

** \h'|86724u'\c Footnote number, actually \\*[\\**n($f**\*]. This macro is incremented after each call to **.)f**.

*# \h'|86724u'\c Delayed text number. Actually [\**n($d**].

*[\h'|86724u'\c Superscript. This string gives upward movement and a change to a smaller point size if possible, otherwise it gives the left bracket character ('[').
Extra space is left above the line to allow room for the superscript.

*] \h'|86724u'\c Unsuperscript. Inverse to \*[. For example, to produce a superscript you might type **x**\***[2**\*], which will produce **x.**

*< \h'|86724u'\c Subscript. Defaults to '<' if half-carriage motion not possible. Extra space is left below the line to allow for the subscript.

*> \h'|86724u'\c Inverse to \\*<.

***(dw** \h'|86724u'\c The day of the week, as a word.

***(mo** \h'|86724u'\c The month, as a word.

***(td** \h'|86724u'\c Today's date, directly printable. The date is of the form April 8, 1993. Other forms of the date can be used by using \n(dy (the day of the month; for example, 8), \***(mo** (as noted above) or \n(mo (the same, but as an ordinal number; for example, April is 4), and \n(yr (the last two digits of the current year).

***(lq** \h'|86724u'\c Left quote marks. Double quote in NROFF.

***(rq** \h'|86724u'\c Right quote.

*− \h'|86724u'\c ¾ em dash in TROFF; two hyphens in NROFF.

13. Special Characters and Marks

There are a number of special characters and diacritical marks (such as accents) available through −me. To reference these characters, you must call the macro **.sc** to define the characters before using them.

.sc \h'|86724u'\c Define special characters and diacritical marks, as described in the remainder of this section. This macro must be stated before initialization.

The special characters available are listed below.

Name	Usage	Example	
Acute accent	*´	a*´	á
Grave accent	*`	e*`	è
Umlat	*:	u*:	ü
Tilde	*~	n*~	ñ
Caret	*^	e*^	ê
Cedilla	*,	c*,	ç
Czech	*v	e*v	ě
Circle	*o	A*o	Å
There exists	*(qe		∃
For all	*(qa		∀

Acknowledgments

I would like to thank Bob Epstein, Bill Joy, and Larry Rowe for having the courage to use the −me macros to produce non-trivial papers during the development stages; Ricki Blau, Pamela Humphrey, and Jim Joyce for their help with the documentation phase; and the plethora of people who have contributed ideas and have given support for the project.