RISC-V processors with Bluespec

High Performance Processors and System
A.Y. 2020/2021

Riccardo Nannini

June 15, 2021

Tutors: Emanuele Del Sozzo,
Davide Conficconi

Professor: Marco Domenico Santambrogio

Contents

Abstract

Bluespec System Verilog (**BSV**) is a state-of-the-art Hardware Description Language. Bluespec compilation toolchain (**BSC**) has been recently released as open source [**bsc**]. The goal of the project was investigating the potentiality of said toolchain implementing different **RISC-V** processors of increasing complexity.

1 Introduction

This report covers chronologically the path that I have followed during the development of this project.

It starts from a quick overview on the *RISC-V* ISA, focused on the key ingredients that make this ISA one of the most trend topics in Computer Engineering.

It proceeds with the analysis of the *Bluespec System Verilog* language, outlining its novelties with respect to other hardware description languages, as well as its main features and capabilities.

The last section is devoted to the development of various RISC-V processor designs, describing in details the characteristics of each processor and motivating the various design choices, starting from a one cycle non pipelined processor and ending with a 6 stage pipelined one enriched with multiple branch predictors.

2 RISC-V

In the following, Equation (1) shows an example of equation centered within the page.

$$\text{maximize} \sum_{i=17}^{31} \sum_{j=i+1}^{32} [x_{i,j} \times s_{i,j} + (1 - x_{i,j}) \times d_{i,j}] \tag{1}$$

To type any mathematical expression in the text without breaking the line, you can surround it with the $ symbol, for example to refer to i and to $\sum_{j=i+1}^{32}[x_{i,j} \times s_{i,j} + (1 - x_{i,j}) \times d_{i,j}]$.

If you need to show code snippets, you can use the *listing* environment, as in the following example. As for the other elements, you can refer to a listing through its label as in algorithm 1. Remember to make your code well readable by indenting it and using concise pseudo-code snippets, without pasting your own code as it is (unless it is REALLY expressive and short).

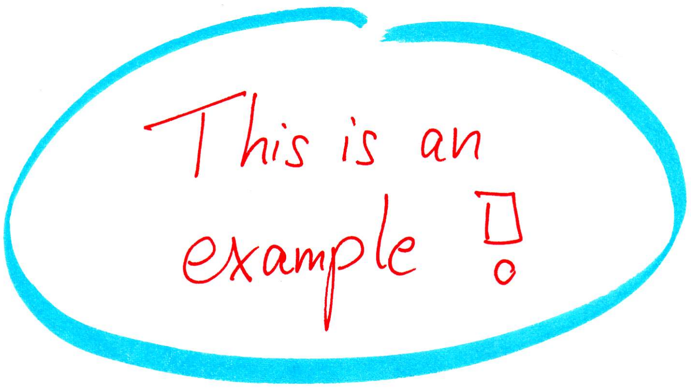

Figure 1: Example caption.

Algorithm 1 Example of code snippet

```
globaldata: list_head buddies[MAX_ORDER][MAX_COLORS]

procedure InsertBuddy(buddy b, order ord)
  buddy twin
  mcolor mcol

  mcol = Mcolor(b, ord)
  twin = GetTwinBuddy(b,ord)
  if ord < MAX_ORDER-1 AND BuddyIsFree(twin)
    RemoveFromList(buddies[ord][Mcolor(twin,ord)])
    b = CoalesceBuddy(b, twin, ord)
    InsertBuddy(b, ord+1)
    return
  else
    InsertHead(buddies[ord][mcol],b)
end procedure
```

2.1 Subsection 1

This is the way to refer to Figure 1, and similarly for Section 2. You will notice LaTeX freely moves elements like figures and tables around the page, and often in the pages around the current paragraph. In particular, LaTeX always places these elements at the bottom or top of the page (otherwise instructed): this choice obeys to the main typesetting guidelines, and should work well most of the times. You should not force a specific position for these ele-

Table 1: Table title (without stop!)

label0	**label1**	**label2**
row 0, col 0	row 0, col 1	row 0, col 2
row 1, col 0	row 1, col 1	row 1, col 2
row 2, col 0	row 2, col 1	row 2, col 2
row 3, col 0	row 3, col 1	row 3, col 2

Table 2: Table title (without stop!)

label0	**label1**	**label2**
row 0, col 0	row 0, col 1	row 0, col 2
row 1, col 0	row 1, col 1	row 1, col 2
row 2, col 0	row 2, col 1	row 2, col 2
row 3, col 0	row 3, col 1	row 3, col 2

ments, and keep in mind that *LaTeX most of the time is right* (it is its job to do lay out elements, not yours). If you need to move an element, move its LaTeX code up or down.

2.2 Subsection 2

Table 1 provides an example of a table. According to many people, this table style (without vertical lines separating columns) is the most elegant and clean possible; to set this tables style, this document adds the `\usepackage{booktabs}` directive at the beginning. In the LaTeX code, you can notice that an ampersand ($) separates columns and a double backslash (\\) moves to a new line.

Since tables in LaTeX are verbose, you should:

- place them in a specific file, to be included with a `\input{filename}` directive
- for large tables, fill them on applications or websites like `https://www.tablesgenerator.com/`, then copy their code and paste it in the dedicated file; finally, you can customize the style from the LaTeX code

Here you can see the same table as before but included from an external file *table.tex*: the result is the same.