System Verilog Style Guide

Benjamin Davis, allrisc.dev@gmail.com

2023-03-04

Contents

Chapter 1

Summary

Robert C. Martin, author of *Clean Code: A Handbook of Agile Software Craftsmanship*, famously said:

> Indeed, the ratio of time spent reading versus writing is well over 10 to 1. We are constantly reading old code as part of the effort to write new code. ... [Therefore,] making it easy to read makes it easier to write.

This quote certainly holds true whether the code is traditional software, RTL, or Verification frameworks. Thus this document puts forward a set of rules and guidelines for writing easy to understand and consistent SystemVerilog.

A vast majority of these rules apply to SystemVerilog for synthesis and for verification. When this is not the case it will be noted and further guidance shall be provided.

While these are referred to as rules, they may not be the correct solution for all code-bases. When an instance arises where a developer feels that there is a method of writing the code which is clearer they should be empowered to do so. But, in general these moments are few and far-between, so the developer is asked to think critically about whether this approach is truly better.

1.1 Terminology

Unless otherwise stated, the following terminology conventions apply to the guidance layed out in this document:

- The word **must** indicates a mandatory requirement.
- Similarly, **do not** indcations a strict prohibition.
- The word **recommended** indicates a strong suggestion that should be applied, unless the implications and reasons for taking an alternative approach are completely understood.
- The word **may** indicates a course of action is permitted and optional.
- The word **can** indicates a course of action is possible given material, physical, or causal constraints.

1.2 General

The following general style constraints must be adhered:

- Generally, names should be descriptive and avoid abbreviations.
- Non-ASCII characters are forbidden.
- Indentation uses spaces, no tabs. Indentation is two spaces for nesting.
- Place a space between `if` and the parenthesis in conditional expressions.
- Use horizontal whitespace around operators, and avoid trailing whitespace at the end of lines.

1.3 Which Verilog to Use

Prefer SystemVerilog-2017.

All RTL and tests should be developed in SystemVerilog, following the IEEE 1800-2017 (SystemVerilog-2017) standard, except for prohibited features.

The standards document is available free of cost through IEEE GET (a registration is required).

Chapter 2

SystemVerilog Conventions

This chapter seeks to standardize asthetic aspects of SystemVerilog.

2.1 General File Rules

2.1.1 File Extensions

2.1.2 Characters

2.1.3 Line Length

2.1.4 No Tabs

2.1.5 Trailing Whitespace

2.2 Begin and End

2.3 Indentation

2.4 Spacing

2.5 Parentheses

2.6 Ternary Expressions

2.7 Comments

2.8 Declarations

Chapter 3

Naming

Chapter 4

Language Features

4.1 Problematic Language Features

Chapter 5

Design Conventions

Appendix A

Condensed Style Guide

Appendix B

Basic Template