MODELSIM LAB #1

Fall 2020

CSE-204L Computer Organization and Architecture Lab

Submitted by: **Shah Raza**

Registration No.: **18PWCSE1658**

Class Section: **B**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: ______________

Submitted to:

Engr. Amaad Khalil

Sunday, February 28, 2021

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

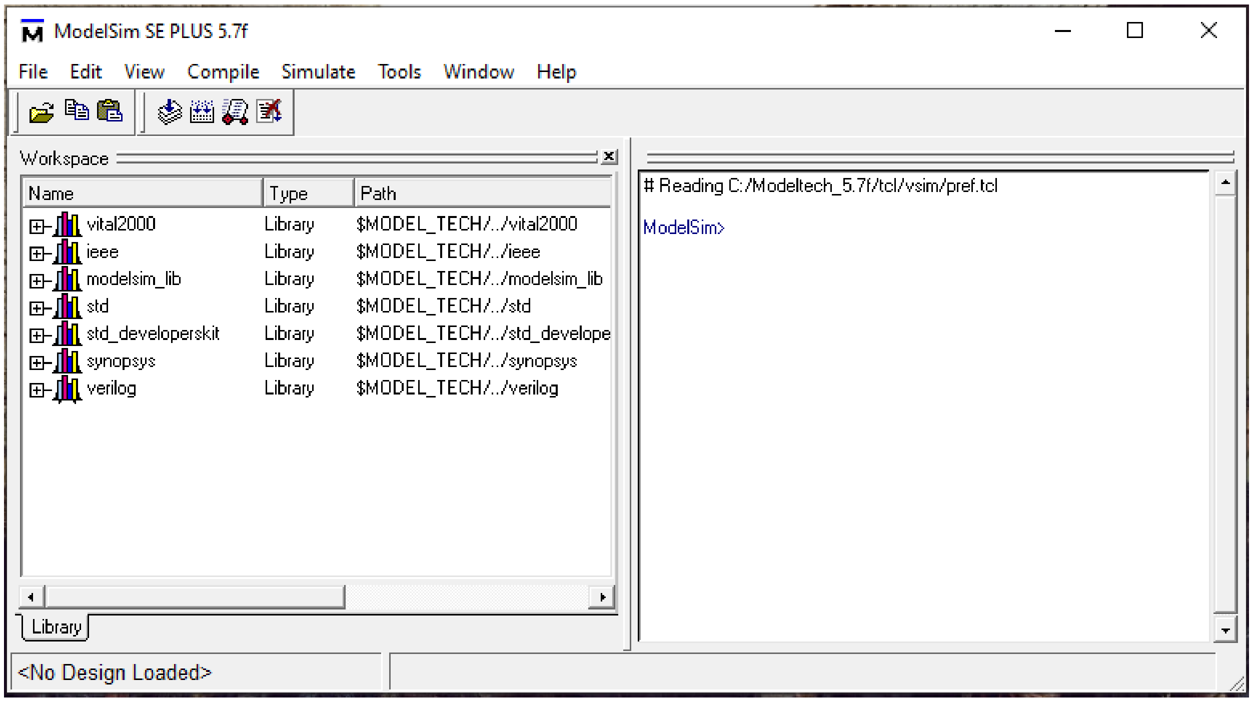
MODELSIM:

MODELSIM is used to simulate HDL and VHDL.

Steps of Installation:

1) Unzip, unRAR or use the installer
2) Install using "setup.exe"
3) After installation, copy "mgls.dll", "modeltech.exe" and "vlm.exe" into your "<installdir>\Win32" dir, overwrite when prompted
4) Copy "license.dat" into your "<installdir>\Win32" dir and into "C:\FlexLM", overwrite when prompted.
5) Done

Interface of ModelSim:



Name	Type	Path
vital2000	Library	$MODEL_TECH/../vital2000
ieee	Library	$MODEL_TECH/../ieee
modelsim_lib	Library	$MODEL_TECH/../modelsim_lib
std	Library	$MODEL_TECH/../std
std_developerskit	Library	$MODEL_TECH/../std_develope
synopsys	Library	$MODEL_TECH/../synopsys
verilog	Library	$MODEL_TECH/../verilog

```
# Reading C:/Modeltech_5.7f/tcl/vsim/pref.tcl

ModelSim>
```

<No Design Loaded>

How to use ModelSim?

Step 1
Create a new project:
Select **File > New > Project** (Main window) to create a new project. This opens the **Create Project** dialog. The dialog includes these options:

• **Project Name**
The name of the new project.
• **Project Location**
The directory in which the *.mpf* file will be created.
• **Default Library Name**
The name of the working library.
You can generally leave the **Default Library Name** set to "work." The name you specify will be used to create a working library subdirectory within the Project Location. After selecting OK, you will see a blank Project tab in the workspace area of the Main window and the **Add Items to the Project** dialog.

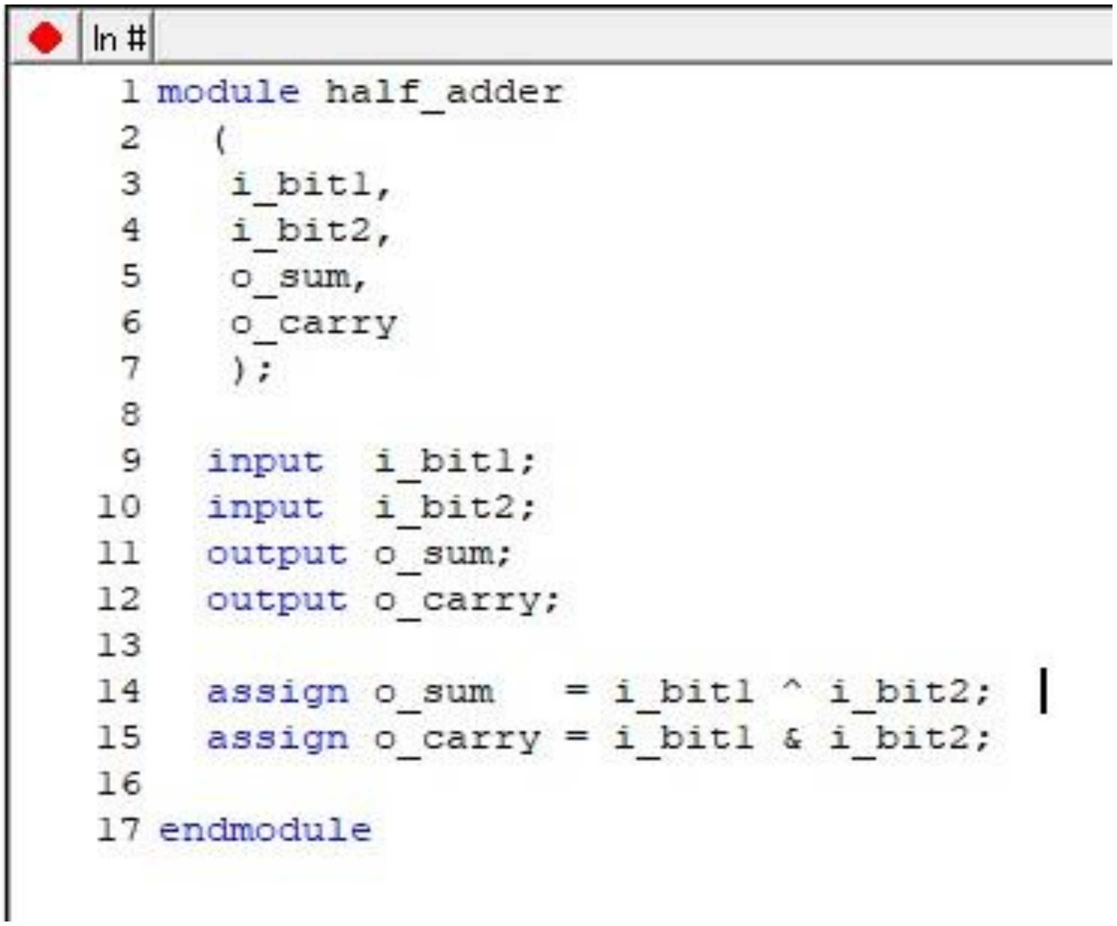
Step 2
Adding items to the project
The **Add Items to the Project** dialog includes these options:

- ➢ **Create New File**
  Create a new VHDL, Verilog, Tcl, or text file using the Source window. See below for details.
- ➢ **Add Existing File**
  Add an existing file. See below for details.
- ➢ **Create Simulation**
  Create a Simulation Configuration that specifies source files and simulator options.
- ➢ **Create New Folder**
  Create an organization folder.
- ➢ **Create New File**
  The **Create New File** command lets you create a new VHDL, Verilog, Tcl, or text file using the Source window. You can also access this command by selecting **File > Add to Project > New File** (Main window) or right-clicking (2nd button in Windows; 3rd button in UNIX) in the Project tab and selecting **Add to Project > New File**.
- ➢ The **Create Project File** dialog includes these options:
- ➢ **File Name**
  The name of the new file
- ➢ **Add file as type**
  Add the type of the new file. Select VHDL, Verilog, TCL, or text.

- **Folder**
  The organization folder in which you want the new file placed. You must first create folders in order to access them here.

  When you select OK, the Source window opens with an empty file, and the file is listed in the Project tab of the Main window workspace.

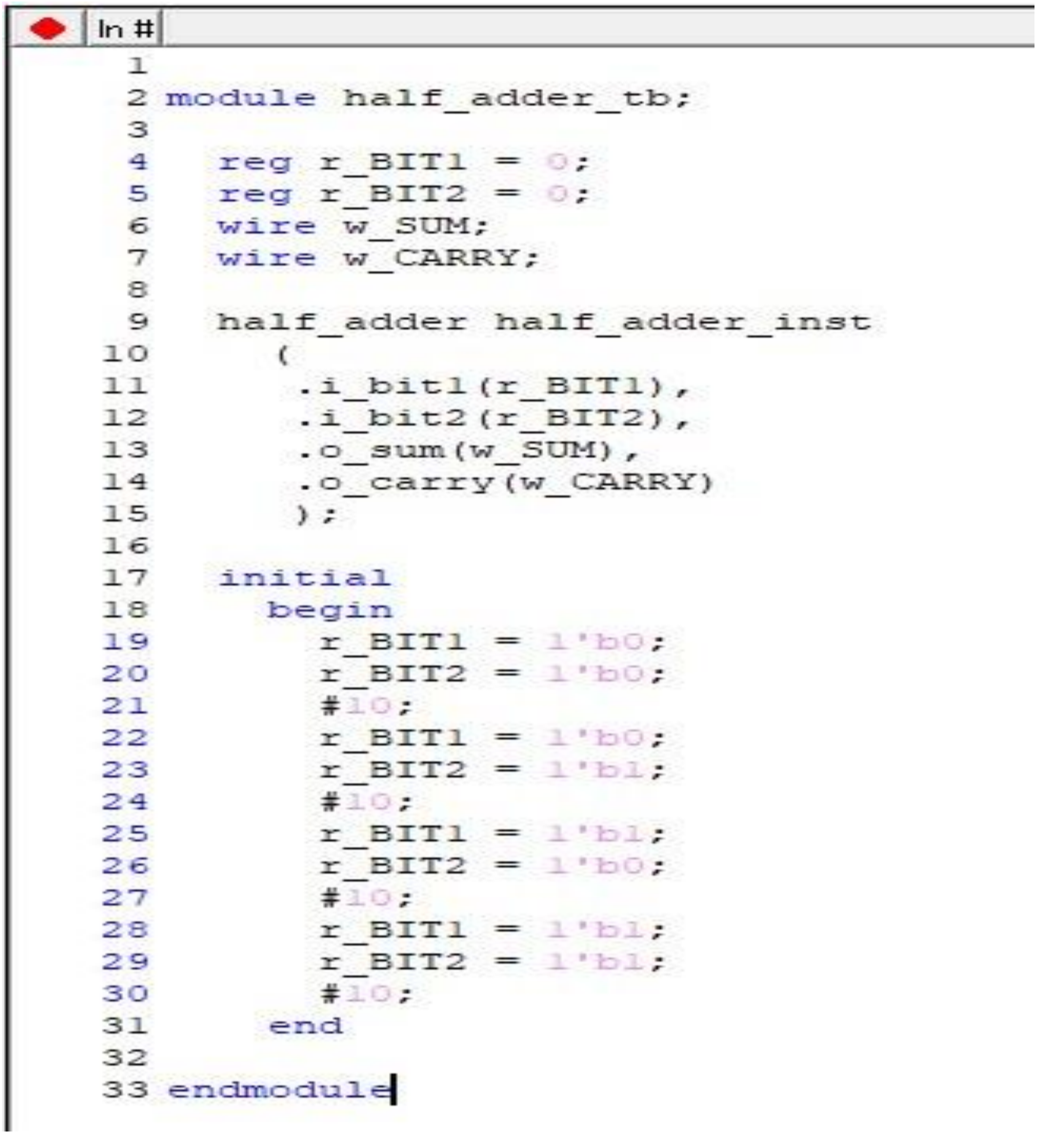
Task: Half Adder
Code:



```
module half_adder
  (
   i_bit1,
   i_bit2,
   o_sum,
   o_carry
   );

  input  i_bit1;
  input  i_bit2;
  output o_sum;
  output o_carry;

  assign o_sum   = i_bit1 ^ i_bit2;
  assign o_carry = i_bit1 & i_bit2;

endmodule
```

Command Explanation:

1. input: Define inputs for the system.
2. output: Define the outputs for the System.
3. ^:X0R gate between the inputs.
4. &:AND gate between the inputs.

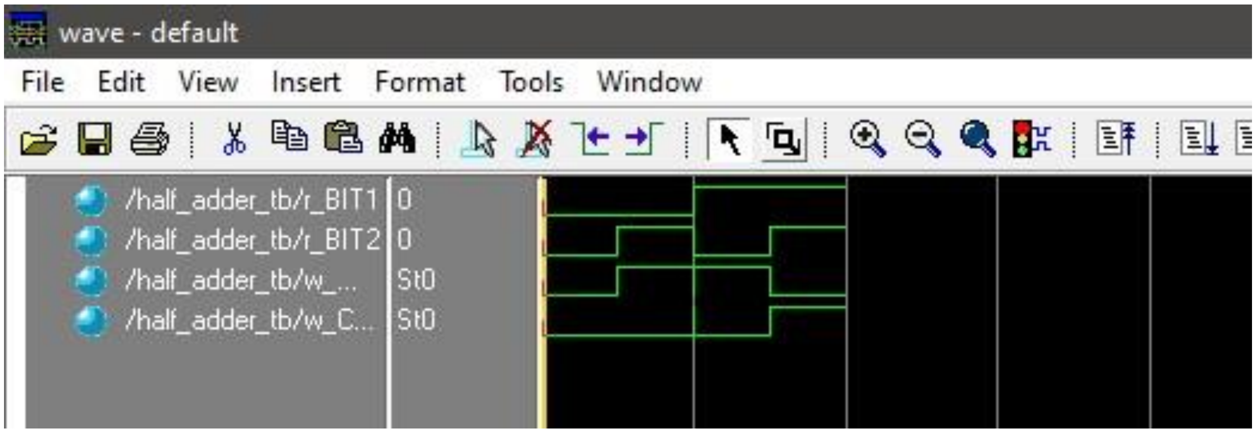


```

module half_adder_tb;

  reg r_BIT1 = 0;
  reg r_BIT2 = 0;
  wire w_SUM;
  wire w_CARRY;

  half_adder half_adder_inst
    (
     .i_bit1(r_BIT1),
     .i_bit2(r_BIT2),
     .o_sum(w_SUM),
     .o_carry(w_CARRY)
     );

  initial
    begin
      r_BIT1 = 1'b0;
      r_BIT2 = 1'b0;
      #10;
      r_BIT1 = 1'b0;
      r_BIT2 = 1'b1;
      #10;
      r_BIT1 = 1'b1;
      r_BIT2 = 1'b0;
      #10;
      r_BIT1 = 1'b1;
      r_BIT2 = 1'b1;
      #10;
    end

endmodule
```

Command Explanation:

1. reg: To store the input values to the system.
2. wire: Store the Outputs of the Gates.
3. 1’b0: Store 1-bit value of zero.
4. 1’b1: Store 1-bit value of 1.
5. #10: Delay of 10 Unit of Processor.

Wave:



```
/half_adder_tb/r_BIT1  0
/half_adder_tb/r_BIT2  0
/half_adder_tb/w_...   St0
/half_adder_tb/w_C...  St0
```