# LaTeX in Different Environments

Modularizing your Documents
and Working in Different IDEs

Authors and Organizers:
Ghassan Arnouk
Alec Bales D'Cruze
Aaron English
November 19, 2022

# Contents

# List of Figures

# List of Tables

# 1 Introduction

A running theme through many of these presentation has been the idea of reuseable code. The idea being that once you code something up in a way that you like, say for example your preamble, you can reuse it the next time you're writing a report. But this is code, so we want to simplify this as much as possible. In fact, we don't ever want to copy and paste, we want it to be set up so that if something goes wrong and we fix some issue that fix propagates out everywhere. So how do we do that? LaTeX has a variety of commands and packages that facilitate this sort of thing at different levels as we explore these, keep in mind a couple of different ideas:

- Reusing material from your report in your presentation
- Modularizing documents; especially big documents
- Reusing general code

# 2 LaTeX Macro Expansion

# 3 A (very) Brief Pathing Primer

When setting up a modular document, often times its desireable to have it use relative paths `../../your-file.tex` instead of absolute ones `/home/your-user-name/your/path/to/file.tex`. Figure 1 shows a sample of the directory for this report.

# 4 Handling Paths Yourself

My preferred solution to the pathing issue is to define some variables and simply define the relative distance to the top level in each file independently. This way, no matter which file or subfile I'm using, the correct path is being followed, and only one variable needs to be set for a file.

```
\providecommand{\toplevel}{../..}
\providecommand{\importPath}{\toplevel/Shared/Imports}
\providecommand{\assetPath}{\toplevel/Assets}
\providecommand{\sharedPath}{\toplevel/Shared}

\documentclass[hidelinks, float=false, crop=false]{standalone}

\input{\importPath/preamble}
\input{\importPath/glossary}
\input{\importPath/symbols}
```

Listing 1: Manually solving the pathing issue

# 5 input Command

`\input{you-file.tex}` is the basic form of importing. Requires no external packages, but also doesn't do anything fancy. Grabs the file and drops it in-place.

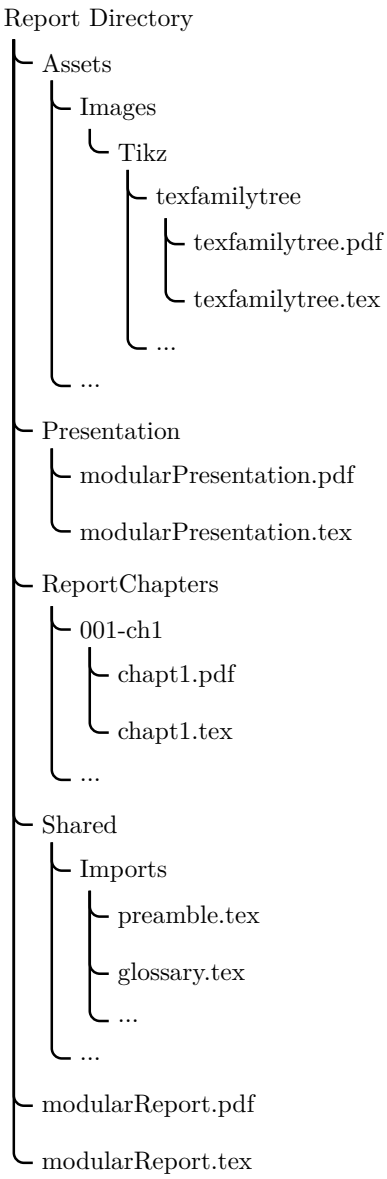


Figure 1: The directory structure for this report (as an example of a modular structure)

- Reusing your preambles (without copying and pasting)
- Reusing glossaries

# 6 import Package

Can aid with pathing, because it separated out the file from its path and so can find files imported from subimports following their source directory.

# 7 standalone Package

The major limitation of the import and input commands is that if they're being use with files you'd like to compile separately (for testing, or for any other reason) having multiple preambles will create break your files. The standalone package is a sophisticated tool that solves this by (optionally) ignoring the preambles of any files you're pulling in (or by combining preambles). This brings us closer to our modularized document because it means TiKz figures can be written separately, compiled, tested and all the rest without having to compile your entire document every time. As an added bonus, it introduces the `\includestandalone[\width=0.5\textwidth]{your-tikz-image}` which as shown, allows you to scale TiKz figures as though the were regular images. Very handy. It also allows you to break your report into multiple chapters.

- Breaking your document into modular parts
- TikZ Images

## 7.1 A Useful Additional Command

When working with a modularized document, it is often desireable to compile it with glossaries or bibliographies added at the end. Of course you can't just add the commands `\printbibliography` because then you would have duplicate bibliographies printed in the super file. My preferred approach to solving this is through the use of boolean flags. Listing 2, gives the code to be added to the preamble that creates a command to conditionally input a bibliography if the boolean flag `standaloneFlag` is set to `true`.

```
% For using the standalone package
\newboolean{standaloneFlag}
\setboolean{standaloneFlag}{true}
% Command to conditionally typeset a bibliography.
\newcommand{\standaloneBib}{%%
\ifthenelse{\boolean{standaloneFlag}}%%
{\printbibliography[heading=bibintoc]
        \printglossary[type=symbols]
        \printglossary[type=acronymtype]
\printglossary[type=main]}{}}
```

Listing 2: Conditionally typeset bibliography and glossaries

In the subfiles, now you can issue the command `\standaloneBib` to have the bibliography and glossaries printed if you in the subfile. In the super file then, issue the command `\setboolean{standaloneFlag}{false}` after loading the preamble, to have all instances of `standaloneBib` blocked, and then use the normal commands to have the bibliography and glossaries typset where you want them.

# 8 catchfilebetweentags Package

## 8.1 catchfilebetweentags

This handy package allows you to store a bunch of different things (say for example, equations) and pull them in to your document. This is really nice because it makes your document's code more easily read. It also makes it that you can reuse the equations in another place, say for example an associated presentations.

You can use it for anything from equations,

$$\begin{aligned}
\Delta E_B &\equiv \sum E_D(Reactants) - \sum E_D(Products) \qquad (1)\\
\Delta E_B &= (E_D(^2\mathrm{H}) + E_D(^2\mathrm{H})) - (E_D(^3\mathrm{He}) + E_D(\mathrm{n}^0))\\
\Delta E_B &= (13.135\mathrm{MeV} + 13.135\mathrm{MeV}) - (14.931\mathrm{MeV} + 8.071\mathrm{MeV})\\
\Delta E_B &= 3.27\mathrm{MeV}
\end{aligned}$$

$$\Delta = \begin{cases} \delta, & \text{for } A \text{ and } N \text{ both even} \\ 0, & \text{for } A + N \text{ odd} \\ -\delta, & \text{for } A \text{ and } N \text{ both odd} \end{cases} \qquad (2)$$

$$E_B(Z,N) = \alpha_1 A - \alpha_2 A^{2/3} - \alpha_3 \frac{Z(Z-1)}{A^{1/3}} - \alpha_4 \frac{(N-Z)^2}{A} + \Delta \qquad (3)$$

to tables,

Table 1: Table of specified parameters and achieved values

| **Parameter** | **Target** | **Calculated** | **Simulated** |
|---|---|---|---|
| $NF_{dsb}$ | $\leq 4$ | 11.17 | 5.95 |
| $IIP3$ | $\geq -22$ | $\geq -2.73$ | $-4.98$ |
| 1 $dB$ Compression | $\geq -32$ | $\geq -12.73$ | $-14.2$ |
| Gain | $\geq 16$ | $\geq -3.26$ | $-4.58$ |
| $I_{bias}$ | | | |
| $I_{buf}$ | | | |
| $I_{ref}$ | | 1 | 1 |
| $R_D$ | $\leq 10$ | 570 | 600 |
| $V_{lo}$ | $\leq 1$ | | |
| $V_{rf}$ | $\leq 1$ | | |

to figures.

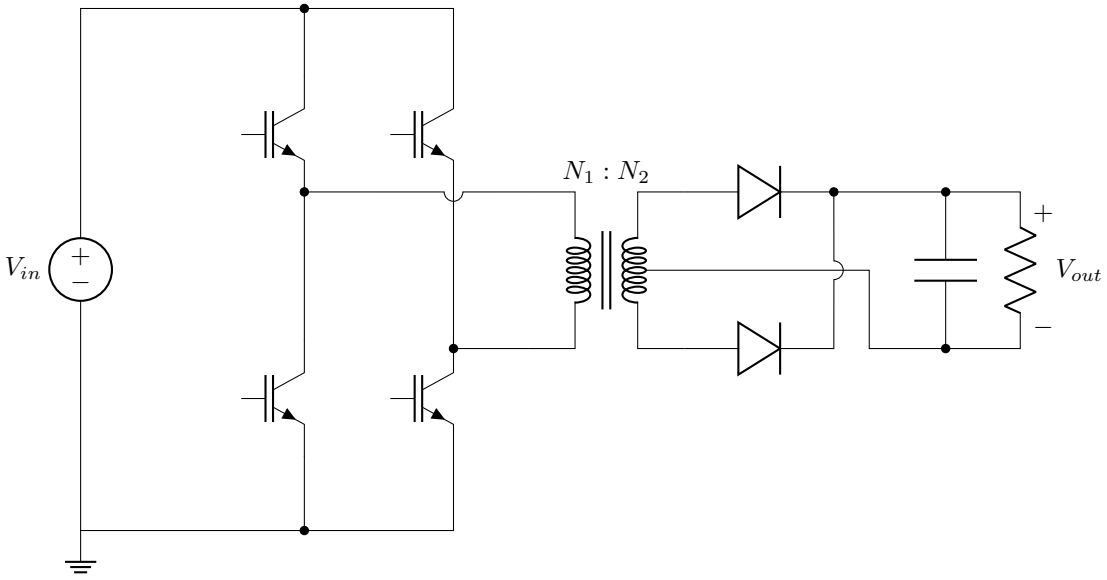


Figure 2: An Example of a circuit (an isolated boost converter) done in circuit TikZ

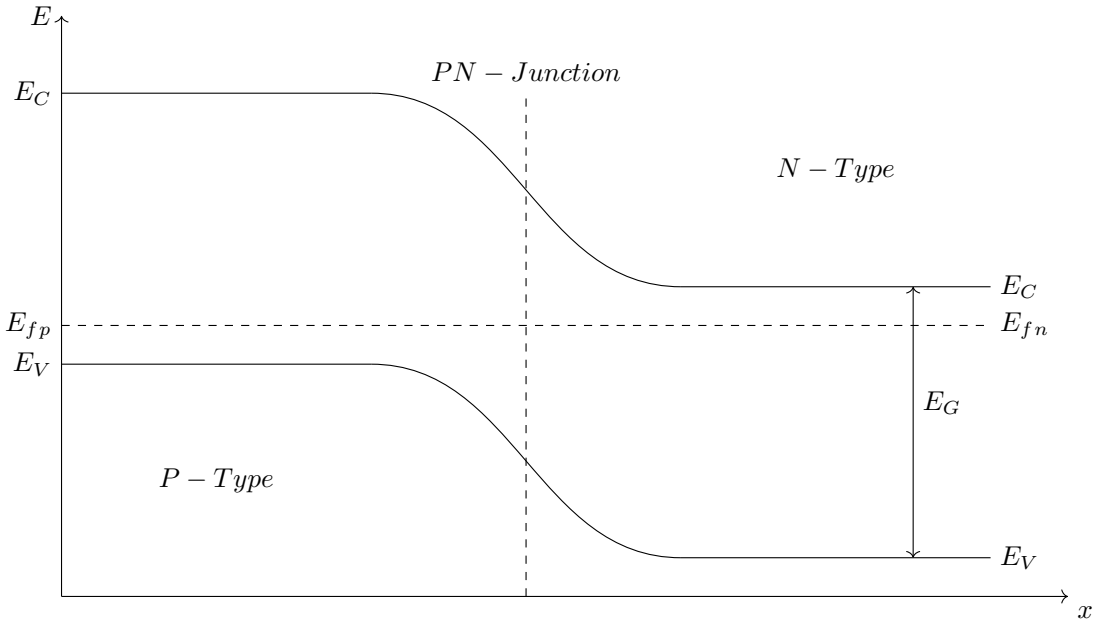


Figure 3: A simple Example TikZ showing the band diagram of a PN-junction

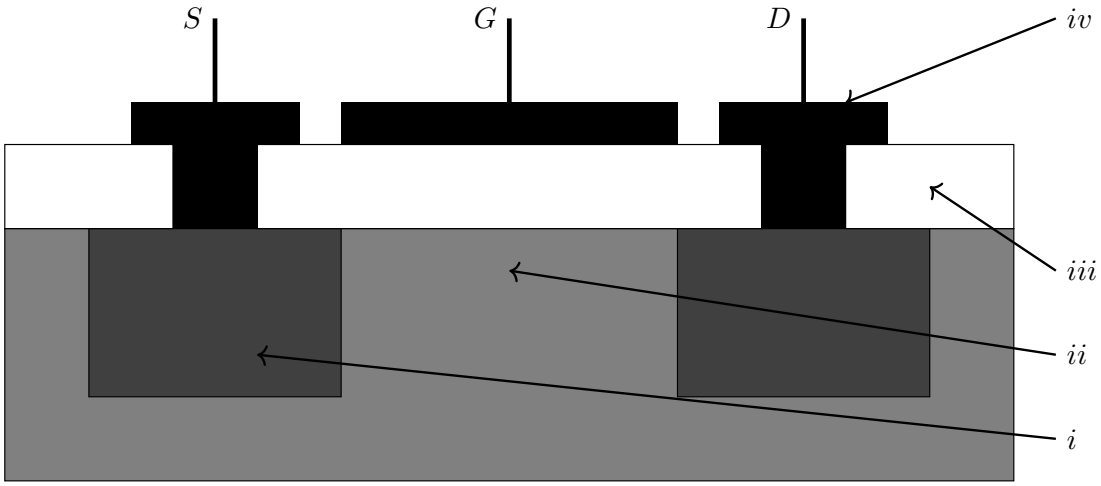


Figure 4: A simple TikZ diagram showing a MOSFET

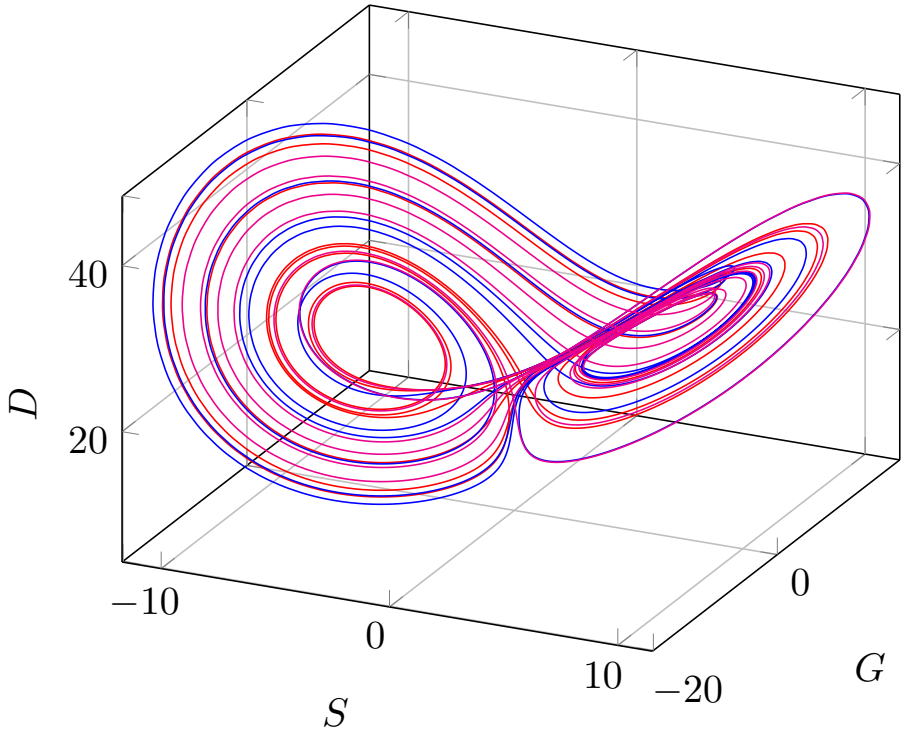


Figure 5: Lorenz Double Scroll Produced in LuaLatex

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \sigma(y - x) \tag{4}$$
$$\frac{\mathrm{d}y}{\mathrm{d}t} = x(\rho - z) - y \tag{5}$$
$$\frac{\mathrm{d}z}{\mathrm{d}t} = xy - \beta z \tag{6}$$
$$\tag{7}$$

Handy!

# 9 The LaTeX Family Tree

The TeX family tree is a large one. You will often see many of these terms thrown around and it can be difficult to make sense of it. The tree shown in Figure **??** shows how each of the major active TeX siblings relate.

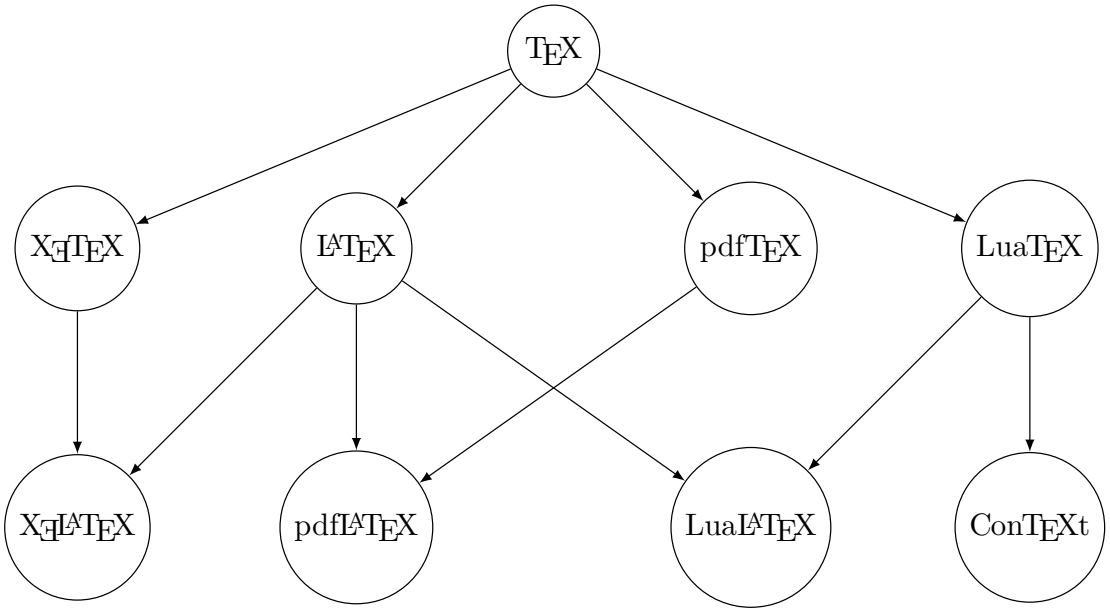


Figure 6: The most commonly used portions of the TeX family tree

## 9.1 TeX

TeX is the original typsetting tool create by Donald Knuth in the 1980s.

## 9.2 PdfTeX

PdfTeX is an extension to the original TeX that enables the creation of PDF files.

## 9.3 LaTeX

As mention TeX primitive are the real way of interacting with the TeX engine, but they are challenging to work with. LaTeX is a collection of macros that ease the use TeX and facilitate writing new packages.

## 9.4 XeTeX

XeTeX is a development on TeX that extends support for languages and glyphs beyond those using just the roman alphabet.

### 9.5 LuaTEX

LuaTEX is a recent and ongoing development which exposes the TEX primitives via the small and fast scripting language Lua. This has many benefits but can be effectively summarized as making it easier to code in TEX.

## 10 Processing TEX Files

### 10.1 Manually Processing Files

Processing a TEX file typically just means running the right programs in a certain order. Because of the sizing, number, and ordering work LATEX is doing during the compilation, it needs to be run at least twice. Typically if our document includes a bibliography (and other elements like glossaries or indices), the compilation procedure would mean executing each of the commands shown in Listing 3. A neat detail of these commands is that they can accept any LATEX commands. This gives the

```
pdflatex -shell-escape -interaction=nonstopmode report
biber report
makeglossaries report
pdflatex -shell-escape -interaction=nonstopmode report
pdflatex -shell-escape -interaction=nonstopmode report
```

Listing 3: Shell commands needed to compile a report directly

possibility of changing variables in a document during the execution (see Listing 4).

```
lualatex -shell-escape -interaction=nonstopmode
↪  "\\providecommand{\\iswhichmode}{draft}\\input{report}"
biber report
makeglossaries report
lualatex -shell-escape -interaction=nonstopmode
↪  "\\providecommand{\\iswhichmode}{draft}\\input{report}"
lualatex -shell-escape -interaction=nonstopmode
↪  "\\providecommand{\\iswhichmode}{final}\\input{report}"
```

Listing 4: Shell commands compiling a document with addition commands provided at compile time

### 10.2 latexmk

latexmk is a tool that tries to automate the latex compilation process by reading in the log files and figuring out what additional programs need to be run, and when to rerun the compilation. It is a very effective tool and in most cases the defacto LATEX compiler.

```
latexmk -pdf report.tex
```

Listing 5: Shell command for compiling with latexmk

### 10.3 arara

Sometimes however, latexmk doesn't know of some additional tool or new intermediate program that needs to be run, and this can create issues. It is also at times trying to be too smart, and ends up creating issues. For these reasons my go-to for a number of years has been ARARA, a compilation tool that allows you to define the compilation process at the beginning of your document. Listing 6 shows the directive syntax for arara, with Listing 7 showing the actual command.

```
% arara: lualatex: { shell: true, interaction: nonstopmode }
% arara: makeglossaries
% arara: biber
% arara: lualatex: { shell: true, interaction: nonstopmode }
% arara: lualatex: { synctex: true, shell: true, interaction: nonstopmode }
```

Listing 6: Shell command for compiling with Arara

```
arara -v report.tex
```

Listing 7: Shell command for compiling with Arara

## 11 TEX Distributions

TEX and LATEX is a very large collection of programs and files, and installing them each individually would be a huge hassle. Thankfully the programs and packages are available bundled together as a single distribution. The distribution of choice where possible is TeXLive. On Linux systems it is always available in the systems package manager. For Mac and Windows systems, there are a few different options because sometimes it can be challenging to get TeXLive to operate (though I have no personal experience with these).

### 11.1 Mac

- TeXLive
- MacTeX

### 11.2 Windows

- TeXLive
- MiKTeX
- ProTeXt

## Acronyms

**CWVM** Cockroft-Walton voltage multiplier. *Glossary:* CWVM

**HV** High Vacuum. *Glossary:* HV

**PIG** Penning Ion Generator. *Glossary:* PIG

**PTFE** Polytetraflouroethylene. *Glossary:* PTFE

## Symbols

$A$ total number of nucleons in nucleus (unitless). 4

$D$ Mosfet Drain. 6

$E_B$ nuclear binding energy (eV). 4

$E_C$ Conduction band energy level. 5

$E_D$ mass defect (eV). 4

$E_G$ Bandgap. 5

$E_V$ Valence band energy level. 5

$E_f$ Fermi Energy of a Material. 5

$E$ Energy. 5

$G$ Mosfet Gate. 6

$N$ number of neutrons in nucleus (unitless). 4

$S$ Mosfet Source. 6

$V_{in}$ Input voltage. 5

$V_{out}$ Output voltage. 5

$Z$ number of protons in nucleus (atomic mass number - unitless). 4

$\Delta E_B$ change in nuclear binding energy i.e. energy released in reaction (eV). 4

$\Delta$ pairing energy parameter (eV). 4

$\beta$ Lorenz Parameter. 7

$\rho$ Lorenz Parameter. 7

$\sigma$ Lorenz Parameter. 7