How to Use the IEEEtran LATEX Class

Michael Shell, *Member, IEEE*

(Invited Paper)

***Abstract*— This article describes how to use the IEEEtran class with LATEX to produce high quality typeset papers that are suitable for submission to the Institute of Electrical and Electronics Engineers (IEEE). IEEEtran can produce conference, journal and technical note (correspondence) papers with a suitable choice of class options. This document was produced using IEEEtran in journal mode. Conference and correspondence papers typically use a subset of the commands discussed here.**

***Index Terms*— Class, IEEEtran, LATEX, paper, style, template, typesetting.**

I. INTRODUCTION

WITH a recent IEEEtran class file, a computer running LATEX, and a basic understanding of the LATEX language, an author can produce professional quality typeset research papers very quickly, inexpensively, and with minimal effort. The purpose of this article is to serve as a user guide of IEEEtran LATEX class and to document its unique features and behavior.

This document applies to version 1.6b and later of IEEEtran. Prior versions do not have all of the features described here. IEEEtran will display the version number on the user's console when a document using it is being compiled. The latest version of IEEEtran and its support files can be obtained from IEEE's web site [2], or CTAN [1]. This latter site may have some additional material, such as beta test versions and files related to non-IEEE uses of IEEEtran.

Complimentary to this document are the files[1] `bare_conf.tex` and `bare_jrnl.tex` which are "bare bones" example (template) files of a conference and a journal paper, respectively. Authors can quickly obtain a functional document by using these files as starters for their own work.

It is assumed that the reader has at least a basic working knowledge of LATEX. Those so lacking are strongly encouraged to read some of the excellent literature on the subject [3]. General support for LATEX related questions can be obtained in the internet newsgroup comp.text.tex. There is also a searchable list of frequently asked questions for this newsgroup [4].

Please note that the appendices sections contain information on installing the IEEEtran class file as well as tips on how to avoid commonly made mistakes.

II. CLASS OPTIONS

There are a number of class options that can be used to control the overall mode and behavior of IEEEtran. These are specified in the traditional LATEX way. For example,

```
\documentclass[9pt,technote]{IEEEtran}
```

is used with correspondence (technote) papers. The various categories of options will now be discussed. For each category, the default option is shown in bold. The user must specify an option from each category in which the default is not the one desired. The various categories are totally orthogonal to each other — changes in one will not affect the defaults in the others.

*A. 9pt, **10pt**, 11pt, 12pt*

There are four possible values for the normal text size. 10pt is used by the vast majority of papers. The two exceptions are technote papers, which use 9pt text, and the initial submissions to some conferences which use 11pt.

*B. draft, draftcls, draftclsnofoot, **final***

IEEEtran provides for three draft modes as well as the normal final mode. The draft modes provide a larger line spacing to allow for editing comments. The standard draft option puts *every* package used in the document into draft mode. With most graphics packages, this has the effect of disabling the rendering of figures. If this is not desired, one can use the draftcls option instead to yield a draft mode that will be confined within the IEEEtran class so that figures will be included as normal. draftclsnofoot is like draftcls, but does not display the word "DRAFT" along with the date at the foot of each page. When using one of the draft modes, most users will also want to select the onecolumn option.

*C. conference, **journal**, technote, peerreview, peerreviewca*

IEEEtran offers five major modes to encompass conference, journal, correspondence (technote) and peer review papers. Journal and technote modes will produce papers very similar to those that appear in many *IEEE Transactions* journals. When using technote, most users should also select the 9pt option. The peerreview mode is much like the journal mode, but produces a single-column cover page (with the title, author names and abstract) to facilitate anonymous peer review. The title is repeated (without the author names or abstract) on the first page after the cover page.[2] Papers using the peer review

Manuscript created February 25, 2002; revised November 18, 2002. This work was supported by the IEEE. The opinions expressed here are entirely that of the author. No warranty is expressed or implied. User assumes all risk.

M. Shell is with the Georgia Institute of Technology.
Email: mshell@ece.gatech.edu See [1] for current contact information.

---

[1] Note that it is the convention of this document not to hyphenate command, file or in-main-text URL names and to display them in `typewriter font`. Within such constructs, spaces are not implied at a line break and will be explicitly carried into the beginning of the next line. This behavior is not a feature of IEEEtran, but is used here to illustrate computer commands verbatim.

[2] A blank page may be inserted after the cover page when using the twoside (duplex printing) option so that the beginning of the paper does not appear on the back side of the cover page.

options require an `\IEEEpeerreviewmaketitle` command (in addition to and after the traditional `\maketitle`) to be executed at the place the cover page is to end — usually just after the abstract. This command will be silently ignored with the non-peerreview modes. See the bare template files for an example of the placement of this command. The peerreviewca mode is like peerreview, but allows the author name information to be entered and formatted as is done in conference mode (see Section III-B.2 for details) so that author affiliation and contact information is more visible to the editors.

1) Conference mode details: Conference mode makes a number of significant changes to the way IEEEtran behaves.

- The figure captions are centered.
- The `\author` text is placed within a tabular environment to allow for multicolumn formatting of author names and affiliations. Several commands are enabled to facilitate this formatting (see Section III-B.2 for details).
- The spacing after the authors' names is reduced. So is the spacing around the section names.
- The special paper notice (if used) will appear *between* the author names and the title (not after as with journals).
- The margins are increased as the height of the text is reduced to about 9.25in. In particular, the bottom margin will become larger than that of the top as IEEE wants extra clearance at the bottom. The text height will not be exactly 9.25in, but will vary slightly with the normal font size to ensure an integer number of lines in a column.
- Headings and page numbers are not displayed in the headers or footers. This, coupled with symmetric horizontal margins, will mean that there will not be a noticeable difference between one and two sided options.
- The following commands are intentionally disabled: `\thanks`, `\PARstart`, `\keywords`, `\biography`, `\biographynophoto`, `\pubid`, `\pubidadjcol`, `\IEEEmembership`, and `\IEEEaftertitletext`. If needed, they can be reenabled by issuing the command: `\IEEEoverridecommandlockouts`.
- Various reminder (related to camera ready work) and warning notices are enabled.

D. **letterpaper**, *a4paper*

IEEEtran supports both US letter (8.5in × 11in) and A4 (210mm × 297mm) paper sizes. Since IEEE uses US letter, authors should select the letterpaper option before submitting their work to IEEE. The main purpose of the a4paper option is to allow authors outside the US to print their work on A4 paper. Changing the paper size will *not* alter the typesetting of the document — only the margins will be affected. In particular, documents using the a4paper option will have reduced side margins (A4 is narrower than US letter) and a longer bottom margin (A4 is longer than US letter). For both cases, the top margins will be the same and the text will be horizontally centered.

Note that authors should ensure that all post-processing (ps, pdf, etc.) uses the same paper specification as the `.tex` document. Problems here are by far the number one reason for incorrect margins. See Appendix II for more details.

For those users who are doing their own binding, the command `\overrideIEEEmargins` can be issued in the document preamble to provide a wider margin on the binding edge. This command will have no effect in draft mode and should not be used for work that is to be submitted to the IEEE.

E. ***oneside***, *twoside*

These options control whether the layout follows that of single sided or two sided (duplex) printing. Because the side margins are normally centered, the main notable difference is in the format of the running headings.

F. *onecolumn*, ***twocolumn***

These options allow the user to select between one and two column text formatting. Since IEEE always uses two column text, the onecolumn option is of interest only with draft papers.

G. *nofonttune*

IEEEtran normally alters the default interword spacing to be like that used in IEEE publications. The result is text that requires less hyphenation and generally looks more pleasant, especially for two column text. The nofonttune option will disable the adjustment of these font parameters. This option should be of interest only to those who are using fonts specifically designed or modified for use with IEEE work.

III. The Title Page

The parts of the document unique to the title area are created using the standard LaTeX command `\maketitle`. Before this command is called, the author must declared all of the text objects which are to appear in the title area.

A. *Paper Title*

The paper title is declared like:

```
\title{A Heuristic Coconut-based Algorithm}
```

in the standard LaTeX manner. Line breaks (`\\`) may be used to equalize the length of the title lines.

B. *Author Names*

The name and associated information is declared with the `\author` command. `\author` behaves slightly differently depending on the document mode.

1) Names in Journal/Technote Mode: A typical `\author` command for a journal or technote paper looks something like this:

```
\author{Michael~Shell,~\IEEEmembership{Member,~IEEE,
} John~Doe,~\IEEEmembership{Fellow,~OSA,} and~Jane~D
oe,~\IEEEmembership{Life~Fellow,~IEEE}%
\thanks{Manuscript received January 20, 2002; revise
d January 30, 2002. This work was supported by the IE
EE.}%
\thanks{M. Shell is with the Georgia Institute of Te
chnology.}}
```

The `\IEEEmembership` command is used to produce the italic font that indicates the authors' IEEE membership status. The `\thanks` command produces the "first footnotes." Because the LaTeX `\thanks` was not designed to contain multiple paragraphs, one will have to use a separate `\thanks` for each paragraph. However, if needed, regular line breaks (`\\`) can be used within `\thanks`. In order to get proper line breaks and spacing, it is important to correctly use and control the spaces within `\author`. Use nonbreaking spaces (`~`) to ensure that name/membership pairs remain together. A minor, but easy, mistake to make is to forget to prevent unwanted spaces from getting between commands which use delimited (`{}`) arguments. Note the two `%` which serve to prevent the code line break on lines ending in a `}` from becoming an unwanted space. Such a space would not be ignored as an end-of-line space because, technically, the last `\thanks` is the final command on the line. "Phantom" spaces like these would append to the end of the last author's name, causing the otherwise centered name line to shift very slightly to the left.

2) Names in Conference Mode: The author name area is more complex when in conference mode because it also contains the authors' affiliations. For this reason, when in conference mode, the contents of `\author{}` are placed into a modified tabular environment. The commands `\authorblockN{}` and `\authorblockA{}` are also provided so that it is easy to correctly format the author names and affiliations, respectively. For papers with three or less affiliations, a multicolumn format is preferred:

```
\author{\authorblockN{Michael Shell}
\authorblockA{School of Electrical and\\
Computer Engineering\\
Georgia Institute of Technology\\
Atlanta, Georgia 30332--0250\\
Email: mshell@ece.gatech.edu}
\and
\authorblockN{Homer Simpson}
\authorblockA{Twentieth Century Fox\\
Springfield, USA\\
Email: homer@thesimpsons.com}
\and
\authorblockN{James Kirk\\
and Montgomery Scott}
\authorblockA{Starfleet Academy\\
San Francisco, California 96678-2391\\
Telephone: (800) 555--1212\\
Fax: (888) 555--1212}}
```

Use `\and` to separate the affiliation columns. The columns will automatically be centered with respect to each other and the side margins.

If there are more than three authors and/or the text is too wide to fit across the page, use an alternate format:

```
\author{\authorblockN{Michael Shell\authorrefmark{1}
, Homer Simpson\authorrefmark{2}, James Kirk\authorr
efmark{3}, Montgomery Scott\authorrefmark{3} and Eld
on Tyrell\authorrefmark{4}}
\authorblockA{\authorrefmark{1}School of Electrical a
nd Computer Engineering\\
Georgia Institute of Technology, Atlanta, Georgia 30
332--0250\\
Email: mshell@ece.gatech.edu}
\authorblockA{\authorrefmark{2}Twentieth Century Fox
, Springfield, USA\\
Email: homer@thesimpsons.com}
\authorblockA{\authorrefmark{3}Starfleet Academy, Sa
n Francisco, California 96678-2391\\
Telephone: (800) 555--1212, Fax: (888) 555--1212}
\authorblockA{\authorrefmark{4}Tyrell Inc., 123 Repl
icant Street, Los Angeles, California 90210--4321}}
```

The `\authorrefmark{}` command will generate a footnote symbol corresponding to the number in its argument. Use this to link the author names to their respective affiliations. It is not necessary prevent spaces from being between the `\authorblock`'s because each block starts a new group of lines and LaTeX will ignore spaces at the very end and beginning of lines.

C. Running Headings

The running headings are declared with the `\markboth{}{}` command. The first argument contains the journal name information and the second contains the author name and paper title. For example:

```
\markboth{Journal of Quantum Telecommunications,~Vol
.~1, No.~1,~January~2025}{Shell \MakeLowercase{\text
it{et al.}}: A Novel Tin Can Link}
```

Note that because the text in the running headings is automatically capitalized, the `\MakeLowercase{}` command must be used to obtain lower case text. The second argument is used as a page heading only for the odd number pages after the title page for two sided (duplex) journal papers. This page is such an example. Technote papers do not utilize the second argument. Conference papers do not have running headings, so `\markboth{}{}` has no effect when in conference mode. Authors should not put any name information in the headings (if used) of anonymous peer review papers.

D. Publication ID Marks

Publication ID marks can be placed on the title page of journal and technote papers via the `\pubid{}` command. The title page of this document has a publication ID that was made with:

```
\pubid{0000--0000/00\$00.00~\copyright~2002 IEEE}
```

Although authors do not yet have a valid publication ID at the time of paper submission, `\pubid{}` is useful because it provides a means to see how much of the title page text area will be unavailable in the final publication. This is especially important in technote papers because, in some journals, the publication ID space can consume more than one text line. If `\pubid{}` is used, a second command, `\pubidadjcol` must be issued somewhere in the *second* column of the title page. This is needed because LaTeX resets the text height at the beginning of each column. `\pubidadjcol` "pulls up" the text in the second column to prevent it from blindly running into the publication ID.

Publication IDs are not to be placed by the author on camera ready conference papers so `\pubid{}` is disabled in conference mode. Instead the bottom margin is automatically increased by IEEEtran when in conference mode to give IEEE room for such marks at the time of publication. In draft mode, the publisher ID mark will *not* be printed at the bottom of the titlepage, but room will be cleared for it.

E. Special Paper Notices

Special paper notices, such as for invited papers, can be declared with:

```
\specialpapernotice{(Invited Paper)}
```

Special paper notices in journal and technote papers appear between the author names and the main text. The title page of this document has an example. For conference papers, the special paper notice is placed between the title and the author names.

Much more rarely, there is sometimes a need to gain access to the space across both columns just above the main text. For instance, a paper may have a dedication [5]. IEEEtran provides the command `\IEEEaftertitletext{}` which can be used to insert text or to alter the spacing between the title area and the main text:

```
\IEEEaftertitletext{\vspace{-1\baselineskip}}
```

Authors should be aware that IEEEtran carefully calculates the spacing between the title area and main text to ensure that the main text height of the first page always is equal to an integer number of normal sized lines. Failure to do this can result in underfull vbox errors and paragraphs being "pulled apart" in the second column of the first page if there isn't any rubber lengths (such as those around section headings) in that column. The contents of `\IEEEaftertitletext{}` are intentionally allowed to bypass this "dynamically determined title spacing" mechanism, so authors may have to manually tweak the height (by a few points) of the `\IEEEaftertitletext{}` contents (if used) to avoid an underfull vbox warning.

IV. ABSTRACT AND INDEX TERMS

The abstract is generally the first part of a paper after `\maketitle`. The abstract text is placed within the abstract environment:

```
\begin{abstract}
We propose ...
\end{abstract}
```

Journal and technote papers also have a list of key words (index terms) which can be declared with:

```
\begin{keywords}
Broad band networks, quality of service, WDM.
\end{keywords}
```

V. SECTIONS

Sections and their headings are declared in the usual LaTeX fashion via `\section{}`, `\subsection{}`, `\subsubsection{}`, and `\paragraph{}`. The numbering for these sections is in upper case Roman numerals, upper case letters, Arabic numerals and lower case letters, respectively. The `\paragraph{}` section is not allowed for technotes as they generally are not permitted to have such a deep section nesting depth. If needed, `\paragraph{}` can be restored by issuing the command `\setcounter{secnumdepth}{4}` in the document preamble.

A. Initial Drop Cap Letter

The first letter of a journal paper is a large, capital, oversized letter which descends one line below the baseline. Such a letter is called a "drop cap" letter. The other letters in the first word are rendered in upper case. This effect can be accurately produced using the IEEEtran command `\PARstart{}{}`. The first argument is the first letter of the first word, the second argument contains the remaining letters of the first word. The drop cap of this document was produced with:

```
\PARstart{W}{ith}
```

Note that some journals will also render the second word in upper case — especially if the first word is very short. For more usage examples, see the `bare_jrnl.tex` template file.

VI. CITATIONS

Citations made with the `\cite{}` command as usual. IEEEtran will produce citation numbers that are individually bracketed in IEEE style. ("[1], [5]" as opposed to the more common "[1, 5]" form.) The base IEEEtran does not sort or produce "ranges" when there are three or more consecutive citation numbers. However, IEEEtran pre-defines some format control macros to facilitate easy use with the LaTeX cite.sty package [6]. So, all an author has to do is to call cite.sty:

```
\usepackage{cite}
```

and the citation numbers will automatically be sorted and ranged IEEE style.

Note that, if needed, the cite.sty's `\cite{}` command will automatically add a leading space. i.e., "(`\cite{mshell01}`)" will become like "([1])." If this behavior is not desired, use the cite package's noadjust option (cite.sty V3.8 and later) which will turn off the added spaces:

```
\usepackage[noadjust]{cite}
```

`\cite{}` also allows for an optional note. e.g., `\cite[Th. 7.1]{mshell01}`. If the `\cite{}` with note has more than one reference, the note will be applied to the last of the listed references. It is generally desirable that if a note is given, only one reference should be listed in that `\cite{}`.

VII. EQUATIONS

Equations are created using the traditional `equation` environment:

```
\begin{equation}
\label{eqn_example}
x = \sum\limits_{i=0}^{z} 2^{i}Q
\end{equation}
```

which yields

$$x = \sum_{i=0}^{z} 2^{i} Q. \tag{1}$$

Use the `displaymath` environment instead if no equation number is desired. When referring to equations, articles in IEEE publications do not typically use the word "equation," but rather just enclose the equation number in parentheses, e.g.,

```
... as can be seen in (\ref{eqn_example}).
```

IEEE's two column format puts serious constraints on how wide an equation can be. So, a fair portion of the effort in formatting equations usually has to be devoted to properly breaking them. It is the author's responsibility to ensure that all equations fit into the given column width. In rare circumstances, it is possible to have a few equations that span both columns (see Section IX-C.1), but the vast majority of over-length equations have to be broken across multiple lines.

VIII. MULTI-LINE EQUATIONS

Perhaps the most convenient and popular way to produce multiline equations is LaTeX 2_ε's `eqnarray` environment. However, `eqnarray` has several serious shortcomings:

1) the use of $2\times$`\arraycolsep` for a column separation space does not provide natural math spacing in the default configuration;
2) column definitions cannot be altered;
3) is limited to three alignment columns;
4) column alignment cannot be overridden within individual cells.

There are a number of vastly superior packages for formatting multiline mathematics. Perhaps the most popular is the amsmath package [7]. Amsmath is a comprehensive work which contains many helpful tools besides enhanced multiline alignment environments. So, all authors should give serious consideration to its use — regardless of what they use to generate aligned equations. One thing to be aware of is that, upon loading, amsmath will configure LaTeX to disallow page breaks within multiline equations (even within non-amsmath defined environments). The philosophy here is that author should manually insert breaks where desired so as to ensure that breaks occur only at acceptable points. To restore IEEEtran's ability to automatically break within multiline equations, load amsmath like:

```
\usepackage{amsmath}
\interdisplaylinepenalty=2500
```

Another extremely powerful set of alignment tools, one of which is a totally rewritten `eqnarray` environment, is provided by mathenv.sty which is part of Mark Wooding's MDW Tools [8].

Finally, IEEEtran provides a fully integrated custom IEEEeqnarray family of commands (see Appendix VI) that are designed to have almost universal applicability for many different types of alignment situations.

Nevertheless, it is instructive to show a simple example using the standard `eqnarray` in order to explain some of the fine points of math spacing under LaTeX. As shown in Table I, TeX normally draws from four different spacings when typesetting mathematics. In order to produce precision (and correct) mathematical alignments, it is crucial to understand how to control such spacing. Consider a multiline equation

$$\begin{aligned} Z ={}& x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \\ &+a + b && (1)\\ &+a + b && (2)\\ &{}+a + b && (3)\\ &+\:a + b && (4)\end{aligned}$$

(in typical IEEE style) which was produced by

```
\setlength{\arraycolsep}{0.0em}
\begin{eqnarray}
Z&{}={}&x_1 + x_2 + x_3 + x_4 + x_5 + x_6\nonumber\\
&&+a + b\\
&&+{}a + b\\
&&{}+a + b\\
&&{+}\:a + b
\end{eqnarray}
\setlength{\arraycolsep}{5pt}
```

Lines one through four show some possible ways the $+a+b$ line could be implemented.[3] Only number four is the correct way for most IEEE purposes. In TeX's math mode, spacing around operators can be inhibited by enclosing them within braces (e.g., `{=}`) or forced by surrounding them with "empty ords" (e.g., `{}={}`). It is important to understand that the empty ords do not have width themselves. However, their presence causes TeX to place space around the operators as if they were "next to something." With this in mind, the first step in the example is to set `\arraycolsep` to zero to prevent `eqnarray` from putting in the unwanted, artificial, inter-column spacing. Placing empty ords around the equal sign then forces the correct natural spacing. Alternatively, `\arraycolsep` could have been set to 0.14 em and the empty ords around the equal sign eliminated.[4] It is important to remember to restore `\arraycolsep` to its default value of 5 pt after the `eqnarray` is complete as other environments (such as `array`) depend on it.

TABLE I

MATH SPACINGS USED BY LaTeX

Size	Width	Cmd.	Used for	Example
small	1/6 em	`\,`	symbols	$a\,b$
medium	2/9 em	`\:`	binary operators	$a+b$
large	5/18 em	`\;`	relational operators	$a=b$
negative small	−1/6 em	`\!`	misc. uses	$a\!b$

The first line is incorrect because a is being indicated as a positive quantity rather than something that must be added to the previous line. (i.e., the $+$ is being treated as a unary, rather than a binary, operator.) In line two, adding an empty ord to the right side of the plus sign does nothing, except to demonstrate that empty ords have zero width. Adding an empty ord to the left side of the plus sign (line three) does engage binary spacing, but causes an unwanted[5] right shift of the line. Finally, manually adding a medium space to the right side only of the plus sign in line four does the trick. The suppression of automatic spacing around the plus sign (`{+}`) is unneeded in this case, but may be required in other alignment environments that "expand" such operators by default.

Another way around the spacing problem is to use only two alignment columns (as is done by amsmath.sty's `\align`). e.g., in the previous example, "$Z =$" would be contained

---

[3] In this absurd example, the equation numbering system is used to identify lines.

[4] This assumes that 1 em in the text font has the same width as 1 em in the math font. For the standard fonts, this is indeed the case.

[5] IEEE normally wants all of the lines left aligned, but there are cases when such an indention may be desirable.

in the first column.

A. Cases Structures

Incidentally, the `numcases` (or `subnumcases`) environments in Donald Arseneau's cases.sty package [9] should be used when "cases" structures in which each branch can be referenced with a different equation (or subequation) number are needed:

$$|x| = \begin{cases} x, & \text{for } x \geq 0 \qquad (2a) \\ -x, & \text{for } x < 0 \qquad (2b) \end{cases}$$

because those built from the `array` or amsmath `cases` environments will have a single equation number that encompasses both branches.

IX. FLOATING STRUCTURES

A. Figures

Figures handled in the standard LaTeX manner. For example:

```
\begin{figure}
\centering
\includegraphics[width=2.5in]{myfigure}
\caption{Simulation Results}
\label{fig_sim}
\end{figure}
```

Note that (1) figures should be centered via the LaTeX `\centering` command. This is a better approach than using the `center` environment which adds unwanted vertical spacing; (2) the caption follows the graphic; and (3) any labels must be declared *after* (or within) the caption command.

When referencing figure numbers in the main text (via `\ref{}`), IEEE uses the abbreviation "Fig." rather than "Figure".

The `\includegraphics` command is the modern, preferred, way of including images and provides a flexible interface that makes it easy to scale graphics to size. To use it, the graphics or graphicx (recommended) must first be loaded.

It is strongly recommended that authors be familiar with the graphics package documentation [10] as well as Keith Reckdahl's excellent *Using Imported Graphics in LaTeX 2$_{\varepsilon}$* [11]. The reader is reminded that the "draftcls" or "draftclsnofoot", not "draft", class option must be selected in order to get draft papers with visible figures.

As explained in Appendix IV, Encapsulated PostScript (EPS) is the preferred graphics format for LaTeX work. Furthermore, the user's drawing/graphing application should be capable of outputing directly in EPS vector form (which will not degrade or pixelize when magnified) — although photos will likely have to be in (EPS) bitmap form.

The psfrag package [12] might also be of interest. Psfrag allows the user to "go into" an EPS graphic and replace text strings contained in it with real LaTeX code! In this manner, LaTeX's extensive support of mathematical symbols and fonts can be extended to figures made with applications with more modest glyph support. Using psfrag does require the use of the dvips DVI to PostScript conversion step (not pdfLaTeX[6]) as some of the features of the PostScript language have to be utilized. There is additional usage information on psfrag in the *Using Imported Graphics in LaTeX 2$_{\varepsilon}$* guide [11].

TABLE II
A SIMPLE EXAMPLE TABLE

First	**Next**
1.0	2.0

B. Tables

Tables are handled in a similar fashion, but with a few notable differences. For example, the code

```
\begin{table}
\renewcommand{\arraystretch}{1.3}
\caption{A Simple Example Table}
\label{table_example}
\centering
\begin{tabular}{c||c}
\hline
\bfseries First & \bfseries Next\\
\hline\hline
1.0 & 2.0\\
\hline
\end{tabular}
\end{table}
```

results in Table II. Note that IEEE places table captions *before* the tables. Within the table environment, the default text size is footnotesize which is what IEEE typically uses for tables. When using the tabular environment to construct tables, it is usually a good idea to increase the value of `\arraystretch` above unity to "open up" the table rows a tad. Also, IEEE often uses tables with "open sides," (without vertical lines along each side) although the "closed side" form (e.g., Table I) is more commonly used for the tables within this document.

Unfortunately, the standard LaTeX 2$_{\varepsilon}$ tabular environment has a number of shortcomings. Two notable problems are (1) the corners where lines meet are improperly formed; and (2) it is not very flexible in terms of user control. For these reasons, authors are urged to look into some of the other packages for making tables. A good one that provides revised "drop-in replacements" for both the tabular and array environments is Frank Mittelbach's and David Carlisle's array package [13]. Even more powerful (and complex) is the tabular and array environments provided by the mdwtab.sty package which is part of Mark Wooding's MDW Tools [8].

As an alternative, IEEEtran offers the IEEEeqnarraybox command which can also be used to produce tables[7] of high quality. See Appendix VI for more details.

1) Footnotes Within Tables: Footnotes normally cannot be placed directly within some commands and environments such as `\parbox`, and `tabular`, etc., because they become "trapped" inside. One way around this is to split the place the footnote marker (`\footnotemark`) is located (within the table) from where the footnote text itself is declared (outside of the table using `\footnotetext`).

---

[6] pdfLaTeX users currently have to "preprocess" their figures by importing them into a dummy document using psfrag, running LaTeX followed by dvips, then converting the PostScript output to a PDF graphic for importation into the main document which is then processed by pdfLaTeX.

[7] Table I was made using this command.

TABLE III

THE SKEWING ANGLES (β) FOR $\mathrm{Mu(H)} + \mathrm{X}_2$ AND $\mathrm{Mu(H)} + \mathrm{HX}$ [a]

	$\mathrm{H(Mu)} + \mathrm{F}_2$	$\mathrm{H(Mu)} + \mathrm{Cl}_2$
β(H)	80.9° [b]	83.2°
β(Mu)	86.7°	87.7°

[a] for the abstraction reaction, $\mathrm{Mu} + \mathrm{HX} \rightarrow \mathrm{MuH} + \mathrm{X}$.

[b] 1 degree $= \pi/180$ radians.

Another approach is to use the footnote.sty package (which is part of Mark Wooding's MDW Tools [8]) which allows environments to be configured so as not to trap footnotes:

```
\usepackage{footnote}
\makesavenoteenv{tabular}
```

Note that is probably not a good idea to use footnotes in floating structures (like `table`) because the position of each can move relative to one another. To put the footnote at the end of a table instead of at the bottom of the page, just enclose `tabular`, etc., inside a minipage (no footnote package needed). A very good approach for handling footnotes within tables (including those that float) is to use Donald Arseneau's threeparttable package [14] which was used to generate Table III (the code of which is an example in the threeparttable.sty file).

C. Double Column Floats

LaTeX's `figure*` and `table*` environments produce figures and tables that span both columns. This capability can be used with the Steven Douglas Cochran's subfigure package [15] to format a set of related figures:

```
\begin{figure*}
\centerline{\subfigure[Case I]{\includegraphics[widt
h=2.5in]{subfigcase1}
\label{fig_first_case}}
\hfil
\subfigure[Case II]{\includegraphics[width=2.5in]{su
bfigcase2}
\label{fig_second_case}}}
\caption{Simulation results}
\label{fig_sim}
\end{figure*}
```

Note how labels can be tagged to each of the subfigures as well as to the overall figure. `\hfil` is used as a subfigure separator to achieve equal spacing around the graphics. More complex implementations are possible. See the subfigure documentation as well as the *Using Imported Graphics in LaTeX2ε* guide [11] for more details.

It is a limitation of the LaTeX2ε kernel that double column floats cannot be placed at the bottom of pages. That is to say "`\begin{figure*}[!b]`" will not normally work as intended. Authors that need this capability should obtain and load Sigitas Tolušis' stfloats package [16] which patches the LaTeX2ε output routine to allow it to handle double column floats at the bottom of pages. Please note that stfloats is a very invasive package which may not work with versions of LaTeX other than the standard LaTeX2ε release and may cause problems with other packages that modify the output routines (such as those that balance columns, alter the placement of floating figures, etc.). IEEE authors are warned *not* to use packages that allow material to be placed across the middle of the two text columns (such as cuted.sty, midfloat.sty, etc.) as IEEE does not do this.

Another LaTeX2ε limitation (patched with stfloats or not) is that double column floats will not appear on the same page where they are defined. So, the user will have to define such things prior to the page on which they are to (possibly) appear.

LaTeX2ε (patched with stfloats or not) does not attempt to keep double and single column floats in sequence with each other. This can be fixed by loading David Carlisle's fix2col package (already installed on most LaTeX systems) [17]. However, fix2col should not be used with the stfloats package as they both modify some of the same output routines in different ways.

Finally, authors should also be aware that the LaTeX2ε kernel (patched with stfloats or not) has a long standing limitation in that it will not allow rubber space that spans both columns to stretch or shrink as needed for each of the two main text columns. Therefore, it is possible for double column floats to cause underfull vbox errors because the remaining text height may not be equal to an integer number of normal size lines. The problem can occur in main text columns (on pages with double column floats) that do not have vertical rubber spacing (such as that around section headings, equations, etc.) and results in underfull vbox warnings coupled with paragraphs that are "pulled apart" from each other. To correct this, users can manually tweak the amount of space between the double column structure and main text by inserting a command like

```
\vspace*{-3pt}
```

(adjusted as needed) within the double column structure. Incidentally, IEEEtran automatically compensates for this problem when forming the paper title.

1) Double Column Equations: It is possible, but not pleasant, to use `figure*` to obtain double column equations. IEEE rarely uses double column equations because they can waste space, so this capability is easy to abuse. Authors who are considering the use of a double column equation should verify that there are a few examples of such in papers previously published in the journal they plan to submit to.

There are complications. Although IEEE does not place constraints on the order of the double column equations relative to the equations of the main text (that is to say a set of double column equations can be at the top or bottom of a page in which they would normally appear in the middle had they been regular equations), the double column equation numbers must increase as one progresses down the page (i.e., double column equations at the bottom of a page must be of higher number than those at the top). Furthermore, double column equations should appear on the same page where they are referenced (on the page they would have appeared had they been regular equations). Compounding the difficulty even further is the fact that LaTeX2ε will not place double column equations on the same page on which they are defined. Finally, IEEE does not generally allow other figures or tables to come between the double column equations and the main text (which are separated from each other by a rule). All of this means that

$$x = 5 + 7 + 9 + 11 + 13 + 15 + 17 + 19 + 21 + 23 + 25 + 27 + 29 + 31 \tag{6}$$

$$y = 4 + 6 + 8 + 10 + 12 + 14 + 16 + 18 + 20 + 22 + 24 + 26 + 28 + 30 \tag{7}$$

the place where a double column equation must be defined has to be "disconnected" from the place where it will eventually be referred to in the text — and the user will have to manually intervene in the equation numbering system.

Therefore, users have to (1) define double column equations on the page *prior* to the one that they are to appear; (2) reset the equation counter when the double column equations are defined so as not to disturb the regular equation numbers; (3) manually set the double column equation numbers and (4) increment the equation counter at the point the double column equations are referenced in the text so that they are accounted for in the numbering of the regular equations after that point.

To do all of this, it is convenient to have a "scratch pad" counter to temporarily save equation numbers. This can be done via a command such as

```
\newcounter{mytempeqncnt}
```

in the preamble of the document. Now, the double column equations are defined on the page *prior* to the one in which they are to appear (and in this example supposed that they are to be equation numbers six and seven):

```
\begin{figure*}[!t]
% ensure that we have normalsize text
\normalsize
% Store the current equation number.
\setcounter{mytempeqncnt}{\value{equation}}
% Set the equation number to one less than the one
% desired for the first equation here.
% The value here will have to changed if equations
% are added or removed prior to the place these
% equations are referenced in the main text.
\setcounter{equation}{5}
\begin{equation}
\label{eqn_dbl_x}
x = 5 + 7 + 9 + 11 + 13 + 15 + 17 + 19 + 21+ 23 + 25
 + 27 + 29 + 31
\end{equation}
\begin{equation}
\label{eqn_dbl_y}
y = 4 + 6 + 8 + 10 + 12 + 14 + 16 + 18 + 20+ 22 + 24
 + 26 + 28 + 30
\end{equation}
% Restore the current equation number.
\setcounter{equation}{\value{mytempeqncnt}}
% IEEE uses as a separator
\hrulefill
% The spacer can be tweaked to stop underfull vboxes.
\vspace*{4pt}
\end{figure*}
```

The result of which is shown at the top of this page. This technique allows the definition of the equations to be positioned arbitrarily as needed so that the (floating) equations will appear where desired. The "[!t]" option forces LATEX to do its best to place the equations at the top of the next page. Had it been "[!b]" instead, then the stfloats package would need to be loaded and the `\vspace` command, followed by the `\hrulefill` command, would have to occur *before* the equations in the figure.

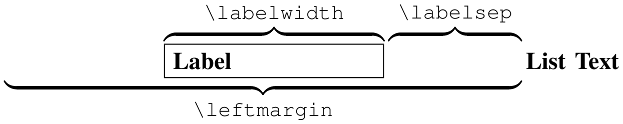
The double column equations can then been referenced in the main text like:

```
% The previous equation was number five.
% Account for the double column equations here.
\addtocounter{equation}{2}
As can be seen in (\ref{eqn_dbl_x}) and
(\ref{eqn_dbl_y}) at the top of the page ...
```

Thankfully, double column equations are rare.

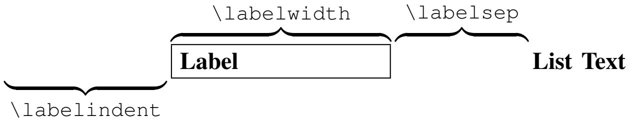
X. LISTS

The traditional LATEX itemize, enumerate and description (IED) list environments are ill-suited for producing the style of lists used in IEEE publications. The main problem is that they do not provide the user a means for controlling the parameters of the resultant list. Furthermore, making global changes to the parameters of the underlying `\list` will result (often unexpectedly to a user) in the improper behavior of other commands that depend on it, such as `\quote`. Finally, LATEX's `\list` considers the left margin of the list text to be the reference point that determines how the list is positioned relative to the left margin of the main text:



This contrasts with IEEE lists which use the label box as the reference point for the list structure. i.e., for a given circumstance, the list labels will be indented by a certain amount, the list text block will be indented from the label boxes by a given amount and these spacings will determine the position of the list text.

For these reasons, IEEEtran provides enhanced IED list environments that make it much easier to produce IEEE style lists. The underlying `\list` remains the same as in traditional LATEX so as not to break code that depends upon it. IEEEtran uses a new length variable, `\labelindent`, so that users can specify IED list structures directly in IEEE fashion:



The IEEEtran IED lists ignore all "external" changes to the list length parameters. Instead, IED lists are controlled exclusively via two interfaces:

1) "global" control via the `\iedlistdecl` command; and
2) "local" control via an *optional* argument that can be provided to `\itemize`, `\enumerate`, and `\description`.

For example, declaring

```
\renewcommand{\iedlistdecl}{\settowidth{\labelwidth}
{Hello}}
```

in an IEEEtran document will set the default width of the label boxes in *all* later IED lists to be equal to the width of "Hello". Note: Because setting a `\labelwidth` is so commonly performed, IEEEtran provides a command: `\setlabelwidth{X}` which is a shorter form of: `\settowidth{\labelwidth}{X}`.

The local control is used if the parameters are to apply only to an individual IED list:

```
\begin{itemize}[\setlabelwidth{$\gamma$}]
```

Within an IED list, the local control is executed just after the global control and therefore, the commands in the local control can both augment and countermand those in the global control. Please note that the code in the local and global controls are executed in the same manner as normal LaTeX code. Therefore, the user should ensure that unwanted blank spaces do not appear in the controls. If a control definition is too long to fit on one line, shield the end of lines with "%" to prevent them from being interpreted as blanks. (Section III-B.1 has some information on this topic.) Also, note that the LaTeX parser requires that braces be placed around commands with optional arguments that are placed directly within the optional arguments of other commands:

```
\begin{itemize}[{\mycmd[1]{example}}]
```

This IEEEtran IED implementation makes it easy to control IED lists, even when they are deeply nested.

The default spacings the IED lists use are stored in lengths which begin with "IEEE", i.e., `\IEEEilabelindent`. Changes to these "master" defaults are rarely needed and should be done only at the beginning of the document, *not in the IED list controls*. These constants will now be briefly explained.

`\IEEEilabelindent`: This length is the default amount the itemized list label boxes are indented from the left margin. IEEE seems to use at least two different values. For example, in *The Journal of Lightwave Technology* and *The Journal on Selected Areas in Communications*, they tend to use an indention equal to `\parindent`, while for *Transactions on Communications* they tend to indent itemized lists a little more (`1.3\parindent`). The shorter length is stored as `\IEEEilabelindentA` and the longer as `\IEEEilabelindentB`. The default is to use the shorter version. To use the longer version do a

```
\setlength{\IEEEilabelindent}{\IEEEilabelindentB}
```

at the beginning of the document.

`\IEEEelabelindent`: This length is the default amount the enumerated list label boxes are indented from the left margin. Normally, the same as `\parindent`.

`\IEEEdlabelindent`: Ditto for description list labels. Normally, the same as `\parindent`.

`\IEEEiednormlabelsep`: This length is the normal default spacing between the IED list label boxes and the list text.

`\IEEEiedmathlabelsep`: For nomenclature description lists (a list of math symbols and their explanations), IEEE usually increases the separation between the terms and the definitions. This length is set to the longer than normal length. To invoke its use, just issue the command `\usemathlabelsep` in a list control.

`\IEEEiedtopsep`: This length is the extra vertical separation put above and below each IED list. IEEE usually puts a little extra spacing around each list. However, this extra spacing is barely noticeable.

`\IEEElabelindentfactori` through **`\IEEElabelindentfactorvi`**: These contain the factors by which the effective `\labelindent` is reduced as the list nesting depth increases. IEEE normally decreases the amount of indention as the list nesting level increases because there isn't much room to indent with two column text. IEEEtran has an "automatic indention cut-back" feature that provides this behavior. The actual amount the label boxes will be indented is `\labelindent` multiplied by the `\IEEElabelindentfactorX` corresponding to the level of nesting depth (where "X" is the nesting depth in roman numerals). This provides a means by which the user can alter the effective `\labelindent` for deeper levels. There may not be such a thing as correct "standard IEEE" values. What IEEE actually does may depend on the specific circumstances. The first list level almost always has full indention. The second levels usually have only 75% of the normal indentation. Third level and greater nestings are very rare, and probably don't use any indentation. These factors are not lengths, but rather constant macros like `\baselinestretch` so `\renewcommand` should be used if they need to be changed. The default values are

```
\IEEElabelindentfactori   1.0
\IEEElabelindentfactorii  0.75
\IEEElabelindentfactoriii 0.0
\IEEElabelindentfactoriv  0.0
\IEEElabelindentfactorv   0.0
\IEEElabelindentfactorvi  0.0
```

The use of these factors in IED lists may be suspended by issuing the command `\nolabelindentfactortrue` in a list control (which has the same effect as setting all the indent factors to 1.0).

Normally, IEEEtran automatically calculates `\leftmargin` based upon the current values of `\labelindent`, `\labelwidth` and `\labelsep`. To stop this auto-calculation so that a manually specified value of `\leftmargin` is used instead, just use `\nocalcleftmargintrue` in a list control. This feature should not be needed during the course of normal IEEE related work.

IEEEtran provides a means to manually specify the justification within the IED list label boxes. The commands `\iedlabeljustifyl`, `\iedlabeljustifyc` and `\iedlabeljustifyr` can be used in a list control to justify the list labels to the left, center, and right sides, respectively. Itemize and enumerate lists automatically default to right justification, while description defaults to left justification. The justification commands should not be needed during the course of normal IEEE related work.

In addition to modifying the behavior of `itemize`, `enumerate` and `description`, IEEEtran also provides the respective aliases `IEEEitemize`, `IEEEenumerate` and `IEEEdescription`, which provides a way for the user to access the IEEE style list environments even in the event another package is loaded that overrides the IED list environments. For specialized applications, the original LaTeX IED list environments are retained as `LaTeXitemize`, `LaTeXenumerate` and `LaTeXdescription`.

A. Itemize

The itemized lists will normally automatically calculate the width of whatever symbol the current list level is using so that a user can just call `\begin{itemize}...\end{itemize}` without doing anything special. Furthermore, the auto-label-width feature will work properly even if `\labelitemX` has been redefined (where "X" indicates "i,ii, .. iv", whichever is appropriate) *before* before the list begins. However, if any item symbols are to be specified via `\item[X]` (this is rare and may well be nonstandard as far as IEEE related work is concerned), then the following form can be used:

```
\begin{itemize}[\setlabelwidth{Z}]
\item[X] blah
\item[Y] blah
.
.
\end{itemize}
```

where "Z" is the longest label in the list.

B. Enumerate

The important thing to note about enumerated lists is that the `\labelwidth` will default to the length of "9)" in the normal size and style. Therefore, the width of the longest label will have to be manually specified if *any* of the following conditions are true:

1) a top level list has more than 9 items;
2) a relevant `\labelenumX` or `\theenumX` has been redefined;
3) `\item[X]` has been used to manually specify labels;
4) the labels are using a font that is not the normal size and style;
5) the enumerated list is nested (i.e., not at the top level) and is therefore not using Arabic digits as labels.

For example:

```
\begin{enumerate}[\setlabelwidth{12)}]
\item blah
\item blah
.
.
% 12 items total
\end{enumerate}
```

C. Description

Generally speaking, the longest label width will always have to be specified for description lists. Furthermore, the author may wish to use `\IEEEmathlabelsep` for `\labelsep` when building a math symbol list. For example:

```
\begin{description}[\setlabelwidth{$\alpha\omega\pi
\theta\mu$}\usemathlabelsep]
\item[$\gamma\delta\beta$] Is the index of..
\item[$\alpha\omega\pi\theta\mu$] Gives the..
.
.
\end{description}
```

Sometimes it can be difficult to ascertain from inspection which of the labels is the longest. For such cases, a little diagnostic code may be helpful to measure a length and then to display the result on the console:

```
\newlength{\mydiaglen} % put in preamble
.
.
\settowidth{\mydiaglen}{$\alpha\beta\gamma$}
\showthe\mydiaglen
```

XI. THEOREMS AND PROOFS

Theorems and related structures such as axioms, corollaries and lemmas, are handled in the traditional LaTeX fashion. The user must first declare the structure name via the

```
\newtheorem{struct_type}{struct_title}[in_counter]
```

command where *`struct_type`* is the user chosen identifier for the structure, *`struct_title`* is the heading that is used for the structure and *`in_counter`* is an optional name of a counter whose number will be displayed with the structure number and whose update will reset the structure counter. Most IEEE papers use sequential theorem numbering throughout the entire work, so an *`in_counter`* is usually not specified. However, those papers that do use *`in_counter`* usually use "`section`" such that the section number is the first part of each theorem number. After the structure is defined it can be used via

```
\begin{struct_type}[extra_title]
.
.
\end{struct_type}
```

where *`extra_title`* is an optional name that is displayed with the structure.

For example, the most common way to do theorems would be to use

```
\newtheorem{theorem}{Theorem}
```

followed as needed by environments like

```
\begin{theorem}[Einstein-Podolsky-Rosenberg]
```

Sometimes it is desirable that a structure share its counter with another structure. This can be accomplished by using the alternate form of `\newtheorem`

```
\newtheorem{struct_type}[num_like]{struct_title}
```

where *`num_like`* is the name of an existing structure.

A. Proofs

Proofs are easily handled by the predefined proof environment:

```
\begin{proof}
.
.
\end{proof}
```

The Q.E.D. symbol "■" is automatically placed at the end of each proof. If needed, the symbol can be manually accessed via the `\QED` command. Both the closed (default) "■" and open "□" forms are provided as `\QEDclosed` and `\QEDopen`, respectively. To change the default from closed to open (some journals and/or authors prefer the open form), just redefine `\QED` as desired:

```
\renewcommand{\QED}{\QEDopen}
```

XII. END SECTIONS

A. Appendices

The `\appendix` command is used to start a single appendix. An optional argument can be used to specify a title:

```
\appendix[Proof of the Zonklar Equations]
```

After issuing `\appendix`, the `\section` command will be disabled and any attempt to use `\section` will be ignored and will cause a warning message to be generated. (The single appendix marks the end of the enumerated sections and the section counter is fixed at zero — one does not state "see Appendix I" when there is only one appendix, instead "see the Appendix" is used.) However, all lower `\subsection` commands and the `\section*` form will work as normal as these may still be needed for things like acknowledgments.

`\appendices` is used when there is more than one appendix section. `\section` is then used to declare each appendix:

```
\section{Proof of the First Zonklar Equation}
```

The mandatory argument to section can be left blank (`\section{}`) if no title is desired. It is important to remember to declare a section before any additional subsections or labels that refer to section (or subsection, etc.) numbers. As with `\appendix`, the `\section*` command and the lower `\subsection` commands will still work as usual.

There are two appendix numbering conventions used by IEEE. Capital letters (e.g., "Appendix A") and Roman numerals (e.g., "Appendix I"). The former appears to be more popular. However, since subsection numbering conventions are preserved in the appendices, this creates rather awkward subsection references like "A-A". For this reason, IEEEtran defaults to using Roman numerals when labeling appendices. Appendix numbering can be changed to capital letters by issuing the command

```
\useRomanappendicesfalse
```

before `\appendices` is called.

Some authors prefer to have the appendix number to be part of equation numbers for equations that appear in an appendix. This can be accomplished by redefining the equation numbers as

```
\renewcommand{\theequation}{\thesection.\arabic{equa
tion}}
```

before the first appendix equation. For a single appendix, the constant "A" should be used in place of `\thesection`.

B. Acknowledgements

Acknowledgements and other unnumbered sections are created using the `\section*` command:

```
\section*{Acknowledgment}
\addcontentsline{toc}{section}{Acknowledgment}
```

The second, optional, command is needed to manually add such sections to the table of contents (which is rarely used, but some authors may do so with draft papers).

C. Bibliographies

Bibliographies are most easily (and correctly) generated using the IEEEtran BIBTEX package [18] which is easily invoked via

```
\bibliographystyle{IEEEtran}
\bibliography{IEEEabrv,mybibfile}
```

See the IEEEtran BIBTEX package documentation for more information.

When submitting the document source (.tex) file to external parties (such as IEEE), it is strongly recommended that the BIBTEX .bbl file be manually copied into the document (within the traditional LATEX bibliography environment) so as not to depend on external files to generate the bibliography and to prevent the possibility of changes occurring therein.

D. Biographies

Biographies for journal articles are created using the biography environment which supports an optional argument for the inclusion of a photo:

```
\begin{biography}[{\includegraphics[width=1in,height
=1.25in,clip,keepaspectratio]{./shell.eps}}]{Michael
 Shell}
.
.
\end{biography}
```

Note the extra set of braces that are required to prevent the LATEX parser from becoming confused when commands with optional arguments are used within an optional argument of another command. Alternatively, a LATEX macro (command) could be defined to facilitate a shorthand notation for the author photos. If the optional argument is not used, space will be reserved for a photo and the message "PLACE PHOTO HERE" will be displayed in place of a photo.

IEEEtran is a tad overly cautious about preventing the biography photo area from being broken across pages. If it looks as though a biography should be able to be "squeezed" at the end of a page, but instead it begins on a new page, try inserting

```
\vspace*{-2\baselineskip}
```

or so before the biography and see if it can fit.

IEEE's algorithm for spacing around biographies can be a tad complex because esthetics must be considered. IEEEtran places `\vfil` above biographies. This allows the user to shove biographies down or up as desired by placing the infinitely more stretchable `\vfill` before or after the biographies.

The photo area is 1 in wide and 1.25 in long. IEEE recommends that author photo images should be of 220 dpi (dots per inch) resolution and in gray scale with 8 bits/sample.

If no photo is available, the `\biographynophoto` environment, which does not support an optional argument or reserve space for a photo, can be used instead.

XIII. LAST PAGE COLUMN EQUALIZATION

IEEE (coarsely) equalizes the lengths of the columns on the last page. The balance is coarse in the sense that reference or biography entries are not usually broken — so the column lengths are not usually perfectly equal.

Balancing the last two columns is especially important for camera ready work. It is recommended that authors use the manual approach by putting in `\newpage` at the appropriate point or `\enlargethispage{-X.Yin}` somewhere at the top of the first column of the last page where "X.Yin" is the amount to effectively shorten the text height of the given page.

Sometimes such a command has to be located between bibliography entries. This can be a problem because, although the command can be placed within the .bbl file, it will get overwritten the next time BIBTEX is run. For this situation, IEEEtran offers a way to invoke commands just before a given reference number via the `\IEEEtriggeratref{}` command. For instance, issuing the command

```
\IEEEtriggeratref{10}
```

before the bibliography will insert a page break just before reference number ten. The command that is executed defaults to `\newpage`. However, this can be changed via the `\IEEEtriggercmd` command:

```
\IEEEtriggercmd{\enlargethispage{-5.35in}}
```

Note that manually set break points or page sizes will have to be readjusted if the document content ever changes.

There are LATEX packages, such as balance.sty [19] and flushend.sty [20], that are designed to automatically balance the columns on the last page. Flushend does not require the placement of any special command in the first column of the last page, balance.sty may. However, the use of these packages is not recommended because they are known to be less than perfectly reliable in their operation. The author of balance.sty does not guarantee that it will work with every possible type of page, especially pages with figures. Under certain circumstances, flushend.sty will cause a spacing anomaly between two lines within a reference in the second column of the last page (becomes larger than the space between references). This problem seems to result because the bibliography in IEEEtran is a list with zero space between the list items which are in footnotesize. The problem can also occur under article.cls for the same type of list. It may be possible to manually correct the flushend anomaly by tweaking the spacer at the column break via a flushend command such as "`\atColsBreak{\vskip-2pt}`", but having to do so partially defeats the purpose of using the package in the first place. If using flushend.sty or balance.sty, be sure to check the document carefully for any spacing problems — especially on the last page.

APPENDIX I
INSTALLING IEEETRAN

LATEX .cls files can be accessed system-wide when they are placed in the

```
<texmf>/tex/latex
```

directory, where `<texmf>` is the root directory of the user's TEX installation. It is recommended that the user create a subdirectory

```
<texmf>/tex/latex/IEEE
```

for all IEEE related LATEX class and package files. On some LATEX systems, the directory look-up tables will need to be refreshed after making additions or deletions to the system files. For teTEX and fpTEX systems this is accomplished via executing

```
texhash
```

as root. MiKTEX users can run

```
initexmf -u
```

to accomplish the same thing.

Users not willing or able to install the files system-wide can make the copies local, but will then have to provide the path (full or relative) as well as the filename when referring to them in LATEX.

APPENDIX II
POSTSCRIPT/PDF OUTPUT

Unfortunately, many LATEX systems are not properly configured to produce quality PostScript and/or PDF output. This is even more of a problem with IEEE related work because the font combination IEEE uses is known to cause problems with some LATEX setups — especially those that were installed prior to mid-2002. This has been a chronic and very aggravating issue for many organizations that accept work created using LATEX.

To assist IEEE authors in detecting and correcting problems with LATEX PostScript/PDF generation, the "Testflow" diagnostic suite was developed [21]. Authors are *strongly* encouraged to take the time to go through the testflow diagnostic and identify and correct potential problems *before* their LATEX systems have to be relied on for production work. Papers with problems such as incorrect margins, font types, PDF format errors and/or improper font embedding can incur delays during the manuscript acceptance process.

APPENDIX III
OTHER USEFUL EXTERNAL PACKAGES

A. The url.sty Package

Papers that contain URLs, email address, etc., can likely benefit from the use of the url.sty LATEX package [22] which provides for more intelligent line breaking within such structures.

B. The IEEEtrantools Package

Some of the unique commands provided by the IEEEtran LATEX class may be of use in non-IEEE related work using other class files (e.g., dissertations, technical reports, etc.). The IEEEtrantools.sty package [23] provides several popular IEEEtran commands including `\PARstart`, the IEEE style IED list environments, and the IEEEeqnarray family of commands. The IEEEtrantools package is not needed under, and should not be loaded with, the IEEEtran class. See the IEEEtrantools documentation for more details.

APPENDIX IV
COMMON USER MISTAKES

Most user mistakes with IEEEtran involve doing too much rather than too little. Older class files may have required hacks in order to get the formatting closer to that of IEEE. These tweaks are no longer needed. Users should carefully check all the loaded packages to ensure that they are still useful under the latest version of IEEEtran. Don't load packages just because "this is the way it always has been done." The same is true for manually adjusted spacing, margins, paper sizes, etc.

Another common category of error is to do things that compromise the portability of the LATEX code. If the .tex source code will ever have to be shared with another party (IEEE, etc.) it is important to code in traditional LATEX fashion and not to utilize system dependent features, interfaces, graphics formats, drivers or fonts — and pdfLATEX users should ensure that their .tex code will also compile successfully without modification on traditional LATEX/DVIPS systems.

Below are a few of the more commonly encountered mistakes to avoid.

Altering the default fonts: Authors should allow IEEEtran to manage the fonts. Do not attempt to use packages that override the default fonts such as pslatex, mathptm, etc. Likewise, authors should not attempt to alter the default font encoding (fontenc.sty) or input encoding (inputenc.sty).

Altering the default spacings, section heading styles, margins or column style: Authors should not attempt to manually alter the margins, paper size (except as provided in IEEEtran class options) or use packages that do so (geometry.sty, etc.) There should be no need to add spacing around figures, equations, etc. (Except possibly for double column floats as described in Section IX-C.)

Using incorrect graphics file formats: LATEX has always favored the use of Encapsulated PostScript (EPS) for graphics — and for good reason. EPS supports both vector (that is, containing objects such as lines, circles, etc., that are mathematically described) and bitmap (that is containing only samples in the form of pixels) images. The former should always be used for drawings, graphs, charts, etc., while the latter usually has to be employed with photos (because their contents usually cannot be easily described mathematically). The drawing and graphing tools used by the author should be capable of outputing *directly*[8] in vector (EPS) format. Vector EPS images can be scaled, rotated and magnified without undergoing degradation such as pixelization or becoming gray or "jaggedy." For photos, IEEE recommends the use of EPS (which is easy to directly import into LATEX in a portable manner) or TIFF. The use of other graphic formats such as BMP, EMF, etc., is currently unacceptable for IEEE journals. IEEE conferences typically support more graphics formats than IEEE journals. For example, the use of PDF graphics (with pdfLATEX) should be acceptable for conference papers.

Using older graphics packages: Authors should not use anything other than the graphics and/or graphicx (preferred) package for figures. Older interfaces such as psfig, epsf, etc., have been obsolete for many years.

Specifying graphics drivers: When loading the graphics or graphicx package, authors should not specify an optional graphics driver, e.g.,

```
\usepackage[pctex32]{graphicx}
```

because such code will not work properly on other systems that use a different graphics driver. If the default driver (which is loaded when none is specified) is incorrect for a LATEX system, it is best to alter the `<texmf>/tex/latex/config/graphics.cfg` file than to have to manually specify a graphics driver in every .tex file. It should be acceptable to load the pdftex driver provided that the user's .tex code first tests for the use of pdflatex. The bare example files contain code that illustrates how to properly do this.

Failing to properly divide long equations: It is the author's responsibility to ensure that all equations fit within the width of their columns. Admittedly, breaking an equation is not always easy to do and two column formatting places serious constraints on allowed equation width. However, only the author can divide his/her equation without unintentionally altering its meaning or affecting readability. Using subfunctions is a valid way to reduce to width of an equation, but altering the math font size is not.

Manually formatting references: Not only is this error prone, but requires a lot of work as well. It is better to use the IEEEtran BIBTEX style [18].

APPENDIX V
KNOWN ISSUES

The small caps font used in the free LATEX systems have about 80% the height of normal sized letters. However, the small caps font IEEE uses in the journals is slightly smaller with a ratio of around 75%. So, the widths of the section headings produced under the free LATEX systems will be slightly wider than that used in actual journals. The small caps font used in many commercial LATEX systems (such as those from YandY) has a ratio of about 65%. So, those systems will produce section headings that are narrower than those in IEEE publications. Such variations should not be cause for concern.

hyperref.sty versions prior to 6.72u will interfere with the optional argument to `\appendix`. It is difficult to ensure that LATEX packages are fully compatible with hyperref.sty. Therefore, the use of the hyperref package is not recommended for documents that are to be submitted to IEEE. In particular, it is known that current versions of cite.sty and hyperref.sty do not work together. IEEEtran.cls mediates these two packages in a way that allows cite.sty to work as normal, but without the citation numbers being hyperlinked by hyperref.sty. See the comments in the bare template files for more details.

The amsthm.sty package will cause a name clash error as it will try to define a proof environment which is already provided by IEEEtran.cls. It is generally not a good idea to load packages that attempt to provide replacements for any of the custom commands provided by IEEEtran.cls.

---

[8] Once an image in EPS vector form is converted to a bitmap form (GIF, TIFF, JPEG, etc.) it will almost always be irretrievably locked into bitmap form even if it is later converted back into EPS.

APPENDIX VI
THE IEEEEQNARRAY COMMANDS

(*Optional — for advanced users*)

Virtually all LaTeX alignment commands such as `\eqnarray`, `\array` and `\tabular` are based on the TeX command `\halign`. LaTeX's goal of simplifying the use of `\halign` is noble. However, in hiding much of the lower level interface, a fair degree of flexibility is lost. This has resulted in the development of several packages such as amsmath [7], array.sty [13], and the MDW tools [8], each of which provides much more powerful alignment structures.

IEEEtran also provides its own unique set of alignment tools which are known as the IEEEeqnarray family. The design philosophy of the IEEEeqnarray family is to provide a LaTeX alignment interface that is based more closely on the underlying `\halign`, but to couple this with high level column definition management and automated preamble building mechanisms (which are tedious to do in TeX). As a result, the IEEEeqnarray family of commands are flexible enough to be almost universal replacements for all the other LaTeX commands for producing multiline equations and aligned box structures such as matrices and tables of text and/or mathematics. Because the user is shielded less from the `\halign` underpinnings, the rules of operation are more involved. So, the IEEEeqnarray commands are aimed primarily toward the more advanced LaTeX users.

The use of the IEEEeqnarray family of tools described in this section is totally *optional*. The IEEEeqnarray code is self-contained and does not depend on other alignment packages — which can be used along side, or in place of, it. The IEEEtrantools.sty package (See Appendix III-B) is available for those who wish to use the IEEEeqnarray family outside of IEEEtran.cls.

A. IEEEeqnarray

The IEEEeqnarray environment is for multiline equations that occupy the entire column. It is used in much the same way as `\eqnarray`, but with two additional arguments, one of which is mandatory and the other is optional:

```
\begin{IEEEeqnarray}[decl]{cols}
.
\end{IEEEeqnarray}
```

The optional argument is for commands that are to be executed within the environment, but before the alignment actually begins. This is just like the local control of the IEEEtran IED list environments. There is also a global control, `\IEEEeqnarraydecl`, which is executed just prior to the local control. By default, `\IEEEeqnarraydecl` is defined to be `\relax`. As mentioned in Section X, users should be careful not to allow unwanted spaces to occur in these controls because such things will appear just before the IEEEeqnarray structure. Furthermore, remember that, to prevent the LaTeX parser from becoming confused, the contents of an optional argument must be enclosed in braces if the argument contains commands with optional arguments.

TABLE IV
IEEEEQNARRAY PREDEFINED COLUMN TYPES

I.D.	**Description**	**I.D.**	**Description**
`l`	left math	`v`	vertical rule
`c`	centered math	`vv`	two vertical rules
`r`	right math	`V`	double vertical rule
`L`	left math with ords	`VV`	two double vertical rules
`C`	centered math with ords	`h`	horizontal rule
`R`	right math with ords	`H`	double horizontal rule
`s`	left text	`x`	empty
`t`	centered text	`X`	empty math
`u`	right text		

Note: `S`, `T`, `U`, `p`, and `P` are likely to be used in future versions.

TABLE V
IEEEEQNARRAY PREDEFINED COLUMN SEPARATION (GLUE) TYPES

I.D.	**Width***	**I.D.**	**Width**
`!`	$-1/6$ em	`.`	`0.5\arraycolsep`
`,`	$1/6$ em	`/`	`1.0\arraycolsep`
`:`	$2/9$ em	`?`	`2.0\arraycolsep`
`;`	$5/18$ em	`*`	0pt plus 1fil
`'`	1 em	`+`	1000pt minus 1000pt
`"`	2 em	`-`	0pt

*All em values are referenced to the math font.
1 em = `\quad`, 2 em = `\qquad`

The mandatory argument `cols` contains the column and inter-column separator spacing ("inter-column tabskip glue" in TeXspeak) type specifiers. Column types are identified by letters. Several predefined column types are available as shown in Table IV. There are two kinds of glue types. Predefined glue types are indicated by various punctuation marks as shown in Table V. User defined glue types are indicated by numbers.

The rules for placing these specifiers are as follows: (1) no two glue specifiers can appear next to each other — they are not additive and must be separated from each other by at least one column specifier; (2) zero inter-column spacing will be assumed between back-to-back column specifiers; (3) because of rule one, back-to-back numerals will be considered as being a single glue specified by the numerical value represented by all the digits; (4) a multiletter column specifier can be accessed by enclosing the letters within braces (otherwise it would be interpreted as being several single letter column specifiers). Because of rule three, braces are not needed around multidigit glue specifiers; (5) there must be at least one column specifier, but there is no fixed upper limit of how many columns can be supported; and (6) for `\IEEEeqnarray`, "+" centering glue will be assumed at each end of the `cols` specification if no column glue is specified there. This results in a centered structure like `\eqnarray` (the 1000pt minus 1000pt glue on each side "compresses" as needed from each side of the main text column to center that which is between). Also, `\IEEEeqnarray` automatically adds a hidden column for equation numbers to the right of the last specified column. Currently, there is no support for equation numbers on the left side.[9]

---

[9] This is not to say that its impossible with the existing capability, just ugly.

B. Defining Column Types

New column types are defined with the

```
\IEEEeqnarraydefcol{col_id}{predef}{postdef}
```

command. The *`col_id`* argument contains the name of the column specifier which should consist only of one or more letters. A given column specifier, even the predefined ones, can be redefined at will without warning or error.[10] The *`predef`* argument contains the commands that will be inserted before each cell in the column. The *`postdef`* argument contains the commands that will be inserted after each cell in the column. For example,

```
\IEEEeqnarraydefcol{g}{\hfil$\clubsuit$}{$\diamondsu
it$\hfil}
```

Will define a "g" text column which will place club and diamond suit symbols on either side of a cell's contents and center the respective structure within the cell. e.g.,

$\clubsuit$Hello$\diamondsuit$

Using `\hfil` to control cell alignment allows the user to override the column alignment on a cell-by-cell basis by placing the infinitely more stretchable `\hfill` on one or both sides of a cell's contents. `\hfill` can even be placed between items in a cell to force them apart and against the "cell walls." The IEEEeqnarray predefined columns are designed to allow user overrides via `\hfill` whenever possible (even for the math mode cells).

Please note that TeX will not allow unmatched braces within the arguments of commands. If braces are needed, such as for the argument of a command, they will have to be provided within the cells themselves. For example,

```
\IEEEeqnarraydefcol{myp}{\parbox[c]{0.5in}}{}
\begin{IEEEeqnarraybox}{{myp}c}
{first\\second}&\alpha\\
&\beta%
\end{IEEEeqnarraybox}
```

defines a column type named "myp" that will place text within a 0.5 inch wide parbox which is centered on the cell's baseline. Note that because the column type name consists of more than one letter, it has to be enclosed within an extra set of braces in the column specifications or else it would be interpreted as three adjacent columns "m," "y," and "p." Also, the contents of the cell must be enclosed within braces so that (1) the `\parbox` command sees the entire contents as its argument; and (2) the newline within the parbox will not be interpreted as being the end of the alignment row. Be aware that it can happen that a column is given an empty cell, such as in the second row in the example, or when entering blank separator rows. When this happens, a `\relax` will appear in the column which will be acquired as the command's argument. Therefore, commands in column definitions that acquire arguments from the cells should not choke if fed `\relax`.

For reference, the definitions of the predefined column types are shown here:

```
% math
\IEEEeqnarraydefcol{l}{$\IEEEeqnarraymathstyle}{$\hfil}
\IEEEeqnarraydefcol{c}{\hfil$\IEEEeqnarraymathstyle}{$\hfil}
\IEEEeqnarraydefcol{r}{\hfil$\IEEEeqnarraymathstyle}{$}
\IEEEeqnarraydefcol{L}{$\IEEEeqnarraymathstyle{}}{{}$\hfil}
\IEEEeqnarraydefcol{C}{\hfil$\IEEEeqnarraymathstyle{}}{{}$\h
fil}
\IEEEeqnarraydefcol{R}{\hfil$\IEEEeqnarraymathstyle{}}{{}$}
% text
\IEEEeqnarraydefcol{s}{\IEEEeqnarraytextstyle}{\hfil}
\IEEEeqnarraydefcol{t}{\hfil\IEEEeqnarraytextstyle}{\hfil}
\IEEEeqnarraydefcol{u}{\hfil\IEEEeqnarraytextstyle}{}
% vertical rules
\IEEEeqnarraydefcol{v}{}{\vrule width\arrayrulewidth}
\IEEEeqnarraydefcol{vv}{\vrule width\arrayrulewidth\hfil}{\h
fil\vrule width\arrayrulewidth}
\IEEEeqnarraydefcol{V}{}{\vrule width\arrayrulewidth\hskip\d
oublerulesep\vrule width\arrayrulewidth}
\IEEEeqnarraydefcol{VV}{\vrule width\arrayrulewidth\hskip\do
ublerulesep\vrule width\arrayrulewidth\hfil}%
{\hfil\vrule width\arrayrulewidth\hskip\doublerulesep\vrule
 width\arrayrulewidth}
% horizontal rules
\IEEEeqnarraydefcol{h}{}{\leaders\hrule height\arrayrulewidt
h\hfil}
\IEEEeqnarraydefcol{H}{}{\leaders\vbox{\hrule width\arrayrul
ewidth\vskip\doublerulesep\hrule width\arrayrulewidth}\hfil}
% plain
\IEEEeqnarraydefcol{x}{}{}
\IEEEeqnarraydefcol{X}{$}{$}
```

Note the inclusion of the commands `\IEEEeqnarraymathstyle` and `\IEEEeqnarraytextstyle` in the math and text columns, respectively. These commands allow the user to control the style of all of the math and text columns. However, because the changes apply to all the columns, the user will have to define new column types if different styles are needed in the same alignment (or different styles can be manually specified in each cell). The default definitions for these commands are

```
\newcommand{\IEEEeqnarraymathstyle}{\displaystyle}
\newcommand{\IEEEeqnarraytextstyle}{\relax}
```

which allows the text columns to be in whatever style was in effect when the alignment was started and the default math style will be in display style, but can be easily changed as needed. e.g.,

```
\begin{IEEEeqnarray}[\renewcommand{\IEEEeqnarraymath
style}{\scriptstyle}]{rCl}
```

will result in script style math columns.

The columns relating to vertical and horizontal lines will be discussed in Section VI-K as they are typically used only when producing tables.

The "x" and "X" columns are basic empty text and math mode columns without any formatting or style controls.

C. Defining Glue Types

New column separation glue types are defined with the

```
\IEEEeqnarraydefcolsep{colsep_id}{def}
```

command. The *`colsep_id`* argument contains the number of the column separation glue specifier which should consist only of numerals. Different glue type names must have different numerical values. (Don't get too cute — "007" is identical to "7".) User defined column glue specifiers can be redefined at will without warning or error. The *`def`* argument contains the width of the given glue type. Widths may be specified as absolute values or reference length commands:

---

[10] Thus allowing new predefined column types to be added without breaking existing code.

```
\IEEEeqnarraydefcolsep{9}{10pt}
\IEEEeqnarraydefcolsep{11}{2\tabcolsep}
```

The glue type widths are *not* evaluated when defined, but are evaluated each time they are actually referenced as IEEEeqnarray column specifiers. Thus, for the second definition in the example above, if `\tabcolsep` were to be revised after the glue type was defined, the revised value would be what is used.

Rubber lengths are allowed too. The fact that the centering glue "+" is a known value can be exploited to achieve some interesting effects. For example,

```
\IEEEeqnarraydefcolsep{17}{200pt minus 200pt}
```

will produce a column separation glue that is always $1/5$ of the width of the distance from the equation sides to the ends of the main text columns. And, of course, "+" can be used as needed to produce groups of equally spaced equations as in amsmath's `\align`:

```
\begin{IEEEeqnarray}{Rl+Rl+Rl}
```

D. A Simple Example of Use

The example in Section VIII can be implemented using `\IEEEeqnarray` via

```
\begin{IEEEeqnarray}{rCl}
Z&=&x_1 + x_2 + x_3 + x_4 + x_5 + x_6\IEEEnonumber\\
&&+\:a + b%
\end{IEEEeqnarray}
```

As shown in Table IV the "C" column type is a centered math mode column with empty ords ("`{}`") on either side. So, there is no need to place empty ords around the equal sign. As with `\eqnarray`, the `&` separate the column cells and are where the inter-column separation glue will appear (when nonzero).

Note the presence of the `%` at the end of the second row. TeX does *not* ignore spaces that occur *before* commands or `&` column delimiters, but does ignore those that occur *after*. Most LaTeX alignment implementations shield the user from this behavior by removing all spacing previous to `&`, `\\` and `\end`. The IEEEeqnarray family does not do this! So, it is important to prevent spaces (including implied ones at the end of lines) from occurring before such commands unless they are wanted. Suspect this problem if there is an unexplained offset within a column. In the given example, unwanted spacing is not an issue because end spacing is ignored in math mode anyway. However, it would be an issue if the columns used text mode instead.

Alternatively, one could use a two column form:

```
\begin{IEEEeqnarray}{Rl}
Z=&x_1 + x_2 + x_3 + x_4 + x_5 + x_6\IEEEnonumber\\
&+\:a + b%
\end{IEEEeqnarray}
```

E. Equation Numbering

Like `\eqnarray`, `\IEEEeqnarray`, has a "star form," `\IEEEeqnarray*`, which does not place equation numbers at the end of each row by default. The default behavior of individual rows can be overridden by placing the commands `\IEEEnonumber` or `\IEEEyesnumber` as needed in the last column.[11] `\IEEEeqnarray` also provides a `\IEEEyessubnumber` which can be used to enable a subequation number for the given line:

```
&+\:a + b\IEEEyessubnumber\label{myfirstsubeqn}\\
```

Note that labels for subequations must be placed *after* the `\IEEEyessubnumber` command because, prior to that point, the label will reference the equation number *that would have been used* if there had not been a request for a subnumber. To support this feature, IEEEtran defines its own `IEEEsubequation` counter (reset with changes to `equation`) and `\theIEEEsubequation` command.[12] The first line without an `\IEEEyessubnumber` will revert back to conventional equation numbering. Currently, there is no support for subequation numbering to be the default for all the lines. So, users will have to call `\IEEEyessubnumber` every time a subequation number is needed.

Please be aware that `\IEEEeqnarray`, like `\eqnarray`, will, if the equation is long enough, overwrite the equation number without warning![13] For cases when this happens, users can insert a `\IEEEeqnarraynumspace` command at the end of the line (after any `\IEEEyessubnumber` if used) which will insert a space equal in width to the displayed equation number:

```
··· + x_z \IEEEyessubnumber\IEEEeqnarraynumspace\\
```

As a result, the entire multiline equation will be slightly shifted to the left. IEEE often does the same thing in its journals when confronted by this situation. If an overfull hbox results, the offending equation line will have to be further divided.

F. Extra Vertical Spacing and Page Breaks

Like `\eqnarray`, `\IEEEeqnarray`'s `\\` command supports a star form which inhibits page breaks at the given line as well as an optional extra vertical spacing argument:

```
&+\:a + b\\*[5pt]
```

Users are reminded from Section VIII that amsmath will configure LaTeX to disallow page breaks within multiline equations — including those made by `\IEEEeqnarray` because it also honors the value of `\interdisplaylinepenalty`.

Also like `\eqnarray`, `\IEEEeqnarray` normally places a small amount of extra spacing (as specified by the length command `\jot`) between lines to "open up" equations as well as to prevent large symbols from coming to close to the lines above them.

G. IEEEeqnarraybox

`\IEEEeqnarray` is not suitable for producing structures such as matrices and tables because it must have exclusive

---

[11] The aliases `\nonumber` and `\yesnumber` are also provided. However, their use is not recommended because some packages (such as MDW tools) redefine them in an IEEEeqnarray incompatible way.

[12] What is actually displayed is the `\theIEEEsubequationdis` command.

[13] This behavior is extremely difficult to avoid if the equation line is to remain centered irrespective of the width of the equation number — without even considering the subjective question of how close is too close for any given case!

access to the main text column and cannot be nested within other structures. For these applications, the `\IEEEeqnarraybox` command is provided. `\IEEEeqnarraybox` differs from `\IEEEeqnarray` in the following ways:

1) the entire contents are wrapped within a box and therefore, can be nested within other display or alignment structures (such as `\equation`, `\IEEEeqnarray` or even another `\IEEEeqnarraybox`). Note that, like all box structures, page breaks are *not* allowed between the rows of an `\IEEEeqnarraybox`;
2) the default glue at the outer ends of the first and last columns is 0 pt ("-"), not "+" centering glue as with `\IEEEeqnarray`;
3) no automatic (hidden) equation number column is provided;
4) the star form "`\IEEEeqnarraybox*`" turns off the extra `\jot` vertical spacing after each row;
5) `\IEEEeqnarrayboxdecl` is the global control.

Two subforms are provided: `\IEEEeqnarrayboxm` which is for use within math modes and is analogous to `\array`; and `\IEEEeqnarrayboxt` which is for use within text modes and is analogous to `\tabular`. If called via `\IEEEeqnarraybox` the current math/text mode will be auto-detected and the proper subform will automatically be selected. Therefore, `\IEEEeqnarraybox` can replace `\array` as well as `\tabular`.

The syntax for `\IEEEeqnarraybox` is similar to `\IEEEeqnarray`, but with two additional optional arguments:

```
\begin{IEEEeqnarraybox}[decl][pos][width]{cols}
.
\end{IEEEeqnarraybox}
```

The `pos` argument can be one of `t`, `c`, or `b` to control where the box will be vertically aligned relative to the current baseline: `t` at the top row; `c` at the center;[14] `b` at the bottom row. The default is `b`.

The `width` argument specifies the width the box. **Warning:** if a width is specified, there *must* be one or more rubber lengths in the inter-column glue specifiers (such as that of "*" or "+") so that the box can be sized as needed. If there is no such glue, or the glue provided cannot stretch/shrink enough as needed, the box cannot be sized to `width` and an underfull or overfull hbox error will result. If no `width` argument is provided, the box will be set to its natural width (and the use of rubber inter-column glue will not be required).

`\IEEEeqnarraybox` uses the same column and glue type specifiers/definitions as `\IEEEeqnarray`.

H. Line Spacing in LaTeX

Before discussing some of the more advanced aspects of vertical spacing control in the IEEEeqnarray family, it is important to review the details of LaTeX's line spacing algorithm. Normally, baselines are separated by the amount given by the length command `\baselineskip`. With each change in font size, `\baselineskip` is reset to the default value for that font size (multiplied by `\baselinestretch`). Then the value of `\baselineskip` is saved to the length variable `\normalbaselineskip` (so that the normal value can be later referenced even if `\baselineskip` is set to another value by the user). However, if the top of a line ever gets closer than `\lineskiplimit` to the bottom of the line above it, the use of `\baselineskip` will be suspended and `\lineskip` spacing will be placed between the two lines.[15]

This system works well for text. However, for mathematics, whose symbols have a much higher dynamic range of heights and depths, it is usually better to go ahead and always add an extra fixed amount of space (`\jot`) as mentioned in Appendix VI-F.

When the IEEEeqnarray family is loaded, two new length commands are defined: `\normaljot` and `\normalsizebaselineskip` which store the nominal values of `\jot`[16] and the `\baselineskip` of the normalsize font, respectively, so that they can be always be referred to even if other values are currently being used.

At the start of `\IEEEeqnarray/box`, but before the local or global controls, the following initialization takes place:

```
\lineskip=0pt
\lineskiplimit=0pt
\baselineskip=\normalbaselineskip
\jot=\normaljot
```

Thus, `\baselineskip` is set to the normal value for the current font, `\jot` is restored to its nominal value and the `\lineskiplimit` system is disabled.[17]

This system is designed to better facilitate nested IEEEeqnarraybox structures as well as to help prevent the user from encountering seemingly uncontrollable spacing behavior (e.g., "How *do* I get rid of that unwanted space?!").

I. The IEEEeqnarray Strut System

When creating tables, especially tables with vertical rules, vertical space between the rows of the table is not generally desirable because such space will suspend the column cell definitions and "cut across" any vertical rules that may be present. Yet, there must be a way to keep rows adequately spaced apart. To solve this problem, the IEEEeqnarray/box commands provide an integrated system to manage struts[18] contained in a hidden column on the right end of each IEEEeqnarray/box structure.

The struts in each row will be set to a default strut height and depth. Normally, the default strut height and depth are initialized to zero, so the struts will effectively not be present. The user can set the default strut values via the

```
\IEEEeqnarraystrutsize{height}{depth}[decl]
```

command which can be placed in a local or global control. The optional argument is for commands which will be executed

---

[14] Centering is actually done along the "math axis", not exactly on the text baseline (but quite close to it). Most LaTeX users are not aware of this minor distinction.

[15] Within IEEEtran.cls, `\lineskiplimit` and `\lineskip` are zero — if things get too close it is the author's responsibility to correct the problem without having IEEEtran.cls second guessing the author's intent.

[16] Within IEEEtran.cls, the nominal value of `\jot` is 25% of the baselineskip for the normalsize font.

[17] As long as rows cannot be of negative height.

[18] "Struts" are vertical rules of zero width, but of finite height.

prior to the evaluation of the height and depth arguments. Thus,

```
\IEEEeqnarraystrutsize{0.5\baselineskip}{}[\large]
```

will set the default strut height to half the baselineskip used by the large font size, even if the current baselineskip (and/or font size) is different. The commands which are executed within the optional argument are contained within their own environment so as not to have any effects outside of the `\IEEEeqnarraystrutsize` command. For mimicking the action of `\baselineskip`, the typically recommended height and depth of struts is 70% and 30%, respectively, of `\normalbaselineskip`[19]. `\IEEEeqnarraystrutsize` will assume these values if its height and/or depth arguments are left blank. e.g., in the previous example, the strut *depth* will be set to 30% of `\normalbaselineskip` for the large font size.

There is also a

```
\IEEEeqnarraystrutsizeadd{height}{depth}[decl]
```

command which will *add* to the current default strut values and can be used much like the `\extrarowheight` parameter of the array.sty package. Empty arguments are assumed to be 0 pt.

`\IEEEeqnarraystrutsize` and `\IEEEeqnarraystrutsizeadd` can also be used at the *end of the last* column to alter the strut size used for a particular row (the default strut values of the other rows will not be affected).

There is also a

```
\IEEEstrut[height][depth][decl]
```

which produces a strut. It can be used whenever a "manual" strut is needed — even outside the `\IEEEeqnarray/box` environments. If a height or depth argument is not provided (or empty) then these will be assumed in the same way as `\IEEEeqnarraystrutsize` does.

For diagnostic purposes (in order to see if any row objects exceed the height of the struts), the command `\IEEEvisiblestrutstrue` can be placed with an `\IEEEeqnarray/box` or `\IEEEstrut` control to make the struts visible.

When using `\IEEEeqnarraybox` to produce tables that contain vertical lines, it is usually desirable to shutdown the `\baselineskip` system and switch over to pure strut spacing. The following command sequence, placed within a local or global control, will serve this purpose:

```
\IEEEeqnarraystrutsize{0.7\normalbaselineskip}{0.3\n
ormalbaselineskip}[\relax]
\setlength{\baselineskip}{0pt}%
\setlength{\lineskip}{0pt}%
\setlength{\lineskiplimit}{0pt}%
\setlength{\jot}{0pt}%
```

Note the use of "%" to prevent the ends of the lines that end in braces from being interpreted as unwanted spaces. Because of the frequent need to call this sequence, the IEEEeqnarray family provides the `\IEEEeqnarraystrutmode` command which does the same thing.

J. Overriding Column Types

Within a row, one or more column types can be overridden by placing the command

```
\IEEEmulticol{num_cols}{col_type}{text}
```

as the *very first* command in a cell. This command is the IEEEeqnarray equivalent of `\multicolumn`. The first argument is the number of columns to override (cutting through any inter-column glues as needed). The second argument is the column type specifier to use. The third argument contains the cell text. The third argument will have to be enclosed within an extra set of braces if the column type is to acquire it as an argument — as was done with the "myp" parbox column type in the example earlier in this Appendix (VI-B).

There is also the `\IEEEeqnarrayomit` command which, when used as the very first command in a cell, will suspend the use of the normal column type for that cell. This is somewhat like a quicker version of `\IEEEeqnarraymulticol{1}{x}{}`.

Users are cautioned not to use commands like these (e.g., `\multicolumn`) that are designed for other alignment environments.[20]

K. Predefined Column Types for Rules

Several of the predefined column types produce vertical or horizontal lines. Note that, in the IEEEeqnarray family, rules are declared and treated as normal column types — they are not hidden. Although this approach may increase the number of columns the user has to keep track of, especially when creating tables, it does offer a great deal of flexibility by allowing the user to override, or otherwise manipulate, any column type (including those that produce rules) at will.

All of the predefined rule column types use the `\arrayrulewidth` length to determine the line thickness and `\doublerulesep` for the spacing of double rules.

The "`v`" column type produces a vertical rule, "`vv`" produces two back-to-back vertical rules which will appear as one rule of twice the normal thickness. "`V`" produces a double vertical rule with `\doublerulesep` spacing between its two lines. "`VV`" produces two back-to-back double vertical rules which will appear to be three vertical rules, the middle one of which being twice as thick as the other two. It is possible to "spread apart" the "`vv`" and "`VV`" types by placing a spacer within their columns — thus they can be used to generate two single, or double, vertical rules whose separation distance is programmable.

The "`h`" and "`H`" types produce single and double horizontal rules, respectively. Horizontal rule types are not normally used in the column specifications, but rather with the `\IEEEeqnarraymulticol` command in order to draw a horizontal rule across one or more column(s).

Please be aware that the line commands of other alignment environments may not work properly within the IEEEeqnarray

---

[19] Note that this *not* the *normalsize* baselineskip, but the normal baselineskip for the current font.

[20] Those familiar with TEX may be interested to know that TEX's `\omit`, `\span` and `\multispan` should work in `\IEEEeqnarraybox`, but not in `\IEEEeqnarray` because of the need to track column usage with a hidden counter.

family which provides its own ways of doing these types of things. In particular, `\cline` is totally incompatible — users should use the `\IEEEeqnarraymulticol{num_cols}{h}{}` command instead. However, `\vline` and `\hline` should work — unless another LaTeX package redefines them in some incompatible way. The IEEEeqnarray family provides its own version of `\vline`:

```
\IEEEeqnarrayvrule[rule_thickness]
```

which produces a vertical rule extending from the top to bottom of a cell without overriding the column type. The optional argument is for specifying the rule thickness which defaults to `\arrayrulewidth` if no argument is provided.

The IEEEeqnarray row commands (discussed in the next section) provide some alternatives to `\hline`.

L. Row Commands

The IEEEeqnarray family has several commands that can be used to produce special rows which span all the columns. Unless otherwise noted, the commands described here must issued as the very first command in a given row.

To produce a spacer row which relies on the strut system, use

```
\IEEEeqnarrayseprow[height][decl]
```

The first argument specifies the height of the strut row, if left blank or empty, the default value of `0.25\normalbaselineskip` will be assumed. The second optional argument is for commands which will be executed prior to the evaluation of the first argument as is done with `\IEEEeqnarraystrutsize`. `\IEEEeqnarrayseprow` will *not* interrupt the column definitions, so it will not cut vertical lines. If column definition suspension is desired, use the cutting form which will override all the column types in the entire row:

```
\IEEEeqnarrayseprowcut[height][decl]
```

To produce a horizontal rule row, use:

```
\IEEEeqnarrayrulerow[rule_thickness]
```

which will override all the column definitions with one that produces a horizontal rule. If the optional rule thickness is not specified, the value of `\arrayrulewidth` will be used.

To produce a double rule row, use:

```
\IEEEeqnarraydblrulerow[rule_thickness][spacing]
```

which will produce a rule row, a (noncutting) separation row, followed by another rule row. If the optional rule thickness is not specified, the value of `\arrayrulewidth` will be used when producing each of the two rule rows. If the optional separation distance is not specified, the value of `\doublerulesep` will be used. There is also a cutting form:

```
\IEEEeqnarraydblrulerowcut[rule_thickness][spacing]
```

which works the same way except that the separation row will override all the column definitions. (Vertical rule columns will not appear inside the double rule row produced by this command.)

M. Useful Low Level TeX Commands

Although the use of lower level TeX commands is generally frowned upon in LaTeX, some of them are just too helpful to ignore.

`\phantom{}` produces an invisible box with the width, height and depth of its contents, but the contents themselves will not appear in the output. There is also the `\hphantom{}` and `\vphantom{}` forms which retain only the contents' width, or its height and depth, respectively. As an example, look carefully at the footnotes at the bottom of Table V. This table was produced using the `\IEEEeqnarraybox` command. The footnotes are actually contained within the last two rows of the table. Note how the left sides of the footnotes line up, even though the first one has a superscript asterisk for a footnote symbol. The reason that the second row lines up is because, at its left side, it employs a horizontal phantom of the very same symbol:

```
\hphantom{\textsuperscript{*}}
```

Vertical phantoms can be used to equalize row height or spacing — such as to get matrices that fit within brackets of the same size even though one has "tall" symbols and the other not.

The opposite of `\hphantom{}` is `\rlap{}` which displays its contents, but with zero width. There is also an `\llap{}` which does the same thing, but the contained object will appear just to the left of the given point, rather than after as with `\rlap`. For example, look closely at the first "Width" column heading in Table V. The word "Width" is centered irrespective of the asterisk. That is because the width of the asterisk was zeroed:

```
Width\rlap{\textsuperscript{*}}
```

The vertical analog of `\rlap` is `\smash{}` which reduces the apparent height and depth of its contents to zero. (LaTeX's `\raisebox{0pt}[0pt][0pt]{}` does about the same thing, and also provides an adjustable vertical offset.) `\smash` can be used when space is already reserved for an object, but that LaTeX does not "know" this and would allocate unwanted additional vertical space. One good use of smash for table objects that are to be "slipped" into a hidden row of zero height, or into a row which is to be no higher than the "short" things, such as horizontal rules, that are in its other columns.

The TeX `\noalign{}` command can be used within IEEEeqnarray family to inject text which is outside of the alignment structure. For example,

```
\begin{IEEEeqnarray}{rCl}
A_1&=&7\IEEEyessubnumber\\
A_2&=&b+1\IEEEyessubnumber\\
\noalign{\noindent and\vspace{\jot}}A_3&=&d+2\IEEEyessubnumber%
\end{IEEEeqnarray}
```

produces

$$A_1 = 7 \tag{3a}$$

$$A_2 = b + 1 \tag{3b}$$

and

$$A_3 = d + 2 \tag{3c}$$

When employed, `\noalign` must be the *very* first command in a row — even before any `\IEEEeqnarraymulticol`, `\IEEEeqnarrayomit`, or row commands.

Be forewarned that the proper use of `\noalign` can be tricky. There are three potential issues. (1) Remember that `\noalign` will place its contents outside of the alignment. So, the line spacing controls of the IEEEeqnarray commands will not be in effect. The user may have to manually add `\baselineskip` and/or `\jot` spacing as needed (which was done in the previous example). (2) Furthermore, `\noalign` does *not* automatically place its contents within a box. However, nonaligned material *must* be placed within a *horizontal* box when within the vertical box produced by the `\IEEEeqnarraybox` command. Therefore, when using `\noalign` within a `\IEEEeqnarraybox`, be sure to wrap things up in an `\hbox{}`:[21]

```
\noalign{\hbox{and therefore}}
```

(3) Finally, there may be some issues related to how easily page breaks occur around the `\noalign` lines. This is only an issue with `\IEEEeqnarray` because page breaks cannot occur within the box produced by `\IEEEeqnarraybox`. If needed, page break desirability can be altered by manually entering `\pagebreak`, or `\nopagebreak`, etc., at the end of the `\noalign` contents.

N. More Practical Examples of Use

The use of the IEEEeqnarray is somewhat complicated. However, once the building blocks and core concepts are understood, the user may find that is easier to use the IEEEeqnarray family for just about every alignment situation rather than to have to remember all the interfaces and unique behaviors of many different tools.

A few “real world” examples will now be demonstrated.

1) IEEEeqnarray Cases Structures: Cases structures can be obtained using `\IEEEeqnarraybox`:

$$|x| = \begin{cases} x, & \quad\text{for } x \geq 0 \\ -x, & \quad\text{for } x < 0 \end{cases} \tag{4}$$

which was produced using the code:

```
\begin{equation}
\setlength{\nulldelimiterspace}{0pt}
|x|=\left\{\begin{IEEEeqnarraybox}[\relax][c]{l’s}
x,&for $x \geq 0$\\
-x,&for $x < 0$%
\end{IEEEeqnarraybox}\right.
\end{equation}
```

Note the use of the large `\quad` (1 em) spacing before the conditional statements. The zeroing of `\nulldelimiterspace`, an optional step, eliminates the width of the nonvisible closing brace “`\right.`” in order to perfectly center the visible portion of the equation.[22]

Note that both branches share a common equation number. If an equation (sub)number is wanted for each branch, the preferred solution is to use the cases.sty package as discussed in Section VIII-A. However, it is possible to use `\IEEEeqnarray` to build such a thing — although it takes extra work and a few tricks to do so. For example,

$$|x| = \begin{cases} x, & \quad\text{for } x \geq 0 \qquad\qquad \text{(5a)} \\ -x, & \quad\text{for } x < 0 \qquad\qquad \text{(5b)} \end{cases}$$

was produced using the code:

```
\begin{IEEEeqnarray}[\setlength{\nulldelimiterspace}
{0pt}][{rl’s}
&x,&for $x \geq 0$\IEEEyessubnumber\\*[-0.625\normal
baselineskip]
\smash{|x|=\left\{\IEEEstrut[3\jot][3\jot]\right.}&&
\nonumber\\*[-0.625\normalbaselineskip]
&-x,&for $x < 0$\IEEEyessubnumber
\end{IEEEeqnarray}
```

A hidden middle row is used to hold the left hand side of the equality. In order to prevent this row from altering the spacing between the two branches, its height must be smashed and the extra line spacing (which consists of `\baselineskip`, plus `\jot` which is normally `0.25\baselineskip` for IEEEtran.cls.) must be removed, half from above and half from below, — making it look as though the middle row never occurred. Because the large brace cannot “see” the height of the branches, it must be manually sized with a strut. The star form of the new line commands is used to prevent the possibility of a page break within the structure.

2) Matrices: Displayed matrices can easily be created with `\IEEEeqnarraybox`:

$$I = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{6}$$

The code of this example is quite simple:

```
\begin{equation}
I = \left(\begin{IEEEeqnarraybox*}[][c]{,c/c/c,}
1&0&0\\
0&1&0\\
0&0&1%
\end{IEEEeqnarraybox*}\right)
\end{equation}
```

Because the example matrix has elements of normal height, one can use the star form of `\IEEEeqnarraybox` to turn off the extra `\jot` component of the line spacing so as to make a more compact matrix. If larger symbols had been used in the matrix, the nonstar form would be the better choice. `\arraycolsep` typically serves quite well as an element column separator. A standard small math space is added to the ends of the matrix to provide a little distance between it and its enclosing parentheses.

It is instructive to show how to construct a “small” matrix[23],

$$S = \begin{bmatrix} \scriptscriptstyle 1/2 & \scriptscriptstyle 0 \\ \scriptscriptstyle 0 & \scriptscriptstyle 3/4 \end{bmatrix} \tag{7}$$

which was produced via

```
\newcommand{\mysmallarraydecl}{\renewcommand{%
\IEEEeqnarraymathstyle}{\scriptscriptstyle}%
\renewcommand{\IEEEeqnarraytextstyle}{\scriptsize}%
\renewcommand{\baselinestretch}{1.1}%
```

---

[21] LaTeX’s `\mbox` will *not* work!

[22] The width of null delimiters is typically only 1.2 pt, and so can usually be safely ignored.

[23] IEEE authors should note that the use of small matrices is not recommended as IEEE does not usually reduce font sizes in equations or alter the main text baselineskip to accommodate in-text mathematics.

TABLE VI

NETWORK DELAY AS A FUNCTION OF LOAD

β	Average Delay	
	λ_{min}	λ_{max}
1	0.057	0.172
10	0.124	0.536
100	0.830	0.905*

*limited usability

```
\settowidth{\normalbaselineskip}{\scriptsize
\hspace{\baselinestretch\baselineskip}}%
\setlength{\baselineskip}{\normalbaselineskip}%
\setlength{\jot}{0.25\normalbaselineskip}%
\setlength{\arraycolsep}{2pt}}
%
\begin{equation}
S=\left[\begin{IEEEeqnarraybox*}[\mysmallarraydecl]
[c]{,c/c,}
1/2&0\\
0&3/4%
\end{IEEEeqnarraybox*}\right]
\end{equation}
```

The use of a user defined command, `\mysmallarraydecl`, to contain the IEEEeqnarray setup code, demonstrates how users can easily recreate their most commonly used structures by fully exploiting the on-the-fly configurability of the IEEEeqnarray family.

This example is more complex than need be in order to demonstrate a few techniques. It would be easy enough to set `\baselineskip` to the desired value, but suppose that the matrix rows are to be spaced some multiple of the `\baselineskip` of the `\scriptsize` font. Complicating matters even more is the fact that most LaTeX class files will not allow the user to execute text font size commands within math mode — and the matrix is within an equation. So, `\scriptsize` cannot be used to directly set the `\baselineskip`.

The first step is to set the math and text columns to their desired styles. Then `\baselinestretch` is setup to be used like `\arraystretch`. The trick is to run `\scriptsize` within a `\settowidth` command which stores the `\baselineskip` of the `\scriptsize` font, multiplied by `\baselinestretch`, in `\normalbaselineskip` which is then used to set `\baselineskip`, `\jot`, etc. Finally, `\arraycolsep` is reduced to better suit the smaller font. Note the use of "%" to prevent unwanted spaces from appearing after the braces at the end of lines in `\mysmallarraydecl`.

3) Tables: Tables, especially those with lines, tend to be a little more complicated. Table VI was made with the following code:

```
\begin{table}
\centering
\caption{Network Delay as a Function of Load}
\label{table_delay}
\begin{IEEEeqnarraybox}[\IEEEeqnarraystrutmode\IEEEe
qnarraystrutsizeadd{2pt}{0pt}]{x/r/Vx/r/v/r/x}
\IEEEeqnarraydblrulerowcut\\
&&&&\IEEEeqnarraymulticol{3}{t}{Average Delay}&\\
&\hfill\raisebox{-3pt}[0pt][0pt]{$\beta$}\hfill&&\IE
EEeqnarraymulticol{5}{h}{}%
\IEEEeqnarraystrutsize{0pt}{0pt}\\
&&&&\hfill\lambda_{\mbox{min}}\hfill&&\hfill
\lambda_{\mbox{max\vphantom{i}}}\hfill&\IEEEeqnarray
strutsizeadd{0pt}{2pt}\\
\IEEEeqnarraydblrulerowcut\\
&1&&&   0.057&& 0.172&\\
&10&&&  0.124&& 0.536&\\
&100&&& 0.830&& 0.905\rlap{\textsuperscript{*}}&\\
\IEEEeqnarraydblrulerowcut\\
&\IEEEeqnarraymulticol{7}{s}{\scriptsize\textsupersc
ript{*}limited usability}%
\end{IEEEeqnarraybox}
\end{table}
```

TABLE VII

POSSIBLE Ω FUNCTIONS

Range	$\Omega(m)$
$x < 0$	$\Omega(m) = \sum_{i=0}^{m} K^{-i}$
$x \geq 0$	$\Omega(m) = \sqrt{m}$

Because this table has lines, the first step is to enable strut mode line spacing. The strut height is then increased by a couple of points to provide a little more headroom above the letters.[24] This table uses cutting horizontal rules and open sides as is commonly done in IEEE publications. There are three extra "x" columns which serve as place holders. The "x" columns at each end serve as a quick way to get the horizontal rules to extend a little past the contents of the table. The middle "x" column serves as an attachment point for the horizontal rule that is below "Average Delay." Without this extra column, the left side of that horizontal rule would cut into the middle double vertical rule.[25] Notice how the "β" is smuggled in as part of the row containing the horizontal rule. β has to be smashed so that it will not add unwanted vertical spacing. Likewise, the strut for that row is disabled. Also, `\raisebox` is used instead of `\smash` so that β can be vertically lowered — otherwise it would appear on its baseline which is too high for the purpose at hand. The `\hfill` on either side of β changes the justification of that cell to centered. The "min" and "max" subscripts would not normally sit at the same level because the "i" in min is slightly higher than the letters in "max". To fix this, a `\vphantom` "i" is added to "max". Because these subscripts sit so low, the depth of that line's strut is increased a couple of points. Alternatively, one could have just smashed the "i". The asterisk next to "0.905" is reduced to zero width via `\rlap` so that it will not affect its cell's width or alignment. This example also illustrates how to integrate table footnotes into the end of a table without the help of external packages.

Strut spacing does not work so well for rows that contain tall symbols because such objects routinely exceed the height of the struts. Furthermore, increasing the strut height is often not an option because (1) the height and depth of the tall symbols must be measured or guessed; and (2) there may be other rows which have normal line height. Table VII illustrates such a situation. Its code is shown here:

---

[24] Knuth calls this extra step a mark of quality.

[25] Some may even think it would be better that way, but we want to show some tricks in these examples.

```
\begin{table}
\centering
\caption{Possible $\Omega$ Functions}
\label{table_omega}
\begin{IEEEeqnarraybox}[\IEEEeqnarraystrutmode\IEEEe
qnarraystrutsizeadd{2pt}{1pt}]{v/c/v/c/v}
\IEEEeqnarrayrulerow\\
&\mbox{Range}&&\Omega(m)&\\
\IEEEeqnarraydblrulerow\\
\IEEEeqnarrayseprow[3pt]\\
&x < 0&&\Omega(m)=\sum\limits_{i=0}^{m}K^{-i}&\IEEEe
qnarraystrutsize{0pt}{0pt}\\
\IEEEeqnarrayseprow[3pt]\\
\IEEEeqnarrayrulerow\\
\IEEEeqnarrayseprow[3pt]\\
&x \ge 0&&\Omega(m)=\sqrt{m}\hfill&\IEEEeqnarraystru
tsize{0pt}{0pt}\\
\IEEEeqnarrayseprow[3pt]\\
\IEEEeqnarrayrulerow
\end{IEEEeqnarraybox}
\end{table}
```

The solution is to use `\IEEEeqnarrayseprow` to manually add in a fixed amount of extra space as needed. In this way, `\IEEEeqnarrayseprow` can do for lined tables what `\jot` does for multiline equations. Of course, using this method, the baselines of the rows will no longer be equally spaced.

The `\hfill` in the square root cell is a cheap, but effective, way of getting the equal signs to line up without the need of additional columns.

ACKNOWLEDGMENT

The author would like to thank Laura Hyslop, Cathy Cardon and Ken Rawson of the IEEE for their help and support in making this work possible. The knowledge and prior work of TeX gurus such as Donald Arseneau, Fred Bartlett, David Carlisle, Tony Liu, Frank Mittelbach, Piet van Oostrum, Roland Winkler and Mark Wooding were instrumental in developing the complex IEEEeqnarray family of commands. The author is also grateful to Peter Wilson and Donald Arseneau for allowing the inclusion of their `\@ifmtarg` command.

Finally, this work might not have been possible had it not been for the efforts of the prior IEEEtran developers: Gerry Murray, Silvano Balemi, Jon Dixion, Peter Nüchter and Juergen von Hagen. Their work still lives on to some degree within IEEEtran.

REFERENCES

[1] M. Shell. (2002) IEEEtran homepage on CTAN. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/supported/IEEEtran/

[2] (2002) The IEEE website. [Online]. Available: http://www.ieee.org/

[3] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[4] R. Fairbairns. (2002) The TeX FAQ. [Online]. Available: http://www.tex.ac.uk/cgi-bin/texfaq2html

[5] A. Gefen, "Simulations of foot stability during gait characteristic of ankle dorsiflexor weakness in the elderly," *IEEE Trans. Neural Syst. Rehab. Eng.*, vol. 9, no. 4, pp. 333–337, Dec. 2001.

[6] D. Arseneau. (2001, Nov.) The cite.sty package. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/supported/cite/

[7] (2000, July) The amsmath.sty package. The American Mathematical Society. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/required/amslatex/math/

[8] M. D. Wooding. (1999, Mar.) The MDW tools package. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/supported/mdwtools/

[9] D. Arseneau. (2002, May) The cases.sty package. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/other/misc/

[10] D. Carlisle. (2001, Sept.) Packages in the 'graphics' bundle. grfguide.ps. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/required/graphics/

[11] K. Reckdahl. (1997, Dec.) Using imported graphics in LaTeX2ε. esplatex.ps or epslatex.pdf. [Online]. Available: http://www.ctan.org/tex-archive/info/

[12] C. Barratt, M. C. Grant, and D. Carlisle. (1998, May) The psfrag.sty package. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/supported/psfrag/

[13] F. Mittelbach and D. Carlisle. (2001, Sept.) The array.sty package. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/required/tools/

[14] D. Arseneau. (1999, May) The threeparttable.sty package. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/other/misc/

[15] S. D. Cochran. (2002, Apr.) The subfigure.sty package. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/supported/subfigure/

[16] S. Tolušis. (1999, Oct.) The stfloats.sty package. Documentation is in the stfloats.sty comments in addition to the presfull.pdf file. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/supported/sttools/

[17] D. Carlisle. (1998, Aug.) The fix2col.sty package. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/supported/carlisle/

[18] M. Shell. (2002) The IEEEtran BibTeX style. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/supported/IEEEtran/bibtex

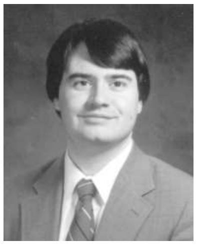
[19] P. W. Daly. (1999, Feb.) The balance.sty package. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/other/preprint/

[20] S. Tolušis. (1997, Oct.) The flushend.sty package. Documentation is in the flushend.sty comments in addition to the presfull.pdf file. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/supported/sttools/

[21] M. Shell. (2002) The testflow diagnostic suite. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/supported/IEEEtran/testflow

[22] D. Arseneau. (1999, Mar.) The url.sty package. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/other/misc/

[23] M. Shell. (2002) The IEEEtrantools.sty package. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/supported/IEEEtran/tools



Michael Shell (M'87) received the B.E.E. and M.S.E.E. degrees in electrical engineering from the Georgia Institute of Technology, Atlanta, in 1991 and 1993, respectively, and is currently working toward the Ph.D. degree. He has developed several all-optical packet-switched network subsystems and node demonstrations and is currently working on new analysis tools and techniques for the optical shared memory architecture class of packet switches. His research interests include all-optical packet-switched networks, high speed opto-electronic interface design, discrete simulation and exact Markov models for buffered packet switches.

Mr. Shell is also the author of the most recent versions of the IEEEtran LaTeX class and BibTeX style packages and is the current maintainer of both.