SystemVerilog Tools in Python

Michał Czyż
PG ETI UE CHIP

The Other Dude
His Company / University

September 2, 2018

Contents

1 Introduction **2**

2 Acceptable constructs in SystemVerilog. **2**
2.1 Module declaration. 2
2.2 Module instantiation. 2
2.3 Interface declaration. 2

3 Usage. **2**

Abstract

Documentation for SystemVerilog Tools project available at:

`https:www.github.com/emcezet/sv-tools`

1 Introduction

The goal of project is to develop a SystemVerilog (SV) parsing engine and provide scripts for simple module generation, instantiation in both named ports and interface modports styles.

2 Acceptable constructs in SystemVerilog.

2.1 Module declaration.

The basic building block of SystemVerilog language is a module. Any SV module consists of a name, a list of parameters and a list of ports. An example is provided below:

```
`define DEFINED_IN_A_MACRO 8
`define DEFINED_IN_ANOTHER_MACRO 16

module basic_module#(
    // parameters
    parameter paramOne = 8,
    parameter paramTwo = `DEFINED_IN_A_MACRO,
    parameter paramThree = paramOne * paramTwo
    )(
    // IN
    input clk,
    input arst,
    input [7:0] d1,
    // INOUT
    inout [paramOne:0] d2,
    // OUT
    output [0:`DEFINED_IN_ANOTHER_MACRO] d3,
    clk_if.sink clkif
    );

/*
 * Module body.
 *
*/
```

endmodule

Keywords **module** and **endmodule** are always at the start and the end of the module. After the keyword **module** a module name is given. It is optional to end the module declaration with **endmodule : module_name**. The # sign marks the beginning of list of parameters, which is comma separated. List of parameters is optional. An empty list is also accepted.

Port list is also a comma separated list of entries. Each port requires direction and name. There are three supported directions: **input**, **output**, **inout**. Bus signals have width encoded in square brackets. Interfaces are used via modports. The declaration is **interface_name.modport_name port_name.**

SV supports C-style comments and macros. The only difference is that used macro requires a tick. Whitespaces are ignored.

2.2 Module instantiation.

2.3 Interface declaration.

3 Usage.

All scripts support -h and –help parameters for usage help.