TEXsis
TEX Macros for Physicists

Eric Myers

Center for Relativity and Physics Department
University of Texas at Austin
Austin, Texas 78712 USA

and

Frank E. Paige

Physics Research Division
Superconducting Super Collider Laboratory
2550 Beckleymeade Avenue
Dallas, Texas 75237 USA

TEXsis is a collection of TEX macros for typesetting physics documents such as papers and preprints, conference proceedings, books, theses, referee reports, letters, and memos. TEXsis macros provide automatic numbering of equations, automatic numbering and formatting of references, double column formatting, macros for making tables and figures, with or without captions, including tables with horizontal and vertical rules. TEXsis supports a wide variety of type sizes and a number of specialized document formats, and it even includes macros for making form letters for job applications or letters of recommendation. This document describes version 2.15/h of TEXsis.

Printed 2 May 1993

TEXsis: TEX Macros for Physicists

Source code and documentation for TEXsis are available via anonymous ftp over the Internet from lifshitz.ph.utexas.edu. To copy the files, run ftp on your computer and connect to lifshitz.ph.utexas.edu (or 128.83.131.57). Give the login name "anonymous" and your username as the password. The TEXsis files for Unix machines are in a (possibly compressed) tar file in the subdirectory `texsis`, or in shar bundles in the subdirectory `texsis/Unix`. For Vax/VMS machines the files are available in DCL bundles (.COM files which unpack themselves into many files) in the subdirectory `texsis/VMS`. The code is the same in any case, it's just packed differently. The macros for making ruled tables known as `ruled.tex` are available independently in the subdirectory `texsis/Tables`.

If you cannot obtain the files by anonymous ftp, they are available by e-mail from the authors via Bitnet or the Internet, or we can try to make arrangements to make copies on magnetic tape or floppy disks (you provide the media or pay for it).

We would be grateful to hear of any errors found in this documentation or in the software, and we also invite suggestions for improvements in both. When reporting an error in the code it would help us a great deal if you would send us a copy of the relevant sections of the manuscript file, the log file, and if possible of any output generated.

To contact us:

U.S.Mail:	Eric Myers Center for Relativity Physics Department University of Texas Austin, TX 78712 USA	Frank E. Paige Physics Research Division SSC Laboratory 2550 Beckleymeade Ave. Dallas, TX 75237 USA
Internet Electronic Mail:	myers@landau.ph.utexas.edu	paige@sscvx1.ssc.gov
Bitnet Electronic Mail:	MYERS@UTAPHY	PAIGE@SSCVX1
Usenet Electronic Mail:	myers@ut-emx.UUCP	

1. Overview

TEX is a computer typesetting language developed by Donald Knuth and described in *The TEXbook.*[1] Its greatest strength is its ability to typeset technical material and mathematical equations using a syntax that is based on content; TEX automatically handles the actual placement of the characters on the page. It also has a powerful macro capability, giving it great flexibility. This has been exploited by many authors to write "formats," or macro packages, providing features not contained in the basic language, which is sometimes referred to as "Plain" TEX.

TEXsis is a TEX macro package which is designed primarily for typesetting physics papers and preprints, conference proceedings, and related documents. It has been developed from the TechRpt format described by W. Groppe[2] and from TEXsis 1.01, the TEX thesis format by E. Myers.[3] The general scope of TEXsis is similar to the SLAC format PHYZZX,[4] but it offers some significant advantages, such as a more powerful and flexible way of referring to references, equations, and other numbered objects. It is functionally a superset of Plain TEX, so that everything described in *The TEXbook* will work in TEXsis, although a few of the Plain TEX macros have actually been modified.

We should mention at the outset that LATEX, another popular TEX-based macro package, is not compatible with Plain TEX, and as a consequence it is not possible to use TEXsis with LATEX or vice versa.

This document explains how to use TEXsis. Since it is a user's guide, it makes no attempt to describe how the various macros work; for this information one should consult the source code, which is extensively commented. However, we do assume that the reader is already familiar with Plain TEX. New users should therefore first see either *The TEXbook*, especially chapters 1-10 and 16-19, or *A Gentle Introduction to TEX* by Michael Doob,[5] or the manual by Spivak[6] which accompanies PCTEX (which is an implementation of TEX for the IBM PC). The sections of the present document cover the following topics:

- Section 2 – Getting Started. How to run and initialize TEXsis, and the "master file" concept.
- Section 3 – Selecting sizes and fonts of type, choosing single, double, or triple spacing, and automatically handling quotation marks.
- Section 4 – Automatic numbering of equations, automatic sizing of parentheses and brackets, and special symbols for physics equations.
- Section 5 – Defining new references, for which numbers are assigned automatically, citing references already so defined, and making a list of references.
- Section 6 – Dividing a document into chapters, sections, subsections, and appendices, and producing a table of contents.
- Section 7 – Using pre-defined layouts for documents such as preprints, conference proceedings and theses. Creating a title page.

- Section 8 – Displaying text in special ways using "environments," including those for centered lines, lists of things, listings of computer programs, and TEX examples.

- Section 9 – Producing automatically numbered tables and figures and producing ruled tables.

- Section 10 – Typing letters, form letters, memos and referee reports.

- Section 11 – Producing double column output.

- Section 12 – Labeling things like section numbers, figures, items in an outline, or page numbers, and referring to them in other parts of the document.

- Section 13 – Miscellaneous macros for ignoring text and doing various other useful things, and a checkpoint/restart feature for printing very long documents.

Appendix A explains how to install TEXsis on a VAX/VMS system, on an IBM PC, and on the Berkeley Unix operating system. It also explains how to customize TEXsis for your own installation. Appendix B lists the fonts usually available with TEXsis. Appendix C lists changes that have been made to TEXsis in recent revisions.

Two examples of TEXsis documents are included in the standard distribution. The first is a short (3 pages) example of a physics paper typeset with TEXsis, which is contained in the file `Example.tex`. The second is the manual itself, which was also created with TEXsis. If you would like to look at the files used to create the Manual the master file is called `TXSdoc.doc`, and the various sections of the document are in files with names of the form `TXS`*xxxx*`.doc`. For those who are comfortable programming in TEX it is possible to extract specific macros from the TEXsis source code or to modify or even rewrite these macros. The source code is contained in files with names of the form `TXS`*xxxx*`.tex`. We have tried wherever possible to include illuminating comments in the source code.

1.1. Some Philosophy

In designing the TEXsis macros we have tried to follow some basic guidelines which we feel make TEXsis a more useful tool. It may help you while learning TEXsis to know what these guidelines are:

1) Most importantly, TEXsis will always be a superset of Plain TEX, so that anything you can't do in TEXsis (and even those that you can do) can be done with Plain TEX instead. This also means that TEXsis should be compatible with Plain TEX macro packages, such as `epsf.tex` and `index.tex`, or other specialized macro sets.[†] Even if some of the Plain macros are actually redefine in TEXsis they are required to still behave as described in *The TEXbook*.

[†] We cannot completely guarantee this because other macro packages might define macros with the same names as used by TEXsis, although we have taken several steps to prevent this kind of conflict.

2) There are two different ways in which arguments may be passed to macros, depending on the length of the argument. Short arguments are simply enclosed in curly brackets, as in:

```
\section{The nonlinear $\sigma$-model}
```

Longer amounts of text (many lines) and text which is considered to be enclosed in special environments is begun with something like `\thing` and terminated with `\endthing`. For example,

```
\abstract
A mechanism for total confinement of quarks, similar to that of
Schwinger, is defined which requires the existence of Abelian or
non-Abelian gauge fields.  It is shown how to quantize ...
\endabstract
```

This is similar to the `\begin{thing} ... \end{thing}` construction in LATEX but is slightly easier to type and, more important, is compatible with Plain TEX. The reason for preferring this way of delimiting text or arguments is that for very long pieces of material it's much easier to look for `\thing` and `\endthing` than it is to count curly brackets.

3) When a TEXsis macro is similar to a Plain TEX macro it should, when possible, have a similar name and syntax. Thus `\Footnote` in TEXsis is used exactly like `\footnote` in Plain TEX, and in ruled tables "`&`" and "`\cr`" can be used exactly as in an `\halign` (the exception being that, following rule 2 above, you begin the ruled table with `\ruledtable` and end it with `\endruledtable` instead of enclosing the whole body of the table in curly brackets).

4) Macro names will be short and simple but descriptive. Almost all Plain TEX macro names are in lower case for ease of typing, so macros with names containing uppercase letters can be immediately recognized as being a part of TEXsis (instead of Plain TEX). As a general rule though we also try to use lowercase for frequently used macro names for ease of typing.

5) The source code defining the macros should be available to the users, so that those who want to see how something is done can do so, and those who want to make their own modifications can do so. The macro definitions should include sufficient comments that (usually) someone with only a little bit of experience with TEX can understand what is going on. Also, macros which perform similar functions and work together should be kept in a single file, but separate parts of the whole package should be kept in separate files. When possible these individual files should be usable on their own with Plain TEX. For example, right now it's possible to use just the double column macros of TEXsis by saying `\input TXSdcol.tex`, and the ruled table macros can be used in Plain TEX by saying `\input TXSruled.tex`.

2. Getting Started

Assuming that TEXsis has been properly installed on your computer, using TEXsis is as simple as using Plain TEX. To run TEX with the TEXsis format to process a manuscript file called `myfile.tex` you simply give the command:

```
texsis myfile
```

As usual, the output from TEX is put in the file `myfile.dvi`, which you may print on a laser printer or other output device.

Since TEXsis is a superset of Plain TEX, any file that you process with Plain TEX can be processed with TEXsis, and you will get the same results. In particular, Plain TEX defaults to 10 pt. type and single spacing between lines, and so does TEXsis unless you initialize its special features. But since you are probably using TEXsis because of these special features, you should begin your manuscript file with the initialization command

```
\texsis
```

One of the things this command does is switch the output to 12 pt. type, double spaced, which generally looks better on a small laser printer. Commands that allow you to change the type size or line spacing yourself are described in Section 3. More specialized initialization commands may be used in place of (or in addition to) the `\texsis` command when you are using a pre-defined document layout. These are described in Section 7.

Note also that just as in Plain TEX there is a difference between uppercase and lowercase letters in control word names. Thus if you say `\TeXsis` instead of `\texsis` you won't initialize any special features but you will print out "TEXsis."

2.1. The Master File

An important and useful concept for typesetting a paper or any moderately long document is that of the *Master File.* The idea is to create one small file which initializes TEXsis with `\texsis`, makes any special definitions or sets parameters, and then reads in the actual text of the document in sections using the `\input` command. As a simple example consider a manuscript called MUONS. The master file is called `MUONS.tex`, and contains the following:

```
\texsis              % initialize TeXsis
\input MyMacros      % read special macros for this paper
\input MUONSr        % read references for the paper
\input MUONS0        % read title page for the paper
\input MUONS1        % read first section of manuscript
\input MUONS2        % read second section of manuscript
\ListReferences      % print the reference list
\end
```

The file `MyMacros.tex` contains any special macro definitions used in this paper. (If there are none, leave this line out.) The file `MUONSr.tex` contains the references used in this paper, beginning with `\referencelist` and ending with `\endreferencelist`, as described in Section 5. The file `MUONS0.tex` contains the commands for producing the title page of the paper, as described in Section 7. The actual text of the manuscript is divided into two parts, which are kept in the files `MUONS1.tex` and `MUONS2.tex`. To process the entire document one simply runs TEXsis on the master file with the command

```
texsis MUONS
```

If you are familiar with computer programming you may think of the master file as the main program and the other files as subroutines called by the main program.

One of the great advantages of using a master file, aside from keeping the manuscript organized, is that it lets you work on one section of the paper at a time. For example, suppose you want to run off a draft of just the first section of the paper called MUONS above. You don't need to process the references, the title page, or the second section of the paper. Recall that the percent sign (`%`) is the comment character in TEX — anything appearing on a line after a `%` is ignored by TEX. Thus we simply edit the master file and put a `%` in front of certain lines, like this:

```
\texsis                     % initialize TeXsis
\draft                      % optional draft mode
\input MyMacros             % read special macros for this paper
% \input MUONSr             % read references for the paper
% \input MUONS0             % read title page for the paper
\input MUONS1               % read first section of manuscript
% \input MUONS2             % read section section of manuscript
% \ListReferences           % print the references
\end
```

Now when you run TEXsis on this file only the first section of the paper will be processed, but it will use the macros defined in `MyMacros.tex`, as well as any other definitions that you might add to the master file. Once the first section is error free you can repeat the process on the second section of the paper, without rerunning the first section. After both sections are correct you can reinstate all lines in the master file and process the whole document.

3. Type Sizes, Type Styles, and Line Spacing

Plain TEX is set up to use 10 pt type, with smaller sizes for superscripts and subscripts. Unfortunately 10 pt. type is too small to print clearly on most small laser printers. TEXsis expands the available type sizes to include 11, 12, 14, and 16 pt type, and TEXsis makes it easy to change from one size of type to another. Type sizes are selected with the following commands:

`\ninepoint`	Gives 9 pt type (not illustrated).
`\tenpoint`	Gives 10 pt type.
`\elevenpoint`	Gives 11 pt type. (not illustrated).
`\twelvepoint`	Gives 12 pt type.
`\fourteenpoint`	Gives 14 pt type.
`\sixteenpoint`	Gives 16 pt type.
`\twentypoint`	Gives 20 pt type (not illustrated).
`\twentyfourpoint`	Gives 24 pt type (not illustrated).

In many implementations of TEX there is insufficient room to load all of these fonts, so fonts are loaded only when they are first used, and not all of them can be illustrated here. TEXsis starts out with `\tenpoint`, just like Plain TEX, but switches to `\twelvepoint` once you say `\texsis`.

In Plain TEX you can switch between Roman, Bold, or Italic typestyles with the commands `\rm`, `\bf`, and `\it`. The same is true with TEXsis. These commands change the style of the type, but not the size. The complete set of commands to change typestyles is:

`\rm`	Gives Roman type. Available in all sizes.
`\it`	Gives *italic type* for emphasis. Available in all sizes.
`\sl`	Gives *slanted type* for book titles, etc. Available in all sizes.
`\bf`	Gives **bold face type** for emphasis. Available in all sizes.
`\tt`	Gives `typewriter type` for listing computer programs, TEX examples, etc. Available in 10, 11, and 12 pt.
`\mib`	Gives math italic bold in math mode (`$...$`) for titles. Available in 10, 11, 12, 14, 16, 20, and 24 pt base sizes, but the minimum size for subscripts and superscripts is 10 pt.
`\scr`	Gives script capital letters (using Ralph Smith's Formal Script fonts, the `rsfs` fonts, if they are available on your computer). Use as you would use `\cal`.

TEXsis includes the mathematics fonts in all sizes; these are used automatically when you type equations between the math quotes, `$...$`. Limited sans-serif fonts are also available using the commands `\tenss` and `\twelvess`.

Two special symbols, `\Tbf` and `\tbf`, are used to produce “Title bold face” and “small title bold face.” The first is used to print the title of a paper and titles of the chapters of a paper, and the second is used to print the titles of sections of a paper. You are free to change the definitions of these control words, but if you do not they default to 14 pt boldface for `\Tbf` and 12 pt boldface for `\tbf`.

Math italic bold (`\mib`) is listed with the other fonts but is actually different: it substitutes boldfaced versions of the math italic and math symbol fonts for the normal ones in math mode (`$...$`). These fonts are available only with with Computer Modern fonts and only in 10 pt and larger sizes; the standard math fonts are used for smaller superscripts. Since `\mib` effectively redefines `$..$`, it must be placed before the math material (i.e., outside the math quotes) and the whole formula will be bold. For example,

```
{\tbf\mib Observation of $Z^0 \to \mu^+\mu^-$}
```

will produce the indicated text in `\tbf` size with a bold formula, provided you have the right fonts. The easiest way to find out is to try it. Since the `\mib` fonts are not available everywhere, they are never loaded or used unless you explicitly ask for them.

Plain TEX includes a calligraphic script font known as `\cal`, but the letters in this font are not quite what most physicists are accustomed to using for things like Lagrangian or Hamiltonian densities or functional measures. Ralph Smith of U.C. San Diego has created a new “formal script” font with much better letters for such purposes, and TEXsis supports this font provided that it is available on your computer. To get this new font you simply use `\scr` in place of `\cal`. An example is:

```
$$
   {\scr L} = - \del^\mu \phi^\ast \del_\mu \phi + m^2 \phi^2
$$
```

Unfortunately we don't know if your computer has the `rsfs` fonts so we won't try to show you the output from this example. The quickest way to determine whether the fonts is available on your system is to try this example and verify that TEX does not complain about the font. If your system does not have the font we have made it available via anonymous ftp from lifshitz.ph.utexas.edu in the directory `texsis/rsfs`. If you don't have a Sun 4 computer you will have to use MetaFont on the `*.mf` files to create the fonts on your machine. We hope that someday the `rsfs` fonts will become a part of the standard TEX distribution.

When you use the macros `\tenpoint`, `\twelvepoint`, etc., the style of the type is reset to Roman, as if you'd also typed `\rm`. The baseline skip (the distance between lines) is also set to be suitable for the type size. Several commands are provided that vary the interline spacing, both in the text and in auxiliary material such as tables. These are:

`\doublespaced` makes the baseline skip 1.5 times its normal value. True double spacing looks too wide with typesetting.

`\singlespaced` resets the baseline skip to its normal value.

`\TrueDoubleSpacing`
makes the baselineskip twice its normal value. This usually looks too wide for a final document, but it is desirable in a manuscript being submitted to journals such as Physical Review.

`\triplespaced` makes the baseline skip 3 times its normal value. This should only be used for drafts of papers to leave room for notes or corrections.

`\widenspacing` increases the current spacing by a factor of 1.25. With single spacing this is the analog of $1\frac{1}{2}$ spacing on a typewriter. Repeating `\widenspacing` increases the spacing by 1.25 times whatever it already was; it does not reset the spacing to singlespaced first.

When TEXsis is first started the default is `\singlespaced`, but once TEXsis is initialized with the `\texsis` command (or `\preprint` or `\paper`) the spacing is set to `\doublespaced`.

When a paper is double spaced you may want to have the footnotes single spaced. Also, if the text is in 12 pt type you may nevertheless want to have the footnotes in 10 pt type. In TEXsis you can do this easily using `\Footnote` in place of `\footnote`, as in this example:[†]

```
... as in this example:\Footnote\dag{This is an example of
the use of {\tt\bs Footnote} in \TeXsis.}
```

`\Footnote` also puts the reference mark (in this case the dagger symbol `\dag`) in a superscript,[*] unlike `\footnote`. The typestyle used for the text of the footnote is controlled by `\FootFont`, which defaults to:

```
\def\FootFont{\tenpoint}
```

† This is an example of the use of `\Footnote` in TEXsis.

* But note that outside of math mode the asterisk character is already raised slightly, so it does not work well as a reference mark for `\Footnote`. Instead, you can use the asterisk from math mode, as in the following example: `\Footnote{$*$}{But note that outside of math mode... }`.

If you want to change this you can do so somewhere near the beginning of your document. The spacing between multiple paragraphs in a `\Footnote` (if you are prone to being so verbose) is set to zero, regardless of the `\parskip` in the main part of the document, but you can also change this by adding `\parskip=`⟨amount⟩ to your new definition of `\FootFont`. If you want a singlespaced footnote in the `\FootFont` typestyle but without putting a reference mark in the text you can use `\Vfootnote` just as you would use `\vfootnote` in Plain TEX. And of course if `\Footnote` or `\Vfootnote` cannot do what you need even after changing `\FootFont` then you can always use `\footnote` or `\vfootnote` from Plain TEX.[‡]

3.1. Automatic Quote Balancing

As you may recall from *The TEXbook*, one of the differences between typing and typesetting is the way in which quote marks are handled. On a typewriter you would simply use the non-oriented double-quote mark (`"`) both to begin and end the quoted material. In typesetting, however, quoted text begins with a left-quote mark (“) and ends with a right-quote mark (”). To produce these in TEX you have to type two single-quote marks, either ` `` ` or `''`. This is not very difficult to do, but it is easy to forget and use the double-quote marks instead. With TEXsis that's okay. If you say

```
\quoteon
```

then the next time a double-quote mark comes along it will be translated into a left-quote mark. The double-quote mark after that will become a right-quote mark, and so on, so that material enclosed in double-quote marks will be typeset correctly. All you have to do is remember to use the double-quote marks in pairs.

In some special cases you may not want TEXsis to be so smart about double-quotes. You can turn this feature off by saying

```
\quoteoff
```

The default though, once you have said `\texsis`, is `\quoteon`.

[‡] A technical footnote: With `\footnote` the group is closed before the footnote is typeset, so any line spacing you set is lost. You can solve this by putting a `\par` or `\vskip 0pt` at the end of your footnote, but then the final strut is put on a separate line. Getting footnotes to work just right is tricky business.

4. Equations

Plain TEX was designed to make it easy to typeset equations and mathematical formulae. TEXsis provides additional features that make the job even easier. Among these are the automatic numbering of equations, automatic sizing of parentheses, and a collection of special symbols that are used by physicists.

4.1. Equation Numbering

In Plain TEX equation numbers are added to displayed equations with the `\eqno` command, as described in Chapter 19 of *The TEXbook*. For example, typing

```
$$ E=mc^2.  \eqno (47) $$
```

will produce:

$$E = mc^2 \tag{47}$$

in the output. With TEXsis the procedure for getting automatically numbered equations is similar, except that you use `\EQN` instead of `\eqno`, and what follows this (up to the "`$$`") is not used as the number of the equation, but as an internal *label* for the equation. For example, if you had already used `\EQN` to create three numbered equations, then typing

```
$$ \vec F = -{G_N Mm \over r^2} \hat r
   \EQN Newton $$
```

would produce

$$\vec{F} = -\frac{G_N Mm}{r^2}\hat{r} \tag{4.4}$$

TEXsis automatically assigns the next available equation number to the equation and prints this number on the right, in parentheses. The equations that follow the one above would be numbered 4.5, 4.6, and so on.

The purpose of the "label" after the `\EQN` is to let you refer to the equation in the text without knowing the equation number that has been assigned to it. For example, to refer to Eq. (4.4) above, which has the label "Newton", you might type

```
The dependence of the force on the inverse square of the distance
can be seen in \Eq{Newton}.
```

This would be printed as:

> The dependence of the force on the inverse square of the distance can be seen in Eq. (4.4).

The control sequence `\Eq{`⟨label⟩`}` gives the same result as typing "`Eq.~(`*nn*`)`", where *nn* is the appropriate equation number for the given label. Two other similar control words allow you to refer to a range of equations, or to refer to an equation simply by number, without the preceding "`Eq.~`." Typing `\Eqs{`⟨label⟩`}` produces "Eqs. (*nn*)", while `\Ep{`⟨label⟩`}` gives just the equation number associated with the label, enclosed in parentheses. Thus typing

```
The class of metrics defined by \Eqs{cond-1} through
\Ep{cond-5} can all be shown to satisfy inequality
\Ep{triangle}, as well as the condition in \Eq{Killing}.
```

would produce in the output something like:

> The class of metrics defined by Eqs. (14) through (19) can all be shown to satisfy inequality (23), as well as the condition in Eq. (7).

Almost any sequence of characters can be used as a "label" for an equation. This includes letters, numbers, and punctuation marks, but it is best to avoid TEX's special characters, such as `\`, `&`, `}`, `{`, `$`, `_`, and `^`. Spaces may be included but will be ignored, and there *is* a difference between uppercase and lowercase letters. Also, the semicolon (`;`) is reserved for a special purpose, as described below.

You may want to give equations "names," as in the example called "Newton" above, or you can give the equations labels that are numbers. This is the best thing to do if the equations in the original manuscript already have numbers, but if you do this then you should be sure to remember that the internal label and the number which is printed (the "external" number) are not necessarily the same. If the equation *labeled* "15" is the fifteenth equation in the document then it will also be equation number 15, but if you move the equation around in the file so that it is the sixth equation in the document it will be printed as equation 6, even though it has the label "15".

If you have a heavily edited document and the arrangement of the numbers and labels gets confusing there is a useful aid to help straighten things out. If you say

```
\eqnotracetrue
```

then the equation numbers will be printed as usual, but with the label appearing next to it in the right margin in square brackets. For example, the equation

```
$$
(i\gamma^\mu \partial_\mu - m)\psi (x) = 0
   \EQN Dirac $$
```

when printed with `\eqnotracetrue` would appear as

$$(i\gamma^\mu \partial_\mu - m)\psi(x) = 0 \tag{4.5}$$ ["Dirac"]

Since the label appears in the margin the spacing of your document remains exactly the same. Obviously you would not print the final copy of the document this way, but it can be very useful for editing. You can turn off this equation tracing feature anytime by saying `\eqnotracefalse`. If you say `\draft` at the beginning of your document then the equation tracing will be turned on automatically (along with some other useful features).

Sometimes when several equations are closely related you will want to give the whole set the same equation number. To distinguish separate formulae in such a group you then generally add a letter to the number. You might, for example, have the equation

$$J^2|j,m\rangle = j(j+1)|j,m\rangle \tag{4.6a}$$

and then a short while later you might have the related equation

$$J_z|j,m\rangle = m|j,m\rangle \qquad (4.6b)$$

These may be referred to together as Eqs. (4.6) or may be referred to separately as Eqs. (4.6a) and (4.6b). To get an equation number with a letter in it you still use `\EQN`, but you add the appropriate letter to the end of the label, separated from the label by a semicolon. The equations above were created by typing:

```
$$
J^2\ket{j,m} = j(j+1) \ket{j,m} \EQN eigen;a
$$
```

and

```
$$
J_z\ket{j,m} = m \ket{j,m} \EQN eigen;b
$$
```

The semicolon is not printed — it is just used to separate the letter from the label. If the letter is "a" then the next available equation number is used, as usual, but if the letter is anything else then the equation number is not incremented and instead the number from the previous equation is used. You can use any sequence of letters you like, although it is usually best to use "a,b,c..." for consecutive equations.

To refer to equations that have both numbers and letters you can either use the letter to pick out a specific equation, or simply leave out the letter and use the label to refer to the whole lot all at once. You can use either `\Eq` or `\Eqs` to do this. For example, the sentence after Eq. (4.6b) above was created by typing

```
These may be referred to together as \Eqs{eigen}
or may be referred to separately as
\Eqs{eigen;a} and \Ep{eigen;b}.
```

You will eventually encounter an equation that is so long that it has to be broken up into several lines. Or you will want to print several equations together, lined up by their "=" signs. In Plain TEX you would use `\eqalign` to display these kinds of equations, or `\eqalignno` to display them with equation numbers. To get automatically assigned equation numbers in aligned equations with TEXsis you should use `\EQNalign` in place of `\eqalign` or `\eqalignno`. As usual the alignment of the equations is controlled by `&` and `\cr`. An equation number is assigned to a particular line by putting `\EQN` followed by a label on that line, before the `\cr`. The label is everything from the `\EQN` up to the `\cr`. For example,

```
$$
\EQNalign{
{dp \over dt} &= - {\partial H \over \partial q}
\EQN Hamilton-1 \cr
{dq \over dt} &= {\partial H \over \partial p}
\EQN Hamilton-2 \cr}
$$
```

produces the two aligned and numbered equations

$$\frac{dp}{dt} = -\frac{\partial H}{\partial q} \tag{4.7}$$

$$\frac{dq}{dt} = \frac{\partial H}{\partial p} \tag{4.8}$$

`\EQNalign` can be used both for typing multi-part equations or for splitting an equation which is too long to fit on a single line. However, when splitting a single long equation into several lines it may actually be better to use `$$\eqalign{`⟨formula⟩`}` followed by `\EQN` ⟨label⟩`$$`. For example, in

$$\begin{aligned} \mathcal{L}_{NC} &= \frac{-g}{2\cos\theta_W} \sum_i \bar\psi_i [(t_{3,i}-2Q_i\sin^2\theta_W) \gamma_\mu - t_{3,i}\gamma_\mu\gamma_5]\psi_i Z^\mu \\ &\quad - e \sum_i Q_i \bar\psi_i \gamma_\mu \psi_i A^\mu \end{aligned} \tag{4.9}$$

the equation number is centered between the two lines of the equation. This equation was produced using `\eqalign` by typing:

```
$$
\eqalign{
{\cal L}_{NC} &= {-g \over 2\cos\theta_W} \sum_i
   \bar\psi_i [ (t_{3,i}-2Q_i\sin^2\theta_W) \gamma_\mu
   - t_{3,i}\gamma_\mu\gamma_5 ]\psi_i Z^\mu \cr
&\quad - e \sum_i Q_i \bar\psi_i \gamma_\mu \psi_i A^\mu \cr}
\EQN{NC}
$$
```

Closely related equations can be displayed together with the same equation number by following the label with a semicolon and a letter, as already described above. For example, the set of equations,

$$\nabla \cdot \vec D = 4\pi\rho \tag{4.10a}$$

$$\nabla \times \vec E = -\frac{1}{c}\frac{\partial \vec B}{\partial t} \tag{4.10b}$$

$$\nabla \cdot \vec B = 0 \tag{4.10c}$$

$$\nabla \times \vec H = \frac{4\pi}{c}\vec J + \frac{1}{c}\frac{\partial \vec D}{\partial t} \tag{4.10d}$$

is produced by typing:

```
$$
\EQNalign{
\nabla \cdot \vec D    &=  4 \pi \rho \EQN EM;a       \cr
\nabla \times \vec E   &=
        - {1 \over c} {\partial \vec B \over \partial t}
```

```
        \EQN EM;b                                          \cr
\nabla \cdot \vec B     &=  0 \EQN EM;c                    \cr
\nabla \times \vec H    &=  {4 \pi \over c} \vec J
        + {1 \over c} {\partial \vec D \over \partial t}
        \EQN EM;d                                          \cr}
$$
```

Note that the alignment of the equations is controlled by `&` and `\cr`, not by the arrangement used when the equation was typed into the manuscript file. Still, it is recommended that you arrange long sets of equations like these in some sort of clear pattern when you type them. This will both reduce errors when the equation is typed and make it easier for you to read the file when you are looking at it later. Remember, TEX ignores all spaces and tabs when it reads material for equations or displayed equations.

`\EQNdoublealign` is similar to `\EQNalign` but makes equations with two alignment tabs. For example,

```
$$
\EQNdoublealign{
\sigma_i^2 &= 1 & \EQN Pauli;a \cr
\sigma_i\sigma_j + \sigma_j\sigma_i &= 0 \qquad & (i \ne j)
  \EQN Pauli;b \cr
\sigma_i\sigma_j &= i\sigma_k \qquad & (i,j,k\ {\rm cyclic})
   \EQN Pauli;c \cr}
$$
```

produces

$$\sigma_i^2 = 1 \tag{4.11a}$$
$$\sigma_i\sigma_j + \sigma_j\sigma_i = 0 \qquad (i \ne j) \tag{4.11b}$$
$$\sigma_i\sigma_j = i\sigma_k \qquad (i,j,k \text{ cyclic}) \tag{4.11c}$$

with both the equal signs and the parentheses aligned. The second `&` is required in the first line, because without it the `\EQN` would have aligned the equation number with the parentheses.

Plain TEX also lets you display formulae on several lines without any alignment by using `\displaylines`. With TEXsis you can use the similar control sequence `\EQNdisplaylines` to get an automatically assigned equation number for any line in such a display. The pattern is the same as in `\displaylines`, but you simply insert `\EQN`⟨label⟩ in front of the `\cr` on the line you want numbered. `\EQNdisplaylines` produces less space between the equations than would result from typing them separately, and generally looks better.

All of the equation alignment macros are special cases of the plain TEX macro `\halign`. Therefore, if necessary, the vertical spacing can be adjusted by inserting

`\noalign\vskip`⟨dimen⟩

where ⟨dimen⟩ can be either a positive or a negative dimension.

In a long document there are several different ways to number the equations. You could just start at "1" and count up for each new equation. Sometimes though, it is better to number the equations consecutively within each chapter or section of a document. In TEXsis the default form for an equation number is *cc.ss.nn* where *cc* stands for the chapter number, *ss* is the section number, and *nn* is the equation number within that section. Whenever you move to a new section or chapter (using `\section` or `\chapter`, as described in Section 6) the number *ss* or *cc* is incremented by one and the number *nn* is reset to one. If you are not using chapter divisions in your document then the chapter number *cc* is not made a part of the equation number, and the number *nn* is reset only at the beginning of a section. If you are not using section divisions in your document then the section number *ss* is not made a part of the equation number, and the number *nn* is reset to one only at the beginning of a chapter. If you use neither chapter nor section divisions in your document then neither the chapter number *cc* nor the section number *ss* will be made a part of the equation number, and the equations will simply be numbered consecutively, 1, 2, 3. . . .

Although the system just described for forming equation numbers gives a result that is usually what you will want for your equation numbering you can change the method by setting either of two switches. If you say `\showchaptIDfalse` then the chapter number *cc* will not be made a part of the equation number even though you may be using chapter level divisions in your document. Similarly, saying `\showsectIDfalse` prevents the section number *ss* from being used in the equation number. The opposites of these two switches are `\showchaptIDtrue` and `\showsectIDtrue`. The equation number *nn* is reset to one only when a section or chapter number that is supposed to be visible is incremented, as you might expect.

4.2. Automatic Parentheses Sizing

Plain TEX provides automatic adjustment of the sizes of parentheses and other delimiters when you use `\left` and `\right`, as in

`$$ \left(` ⟨math material ⟩ `\right) $$`

TEXsis provides similar automatic sizing of parentheses and square brackets in displayed equations without the need of typing `\left` and `\right`. You must still use `\left` and `\right` for all other delimiters, and they can be used for parentheses and square brackets if desired. This feature is turned on and off with `\autoparens` and `\offparens` respectively; the default, once you say `\texsis`, is `\autoparens`. Thus, for example,

```
\autoparens
 $$
( a \over b )
 $$
```

produces

$$\left(\frac{a}{b}\right)$$

whereas

```
\offparens
 $$
( a \over b )
 $$
```

produces

$$\frac{(a}{b)}$$

which is not what one normally desires.

The automatic sizing of parentheses works by making `)`, `(`, `]`, and `[` active characters in displayed equations and defining them to be macros which use `\left` and `\right` to size the parentheses. While this normally works properly and simplifies the typing of equations, it will fail if the `\left` and `\right` operators are not properly balanced, such as when a long equation is split between two or more lines with `\eqalign` or `\EQNalign`. Almost all such problems can be solved by using the control sequences `\lparen`, `\rparen`, `\lbrack`, and `\rbrack`, which are taken from Plain TEX and which give just the standard parenthesis and bracket characters. For example,

$$\begin{aligned} F &= \Biggl[\left(\frac{a}{b}\right) + \left(\frac{c}{d}\right) \\ &\quad + \left(\frac{e}{f}\right) + \left(\frac{g}{h}\right) \Biggr] \end{aligned}$$

can be produced with

```
$$
\EQNalign{
F &= \Biggl\lbrack (a \over b) + (c \over d) \cr
&\quad + (e \over f) + (g \over h) \Biggr\rbrack \cr}
$$
```

Note that in this example the parentheses are balanced on one line and so can be sized automatically, while the brackets span two lines and so must be sized "by hand." The `\vbig` macro described in the next section provides additional flexibility for sizing.

If all else fails, you can use `\offparens` to turn off the automatic sizing of parentheses and put in `\left` and `\right` or explicit sizes yourself. In any case, you must put `\left` and `\right`, if needed, in in-line equations (math expressions which are not displayed equations). Note also that previous versions of TEXsis used the control sequence `\onparens` to provide automatic parentheses sizing in *all* math-mode expressions. While you can still use, this we do not encourage it since it may cause problems with other uses of these characters.

4.3. Special Symbols

TEXsis defines some special symbols and other constructions which are not included in Plain TEX and which either are commonly used in high energy physics or are not trivial to construct. The following macros, which are listed here by related groups, provide these additional special symbols for use in equations:

- `\bra{`⟨text⟩`}` and `\ket{`⟨text⟩`}` produce the notations $\langle text|$ and $|text\rangle$ respectively for quantum mechanical states. `\vev{`⟨text⟩`}` produces $\langle text\rangle$ for a VEV (Vacuum Expectation Value).
- `\del`, `\grad`, and `\square` produce the notations ∂, ∇, and $\Box$ for various derivatives.
- `\frac{`⟨numerator⟩`}{`⟨denominator⟩`}` produces the fraction `{`⟨numerator⟩ `\over` ⟨denominator⟩`}`, while `\smallfrac{`⟨numerator⟩`}{`⟨denominator⟩`}` produces the same fraction in the smaller subscript style type. `\half` produces a $\frac{1}{2}$ in a size appropriate to its usage.
- `\simle` and `\simge` make the symbols $\lesssim$ and $\gtrsim$ to indicate "less than about" and "greater than about," respectively. Synonyms for these are `\ltsim` and `\gtsim`.
- `\buildchar{`⟨#1⟩`}{`⟨#2⟩`}{`⟨#3⟩`}` gives character `#1` with `#2` above it and `#3` below it, allowing the construction of a variety of special symbols. `\overcirc` is a special case of this which puts a small circle above a character, as in $\overset{\circ}{R}$. This is often used in general relativity and related fields.
- `\parenbar` puts a bar in parentheses above a character; this is often used to indicate either a particle or its antiparticle. `\nunubar` and `\ppbar` are special cases. Thus

  ```
  $$ n \parenbar{n} \qquad \nu \nunubar \qquad p \ppbar $$
  ```

  gives

  $$n\overset{(-)}{n} \qquad \nu\overset{(-)}{\nu} \qquad p\overset{(-)}{p}$$

- `\subrightarrow{`⟨text⟩`}` puts the text under a right arrow of the appropriate length. Thus

  ```
  $$ f(x) \subrightarrow{x \to 0} f(0) $$
  ```

  gives

  $$f(x) \xrightarrow[x\to 0]{} f(0)$$

- `\slashchar{`⟨letter⟩`}` puts a slash through a character, the notation for the contraction of a vector with Dirac γ matrices in Feynman graphs. For example

  ```
  $$ {\cal L} =
      \bar\psi[i\slashchar{\partial} - e\slashchar{A}-m]\psi $$
  ```

gives

$$\mathcal{L} = \bar{\psi}\,[i\not\partial - e\not A - m]\,\psi\,.$$

One thing this example demonstrates is that the results produced by `\slashchar` may be close to what you want but may not look as good as you would hope for. To really get a good looking slashed character you would have to define a control sequence for each individual character. For example, a slightly better slashed A can be had by defining

```
\def\Aslash{A\kern -0.47em/}
```

which produces "$A\!\!\!/$". The amount to kern back for the slash will be different for each character. Note that the angle of the slash is still quite what one would like for the slashed "A." To really get good looking slashed characters someone will have to create a new slashed character math font using MetaFont. Meanwhile you can use `\slashchar` or define your own using kerning.

- `\vbigl{`⟨delim⟩`}{`⟨size⟩`}` and `\vbigr{`⟨delim⟩`}{`⟨size⟩`}` produce very (or variably) big delimiters, generalizing the Plain TEX macros `\bigl`, `\Bigr`, etc. They produce, respectively, the left and right delimiters ⟨delim⟩ with a size ⟨size⟩ which can be given in pt., cm., in., or any other valid units. `\Leftcases{`⟨size⟩`}` and `\Rightcases{`⟨size⟩`}` give left and right braces of size ⟨size⟩ smashed to zero height; these can be used in place of the `\cases` of Plain TEX when it is desired to put an equation number on each line. For example,

```
$$\EQNalign{
m_e    &= ~.511~\MeV \EQN lep;a \cr
m_\mu  &= ~105~\MeV
          \quad \Rightcases{50pt} \quad {\rm Leptons}
          \EQN lep;b \cr
m_\tau &= 1784~\MeV \EQN lep;c \cr}
$$
```

gives

$$\left.\begin{aligned} m_e &= \ \ .511\ \ \mathrm{MeV} \\ m_\mu &= \ \ 105\ \ \mathrm{MeV} \\ m_\tau &= 1784\ \ \mathrm{MeV} \end{aligned}\right\}\quad \text{Leptons} \qquad \begin{aligned} &(4.12a) \\ &(4.12b) \\ &(4.12c) \end{aligned}$$

Note that `\Leftcases` and `\Rightcases` do not provide automatic spacing, which must be adjusted by the user as in the previous example. There seems to be no simple way to obtain automatic spacing and to allow an equation number on each line.

Abbreviations for units are normally set in Roman type even when used in formulas. Special function (e.g., sine, cosine and logarithm) or operator names are also set in Roman type, and TEXsis adds some new names to go with the old standards.

- `\Tr` and `\tr` give "Tr" and "tr" respectively for the trace of an operator or of a matrix.
- `\real` and `\imag` give the symbols Re and Im for the real and imaginary parts of an expression; the spacing is like that for any other function.
- `\eV`, `\KeV`, `\MeV`, `\GeV`, and `\TeV` give the abbreviations for energy units eV, KeV, MeV, GeV, and TeV respectively.
- `\mb`, `\mub`, `\nb`, and `\pb` give the abbreviations for cross section units mb, μb, nb, and pb respectively.
- `\cmsec` gives $\mathrm{cm}^{-2}\mathrm{sec}^{-1}$, the standard units for luminosity in colliding beam accelerators.

Note that all of the abbreviations above are defined without any spacing; to produce

$$\mathcal{L} = 10^{33}\,\mathrm{cm}^{-2}\mathrm{sec}^{-1}, \qquad p_\perp = 8.3\,\mathrm{GeV}$$

you should type

```
$$
{\cal L} = 10^{33}\,\cmsec, \qquad p_\perp = 8.3\,\GeV
$$
```

with a `\,` to make a thin space.

5. References and Citations

TEXsis includes a set of macros for making citations to the literature in the text of a document. References are processed in the standard format used by the American physics journals: they are automatically assigned consecutive numbers as they appear in the text and are listed in numerical order at the end of the document. In the text a citation appears as a superscript number corresponding to the reference cited, although it is possible to change this so that the reference number appears in-line in the text in square brackets, as is done in some European journals. When each reference is defined it is given a *label*, which is associated with the number assigned to the reference. Subsequent citations to the same reference are made using this label; there is no need to know the particular number assigned to the reference, and indeed this number may change as the document goes through editing and revision.

TEXsis is flexible enough that the text of the references may be entered in two different ways, either all references at once at the beginning of the document or each one individually at the point in the text where the work is first cited. In either case TEXsis stores the text of all of the references in numerical order until it is asked to print the list at the end of the document by saying `\ListReferences`.

To make a citation to a new reference and to enter the text of the reference use the control sequence `\reference{`⟨label⟩`}`, where ⟨label⟩ is the label to be used for the number referring to this reference. This is followed by the text of the reference, which is terminated by `\endreference`. An example is:

```
To this end we will follow Feynman\reference{Feynman, 1949}
R.P.~Feynman, Phys.\ Rev.\ \vol{76}, 749 (1949)
\endreference
in developing the propagator approach.
```

This would appear in the text of the document as,

> To this end we will follow Feynman[7] in developing the propagator approach.

and the text of the reference would be saved until the list of references is printed at the end of the document.

There are several things to take note of in the example above. Most importantly, there is no space between the word in the text where the citation is made ("Feynman") and the `\reference`. This is so the superscript reference number will appear in the text right after the word to which the citation is attached, with no intervening space. Note also in the reference text the tie (`~`) between the initials and the name, the control spaces (`\ `) after the abbreviations, and the use of the `\vol` macro for the volume number. `\vol{`⟨number⟩`}` prints the volume number ⟨number⟩ with an underscore below it (this is easier to read than boldfaced volume numbers on small laser printers, but if you prefer something else you can simply redefine the `\vol` macro). Similarly, `\booktitle{`⟨title⟩`}` will print the title of a book, in slanted type. Finally, note that there is no punctuation at the end of the reference text. This is because TEXsis automatically supplies the appropriate punctuation when it prints the list of references. Normally this will simply be a period (.), but it is possible to have multiple references assigned to the same reference number, in which case the references are separated by semi-colons, with the last reference terminated by a period. How to do this is described below.

Since typing the text of a journal citation is somewhat complicated the `\journal` macro was created to make things a little easier. The format is

`\journal` ⟨journal name⟩`;`⟨volume⟩`,`⟨page number⟩`(`⟨year⟩`)`

This constructs the journal citation for you, with periods automatically replaced by "`.\ `" in abbreviations and the volume number underlined for you by `\vol`. It is important that the field separators, the semi-colon, the comma and the parentheses, all be present, or TEX will look as far forward as it takes to find one. They must also be in the right order, and you will find that you don't want any spaces between anything.

Besides making it a little easier to type a citation to a journal article, using `\journal` lets you switch from the American form of citation to the European form very easily. In the American form the page number follows the volume, followed by the year in parentheses. In the European form the year follows the volume number and the the page number comes after that. Normally TEXsis uses the American form, but you can change to the European form by simply saying

```
\Eurostyletrue
```

The `\journal` arguments are always in the American order.

To enter several references with the same reference number simply begin the next `\reference` immediately after the previous `\endreference` and begin the labels of all but the first reference with an asterisk "`*`", as in the following example:

```
The vacuum polarization term can be regularized
by introducing a cut-off at high
frequencies.\reference{Pauli and Villars, 1949}
W.~Pauli and F.~Villars, \journal Rev. Mod. Phys.;21,434(1949)
\endreference
\reference{*Schwinger, 1948}
J.~Schwinger, \journal Phys. Rev.;74,1439(1948)
\endreference
```

This produces the single citation

> The vacuum polarization term can be regularized by introducing a cut-off at high frequencies.[8]

The "`*`" at the beginning of the second label tells TEXsis not to assign a new reference number to this reference, but instead just to add it to the text of the previous one. Of course you could type the text of both references between one `\reference` and `\endreference`. An advantage to following the example above is that the second reference can be given a reference number of its own simply by deleting the "`*`" at the beginning of the label. Similarly, two references with different reference numbers can be lumped together into one citation by inserting a "`*`" in front of the label of the second reference. Another advantage to keeping the references separate is that it is easy to move references around, either within a document that is being revised or between documents. It is therefore recommended that each reference be entered separately, although this is obviously not required.

Instead of putting the `\reference ...\endreference` in the text it may be more convenient to type them all together, especially if they already appear that way in the manuscript. This can be done without producing any unwanted printed numbers by using

```
\referencelist
\reference{1} text 1... \endreference
\reference{2} text 2... \endreference
...
\reference{last} last text... \endreference
\endreferencelist
```

The `\referencelist` must appear at the beginning of the document, so that the reference labels are defined before they are used. This is best put into a separate file which is `\input` by the master file. Of course the list of references is not printed until the end of the document, when `\ListReferences` is called.

If `\reference ...\endreference` is used in the text of the document, then the reference number is printed there. The default format is a superscript (like this[137]), as is standard in most American journals. The command `\superrefsfalse` will cause the reference numbers to be printed in square brackets (like this [137]), the style used in most European journals. `\endreference` looks ahead to see if the next token is another `\reference`; if so, then a range of numbers (like this[137–140]) is printed. If a citation appears at the end of a sentence, it looks best if it comes after the punctuation for superscript references but before it for references in brackets. You will have to arrange for that yourself.

All subsequent citations of a previously defined reference are made using its label. If a `\referencelist` is used, then all citations appearing in the text will be made in this way, since then the initial reference definitions print nothing. The construction `\cite{`⟨label⟩`}` prints the reference number corresponding to the label as a superscript (or in square brackets if `\superrefsfalse` is selected.) For example,

```
\TeX\ is described by Knuth.\cite{Knuth} He explains\dots
```

produces

TeX is described by Knuth.[1] He explains...

since *The TeXbook* is in fact Ref. 1 of this document and has the label "`Knuth`." Note that some extra space is inserted after the reference number, just as it usually is at the end of a sentence. If several `\cite`'s are given in succession then the corresponding numbers are printed separated by commas. Unlike `\reference`, which defines the reference numbers, `\cite` does not know if references are sequential, so it cannot replace a sequence of numbers by a range. To avoid a long sequence of reference numbers use instead `\refrange{`⟨first label⟩`}{`⟨last label⟩`}` to print the numbers corresponding to the first and last labels as a range.

The construction `\Ref{`⟨label⟩`}` produces "Ref. *nn*" in the text, where *nn* is the number of the reference corresponding to the label. For example, the "Ref. 1" in the previous paragraph was typed in the manuscript as `\Ref{Knuth}`. `\Ref` is simply shorthand for `Ref.~\use{Ref.`⟨label⟩`}`, and you can change what is printed simply by changing the definition of `\Ref`. For example,

```
\def\Ref#1{reference~\use{Ref.#1}}
```

Then `\Ref{Knuth}` would produce "reference 1" instead. For more on how to use `\use` see Section 12.

After the text of the document is complete a printed list of the references can be obtained by saying `\ListReferences`. This prints the numbers at the left hand edge of the page with the text indented by `\refindent`, which is chosen so that the equation numbers line up, and it puts a `\refskip` (initially equal to a `\smallskip`) between references. The reference format can be changed by redefining the command `\refFormat`, which is executed just before the references are printed and which initially is simply `\relax`. For example, to have the reference list printed with a ragged right margin one could say

```
\def\refFormat{% format for the list of references
  \def\refskip{\vskip0pt plus 2pt}% % no space but some stretch
  \raggedright}                     % ragged right margin
\section{References}                % heading
\ListReferences                     % do it
```

No heading is printed for the reference list — you have to do this yourself in whatever format is appropriate. Usually you will want to use `\nosechead`, but sometime you may want to use `\chapter` or `\section` from Section 6 instead.

6. Chapters and Sections

Long documents are typically divided into chapters, sections, subsections, and/or appendices. The following macros can be used to create and automatically number such divisions of a document:

`\chapter{`⟨title⟩`}`
: Begins a chapter with ⟨title⟩ printed in `\Tbf` type on a new page.

`\section{`⟨title⟩`}`
: Begins a section with the ⟨title⟩ printed in `\tbf` type. This does *not* begin on a new page (although you certainly can start a section on a new page by saying `\vfill\supereject` first.)

`\subsection{`⟨title⟩`}`
: Begins a subsection with the ⟨title⟩ printed in `\bf` type.

`\subsubsection{`⟨title⟩`}`
: Begins a sub-subsection entitled ⟨title⟩. This is a lower level than even a subsection and should only be used when truly appropriate.

`\Appendix{`⟨label⟩`}{`⟨title⟩`}`
: Begins a chapter-like appendix labeled by the letter ⟨label⟩ (generally A, B, C, ... or I, II, III, ...). The ⟨title⟩ is printed in `\Tbf` type at the top of a new page.

`\appendix{`⟨label⟩`}{`⟨title⟩`}`
: Begins a section-like appendix labeled by the letter ⟨label⟩ (generally A, B, C, ... or I, II, III, ...). The ⟨title⟩ is printed in `\tbf` type.

`\nosechead{`⟨title⟩`}`
: The ⟨title⟩ is typeset like a subsection headline, but without a subsection number. This is suitable for acknowledgments, lists of references and such.

The title appears at the top of the section or subsection, left justified in the appropriate typestyle and with an appropriate amount of vertical spacing below. In general `\chapter` is appropriate mainly for books or theses; the manual you are reading now uses only `\section` and `\subsection` divisions.

By default chapter, section, and subsection numbers are printed with the titles, and the chapter and section numbers are also attached to equation, figure and table numbers in the form "*cc.ss.nn*", where *cc* is the chapter number, *ss* the section number, and *nn* the equation number. The inclusion of the chapter and section numbers can be turned on or off with the commands

`\showchaptIDtrue` Display chapter numbers.

`\showchaptIDfalse`	Do not display chapter numbers.
`\showsectIDtrue`	Display section numbers.
`\showsectIDfalse`	Do not display section numbers.

`\Appendix` and `\appendix` give, respectively, chapter-like and section-like appendices. The title is printed in the same format as a chapter or section, and the chapter or section number is replaced by a letter given as the first argument. This letter is attached to the equation numbers if `\showchaptIDtrue` or `\showsectIDtrue` is used. You can omit the letter by using an empty first argument, as in

```
\appendix{}{Leading logarithm approximation}
```

in which case the result is the same as setting `\showchaptIDfalse` or `\showsectIDfalse`.

A skip `\sectionskip` is used before the title of any section-like division, that is, a `\section` or an `\appendix`; a skip `\subsectionskip` is used for any lower-level divison, that is, a `\subsection`, a `\subsubsection`, or a `\nosechead`. Of course these are all removed by TeX if the section starts on a new page. Furthermore, any of these divisions will start on a new page if less than a minimum length `\sectionminspace` remains on the current page. The defaults for these parameters are

```
\sectionskip=2cm plus8pt minus8pt
\subsectionskip=1cm plus4pt minus4pt
\sectionminspace=0.25\vsize
```

but they can be changed to any legal TeX skip or dimension either throughout the document or in a local group. For example, you could use

```
\begingroup
\subsectionskip=\sectionskip
\fourteenpoint
\nosechead{<title>}
\endgroup
```

to make a `\nosechead` which looks like a `\section`.

Ordinarily the order of appearance of the sections of a document are not changed, so there is no automatic mechanism for labeling the sections. Any of the divisions described above can be labeled by inserting `\label{`⟨label⟩`}` into the title text. The ⟨label⟩ should have the form ⟨type.name⟩, where ⟨type⟩ is `chap`, `sect`, etc., to avoid possible conflicts with other kinds of labels. The corresponding numbers can then be referred to with `\use{`⟨type.name⟩`}`.

If the title of a chapter or section is too long a break in the line can be inserted with `\n`. This will also break the line in the title when it appears in the table of contents. If running headlines are being used (see Section 7.2) then only the first part of the title, up to the `\n`, will be used for the running headline. You can always set the headline by hand with `\setHeadline`; see Section 7.2.

You can customize the behavior of the section-making macros by adding your own definitions for the macros

`\everychapter`, `\everysection`, `\everysubsection`, `\everysubsubsection`,

which are called immediately before making the corresponding division of the document, or for

`\afterchapter`, `\aftersection`, `\aftersubsection`, `\aftersubsubsection`,

which are called immediately afterwards. By default all of these macros do nothing. As an example, if you wanted every section to begin on a new page you could define:

```
\def\everysection{\vfill\supereject}
```

For more specialized customization one can look in the source file `TXSsects.tex` for the definitions of `\section` and `\chapter` and modify these accordingly.

6.1. Table of Contents

The section-making macros just described automatically make an entry in a table of contents for the document being prepared. The title of the chapter or section and the page number are stored in a table of contents file (which ends with the extension "`.toc`") until needed. The table of contents is created simply by saying `\Contents`. If your document is divided into chapters, like a book, the table of contents should appear on a page with a Roman numeral page number (which is obtained by setting `\pageno` to a negative value). For example:

```
\book                    % initialize
\input ...               % input chapter or section files
...
\vfill\supereject        % eject last page, including figures
\nosechead{Contents}     % label TOC without section #
\pageno=-3               % print on pages iii, iv, ...
\Contents                % generate table of contents
```

The table of contents should be printed at the *end* of the document so that all the page numbers are correct. You then simply move these pages to the front of the document.

Nothing is written to the table of contents file unless `\ContentsSwitchtrue` is set. The default for most documents is `\ContentsSwitchfalse`, since one usually does not want a table of contents for a short paper or a memo or a letter. The exceptions are `\book` and `\thesis`, which automatically set `\ContentsSwitchtrue`. In any case, if you want a table of contents you can explicitly state `\ContentsSwitchtrue`.

While an entry in the table of contents is created by `\chapter` or `\section`, no entry is made for `\nosechead`. In this case, or in any case, you can insert an entry into the table of contents "by hand" by saying

`\addTOC{`⟨level⟩`}{`⟨title⟩`}{`⟨page⟩`}`

where ⟨level⟩ is the level of division for the entry being made, with 0 for chapters, 1 for sections and 2 for subsections. This controls how much vertical space is put around the table of contents entry and how much the entry is indented. The ⟨title⟩ is the text of the title as it should appear in the table of contents. Text which is too long for one line will be split over several lines, but you can force a line break where you want one with `\n`. The ⟨page⟩ is the page number for the entry you are making. Generally you'll use `\folio` here to just use the current page number, whatever that may be. Thus to make a section level entry in the table of contents having the title "Acknowledgments" you would simply say

```
\addTOC{1}{Acknowledgments}{\folio}
```

Entries made in the table of contents by `\chapter`, `\section`, etc. will optionally begin with the chapter or section number, depending upon whether `\showchaptIDtrue` and `\showsectIDtrue` are selected. The decision whether or not to show the chapter or section number is made when the table of contents is created by `\Contents`, not when the `\chapter` or `\section` is begun. Thus it is possible to have the section numbers showing at the beginning of the section, but to then have no section numbers in the table of contents by saying `\showsectIDfalse` immediately before saying `\Contents`.

What if you want the table of contents to appear in the table of contents itself!? Naturally it should appear first (even though it is printed at the end of the document). You can use the tagging mechanism described in Section 12 to make a "forward reference" to the page number. At the beginning of the document you can create the table of contents entry by saying

```
\addTOC{0}{Table of Contents}{\noexpand\use{Pg.tabcontents}}
```

The `\noexpand` prevents the `\use` from being expanded until later, after the table of contents file is read in. Later, when you actually print the table of contents you would first tag the page number, then invoke `\Contents`, like so:

```
\nosechead{Table of Contents}
\tag{Pg.tabcontents}{\folio}
\Contents
```

For more information about `\tag` and `\use` see Section 12.

7. Document Formats

The overall format of a document can be changed to any of a number of general layouts by replacing the basic initialization command `\texsis` with one of the following:

`\paper`	Produces a paper with the title and abstract at the top of the first page.
`\preprint`	Produces a paper in a "pre-print" format, with the title material, abstract and a banner on the first page. The text begins on the second page.
`\book`	For a book or other long document with a separate title page, running headlines, a table of contents, etc. Page numbers are printed in opposite corners on left and right handed pages.
`\Manuscript`	Similar to `\preprint`, but with true double spacing for manuscripts submitted for publication in The Physical Review.
`\letter`	A letter format very much like the letter-making macros in the back of *The TEXbook*. This is described in more detail in Section 10.
`\memo`	A format for making memoranda. Described in more detail in Section 10.
`\referee`	A format similar to `\memo` but for making Referee reports. See Section 10.
`\slides`	A format for slides or transparencies, using larger type and wide line spacing.

You can also begin your document with `\texsis` and *then* invoke one of the formats above, and this is actually the preferred way to do it, because it labels the file as a TEXsis document rather than a Plain TEX file. That way if you send the file to someone else they immediately know that they need to run it through TEXsis to print the paper, not something else.

A number of even more specialized document formats, suitable for conference proceedings or submissions to specific journals, are described later in Section 7.3. The general idea is that you can create a document using any one format and then switch to a different format simply by changing one instruction at the beginning of the document. You could write and edit the paper using `\paper`, then create a preprint for initial distribution using `\preprint`, or create a camera-ready copy for a conference proceedings using `\nuclproc`.

The `\slides` format at the end of the list above is more specialized and is intended for making overhead transparencies or similar material with 24 pt. type and ragged right and bottom margins as the default. The default size is for $8\frac{1}{2}$ in $\times$ $10\frac{1}{2}$ in transparencies in the "portrait" orientation. `\slides` also defines `\bl` to insert a blank line and `\np` to create a new page, since these are often used for slides. At least one of the authors of TEXsis believes that it is often better to make neatly lettered transparencies by hand using colored pens than to use a computer and laser printer, but to each his own...

7.1. The Title Page

The `\preprint` and `\paper` formats differ mainly in their treatment of the title material. `\preprint` produces a separate title page headed by a banner (made by `\banner`) and starts the text on the second page. `\paper` puts the title material at the top of the first page in a format appropriate for a generic conference proceedings. For all formats the title, the author(s), the abstract, and related material can all be automatically formatted using the following macros:

`\titlepage`
: Begins the "title page." All of the title material (title, authors' names and addresses, abstract, etc.) should be enclosed between `\titlepage` and `\endtitlepage`.

`\title`
: Begins the title of the paper. All lines up to a following `\endtitle` are centered and printed in `\Tbf` type (see Section 6). Line endings in the input file *are* respected, just as with `\obeylines`.

`\author`
: Begins an entry for an author's name and address, which should appear on the lines that follow. Line endings in the input file *are* respected. The first line to follow is the author's name, which is printed centered in `\bf` type. All subsequent lines are centered in normal type. End with `\endauthor`.

`\and`
: Prints the word "and," centered and with appropriate vertical separation. It can be used either between sets of `\author` ... `\endauthor` or between multiple addresses for the same author.

`\authors`
: An alternative to `\author` for papers with many authors. Here line endings are not respected; instead, the list of authors is divided into lines as evenly as possible, and each line is centered and printed in Roman type. End with `\endauthors`.

`\institution{`⟨symbol⟩`}{`⟨address⟩`}`
: Prints the symbol and the institution address centered on a line. A `\medskip` is placed before the first address to separate the names and addresses. `\institution` should be placed after the author list but before `\endauthors` to work correctly.

`\abstract` Begins the abstract. The word "ABSTRACT" is centered above the abstract. The left and right margins are brought in relative to the text of the paper. End with `\endabstract`.

`\pacs{`⟨codes⟩`}`
Prints the PACS (Physics and Astronomy Classification Scheme) codes on the title page.

`\submitted{`⟨Journal⟩`}`
Prints the phrase "Submitted to *Journal*" centered on the page, with the name of the journal printed in `\sl` type.

`\toappear{`⟨Journal⟩`}`
Similar to `\submitted`, this prints the phrase "To appear in *Journal*" on the title page, with the name of the journal printed in `\sl` type.

`\disclaimer{`⟨contract⟩`}`
Prints a standard DOE disclaimer for contract number ⟨contract⟩, printed in 10 pt. type, as a footnote.

`\endtitlepage`
Ends the title material begun with `\titlepage`.

The names `\titlepage` and `\endtitlepage` are actually somewhat misleading, since sometimes the title material does not appear on a separate page. They simply serve to mark off the material that normally appears at the beginning of a document. Also, if you produce a document with, say, the `\paper` format and want the title material to appear on a separate page you can simply change `\paper` to `\preprint` or `\Manuscript`.

The standard `\footnote` macro cannot be used to make footnotes to titles and authors because these set up special environments using internal vertical mode where footnotes are not allowed. You should instead use the `\vfootnote` macro of Plain TeX for such purposes. In this case you must put the reference mark in the title or author list yourself (for example, with `${}^*$`), and then say `\vfootnote*{The footnote}` somewhere else on the page, outside of the internal vertical mode material (the author or title blocks).

"Footnotes" identifying institutions in a long list of authors are often centered on lines immediately below the author list. `\authors` and `\institution` can be used for this. For example, the author list of a famous paper[9] could be typed as follows:

```
\authors
E.D.~Bloom,$^a$ M.~Breidenback,$^b$ D.H.~Coward,$^a$
H.~DeStaebler,$^a$ J.~Drees,$^a$ J.I.~Friedman,$^b$
G.C.~Hartmann,$^{b,c}$ H.W.~Kendall,$^b$ G.~Miller,$^a$
L.W.~Mo,$^a$ R.E.~Taylor$^a$
\institution{a}{Stanford Linear Accelerator Center,
   Stanford, CA 94305}
\institution{b}{Department of Physics and Laboratory for Nuclear
```

```
Science, Massachusetts Institute of Technology,
   Cambridge MA 02139}
\institution{c}{Xerox Corporation, Rochester, NY}
\endauthors
```

This produces

E.D. Bloom,[a] M. Breidenback,[b] D.H. Coward,[a] H. DeStaebler,[a] J. Drees,[a]
J.I. Friedman,[b] G.C. Hartmann,[b,c] H.W. Kendall,[b] G. Miller,[a] L.W. Mo,[a] R.E. Taylor[a]

[a] Stanford Linear Accelerator Center, Stanford, CA 94305

[b] Department of Physics and Laboratory for Nuclear Science,
Massachusetts Institute of Technology, Cambridge MA 02139

[c] Xerox Corporation, Rochester, NY

Note that `\institution` must come *before* `\endauthors`.

In the `\preprint` format `\titlepage` automatically puts a banner with the name of your organization at the top of title page. The name of the institution can be changed by saying

`\def\ORGANIZATION{`⟨name⟩`}`

and this change can be made permanent by putting it into the file `TXSsite.tex` before compiling TEXsis. The default at this installation is "PARKS COLLEGE". The preprint date and document number can optionally be specified by

`\pubdate{`⟨date⟩`}`
`\pubcode{`⟨code-number⟩`}`

before the `\titlepage`. If these are omitted, the date defaults to the current month and year, and the document number defaults to the current TEXsis version number, "Parks-Phy ".

If you are submitting a paper to a mail server such as `hep-lat`, `hep-ph`, or `hep-th` you can have the server automatically insert the preprint number it assigns your document when it is submitted. To do this put

`\pubcode{hep-lat/yymmnnn}`

near the top of the document (change "hep-lat" as appropriate to the server you are using). When you submit a paper to a server with the "put" command it will translate any occurance in the input file of the sequence "`hep-lat/yymmnnn`" to "`hep-lat/`" followed by the 7 digit paper number. You must use the exact string of characters "`yymmnnn`" for this to work. When someone else gets your paper from the server and prints a copy it will have the assigned document code in the banner at the top of the title page.

Here is an example of the begining of a TEXsis manuscript file showing how the macros described above can be used to generate the title material at the beginning of a paper:

```
\texsis
\preprint
\pubdate{July 8, 1957}
\pubcode{hep-cats/yymmnnn}
\titlepage
\title
Theory of Superconductivity
\endtitle
\author
J.~Bardeen
Department of Physics
University of Illinois
Urbana, Illinois 61801
\endauthor
\and
\author
L.~N.~Cooper
Department of Physics and Astronomy
The Ohio State University
Columbus, OH 43210
\endauthor
\and
\author
J.~R.~Schrieffer
Department of Theoretical Physics
University of Birmingham
Birmingham, England B15 2TT
\endauthor
\abstract
A theory of superconductivity is presented, base on
the fact that...

\endabstract
\toappear{Physical Review}
\endtitlepage

\section{Introduction}
The main facts which a theory of superconductivity
must explain are...
```

7.2. Headlines and Footlines

In Plain TeX whenever a page is output an optional headline and footline are added at the top and bottom of the page. In TeXsis the footline is normally blank, while the headline contains the page number in the upper right corner. The `\book` and `\thesis` formats also display a running headline centered in 10 pt. italic type containing the chapter or section title. Running headlines and page numbers are not printed on any page containing a `\title` or a new `\chapter`.

`\nopagenumbers` turns off the running headlines and footlines, just as it does in Plain TeX. `\pagenumbers` restores the standard headlines and footlines.

`\bookpagenumbers` puts odd page numbers in the upper right corner and even page numbers in the upper left corner, the normal convention for books. This is the default for `\book`.

`\RunningHeadlinestrue`, the default for `\book` (and `\thesis`), displays the running headline text `\HeadText`, as well as the page number. `\RunningHeadlinesfalse` turns off the running headlines but leaves the page numbers. `\HeadText` is defined by `\chapter` to be the chapter title in 10 pt. italic type; if no chapters are used, then it is defined by `\section` to be the section title. You can change the running headline yourself with the command

`\setHeadline{`⟨text⟩`}`

The headline text appears in typestyle `\HeadFont`, which you can change to suit your tastes. The default is

```
\def\HeadFont{\tenit}
```

7.3. Specialized Document Formats and Style files

The following specialized document formats are available with the standard distribution of TeXsis 2.15/h:

`\thesis`	Similar to `\book`, but for printing a thesis. The default `\thesis` format conforms to the requirements of the Yale Graduate School, but it can (and probably should) be modified for your own installation. [thesis.txs]
`\AIPproceedings`	Similar to `\paper`, but specialized to the American Institute of Physics (AIP) format. [AIP.txs]
`\IEEEproceedings`	IEEE format using oversized double column paper. See Section 11. [IEEE.txs]
`\IEEEreduced`	IEEE format reduced to 75% of full size and printed with 9 pt. type for $8\frac{1}{2}$ in $\times$ 11 in paper. [IEEE.txs]

`\NorthHolland`	A format for producing camera-ready copy for proceedings published by North-Holland/Elsevier. This is a single column format with titles and authors left justified and in eleven point type. [Elsevier.txs]
`\NorthHollandTwo`	The North-Holland double column format for oversized paper. [Elsevier.txs]
`\WorldScientific`	Similar to `\paper`, but specialized to the World Scientific proceedings format. [WorldSci.txs]
`\nuclproc`	The "NUCLPROC" style for Nuclear Physics Proceedings. [nuclproc.txs]

To invoke a particular format use it as you would `\paper` or `\preprint`. Rather than loading the instructions for all of these special layouts when only one will ever be used the definitions for different formats are kept in "style" files, one of which is loaded at run time when you call for it. TeXsis style files end with the extension `.txs`; the names of the files in which the formats are kept are included in the list above in brackets.

It is possible for you to create your own document styles to add to TeXsis's repertoire, either by modifying an existing style or by writing one completely from scratch. When a style file is called for it is loaded by TeX from one of the "usual" places. Thus if `WorldSci.txs` is to be loaded and you have such a file (modified or not) in your current directory that file will be loaded. If you don't have a file by this name in your own directory TeX will look in the appropriate system directory for the file. On a Unix machine style files are usually kept in `/usr/local/lib/tex/inputs`.

7.4. Autoloading

To make it easier for a particular style to be loaded without having to `\input` the file in which it's definition is kept a macro can can be declared to be an "autoload" instruction. For example, if the definitions for a style called `\TASI` are kept in the file `TASI.txs` then one would say:

```
\autoload\TASI{TASI.txs}
```

to tell TeXsis that when `\TASI` is called for the file `TASI.txs` should be loaded to get the real definition of `\TASI`, and that `\TASI` should be invoked once that file has been read.

If a style file is not set up to autoload then you will have to first read the file with `\input` and then invoke the style by name. Locally defined style files can be `\autoload`'ed in the `TXSsite.tex` file so that they are already know to TeXsis. Styles that might be of general interest should be forwarded to the authors of TeXsis so that we can include them in future distributions.

8. Environments

TEX ordinarily provides a layout appropriate for straight text and equations. The macros described below create environments that bypass the normal layout to format text in special ways, to create different types of lists, to display theorems and proofs, to display computer code, and even to print TEX examples. With the exception of some of the TEX example macros, the syntax for all of these is generally `\thing...` to begin the environment and `\endthing` to end it.

8.1. Flush or Centered Text

There are three environments in TEXsis which respectively center lines, set them flush left, and set them flush right:

`\center`	Centers all lines that follow, up to `\endcenter`.
`\flushleft`	The lines that follow, up to `\endflushleft`, are made to line up flush to the left margin.
`\flushright`	The lines that follow, up to `\endflushright`, are made to line up flush to the right margin.

In all of these environments line endings in the input file *are* obeyed, as with `\obeylines`. For example,

```
\center
A Centered Line
and Another
\endcenter
```

produces

A Centered Line
and Another

There is also a `\raggedcenter` environment which does not respect line breaks but instead produces centered lines each of which is made as long as possible:

`\raggedcenter`	Center lines without respecting line endings. Terminate with `\endraggedcenter`.

For example

```
\raggedcenter
In the ragged centering environment each line
is made as long as possible, and
the successive lines are centered.
\endraggedcenter
```

produces

In the ragged centering environment each line is made
as long as possible, and the successive lines are centered.

Note that `\raggedcenter` does not hyphenate words.

8.2. List Environments

When you want to make a list of things there are several ways to do it. The items in the list can be marked by a simple marker (like a bullet: •), by a number or by numbers and letters (as in an outline), or by some sort of descriptive text. In Plain TEX you would make all of these kinds of lists using the `\item` macro. The "list environments" described below are generalizations of `\item` for making varous kinds of lists:

`\itemize`	List of paragraphs marked by a bullet (•) or other symbol.
`\enumerate`	List of paragraphs labeled by numbers or by combinations of numbers and letters, as for an outline.
`\description`	List of items and paragraphs describing them.

In each list environment `\itm` is used to introduce each individual item. The environments are terminated with `\enditemize`, `\endenumerate` and `\enddescription`, respectively. The three different kinds of environments can be nested to produce a list with several levels. The indentation of and spacing above and below the list environments can be controlled. This is described in Section 8.2.4.

8.2.1. Itemized lists

`\itemize` produces paragraphs marked by bullets (•) or some other appropriate symbol; this is usually used for a short list with only a single level of nesting where the order either does not matter or is already contained in the items of the list. For example, `\itemize` was used to produce the list of sections of this manual which begins on page 1. The first few items in this list were produced by saying:

```
\itemize
\itm Sect.~\use{sect.start}:
Getting Started.  How to run and initialize \TeXsis,
and the "master file" concept.
\itm Sect.~\use{sect.fonts}:
Selecting sizes and fonts of type, choosing single,
double, or triple spacing, and automatically handling
quotation marks.
\itm Sect.~\use{sect.eqns}:
Automatic numbering of equations, automatic sizing of
```

```
parentheses and brackets, and special symbols for
physics equations....
```

A paragraph begun by `\itm` is not indented beyond the new left margin, but paragraphs that follow *are* indented unless `\noindent` is specified (or if they begin with `\itm`).

You can change the symbol used to mark each item in the list to be anything you want by creating your own definition of `\itemmark`. For example, if you want the symbol to be a `\square` you would define `\itemmark` to be

```
\def\itemmark{$\square$}
```

before invoking `\itemize`, or at least before the first `\itm`. An example is the following important checklist:[10]

- □ Event timer started (display: SINED)
- □ Antenna Aligned (display: SEELED)
- □ Oscillation overthruster armed (display: DELIVERED)

This was produced by saying

```
\itemize
\def\itemmark{$\square$}
\itm    Event timer started (display: SINED)
\itm    Antenna Aligned (display: SEELED)
\itm    Oscillation overthruster armed (display: DELIVERED)
\enditemize
```

`\itemize` starts a new group, which is ended by the `\enditemize`, so in this example `\itemmark` reverts back to it's previous definition when the list is finished. You can get a "check-mark" symbol instead of a square by replacing `\square` with `\surd`, or you can use any other symbol that suits you. Note though that in this example the `\square` is enclosed in math quotes — you will have to remember this for any symbol which is defined in math mode.

8.2.2. Enumerated lists

`\enumerate` produces numbered paragraphs, either with a single level or nested several levels deep, like so:

```
\enumerate
\itm Paragraph 1.
\enumerate
\itm sub-paragraph 1a.
\enumerate
\itm sub-sub-paragraph 1a(i).
\itm sub-sub-paragraph 1a(ii).
\endenumerate
\itm sub-paragraph 1b.
```

```
\itm and so on...
\endenumerate
\itm Paragraph 2.
\itm and so on...
\endenumerate
```

This produces the enumerated list:

1. Paragraph 1.
    1.1. sub-paragraph 1a.
        1.1.1. sub-sub-paragraph 1a(i).
        1.1.2. sub-sub-paragraph 1a(ii).
    1.2. sub-paragraph 1b.
    1.3. and so on...
2. Paragraph 2.
3. and so on...

The default form for the paragraph numbering is called `\enumPoints`, which produces numbers of the form $ii.jj.kk$. You can also have the list printed in a standard outline format with a mixture of Roman numerals, letters, and Arabic numerals by saying `\enumOutline`. Doing so to the example above produces:

I. Paragraph 1.
    A. sub-paragraph 1a.
        i. sub-sub-paragraph 1a(i).
        ii. sub-sub-paragraph 1a(ii).
    B. sub-paragraph 1b.
    C. and so on...
II. Paragraph 2.
III. and so on...

Unless enclosed within a group, the `\enumPoints` or `\enumOutline` style selected remains in effect until it is changed. The punctuation following the item number is `\enummark`, which by default is a period (.) but which can be redefined by the user [for example, you can change it to `\rparen` to obtain numbering of the form "2)"]. `\enummark` can also be used to change the spacing between the numbers and the text that follows.

A third enumeration style, called `\enumNumOutline`, is also available. Top level items are labeled by arabic numbers, the level below by lower case roman letters, and the level below that by lower case roman numerals. With `\enumNumOutline` the example above becomes:

1. Paragraph 1.
    a. sub-paragraph 1a.

i. sub-sub-paragraph 1a(i).

ii. sub-sub-paragraph 1a(ii).

b. sub-paragraph 1b.

c. and so on...

2. Paragraph 2.
3. and so on...

If you would like a different style of enumeration you can create your own `\enumWhatever` to use in place of `\enumPoints` or `\enumNumOutline`. To do so copy the definition of `\enumPoints` or one of its brothers from the file `TXSenvmt.tex`, along with the comments that go with them, and then make the appropriate changes as described in those comments.

8.2.3. Lists Labeled by Text

`\description` is used to create a list of words or other items labeled by some text. The syntax is

```
\description{⟨sample text⟩}
\itm{⟨label 1⟩} Paragraph 1...
\itm{⟨label 2⟩} Paragraph 2...
\itm{...}
\enddescription
```

The argument ⟨sample text⟩ is not printed, but its length is used to determine the indentation for the labels on the items, so it should be the longest item, followed by some extra space. For example, the descriptions of the flush and centered environments at the beginning of this section of the manual were typeset with

```
\description{~flushright~\qquad}
\itm{\CS{center}}
Center each line.
\itm{\CS{flushleft}}
Set each line flush left.
\itm{\CS{flushright}}
Set each line flush right.
\enddescription
```

since `\flushright` is the longest of the three items. (`\CS` is a special macro defined for this manual for printing control sequence names.) Each item in the list is separated by a `\parskip`, just as in itemized or enumerated lists.

If the label text for an item comes out wider than the space allowed, the text will usually be broken into several lines so that it will fit in the indented margin (if this is possible!). Saying `\singlelinetrue` will instead put the label text on a line by itself above the item. `\singlelinefalse` is the default, but this manual has been printed with `\singlelinetrue`.

8.2.4. Spacing of List Environments

Each of the list environments described above is surrounded by the following skips:

```
\EnvTopskip    = \medskipamount     % space above the list
\EnvBottomskip = \medskipamount     % space below the list
\EnvLeftskip   = 2\parindent        % left indent for the list
\EnvRightskip  = \parindent         % right margin moves in too
```

These can be changed to any legal TEX dimension. The space between items which are begun by `\itm` is taken from the `\parskip`. Since the list environment begins a group you can change the `\parskip` inside this group to control the spacing between `\itm`'s. For example, if you wanted no extra space between the items in the list you could say set the `\parskip` to zero, as in,

```
\itemize
\parskip=0pt    \def\itemmark{$\triangleright$}
\itm    Event timer started (display: SINED)
\itm    Antenna Aligned (display: SEELED)
\itm    Oscillation overthruster armed (display: DELIVERED)
\enditemize
```

which would produce:

▹ Event timer started (display: SINED)
▹ Antenna Aligned (display: SEELED)
▹ Oscillation overthruster armed (display: DELIVERED)

8.3. Simple Examples

The `\example` environment can be used to display an example of something in a way that sets it off from the main text. The indentation of an example is the same as in the list environments (see Sect.8.2.4). The text in an example will be single spaced, even if the rest of the document is double spaced. End this environment with `\endexample`. An example of the use of `\example` is

```
\example
Physics is not a finished logical system.  Rather, at any
moment it spans a great confusion of ideas, some that
survive like folk epics from the heroic periods of the past,
and other that arise like utopian novels from our dim
premonitions of a future grand synthesis.
\endexample
```

which produces[11]

> Physics is not a finished logical system. Rather, at any moment it spans a great confusion of ideas, some that survive like folk epics from the heroic periods of the past, and other that arise like utopian novels from our dim premonitions of a future grand synthesis.

8.4. Theorems and Proofs

Theorems, lemmas, and corollaries are typeset in a quasi-SIAM format, with the number in boldface and the text in slanted type. They are automatically numbered, including the chapter and section numbers if `\showchaptIDtrue` and `\showsectIDtrue` are being used. The syntax for a theorem is

```
\theorem{⟨label⟩}
⟨text⟩
\endtheorem
\proof
⟨text⟩
\endproof
```

and similarly for `\lemma`, `\corollary` and `\definition`. Definitions are printed in Roman type, not slanted. The `\endproof` macro inserts a black box to mark the end of the proof. Subsequent references to theorems, lemmas, corollaries and definitions can be made in the text automatically using `\Theorem{⟨label⟩}`, `\Lemma{⟨label⟩}`, `\Corollary{⟨label⟩}`, and `\Definition{⟨label⟩}`, which give the appropriate name and number. An example of a Theorem is:

```
\example
\theorem{Cauchy}
If $f(z)$ is a function of $z$ which is analytic on and inside of
a contour $C$, then
$$
\int_C f(z)\,dz = 0
$$
\endtheorem
\proof
Left as an exercise to the reader.
\endproof
\endexample
```

which produces

> **Theorem 8.1.** *If $f(z)$ is a function of z which is analytic on and inside of a contour C, then*
>
> $$\int_C f(z)\,dz = 0$$
>
> *Proof.* Left as an exercise to the reader. ∎

Note the use of `\example` to indent the theorem and proof and print them single spaced. Had the `\example` been omitted the theorem and proof would have had the same margins and spacing as the rest of the text. This theorem would be referred to in the text with `\Theorem{Cauchy}`, which gives "Theorem 8.1." This should be sufficient in most cases, but if necessary one can refer to theorem or lemma numbers with the primitive `\use{⟨type⟩.⟨label⟩}`, where ⟨type⟩ is either `Thm`, `Lem`, respectively. See Section 12 for more on the use of `\use`.

8.5. Listing Computer Code

The `\Listing` environment can be used to display computer code or fragments of computer code. The text is printed in `\tt` type with line endings respected and every space counted. You can still use TeX control sequences in this environment, but many of the special characters like `%`, `&`, and `#` are simply printable characters, since they are used in languages like C. (`$`, `_` and `^` also loose any special meaning in this environment.) Since the curly brackets used by TeX are also used in C (and other languages) they are also simply characters in the `\Listing` environment. To perform grouping you should use `\bgroup` and `\egroup`. The backslash is also used in C as an escape character, so there has to be some way to print it — you simply use `\\`; thus `\n` should be typed as `\\n`. You end the listing environment with `\endListing`. As an example here is a very famous little program,[12]

```
main()
{
   printf("hello, world\n");
}
```

which was created by typing,

```
\Listing
main()
{
   printf("hello, world\\n");
}
\endListing
```

The indentation and spacing of the of the `\Listing` environment are controlled by the same parameters that control the indentation of the list environments, as described in Sect. 8.2.4.

An alternative way to list computer code is `\ListCodeFile{`⟨filename⟩`}`, which will list the contents of a source code file exactly as it appears in the file (with no need to make `\n` into `\\n` and such). The call to this macro should *not* be enclosed in a `\Listing` environment. If the example program above were contained in the source file `hello.c` it could be listed simply by typing

```
\ListCodeFile{hello.c}
```

With `\ListCodeFile` the right margin of the page is extended outward to make room for long lines. If you find that you have problems getting the listing to fit on the page you may want to make the type size smaller before you invoke `\ListCodeFile`. Note also that tab characters in the code file may not be treated exactly correctly. In the present implementation they will simply be replaced with eight spaces. We hope to be able to improve how this macro deals with tabs in in a future release of TeXsis, but for now it may be best to change tabs to spaces in the source file if proper tab stops are needed.

8.6. TeX example macros

Typing TeX examples can't be done in either the `\example` environment nor the `\Listing` environment, because the backslash must be turned into a simple printing character. There are several ways TeXsis allows you to show examples of TeX code:

`\TeXexample`	Prints an example verbatim in `\tt` type, with "`|`" made the escape character. Note: you must say `|endTeXexample` to end the example, since "`\`" is no longer the escape character!
`\TeXquoteon`	Make "`|`" a "TeX quote" so that everything between `|`'s is printed verbatim in `\tt` type.
`\TeXquoteoff`	Turns off "TeX quotes."

All of the displayed examples in this document have been typed using `\TeXexample`, and many of the inline examples have been typed by putting `\TeXquoteon` at the beginning of the document and using `|\whatever ... |`. For both `\TeXexample` and `\TeXquoteon` the only special character is the escape character "`|`". In particular one must type

```
\TeXexample
⟨TeX stuff⟩
|endTeXexample
```

with `|endTeXexample` instead of `\endTeXexample`. All other characters are set in `\tt` type without any interpretation. The input line breaks and spacing are respected, including any leading spaces on the lines. TeX commands can be used within a `\TeXexample` environment by typing `|macro` in place of `\macro`, `|bgroup` and `|egroup` in place of `{` and `}`, etc. In particular, a `|` can be obtained by typing `|char124`. If you want to display the results of a complicated macro inside the `\TeXexample` environment you should probably define a different macro to display these results and then invoke that macro inside the environment. For example, in this manual we have sometimes inserted "meta" constructions showing where to insert a particular piece of text by using a `\meta` macro which encloses it's argument in angle brackets (see the example immediately above). But in a `\TeXexample` you cannot say "`|meta{TeX stuff}`," because the "`{`" and "`}`" are no longer grouping characters. Instead you could define the temporary macro `\tmp` as

```
\def\tmp{\meta{TeX stuff}}
```

and then inside the `\TeXexample` environment simply type `|tmp` to insert the phrase "⟨TeX stuff⟩".

The indentation and spacing of the `\TeXexample` environment are controlled by the same parameters that control the indentation of the list environments, as described in Sect. 8.2.4. The general method used by all of these macros to turn off TeX's usual mechanism for interpreting special characters is discussed in *The TeXbook*, p. 421ff. The `\begintt` and `\beginlines` macros described there are included in TeXsis for completeness.

9. Figures and Tables

Technical papers often include tables and/or figures. Regardless of how the contents of the figures or tables are created, these share a lot in common in the way they are generally treated. TEXsis makes use of TEX's floating insertion capabilities (see Chapter 15 of *The TEXbook*) to insert figures and tables on the current page or on a following page. It is also possible to have figures or tables saved and printed at the end of the document. Captions are easily added either above or below the figure or table. Both kinds of objects are automatically assigned numbers, which are associated with internal labels so that they may be referred to in the text without having to know their order. It is possible to create a list of the figures and tables which appear in a document, much like creating a table of contents. Figures are generally created outside of TEX, either by hand or by computer graphics programs, while tables can be created using TEX commands. The most complicated tables have vertical and horizontal lines (called "rules") to guide the reader's eye and to make the table more readable. TEXsis has some very easy to use commands for making ruled tables.

9.1. Figures

TEX is designed for technical typesetting, but does not include a graphics standard, so to include a figure or drawing it is necessary to leave blank space and to either glue the drawing in later or to arrange for the figure material to be inserted by some mechanism outside of TEX when the page is actually printed (e.g., using the `\special` command or the `\epsfbox` command described later).

Imagine that you wish to insert into the text a drawing of a "penguin diagram," which is about 6 cm tall. You could create the space for this figure at the top of a page, along with a caption, with the following commands:

```
\figure{Penguin}
\hrule
\vskip 3cm
\line{\hss (glue your penguin here) \hfil}
\vskip 3cm
\hrule
\Caption
Sample figure showing how easy it is to insert
numbered figures with captions in \TeXsis.
\endCaption
\endfigure
```

This produces Fig. 9.1 at the top of the next page (we put the `\hrule`'s in just to show off the empty space). "Penguin" is the label for the figure, so to refer to the figure in the text we typed `\Fig{Penguin}`, which produced "Fig. 9.1." The caption can be placed either before or after the figure: if it is to be at the top then `\Caption` must be the first token after `\figure`. As shown, the caption text must be terminated with `\endCaption`. When the caption is printed it will automatically start with the figure number in bold

(glue your penguin here)

Figure 9.1: Sample figure showing how easy it is to insert numbered figures with captions in TEXsis.

type, followed by the caption text. The caption is indented on both sides if it is more than one line long and centered otherwise.

`\caption{`⟨text⟩`}` is equivalent to `\Caption` ⟨text⟩ `\endCaption` and may be more convenient for short captions.

There are figure insertion commands for each of the insertion classes of Plain TEX, plus special commands for wide figures in double column formats (which are described later in Section 11):

`\figure{`⟨label⟩`}` Inserts a figure at the top of the next available page.

`\midfigure{`⟨label⟩`}` Inserts a figure at the current position if there is room or at the top of a page if necessary.

`\fullfigure{`⟨label⟩`}`
: Inserts a figure on a separate page. The body of the figure should normally contain a `\vfill`.

`\widetopfigure{`⟨label⟩`}`
: Inserts a figure at the top of a page across both columns in double column format. Equivalent to `\figure` in single column format. `\widefigure` is a synonym.

`\widefullfigure{`⟨label⟩`}`
: Inserts a full page figure across both columns in double column format. Equivalent to `\fullfigure` in single column format.

`\endfigure` Ends any of the figure insertions.

In all cases ⟨label⟩ is an internal label which is assigned to the figure number. The figure number will include the chapter and section numbers in the format "*cc.ss.nn*" if `\showchaptIDtrue` and `\showsectIDtrue` have been selected.

The body of the figure can include any vertical mode commands, as in the example above. When you are simply leaving space for a figure to be glued in later a convenient way to specify the amount of space to leave is to `\vskip` by some amount proportional to the width of the current column or page. The dimension `\colwidth` is the width of the page (the `\hsize`) in single column mode and the width of the current column in double column mode. So if, for example, you have a figure which was half again as tall as it is wide you could say

```
\vskip 1.5\colwidth
```

Another way to standardize the space reserved for figures is to define your own dimension, such as `\myfigsize`, to use in the `\vskip` above. For example,

```
\newdim\myfigsize
\myfigsize= 1.5\colwidth
```

and then `\vskip\myfigsize`. This has the same effect as the previous example, but `\myfigsize` can be changed to any valid dimension.

9.2. Encapsulated PostScript Graphics

Instead of simply gluing figures into a paper by hand it is possible to have computer generated graphs and drawings automatically inserted into a document. Unfortunately, the method can vary radically from computer to computer and printer to printer. Recall that TEX produces as output a DVI file, which contains device independent instructions for typesetting the document. To actually print the document the DVI file must be translated into instructions for the specific printer being used by a suitable device driver. It is at this stage, when the device driver is converting the DVI file to printer instructions, that graphs and drawings may be inserted. However, since different printers may require different kinds of instructions, there are many different device drivers, each having its own way of including drawings in the document.

In recent years some standardization has become possible, because many printer manufacturers have chosen PostScript as the "language" for giving instructions to their printers. PostScript contains commands both for drawing and for typesetting text. Still, there are some problems, because there are many different device drivers for converting DVI files to PostScript, and they all treat the inclusion of figures slightly differently. Also, while all of these filters require that the drawings to be included are themselves already in PostScript, there are several variations of PostScript, leading to further problems of incompatibility.

Until there is some standardization for inserting drawings into TEX documents (and perhaps to encourage it) we recommend using the scheme which (a) seems to be the easiest to use and the most well thought out, and (b) is potentially available to the widest group of users. Our choice is, first, to use a specific variation of PostScript known as

"Encapsulated" PostScript (EPS). Most newer computer programs which produce graphics output, including Mathematica, `xfig`, and `paw`, can produce EPS output files. To include these in the text we have chosen to rely on the program `dvips` by Tomas Rokicki,[13] along with his TEX macros for including EPS graphics, called `epsf.tex`. Be warned that there are other programs called "dvips," but as far as we know only Rokicki's version will work with `epsf.tex`. We expect that when including drawings in TEX is standardized, it will be based on Rokicki's scheme or something close to it.

To include EPS graphics in a document using the `epsf.tex` macros you must first input the macros. Somewhere early in your manuscript file you should put:

```
\input epsf.tex
```

You can have the macros report the size of a figure when it is included by saying:

```
\epsfverbosetrue
```

At the point where you want a graph or drawing included you simply say:

```
\epsfbox{foo.ps}
```

This will include the drawing from the file called '`foo.ps`,' which must already exist in the current directory. TEX will leave enough room for the figure (by creating a box of the correct size) and will use a `\special` to insert instructions into the DVI file telling the DVI to PS translator to read in the EPS file `foo.ps`.

Once the document has been processed by TEXsis and the DVI file has been produced you will convert it to a PS file with the "dvips" program. If your DVI file is called `myfile.dvi` then the command is simply:

```
dvips myfile
```

This will convert `myfile.dvi` to PostScript, and depending on how dvips is set up at your installation it will either put the output in the file `myfile.ps` or it will send it directly to the default printer (see the man pages for dvips on how to change this behavior, or try a short example to see how it is set up on your computer). Note that the EPS files to be included must be in the same directory as `myfile.dvi` when you run dvips, so that they can be merged into the final output file.

Note that on Unix systems you cannot use the `lpr -d` command if you have included EPS files in the document. This command runs the DVI file through dvips (or some other DVI to PS filter) in a different directory, and the EPS files will therefore not be included.

Also note that if you preview the DVI file on a workstation with `texsun` or a similar previewer, you will not see your drawings in the previewed document, because the previewer does not know how to deal with the `\special` commands in the DVI file. You will, however, see that TEX has left room for the figure. Once you have converted your DVI and EPS files into a PS file you can preview that file if you have a PostScript previewer, such as GhostScript. Then your graphs and drawings will appear just as they would if you had printed the PS file.

Sometimes you will find that more room has been left for the figure than is really needed. The way `\epsfbox` decides how much room to leave is by reading the first part of the EPS file and looking for a line like

```
%%BoundingBox: 67 176 544 616
```

The two pairs of numbers are the coordinates of the lower left and upper right corners of the figure, respectively. These are measured in printer's points (1/72nd of an inch). If you don't like the positioning of the figure given by the original bounding box in the PostScript file you can override it by specifying a new bounding box like so:

```
\epsffile[0 180 540 700]{file.ps}
```

You may have to experiment with positioning the BoundingBox for the best effect, and it may help to look at the PostScript file to see what the default BoundingBox is and adjust things using that "coordinate system."

You can also control the size of the included graphic using `\epsfxsize` or `\epsfysize` before you call `\epsfbox`. The figure will be enlarged or reduced to fit the dimension you specify. Saying

```
\epsfxsize=⟨dimen⟩ \epsfbox{figure.ps}
```

will scale the drawing in the file `figure.ps` so that its *width* is ⟨dimen⟩, with the height scaled accordingly, while saying

```
\epsfysize=⟨dimen⟩ \epsfbox{graph.ps}
```

will scale the drawing in the file `graph.ps` so that its *height* is ⟨dimen⟩, with the width scaled accordingly. Whatever change of size you specify will only be applied to the `\epsfbox` command which immediately follows. After that, subsequent included graphics will revert to their natural sizes unless you also specify a particular size for them.

So now consider the earlier example of including a 6 cm tall penguin diagram in a figure, but now assuming that the drawing exists as an EPS graphic in the file `penguin.ps`, and that we need to override the default bounding box in that file. The commands to do this are:

```
\figure{Penguin}
\epsfysize=6cm \epsffile[0 180 540 700]{penguin.ps}
\Caption
Example of including EPS graphics in a figure.
\endCaption
\endfigure
```

9.3. Simple Tables

Simple tables (without rules) are constructed in TEX with alignment tabs or with `\halign`. The reader who is not familiar with these should look at Chapter 22 in *The TEXbook*. In TEXsis simple tables with table numbers and captions are treated much the same as figures. For example, saying

```
\midtable{table example}
\caption{Charged leptons.}
\centerline{\vbox{
\halign{#\ \hfil & #\ \hfil & #\hfil \cr
Name     & Mass     & Lifetime \cr
$e^-$    & .511 MeV & $>2 \times 10^{22}$ yr. \cr
$\mu^-$  & 105 MeV  & $2.197 \times 10^{-6}$ sec. \cr
$\tau^-$ & 1784 MeV & $3.3 \times 10^{-13}$ sec. \cr}}}
\endtable
```

produces the following table:

Table 9.1: Charged leptons.

Name	Mass	Lifetime
e^-	.511 MeV	$> 2 \times 10^{22}$ yr.
μ^-	105 MeV	2.197×10^{-6} sec.
τ^-	1784 MeV	3.3×10^{-13} sec.

Note the use of `\centerline` and a `\vbox` to center the table. The reference to this table in the text was produced by typing "`\Tbl{table example}`." Note also that we used `\caption` followed by the text in brackets as an alternative method of defining the caption; `\caption` should only be used when the caption is short. Table captions are traditionally placed above the table, so `\Caption` or `\caption` should be the first token after `\table{`⟨label⟩`}` and its brothers.

There are a number of different kinds of table insertions, and they match closely the different kinds of insertions used for figures:

`\table{`⟨label⟩`}` Inserts a figure at the top of next available page.

`\midtable{`⟨label⟩`}` Inserts a figure at the current position if there is room or at the top of a page if necessary.

`\fulltable{`⟨label⟩`}` Inserts a figure on a separate page. The body of the figure should normally contain a `\vfill`.

`\widetoptable{`⟨label⟩`}`
Inserts a figure at the top of a page across both columns in double column format. Equivalent to `\table` in single column format.

`\widefulltable{`⟨label⟩`}`
: Inserts a full page figure in double column format. Equivalent to `\fulltable` in single column format.

`\endtable` Ends any of the tables above.

Again, in these macros ⟨label⟩ is an internal label which is assigned to the table number. The table number will include the chapter and section numbers in the format "*cc.ss.nn*" if `\showchaptIDtrue` and `\showsectIDtrue` have been selected. The table can then be referred to in the text by saying `\Tbl{`⟨label⟩`}`.

9.4. Ruled Tables

While `\halign` in Plain TEX is convenient for making simple tables, it is still difficult to make tables with horizontal and vertical rules. Indeed, typesetting ruled tables in TEX has often been likened to programming a computer in assembler language. Even *The TEXbook* acknowledges that "people who know how to make ruled tables are generally known as TEX Masters."[14] The macros described below take a great deal of the difficulty out of making ruled tables, so that anybody can become a TEX Master, or at least a master of ruled tables.

The basic idea behind making a ruled table is similar to `\halign`, and the syntax is the same as the Princeton table macros known as `TABLES.TEX` by Ray Cowan,[15] although the TEXsis macros are quite different internally and run about 5 times faster. You begin the table material with `\ruledtable` and end it with `\endruledtable`. The table material is a list of items to appear in the rows and columns of the tables. Just as with `\halign` you can separate items in different columns with `&` and end a row with `\cr`. The `&` alignment character separates columns without putting a vertical rule between the columns, while `\cr` ends a line and puts a thin horizontal rule under it. To get a vertical rule between columns replace the `&` with a '`|`' (the vertical line), and to end a line without a horizontal rule under it replace the `\cr` with `\crnorule`. One important thing to remember is to end the last line in the table with `\endruledtable` instead of `\cr`!

As a first example, here is a simple ruled table,

Year	World Population
8000 B.C.	5000000
50 B.C.	200000000
1650 A.D.	500000000
1850 A.D.	1000000000
1945 A.D.	2300000000
1980 A.D.	4400000000

and here are the instructions that created it,

```
\ruledtable
Year       |    World Population \cr
8000 B.C. |   ~~~5000000        \crnorule
~~50 B.C. |   ~200000000        \crnorule
1650 A.D. |   ~500000000        \crnorule
1850 A.D. |   1000000000        \crnorule
1945 A.D. |   2300000000        \crnorule
1980 A.D. |   4400000000
\endruledtable
```

This example also appears in *The TEXbook*, where it is typeset using `\halign`. Compare how the two are constructed and see which way you think is easier.

By default each item in the table is centered in its column. Inside a table the tie character "`~`" takes up the space of exactly one digit (it is the same as saying `\phantom{0}`), and this has been used to line up the digits of the entries in both columns. It is also possible to change the definition of the `\TableItem` macro so that each entry in each column is left or right justified. This is described a little bit later.

Something to note about this example table is that the outer rules are thicker than the others. You can also get a thick vertical rule between columns by using "`\|`" in place of "`|`", and you can get a thick horizontal rule below a row by replacing the `\cr` with `\crthick`. The widths of thick and thin rules are controlled by the dimensions `\thicksize` and `\thinsize`. If you don't want thick rules at all simply say `\thicksize=\thinsize`, which makes the thick rules as thin as the thin rules.

Some synonyms: `\CR` is the same as `\crthick`, `\nr` is the same as `\crnorule`, and `\crrule` is the same as `\cr` (actually it's the other way around!). `\tab` is the same as `&` — it separates columns without a vertical rule. `\vb` is the same as `|`, it separates columns with a thin vertical bar, while `\Vb` is the same as `\|` and separates the columns with a thick vertical bar. You can also use `\dbl` to separate columns with a double vertical rule.

As in Plain TEX you can use `\span` in place of `&` (or the other column separators) to join two columns together into one, and you can use `\omit` as the very first token in a column to tell it to omit the fancy column layout (the centering or other spacing). You can also use `\multispan` to `\span` and `\omit` several columns, just as in an `\halign`. The only difference is that the number of columns you tell it to `\multispan` is the number of columns of the table it spans — in a ruled table constructed with `\halign` you would count the columns *and* any vertical rules.

Table 9.2 is another sample table, again taken from *The TEXbook*. Something important to notice is that the last ruled table example didn't have a table number or a caption, while this one does. That's because the last example was created using just `\ruledtable`, while this example is inside of `\table`. The point is that `\table` and `\ruledtable` do separate things and can be used either separately or together; `\table` creates the space for the table and gives it a caption and a number, while `\ruledtable` actually constructs the body of the ruled table. The instructions that created Table 9.2 are:

```
\table{example-II}
\caption{Another example of a ruled table.}
```

Table 9.2: Another example of a ruled table.

AT&T Common Stock		
Year	Price	Dividend
1971	41-54	$2.60
2	41-54	$2.70
3	46-55	$2.87
4	40-53	$3.24
5	45-52	$3.40
6	51-59	.95

```
\singlespaced
\ruledtable
\multispan3\hfil AT\&T Common Stock \hfil\CR
Year \dbl Price | Dividend \cr
1971 \dbl 41-54 | \$2.60   \cr
~~~2 \dbl 41-54 | \$2.70   \cr
~~~3 \dbl 46-55 | \$2.87   \cr
~~~4 \dbl 40-53 | \$3.24   \cr
~~~5 \dbl 45-52 | \$3.40   \cr
~~~6 \dbl 51-59 |  ~~.95
\endruledtable
\endtable
```

Sometimes it is desirable to have rules across only some of the columns of a table. In this case put `\crule` in the columns that are to have rules, `\cskip` in the columns where no rules are desired, and end the line with `\crpart`, which signals that the line contains a partial rule. The column headings in Table 9.3 are an example of partial rules in a table. That table was created with the commands,[16]

```
\table{SiHCl}
\caption{...}
\singlespaced\tenpoint

\ruledtable
\vctr{Molecule} | \vctr{Region} | Radius |
  \vctr{$\alpha _{\rm HF}$}|\multispan{2} $l_{\rm max}$
\crnorule
\cskip | \cskip | \cskip |\cskip|    \crule     |  \crule \crpart
       |        | (au)   |      | initial state | final state \CR

SiCl$_4$      | outersphere | 6.21 | 0.7238 | 4 | 7 \crnorule
              | Si          | 2.20 | 0.7275 | 2 | 3 \crnorule
              | Cl          | 2.40 | 0.7233 | 2 | 3 ...
```

Table 9.3: Parameters used in the MS-Xα calculations for the chlorosilane molecules, H_xSiCl_{4-x}; x=0–4.

Molecule	Region	Radius (au)	α_{HF}	l_{max} initial state	l_{max} final state
$SiCl_4$	outersphere	6.21	0.7238	4	7
	Si	2.20	0.7275	2	3
	Cl	2.40	0.7233	2	3
$SiHCl_3$	outersphere	6.21	0.7260	4	7
	Si	2.16	0.7275	2	3
	Cl	2.40	0.7233	2	3
	H	1.20	0.7772	1	2
SiH_2Cl_2	outersphere	6.25	0.7295	4	7
	Si	2.20	0.7275	2	3
	Cl	2.40	0.7233	2	3
	H	1.20	0.7772	1	2
SiH_3Cl	outersphere	6.30	0.7360	4	7
	Si	2.20	0.7275	2	3
	Cl	2.40	0.7233	2	3
SiH_4	outersphere	4.10	0.7524	4	7
	Si	2.04	0.7275	2	3
	H	1.20	0.7772	1	2

```
\endruledtable
\endtable
```

Ruled tables are normally centered on the page, but you can turn off the centering with `\noncenteredtables`. Then the table becomes just another piece to add to a horizontal list. One thing this lets you do is have two tables side by side on the page, like so:

Item ABC	Item DEF
Item GHI	Item JKL

Data 111	Data 222
Data 333	Data 444
Data 555	Data 666

This pair of tables was produced by saying:

```
\noncenteredtables
\medskip
\line{
\ruledtable
Item ABC | Item DEF \cr
Item GHI | Item JKL \endruledtable
\hfil
\ruledtable
```

```
Data 111 | Data 222 \cr
Data 333 | Data 444 \cr
Data 555 | Data 666 \endruledtable}
\smallskip
```

Saying `\centeredtables` turns table centering back on. Tables are then centered across the page, and a table is then added to a *vertical* list.

Usually `\ruledtable` makes tables as wide as their natural width, but it is possible to make them wider by setting the dimension `\tablewidth` to the width desired before saying `\ruledtable`. We can make the same table in Table 9.2 but with the width set to 10cm:

AT&T Common Stock		
Year	Price	Dividend
1971	41-54	$2.60
2	41-54	$2.70
3	46-55	$2.87
4	40-53	$3.24
5	45-52	$3.40
6	51-59	.95*

* (first quarter only)

The careful reader will notice that the version of Table 9.2 in *The TEXbook* had a footnote to the last entry, and we have added that here. To do so we had to put the footnote in an `\hbox` and stack it under the table, so we had to turn off table centering. The table is therefore a bit more complicated to make, but not much:

```
\noncenteredtables
\tablewidth=10cm
\line{\hfil\vbox{\singlespaced
\ruledtable
\multispan3\hfill AT\&T Common Stock \hfill\CR
Year \dbl Price | Dividend \cr
1971 \dbl 41-54 | \$2.60   \cr
~~~2 \dbl 41-54 | \$2.70   \cr
~~~3 \dbl 46-55 | \$2.87   \cr
~~~4 \dbl 40-53 | \$3.24   \cr
~~~5 \dbl 45-52 | \$3.40   \cr
~~~6 \dbl 51-59 |  ~~.95\rlap*
\endruledtable
\hbox{* (first quarter only)}
\vss}\hfil}
```

If we had not wanted to add the footnote, then the table would have been typed the same way as Table 9.2 except for setting the `\tablewidth`.

Important note: `\tablewidth` is turned "off" after the table is made, so the next table will have its natural width unless you explicitly set `\tablewidth` again!

`\ruledtable` normally inserts a `\quad` of space around each item. If you have a very wide table, especially one with many columns, you may wish to reduce this. `\TightTables` make this spacing a single space, the minimum acceptable. `\LooseTables` restores the default spacing. You can define this space to be something else by re-defining `\tablespace`, for example as

```
\def\tablespace{\ \ }
```

If the table is still too wide with `\TightTables`, you will probably have to reformat it or print it in landscape mode.

The careful reader may also have noticed that in the examples in Table 9.2 and Table 9.3 we said `\singlespaced`. That's because the spacing of the rows of a table can be controlled just like the spacing of the text with `\singlespaced`, `\doublespaced`, etc. The space between the rows of a table is set by a strut called `\tstrut`, which holds the rows apart. Its default definition is

```
\def\tstrut{\vrule height 3.1ex depth 1.2ex width 0pt}
```

You can change the spacing between rows by changing the definition of this strut. One of the things `\singlespaced` does is define the `\tstrut` to be 0.5ex higher and deeper than a parenthesis. `\doublespaced`, `\triplespaced`, and `\widenspacing` all increase the height and depth of the strut in proportion to the increase in the baseline spacing.

If you put something in a table which is larger than the strut, then it will touch the rules above or below. You can fix this by inserting your own strut into that row. More simply, you can say `\bigitem{`⟨stuff⟩`}`, and "stuff" will be positioned correctly with a nice amount of space above and below.

You can also decide whether or not you want diagnostic messages printed when a table is created. Saying `\tableinfotrue` causes a message to be printed on your terminal every time a ruled table is created. The message tells you how many rows and columns were in the table. This is useful information, because it can tell you immediately if you made a mistake in the table or left out a row or column. `\tableinfotrue` is the default, but you can turn off these messages by saying `\tableinfofalse`. In any case, you will be warned if a row has a different number of columns from the previous one.

The tables we have demonstrated so far all contain numbers, but sometimes it is desirable to have an entry in a table which is a paragraph of text. For that purpose, you can use `\para{`⟨text⟩`}`. The width of the paragraph is determined by the dimension `\parasize`, which defaults to 4 inches.

Tables 9.4a and 9.4b on the next two pages summarize the most important commands for making ruled tables.

Table 9.4a: Basic table macros for use with `\ruledtable`.

<table>
<tr><th>Macro name</th><th>Description</th></tr>
<tr><td><code>\ruledtable</code></td><td>Starts a ruled table.</td></tr>
<tr><td><code>|</code> (vertical bar)
or <code>\vb</code></td><td>Separates one column from the next, with a a thin vertical rule between them.</td></tr>
<tr><td><code>&</code> or <code>\tab</code></td><td>Separates one column from the next, but with no rule between them.</td></tr>
<tr><td><code>\cr</code></td><td>Ends the current row, and starts the next one. The completed row will be separated from the next with a thin horizontal rule.</td></tr>
<tr><td><code>\crthick</code>
or <code>\CR</code></td><td>Similar to <code>\cr</code>, but the rows will be separated with a thick horizontal rule.</td></tr>
<tr><td><code>\crnorule</code>
or <code>\nr</code></td><td>Similar to <code>\cr</code>, but the rows will not be separated by a rule.</td></tr>
<tr><td><code>\|</code></td><td>Same as <code>|</code>, but with a thick vertical rule.</td></tr>
<tr><td><code>\dbl</code></td><td>Same as <code>|</code>, but with two thin vertical rules.</td></tr>
<tr><td><code>\endruledtable</code></td><td>Ends the ruled table.</td></tr>
</table>

Continued...

Finally we can discuss how you can change how each column of a table is constructed. The macros which build the table take each entry and pass it to a macro called `\TableItem`. The default definition of `\TableItem` is:

```
\let\TableItem=\@TableItem       % default is centered
\def\@TableItem#1{%              % centers item in ruled table
   \hfil\tablespace              % left glue
   #1\killspace                  % item, no space after
   \tablespace\hfil              % right glue
    }%
```

The `\hfil` glue on both sides of the argument causes the item to be centered in the column, while `\tablespace` (which defaults to `\quad`) insures that there will be at least some amount of space between the item and the vertical rules on each side. Because of the way that TEX handles macro arguments, if you type a table with spaces between the text and the next column separator, an extra `\space` will be inserted into the table. If you first say `\NoTrailingSpaces`, then `\killspace` in the above macro is defined to remove any trailing glue items. `\hfill` is redefined by the table macros so that if it is the last token in an entry, then that entry is properly left justified, but to insert any other trailing glue, you must say

⟨text⟩⟨glue⟩`\null`

Table 9.4b: Additional macros for use with `\ruledtable`.

Continued...

Macro name	*Description*
`\multispan{`n`}`	Makes the next entry span the next n columns, where n is an integer, $n > 0$. See other notes on `\multispan` below.
`\omit`	This TeX primitive causes the normal template for its entry to be omitted, allowing the user to do something else with this entry.
`\para{`⟨text⟩`}`	Formats *text* into a neat little paragraph like this one. The width of the paragraph produced is determined by the dimension `\parasize`. The default is 4 inches.
`\parasize=`⟨dimen⟩	Sets the width of paragraphs for the `\para` macro.
`\TableItem{`⟨stuff⟩`}`	Used in the standard template, this macro centers its argument in the column. The user can redefine it for special effects.
`\tablespace`	The spacing around each item for `\TableItem`.
`\LooseTables` `\TightTables`	Set `\tablespace` to `\quad` and `\ ` respectively.
`\TrailingSpaces` `\NoTrailingSpaces`	Use trailing spaces and glue items in table entries (default) or ignore them. See text.
`\bigitem{`⟨stuff⟩`}`	Use this for something that would normally not fit in one line of a table to make the spacing between rows large enough for this item.
`\vctr{`⟨stuff⟩`}`	Centers the item *vertically* so that it can appear between two rows.
`\centeredtables`	Turns table centering on. Each table will be centered left-to-right on the page. This is the default.
`\noncenteredtables`	Turns table centering off. Each table is returned as an `\hbox`, and it is up to the user to place it as desired.
`\tableinfotrue` `\tableinfofalse`	Turns on or off the diagnostic message telling you how many rows and columns were found in the table.
`\thicksize=`⟨dimen⟩	This dimension specifies the thickness of the thick rules in the table. The default size is 1.5 points.
`\thinsize=`⟨dimen⟩	This dimension specifies the thickness of thin rules in the table. The default size is 0.8 points.
`\tablewidth=`⟨dimen⟩	Specifies how wide to make the next table. If not specified, the table is made to its natural width. This value is reset following the construction of each table.

The default is `\TrailingSpaces`, which does incorporate any trailing spaces into the table but which also treats any trailing glue items normally. You may wish to change this default in your `TXSpatch.tex` file.

You can change the definition of `\TableItem` to suit your own purpose. The simplest modification is to omit the `\hfil` on the left or right so that each entry is right or left justified. This is so commonly used that saying `\RightJustifyTables` or `\LeftJustifyTables` makes the appropriate change for you. Saying `\NoJustifyTables` restores `\TableItem` to its original centering function.

Since `\TableItem` is used to create every column of the table, you cannot use it to modify a single column. To justify just one column you can simply put `\hfill` to the left or right of each item in the column, since `\hfill` is infinitely more stretchy than `\hfil`. (The behavior of glue items other than `\hfill` depends on whether `\NoTrailingSpaces` has been set.) With a little more work it is possible to design special or unusual tables by redefining the "preamble" of the table. This is possible because `\ruledtable` uses `\halign` with a preamble called `\TablePreamble`. The default definition of `\TablePreamble` is:

```
\def\TablePreamble{%                      % \ruledtable preamble
   \TableItem{####}%                      % the first item
   \plaintab\plaintab                     % && means repeat this
   \TableItem{####}%                      % the subsequent items
   \plaincr}%                             % end of preamble
```

The syntax for `\TablePreamble` is like an `\halign` preamble in Plain TeX but with `&` replaced by `\plaintab`, `\cr` replaced by `\plaincr`, and `#` replaced by `####`. If you have read this far and want to try changing `\TablePreable` then you should read the source code and the comments that go with it in the file `TXSruled.tex`.

As a final note we point out that it is also possible for you to create your own column separators with the macro `\nextcolumn`. This macro takes a single argument which is used to separate one column of the table from another. Thus for example, the `\dbl` macro for creating a double vertical rule between columns is defined as:

```
\def\dbl{%
   \nextcolumn{\vrule width\thinsize
   \hskip\thinsize\vrule width\thinsize}}
```

9.5. Lists of Figures and Tables

In large technical documents it is often desirable to have a list of the figures and/or tables which appear in the document. This is similar to a table of contents, but is separate from it. In TEXsis you can add a figure or table to a list of figures or tables by using `\infiglist` after `\figure` or `\intablelist` after `\table`, similar to the way you would use `\caption`. The syntax is `\infiglist{`⟨text⟩`}`. The text is the text of the (possibly abbreviated) caption which is to appear in the list. Remember though, if a table has a caption it should appear above the table body, so the `\caption` must still be the first thing after `\table`; the `\infiglist` can come later.

To produce the list of figures at the end of the document you say `\ListFigures`, and similarly to produce the list of tables say `\ListTables`. Even if the list is to appear at the front of the document you should print it at the end and move the list to the front afterwards. This is because until the document is processed TEX does not know on which page the figure or table appears.

One word of caution: figures and tables are produced by putting them in "floating inserts", which TEX saves up and prints when there is room. The entry into the list of figures or tables is not made until the item actually makes its way into the output DVI file. If you ask for the list of figures or tables before one of these items is output it will not be in the list, and in fact that entry for the figure or table list will later end up spewing out on your terminal. To be sure that all floating insertions have been output you should say `\supereject` before you print the list of figures or tables.

9.6. Printing Figures and Tables at the End of the Document

When working on a paper to be sent for publication it is sometimes more useful to print all of the figures at the end of the document. To cause this to be done you simply have to say

```
\FiguresLast
```

at some point before you call `\figure` and its relatives, and then put

```
\PrintFigures
```

at the end of the document. `\FiguresLast` causes `\figure` to save the definition of your figure in an auxiliary file instead of immediately creating the figure, and `\PrintFigures` reads the material back in from that file. If you say `\PrintFigures` without first saying `\FiguresLast` nothing will happen. You can return to having the figures printed where they are defined by saying `\FiguresNow`. Remember, you have to have said `\FiguresLast` for `\PrintFigures` to read any stored figures back in.

Tables can also be saved until the end of the document by saying `\TablesLast` and then `\PrintTables`. `\TablesNow` is the opposite of `\TablesLast`.

Figure 1.17: A bizarrely placed figure for Section 1

One important note: Since the figure or table material is copied to the file in which it is held one line at a time it is important to use `\Caption` and `\endCaption` for long captions instead of `\caption`. If you use `\caption` TeX will try to read the whole caption at once, which may result in the error message "`! TeX capacity exceeded, sorry.`"

9.7. Nonstandard Figure and Table Placement

On rare occasions you may need to have a figure or table from one section of your document appear in another section of the document. For example, if the introductory section is very short but has a figure, that figure might end up at the top of the first page when you really want it to appear at the top of the second page. You can cause the figure to come out later by moving it farther back in the manuscript file, but if it is moved into the next section it will then have the wrong figure number.

The solution to this problem is to have a way to override or modify the way figure and table numbers are assigned. Since the first part of the figure or table number is the chapter or section number you need to change this by saying

`\setSectionID{`⟨something-else⟩`}`

This changes the section number part of the figure and table numbers to whatever you use as ⟨something-else⟩. It will also effect equation numbers, so it is best to have this inside of a group, so that the change goes away at the end of the group. Next you have to change the figure or table number. TeXsis keeps count of tables and figures in global variables, and increments these counters globally when a new figure or table is created. Thus just changing the counter inside a group won't work, because the counter outside of the group would still be modified. To temporarily change the value of such a counter you use

`\setcnt{`⟨counter⟩`}{`⟨value⟩`}`

where ⟨value⟩ is the temporary value to assign to the counter ⟨counter⟩. This should be done inside of a group, and when the group ends the counter will go back to its previous global value. You should only use `\setcnt` once within a given group!

An example will make this procedure clear. Figure 1.17 is a part of Section 1, but we wanted to show it much later in this Manual. It was created in this section by saying:

```
\begingroup       % changes are temporary
\setSectionID{1}  % changes the section ID
\setcnt\fignum{16}% figure counter is 1 less than we want
\figure{intro.17}
\vskip 2in
\caption{A bizarrely placed figure for Section~\use{sect.intro}}
\endfigure
\endgroup         % everything goes back the way it was
```

In this case the figure number in the counter `\fignum` is set to one less than the value we want, so that it comes out correct when `\figure` increments the count. You can use `\setcnt` to modify any of TEXsis's global counters. The complete list is given in Section 12 (where the use of `\use` is also described).

By the way, you should never move a figure out of a *Chapter*, unless you move it to the end of the document, and the appropriate way to do that is with `\FiguresLast`.

10. Letters, Memos and Referee Reports

The commands `\letter`, `\formletters`, `\memo` and `\referee` initialize TEXsis for producing letters or memos. The letter formats follow closely the letter format given in Appendix E of *The TEXbook*.

10.1. Letters

`\letter` is designed for typing basic business letters on letterhead stationary. It uses the following commands:

`\letter` Initialize the letter format.

`\letterhead{`⟨xxxx⟩`}`
Print telephone extension ⟨xxxx⟩ and current date. The default is to position these for BNL letterhead stationary, but you can change it in the `TXSsite.tex` file.

`\address` The name and address of the recipient should follow, in `\obeylines` format. The first line is used as the name of the addressee in a running headline for pages after the first.

`\body` Begins the body of the letter. The next line should be the salutation, "`Dear`...."

`\text` A synonym for `\body`.

`\closing` Begins the closing, in `\obeylines` format. The first line following is the closing (e.g. "Sincerely Yours,") and the second is the sender's name. When the letter is printed room is left for a signature. Optional additional lines can be added after the name for the sender's title, etc.

`\annotations`
Optional annotations follow, in `\obeylines` format.

`\cc` Carbon copy. A "cc:" and the names following are printed, properly aligned.

`\Encl` Enclosure. This is like `\cc` but "Encl:" is printed for a list of enclosures.

`\ps` Optional postscript follows. This is not in `\obeylines` format.

`\endletter`
Ends the letter. You can say `\bye` instead.

`\letter` automatically sets up a running headline for all pages after the first, containing the addressee's name from `\address`, the date, and the page number. If you wish, you can turn off this running headline by saying `\nopagenumbers`.

The use of these macros is illustrated by the following example:

```
\letter
\letterhead{7734}
\address
Prof.~D. Knuth
Stanford University
Stanford, CA~94305
\body
Dear Prof.~Knuth:

     We enclose the documentation and source code for a \TeX\
macro package which we have developed for physics papers.

\closing
Sincerely yours,
Eric Myers
Frank E.~Paige
\ps This stuff is also available by anonymous ftp.
\endletter
```

By default the closing material is indented about 3.5 inches. The actual amount is set by `\longindent`, which you can change. You should put the change on the same line as the `\closing`, like so:

```
\closing\longindent=0pt
Sincerely yours,
Joe User
```

This example puts the closing material flush with the left margin. You should not put `\longindent=0pt` on the line following, because `\closing` uses that line as the closing salutation (like "Sincerely,").

If a letter is somewhat too short or too long to fit on a page, its length can be adjusted indirectly by changing the width of the text. To do this insert `\letterwidth{`⟨width⟩`}` immediately after `\letter` to set the width of the text. This also automatically adjusts the margins and the placement of the telephone extension and the date.

10.2. Form Letters

The `\formletters` layout makes it simple to send identical or similar letters to a list of people. The body of the form letter is stored in one file using the standard TEXsis letter format, while the control commands and a list of addresses are put in a second file. The body of the letter should begin with `\body` and end with `\endletter`, and it can contain an arbitrary number of control sequences defined for each recipient. It can make use of all the features of the letter format. For example it might use

```
\body
Dear \Whoever:

\if P\Job
     I wish to apply for a postdoctoral position
\else\if F\Job
     I wish to apply for a faculty position
\else
     I wish to apply for a postdoctoral or faculty position
\fi\fi
at your institution.  Enclosed you will find my curriculum vitae,
a list of publications, and ...

\closing
Sincerely yours,
Joe Gradstudent
\Encl Publication list
Curriculum vita
\endletter                 % don't say \bye here!
```

This example addresses the letter to `\Whoever`, and it uses the first text if `\Job` is `P`, the second if `\Job` is `F`, and the third otherwise. Thus form letters are very flexible.

The mailing list file initializes the form letter macros and contains the list of addresses. TEXsis should be run on this file, not on the text of the letter! For the example above it would have the form:

```
\formletters{\letterhead{7734}}{letterbody.tex}

\BL     Prof.\ John Doe
        Department of Physics
        University of Nowhere
        Nowhere, OK 73019 USA
        \V\Whoever{Professor Doe}
        \V\Job{F}\EL

\BL     Dr.\ Buckaroo Banzai
        Physics Department
        The Banzai Institute
```

```
          Grovers Mill, NJ 08544
          \V\Whoever{Dr.\ Banzai}
          \V\Job{P}\EL

\BL       ...
          ... \EL
\endform
\bye
```

The first argument of `\formletters` is something like the usual `\letterhead` macro, with your phone number. Depending on your letterhead this might have multiple parameters or might even be several control sequences (all enclosed in a set of curly braces, of course). The second argument to `\formletters` is the name of the file containing the body of the letter. The idea is that the stuff in the first argument gets put into the letter *before* the address, while the body of the letter (read from the file) is what goes after the address. Each letter begins with `\BL` ("Begin Letter") and has the name of the addressee on the same line and the address on the following lines. Any "Variables" used in the letter can be defined using `\V`, which is just an abbreviation for `\gdef`. If necessary, variables can also be defined before the `\BL`. Each address in the mailing list is ended by `\EL` ("End Letter"). When the `\EL` is executed it reads in the file containing the text for the letter and actually produces the letter.

Any address in the list can be ignored simply by replacing the `\BL` with `\IL` ("Ignore Letter"). In particular, this can be done by a simple global edit after each batch of letters is made, and the same file can then be used to send a second mailing by changing `\IL` back to `\BL` for a select subset of addresses.

Mailing Labels

The same address file can be used to produce mailing labels instead of letters by replacing `\formletters{...}{...}` with `\formlabels`. It is assumed that there are three columns of names and addresses to fit on standard photocopy machine mailing labels. With the default parameters nine labels each 1 in. high are printed per column with the top and bottom rows left blank. This can be changed, and the alignment of the labels can be adjusted, by setting appropriate values for the following dimensions:

```
\fullHsize=8.500in    % hsize for label page
\fullVsize=9.999in    % vsize for label page
\lblHsize=2.833in     % width of standard label
\lblVsize=1.000in     % height of standard label
\lblVoffset=0.100in   % vertical starting position
\lblHoffset=-0.500in  % horizontal starting position
```

Choosing appropriate values may require some experimentation with your printer and labels. When the correct values are found, they can be set in the file `TXSsite.tex` before compiling the format (see Appendix A).

10.3. Memos

`\memo` sets up a special format for memoranda, including a header at the top of the page with the name of your organization and the word "MEMORANDUM". The organization name defaults to "PARKS COLLEGE" at this installation but this can be changed by

`\def\ORGANIZATION{⟨name⟩}`

(This can be done once in the file `TXSsite.tex` before compiling TEXsis, as described in Appendix A, or you can explicitly put the name of your organization in your file.) The commands for the memo format are:

`\memo` Initialize TEXsis for memo format.

`\Date` Prints the word "Date:" and the date, which must be on the same line. If no date is given, then the current date is used. If you'd rather have the date in the upper right hand corner, as in a letter, use `\letterdate` instead of `\Date`.

`\To` Prints the word "To:" and the name(s) which follow.

`\From` Prints the word "From:" and the name(s) which follow

`\Subj` Prints the word "Subject:" and the subject which follows. The subject may be several lines long. The first line of the subject field is used in a running headline at the top of all pages following the first.

`\n` New line for `\To`, `\From`, or `\Subj`.

`\nopagenumbers`
Turns off the running subject headlines and the page numbers. This must be typed after `\memo` since it is redefined by the memo format.

`\Text` Begins the text of the memo. You can also say `\body`.

`\annotations`
Optional annotations follow, in `\obeylines` format.

`\cc` Carbon copy. A "cc:" and the names following it are printed, properly aligned in `\obeylines` format.

`\Distribution`
This is like `\cc` but prints "Distribution:" instead.

`\Encl` This is like `\cc` but prints "Encl:" instead, as for letters.

`\ps` Optional postscript follows. This is not in `\obeylines` format.

`\endmemo` Ends the memo. You can say `\bye` instead.

All of these commands with mixed case names have synonyms which are the same name but in all lower case letters. The use of the `\memo` format is illustrated by the following example:

```
\memo
\Date
\To   \TeXsis\ Users
\From E. Myers, F.E. Paige
\Subj \TeXsis\ 2.15
\text
        \TeXsis\ 2.15 is now available. The new version adds
several features and corrects various bugs. Documentation is
also available.
\Distribution
A. Blake
I. Harrity
B. Nichols
...
\endmemo
```

10.4. Referee Reports

`\referee` sets up a document format very much like the `\memo` format, although there are some important differences. Referee reports are supposed to be confidential, so instead of printing the name of your organization at the top of the first page it puts the words "Referee Report." In the `\referee` format you can use the same commands that are used in the `\memo` format, such as `\To` and `\Date`, but if you use `\From` a warning message will be printed on your screen. (This is only a warning, it still does what it has been asked to do!)

There are also some additional commands for the referee report format:

`\title` The title of the paper being reviewed should follow this on the same line. This is similar to `\Subj` in the memo format.

`\author` Prints "AUTHOR:" and the name that follows on the same line.

`\authors` Same as `\author` for a paper with more than one author.

`\MSref` Prints "MANUSCRIPT:" and the manuscript reference number which follows.

The text of the report should begin with `\text` or `\body`, just as for a memo, and the whole document should end with `\endmemo` or `\bye`. An example of the referee report format is given by the following:

```
\referee
\date    1 April 1992
\title   \TeXsis\ -- \TeX\ Macros for Physicists
\authors E.~Myers and F.E.~Paige
\MSref   LP-3105
\text
This is pretty good, but it's too long
for Physical Review Letters.
\bye
```

This produces a report which looks something like:

Referee Report

DATE:	1 April 1992
TITLE:	TeXsis – TeX Macros for Physicists
AUTHORS:	E. Myers and F.E. Paige
MANUSCRIPT:	LP-3105

This is pretty good, but it's too long for Physical Review Letters.

11. Double Column Formats

Typesetting text in multiple columns on a single page is more involved than setting text in a single page-wide column, but since many publishers of "camera-ready" conference proceedings require double column typesetting TEXsis includes a set of macros for doing this. The macros automatically balance columns and provide for equations and floating insertions (such as figures and tables) either within a single column or across both columns. By changing a single instruction it is also possible to take a document prepared for double columns and print it in single column mode. The converse is not true. Typesetting text in double columns with automatic column balancing is a nontrivial task. While the macros generally behave as intended, more attention to the layout of the document is required, and even then they will sometimes fail; it is not possible to divide three lines into two balanced columns! In such cases it is necessary to rearrange text or to explicitly select column or page breaks elsewhere in the document.

The basic double column macros set up the double column format, specifying the width of the columns and, implicitly, the space between columns, and allow you to switch between single and double columns modes:

`\setdoublecolumns{`⟨p-width⟩`}{`⟨p-height⟩`}{`⟨c-width⟩`}`
: Set up the double column format. ⟨p-width⟩ is the overall page width, ⟨p-height⟩ is the page height, and ⟨c-width⟩ is the column width.

`\doublecolumns` Begin double column format.

`\longequation` Begins a "long" equation which cannot fit within one column and therefore must stretch across both columns.

`\endlongequation`
: Ends a long equation.

`\newcolumn` Force a column break, much like `\newpage` forces a page break.

`\leftcolrule` Draw a rule across the bottom of the left column.

`\rightcolrule` Draw a rule across the top of the right column.

`\enddoublecolumns`
: End double column mode, making balanced double columns, and return to single column mode.

`\setdoublecolumns` redefines the `\output` routine, so it should be called immediately after you say `\texsis`, before anything is output, and it should not be called unless double column output is to be used. The `\hsize` and `\vsize` should not be set separately.

A simple example of the use of these double column macros is:

```
\setdoublecolumns{\hsize}{\vsize}{0.47\hsize}
\doublecolumns\tenpoint
{\bf BISMUTH} (Ger.\ {\it Weisse Masse}, white mass; later
{\it Wismuth} and {\it Bisemutum}, Bi; at.~wt. 308.981;
at.~no. 83; m.p.  271.3$^\circ$C; b.p. 1560 $\pm$ 5$^\circ$C;
sp.~gr. 9.747 (20$^\circ$); valence 3 or~5.  In early times
bismuth was confused with tin and lead.  Claude Geoffroy the
Younger showed it to be distinct...
        ...
...  High purity bismuth metal costs about \$4/lb.
\enddoublecolumns
```

The page size is just set to be the existing `\hsize` and `\vsize`. The column size is chosen to be slightly less than half of the `\hsize` to allow some space (called the "gutter") between the two columns. The result is:[17]

BISMUTH (Ger. *Weisse Masse*, white mass; later *Wismuth* and *Bisemutum*, Bi; at. wt. 308.981; at. no. 83; m.p. 271.3°C; b.p. 1560 ± 5°C; sp. gr. 9.747 (20°); valence 3 or 5. In early times bismuth was confused with tin and lead. Claude Geoffroy the Younger showed it to be distinct from lead in 1753. It is a white, crystalline, brittle metal with a pinkish tinge. It occurs native. The most important ores are *bismuthinite* or *bismuth glance* (Bi_2S_3) and *bismite* (Bi_2O_3). Peru, Japan, Mexico, Bolivia, and Canada are major bismuth producers. Much of the bismuth produced in the U.S. is obtained as a by-product in refining lead, copper, tin, silver, and gold ores. Bismuth is the most diamagnetic of all metals, and the thermal conductivity is lower than any metal, except mercury. It has a high electrical resistance, and has the highest Hall effect of any metal (i.e., greatest increase in electrical resistance when placed in a magnetic field). "Bismanol" is a permanent magnet of high coercive force, made of MnBi, by the U.S. Naval Ordnance Laboratory. Bismuth expands 3.32% on solidification. This property makes bismuth alloys particularly suited to the making of sharp castings of objects subject to damage by high temperatures. With other metals, such as tin, cadmium, etc., bismuth forms low-melting alloys which are extensively used for safety devices used in fire detection and extinguishing systems. Bismuth is used in producing malleable irons and is finding use as a catalyst for making acrylic fibers. When bismuth is heated in air it burns with a blue flame forming yellow fumes of the oxide. The metal is also used as a thermocouple material (has highest negativity known), and has found application as a carrier for U^{235} or U^{238} fuel in atomic reactors. Its soluble salts are characterized by forming insoluble basic salts on the addition of water, a property sometimes used in detection work. Bismuth oxychloride is used extensively in cosmetics. Bismuth subnitrate and subcarbonate are used in medicine. High purity bismuth metal costs about $4/lb.

Another example of a double column document is the `\NorthHollandTwo` format mentioned in Section 7.3. This sets up a document with an overall width of 21.3 cm, a page length of 27.1 cm and a column width of 10.0 cm. (Note that this is just about as wide or a little wider than most small laser printers can print, so provisions have been made for shifting the printed page horizontally to print one column at a time if necessary.) A document printed with this format might have the following basic structure:

```
\NorthHollandTwo   % set up double columns
\titlepage         % but still single column mode
   ⟨title page material⟩
\endtitlepage      % ends the title material

\doublecolumns     % now start double column mode
```

```
    ⟨text of paper⟩
\enddoublecolumns  % balance the final columns

\bye
```

Saying `\NorthHollandTwo` only defines the double column layout, it does not turn on double column mode. The title material is printed in single column format across the full page width, then the main body is printed in double column format once `\doublecolumns` appears. It is important to say `\enddoublecolumns` at the end of the document to balance the columns on the last page. This same document can be printed in a single column layout simply by saying `\NorthHolland` instead of `\NorthHollandTwo`. In this case `\doublecolumns` and `\enddoublecolumns` do nothing (they are, in fact, the same as saying `\relax`.) However, if you change back to `\NorthHollandTwo` they will come back to life and perform as required. Similarly, in double column mode `\newcolumn` forces the end of a column and begins the next column, while in single column mode it also does nothing.

In double column format displayed equations that fit within a single column are typed with the usual `$$ ... $$`. If an equation is too long then it may have to be printed across both columns. This is easily done by saying `\longequation` before the `$$` which begins the equation and `\endlongequation` after the `$$` which ends the equation:

```
\longequation
$$ ⟨equation⟩ $$
\endlongequation
```

The command `\longequation` invokes `\enddoublecolumns` to end double column mode and balance the partial double columns, and it will also draw a ruled line across the bottom of the left column (this is the default, but some document layouts may not draw this rule). After the equation has been completed `\endlongequation` will call `\doublecolumns` to go back to double column mode, and it will also draw a rule across the top of the right column (again, some document formats may omit this rule). When dealing with long equations that span both columns you should keep in mind that switching frequently between single and double columns will almost certainly cause trouble in balancing the double columns. The effect of the ruled lines above and below a long equation is to guide the reader's eye from the incomplete left column to the right column above the equation, and to the top of the incomplete right column from the left column below the long equation. The default style of these rules is the same as is used by Physical Review.

Note that in single column mode `\longequation` and `\endlongequation` do absolutely nothing (they are the same as `\relax`), so it is still easy to print the document with a single column layout if desired. The rules described above will not be drawn in single column mode.

Figures, tables, or other floating insertions can be inserted within a single column using the same syntax as Plain TeX. The macros are redefined by `\doublecolumns` to work for two columns:

`\midinsert` Insert at the current position if there is enough room; otherwise inserts at the top of the next page.

`\topinsert` Insert at the top of the current column. Insertions are carried over to the same column of the next page if necessary.

`\pageinsert` Insert a full column.

`\endinsert` End any insertion.

Both `\topinsert` and `\pageinsert` check to see how much material has been output and then place the insertion in the left or right column as appropriate. You can force a particular choice by placing *within* the insertion one of the following:

`\forceleft` Force an insertion to be in the left column.

`\forceright` Force an insertion to be in the right column.

Since insertions in the left and right columns are handled separately, an insertion which fails to fit on a page is moved to the same column of the next page and not to the next column. This means that if there are several figures per page, then the figures can easily appear in the wrong order. Also, a `\topinsert` which follows a wide equation on a page will be placed after the equation, not at the top of the page. Such problems can easily be corrected by moving the insertion to a better place in the manuscript file.

There are also macros to make insertions across both columns for wide figures or tables:

`\widetopinsert` Insert across both columns at the top of the page.

`\widepageinsert` Insert a full page across both columns.

(there is no such thing as `\widemidinsert`). In single column mode `\widetopinsert` and `\widepageinsert` behave just like `\topinsert` and `\pageinsert` to simplify producing a single-column version of a double-column document.

The standard TEXsis macros for table and figure insertions described in Section 9.3 use `\topinsert` and so make insertions within a column in double column format. There are also macros to make two-column tables and figures:

`\widetable{`⟨label⟩`}`
Insert a two-column table with the tag ⟨label⟩ at the top of the current page.

`\widefigure{`⟨label⟩`}`
Insert a two-column figure with the tag ⟨label⟩ at the top of the current page.

`\widetopfigure{`⟨label⟩`}`
A synonym for `\widefigure`.

`\widefulltable{`⟨label⟩`}`
Insert a two-column, full-page table with the tag ⟨label⟩.

`\widefullfigure{⟨label⟩}`
Insert a two-column, full-page figure with the tag ⟨label⟩.

These also behave correctly in single column document formats. See Section 9.3 for more information.

The double column output routine places a running headline and footline on each page just like Plain TeX. But the standard TeXsis headlines and footlines rely on marks and so do not work correctly with double column output. (The marks get reset every time a partial page is output.) This discrepancy will be fixed in a future release of TeXsis. If you wish to use running headlines and footlines, you must redefine `\headline` and `\footline`. For nothing on page 1, the author's name on even pages, and the title on odd pages, you could use

```
\headline={\myhead}
\def\myhead{%
  \ifnum1=\pageno {\hfill} \else
  \ifodd\pageno
    {\hss{\tenit⟨Title⟩}\hss\llap{{\tenrm\folio}}}%
  \else
    {\rlap{{\tenrm\folio}}\hss{\tenit⟨Author⟩}\hss}%
  \fi\fi}
```

Otherwise, just say `\nopagenumbers`.

11.1. Twin Page Output

It is often desirable to produce preprints or other documents in a compact format with two pages on each physical page in landscape orientation. While this can be done with `\setdoublecolumns`, a much simpler approach is sufficient. TeXsis contains a macro, `\twinout`, which redefines the output routine to produce two pages per page without attempting to balance columns or to allow spanned equations, tables, etc., thus avoiding many of the possible problems with the more powerful double column macros. `\twinout` can be used with any of the standard formats described in Section 7 or even with Plain TeX. Since `\twinout` does not select fonts, the user must do so:

```
\texsis
\tenpoint
\def\Tbf{\twelvebf}
\def\tbf{\tenbf}
\twinout
      ⟨text⟩
```

Alternatively, the `\twinfmt` command can be used to produce any TeXsis format in `\twinout` layout with 10 pt fonts. Thus the above commands can be replaced by

```
\texsis
\twinfmt
      ⟨text⟩
```

Here `\twinfmt` automatically sets up 10 pt fonts and appropriate values for the title bold face fonts, just as in the previous example.

`\twinout` use the following dimensions:

`\twinhsize`	Total width for twin column output.
`\twinvsize`	Column height for twin column output.
`\twincsize`	Column width for twin column output.
`\normhsize`	Normal value of `\hsize` in portrait mode.
`\normvsize`	Normal value of `\vsize` in portrait mode.

All of these have sensible default values for 8.5 in × 11 in paper but can be reset before `\twinout` is called. The "normal" sizes should print properly with the default device driver and printer. Appropriate default values for all these dimensions can be set in `TXSsite.tex`.

`\twinout` puts `\LandscapeSpecial` on each page. This is null by default but can be redefined to be an appropriate `\special` command to produce landscape output with your device driver and printer. For example, with `dvips`[13] and a PostScript printer, use

```
\def\LandscapeSpecial{\special{papersize=11in,8.5in}}
```

A default value can be set in `TXSsite.tex`.

`\twinprint` is a substitute for `\preprint` which sets the title, author, and other titlepage material full size in landscape mode and then switches to `\twinout` twin column output with 10 pt type for the text of the paper. A typical preprint might read:

```
\twinprint
\titlepage
\title
        ⟨paper title⟩
\endtitle
\author
        ⟨author's name⟩
        ⟨address⟩
\endauthor
\abstract
        ⟨text of abstract⟩
\endabstract
\endtitlepage
        ⟨text of paper⟩
\bye
```

This produces a nice, compact format for preprints.

12. Labels and Tags

It has already been noted that one of the most useful features of TeXsis is that it automatically numbers things like equations and references and lets you refer to these things by means of symbolic names or "labels." This section explains how to get the most use out of this labeling mechanism.

Here is a rough outline of how labeling or "tagging" works. TeXsis keeps track of numbers for equations, figures, tables, and other useful things in counters. Whenever you create a new equation, figure, table or reference the appropriate counter is incremented. At the same time, a new macro instruction is defined and its definition is simply the (new) value for the counter. Thus if you create a displayed equation and label it "Newton" then a macro called "`\Eq.Newton`" is created and it is automatically defined to be whatever the next number is for the equation.

[The actual mechanism is somewhat more subtle. Normally you can only use letters in macro names, but TeXsis uses `\csname ... \endcsname` to allow numbers and punctuation to be used in labels. Also, an extra `@` is added at the beginning and end of each macro name. If this were not done then, for example, using "`end`" as a label would redefine `\end`, and you would not be able to get out of TeX!]

Once a label macro is defined you can use it to refer to the thing it labels, and it will yield the proper number. In the case of the equation called "Newton," saying `\Eq{Newton}` is really saying `Eq.~(\Eq.Newton)`. (Again, since punctuation is not allowed in names, TeXsis actually uses `\csname ... \endcsname` in the definition.) References, equations, figures, tables, etc., all produce their numbers in this way.

Each of the TeXsis macros which uses labels attaches to the label a unique prefix identifying the kind of object labeled. These objects and the corresponding prefixes are shown in Table 12.1. Because of these prefixes the same label can be used for different kinds of objects. For example, the name "Witten" can be used as a label for an equation and as a reference, since their internal labels are `Eq.Witten` and `Ref.Witten`, respectively. The prefixes are automatically inserted by TeXsis macros like `\Eq` and `\cite`.

At the most primitive level you can always associate a label with some value by saying `\tag{`⟨label⟩`}{`⟨value⟩`}`. You can then reference the value, whatever it is, by it's name using the command `\use{`⟨label⟩`}`. In this case if there is to be a prefix in the label you must put it in explicitly. Thus to say just the number of the equation labeled "Newton" without any parentheses or anything you would say `\use{Eq.Newton}`. To get the number of the reference labeled "Knuth" you would say `\use{Ref.Knuth}`, which produces "1" because "Knuth" is the label of the first document we referenced in this manual and "Ref" is the prefix used for references.

One particularly useful way to use `\tag` and `\use` is to refer to some topic which is discussed on some other page in the document. On the page you want to reference you could define the tag "foo" to be the page number by saying `\tag{foo}{\folio}`. Then somewhere else in the document if you want to refer to the information on that page you could do so with `\use{foo}`, as in "This difficulty is discussed in more detail on page `\use{foo}`." Then even if the document is rearranged so that the page numbers change the page number you referred to will come out correct.

Table 12.1: Prefixes for labels and the corresponding counters for various objects.

Entity	Tag Prefix	Counter
Corollary	`Cor.`	`\corollarynum`
Definition	`Df.`	`\definitionnum`
Equation	`Eq.`	`\eqnum`
Figure	`Fg.`	`\fignum`
Lemma	`Lem.`	`\lemmanum`
Reference	`Ref.`	`\refnum`
Table	`Tb.`	`\tabnum`
Theorem	`Thm.`	`\theoremnum`

Knowing about `\use` and the prefixes in Table 12.1 also lets you change how the macros that refer to these objects work. For example, to refer to figure 9.1 in section 9.1 you would normally say `\Fig{Penguin}`. This produces "Fig. 9.1," but suppose you want it to say the full word "figure" instead of just "Fig." You can make this change throughout the document simply by changing the definition of `\Fig` to:

```
\def\Fig#1{figure~\use{Fg.#1}}
```

Then "`\Fig{Penguin}`" would produce "figure 9.1."

Whenever you create most of the objects listed in Table 12.1 a label is automatically created. Chapters, sections, and subsections, however, are not automatically labeled because their values are not ordinarily changed. If you want to remember the number of a chapter or section of your document to be able to refer to it somewhere else, put `\label{`⟨label⟩`}` in the text of the title of the chapter or section. `\label{`⟨label⟩`}` is the same as `\tag{`⟨label⟩`}{`⟨counter⟩`}`, where ⟨counter⟩ is the current section or chapter number, as appropriate. For example you might have:

```
\section{Supersymmetry and Supergravity \label{sect.SUSY}}
```

To refer to that section later on you would say `\use{sect.SUSY}`. Note that `\label` does not insert any prefix in front of the label, so you must include it explicitly if you want one.

12.1. Forward References

If you label an equation or a figure or a section you can refer to the number of that object later in the text, but what if you want to refer to the thing before it has been tagged? Then the label won't exist yet. This is called making a forward reference. To correct for this TEXsis also writes the definition of every label to an auxiliary file, which has a name ending in `.aux`. The label will still be undefined the first time you make the forward reference, but the next time you run the document through TEXsis it will automatically read in the auxiliary file at the beginning of the job and define the label. Thus on the second pass the label will already be defined when it is called for. Having to make a second pass may sound like a lot of extra trouble, but when you are editing a document and putting the finishing touches on it you will find you end up making the extra pass without thinking about it. Just don't delete the `.aux` file between runs.

If you try to reference a label that does not exist you'll be told so at the end of the run (when you say `\bye`) and another pass will be suggested). Undefined references are also noted in the log file when they occur, and the label itself is inserted into the output text in boldface to make it easier for you to track down the problem. In `\draft` mode a mark is also put in the right margin to help you find the problem.

13. Miscellaneous Macros

There are a number of useful macros in TEXsis which don't fit into any of the topics covered in previous sections but which still deserve some discussion. We will simply lump them all together in this final section.

13.1. `\draft` mode

Saying `\draft` turns on a number of features which are useful for making a draft copy of a document. The disclaimer "`Preliminary Draft`" is added to the bottom of each page of the document, along with the page number and a time and date stamp. The page number will let you keep track of the page order even when using a document format which does not print page numbers (such as the `\nuclproc` or `\NorthHollandTwo` camera-ready formats). The date/time stamp can help you tell later drafts from earlier versions of the same document. `\draft` sets `\eqnotracetrue` so that equations are marked with their internal labels in the right margin next to the equation number (as described in Section 4). Other kinds of errors, such as trying to use undefined labels for citations equations, are also marked in the right-hand margin. `\draft` also sets `\overfullrule=1em` so that over full `\hbox`es are marked with a black box. Without `\draft` initializing TEXsis with `\texsis` sets `\overfullrule=0pt` so that over-full hboxes don't get these kinds of marks.

13.2. Ignoring blocks of text

Sometimes you'd really like to take out a whole paragraph or an even larger section of a document, but only temporarily, and you don't want to actually delete it from the manuscript file. Recalling that in TEX everything on a line following the comment character "`%`" is discarded, you could put a `%` on each line to be ignored, but that could be a lot of work. Instead, you can use `\comment`. The syntax is:

```
\comment/*  anything in here is ignored */
```

Everything between the `/*` and the `*/` is completely ignored, while by removing the `\comment/*` and the `*/` the text can easily be restored to it's previous state. You can include several paragraphs of text in the material to be ignored, or even several pages, but with one important restriction: All the curly brackets `{` and `}` must match in the text to be ignored. If there is a `{` with no closing `}` in the commented text then TEX will skip the `*/` while searching for the closing `}`!

Along the same lines is `\Ignore`, which will ignore everything following it up to `\endIgnore` (with the same restrictions above about matching grouping brackets). The useful thing about these is that if an `\endIgnore` is encountered before an `\Ignore` then the `\endIgnore` is simply ignored. Is that too confusing? What this means is that if you have a document with a large block of conditional text that you may or may not want included you can choose to include the text by simply removing (or commenting out) the `\Ignore`. If the block of conditional text looks like

```
⟨unconditional text⟩
\Ignore
Having obtained these eigenvalues we would like to
know their degeneracies.  For a given $l$ the ...
...
It is relatively easy to show that the number of
independent components of such a tensor is ${N+l \choose l}$.
\endIgnore
⟨more unconditional text⟩
```

then even if the block of text to be ignored spans many pages it can be restored simply by changing the "`\Ignore`" to "`%\Ignore`." Then the `\Ignore` is itself ignored, and when TeX reaches the `\endIgnore` this also does nothing. The effect is the same as if these two commands were completely removed from the manuscript file, but it is easily reversed.

Why are there two ways to ignore blocks of text? `\comment` was created first, and some programmer-types may prefer it because of the similarity it bears to comments in the C programming language. `\Ignore` is newer and probably a little more useful, but `\comment` will remain for those who want to use it.

13.3. Checkpoint/Restart

The checkpoint/restart feature allows long documents to be printed in pieces. You can, for example, process and print the first chapter of a thesis (or a book) and then later do the same for following chapters. Saying `\checkpoint{NAME}` saves the settings of important counters like the current page number and the last equation number in a file called `NAME.chk`. Saying `\restart{NAME}` reads in this file and resets the counters to their previous values.

As an example consider a document with two major chapters. Using a Master File you would first process the first chapters with the following:

```
\thesis
\input chap1
\checkpoint{chap1}
\bye
```

With this printed you could resume at the next page number with the correct subsequent numbers for equations, figures and tables by saying:

```
\thesis
\restart{chap1}
\input chap2
\bye
```

Normally `\checkpoint` and `\restart` are silent, but you can say `\endstat` to see the values of the counters saved by `\checkpoint`. The output looks like this:

```
% LAST PAGE NUMBER IS 63.
% LAST CHAPTER NUMBER IS 0.
% LAST EQUATION NUMBER IS 47.
% LAST FIGURE NUMBER IS 1.
% LAST REFERENCE NUMBER IS 9.
% LAST SECTION NUMBER IS 13.
% LAST TABLE NUMBER IS 0.
```

Modern laser printer drivers or filter programs like `dvi2ps` or QTeX have options that let you select which pages of a document are printed. This is another way to selectively print only part of a document and still get the page and equation numbers correct.

13.4. Other Goodies...

`\topspace` Use this like `\vskip` to create a blank space to the top of a page (because any space inserted with `\vskip` is thrown out if it appears at the top of a page).

`\unobeylines`
The opposite of `\obeylines`

`\unobeyspaces`
The opposite of `\obeyspaces`

`\pagecheck{`⟨dimen⟩`}`
Checks the amount of space remaining on the current page, and if this is less then ⟨dimen⟩ then skips to a new page.

`\leftpar{`⟨text⟩`}`
Turns the ⟨text⟩ into a paragraph with a ragged left margin.

`\undertext{`⟨text⟩`}`
Underlines the ⟨text⟩.

`\theBlank{`⟨length⟩`}`
Creates an underline of the specified length where one can "fill in the blank." For example, saying " `Name: \theBlank{6cm}` produces: "Name: ______________________."

`\loosebox{`⟨box⟩`}`
Puts a "loose" box around the ⟨box⟩, with about one `\jot` of space between the edge of the ⟨box⟩ and the enclosing box. What you do with the resulting box is up to you.

`\tightbox{`⟨box⟩`}`

Puts a ruled box around the argument, with no intervening space. For example, `\tightbox{\hbox{Boo}}` produces "$\boxed{\text{Boo}}$". In contrast, `\tightbox{\loosebox{Boo}}` gives "$\boxed{\ \text{Boo}\ }$."

A. Customization and Installation of TEXsis

(TEXsis version 2.15, as of 19 June 1992)

A.1. Customization

There are three different mechanisms for making modifications to the basic TEXsis macro package. It is important to use one of these mechanisms rather than modifying the original code to allow any changes to be transferred to a new version of TEXsis.

A.1.1. Local Modifications

TEXsis will read the file `TXSmods.tex`, if it exists, from the current directory or search path every time you run TEXsis. This lets you make your own custom modifications or additions to TEXsis. For example, say you have a directory specifically for a particular project you are working on and you are writing several different papers or other documents related to this project. You could add a file called `TXSmods.tex` to this directory to contain extra macro definitions that are used in all of these documents, and these will be loaded automatically whenever you run TEXsis in this directory. If you run TEXsis from another directory these macros will not be loaded.

A.1.2. Site Modifications

If a file called `TXSsite.tex` exits in the current directory at the time the TEXsis format file is created (see below) then it is automatically read in at this time. The purpose of this file is to allow you to add definitions and macros that are local to your computer installation. For example, in this file you would define the control word `\ORGANIZATION` to give the name of your organization or institution. `\ORGANIZATION` is used at the top of memos and at the top of the title page of a `\preprint`. You might also put a customized `\letterhead` macro in this file, and you may well want to change the definition of `\LandscapeSpecial`, which controls how your laser printer is put into landscape mode. Note that the file `TXSsite.tex` is only read *once*, when the TEXsis format file is created, not every time TEXsis is run.

A.1.3. Patches

TEXsis will also automatically read from a file called `TXSpatch.tex`, if it exists, every time TEXsis is run (from the standard input path, or from the directory `TEXSIS_LIB:` in VMS). This file should contain additions or corrections ("patches") to TEXsis. This patch file should not be too long. If major corrections are to be made the format file can be re-generated — but using the patch file allows you to correct small errors without having to re-generate the format each time. The default patch file simply prints a message on the console giving the version number and date of the current TEXsis distribution.

A.2. Installation of TEXsis*

To use TEXsis or any large macro package effectively you should run `initex` to create a pre-compiled format. The general method of doing this is machine independent, but the specific commands depend on the operating system on your computer. This appendix gives the appropriate commands for a computer running UNIX, a VAX running VMS, and for an IBM PC running PCTEX under MS-DOS. It assumes that you are familiar with the operating system and that TEX and the Plain format have already been installed.

A.2.1. Quick Install for Unix and TEX 3.0

If you do not have a Unix system with TEX 3.0 (or greater) then you will have to use the detailed installation instructions given below. If you do have a Unix machine running TEX 3.0 you can install TEXsis simply by unpacking the files into a working directory (e.g. `/usr/local/lib/tex/texsis`) and:

- ☐ 1. Edit the file `Makefile` and change parameters as appropriate, following the directions given there;
- ☐ 2. (Optional) Copy the file `TXSsite.example` to `TXSsite.tex` and edit this file to contain appropriate site dependent information, such as definitions for `\ORGANIZATION` or `\letterhead`. [On MS-DOS systems the example site file is called `TXSSITE.EXA`.]
- ☐ 3. Give the command '`make install`' (or simply '`make`' first to be sure it works, and then '`make install`').

A.2.2. Detailed Installation

To install TEXsis follow the steps below or their equivalents for your computer and operating system. If you are installing a system-wide version of TEXsis then you may want to review the appropriate documentation on how TEX itself is installed on your system.

☐ 1. Create a separate directory to hold the TEXsis files. This is not strictly necessary, but it is strongly recommended, since there are more than 60 files for the source complete code and the documentation. On a Unix machine we recommend using `/usr/local/lib/tex/texsis` or `/usr/local/tex/texsis`, depending on where your other TEX files are kept. For a VAX/VMS system you should use a subdirectory of the main TEX directory. On an IBM PC it is best to use a top-level directory `C:\TEXSIS` on your hard disk, rather than a subdirectory of `C:\PCTEX`, to simplify backing up your files.

☐ 2. Copy all of the TEXsis files into this directory. The TEXsis source files all have names of the form `TXS`*xxx*`.tex`, and the documentation files all have names `TXS`*xxx*`.doc`. TEXsis style files end with the extension `.txs`. There will also be several other files in your distribution of TEXsis, such as `texsis.tex` and `Manual.tex`.

☐ 3. Make any necessary changes in the source code. You will probably want to copy the file `TXSsite.example` to `TXSsite.tex` and edit this file, changing the definition of `\ORGANIZATION`, `\LandscapeSpecial`, and the letterhead macros and (on VMS) defining `\TeXsisLib` to be the TEXsis directory and `\TXSpatches` to be the patch file in that directory. In some cases it may also be necessary to change some of the font definitions in the file `TXSfonts.tex`. Most distributions of TEX now use the "CM" (Computer Modern) fonts, but some places still use "AM" (Almost Modern) fonts. If you have problems with missing fonts look at the documentation in the file `TXSfonts.tex` and make appropriate changes.

☐ 4. Create the format file `texsis.fmt` by running `initex`. On a Unix machine the command is:

```
% initex texsis \&plain '\dump'
```

You must remember the "`\`" to escape the "`&`." On a VMS system the command is essentially the same:

```
$ INITEX TEXSIS &PLAIN "\dump"
```

For an IBM PC with PCTEX give the command,

```
TEX TEXSIS /I
```

and reply to the `*` prompt with "`\dump`." (In some versions of PCTEX the `TEX` command is replaced with `TEX386` or `TEX386b`. Check your manual to be sure.) TEXsis will also work with `emtex` or any other version of "big" TEX for the PC, but you will have to check the documentation for that particular implementation of TEX for specific instructions for creating a preloaded format.

In any case `initex` will load the Plain TEX format and then the various pieces of TEXsis, displaying a message for each piece. Finally it will spew out a list of fonts used and other information. This should contain near the end the statement "No pages of output produced." (If it does not, something may be wrong with your distribution, although it is probably a minor problem. Check your editing of the `TXSsite.tex` file.) The preloaded format will be written to the file `texsis.fmt`.

☐ 5. Copy or move various files to where they need to be:

5.1. Copy the format file to the place where your version of TEX looks for pre-loaded formats. On Unix systems this is usually

`/usr/local/lib/tex/formats`.

or some variation of this. For VMS machines make sure the file is in `TEX$FORMATS`. On a PC the file goes into `\PCTEX\TEXFMTS`.

5.2. Copy the TEXsis style files (which end with the extension `.txs`) to the place your version of TEX usually looks for input. On Unix machines this should be `/usr/local/lib/tex/inputs` (or sometimes `/usr/local/lib/tex/macros`). For VMS machines make sure these files are in the directory `TEX$INPUTS`. On a PC the proper place is `\PCTEX\TEXINPUT`.

5.3. If you have any run-time patches to add to TEXsis copy the file `TXSpatch.tex` to the same TEX inputs directory. If there is a `TXSpatch.tex` from a previous release of TEXsis you should be sure to remove the old file if it is not being replaced.

☐ 6. Having created the format file, you need to make a command `texsis` to run TEX using it. The way to do this is very dependent on the operating system.

6.1. On a Unix system running TEX 3.0 or higher you need only to make a link to the executable `virtex` with the same name as the format you want to load. `virtex` probably lives in `/usr/local/bin` or some similar place. The command to make the link is:

```
ln /usr/local/bin/virtex /usr/local/bin/texsis
```

For TEX versions 2.9x you should probably consider building an undumped version of TEXsis. You will have to see the TEX distribution notes for a description of `undump`.

If you are going to create a "private" version of TEXsis rather than installing it for the whole system then you need to add your TEXsis directory to the environment variables `TEXFORMATS` and `TEXINPUTS` and define the command `texsis` as follows. For the C-shell, put in your `.cshrc` file:

```
alias texsis virtex \\\&texsis
setenv TEXSIS_LIB $HOME/texsis/
setenv TEXFORMATS ".:$TEXSIS_LIB:$TEXFORMATS"
setenv TEXINPUTS  ".:$TEXSIS_LIB:$TEXINPUTS"
```

(This assumes that `TEXFORMATS` and `TEXINPUTS` are already properly defined. If not, insert the proper directories.)

6.2. Under VMS, place in your `login.com` file (or in the system login file) the commands

```
$ DEFINE TEXSIS_LIB ⟨disk⟩:[⟨directory⟩]
$ TEXSIS :== $TEX$:TEX.EXE &TEXSIS_LIB:TEXSIS
```

where `TEX$` is the logical name for the main TEX directory. You can also put these commands in a TEX initialization `.COM` file.

Some versions of TEX on VMS have a `VIRTEX.EXE` in the TEX directory. For these versions it is necessary to create a new `TEXSIS.EXE` file from the `TEXSIS.FMT` file with the commands

```
VIRTEX "&TEXSIS"
\end
save texsis
```

`TEXSIS.FMT` can then be deleted. Your local TEX documentation should provide further information.

6.3. For PCTEX on an IBM PC you should find in the distribution a file `TEXSIS.BAT`. If you don't, create it. The file should contain:

```
ECHO OFF
\PCTEX\TEX  &TEXSIS %1 /F 25000
```

(Replace the `TEX` command with `TEX386` or `TEX386b` if appropriate for your machine. For PC-TEX versions 3.14 and greater the `/F 25000` is no longer needed.) Either put this file in a directory in your command path or add the directory it is in to your `PATH` statement. If you want to be able to load a patch file then you should also include a line such as

```
\def\TXSpatches{C:/TEXSIS/TXSpatch}
```

in the `TXSsite.tex` file. Because TEX and MS-DOS both use backslashes for their own purposes the path is specified with *forward* slashes.

After these steps have been carried out, TEXsis should be installed and available to you and to other users. The command

`texsis` *name*

should run TEXsis on the file *name*`.tex` and produce the file *name*`.dvi`, which can be printed like any `.dvi` file. A good test is to copy all the files `TXS`*xxx*`.doc` from the TEXsis directory into a directory to which you have write access and then to do

```
texsis Manual
```

On the first try this will produce several warning messages coming from undefined tags for forward references. These will be defined later in the document and saved in the file `Manual.aux`; running TEXsis a second time should produce this document with no errors.

B. Font Tables

The Computer Modern fonts that are used either by Plain TEX or by TEXsis and the required sizes for each are listed in Table B.1. The `cmmib10` and `cmbsy10` fonts may not exist in all installations, but they will not be loaded unless `\mib` is used, so they are not required. The command `\printfont{`⟨font⟩`}` can be used to print a table of the characters in any ⟨font⟩, e.g.,

```
\printfont{cmr10}
```

A separate file called `Fonts.tex` is included with the standard distribution of TEXsis. Running `Fonts.tex` through TEXsis will produce tables of all of the fonts used by your installation of TEXsis.

Most versions of TEX now use "CM" (Computer Modern) fonts, but some still use the older "AM" (Almost Modern) fonts. If you have problems with the fonts, try editing `TXSfonts.tex`, replacing all occurrences of `=cm` with `=am` and then recompiling TEXsis. Mixing the two types of fonts will produce errors.

Table B.1: Fonts used in TEXsis, with the design size names and the required sizes in points.

Base Name	Description	Sizes (pt)									
cmr10	Roman	5	7	9	10	11	12	14	16	20	24
cmti10	Text italic	5	7	9	10	11	12	14	16	20	24
cmsl10	Slanted	5	7	9	10	11	12	14	16	20	24
cmbx10	Roman bold	5	7	9	10	11	12	14	16	20	24
cmmi10	Math italic	5	7	9	10	11	12	14	16	20	24
cmsy10	Math symbol	5	7	9	10	11	12	14	16	20	24
cmex10	Math extended				10	11	12	14	16	20	24
cmtt10	Typewriter				10	11	12				
cmss10	Sans serif				10	11	12	14			
cmmib10	Math italic bold				10	11	12	14	16	20	24
cmbsy10	Bold symbol				10	11	12	14	16	20	24

C. Revisions

This appendix lists the major changes in TEXsis as new versions have come out. It is organized by sections of the manual and is intended as a guide for users familiar with the previous version. There are many smaller changes in addition to those listed here.

C.1. Revisions for Version 2.15

Section 1: Added a separate file called `Example.tex` which contains a short example physics paper which uses the TEXsis macros.

Section 3: Support added for `\scr` using Ralph Smith's Formal Script fonts, assuming that they are available.

Section 3: Added `\Footnote` to act like `\footnote` but with the text of the footnote singlespaced and in 10 pt type. (The font can be changed by changing `\FootFont`.)

Section 5: Modified `\reference`, `\cite`, etc., to insert extra space after a reference following the end of a sentence.

Section 6: Added `\pagecheck` to `\section` and `\subsection`, etc. These macros skip to to a new page if less than `\sectionminspace` space is left on the page. The initial setting for `\sectionminspace` is `0.25\vsize`. Also made the skips before the titles of sections and subsections variables `\sectionskip` and `\subsectionskip` in place of the previous fixed values.

Section 7: The definition of `\thesis` has been moved to the style file `thesis.txs` so that it is easier to make local changes. The `\thesis` command automatically loads from this file.

Section 7: Added the style file `PhysRev.txs` containing a definition of `\PhysRev` for a Physical Review format. A page proof layout should be available in the future.

Section 9: Added `epsf.tex` to the standard TEXsis distribution and included documentation for `\epsfbox` and related macros.

Section 9: The macros for making lists of figures and tables have been improved (along the lines of the new table of contents macros).

Section 10: Modified `\formletters` to have a complete `\letterhead` macro or even more complicated things as one of the arguments, rather than just the phone number. This allows you to create more complicated letterheads. Note also that the order of arguments is now reversed from what it was previously! The first argument is the `\letterhead` stuff (it gets done *before* the address) and the second argument is the name of the text file (it controls what is printed *after* the address.

Section 11: Introduced `\longequation` and `\endlongequation` for equations which span both columns. The definitions of these can vary from style to style, so they may or may not include a `\leftcolrule` and `\rightcolrule`.

Section 13: Added this section to the manual to explain miscellaneous useful macros which don't fit anywhere else.

Section 13: Added `\Ignore` and `\endIgnore` to ignore large blocks of text. This is slightly more clever than `\comment`, which remains available.

An index has been added to the manual, using the index making macros of `index.tex` and the MakeIndex program. We have included `index.tex` in the standard distribution of TEXsis, but the only documentation at present is that contained in the source file.

C.2. Revisions for Version 2.14

Section 3: `\TrueDoubleSpacing` generates a document with 2 times the normal spacing, in contrast to `\doublespaced`, which gives 1.5 times the normal spacing.

Section 4: Fixed a bug in `\bordermatrix`.

Section 5: An error which occurred if `\endreference` was followed by additional text on the same line has been fixed.

Section 5: Extra space is now inserted after a reference number which follows a period.

Section 5: `\refrange` can now be used with `\reference` or `\cite`; the correct numbers and ranges separated by commas are printed.

Section 8: The internal spacing mechanism for environments has been modified to work with arbitrary `\baselineskip` and `\parskip` values. Some minor changes have been made to the center and flush environments.

Section 8: The `\Listing` environment has been modified so that `{} % & () $ #` and —are just ordinary printable characters and so that `\\` gives a printed `\`.

Section 8: `\@startenv` and `\@endenv` to begin and end environments have been replaced with `\begEnv` and `\endEnv`, which are accessible to the user.

Section 9.4: The `\ruledtable` macros can now use `\tabskip` glue to produce a table of arbitrary width. Spacing of the tables has been improved and made more robust.

Section 6: `\addsectioncont` has been replaced with `\addTOC`, i.e., "add to table of contents." The new macro is simpler and produces better-looking output.

Section 6: Spacing such as `\singlespaced` placed after the `\texsis` initialization now sets the spacing for the entire document; the explicit spacing commands in `\abstract`, etc., have been removed.

Section 6: A macro `\setSectionID` has been added to allow one to place a figure or table for one section in a different section if necessary and still get the correct number.

Section 7: The `\NorthHolland` format now gives the more common single-column North-Holland/Elsevier format. `\NorthHollandTwo` produces the previous double-column oversized format. `\Manuscript` produces a preprint with true double spacing as requested by The Physical Review.

Section 7: `\endtitle`, `\endauthor`, etc., can now be omitted from the titlepage material, since `\title`, `\author`, etc., now close the previous mode automatically.

Section 10: Letters and memos can now be ended with `\bye`.

C.3. Revisions for Version 2.13

Section 3: Italic fonts are now available in large sizes, e.g. for `\slides`. Math italic bold fonts have been added for titles.

Section 4: By default parentheses and brackets are now ordinary characters except in math mode and so can be used, e.g., in file names. As a result, automatic sizing of parentheses works only in displayed equations. You can restore the old behavior by saying `\onparens` at the beginning of the document.

Section 5: The text for references is now written with `\unexpandedwrite`. In practice this means that references can contain almost anything, including a `\Ref`.

Section 5: The `\journal` macro was added to make it easier to create standard citations to journal articles, and to be able to switch between the American and European styles.

Section 5: Superscript references are now in `\citestyle`, which is initially set to `\scriptstyle` but can be changed.

Section 9.3: A way was created to save all figure and table insertions until the end of the document with `\FiguresLast` and `\PrintFigures` and similarly for tables.

Section 7: The `\NorthHolland` document layout was added to produce camera-ready copy for Proceedings published by North-Holland/Elsevier. This is a double column format with titles left justified and eleven point type.

Saying `\singlespaced` immediately after `\texsis` or its equivalent will now make the whole document single spaced.

Section 11: The double column macros have been modified so that removing the `\setdoublecolumns` will produce a single column document.

C.4. Revisions for Version 2.12

Section 2: A new command `\texsis` performs the initialization, and calling it more than once no longer causes an error. `\inittexsis` remains as a synonym.

Section 3: `\ninepoint`, `\twentypoint`, and `\twentyfourpoint` commands have been added to select these sizes. The 9 pt. size is really too small for a standard laser printer, but it is useful if you must print IEEE format on 8.5 in $\times$ 11 in paper. The large sizes are useful for slides and other special purposes.

Section 9.3: The table and figure insertion macros has been extensively modified, and the syntax has been changed. `topTable`, etc., have been replaced by `\table`, etc., which make the insertion and define the number for the tag. `\Tbl#1` and `\Fig#1` use the number. References to tables defined later in the paper are satisfied using the `.aux` file, so `\TOPTABLE#1`, etc., are no longer needed. The old names remain as synonyms but their use is discouraged.

Section 9.3: `\ruledtable` and many associated macros have been added to make complex ruled tables, including those with entries spanning more than one column. `\Tablebody` is still available, but the new macros are much more flexible and should be used instead.

Section 9.3: New macros have been added to list figures and tables.

Section 7: New formats have been added. `\WorldScientific` and `\IEEEreduced` make conference proceedings in World Scientific format and in IEEE format at 75% of full size. `\referee` is based on the memo format and can be used for referee reports. `\slides` uses large type to make transparencies for overhead projectors and similar material.

Section 7: An `\authors` macro for papers with multiple authors and facilities for putting footnotes on titles have been added.

Section 7: `\bookpagenumbers` causes odd page numbers to be placed in the upper right corner and even ones in the upper left corner. Additional information on headlines is given.

Section 10: The form letter macros have been rewritten. The syntax is the same except that `\v` has been replaced by `\V` to avoid conflict with the Plain TeX accent. Also, `\V` now also works with `\formlabels`.

Section 10: The new `\referee` format is described.

Section 11: The double column format has been extended to handle floating insertions. `\topinsert` and `\pageinsert` produce insertions within a column, while `\widetopinsert` and `\widepageinsert` produce insertions across both columns.

Section 12: The tagging mechanism has been changed to add an `@` before and after the name. The practical effect is that using the name of a defined control sequence like `book` for a tag no longer produces an error.

Acknowledgments

We wish to thank Isabel Harrity for her patience in trying this code while it was being developed. Various other people have also given us suggestions, criticisms, and comments regarding the code and documentation for TEXsis; among them we would like to thank:

Betty Armstrong	`bettya@csa.lbl.gov`
Leo Eskin	`lion@navier.stanford.edu`
Scott Hannahs	`hannahs@buphy.bu.edu`
Arik Kapulkin	`arik@lifshitz.ph.utexas.edu`
Michael Morrison	`mam@jilau1.Colorado.edu`
Anil Trivedi	`trivedi@yukawa.uchicago.edu`
Samir Varma	`varmint@ccwf.cc.utexas.edu`

Additional suggestions are welcome and will be incorporated in future versions.

This work was supported in part by the United States Department of Energy under contracts DE-AC02-76CH00016 and DE-AC02-89ER40509, and in part by the National Science Foundation under grant number PHY8919177.

References

1. D.E. Knuth, *The TEXbook* (Addison Wesley, 1986).

2. W. Groppe, *TechRpt* (unpublished).

3. E. Myers, *TEXsis Version 1* (unpublished).

4. M. Weinstein, SLAC-TN-84-007 (1984).

5. M. Doob, *A Gentle Introduction to TEX (A Manual for Self-Study)*, (TEX User's Group, Providence). Also available as the Plain TEX file `gentle.tex` via anonymous ftp from `lifshitz.ph.utexas.edu` and from various other sources on the net.

6. M. Spivak, *PCTEX User's Guide.* (Documentation distributed with PCTEX).

7. R.P. Feynman, Phys. Rev. 76, 749 (1949).

8. W. Pauli and F. Villars, Rev. Mod. Phys. 21, 434 (1949);

   J. Schwinger, Phys. Rev. 74, 1439 (1948).

9. E.D. Bloom, et al., Phys. Rev. Letters 23, 930 (1969).

10. *The Adventures of Buckaroo Banzai across the Eighth Dimension*, (Twentieth Century Fox, 1984).

11. S. Weinberg, *Gravitation and Cosmology: Principles and Applications of the General Theory of Relativity* (Wiley, New York, 1972) pg. 3.

12. Brian W. Kernighan and Dennis M. Ritchie, *The C Programming Language*, (Prentice Hall, Englewood Clifs, NJ, 1978).

13. Tomas Rokicki, `dvips` User's Manual (unpublished). The C source for the program and the TEX source for the manual are available from several of the TEX anonymous ftp sites.

14. D.E. Knuth, *The TEXbook* (Addison Wesley, 1986), page 245.

15. Ray F. Cowan, *Making Tables with Macros* (unpublished).

16. J.D. Bozek, Ph.D. thesis (unpublished), Department of Chemistry, University of Western Ontario, London, Ontario CANADA.

17. *Handbook of Chemistry and Physics*, 56th Edition, (CRC Press, Cleveland, Ohio, 1974), pg. B-9.

Index

(Boldface page numbers are the defining entries)

Contents

Appendices:

List of Tables

List of Figures

This page was unintentionally left blank.