LaTeX writing standard for students at Aalborg University

Søren Nørgaard

September 6, 2013

Reasoning

As much of the work done as an engineering student at Aalborg University is documented using LaTeX, I found certain guidelines were needed to help structure document typesetting, the same way as a coding standard helps programmers to be able to write good code.

This paper may be used by anyone without any restrictions. If you have any comments or suggestions please contact me at `soerenbnoergaard@gmail.com`.

Contents

1 File and directory names

1.1 Basic folders and files

Directory names are kept at around 3 characters. The root of the SVN drive consists of the basic directories:

```
rep             Documentation/report files.
code            Code developed in the project. Files from
                here may be included from the report!
```

The `rep` folder contains basic folders

```
bib             BibTeX files.
app             Appendix files.
img             Image files.
sec             All included TeX files.
set             Setup files (preamble, macros, etc.)
```

and the files

```
master.tex      Includes all TeX files containing content
                for the report.
master.pdf      Is the output pdf file.
Makefile        Contains rules for compiling the report,
                cleaning and so on.
```

This document does not concern the way the `code` folder is organized.

1.2 Naming conventions

File names are all in lower case letters, and word are separated by underscores (_)! A name should describe the contents of the file as clearly and consistently as possibly, prefixing files that are related with the same label. For example, if some files are concerning an SPI bus, the files could be named `{spi_bus_description.tex, spi_test.tex, spi_results.tex}`. All file names are in English.

2 Makefile

The makefile has dependencies in the `sec` folder, `img` folder, and more. By executing `make` in the folder containing the makefile, the report will be compiled with all references. If, for some reason, `make` does not detect the changes you've made, execute `make force`. To clean the folders from auto generated files, execute `make clean`.

3 Math

3.1 Inline math, single equations, and multiple equations

Inline math is done with dollar signs, `$x = 5$`. A single display equation is written as:

```
\begin{equation}
    f(x) = x^2 + 2
\end{equation}
```

If multiple equations are grouped, they should be aligned – usually by the = sign. This is done with the `align` environment and the `&` sign:

```
\begin{align}
    f(x) &= x^2 + 2 \\
    x    &= 5 \\
    f(x) &= 5^2 + 2 \\
         &=
\end{align}
```

3.2 Units, scientific notation, and comma

Sometimes numbers are easier written in scientific notation. At the same time, if a large number is written, some white space may make it easier on the eyes. To achieve this, write ex.

```
\num{3,53e3}
\num{10000000}
```

To add a unit to a number write ex.

```
\SI{3,35e3}{\kilo\ohm}
\SI{9}{V}
```

To only write the unit, write ex.

```
\si{\ohm}.
```

A comma (,) is always used as decimal marker.

For more on writing units see the `siunitx` package documentation[1].

[1] ftp://ftp.tex.ac.uk/pub//tex/macros/latex/exptl/siunitx/siunitx.pdf

4 Figures, tables, references, and citation

4.1 Image formats

Images follow the same naming conventions as described in Section 1. File formats accepted are (in prioritized order): `pdf, eps, png, jpg`. Vector graphics are highly preferred over bitmap type images.

4.2 LaTeX code for figures and tables

To insert an image/figure:

```
\begin{figure}[H]
    \centering
    \includegraphics{img/<+file+>}
    \caption{<+caption text+>}
    \label{fig:<+label+>}
\end{figure}
```

This way every figure is put exactly where you want it.

To insert a table of variable width:

```
\begin{table}[H]
    \centering
    \begin{tabular}{<+dimensions+>}

    \end{tabular}
    \caption{<+Caption text+>}
    \label{tbl:<+label+>}
\end{table}
```

where dimensions may (basically) be `{l, c, r}`.

To insert a table of *fixed* width:

```
\begin{table}[H]
    \centering
    \begin{tabularx}{<+width+>}{<+dimensions+>}

    \end{tabularx}
    \caption{<+Caption text+>}
    \label{tbl:<+label+>}
\end{table}
```

where dimensions may (basically) be `{l, c, r}` or `X` for long lines that need to wrap into multiple lines. For full page width, width is set to `\linewidth`.

4.3 Label names and referencing

Labels are named/prefixed and references as follows as follows:

```
fig:<+label+>        ... see Figure~\ref{fig:<+label+>}.
tbl:<+label+>        ... see Table~\ref{tbl:<+label+>}.
prt:<+label+>        ... see Part~\ref{prt:<+label+>}.
cha:<+label+>        ... see Chapter~\ref{cha:<+label+>}.
sec:<+label+>        ... see Section~\ref{sec:<+label+>}.
lst:<+label+>        ... see Listing~\ref{lst:<+label+>}.
```

Note that the words like `Figure`, `Table`, etc. are all with a capital starting letter.[2]

4.4 Citation

Citation is done using the `\cite{<+bibentry+>}` command, which outputs a reference to an entry in the reference list. If the command is placed *before* a period (.), the current sentence has a connection with the cited reference. If the command is issued *after* a period, the current/ended paragraph has a connection to the reference.

5 Code

5.1 Inline code

Inline code, like function names used in text, is written like `\texttt{spi\_send()}`. Note that underscores (_) need to be escaped. If some code snippet contains many special characters or is not very long, it may be written as `\verb|$monkey = \<FACE>|`. Here nothing needs be escaped.

5.2 Code blocks/listings, including a code file

Most code blocks can be syntax highlighted in the following way:

```
\begin{lstlisting}[language=<+language+>,caption=<+Listing caption+>,
                   label=lst:<+label+>]
<+CODE+>
\end{lstlisting}
```

[2]Note also, that there is no distinction between sections and subsections. Section types further indented, like subsubsections, are not numbered, and thus, should not be referenced.

where `language` is the language of the code, ex. `{C, VHDL, fortran}` etc.[3]

Please remember not to indent your code like the rest of your LaTeX document, but instead to start indenting from the *first* column in your editor. That way there is no unwanted whitespace columns in the first columns of the code.

To input a source file from a `code` subdirectory:

```
\lstinputlisting[language=<+language+>,caption=<+Listing caption+>,
                 label=lst:<+label+>]{../code/<+subdir+>/<+file+>}
```

[3]For a full list refer to http://en.wikibooks.org/wiki/LaTeX/Source_Code_Listings

TODO

– Top/tail.