ROM Simulation Tutorial

ECE 5440/6370

Instructor : Dr. Yuhua Chen

LOGO

Intruduction

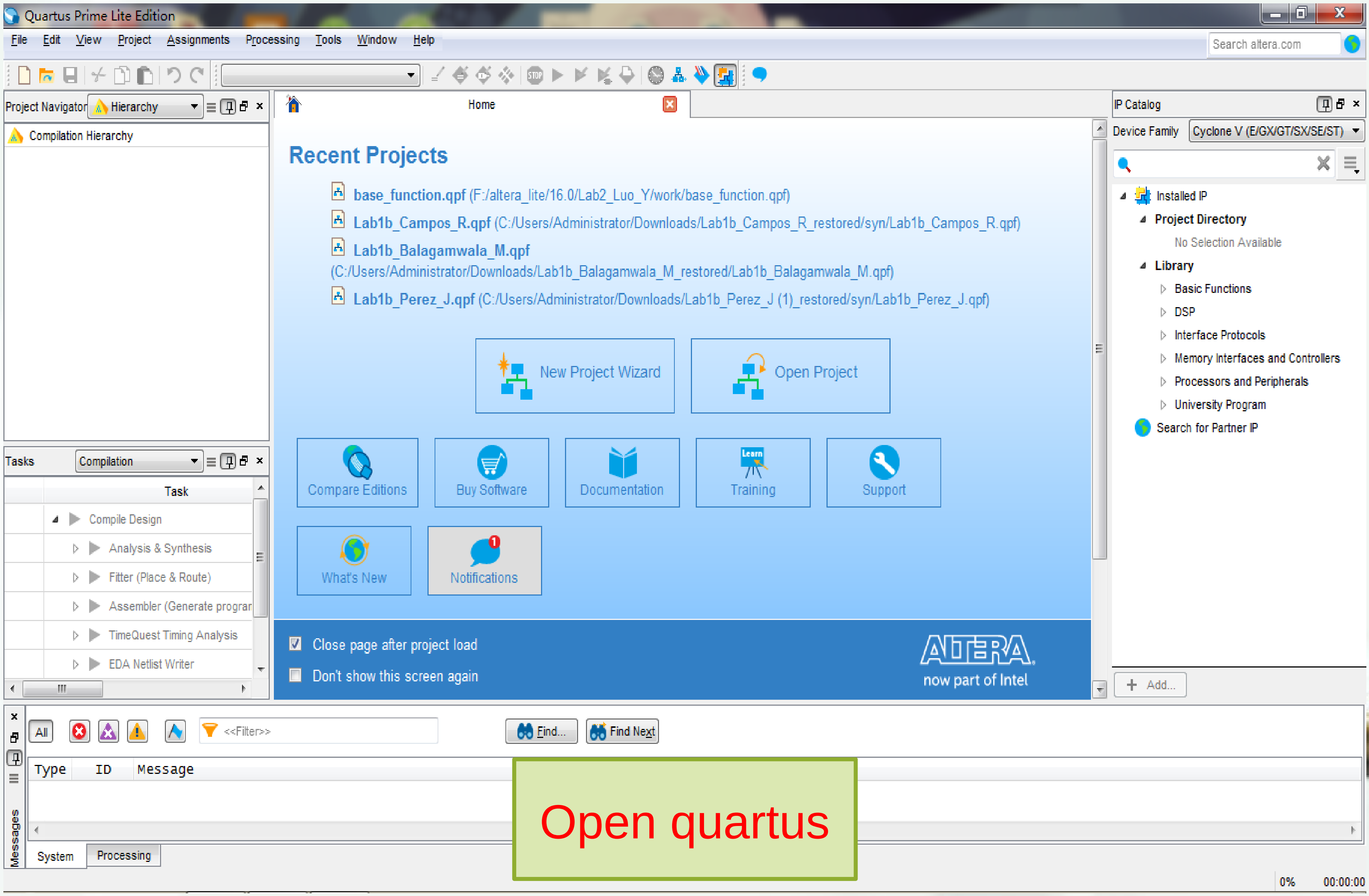
- When doing a quartus project, especially realizing it on the DE2-115 board, we may have to access the SDRAM, SRAM, EPROM and Flash Chips.
- How to access the ROM? The same approach is used to access the others.

Intruduction

- We are going to use Modelsim to run a simulation about ROM access:

- ROM initialization
- Creating a ROM component module
- Writing a testbench
- Configuring the libarary
- Simulating

ROM Initialization



Quartus Prime Lite Edition

File Edit View Project Assignments Processing Tools Window Help

Recent Projects

base_function.qpf (F:/altera_lite/16.0/Lab2_Luo_Y/work/base_function.qpf)

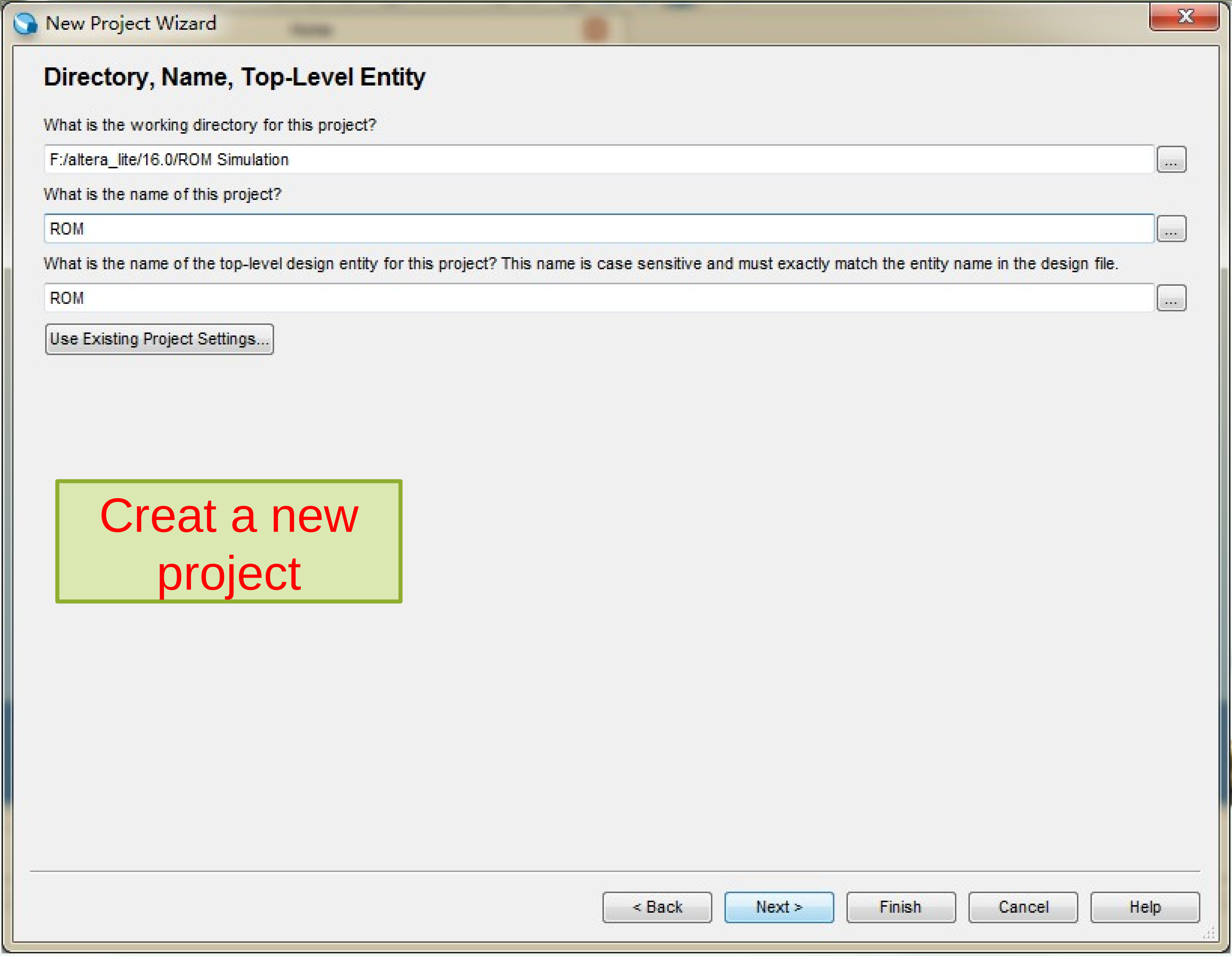
Lab1b_Campos_R.qpf (C:/Users/Administrator/Downloads/Lab1b_Campos_R_restored/syn/Lab1b_Campos_R.qpf)

Lab1b_Balagamwala_M.qpf (C:/Users/Administrator/Downloads/Lab1b_Balagamwala_M_restored/Lab1b_Balagamwala_M.qpf)

Lab1b_Perez_J.qpf (C:/Users/Administrator/Downloads/Lab1b_Perez_J (1)_restored/syn/Lab1b_Perez_J.qpf)

Open quartus

ROM Initialization


New Project Wizard

Directory, Name, Top-Level Entity

What is the working directory for this project?

F:/altera_lite/16.0/ROM Simulation

What is the name of this project?

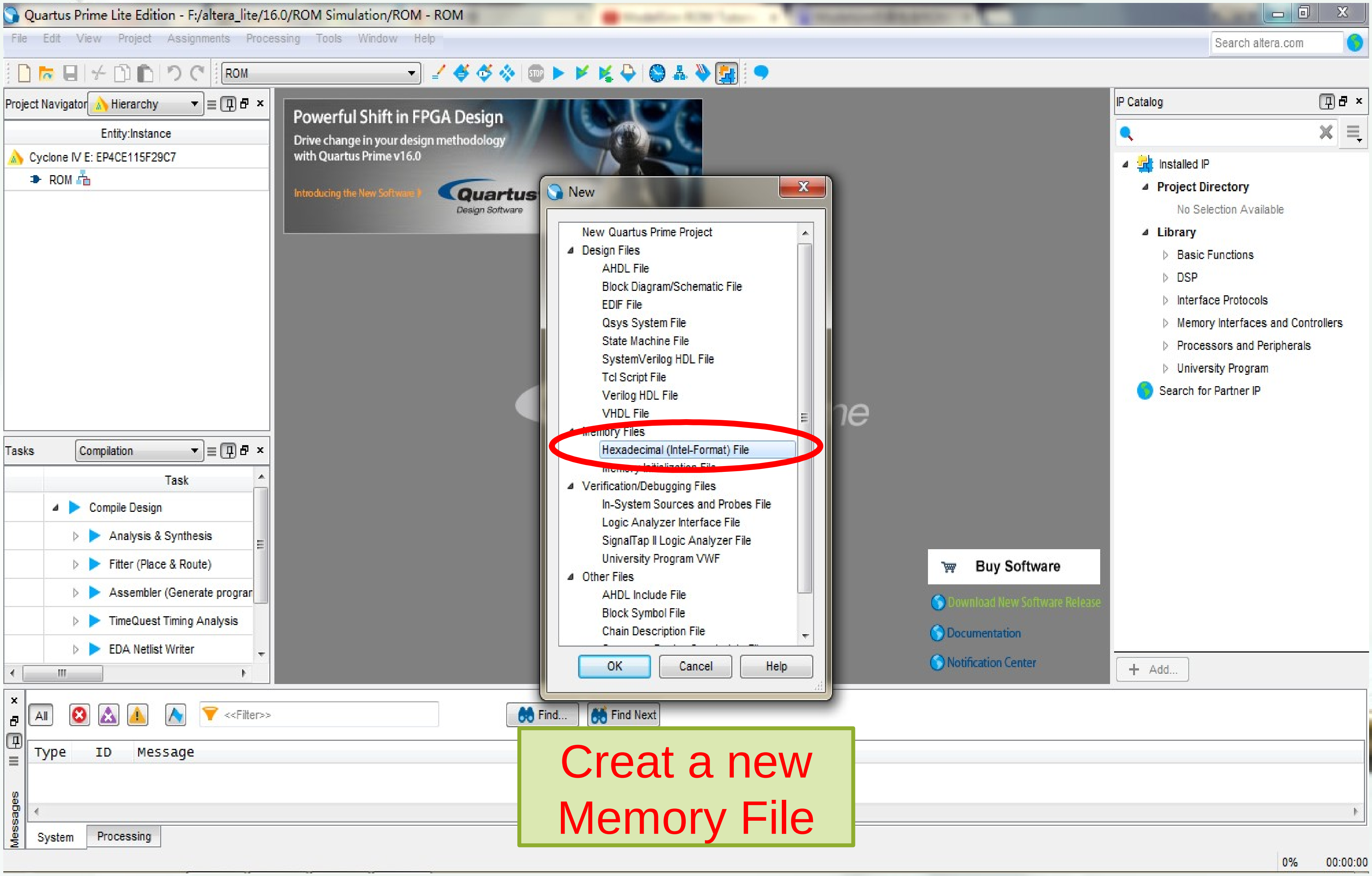
ROM

What is the name of the top-level design entity for this project? This name is case sensitive and must exactly match the entity name in the design file.

ROM

Creat a new project

ROM Initialization



Quartus Prime Lite Edition - F:/altera_lite/16.0/ROM Simulation/ROM - ROM

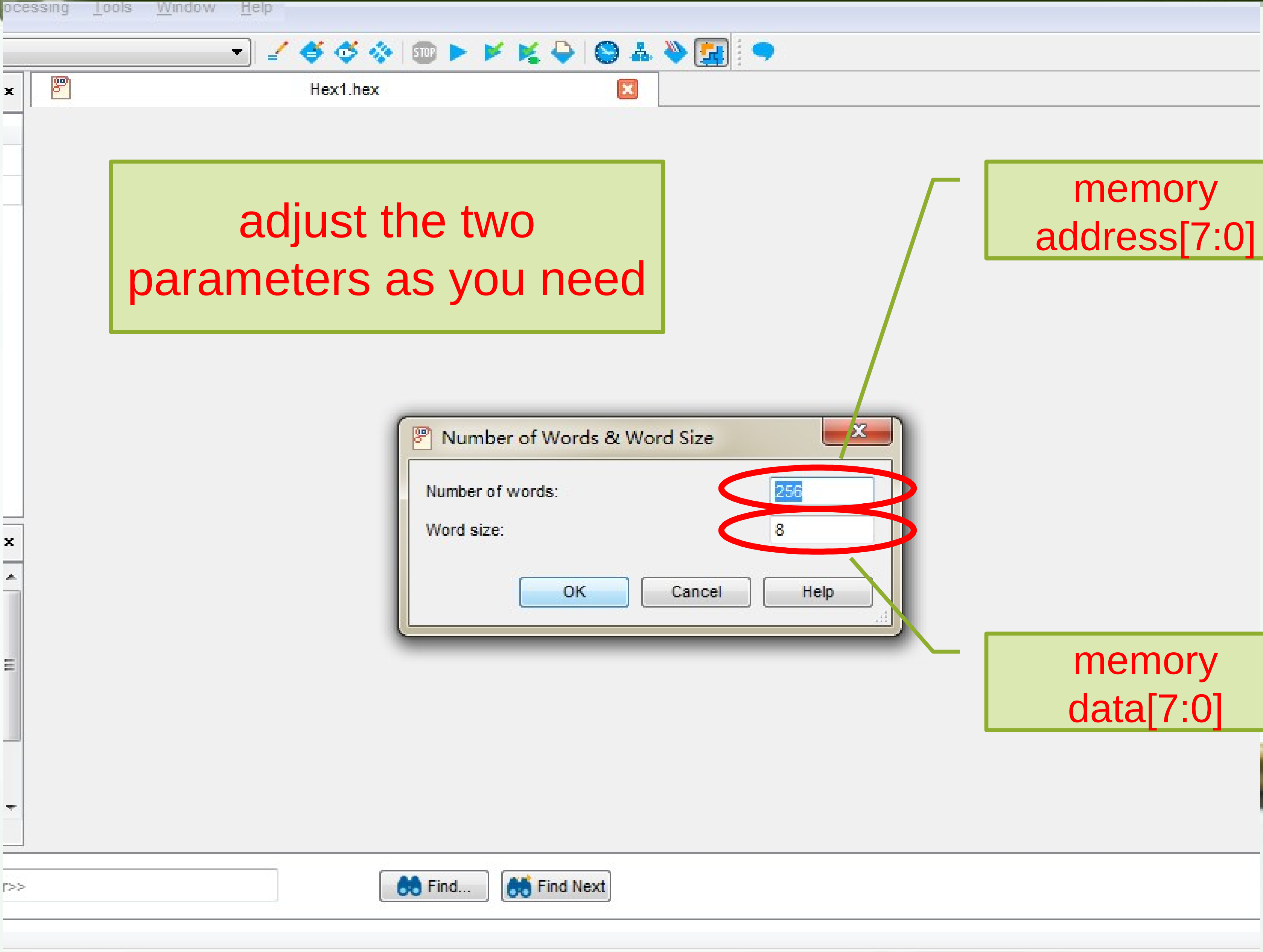
New

- New Quartus Prime Project
- Design Files
  - AHDL File
  - Block Diagram/Schematic File
  - EDIF File
  - Qsys System File
  - State Machine File
  - SystemVerilog HDL File
  - Tcl Script File
  - Verilog HDL File
  - VHDL File
- Memory Files
  - Hexadecimal (Intel-Format) File
  - Memory Initialization File
- Verification/Debugging Files
  - In-System Sources and Probes File
  - Logic Analyzer Interface File
  - SignalTap II Logic Analyzer File
  - University Program VWF
- Other Files
  - AHDL Include File
  - Block Symbol File
  - Chain Description File

OK Cancel Help

Creat a new Memory File

ROM Initialization



Number of Words & Word Size

Number of words: 256

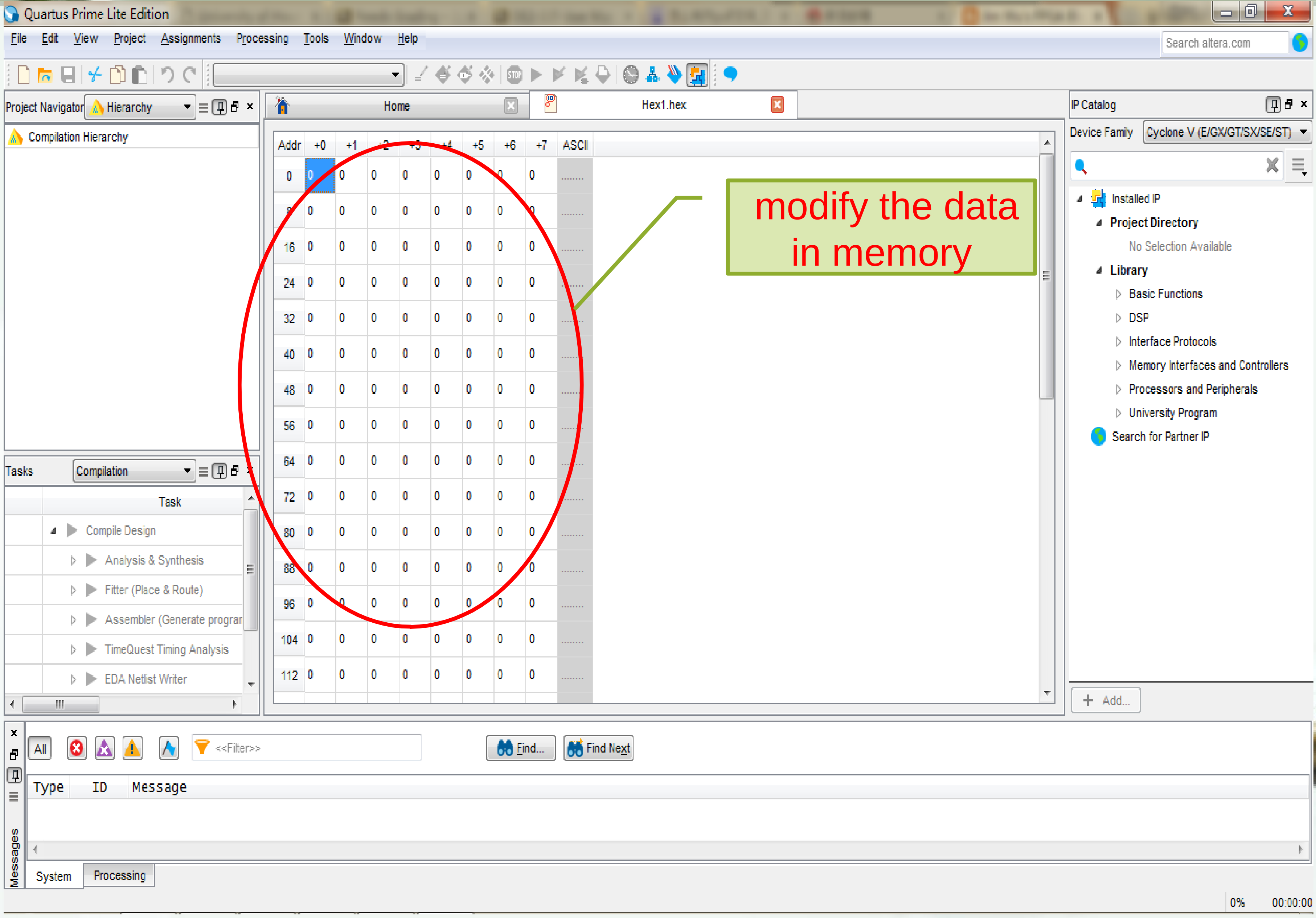
Word size: 8

adjust the two parameters as you need

memory address[7:0]

memory data[7:0]

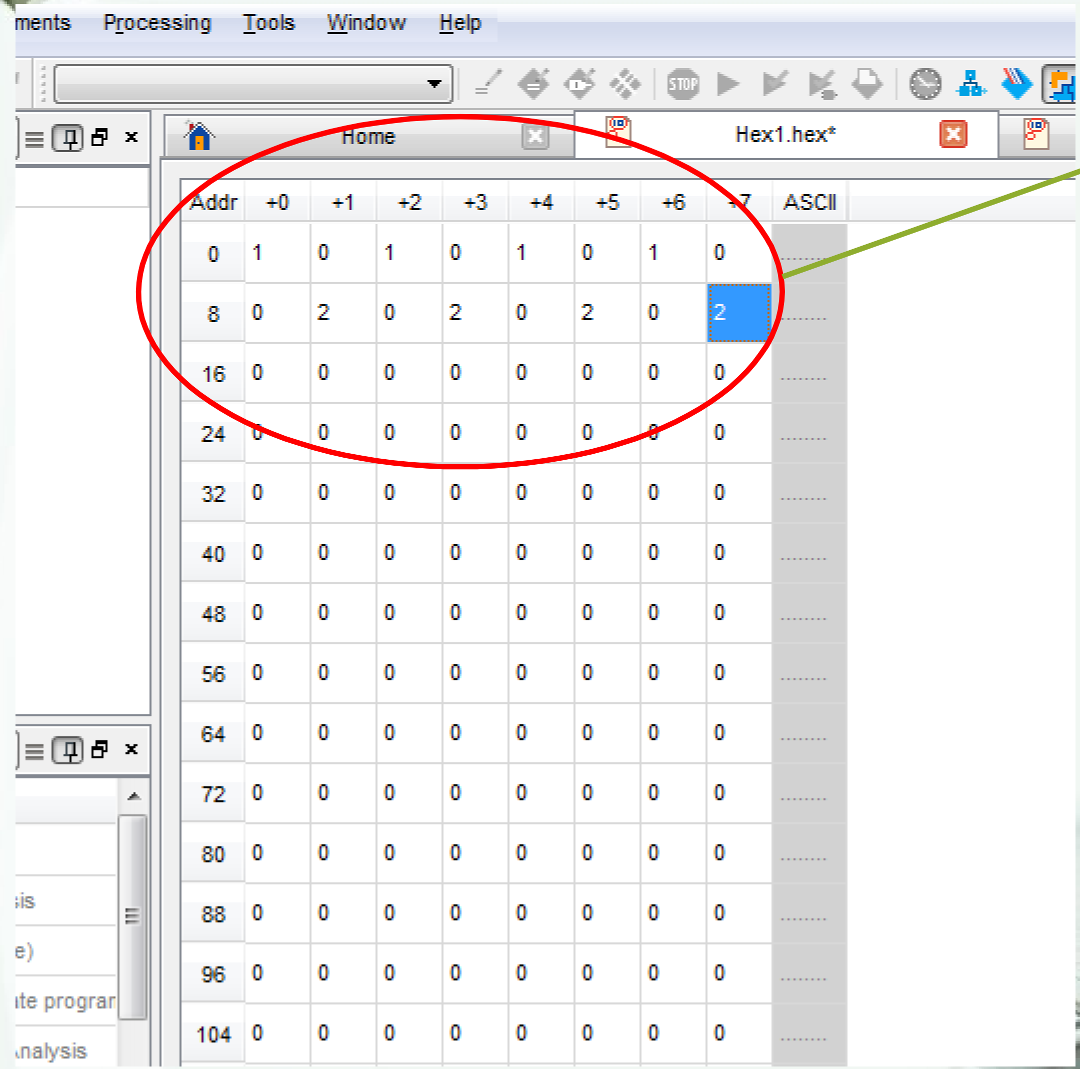
ROM Initialization



Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	0	0	0	0	0	0	0	0	....
8	0	0	0	0	0	0	0	0	....
16	0	0	0	0	0	0	0	0	....
24	0	0	0	0	0	0	0	0	....
32	0	0	0	0	0	0	0	0	....
40	0	0	0	0	0	0	0	0	....
48	0	0	0	0	0	0	0	0	....
56	0	0	0	0	0	0	0	0	....
64	0	0	0	0	0	0	0	0	....
72	0	0	0	0	0	0	0	0	....
80	0	0	0	0	0	0	0	0	....
88	0	0	0	0	0	0	0	0	....
96	0	0	0	0	0	0	0	0	....
104	0	0	0	0	0	0	0	0	....
112	0	0	0	0	0	0	0	0	....

modify the data in memory

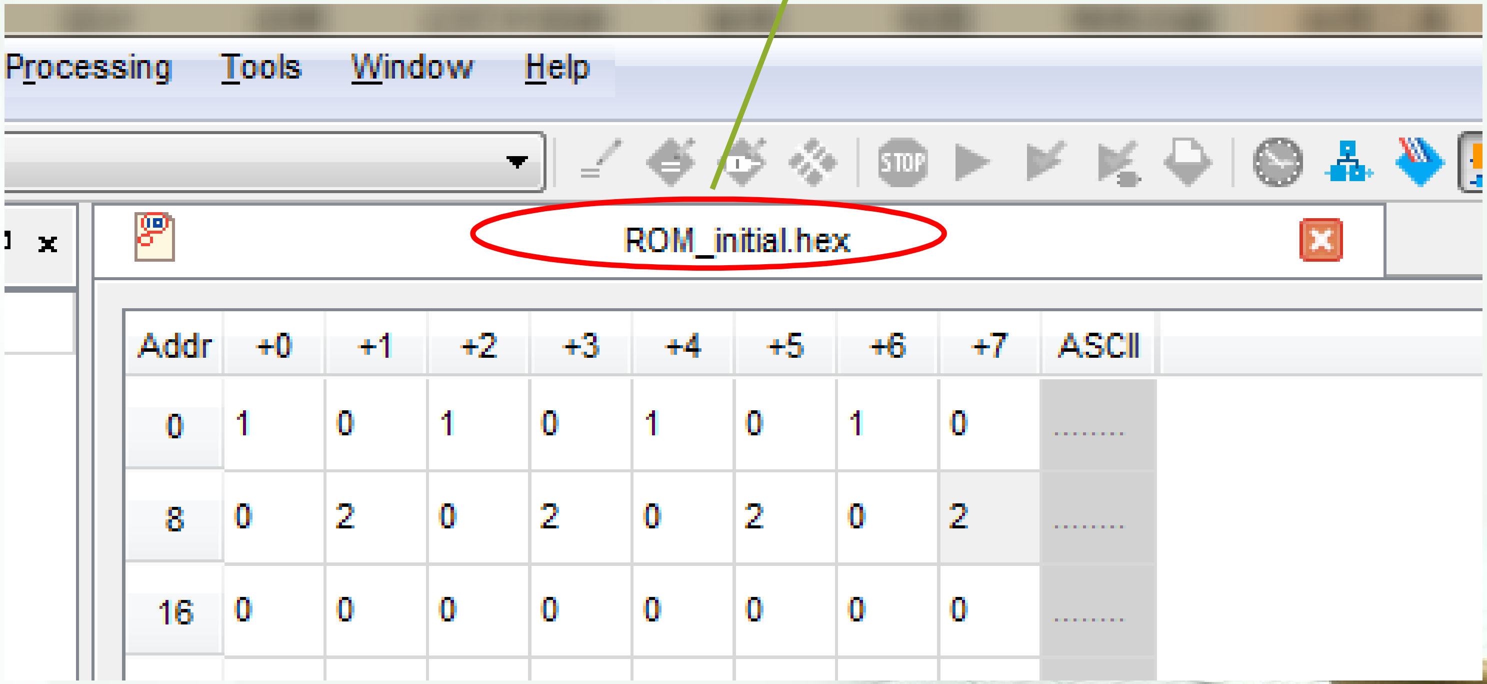
ROM Initialization



Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	1	0	1	0	1	0	1	0	....
8	0	2	0	2	0	2	0	2	....
16	0	0	0	0	0	0	0	0	....
24	0	0	0	0	0	0	0	0	....
32	0	0	0	0	0	0	0	0	....
40	0	0	0	0	0	0	0	0	....
48	0	0	0	0	0	0	0	0	....
56	0	0	0	0	0	0	0	0	....
64	0	0	0	0	0	0	0	0	....
72	0	0	0	0	0	0	0	0	....
80	0	0	0	0	0	0	0	0	....
88	0	0	0	0	0	0	0	0	....
96	0	0	0	0	0	0	0	0	....
104	0	0	0	0	0	0	0	0	....

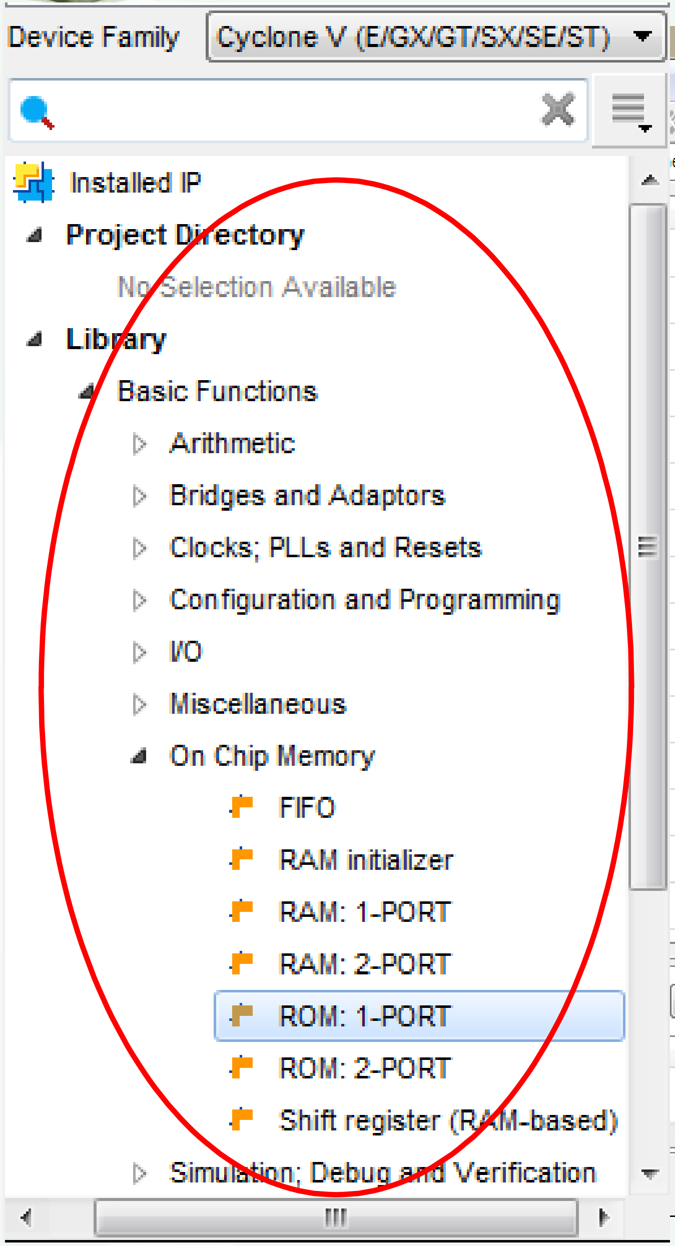
ROM Initialization

save as 'ROM_initial'



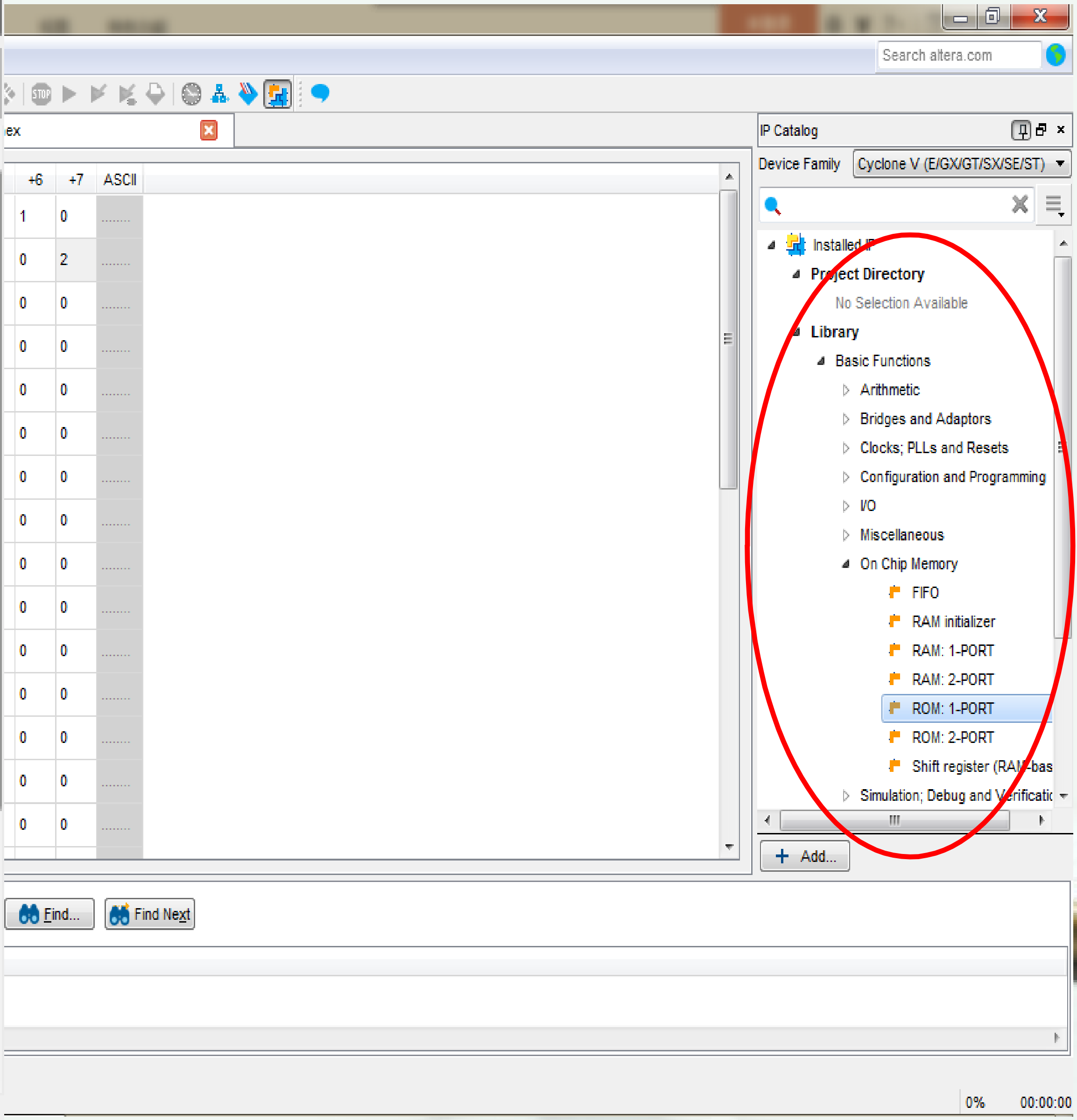
Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	1	0	1	0	1	0	1	0	....
8	0	2	0	2	0	2	0	2	....
16	0	0	0	0	0	0	0	0	....

Creating a ROM component module



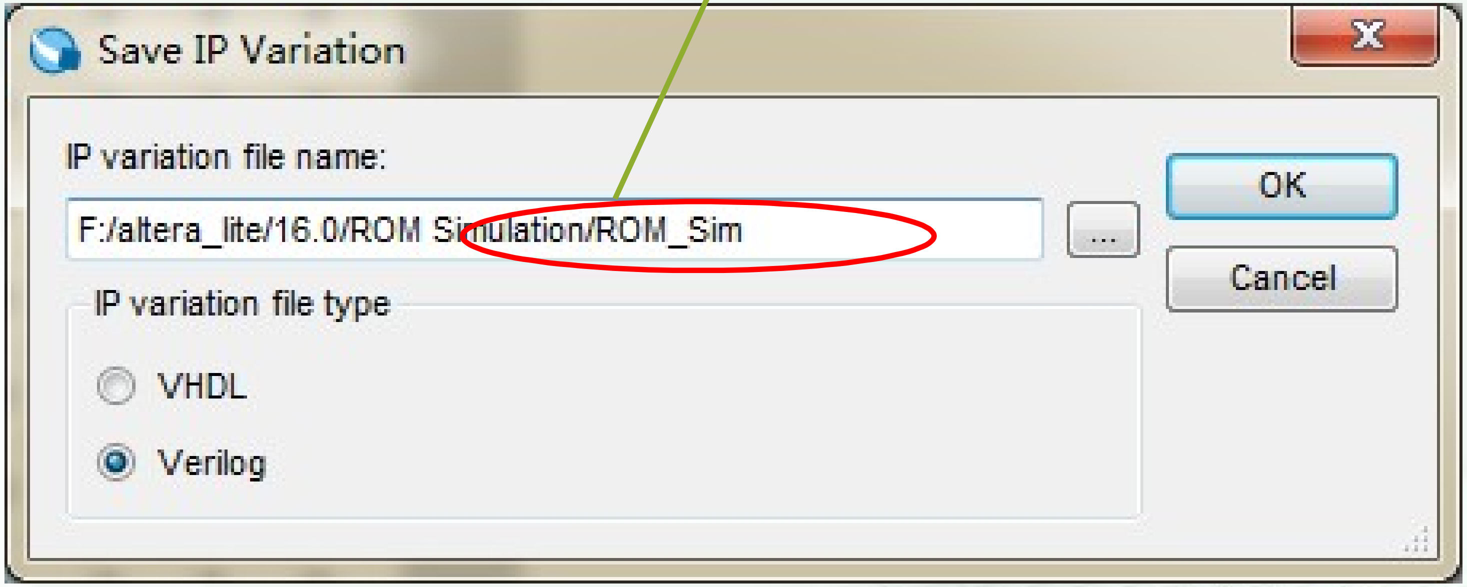
Device Family Cyclone V (E/GX/GT/SX/SE/ST)

- Installed IP
  - Project Directory
    - No Selection Available
  - Library
    - Basic Functions
      - Arithmetic
      - Bridges and Adaptors
      - Clocks; PLLs and Resets
      - Configuration and Programming
      - I/O
      - Miscellaneous
      - On Chip Memory
        - FIFO
        - RAM initializer
        - RAM: 1-PORT
        - RAM: 2-PORT
        - ROM: 1-PORT
        - ROM: 2-PORT
        - Shift register (RAM-based)
      - Simulation; Debug and Verification



Creating a ROM component module

component module 'ROM_Sim'



Save IP Variation

IP variation file name:

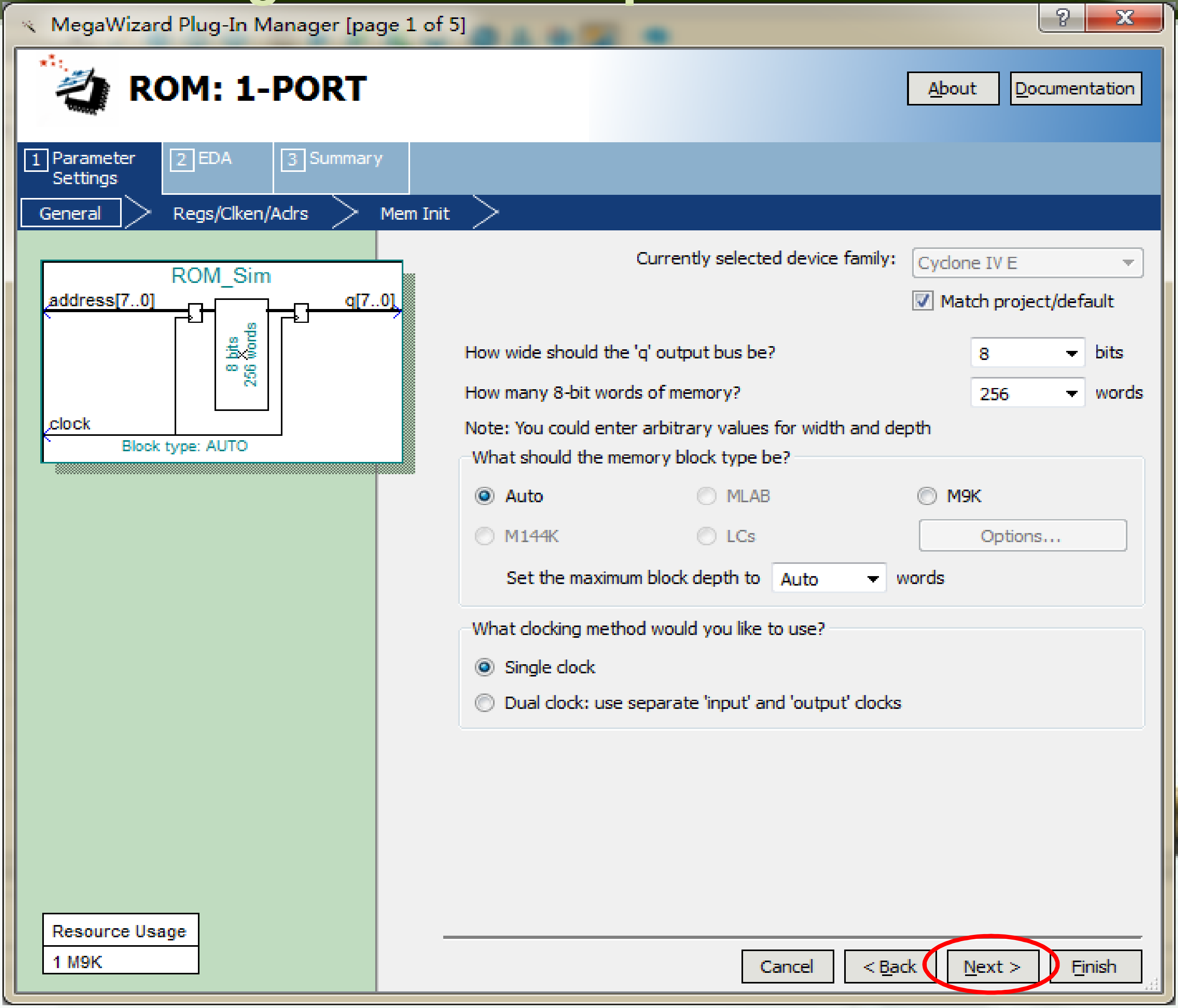
F:/altera_lite/16.0/ROM Simulation/ROM_Sim

IP variation file type

VHDL

[✓ Verilog]

Creating a ROM component module



MegaWizard Plug-In Manager [page 1 of 5]

ROM: 1-PORT

Currently selected device family: Cyclone IV E

[✓] Match project/default

How wide should the 'q' output bus be? 8 bits

How many 8-bit words of memory? 256 words

Note: You could enter arbitrary values for width and depth

What should the memory block type be?

[✓ Auto] MLAB M9K M144K LCs

Set the maximum block depth to Auto words

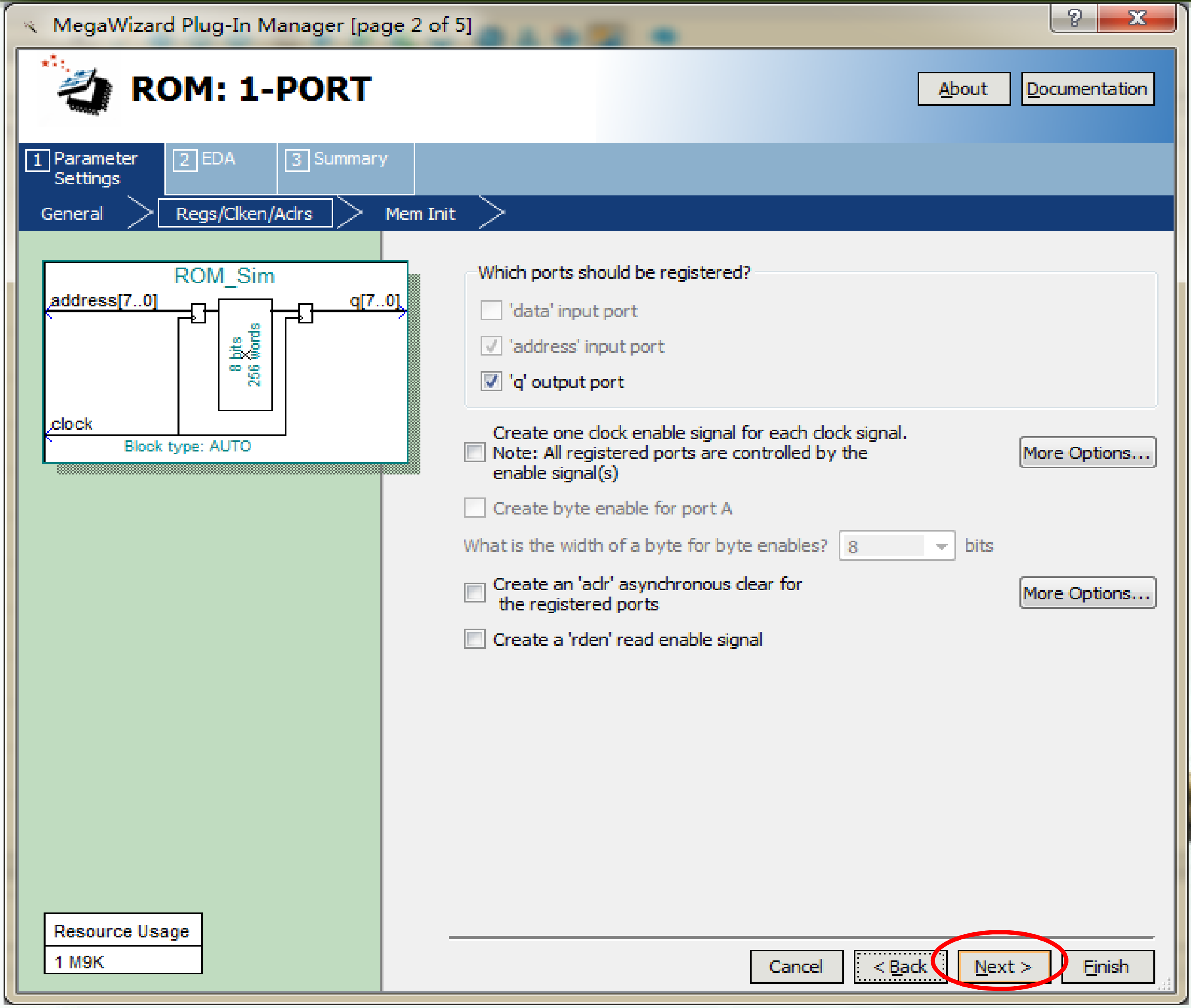
What clocking method would you like to use?

[✓ Single clock]

Dual clock: use separate 'input' and 'output' clocks

Resource Usage
1 M9K

Creating a ROM component module



MegaWizard Plug-In Manager [page 2 of 5]

ROM: 1-PORT

About | Documentation

1 Parameter Settings | 2 EDA | 3 Summary

General > Regs/Clken/Aclrs > Mem Init

Which ports should be registered?

- [ ] 'data' input port
- [✓] 'address' input port
- [✓] 'q' output port

- [ ] Create one clock enable signal for each clock signal. Note: All registered ports are controlled by the enable signal(s) — More Options...
- [ ] Create byte enable for port A

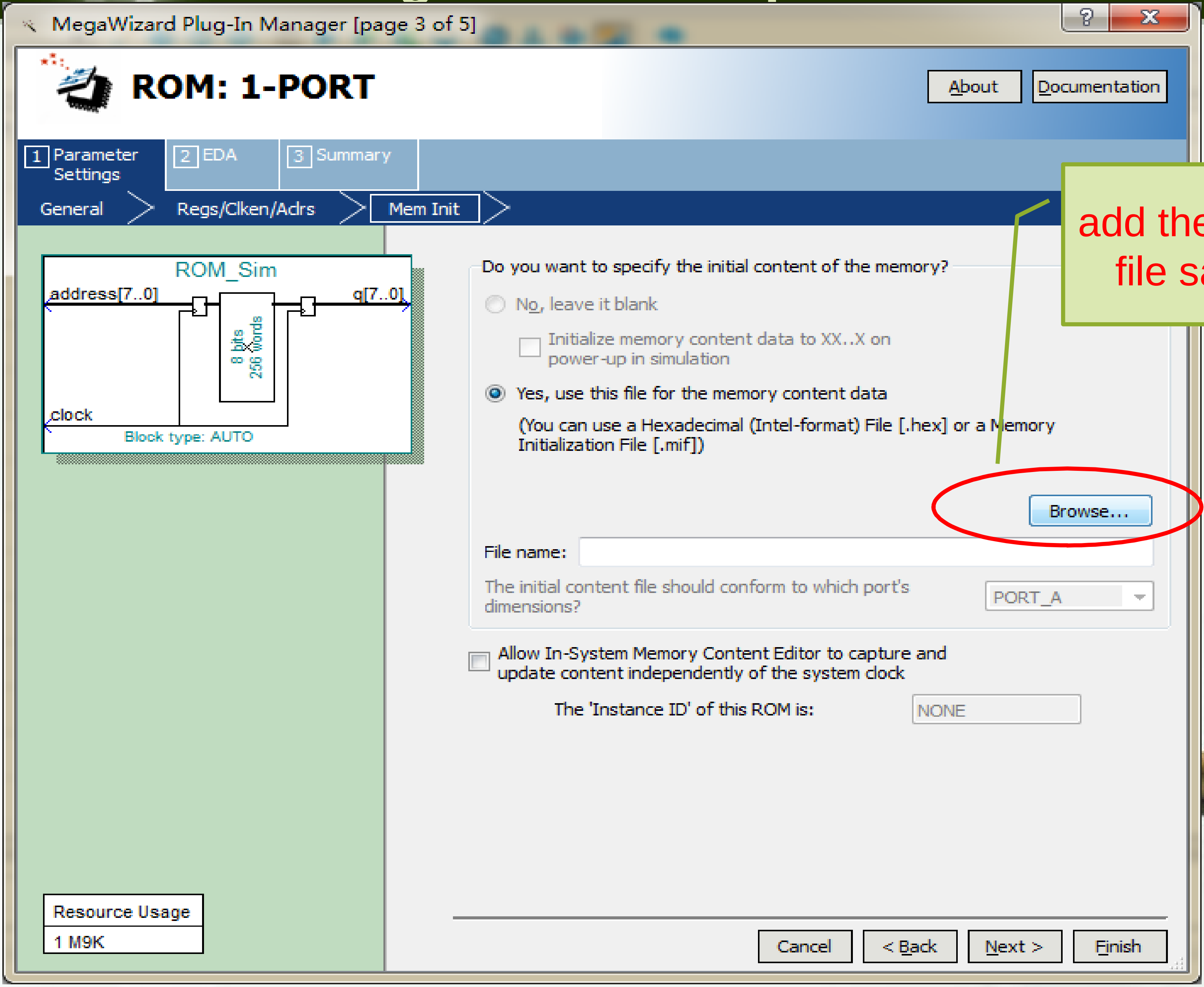
What is the width of a byte for byte enables? 8 bits

- [ ] Create an 'aclr' asynchronous clear for the registered ports — More Options...
- [ ] Create a 'rden' read enable signal

Resource Usage
1 M9K

Cancel | < Back | Next > | Finish

Creating a ROM component module



MegaWizard Plug-In Manager [page 3 of 5]

ROM: 1-PORT

Do you want to specify the initial content of the memory?

No, leave it blank

Initialize memory content data to XX..X on power-up in simulation

[✓ Yes, use this file for the memory content data]

(You can use a Hexadecimal (Intel-format) File [.hex] or a Memory Initialization File [.mif])

File name:

The initial content file should conform to which port's dimensions? PORT_A

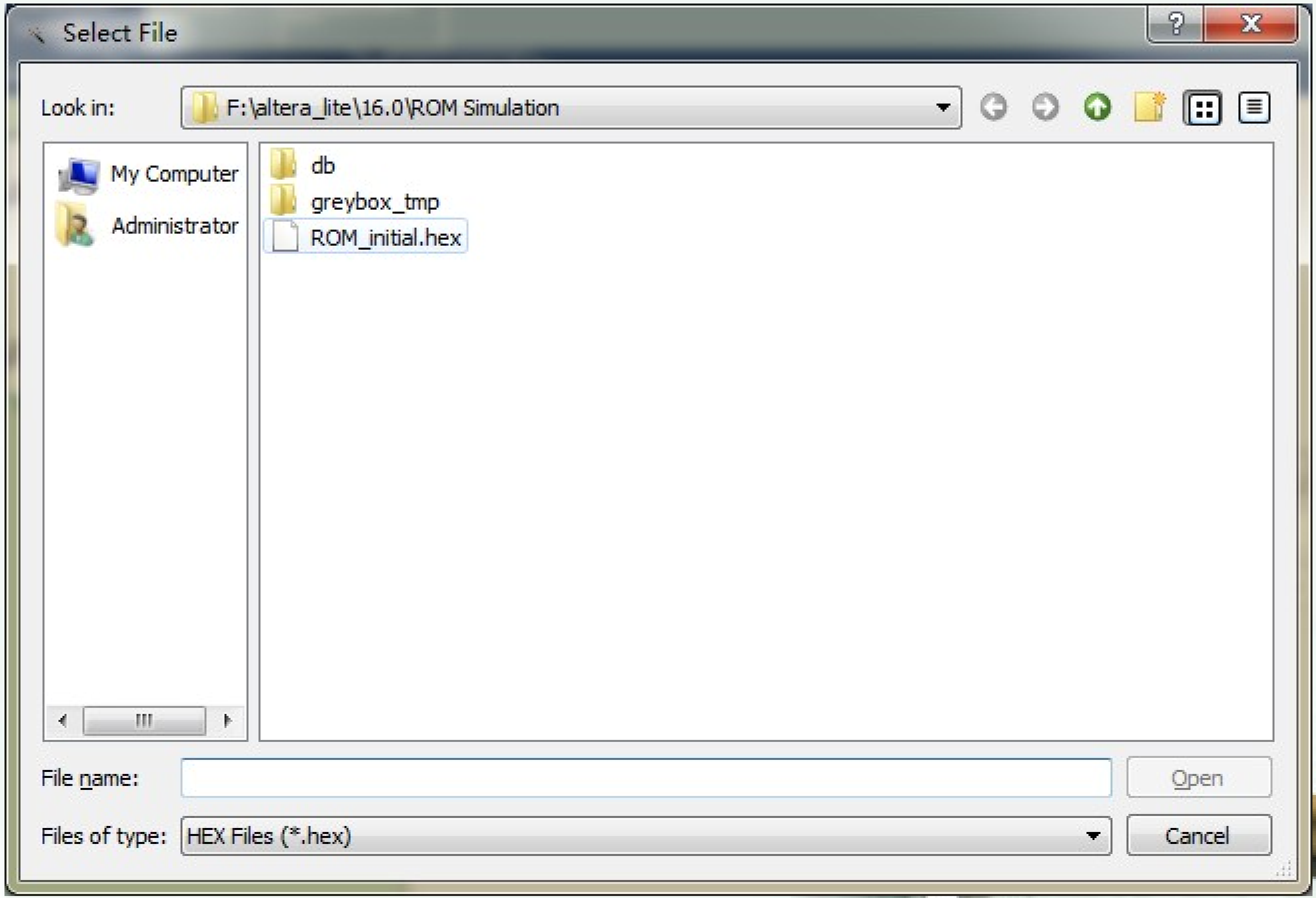
Allow In-System Memory Content Editor to capture and update content independently of the system clock

The 'Instance ID' of this ROM is: NONE

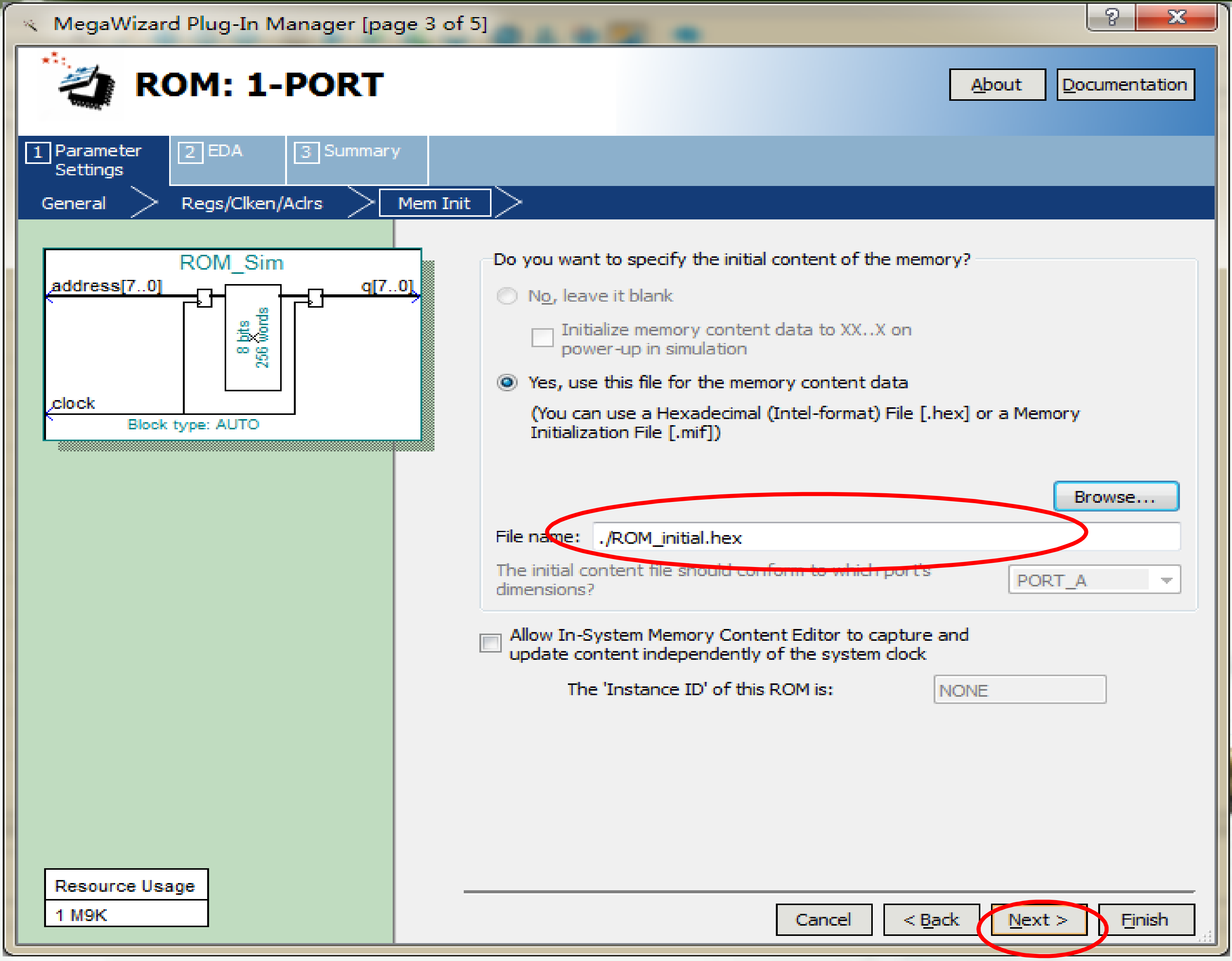
Resource Usage: 1 M9K

add the 'ROM_initial' file saved before

Creating a ROM component module



Creating a ROM component module



MegaWizard Plug-In Manager [page 3 of 5]

ROM: 1-PORT

About | Documentation

1 Parameter Settings | 2 EDA | 3 Summary

General > Regs/Clken/Adrs > Mem Init >

Do you want to specify the initial content of the memory?

- No, leave it blank
  - Initialize memory content data to XX..X on power-up in simulation
- [✓ Yes, use this file for the memory content data]

(You can use a Hexadecimal (Intel-format) File [.hex] or a Memory Initialization File [.mif])

Browse...

File name: ./ROM_initial.hex

The initial content file should conform to which port's dimensions? PORT_A

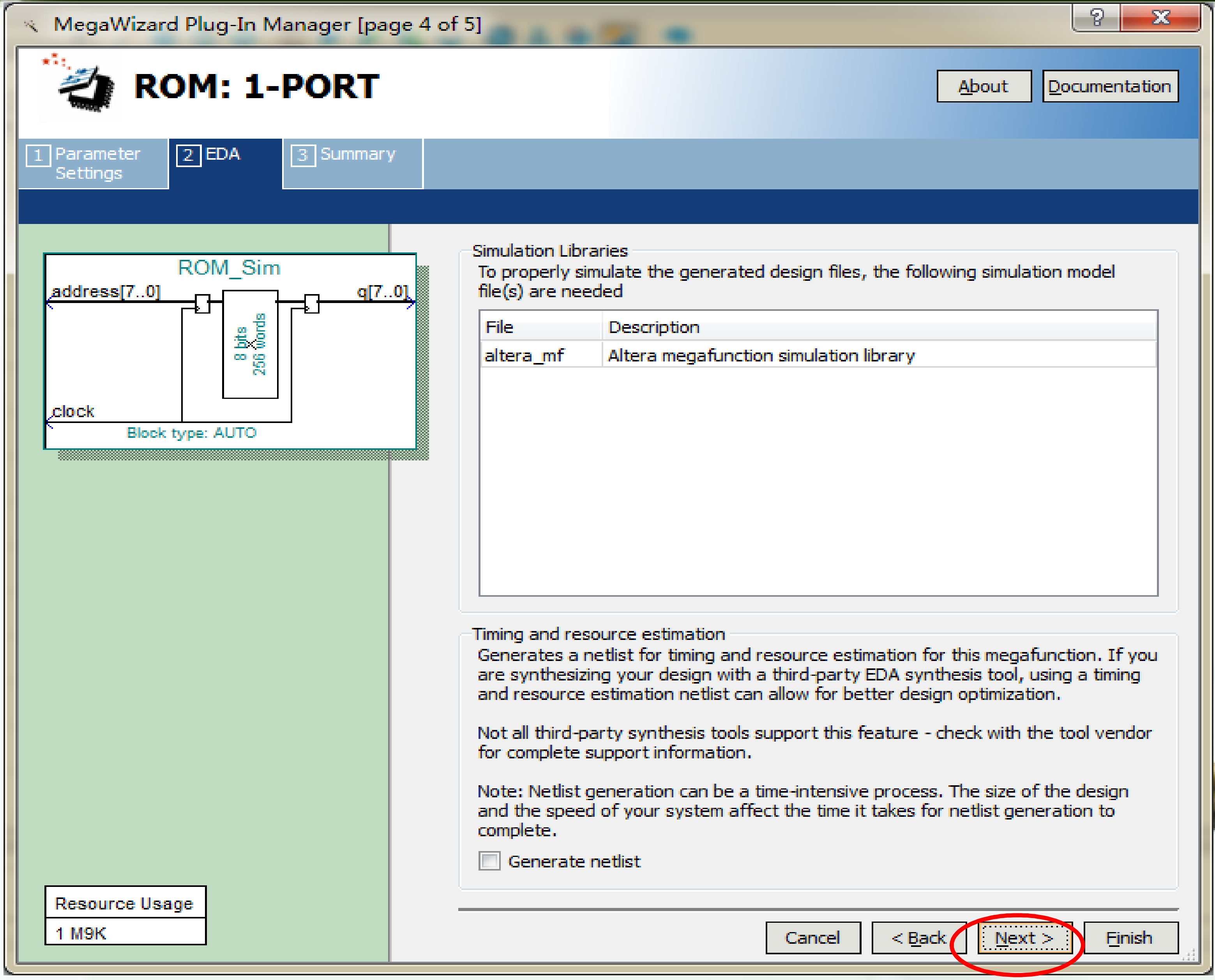
Allow In-System Memory Content Editor to capture and update content independently of the system clock

The 'Instance ID' of this ROM is: NONE

Resource Usage
1 M9K

Cancel | < Back | Next > | Finish

Creating a ROM component module



MegaWizard Plug-In Manager [page 4 of 5]

ROM: 1-PORT

About | Documentation

1 Parameter Settings | 2 EDA | 3 Summary

Simulation Libraries

To properly simulate the generated design files, the following simulation model file(s) are needed

File	Description
altera_mf	Altera megafunction simulation library

Timing and resource estimation

Generates a netlist for timing and resource estimation for this megafunction. If you are synthesizing your design with a third-party EDA synthesis tool, using a timing and resource estimation netlist can allow for better design optimization.

Not all third-party synthesis tools support this feature - check with the tool vendor for complete support information.

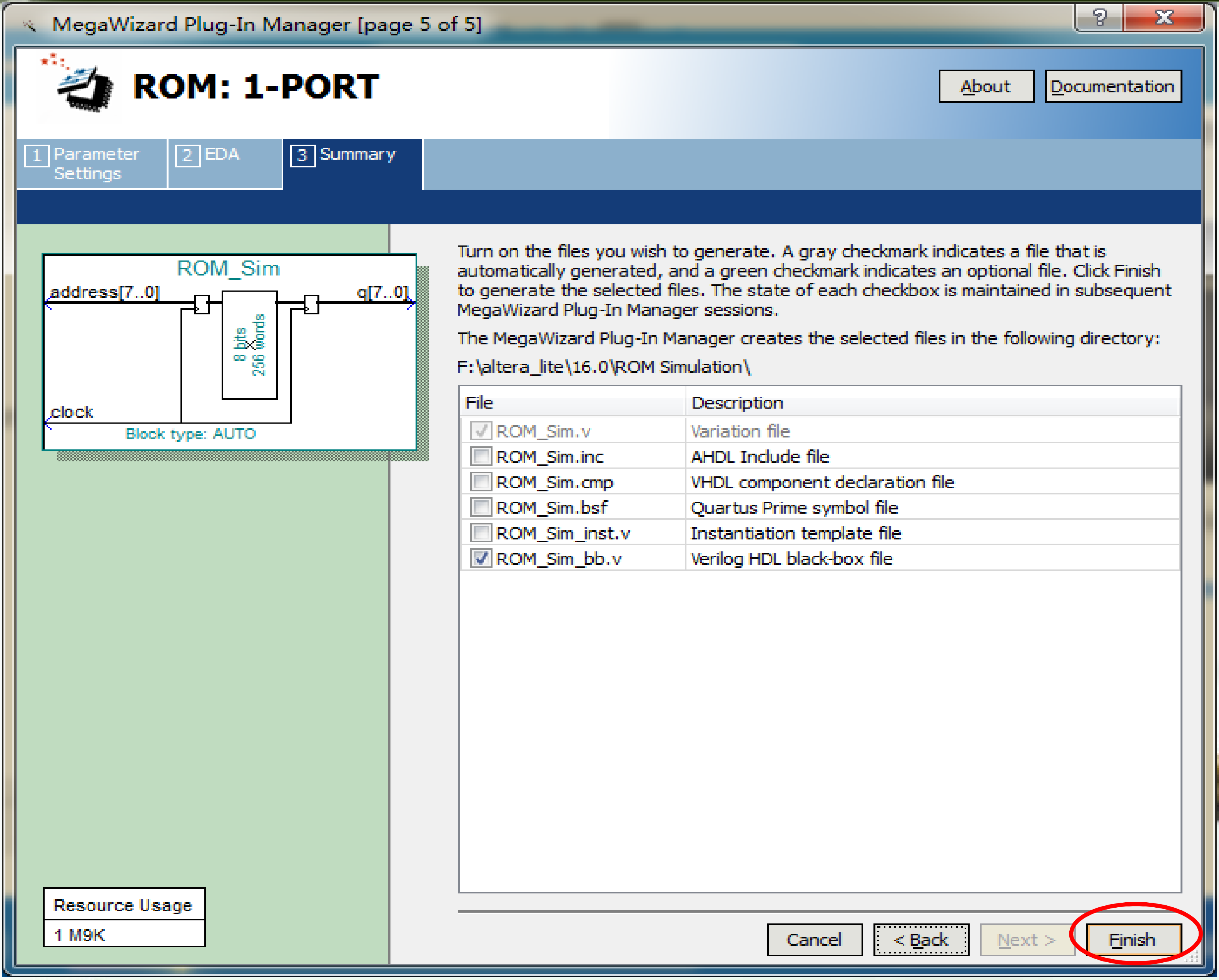
Note: Netlist generation can be a time-intensive process. The size of the design and the speed of your system affect the time it takes for netlist generation to complete.

☐ Generate netlist

Resource Usage
1 M9K

Cancel | < Back | Next > | Finish

Creating a ROM component module



MegaWizard Plug-In Manager [page 5 of 5]

ROM: 1-PORT

Resource Usage

1 M9K

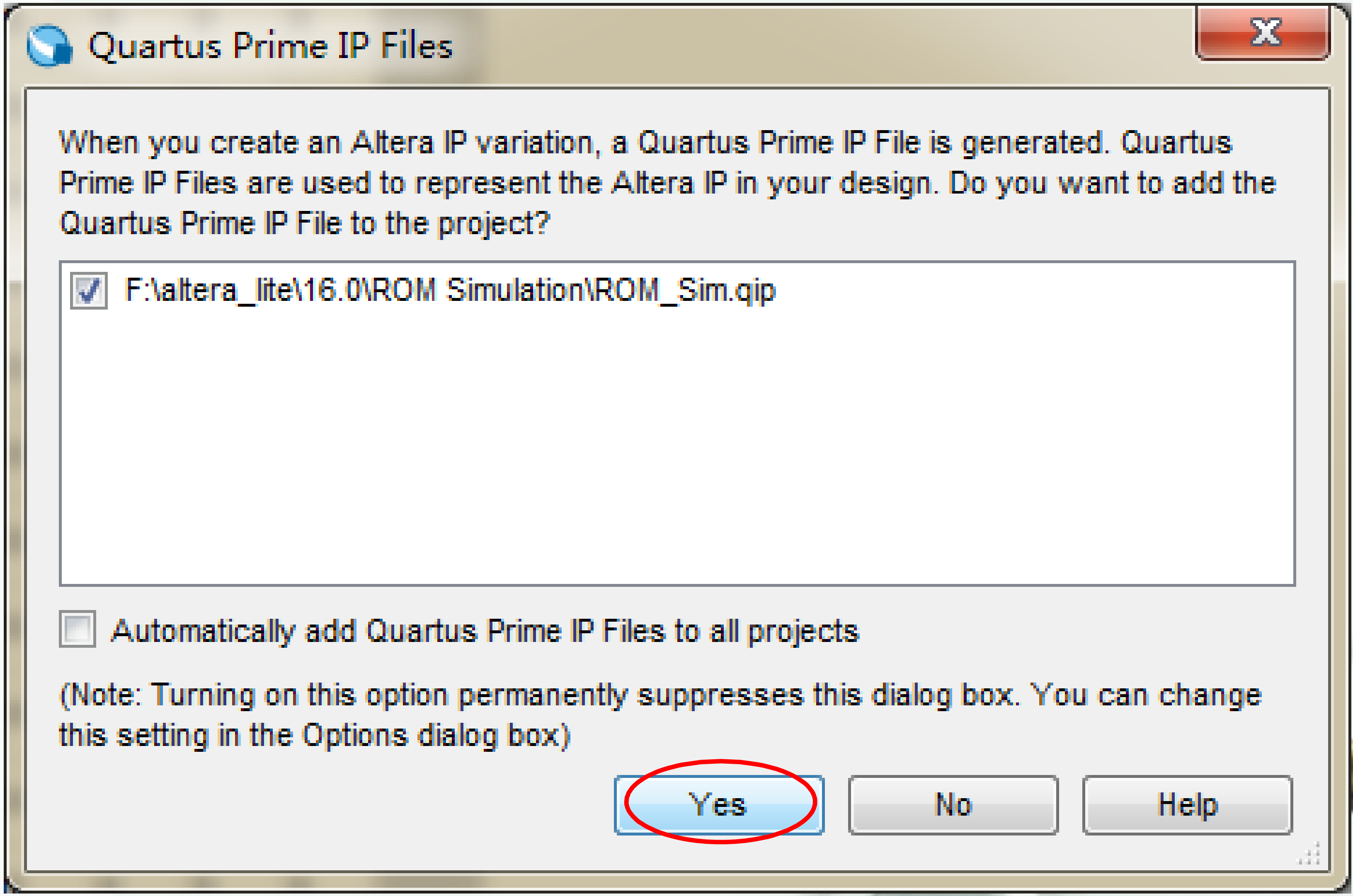
Turn on the files you wish to generate. A gray checkmark indicates a file that is automatically generated, and a green checkmark indicates an optional file. Click Finish to generate the selected files. The state of each checkbox is maintained in subsequent MegaWizard Plug-In Manager sessions.

The MegaWizard Plug-In Manager creates the selected files in the following directory:

F:\altera_lite\16.0\ROM Simulation\

File	Description
✓ ROM_Sim.v	Variation file
☐ ROM_Sim.inc	AHDL Include file
☐ ROM_Sim.cmp	VHDL component declaration file
☐ ROM_Sim.bsf	Quartus Prime symbol file
☐ ROM_Sim_inst.v	Instantiation template file
✓ ROM_Sim_bb.v	Verilog HDL black-box file

Creating a ROM component module


Quartus Prime IP Files

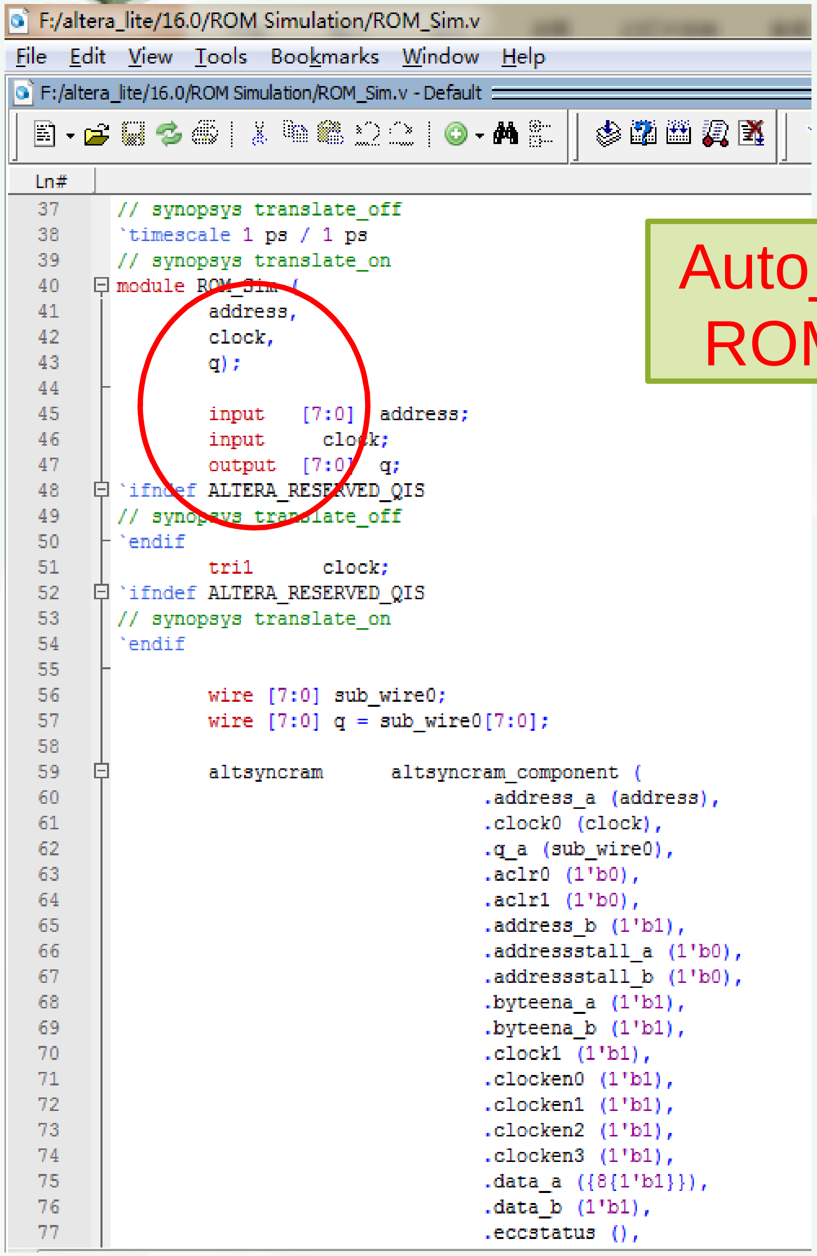
When you create an Altera IP variation, a Quartus Prime IP File is generated. Quartus Prime IP Files are used to represent the Altera IP in your design. Do you want to add the Quartus Prime IP File to the project?

- [✓] F:\altera_lite\16.0\ROM Simulation\ROM_Sim.qip

[ ] Automatically add Quartus Prime IP Files to all projects

(Note: Turning on this option permanently suppresses this dialog box. You can change this setting in the Options dialog box)

Creating a ROM component module



F:/altera_lite/16.0/ROM Simulation/ROM_Sim.v

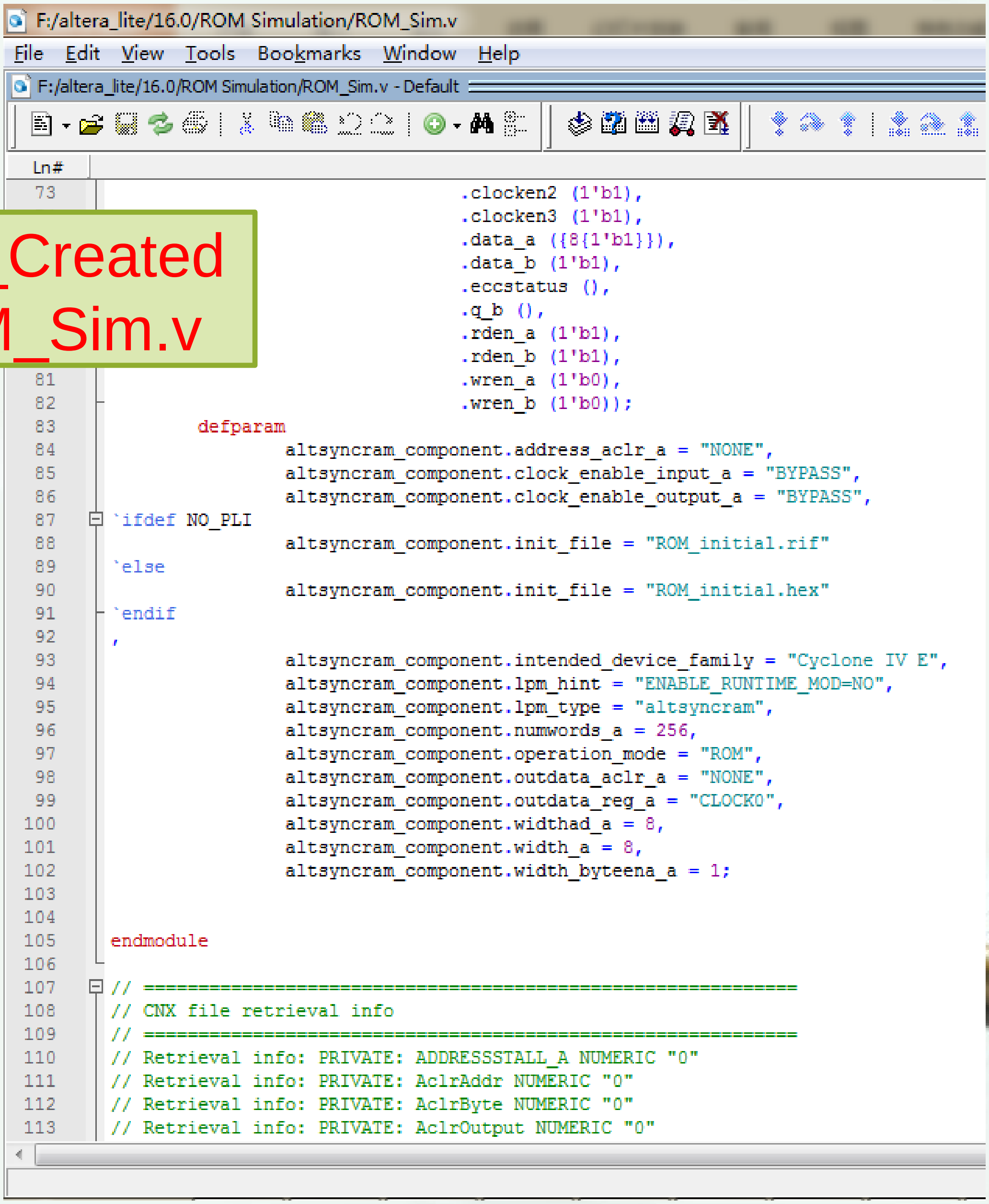
```
// synopsys translate_off
`timescale 1 ps / 1 ps
// synopsys translate_on
module ROM_Sim (
        address,
        clock,
        q);

        input   [7:0]  address;
        input          clock;
        output  [7:0]  q;
`ifndef ALTERA_RESERVED_QIS
// synopsys translate_off
`endif
        tri1           clock;
`ifndef ALTERA_RESERVED_QIS
// synopsys translate_on
`endif

        wire [7:0] sub_wire0;
        wire [7:0] q = sub_wire0[7:0];

        altsyncram      altsyncram_component (
                                .address_a (address),
                                .clock0 (clock),
                                .q_a (sub_wire0),
                                .aclr0 (1'b0),
                                .aclr1 (1'b0),
                                .address_b (1'b1),
                                .addressstall_a (1'b0),
                                .addressstall_b (1'b0),
                                .byteena_a (1'b1),
                                .byteena_b (1'b1),
                                .clock1 (1'b1),
                                .clocken0 (1'b1),
                                .clocken1 (1'b1),
                                .clocken2 (1'b1),
                                .clocken3 (1'b1),
                                .data_a ({8{1'b1}}),
                                .data_b (1'b1),
                                .eccstatus (),
```

Auto_Created ROM_Sim.v



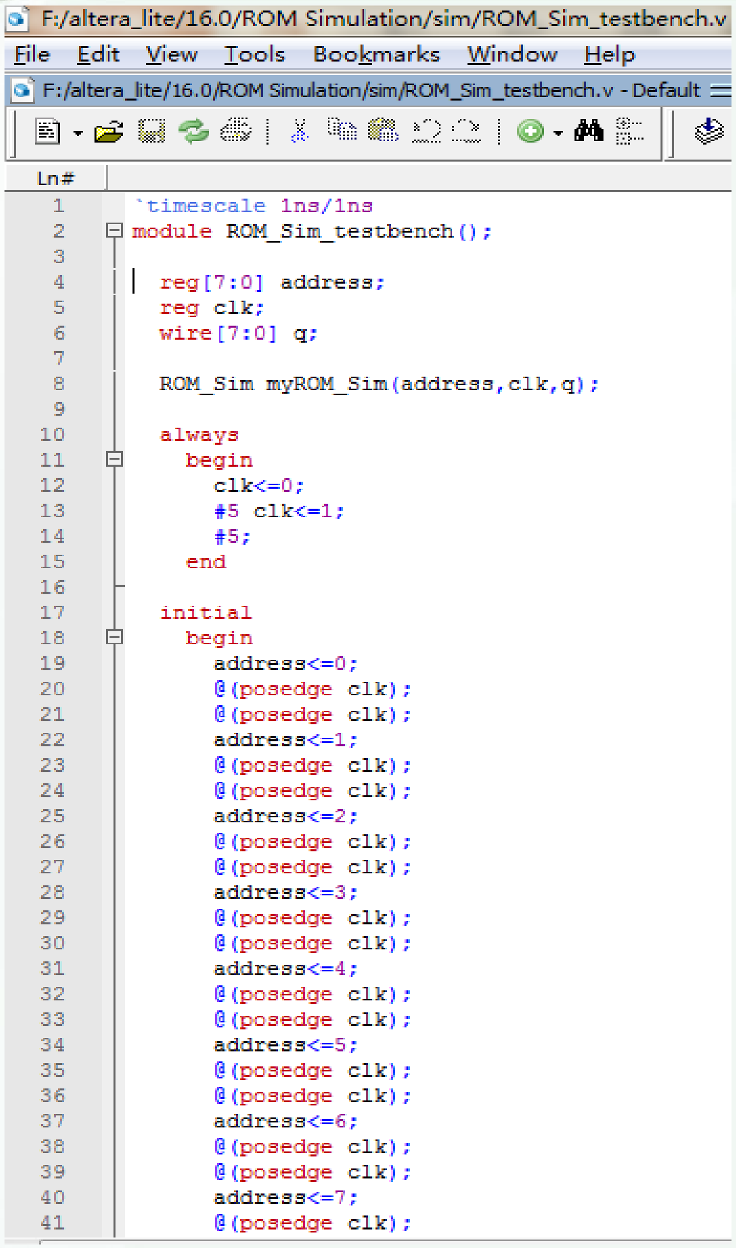
F:/altera_lite/16.0/ROM Simulation/ROM_Sim.v

```
                                .clocken2 (1'b1),
                                .clocken3 (1'b1),
                                .data_a ({8{1'b1}}),
                                .data_b (1'b1),
                                .eccstatus (),
                                .q_b (),
                                .rden_a (1'b1),
                                .rden_b (1'b1),
                                .wren_a (1'b0),
                                .wren_b (1'b0));
        defparam
                altsyncram_component.address_aclr_a = "NONE",
                altsyncram_component.clock_enable_input_a = "BYPASS",
                altsyncram_component.clock_enable_output_a = "BYPASS",
`ifdef NO_PLI
                altsyncram_component.init_file = "ROM_initial.rif"
`else
                altsyncram_component.init_file = "ROM_initial.hex"
`endif
,
                altsyncram_component.intended_device_family = "Cyclone IV E",
                altsyncram_component.lpm_hint = "ENABLE_RUNTIME_MOD=NO",
                altsyncram_component.lpm_type = "altsyncram",
                altsyncram_component.numwords_a = 256,
                altsyncram_component.operation_mode = "ROM",
                altsyncram_component.outdata_aclr_a = "NONE",
                altsyncram_component.outdata_reg_a = "CLOCK0",
                altsyncram_component.widthad_a = 8,
                altsyncram_component.width_a = 8,
                altsyncram_component.width_byteena_a = 1;


endmodule

// ============================================================
// CNX file retrieval info
// ============================================================
// Retrieval info: PRIVATE: ADDRESSSTALL_A NUMERIC "0"
// Retrieval info: PRIVATE: AclrAddr NUMERIC "0"
// Retrieval info: PRIVATE: AclrByte NUMERIC "0"
// Retrieval info: PRIVATE: AclrOutput NUMERIC "0"
```

Writing a testbench


F:/altera_lite/16.0/ROM Simulation/sim/ROM_Sim_testbench.v
File Edit View Tools Bookmarks Window Help
F:/altera_lite/16.0/ROM Simulation/sim/ROM_Sim_testbench.v - Default
Ln#

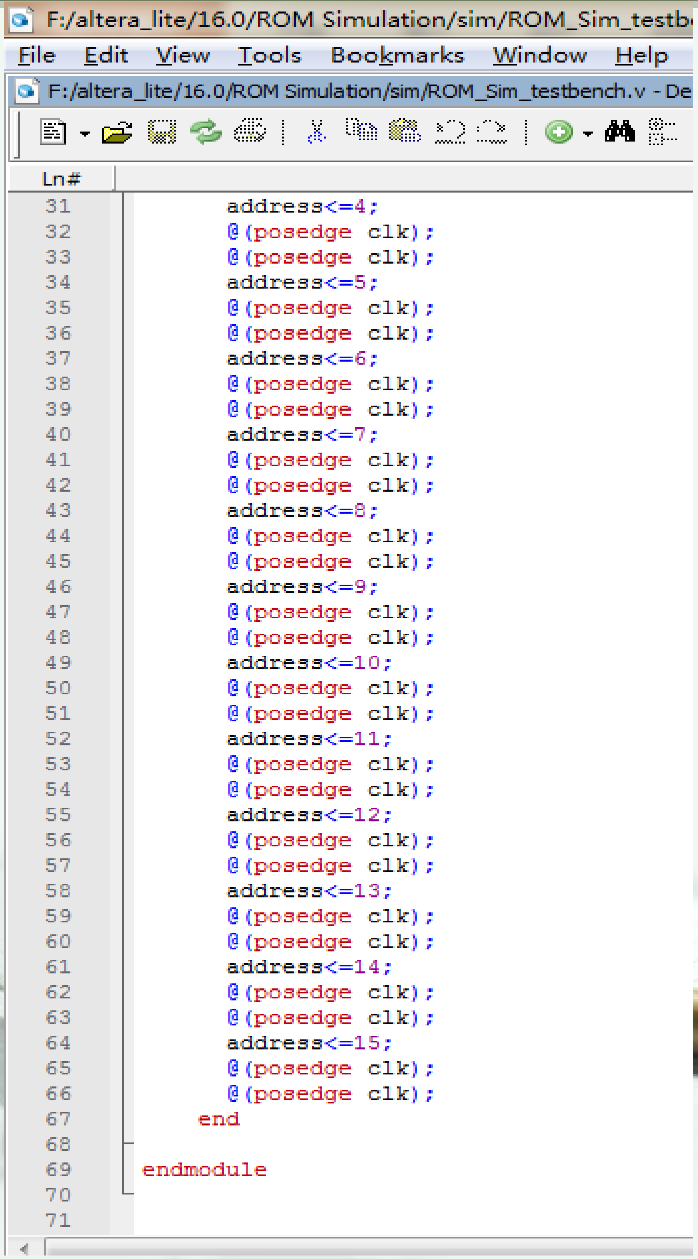
```verilog
`timescale 1ns/1ns
module ROM_Sim_testbench();

  reg[7:0] address;
  reg clk;
  wire[7:0] q;

  ROM_Sim myROM_Sim(address,clk,q);

  always
    begin
      clk<=0;
      #5 clk<=1;
      #5;
    end

  initial
    begin
      address<=0;
      @(posedge clk);
      @(posedge clk);
      address<=1;
      @(posedge clk);
      @(posedge clk);
      address<=2;
      @(posedge clk);
      @(posedge clk);
      address<=3;
      @(posedge clk);
      @(posedge clk);
      address<=4;
      @(posedge clk);
      @(posedge clk);
      address<=5;
      @(posedge clk);
      @(posedge clk);
      address<=6;
      @(posedge clk);
      @(posedge clk);
      address<=7;
      @(posedge clk);
```


F:/altera_lite/16.0/ROM Simulation/sim/ROM_Sim_testb
File Edit View Tools Bookmarks Window Help
F:/altera_lite/16.0/ROM Simulation/sim/ROM_Sim_testbench.v - De
Ln#

```verilog
      address<=4;
      @(posedge clk);
      @(posedge clk);
      address<=5;
      @(posedge clk);
      @(posedge clk);
      address<=6;
      @(posedge clk);
      @(posedge clk);
      address<=7;
      @(posedge clk);
      @(posedge clk);
      address<=8;
      @(posedge clk);
      @(posedge clk);
      address<=9;
      @(posedge clk);
      @(posedge clk);
      address<=10;
      @(posedge clk);
      @(posedge clk);
      address<=11;
      @(posedge clk);
      @(posedge clk);
      address<=12;
      @(posedge clk);
      @(posedge clk);
      address<=13;
      @(posedge clk);
      @(posedge clk);
      address<=14;
      @(posedge clk);
      @(posedge clk);
      address<=15;
      @(posedge clk);
      @(posedge clk);
    end

endmodule
```

Writing a testbench

```
`timescale 1ns/1ns

module ROM_Sim_testbench();

    reg[7:0] address;
    reg clk;
    wire[7:0] q;

    ROM_Sim myROM_Sim(address,clk,q);

    always
      begin
      clk<=0;
      #5 clk<=1;
      #5;
    end
```

Writing a testbench

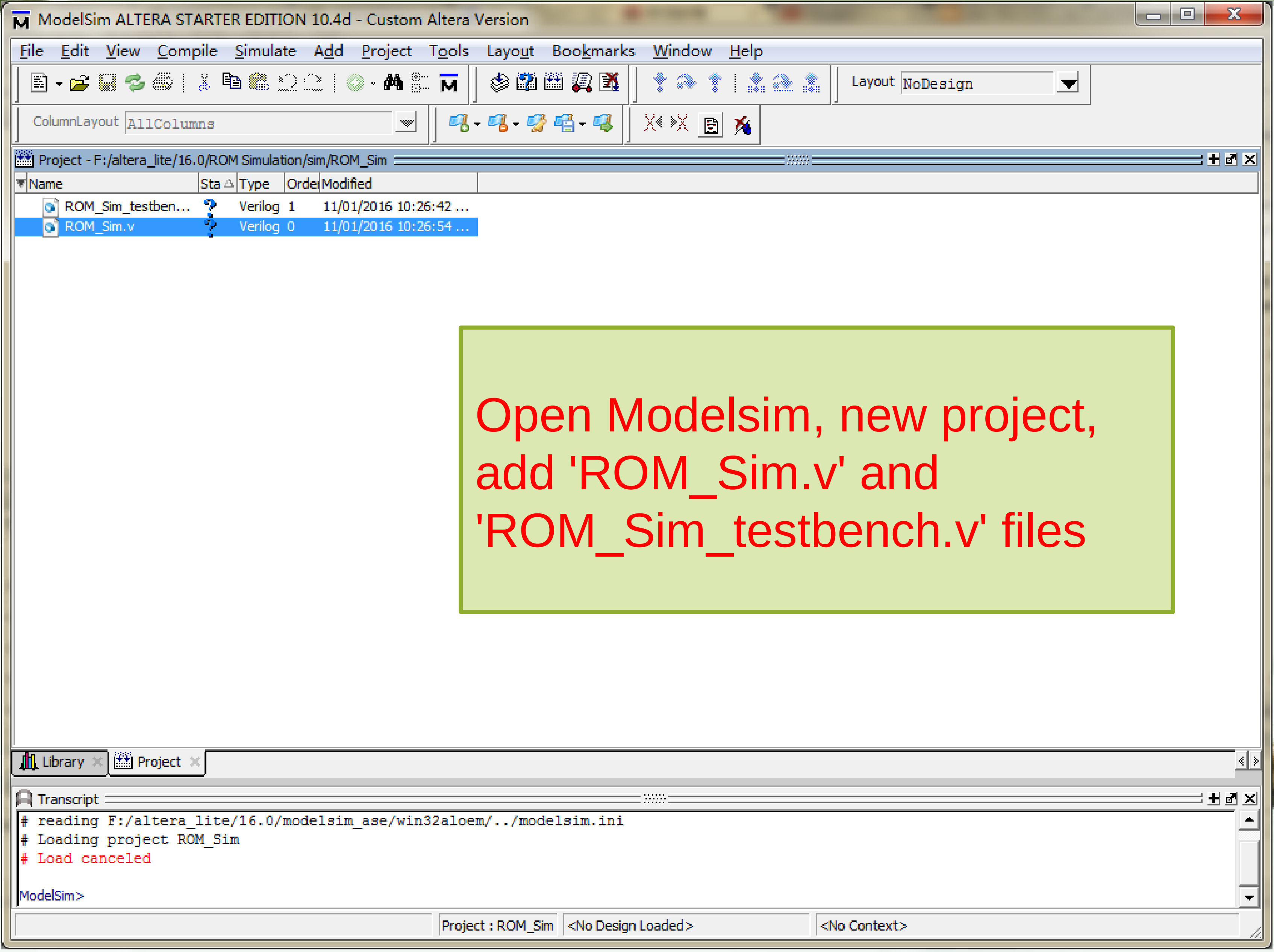
```
initial
    begin
      address<=0;
      @(posedge clk);
      @(posedge clk);
      address<=1;
      @(posedge clk);
      @(posedge clk);
      address<=2;
      ......
      ......
      address<=15;
      @(posedge clk);
      @(posedge clk);
    end

endmodule
```

In the 'ROM_initial' file, we have initialized the memory data addressed from 0 to 15.

So, in the testbench we wanna read the memory data, see whether it the same as the 'ROM_initial' file.

Configuring the libraries



ModelSim ALTERA STARTER EDITION 10.4d - Custom Altera Version
File Edit View Compile Simulate Add Project Tools Layout Bookmarks Window Help
Layout NoDesign
ColumnLayout AllColumns
Project - F:/altera_lite/16.0/ROM Simulation/sim/ROM_Sim
Library
Project
Transcript

Name	Sta	Type	Order	Modified
ROM_Sim_testben...	?	Verilog	1	11/01/2016 10:26:42 ...
ROM_Sim.v	?	Verilog	0	11/01/2016 10:26:54 ...

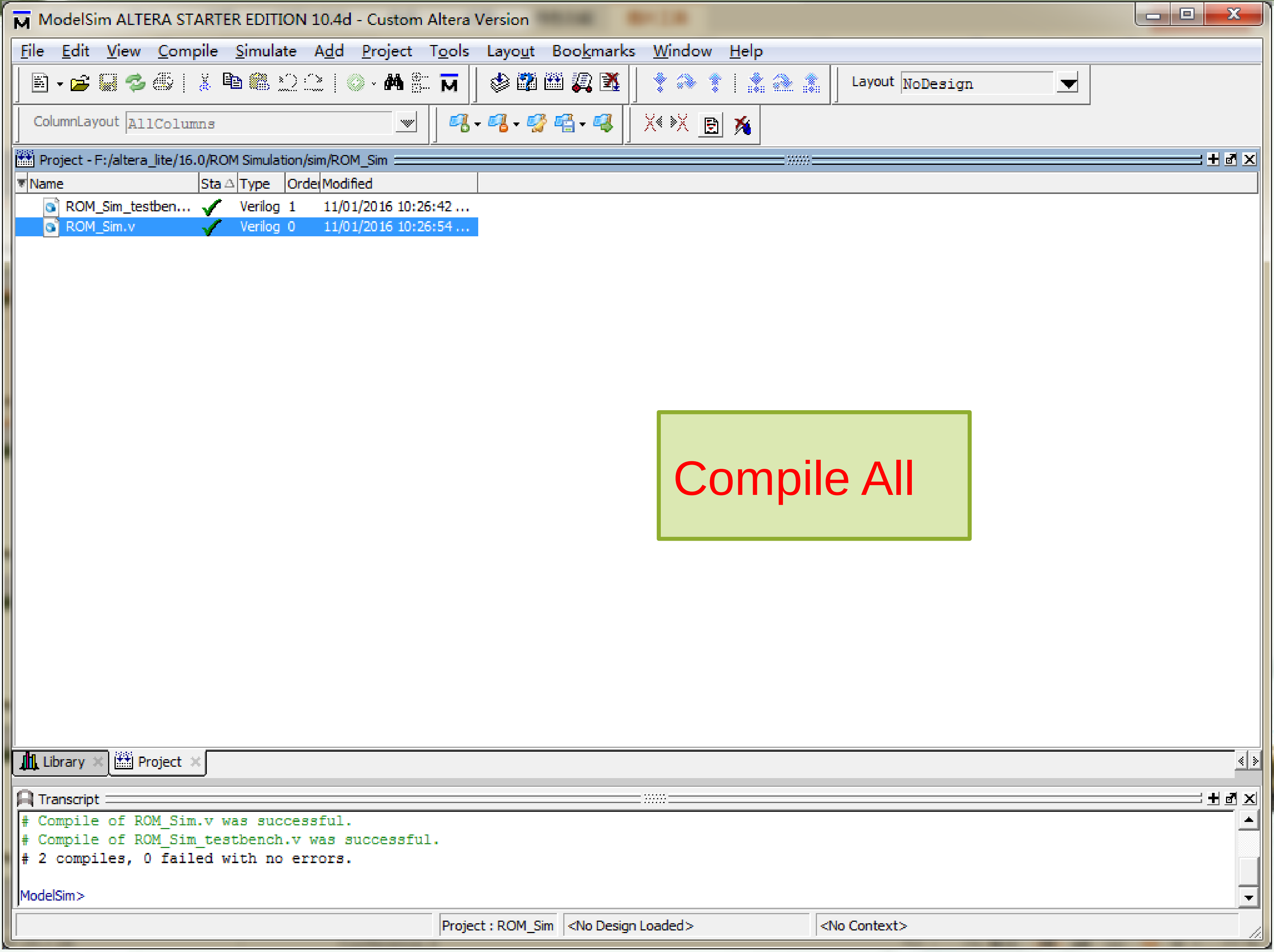
```
# reading F:/altera_lite/16.0/modelsim_ase/win32aloem/../modelsim.ini
# Loading project ROM_Sim
# Load canceled

ModelSim>
```

Project : ROM_Sim <No Design Loaded> <No Context>

Open Modelsim, new project, add 'ROM_Sim.v' and 'ROM_Sim_testbench.v' files

Configuring the libraries


ModelSim ALTERA STARTER EDITION 10.4d - Custom Altera Version
File Edit View Compile Simulate Add Project Tools Layout Bookmarks Window Help
Layout NoDesign
ColumnLayout AllColumns
Project - F:/altera_lite/16.0/ROM Simulation/sim/ROM_Sim
Library
Project
Transcript

Name	Sta	Type	Order	Modified
ROM_Sim_testben...	✓	Verilog	1	11/01/2016 10:26:42 ...
ROM_Sim.v	✓	Verilog	0	11/01/2016 10:26:54 ...

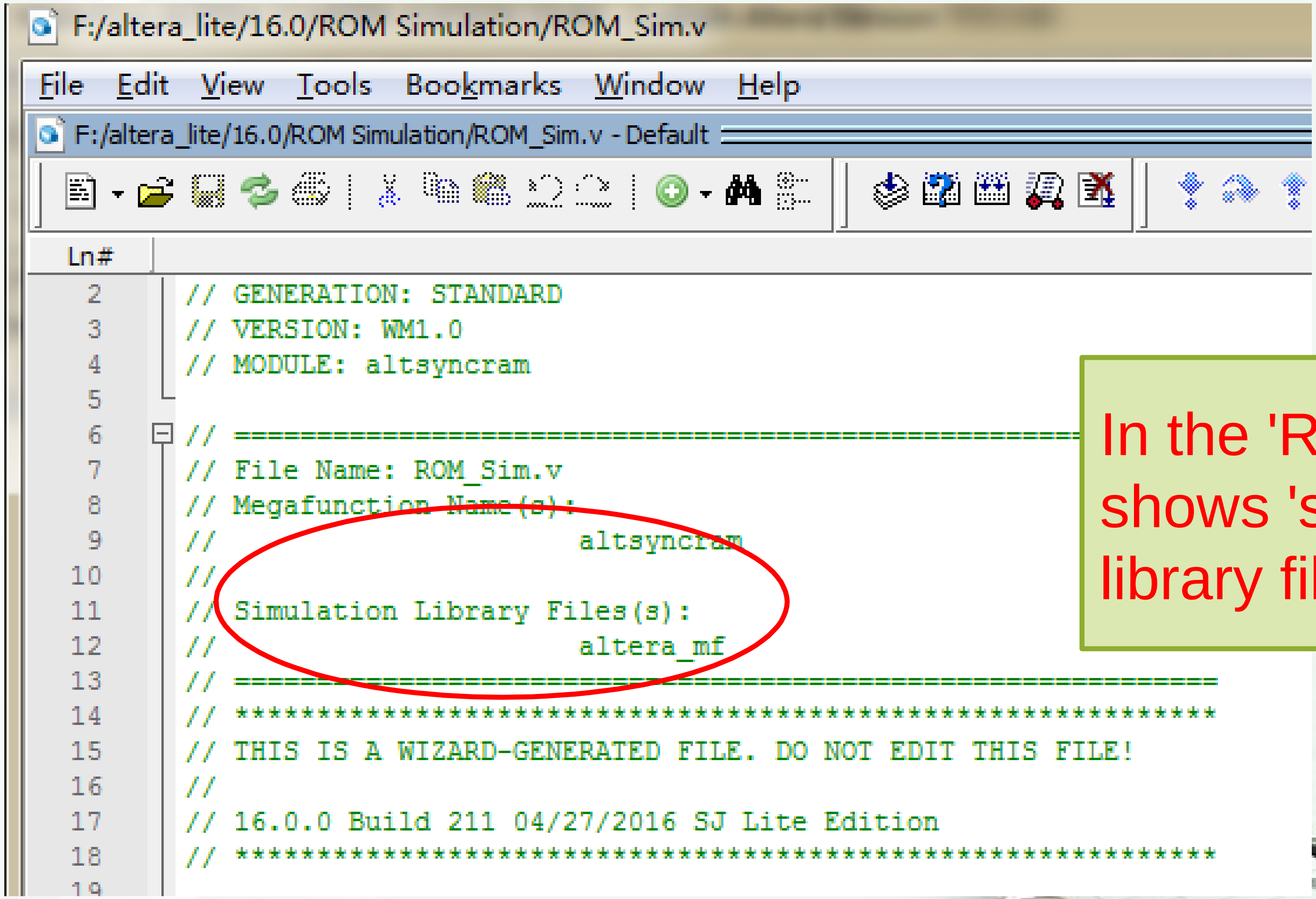
```
# Compile of ROM_Sim.v was successful.
# Compile of ROM_Sim_testbench.v was successful.
# 2 compiles, 0 failed with no errors.

ModelSim>
```

Project : ROM_Sim <No Design Loaded> <No Context>

Compile All

Configuring the libraries

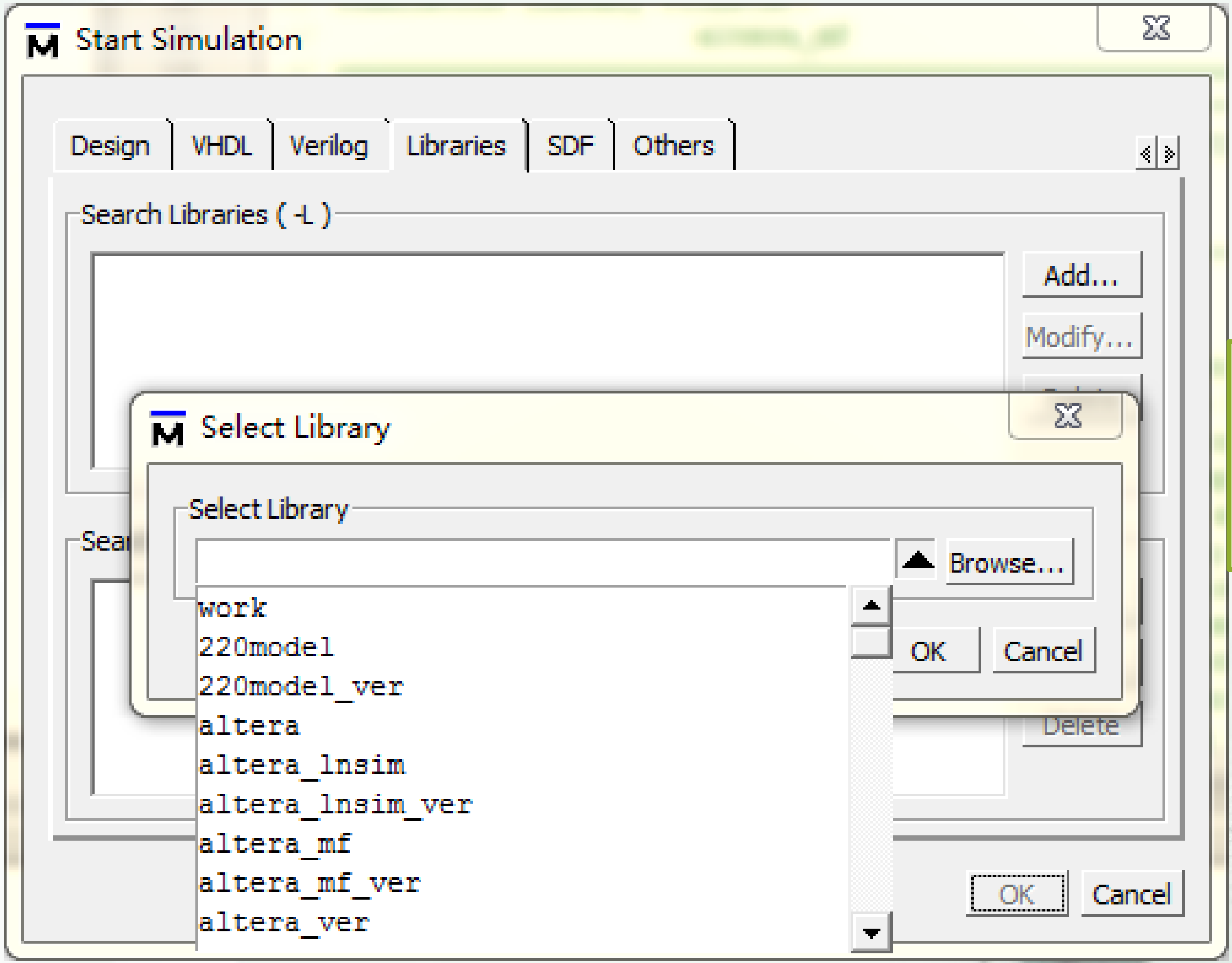

F:/altera_lite/16.0/ROM Simulation/ROM_Sim.v
File Edit View Tools Bookmarks Window Help
F:/altera_lite/16.0/ROM Simulation/ROM_Sim.v - Default
Ln#

```
// GENERATION: STANDARD
// VERSION: WM1.0
// MODULE: altsyncram

// ============================================================
// File Name: ROM_Sim.v
// Megafunction Name(s):
//                          altsyncram
//
// Simulation Library Files(s):
//                          altera_mf
// ============================================================
// ************************************************************
// THIS IS A WIZARD-GENERATED FILE. DO NOT EDIT THIS FILE!
//
// 16.0.0 Build 211 04/27/2016 SJ Lite Edition
// ************************************************************
```

In the 'ROM_Sim.v', it shows 'simulation library files: altera_mf'

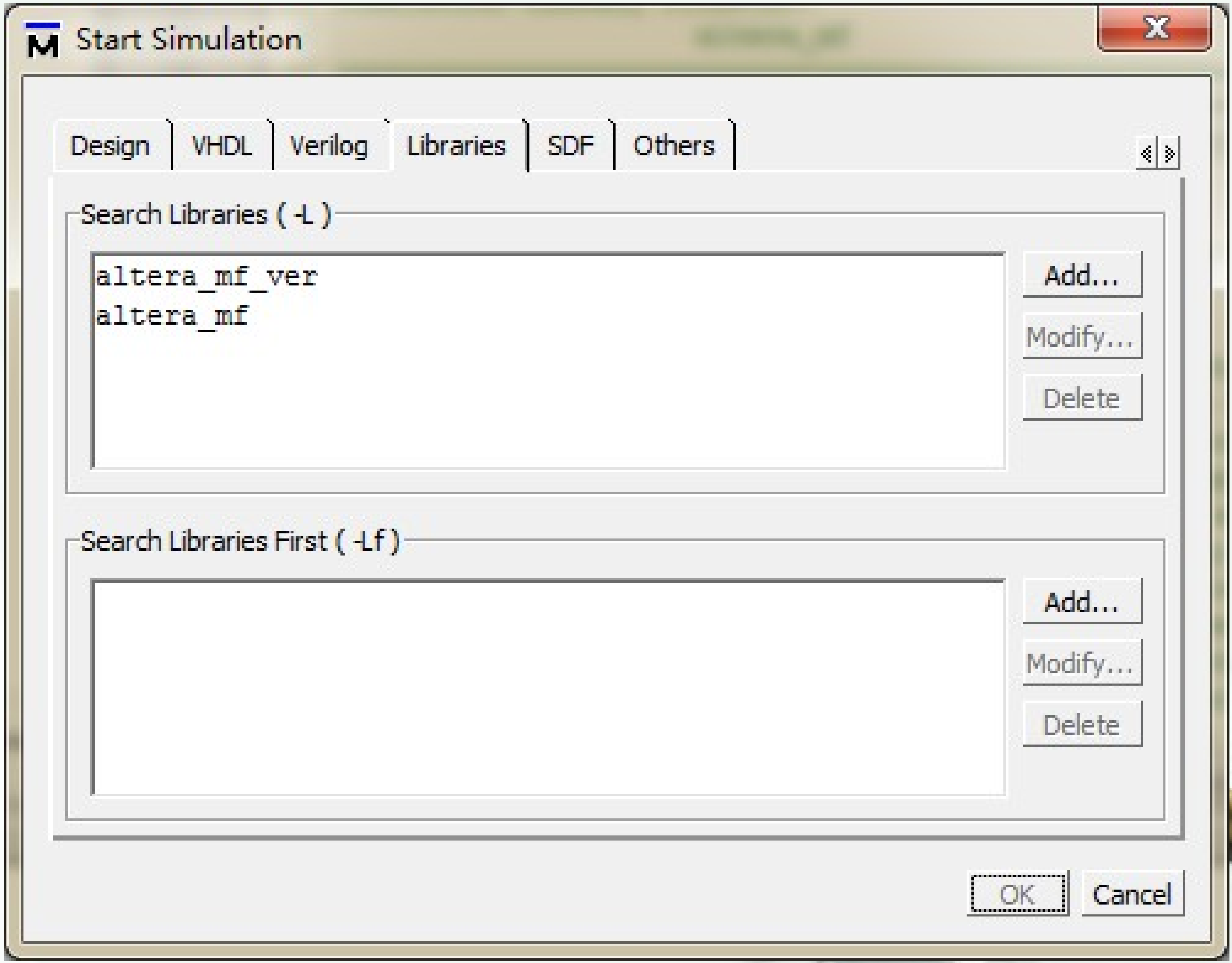
Configuring the libraries



work
220model
220model_ver
altera
altera_lnsim
altera_lnsim_ver
altera_mf
altera_mf_ver
altera_ver

add libraries 'altera_mf' and 'altera_mf_ver'

Configuring the libraries

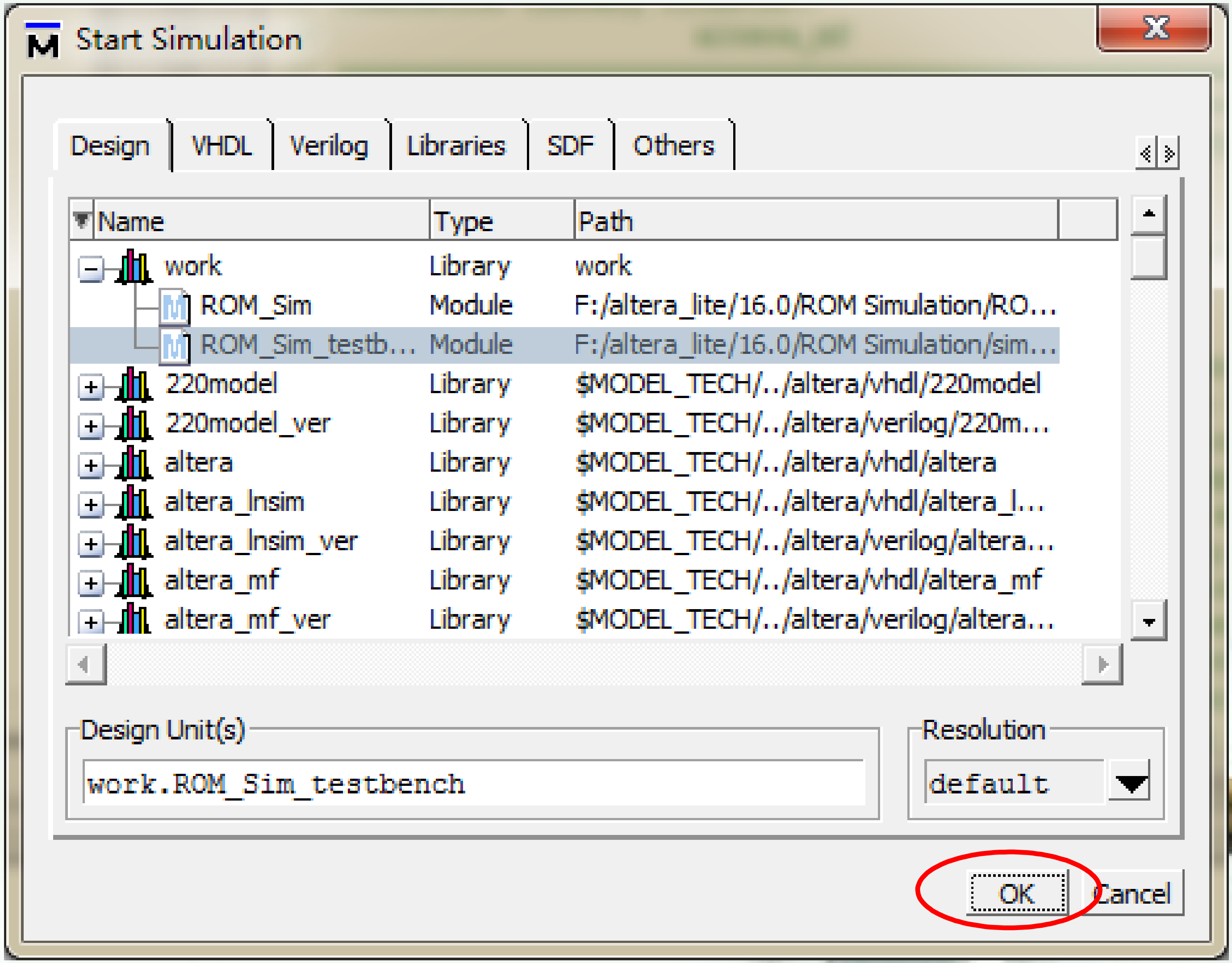


Search Libraries (-L)

```
altera_mf_ver
altera_mf
```

Search Libraries First (-Lf)

Simulating



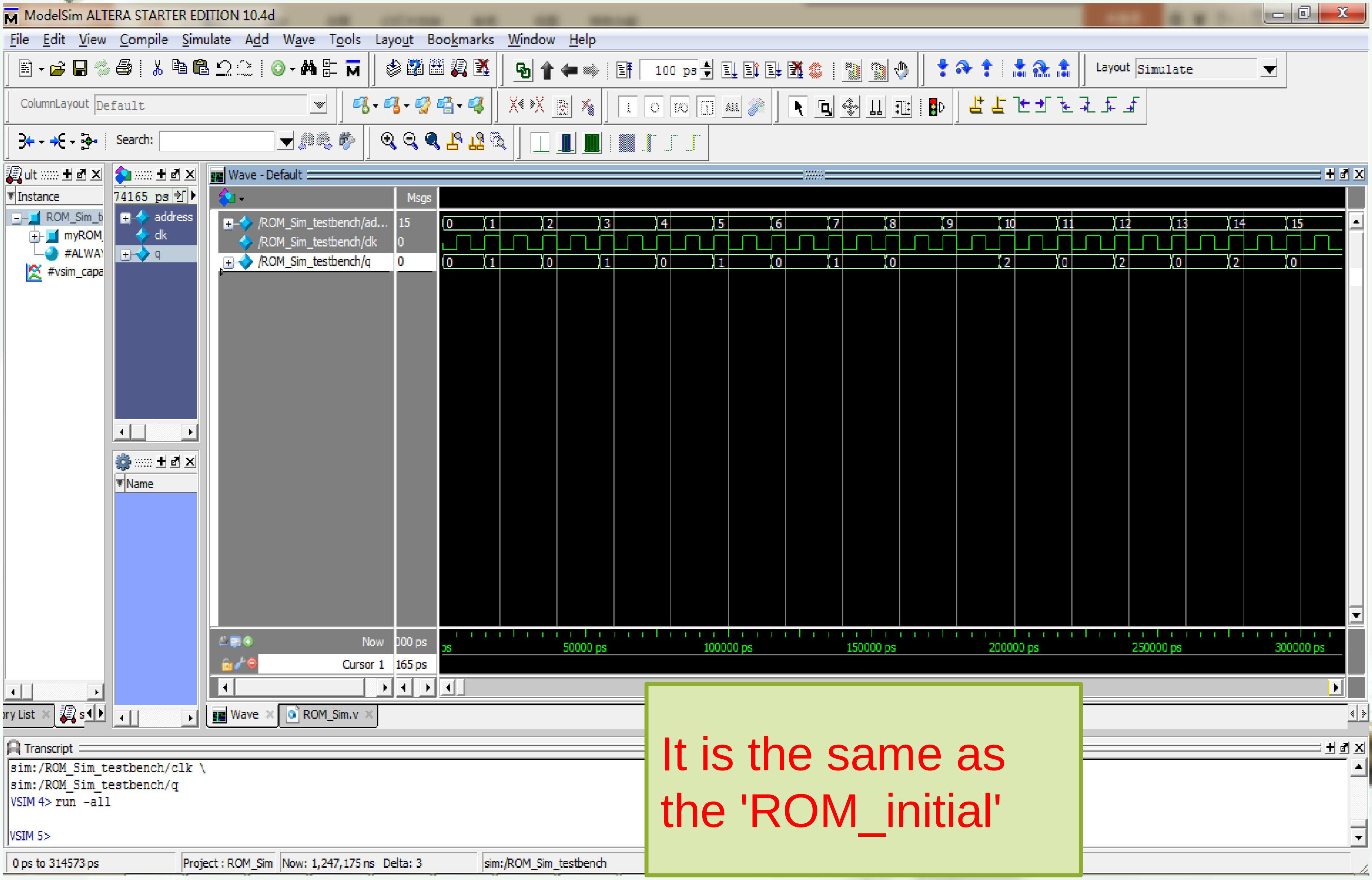
Start Simulation

Name	Type	Path
work	Library	work
ROM_Sim	Module	F:/altera_lite/16.0/ROM Simulation/RO...
ROM_Sim_testb...	Module	F:/altera_lite/16.0/ROM Simulation/sim...
220model	Library	$MODEL_TECH/../altera/vhdl/220model
220model_ver	Library	$MODEL_TECH/../altera/verilog/220m...
altera	Library	$MODEL_TECH/../altera/vhdl/altera
altera_lnsim	Library	$MODEL_TECH/../altera/vhdl/altera_l...
altera_lnsim_ver	Library	$MODEL_TECH/../altera/verilog/altera...
altera_mf	Library	$MODEL_TECH/../altera/vhdl/altera_mf
altera_mf_ver	Library	$MODEL_TECH/../altera/verilog/altera...

Design Unit(s): `work.ROM_Sim_testbench`

Resolution: `default`

Simulating



```
/ROM_Sim_testbench/ad...   15   0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
/ROM_Sim_testbench/clk     0
/ROM_Sim_testbench/q       0    0 1 0 1 0 1 0 1 0 2 0 2 0 2 0
```

```
sim:/ROM_Sim_testbench/clk \
sim:/ROM_Sim_testbench/q
VSIM 4> run -all

VSIM 5>
```

```
0 ps to 314573 ps   Project : ROM_Sim   Now: 1,247,175 ns  Delta: 3   sim:/ROM_Sim_testbench
```

It is the same as the 'ROM_initial'

Summary

YouTube

⬇

https://www.youtube.com/watch?v=gRzqcL1CwfE&index=5&list=PLFzgN2iKn8bQNg8acKxIYV-ZRgdO3iB7-