CprE 381 Toolflow manual

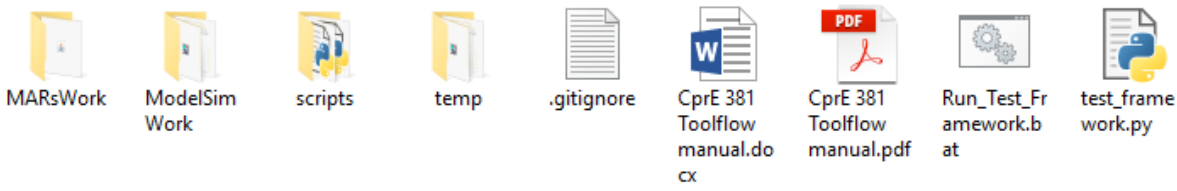
Contents

Testing Framework

The goal of the test framework is to allow students to compare the output of their processor with that of MARS.

Getting Started

Opening the test framework, you should have the following files present.



1. Copy all the source files from your processor into the ModelSimWork/src. There should already be a file named tb_SimplifiedMIPSProcessor.vhd, you do not need to edit this file, and it should not be removed
2. Double-click on Run_Test_Framework.bat.
3. You will be prompted for the path to a MIPS assembly file to run. Example programs are provided in MarsWork/Examples, you may however provide the path to any assembly file you like.
4. The framework should now compile, simulate, and show you the differences between your processor's output in the expected file printed to the command line

Troubleshooting and FAQ

Q: I defined some types in a separate file, so now the program won't compile unless that file is compiled first. How do I fix this?

A: First run the program to find the first file where the error occurs:

```
** Error: (vcom-11) Could not find work.mytypes.
** Error (suppressible): ModelSimWork/src/project2alu.vhd(4): (vcom-1195) Cannot find expanded name "work.mytypes".
** Error: ModelSimWork/src/project2alu.vhd(4): Unknown expanded name.
** Error: ModelSimWork/src/project2alu.vhd(7): VHDL Compiler exiting
End time: 10:32:47 on Feb 13,2019, Elapsed time: 0:00:01
```

Then copy the entire contents of the VHDL file with the types onto the top the file with the error. Delete the types file:

```
library ieee;
use ieee.std_logic_1164.all;

package mytypes is
    type vector_32_array is array (0 to 31) of std_logic_vector(31 downto 0);
end mytypes;

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;
use work.mytypes.all;

entity project2alu is
    port(
      ...
```

Options

The test framework has several flags that can be used to streamline your test framework experience. These flags can be added to line 25 of Run_Test_Framework.bat

Example:

```
%python_path% test_framework.py --nocompile --asm-file aaa/bbb.asm
```

Options:

Command	Description
--asm-file	Accepts a unix-style path to an assembly file to run. If the provided file cannot be simulated the script prompts the user anyways.
--nocompile	Skips recompilation to save time when running multiple assembly files without changing the processor.
--max-mismatches	Accepts a natural number which is the number of mismatches between the expected output and your output before the script stops. Default = 5
--sim-timeout	Accepts a natural number which is the number of seconds before the simulation is interrupted. Default=30

Portability

This framework is only supported on the Windows lab computers and ISU VDI ECpE Student.