

# ModelSim Simulation with and without Quartus Prime Lite

## Contents

Introduction .....	3
Prerequisites .....	3
Lab2_1 “Launching ModelSim from Quartus Prime” .....	4
Objectives.....	4
The project overview.....	4
Creating a project in Quartus Prime.....	6
Add IP Components to the design.....	7
Observing the top level design file.....	11
Observing the Test-bench for the design.....	12
Setting up ModelSim from Quartus .....	12
Additional steps and comments.....	17
Conclusions .....	18
Lab2_2 “Launching ModelSim independently from Quartus Prime” .....	19
Objectives.....	19
The project overview.....	19
Creating a project in Quartus Prime.....	20
Add IP Components to the design.....	20
Launching ModelSim independently from Quartus Prime .....	22
Conclusions .....	24

# Introduction

Design simulations are able to include wide range of analyses that virtually test behavior of a product under various operating and environmental conditions.

As opposed to trial-and-error, a smart simulation process allows targeted implementation of design choices in various stages of the development cycle. Proper functional and timing simulation is important to ensure design functionality and success.

This drastically reduces the need for recurrent, time-consuming testing on expensive physical prototypes, and subsequently shortens the total development time.

ModelSim is a multi-language HDL simulation environment offered by Mentor Graphics. *It can be used independently or with Intel Quartus Prime, which can help create libraries and link the designs to ModelSim. Using the ModelSim Intel FPGA Started Edition software simplifies the debugging with help of simulations.*

## Prerequisites

You are expected to:

- be familiar with basics of logic and digital design,
- be familiar with Verilog constructs,
- have Quartus Prime Lite installed.

# Lab2\_1 “Launching ModelSim from Quartus Prime”

## Objectives

You will use this lab to familiarize yourself with the following:

- 1 How to run ModelSim for simulation purposes from Quartus Prime.
- 2 How to use IP functions from IP Catalog of Quartus Prime with ModelSim.
- 3 Some helpful shortcuts for ModelSim.

## The project overview

This lab utilizes a simple digital logic design shown on Figure 1.

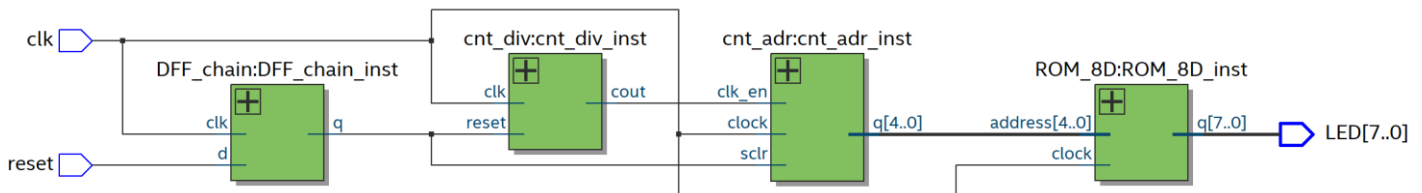


Figure 1

Inputs for the top-level module are:

- Clk – clock signal (frequency is 25MHz).
- Reset – reset signal coming from Push Button.

Outputs for the top-level module are:

- LED[7:0] – FPGA pins connected with Leds on a development board.

Algorithm of the project is:

- Input clock is divided (value of division is parameterized) by **cnt\_div** unit.
- Unit **cnt\_adr** provides addresses for the **ROM\_8D** unit.
- Unit **ROM\_8D** keeps data and, in accordance of the current address, displays the word (8 bits) of data on the LED[7..0] outputs.
- Reset signal synchronously resets all counters having active value “1”.

Unit **DFF\_chain** synchronizes external reset signal. It contains two DFF flip-flops connected in a chain. The source code for the unit is provided. It is in the file **C:\Intel\_trn\Quartus\_ModelSim\Lab2\_1\DFF\_chain.v**.

```

1  module DFF_chain
2  (input  clk,
3   input  d,
4   output reg q );
5
6  reg d_int;
7  always @(posedge clk)
8  begin
9      d_int  <= d;
10     q      <= d_int;
11 end
12
13 endmodule

```

Figure 2

Unit **cnt\_div** divides the incoming clock signal. Division ratio is a parameter for the unit. The source code for the unit is provided. It is in the file **C:\Intel\_trn\Quartus\_ModelSim\Lab2\_1\DFF\_chain.v**.

```

1  module cnt_div
2  #(parameter div = 25)
3  (input clk,
4   input reset,
5   output reg cout );
6
7  reg [31:0] cnt;
8  wire cycle;
9
10 always @(posedge clk)
11     if (reset | cycle)
12         cnt <= 0;
13     else
14         cnt <= cnt+1;
15
16 assign cycle = (cnt == (div-1));
17
18 always @(posedge clk)
19     if (reset)
20         cout <= 1'b0;
21     else
22         if (cycle)
23             cout <= 1'b1;
24         else
25             cout <= 1'b0;
26
27 endmodule

```

Figure 3

Unit **cnt\_adr** provides addresses for ROM\_8D unit. You need to create **cnt\_adr** unit during the Lab by using **LPM\_COUNTER** from IP Library.

Unit **ROM\_8D** keeps data that are stored in the file **C:\Intel\_trn\Quartus\_ModelSim\Lab2\_1\ROM\_8D.hex**. You need to create **ROM\_8D** unit during the Lab by using **ROM: 1-PORT** from IP Library.

Unit Lab2\_1 is the top level unit, highlighted on Figure 1. It connects all units together. The source code for the unit is provided. It is in the file **C:\Intel\_trn\Quartus\_ModelSim\Lab2\_1\Lab2\_1.v**.

In the folder **C:\Intel\_trn\Quartus\_ModelSim\Lab2\_1** there is file **tb\_Lab2\_1.v**. The file contains **tb\_Lab2\_1** module, which is a simple testbench for the unit **Lab2\_1**.

Before going forward with the Lab be sure that you have all files in the working folder.

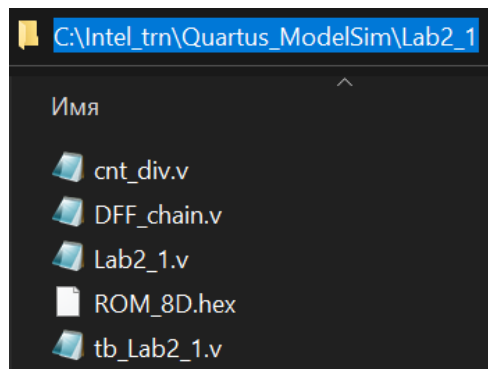


Figure 4

## Creating a project in Quartus Prime

- 1 Start Quartus Prime **Lite** development tool
- 2 Create a project
  - a. Working directory: **C:/Intel\_trn/Quartus\_ModelSim/Lab2\_1**
  - b. Project name: **Lab2\_1**
  - c. Top-Level design entity: **Lab2\_1**
  - d. Project Type: **Empty project**
  - e. Add files from the folder **C:/Intel\_trn/Quartus\_ModelSim/Lab2\_1**: **cnt\_div.v**; **DFF\_chain.v**; **Lab2\_1.v**
  - f. Device: **EP4CE6E22C8**
  - g. EDA tools: **ModelSim Altera**; **Verilog HDL**

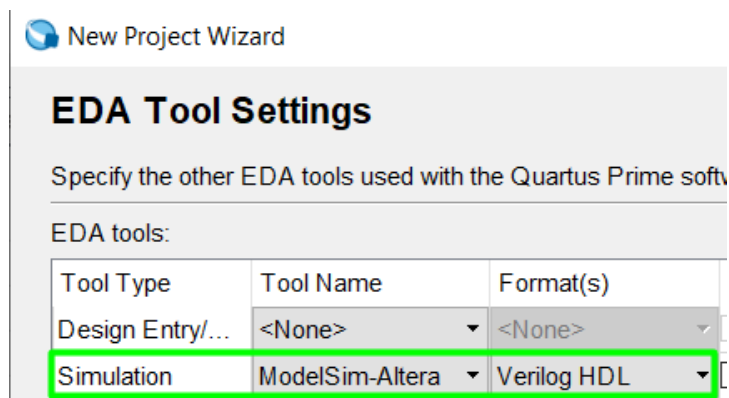


Figure 5

- 1 Creating **cnt\_adr** unit by using LPM\_COUNTER.
  - a. Find LPM\_COUNTER function by typing LPM\_COUNTER in the find field of IP catalog. If you do not have the IP Catalog already open, go to View menu => Utility Windows => IP Catalog to view the window.

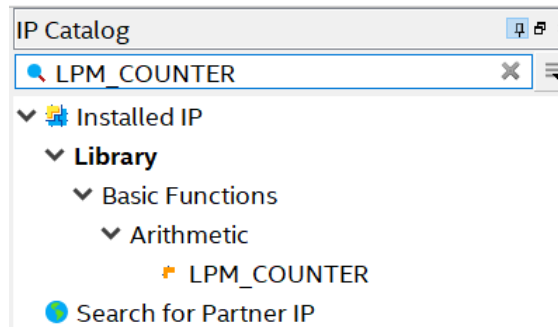


Figure 6

- b. Double click LPM\_COUNTER.
    - c. Name the file as **cnt\_adr** (Please make sure to name it as **cnt\_adr**) in the **Save IP Variation** box that came up and click **OK**.

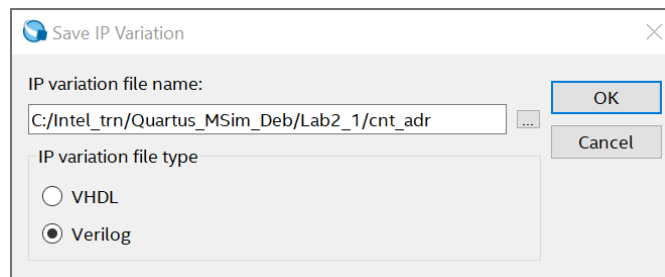


Figure 7

- d. In the MegaWizard, choose **5 bits** for bus “q” and leave remaining settings to default.

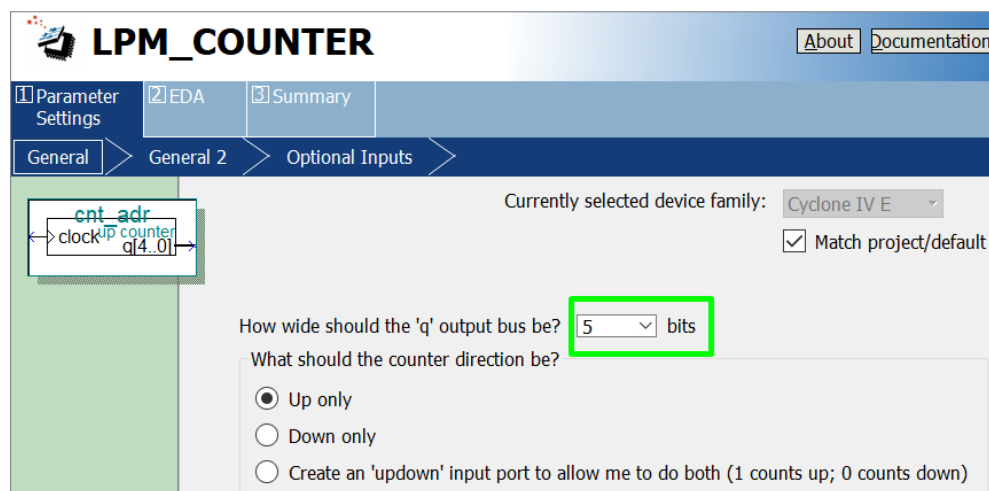


Figure 8

Click **Next**.

- e. In the window appeared select Clock enable check box and leave remaining settings to default.

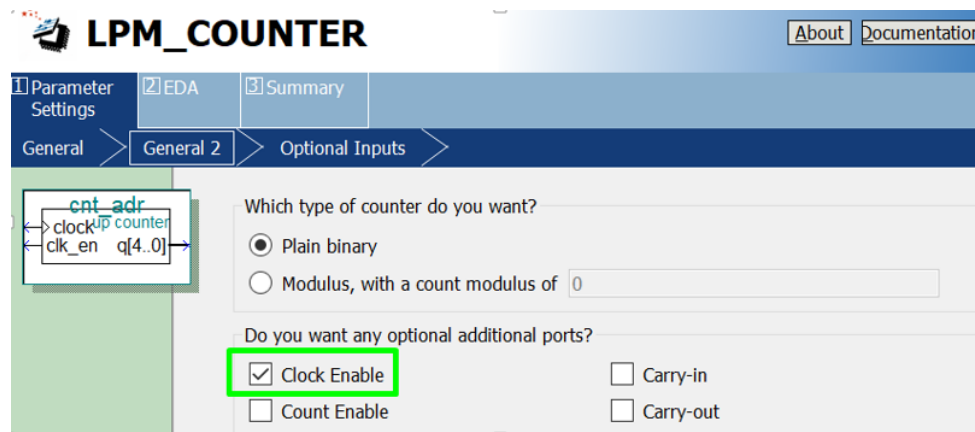


Figure 9

Click **Next**.

- f. In the window appeared select synchronous **Clear** check box and leave remaining settings to default.

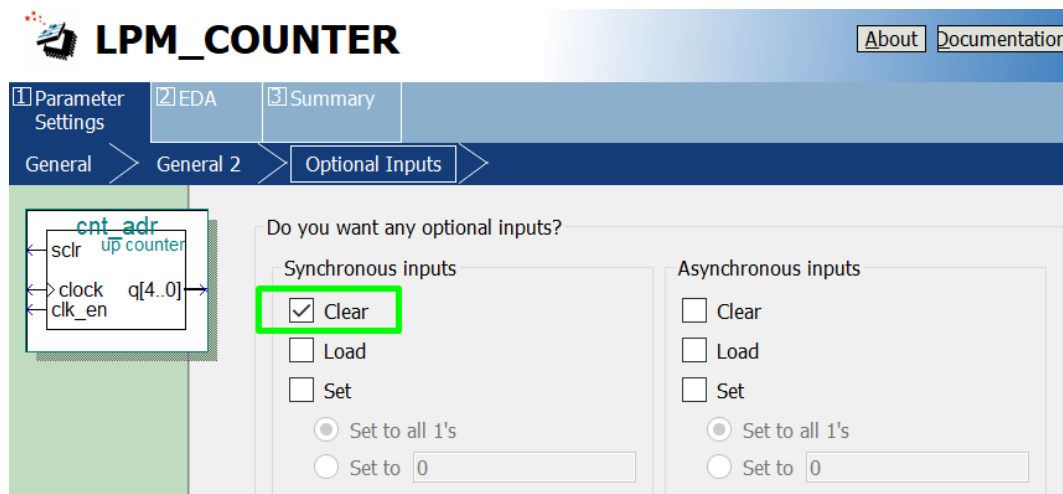


Figure 10

Click **Next**.

- g. In the window appeared pay attention to the simulation library pointed. It is lpm library. When using Native link of Quartus Prime you have all libraries referenced correctly automatically. Leave all settings to default and click **Next**.
- h. In the window appeared select **cnt\_adr\_inst.v** and **cnt\_adr\_bb.v** check boxes and in the working folder you will have templates for using the generated unit in a top level module. Leave the remaining settings to default and click **Finish**.
- i. After you click **Finish** you will get a dialog box asking you if you want to add the IP files generated to the project. Click **Yes**. You will add cnt\_adr.qip file to the project. The file contains information about generated unit, which is for Quartus Prime.

## 2 Creating ROM\_8D unit by using ROM: 1-PORT.

- a. Find ROM: 1-PORT function by typing LPM\_COUNTER in the find field of IP catalog. If you do not have the IP Catalog already open, go to View menu => Utility Windows => IP Catalog to view the window.



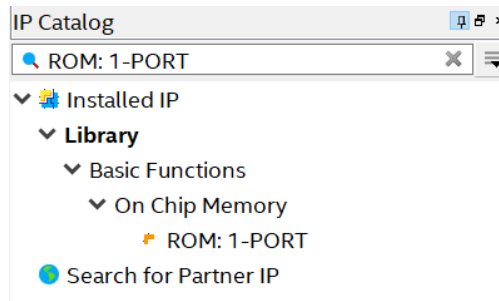


Figure 11

- b. Double click ROM: 1-PORT.
- c. Name the file as **ROM\_8D** (Please make sure to name it as **ROM\_8D**) in the **Save IP Variation** box that came up and click **OK**.

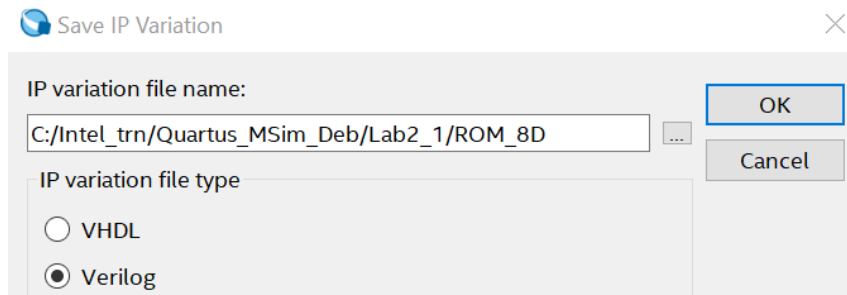


Figure 12

- d. In the window appeared choose 32 words for the memory. Be sure that the bus"q" is 8 bits wide. Leave the remaining settings to default.

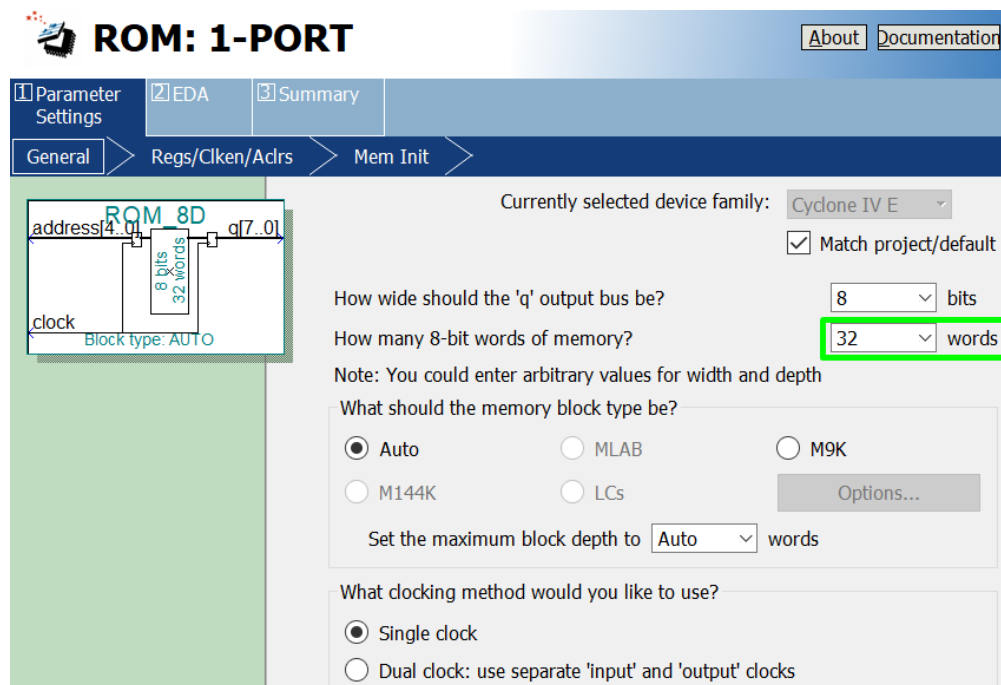


Figure 13

Click **Next**.

- e. In the window appeared leave all settings to default. Click **Next**.
- f. In the window appeared point name ROM\_8D.hex as the name of the file with memory data. You can type the name directly to the field or use Browse button to select the file from working folder. Leave the remaining settings to default.

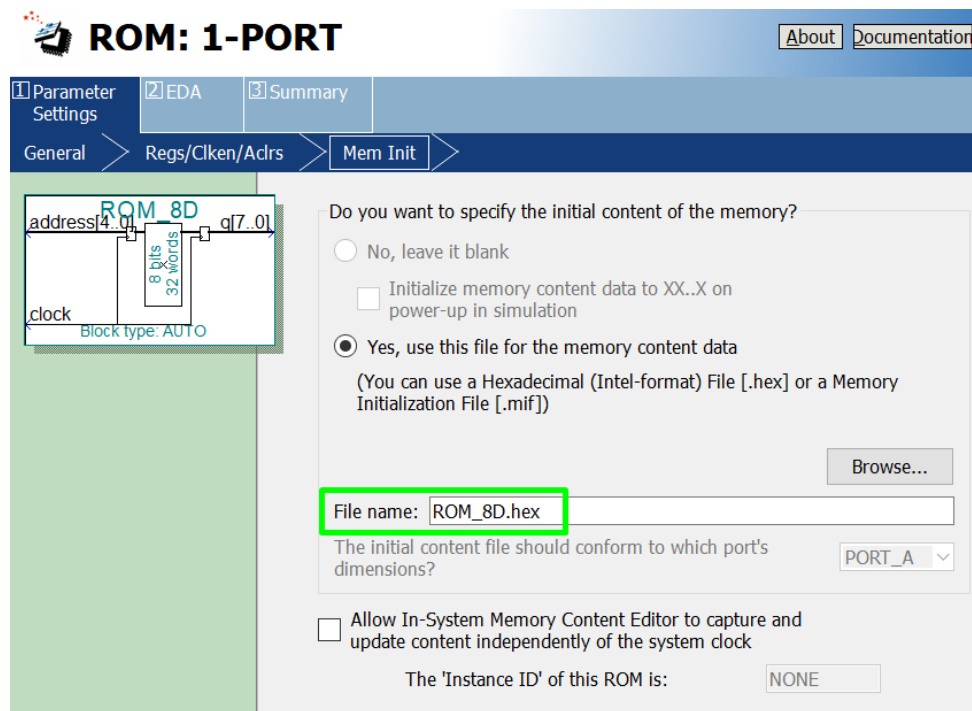


Figure 14

Click **Next**.

- g. In the window appeared pay attention to the simulation library pointed. It is altera\_mf library. When using Native link of Quartus Prime you have all libraries referenced correctly automatically. Leave all settings to default and click **Next**.
  - h. In the window appeared select **ROM\_8D\_inst.v** and **ROM\_8D\_bb.v** check boxes and in the working folder you will have templates for using the generated unit in a top level module. Leave the remaining settings to default and click **Finish**.
  - i. After you click **Finish** you will get a dialog box asking you if you want to add the IP files generated to the project. Click **Yes**.
- 3 To check if the IPs are added in the design: select **Project navigator** window => **IP Components** in the drop down menu.

Project Navigator IP Components		
	Entity	IP Component
✓	cnt_adr	LPM_COUNTER
✓	ROM_8D	ROM: 1-PORT

Figure 15

1 In the Main Menu, go to File => Open... => C:\Intel\_trn\Quartus\_ModelSim\Lab2\_1 and open **Lab2\_1.v** file.

```

1  module Lab2_1
2  #(parameter div_by = 25)
3  (input      CLK,
4   input      RESET,
5   output     [7:0] LED );
6
7  wire [4:0] adr;
8  wire clk_en;
9  wire srst;
10
11 DFF_chain DFF_chain_inst (
12     .clk    (CLK),
13     .d      (RESET),
14     .q      (srst)
15 );
16
17 cnt_div #(div_by) cnt_div_inst (
18     .clk    (CLK),
19     .reset  (RESET),
20     .cout   (clk_en)
21 );
22
23 cnt_adr cnt_adr_inst (
24     .clock   ( CLK),
25     .sclr   ( RESET),
26     .clk_en  ( clk_en),
27     .q       ( adr)
28 );
29
30 ROM_8D ROM_8D_inst (
31     .clock   ( CLK),
32     .address ( adr),
33     .q       ( LED)
34 );
35
36 endmodule

```

*Figure 16*

This is top-level module. Take a look at how the connections are made between the units and the IP units.

- 2 To check if the project is correct:
  - a. Select **Project navigator** window => **Hierarchy** in the drop down menu.
  - b. Select Lab2\_1 top level unit
  - c. Select **Processing** menu => **Start** => **Start Analysis and Elaboration**
  - d. Be sure that you don't have Errors and Critical Warnings.

1 In the Main Menu, go to File => Open... => C:\Intel\_trn\Quartus\_ModelSim\Lab2\_1 and open **tb\_Lab2\_1.v** file.

```
1  `timescale 1ns/100ps
2  module tb_Lab2_1();
3
4  reg tb_clk;
5  reg tb_reset;
6  wire [7:0] tb_led;
7
8  Lab2_1 #(5)
9  DUT (
10     .LED      (tb_led      ),
11     .CLK      (tb_clk      ),
12     .RESET    (tb_reset    )
13 );
14 initial
15 begin
16     tb_clk      = 1'b0;
17     tb_reset    = 1'b1;
18
19     #100 tb_reset = 0;
20     #1000 $stop;
21 end
22
23 always #10 tb_clk = ~tb_clk;
24
25 endmodule
```

Figure 17

This is a simple testbench. Take a look at how the connections are made between the module and the IP.

First, notice all the inputs of the **Lab2\_1.v** module (tb\_clk, tb\_reset) are declared as type **reg** in the **tb\_counter.v** file.

Next, all the outputs of the **Lab2\_1.v** are declared as type **wire** i.e. [7:0] tb\_led.

In the **initial block**, first all the inputs are initialized. Then after 100 ns tb\_reset switched to inactive state, i.e. “0”. Then after 1000 ns, the simulation process will be stopped by command stop. The initial block in Verilog is used generally in test benches. It is a block that only runs once unlike the always block. Using always block *always #10 clock = ~clock;* we create a clock of **50 Mhz** for clk input .

## Setting up ModelSim from Quartus

Once you have completed the above sections you need to launch ModelSim for simulations. To do so you need to check (and may be to change) settings so that Modelsim can open through Quartus Prime Lite.

- 1 Go to **Tools** → **Options** → **EDA Tool Options**.
- 2 In ModelSim-Altera, enter the executable pathway.

For finding the executable pathway, locate where the Quartus was installed by you (For example: C:\intelFPGA\_lite\20.1\modelsim\_ase\win32aloem).

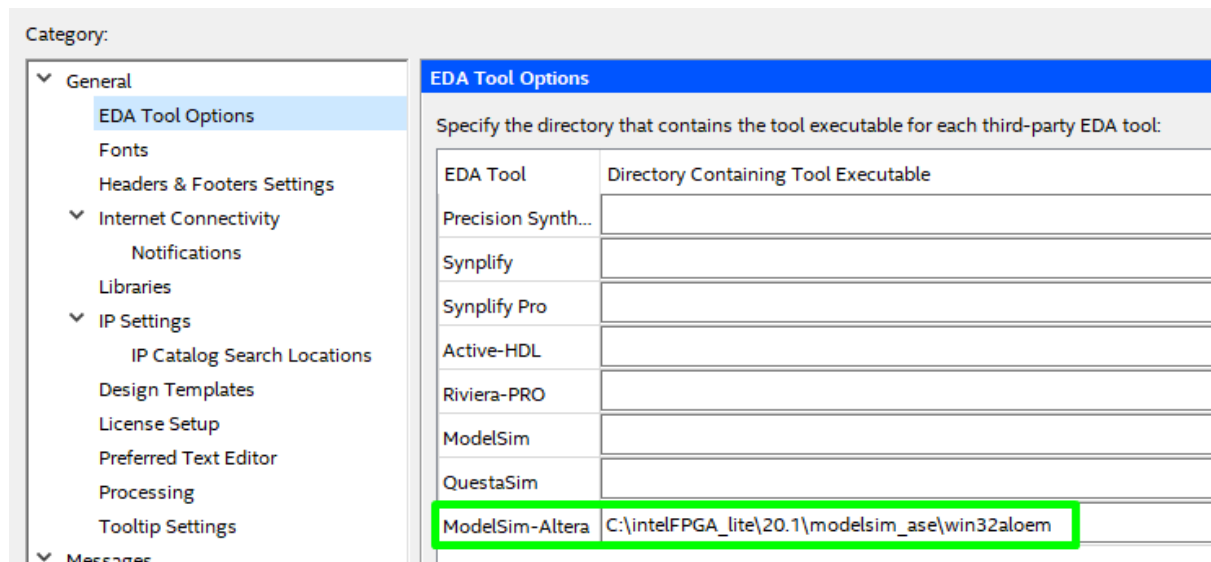


Figure 18

Select **OK** when finished.

- 3 Next step is to make sure that the test bench can be compiled without errors.
  - a. Go to **Assignments => Settings => EDA Tool Settings => Simulation**.
  - b. Select **ModelSim-Altera** in the **Tool Name** field.
  - c. Under **NativeLink Settings** select **Compile Test Benches** and click on **Test Benches...**
  - d. In the **Test Benches** Dialog Box click on **New...**
  - e. In the field Test bench name, print the name **tb\_Lab2\_1** (It is the top module of the testbench too).
  - f. In the field **File name** select the Testbench file **tb\_Lab2\_1.v** to be added. Click **Add**.

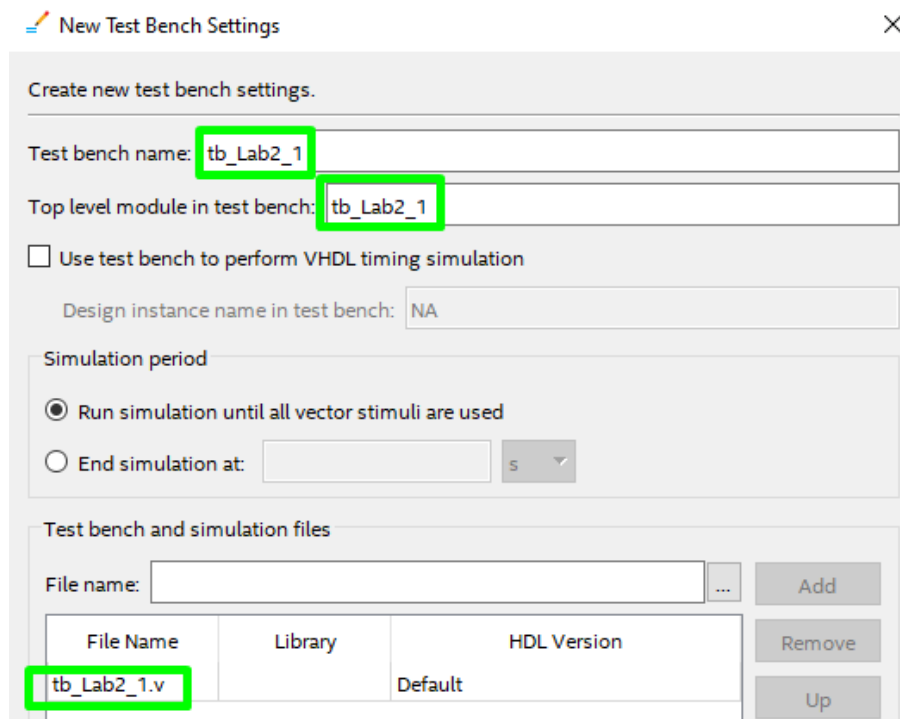



Figure 19

- g. Click **OK**.

- h. Click **OK** again.
- i. Click **OK** again.
- 4 Once all the settings are in place you can go to  in the toolbar for full compilation of the design.
- 5 Next, from Quartus: **Tools=> Run Simulation=> RTL Simulation**
- 6 Once this step is completed. ModelSim should have opened up on the screen. The GUI for ModelSim is shown below.

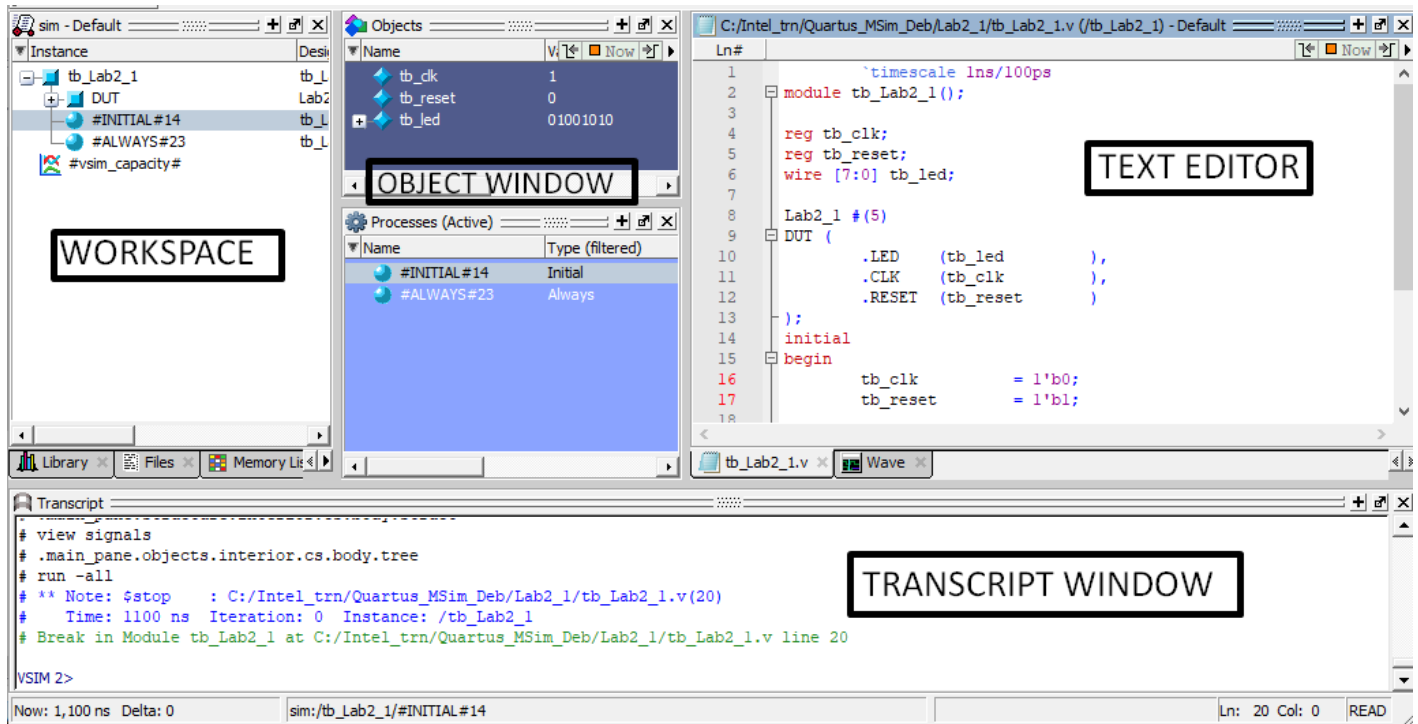


Figure 20

Note: If you get error loading the design related to source files you might have to go to your working folder, where the project is stored, and check and correct the files with errors.

If you have used ModelSim before and know how to move around the various tabs, you may not see the same layout as shown in Figure 20. You can go to Layout => Reset to get the same layout of the GUI.

### **Workspace:**

The workspace consists of various tabs, each tab having a different functionality.

Library tab represents various modules and files present in the design.

Memory List tab will represent the values in the memory units used in the design.

Once the design is simulated, a tab called sim is created which displays the hierarchical structure of the design. You can navigate within the hierarchy by clicking on any line with a '+' (expand) or '-' (contract) icon.

### **Objects Window Pane:**

The Objects pane shows the names and current values of data objects in the current region (selected in the Workspace).

Data objects include signals, nets, registers, constants and variables not declared in a process, generics, and parameters.

### ***Transcript Window:***

The Transcript pane provides a command-line interface and serves as an activity log including status and error messages.

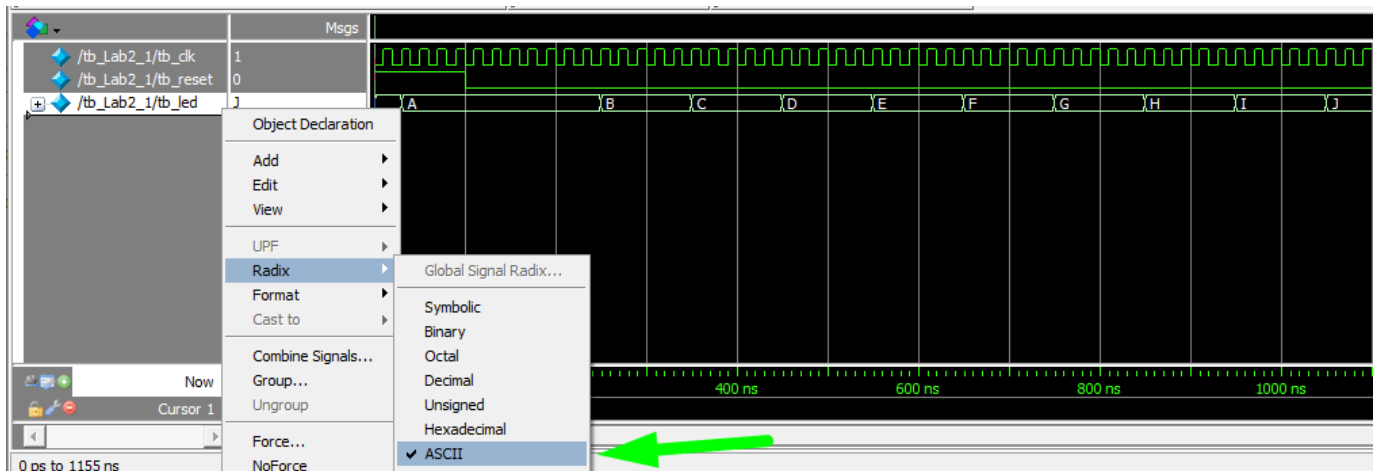
### ***Text Editor:***

This pane allows you to edit and read design files in ModelSim.

### ***The Wave window***

Select the tab Wave to see the wave window. You can now observe the waveform and verify the functionality of the design.

- 7 **Undock** the wave window.
- 8 **Zoom Full** the wave window.
- 9 Set radix for tb\_led bus as ASCII

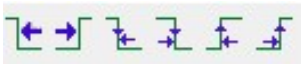


*Figure 21*

You should see A, B, C, D ... letters displayed as on Figure 21.

*Some tips:*

- If you want to get to the next rising or falling edge of a particular signal. Select that signal in

the waveform and click on  in the toolbar for respective actions.

If you want to change the default timescale (for example from ns to us), navigate the cursor on the lower part near the timescale as shown on Figure 22, then right click and select **Grid, Timeline & Cursor Control..**

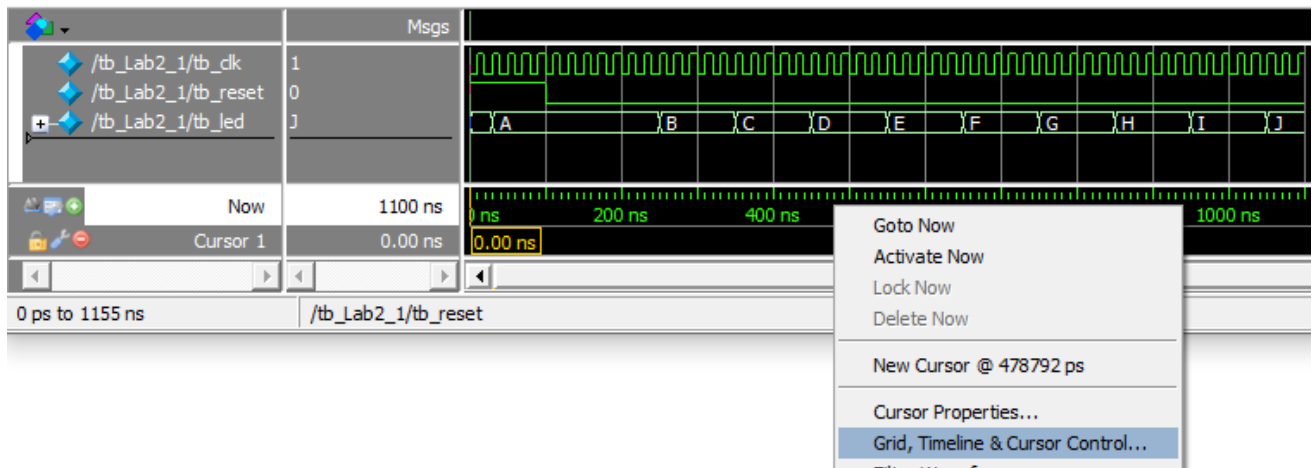


Figure 22

- In the Grid, Timeline & Cursor Control dialogue box, select ns as the Time Units as shown on the Figure 23.

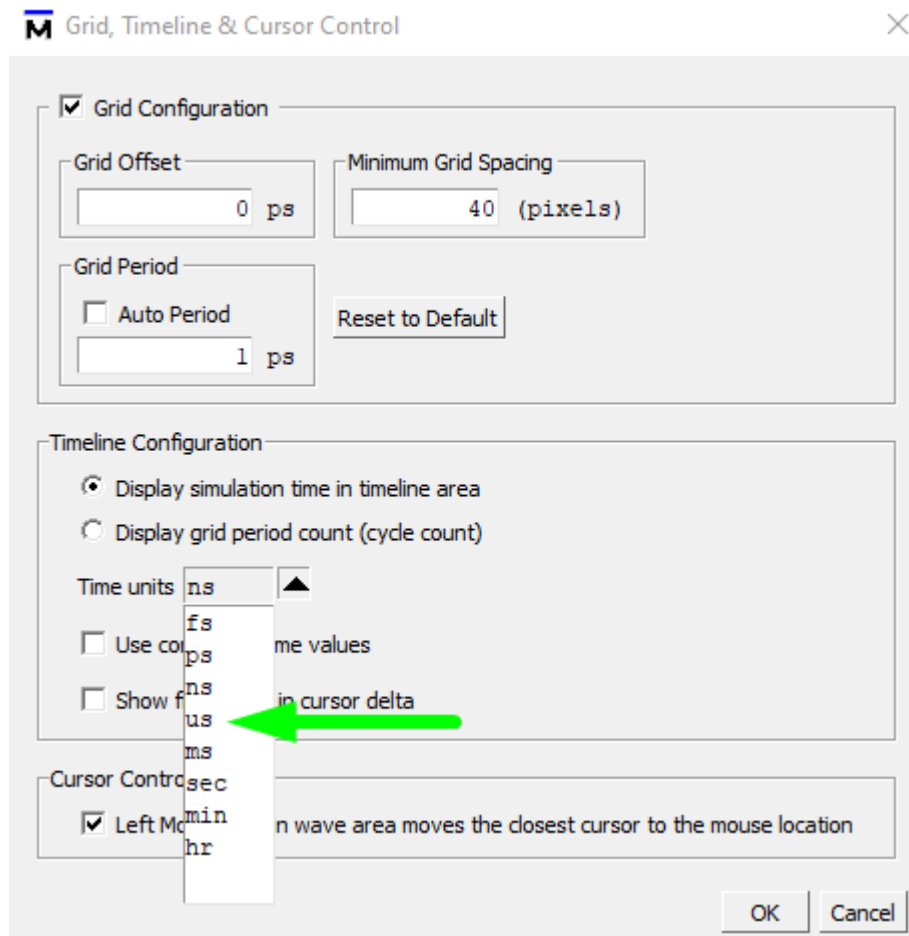


Figure 23

Click **OK** and find differences in the waveform window.

- If you want to save this particular set of signals in the waveform:
  - Go to **File => Save Format** in the wave window.
  - Name it **“wave\_my\_lab2\_1.do”**



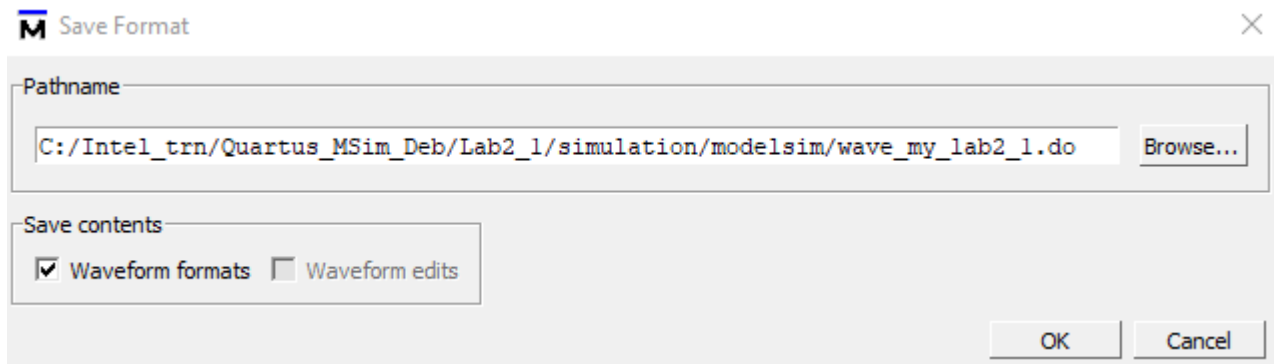


Figure 24

c. Click **Save**.

Next time you, running commands from the Modelsim Transcript, can type “**do wave\_my\_lab2\_1.do**” before a simulation.

d. The last step: Stop simulation and close ModelSim window.

## Additional steps and comments

You can use opened ModelSim as usual ModelSim unless you need to change any IP unit. Therefore, you can change source files and test benches, then recompile ones, restart simulation, debug the source codes. If you need to change IP you need to close ModelSim, switch back to Quartus Prime and change what you want. Then you need to run from Quartus Prime: **Tools=> Run Simulation=> RTL Simulation**.

Pay attention:

- When you run RTL Simulation for the first time, a folder called **simulation** was generated in the Project folder. The folder has sub-folder **Modelsim**.
- Under **Modelsim** folder look for a .do extension file.
- There is a file “**Lab2\_1\_run\_msim\_rtl\_verilog.do**” which was created.
- You can run this do file in Modelsim independently from Quartus Prime:
  - Open ModelSim using the start menu.

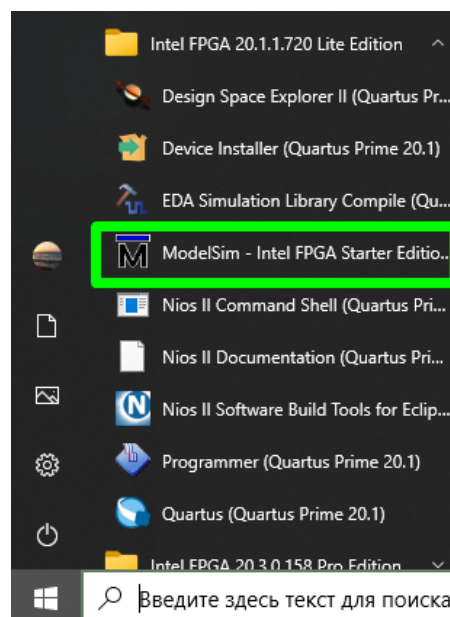
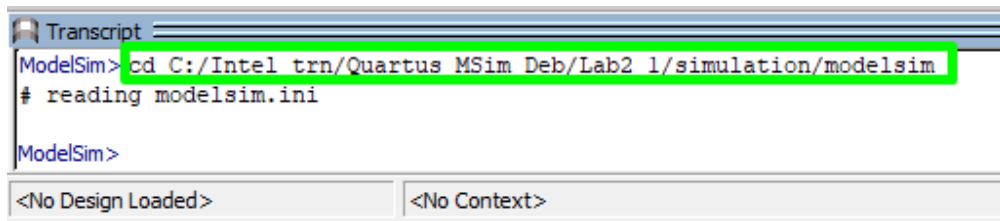


Figure 25

- Once the software loads:
  - Go to the project folder location, typing in the transcript window the command:

*cd C:/Intel\_trn/Quartus\_ModelSim/Lab2\_1/simulation/modelsim*



*Figure 26*

- Once in the project folder, type the command in the transcript window:

*do Lab2\_1\_run\_msim\_rtl\_verilog.do*

The design should load without using the Quartus Software.

- Close the waveform window, if one is open.
- Type in the transcript window the command

*do wave\_my\_lab2\_1.do*

Be sure that you have the wave as it was saved early.

- Stop simulation and close ModelSim window.

For more help and shortcuts on Modelsim, you can go to Help → Documentation → Tutorial.

## Conclusions

With this lab you learnt how to use IP modules in simulation and to launch ModelSim simulation from Quartus Prime by using Native link.

# Lab2\_2 “Launching ModelSim independently from Quartus Prime”

## Objectives

You will use this lab to familiarize yourself with the following:

- 1 How to launching ModelSim independently from Quartus Prime.
- 2 How to use IP functions from IP Catalog of Quartus Prime with ModelSim.

## The project overview

This lab utilizes a simple digital logic design shown on Figure 1. The project will be near the same as for Lab2\_1 with some minor differences.

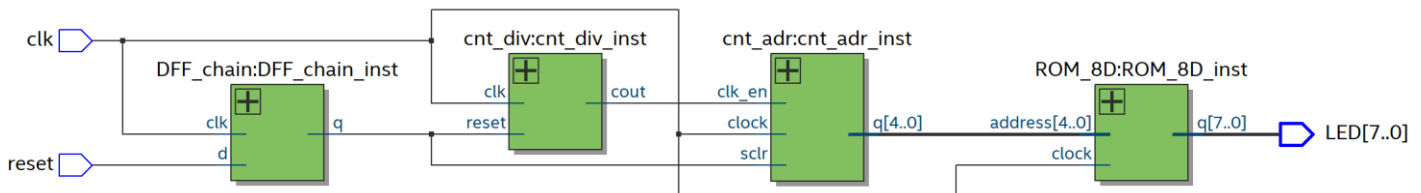


Figure 27

Inputs for the top-level module are:

- Clk – clock signal (frequency is 25MHz).
- Reset – reset signal coming from Push Button.

Outputs for the top-level module are:

- LED[7:0] – FPGA pins connected with Leds on a development board.

Algorithm of the project is:

- Input clock is divided (value of division is parameterized) by **cnt\_div** unit.
- Unit **cnt\_adr** provides addresses for the **ROM\_8D** unit.
- Unit **ROM\_8D** keeps data and, in accordance of the current address, displays the word (8 bits) of data on the LED[7..0] outputs.
- Reset signal synchronously resets all counters having active value “1”.

Unit **DFF\_chain** synchronizes external reset signal. It contains two DFF flip-flops connected in a chain. The source code for the unit is provided. It is in the file **C:\Intel\_trn\Quartus\_ModelSim\Lab2\_2\DFF\_chain.v**.

Unit **cnt\_div** divides the incoming clock signal. Division ratio is a parameter for the unit. The source code for the unit is provided. It is in the file **C:\Intel\_trn\Quartus\_ModelSim\Lab2\_2\DFF\_chain.v**.

Unit **cnt\_adr** provides addresses for ROM\_8D unit. You need to create **cnt\_adr** unit during the Lab by using **LPM\_COUNTER** from IP Library.

Unit **ROM\_8D** keeps data that are stored in the file **C:\Intel\_trn\Quartus\_ModelSim\Lab2\_2\ROM\_8D.hex**. You need to create **ROM\_8D** unit during the Lab by using **ROM: 1-PORT** from IP Library.

Unit Lab2\_2 is the top level unit, highlighted on Figure 27. It connects all units together. The source code for the unit is provided. It is in the file **C:\Intel\_trn\Quartus\_ModelSim\Lab2\_1\ Lab2\_2.v**.

In the folder **C:\Intel\_trn\Quartus\_ModelSim\Lab2\_2** there is file **tb\_Lab2\_2.v**. The file contains tb\_Lab2\_2 module, which is a simple testbench for the unit Lab2\_2.

Before going forward with the Lab be sure that you have all files in the working folder.

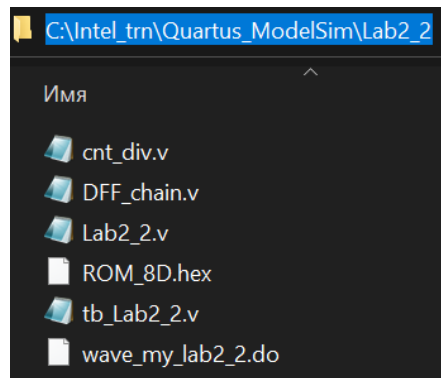


Figure 28

## Creating a project in Quartus Prime

- 1 Start Quartus Prime **Lite** development tool
- 2 Create a project
  - a. Working directory: **C:/Intel\_trn/Quartus\_ModelSim/Lab2\_2**
  - b. Project name: Lab2\_2
  - c. Top-Level design entity: Lab2\_2
  - d. Project Type: Empty project
  - e. Add files from the folder **C:/Intel\_trn/Quartus\_ModelSim/Lab2\_2**: cnt\_div.v; DFF\_chain.v; Lab2\_2.v
  - f. Device: EP4CE6E22C8
  - g. EDA tools: **None**

## Add IP Components to the design

- 1 Creating **cnt\_adr** unit by using LPM\_COUNTER.
  - a. Find LPM\_COUNTER function by typing LPM\_COUNTER in the find field of IP catalog.
  - b. Double click LPM\_COUNTER.
  - c. Name the file as **cnt\_adr** (*Please make sure to name it as cnt\_adr*) in the **Save IP Variation** box that came up and click **OK**.
  - d. In the MegaWizard, choose **5 bits** for bus “q” and leave remaining settings to default. Click **Next**.
  - e. In the window appeared select Clock enable check box and leave remaining settings to default. Click **Next**.
  - f. In the window appeared select synchronous **Clear** check box and leave remaining settings to default. Click **Next**.
  - g. In the window appeared pay attention to the simulation library pointed. It is **lpm** library. We will need to reference the library in ModelSim. Leave all settings to default and click **Next**.

- h. If in the window appeared you select **cnt\_adr\_inst.v** and **cnt\_adr\_bb.v** check boxes, in the working folder you will have templates for using the generated unit in a top level module. Leave the remaining settings to default and click **Finish**.
- i. After you click **Finish** you will get a dialog box. Click **Yes**.

2 Creating ROM\_8D unit by using ROM: 1-PORT.

- a. Find ROM: 1-PORT function by typing LPM\_COUNTER in the find field of IP catalog.
- b. Double click ROM: 1-PORT.
- c. Name the file as **ROM\_8D** (*Please make sure to name it as **ROM\_8D***) in the **Save IP Variation** box that came up and click **OK**.
- d. In the window appeared choose 32 words for the memory. Be sure that the bus"q" is 8 bits wide. Leave the remaining settings to default. Click **Next**.
- e. In the window appeared leave all settings to default. Click **Next**.
- f. In the window appeared point name **ROM\_8D.hex** as the name of the file with memory data. Click **Next**.
- g. In the window appeared pay attention to the simulation library pointed. It is **altera\_mf** library. We will need to reference the library in ModelSim. Leave all settings to default and click **Next**.
- h. If in the window appeared you select **ROM\_8D\_inst.v** and **ROM\_8D\_bb.v** check boxes, in the working folder you will have templates for using the generated unit in a top level module. Leave the remaining settings to default and click **Finish**.
- i. After you click **Finish** you will get a dialog box asking you if you want to add the IP files generated to the project. Click **Yes**.

3 Check if the IPs are added in the design: select **Project navigator** window => **IP Components** in the drop down menu.

4 To check if the project is correct:

- a. Select **Project navigator** window => **Hierarchy** in the drop down menu.
- b. Select Lab2\_1 top level unit
- c. Select **Processing** menu => **Start** => **Start Analysis and Elaboration**
- d. Be sure that you do not have Errors and Critical Warnings.

5 Close Quartus Prime.

## Launching ModelSim independently from Quartus Prime

Once you have completed the above sections you need to launch ModelSim for simulations independently from Quartus Prime.

- 1 Start ModelSim Intel FPGA Starter Edition.

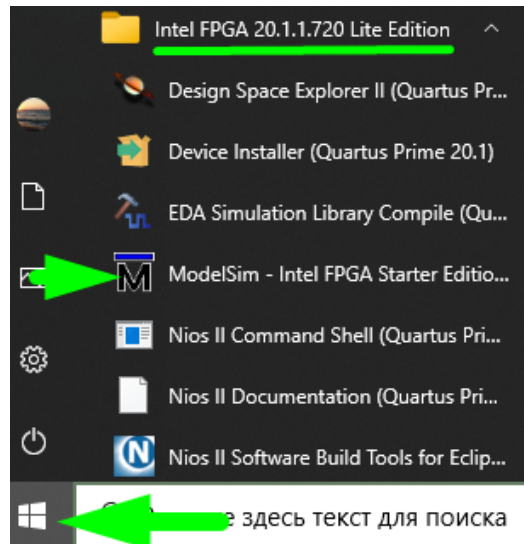


Figure 29

- 2 Change Directory.
  - a. Select File > Change Directory and change to the directory **C:\Intel\_trn\Quartus\_ModelSim\Lab2\_2**.
  - b. OR by typing in the Transcript window: **cd C:/Intel\_trn/Quartus\_ModelSim/Lab2\_2**
- 3 Create the working library.
  - a. Select File => New => Library => work.
  - b. OR by typing in the Transcript window: **vlib work**
- 4 Compile source files:
  - a. Select **Compile** => **Compile** => "DFF\_chain.v" "ROM\_8D.v" "tb\_Lab2\_2.v" "cnt\_adr.v" "cnt\_div.v" "Lab2\_2.v".
  - b. OR by typing in the Transcript window: **vlog -work work "DFF\_chain.v" "ROM\_8D.v" "tb\_Lab2\_2.v" "cnt\_adr.v" "cnt\_div.v" "Lab2\_2.v"**
- 5 Be sure that there are no any Errors. If there are ones you will need to correct source files.
- 6 Now you're ready to load the design into the simulator:
  - a. In the Library window, click the '+' sign next to the work library to show the files contained there and double-click **tb\_Lab2\_2** to load the design.
  - b. OR by typing in the Transcript window: **vsim work.tb\_Lab2\_2**
- 7 You will find two errors reported in the Transcript window

```

ModelSim> vsim work.tb_Lab2_2
# vsim work.tb_Lab2_2
# Start time: 19:31:56 on Feb 07,2021
# Loading work.tb_Lab2_2
# Loading work.Lab2_2
# Loading work.DFF_chain
# Loading work.cnt_div
# Loading work.cnt_adr
# ** Error: (vsim-3033) Instantiation of 'lpm_counter' failed. The design unit was not found.
#   Time: 0 ps  Iteration: 0  Instance: /tb_Lab2_2/DUT/cnt_adr_inst File: cnt_adr.v Line: 54
#   Searched libraries:
#       C:/Intel_trn/Quartus_MSim_Deb/Lab2_2/work
# Loading work.ROM_8D
# ** Error: (vsim-3033) Instantiation of 'altsyncram' failed. The design unit was not found.
#   Time: 0 ps  Iteration: 0  Instance: /tb_Lab2_2/DUT/ROM_8D_inst File: ROM_8D.v Line: 59
#   Searched libraries:
#       C:/Intel_trn/Quartus_MSim_Deb/Lab2_2/work
# Error loading design
# End time: 19:31:56 on Feb 07,2021, Elapsed time: 0:00:00
# Errors: 2, Warnings: 0

```

Figure 30

- 8 You need to reference lpm and altera\_mf libraries to fix the error.
  - a. Select Simulate menu => Start Simulation
  - b. In the window appeared the '+' sign next to the work library to show the files contained there and click **tb\_Lab2\_2** to select it.
  - c. Select **Libraries** folder
  - d. In the field Search Libraries click **Add...**
  - e. In the window appeared select **altera\_mf\_ver** library

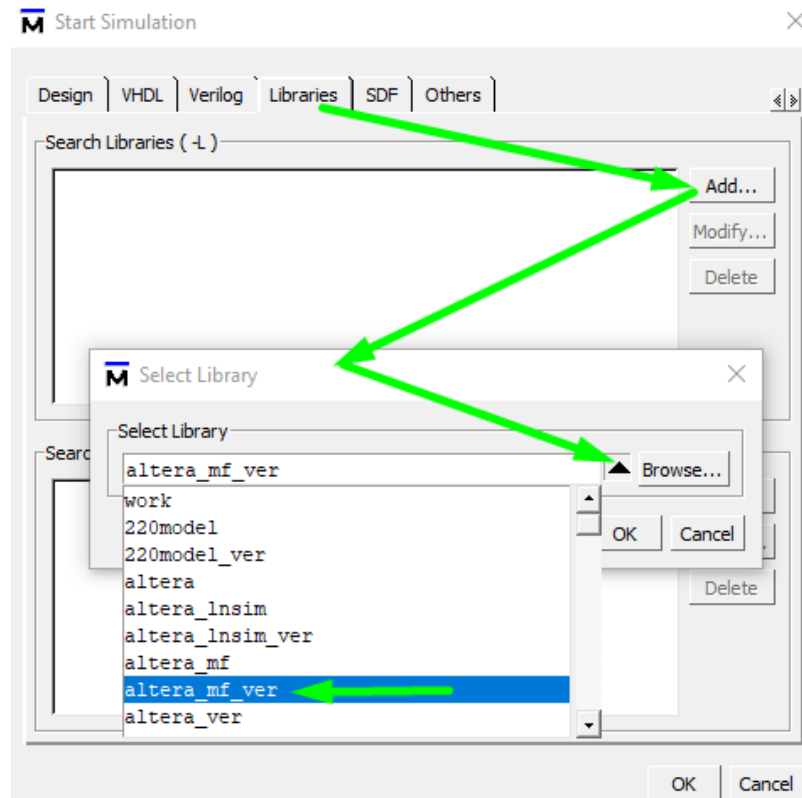
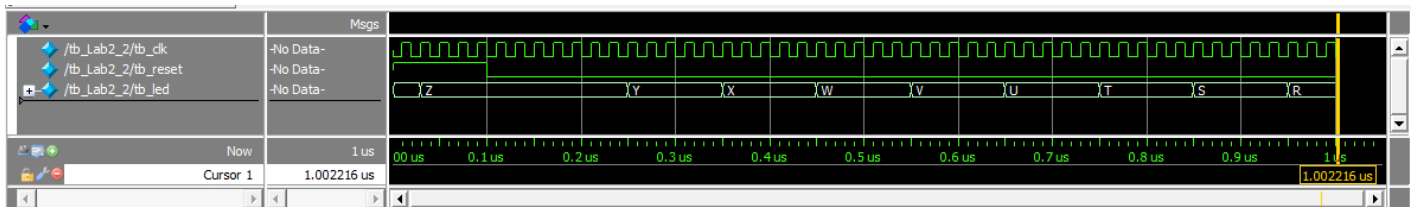


Figure 31

- f. Click OK
- g. Click **Add...** again
- h. Select **lpm\_ver** library.
- i. Click OK
- j. Click OK
- k. Simulation will start without any errors. If you still have some errors you need to check all steps from the beginning.
- l. Type in the transcript window the command: **do wave\_my\_lab2\_2.do**  
Wave window with top-level signals will appear.
- m. Type in the transcript window the command: **run 1000 ns**
- n. Undock wave window
- o. Zoom full wave window.
- p. Finally be sure that you see something like the Figure 32.



*Figure 32*

- 9 End simulation and close ModelSim window.

## Conclusions

With this lab you learnt how to use IP modules in simulation and ModelSim simulation independently from Quartus Prime.