

Verilog simulation with Modelsim

Oprițoiu Flavius
flavius.opritoiu@cs.upt.ro

October 3, 2022

Verilog simulation with the modelsim environment

Objectives:

- ▶ Compile Verilog design with Modelsim
- ▶ Simulate a Verilog design using the Modelsim environment

Windows/Linux modelsim installers are available [▶ here](#).

Note: an Intel account is needed for downloading the installers.

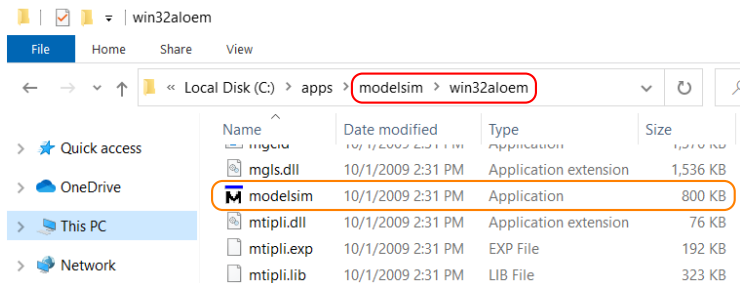
In these guides, the "Modelsim Altera Starter Edition 6.5b" version is used. This edition does not require a license.

Note: the Modelsim environment includes its own source file editor.

Portable modelsim installation for Microsoft Windows

A portable version of the modelsim environment, running on Microsoft Windows OS-es, can be downloaded from [here](#).

After extracting the archive, the Modelsim environment can be started by running "modelsim\win32aloem\modelsim.exe".



Docker-based modelsim instalation for macOS

A Docker image of the modelsim environment was built to be used on macOS computers. The procedure outlined in the following slides was tested on macOS Catalina, version 10.15.7.

Step 1: Install Docker Desktop for macOS as described [▶ here](#).

Step 2: Load the Modelsim Docker image:

- i. Start Docker Desktop for macOS
- ii. Download the archived image from [▶ here](#)
- iii. Open the Terminal app and, assuming the archive was saved in ~/Downloads, run the following commands:

```
cd ~/Downloads  
tar xJf ./msim_dking.tar.xz  
docker load -i ./modelsim_v6.5.tar
```
- iv. (optional) Files ./msim_dking.tar.xz and ./modelsim_v6.5.tar can be deleted

Docker-based modelsim instalation for macOS (contd.)

Step 3: Install and configure XQuartz:

- i. Install XQuartz as described [▶ here](#).

Note: accept installation's final request to log out

- ii. Start the XQuartz app. In the newly opened window (titled "xterm") go to `XQuartz menu` `Preferences` and in the Security tab make sure "Allow connections from network clients" option is enabled
- iii. Restart the computer

Step 4: Download the `start_modelsim_macos.command` script.

Note: Script can be placed anywhere (e.g. on Desktop, for convenience)

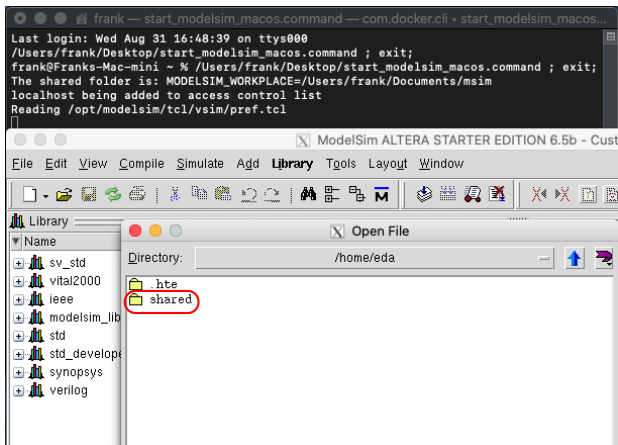
Step 5: Configure the folder to be shared with the Modelsim environment: open the script in a text editor (e.g. TextEdit) and, in line 3, change variable `MODELSIM_WORKPLACE` to indicate to a folder to be shared.

Important: The shared folder is accessible in the Modelsim environment at the `/home/eda/shared` path.

Docker-based modelsim instalation for macOS (contd.)

For starting the Modelsim environment on macOS:

- i. Launch Docker Desktop for macOS
- ii. Launch the `start_modelsim_macos.command` script
- iii. The shared content is accessible through **File** > **Open**, by selecting the "shared" folder



Portable modelsim installation for Ubuntu

A portable version of the Modelsim environment, running on Ubuntu OS, can be downloaded from [▶ here](#). This environment was tested on 64-bit Ubuntu 20.04 LTS and 22.04 LTS OS-es.

For 64-bit Ubuntu OS-es, the package's required dependencies can be installed with the following commands:

```
sudo dpkg --add-architecture i386
sudo apt-get update
sudo apt-get install libc6:i386 libncurses5:i386
sudo apt-get install libx11-6:i386 libxtst6:i386
sudo apt-get install libstdc++6:i386
```

The above commands are included in the "INSTALL" file from the root folder of environment's archive.

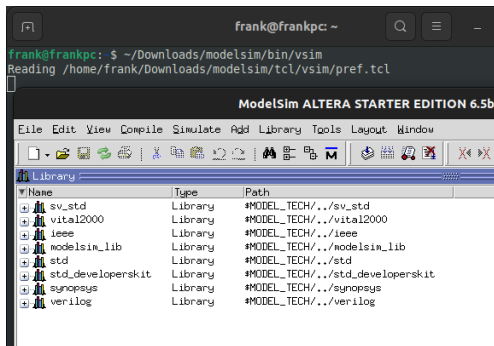
Portable modelsim installation for Ubuntu (contd.)

Assuming the archive was saved in `~/Downloads`, extract it with:

```
cd ~/Downloads && tar xJf ./modelsim.tar.xz
```

The Modelsim environment is started with:

```
~/Downloads/modelsim/bin/vsim
```



It might be possible to adapt this portable installation to other Linux distributions. Otherwise, a Docker-based Linux installation, presented in the next slides, could be used.

Docker-based modelsim installation for Linux

A Docker image of the modelsim environment was built to be used on Linux computers. The procedure outlined in the following slides was tested on 64-bit Ubuntu 22.04 LTS.


Step 1: Install Docker Engine for your distribution: [▶ link](#).

Step 2: Load the Modelsim Docker image:

- i. Download the archived image from [▶ here](#)
- ii. Open a terminal and, assuming the archive was saved in `~/Downloads`, run the following commands:

```
cd ~/Downloads  
tar xJf ./msim_dking.tar.xz  
sudo docker load -i ./modelsim_v6.5.tar
```
- iii. (optional) Files `./msim_dking.tar.xz` and `./modelsim_v6.5.tar` can be deleted

Docker-based modelsim installation for Linux (contd.)

Step 3: Download the `start_modelsim_linux.sh` . Assuming the script was saved in `~/Downloads`, mark it as executable:

```
chmod +x ~/Downloads/start_modelsim_linux.sh
```

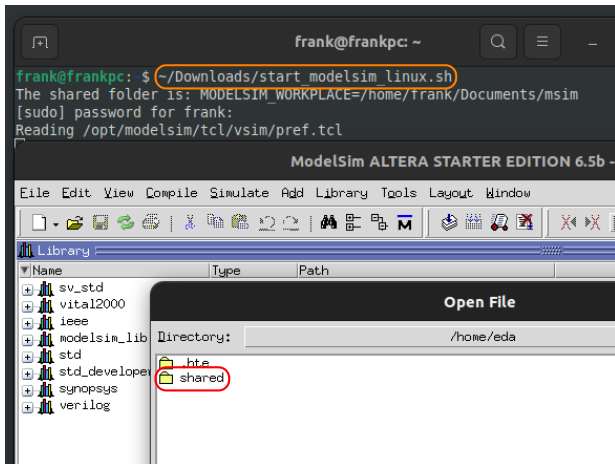
Step 4: Configure the folder to be shared with the Modelsim environment: open the script in a text editor (e.g. gedit) and, in line 3, change variable `MODELSIM_WORKPLACE` to indicate to a folder to be shared.

Important: The shared folder is accessible in the Modelsim environment at the `/home/eda/shared` path.

Docker-based modelsim installation for Linux (contd.)

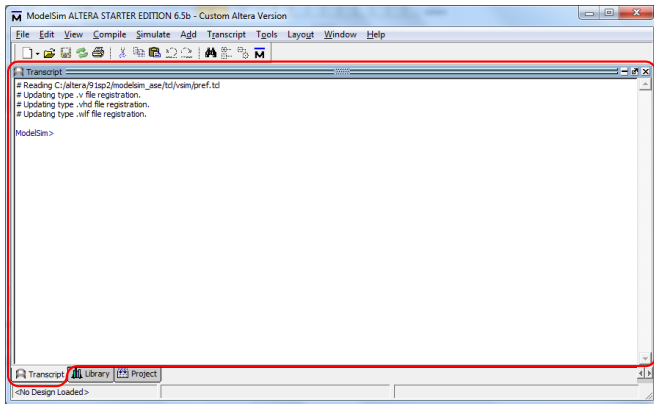
For starting the Modelsim environment on Linux:

- Run the `start_modelsim_linux.sh` script from terminal
- The shared content is accessible through **File** > **Open**, by selecting the "shared" folder



The modelsim environment

The Transcript window is marked out in the image bellow:



All commands will be issued in the Transcript window

Prepare a modelsim project

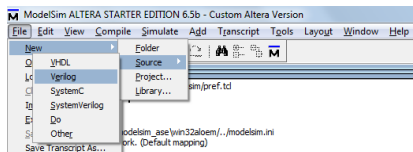
Select or create a folder to hold your project files. In this example, folder "d:\student\msim" will be used as the project's folder.

Change the Modelsim's working directory by issuing the following command in the Transcript window:

```
cd d:/student/msim
```

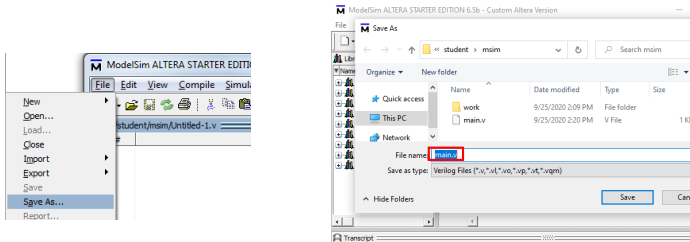
Writing Verilog code using modelsim environment

Use menu **File** **>** **New** **>** **Source** **>** **Verilog** for creating a new file.



Save the file using the **File** **>** **Save As...** menu before writing any code and name it *main.v* for this project.

Important: Don't forget to append extension ".v" to the filename: *main.v*



Verilog example code in modelsim

Add the following Verilog code to the file *main.v*.

```
1 module main;  
2     initial  
3         $display(" Hello , _world!" );  
4 endmodule
```

For compiling and simulating the Verilog module, download the *run_helloworld.txt* script available [▶ here](#).

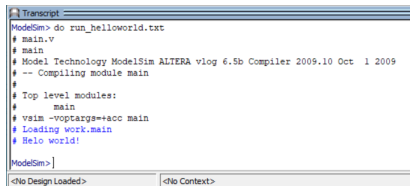
Compile and simulate in modelsim

Issue the following command in the Transcript window:

```
do run_helloworld.txt
```

The *run_helloworld.txt* script will compile the *main.v* source file and, afterwards, launch the simulation of *main* module.

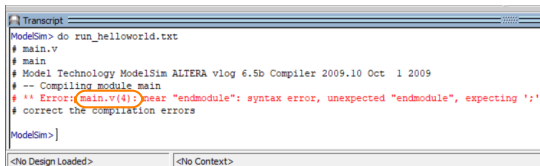
If no error occurred during compilation and simulation, the result in the Transcript window can be seen in the figure bellow:



```
Transcript
ModelSim> do run_helloworld.txt
# main.v
# main
# Model Technology ModelSim ALTERA vlog 6.5b Compiler 2009.10 Oct 1 2009
# -- Compiling module main
#
# Top level modules:
#     main
# vsim -voptargs=+acc main
# Loading work.main
# Helo world!
ModelSim>
<No Design Loaded>    <No Context>
```


modelsim compilation errors

The figure below presents the Transcript window after a compilation error occurred. Highlighted is the filename and an indication of the line containing the incorrect code



The screenshot shows the ModelSim Transcript window. The title bar is labeled 'Transcript'. The text inside the window is as follows:

```
ModelSim> do run_helloworld.txt
# main.v
# main
# Model Technology ModelSim ALTERA vlog 6.5b Compiler 2009.10 Oct 1 2009
# -- Compiling module main
# ** Error: main.v(4): near "endmodule": syntax error, unexpected "endmodule", expecting ';'
# correct the compilation errors

ModelSim>]
```

The error message is highlighted in red. Within this message, the filename 'main.v' and the line number '(4)' are circled in orange. At the bottom of the window, there are two status boxes: '<No Design Loaded>' on the left and '<No Context>' on the right.

After correcting the implementation errors, the same command will be launched again from the Transcript window.