1 Introduction

This is a template for an undergraduate or master's thesis. The first sections are concerned with the template itself. If this is your first thesis, consider reading Section 1.3.

The structure of this thesis is only an example. Discuss with your adviser what structure fits best for your thesis.

1.1 Template Structure

- To compile the document either run the makefile or run your compiler on the file 'thesis_main.tex'. The included makefile requires latexmk which automatically runs bibtex and recompiles your thesis as often as needed. Also it automatically places all output files (aux, bbl, ...) in the folder 'out'. As the pdf also goes in there, the makefile copies the pdf file to the parent folder. There is also a makefile in the chapters folder, to ensure you can also compile from this directory.

- The file 'setup.tex' includes the packages and defines commands. For more details see Section 1.2.

- Each chapter goes into a separate document, the files can be found in the folder chapters.

- The bib folder contains the .bib files, I'd suggest to create multiple bib files for different topics. If you add some or rename the existing ones, don't forget to also change this in thesis_main.tex. You can then cite as usual [?, ?, ?].

- The template is written in a way that eases the switch from scrbook to book class. So if you're not a fan of KOMA you can just replace the documentclass in the main file. The only thing that needs to be changed in setup.tex is the caption styling, see the comments there.

1.2 setup.tex

Edit setup.tex according to your needs. The file contains two sections, one for package includes, and one for defining commands. At the end of the includes and commands there is a section that can safely be removed if you don't need algorithms or tikz. Also don't forget to adapt the pdf hypersetup!!
setup.tex defines:

- some new commands for remembering to do stuff:
  - `\todo{Do this!}`: **(TODO: Do this!)**
  - `\extend{Write more when new results are out!}`: **(EXTEND: Write more when new results are out!)**
  - `\draft{Hacky text!}`: **(DRAFT: Hacky text!)**
- some commands for referencing, 'in `\chapref{chap:introduction}`' produces 'in Chapter 1'
  - `\chapref{}`
  - `\secref{sec:XY}`

  - `\eqref{}`
  - `\figref{}`
  - `\tabref{}`

- the colors of the Uni's corporate design, accessible with `{\color{UniX} Colored Text}`
  - UniBlue
  - UniRed
  - UniGrey

- a command for naming matrices `\mat{G}`, $\mathbf{G}$, and naming vectors `\vec{a}`, $\mathbf{a}$. This overwrites the default behavior of having an arrow over vectors, sticking to the naming conventions normal font for scalars, bold-lowercase for vectors, and bold-uppercase for matrices.

- named equations:

```
\begin{align}
    d(a,b) &= d(b,a)\\ \eqname{symmetry}
\end{align}
```

$$d(a,b) = d(b,a) \tag{1}$$

symmetry

1.3 Advice

This section gives some advice how to write a thesis ranging from writing style to formatting. To be sure, ask your advisor about his/her preferences.
For a more complete list we recommend to read Donald Knuth's paper on mathematical writing. (At least the first paragraph). `http://jmlr.csail.mit.edu/reviewing-papers/knuth_mathematical_writing.pdf`

- If you use formulae pay close attention to be consistent throughout the thesis!
- In a thesis you don't write 'In [24] the data is..'. You have more space than in a paper, so write 'AuthorXY et al. prepare the data... [24]'. Also pay attention to the placement: The citation is at the end of the sentence before the full stop with a no-break space. `... last word~\cite{XY}.`
- Pay attention to comma usage, there is a big difference between English and German. '...the fact that bla...' etc.
- Do not write 'don't ', 'can't' etc. Write 'do not', 'can not'.
- If an equation is at the end of a sentence, add a full stop. If it's not the end, add a comma: $a = b + c \quad (1),$
- Avoid footnotes if possible.
- Use ` `` '' ` for citing, not `""`.
- It's important to look for spelling mistakes in your thesis. There are also tools like aspell that can help you find such mistakes. This is never an excuse not to properly read your thesis again, but it can help. You can find an introduction under `https://git.fachschaft.tf/fachschaft/aspell`.

- If have things like a graph or any other drawings consider using tikz, if you need function graphs or diagrams consider using pgfplots. This has the advantage that the style will be more consistent (same font, formatting options etc.) than when you use some external program.

- Discuss with your advisor whether to use passive voice or not. In most computer science papers passive voice is avoided. It's harder to read, more likely to produce errors, and most of the times less precise. Of course there are situations where the passive voice fits but in scientific papers they are rare. Compare the sentence: 'We created the wheel to solve this.' to 'The wheel was created to solve this', you don't know who did it, making it harder to understand what is your contribution and what is not.

- In tables avoid vertical lines, keep them clean and neat. See **??** for an example. More details can be found in the 'Small Guide to Making Nice Tables' `https://www.inf.ethz.ch/personal/markusp/teaching/guides/guide-tables.pdf`

Bachelor Thesis

Benchmark of RISC-V in BTOR2

Jan Krister Möller

Examiner: Dr. Mathias Fleury

University of Freiburg

Faculty of Engineering

Department of Computer Science

Chair of Computer Architecture

August 15, 2025

Writing Period

24. 06. 2025 – 24. 09. 2025

Examiner

Dr. Mathias Fleury

Declaration

I hereby declare that I am the sole author and composer of my thesis and that no other sources or learning aids, other than those listed, have been used. Furthermore, I declare that I have acknowledged the work of others by providing detailed references of said work.
I hereby also declare that my Thesis has not been prepared for another examination or assignment, either wholly or excerpts thereof.

Place, Date	Signature

Abstract

foo bar [1] [2] [3]

Contents

List of Figures

List of Tables

List of Algorithms

2 Motivation

This is a template for an undergraduate or master's thesis. The first sections are concerned with the template itself. If this is your first thesis, consider reading.

3 RISC-V

3.1 Overview

RISC-V is an open source instruction set architecture first published in May 2011 by A. Waterman et al. [4]. As contained in the name, it is based on the RISC design philosophy. **(TODO: explain risc (compare wiki))** Since 2015 the developement of RISC-V is coordinated by the RISC-V International Association, a non-profit corporation based in switzerland since 2020 [5]. Its goals are among others an *open* ISA that is freely available to all, a *real* ISA suitable for native hardware implementation and an ISA separated into a *small* base integer ISA usable by itself e.g. for educational purpose and optional standard extensions to support general purpose software development [1](Chapter 1).

It currently contains four base ISAs, namely RV32I, RV64I, RV32E and RV64E, wich may be extended with one or more of the 47 ratified extension ISAs [1] (Preface).

(EXTEND: Hier brauchts vlt noch was) **(TODO: little endian erwähnen?)**

For my work, I will focus on a subset of the RV64I ISA.

3.2 The RV64I ISA

RV64I is not complex, but its structure is relevant to understand my later work. So I explain all elements relevant for my thesis.

RV64I has 32 64bit registers called $x0$-$x31$, where $x0$ is hardwired to 0 on all bits. Registers $x1$-$x31$ are general purpose and are interpreted by various instructions as a collection of booleans, two's complement signed binary integers or unsigned integers. Additionaly there is a register called *pc* acting as program counter and holding the address of the current instruction [1](Chapters 4.1, 2.1).

In RV64I, a memory address has the size of 64bit. As the memory model is defined to be single byte adressable, the address space of RV64I is 2^{64} bytes [1](Chapter 1.4).

Like almost all of the standard ISAs of RISC-V, RV64I has a standart encoding lenght of 32bit or 1 *word*. Only the compressed extension C adds instuctions with a length of 16bit [1](Chapter 1.5), but it is irrelevant for us. All instructions of RV64I are encoded in one of the six formats shown in Figure 1.
The design of these format results in the following features:

- As of RISC-Vs little endianess, the *opcode*, wich encodes the general instruction, is always read first. Also, further specification of the instruction by $funct3$ and $funct7$ is found always at the same location.

- If used by the instruction, the destination register rd and the source registers $rs1$ & $rs2$ are always at the same place. This simplifys decoding.

- The highest bit of the immediate value imm is always at the last bit. This makes finding the sign of a signed immediate value trivial.

3.3 Simulation of RISC-V

3.3.1 Saving the State of a RISC-V Processor

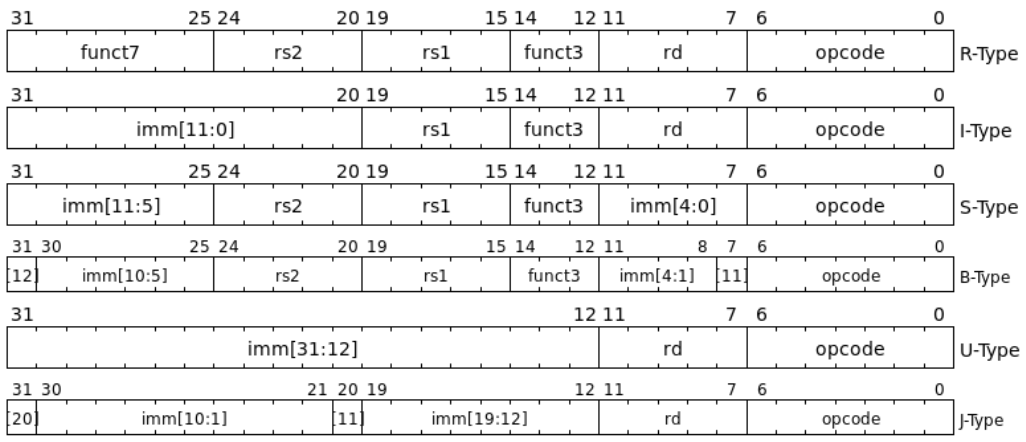
31 25 24 20 19 15 14 12 11 7 6 0
funct7 rs2 rs1 funct3 rd opcode R-Type
31 20 19 15 14 12 11 7 6 0
imm[11:0] rs1 funct3 rd opcode I-Type
31 25 24 20 19 15 14 12 11 7 6 0
imm[11:5] rs2 rs1 funct3 imm[4:0] opcode S-Type
31 30 25 24 20 19 15 14 12 11 8 7 6 0
[12] imm[10:5] rs2 rs1 funct3 imm[4:1] [11] opcode B-Type
31 12 11 7 6 0
imm[31:12] rd opcode U-Type
31 30 21 20 19 12 11 7 6 0
[20] imm[10:1] [11] imm[19:12] rd opcode J-Type

Figure 1: RV64I encoding formats, used in [1](Chapter 2.3) **(TODO: Kopie richtig angeben)**

4 BTOR2

4.1 Model Checking

4.2 The BTOR2 Language

4.3 The BTOR2 Witness

5 Transforming RISC-V to BTOR2

5.1 The Concept

4 3.3.1 Section 3.3.1

5.2 Encoding

5.2.1 Constants

5.2.2 State Representation

5.2.3 Initialization

5.2.4 Computing values

Opcode

funct3 & funct7

Registers

Immediate

5.2.5 Command Detection

5.2.6 Next-State-Logic

5.2.7 Constraints

5.3 Testing for Correctness

5.3.1 State Fuzzer

5.3.2 Automated Logging

5.4 Functional vs Relational Next-State-Logic

6 Benchmarks

6.1 MultiAdd in Functional and Relational Next-State-Logic

6.2 Memory Operations

6.3 Results

Bibliography

[1] *The RISC-V Instruction Set Manual Volume I: Unprivileged ISA*, 2025, version 20250508. [Online]. Available: https://lf-riscv.atlassian.net/wiki/spaces/HOME/pages/16154769/RISC-V+Technical+Specifications

[2] A. Niemetz, M. Preiner, C. Wolf, and A. Biere, "Btor2 , BtorMC and Boolector 3.0," in *Computer Aided Verification*, H. Chockler and G. Weissenbacher, Eds. Cham: Springer International Publishing, 2018, pp. 587–595.

[3] F. Schrögendorfer, "Bounded Model Checking of Lockless Programs," Master's thesis, Johannes Kepler University Linz, August 2021. [Online]. Available: https://epub.jku.at/obvulihs/download/pdf/6579523

[4] A. Waterman, Y. Lee, D. A. Patterson, and K. Asanović, "The risc-v instruction set manual, volume i: Base user-level isa," UC Berkeley, Tech. Rep. UCB/EECS-2011-62, May 2011. [Online]. Available: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2011/EECS-2011-62.html

[5] "History of RISC-V," https://riscv.org/about/, accessed: 15.08.2025.