ECSE 425: Computer Organization and Architecture
VHDL Refresher: Finite State Machines

Due January 22, 2018, 11:59 PM

Introduction

The goal of this deliverable is to build and test a finite-state machine to identify the commented characters in C code. Your finite-state machine will have the following ports:

- clk : in std_logic;
- reset : in std_logic;
- input : in std_logic_vector(7 downto 0);
- output : out std_logic

You will feed one ASCII character per clock cycle to your FSM and will get '0' if the input text is not part of a comment and '1' if it is.

Change Log

- Any changes will be reported in this log

Example

In the following example, the characters where the output should be one is highlighted in green.

```
/*This is the\n
main method*/\n
int main()\n
{\n
                                //this is a comment\n
                                        return 33;\n
}\n
```

Note that the '\n' represent the new-line character (ASCII 10). The exit sequence for the comment ('\n' or "*/") is considered a comment while the opening sequence ("//" or "/*") is not. Thus, the output should be equal to 1 in the clock cycle after the second character of the opening sequence appears, and the output should be equal to 0 in the clock cycle after the second character of the exit sequence appears. For example:

```
//comment\n code

001111111 1 0000
```

Where to start

Three files are provided to you for this assignment:

- `fsm.vhd`: You will implement your FSM in this file. Do not change the port map (entity).
- `fsm_tb.vhd`: You will implement a complete testbench for your fsm in this file.
- `fsm_tb.tcl`: You don't have to edit this file. It is a script to compile and run your testbench.

To compile, open ModelSim and change the directory (File -> Change Directory) to the one containing those three files. In the Transcript section (ModelSim console), run the following command.

```
source fsm_tb.tcl
```

If you don't have compilation errors, you should see the waves appear in the Wave section and your test results in the Transcript section.

Grading

Two aspects of your deliverable will be evaluated: (a) the correctness of your implementation, as evaluated by our testbench, and (b) the completeness of your testbench, with respect to test coverage.

In this case of this deliverable, test coverage is determined by the fraction of finite state machine transitions that have been tested.

Hand In Procedure

Hand in, via MyCourses, in a single ZIP file:

- `fsm.vhd`
- `fsm_tb.vhd`