Duke ECE 496 – Spring 2013 – Project Part 2: Memory

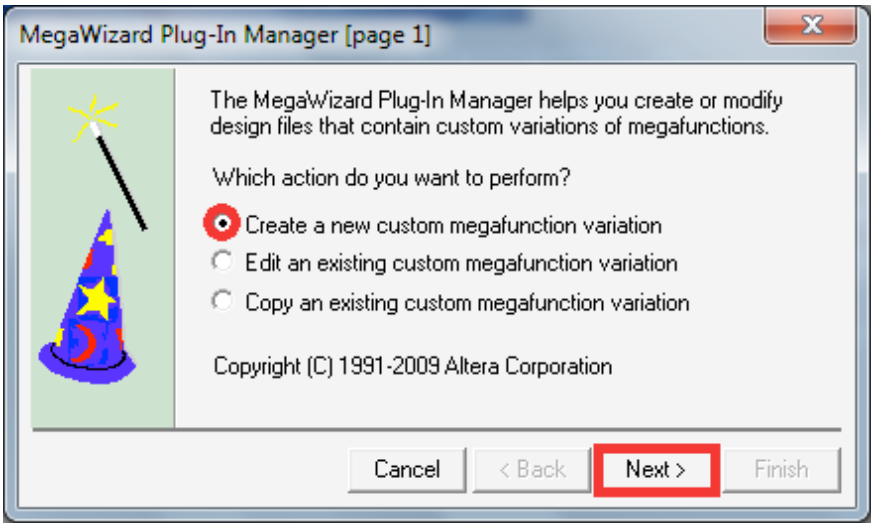
25 Points. Due electronically by 11:59pm on January 25th.

In this part of the project, you will implement the instruction and data memories that you will use in your processor. This assignment is the *only* exception to the rule about not using Quartus megafunctions – memories must be implemented through megafunctions on the FPGA. You will use Quartus' Megafunction Wizard to create the two memories. Before proceeding, make sure that you are using Quartus II Web Edition 9.1 Service Pack 2, as the Megafunction Wizard can differ substantially from release to release.

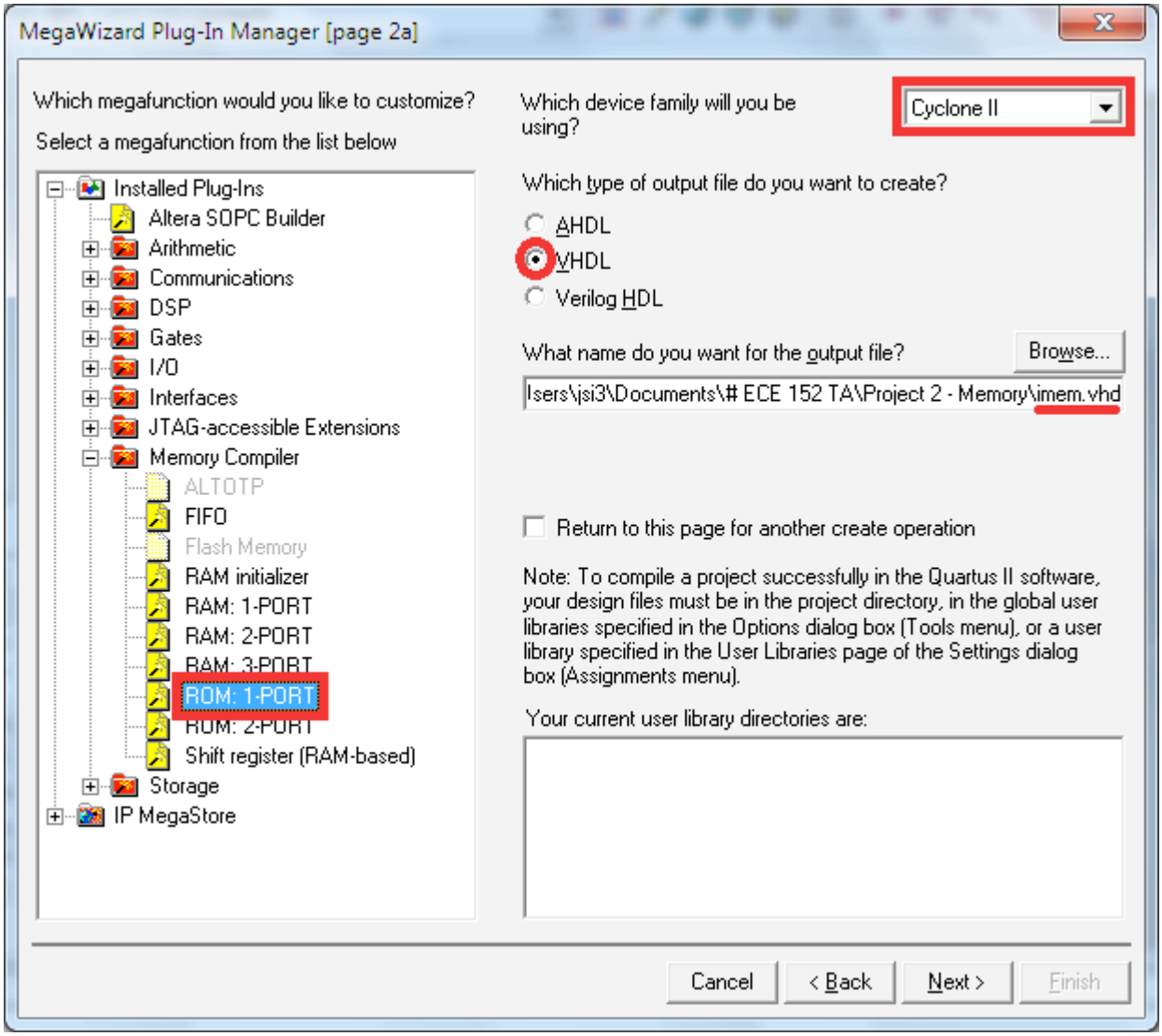
Since the architecture for your processor uses separate instruction and data memory spaces, your microarchitecture should have separate instruction and data memory blocks.

Project Part 2a: Instruction Memory

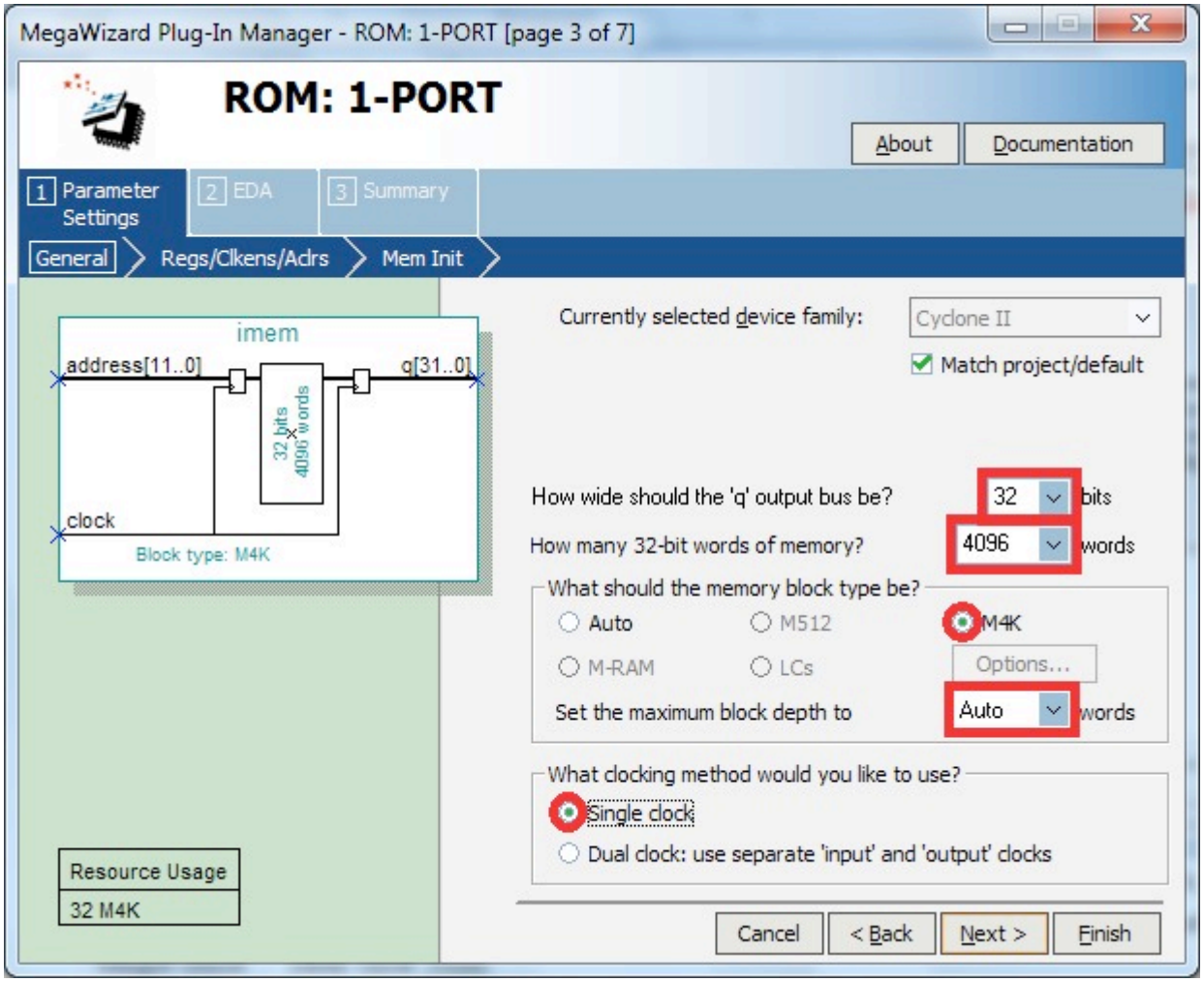
To make the instruction memory (imem.vhd), start by going to Tools ➧MegaWizard Plug-In Manager. On Page 1, select "Create" and click Next.



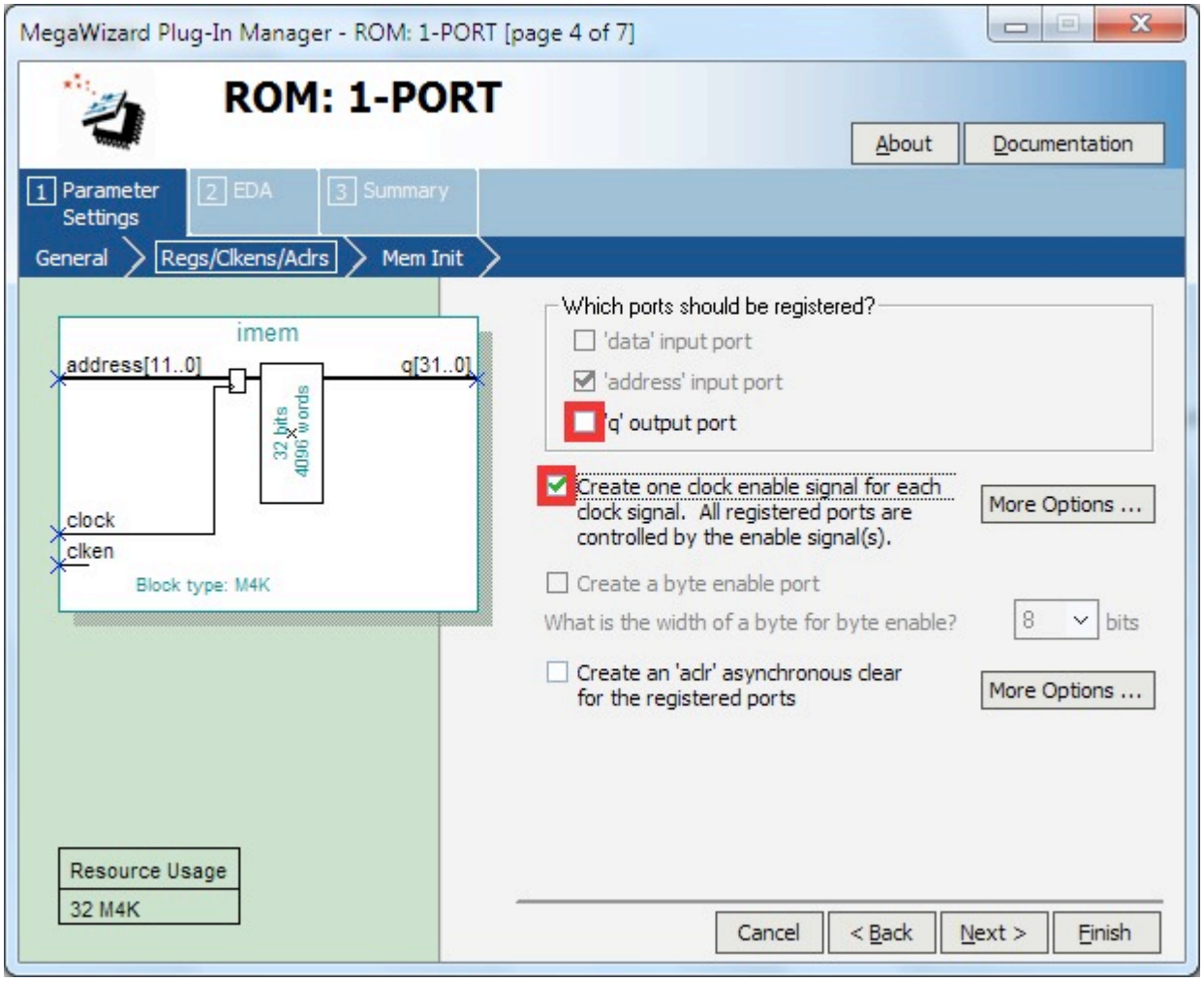
On Page 2a, select “Cyclone II” as the device family, select “VHDL” as the output file type, and set the output file name to “imem.vhd” in your working directory. On the left side, select “Installed Plug-Ins” ➧ “Memory Compiler” ➧ “ROM: 1-PORT”. Then click Next.



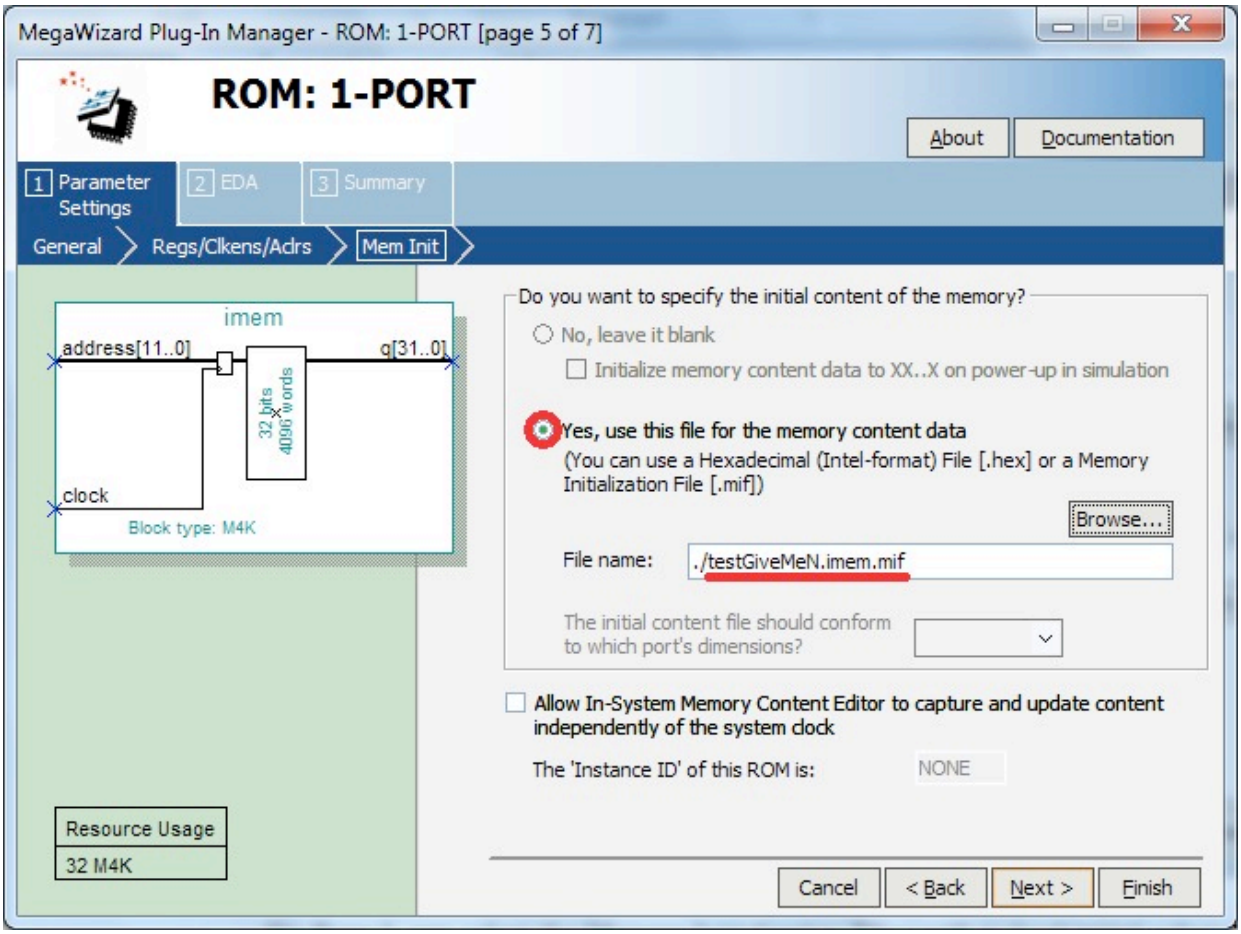
On Page 3, set the output bus width to 32 bits, set the memory size to 4096 words, set the memory block type to “M4K”, set the block depth to “Auto”, and set the clocking method to “Single clock”. Then click Next.



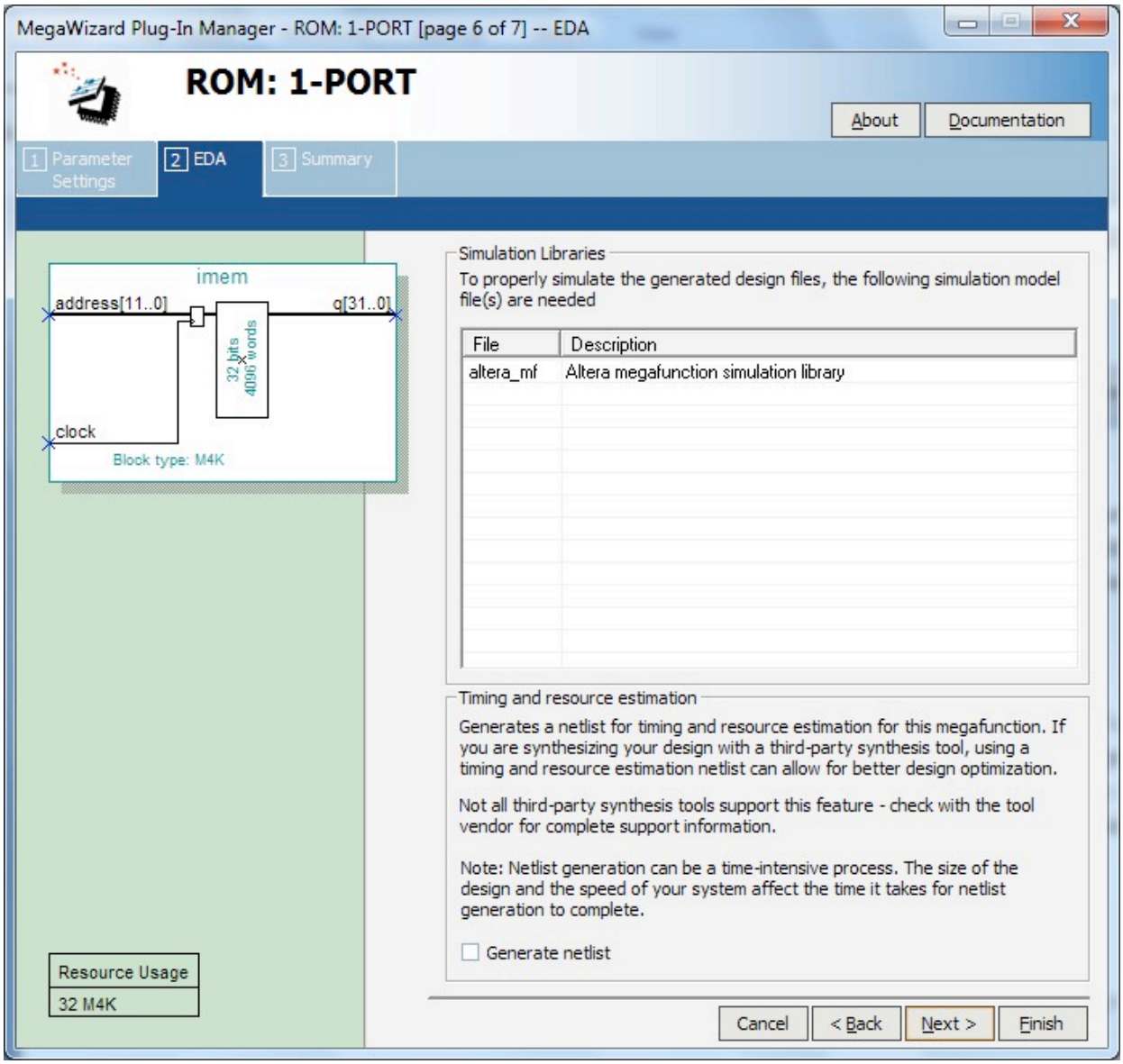
On Page 4, uncheck the register on the output port and check the clock enable. Then click Next.



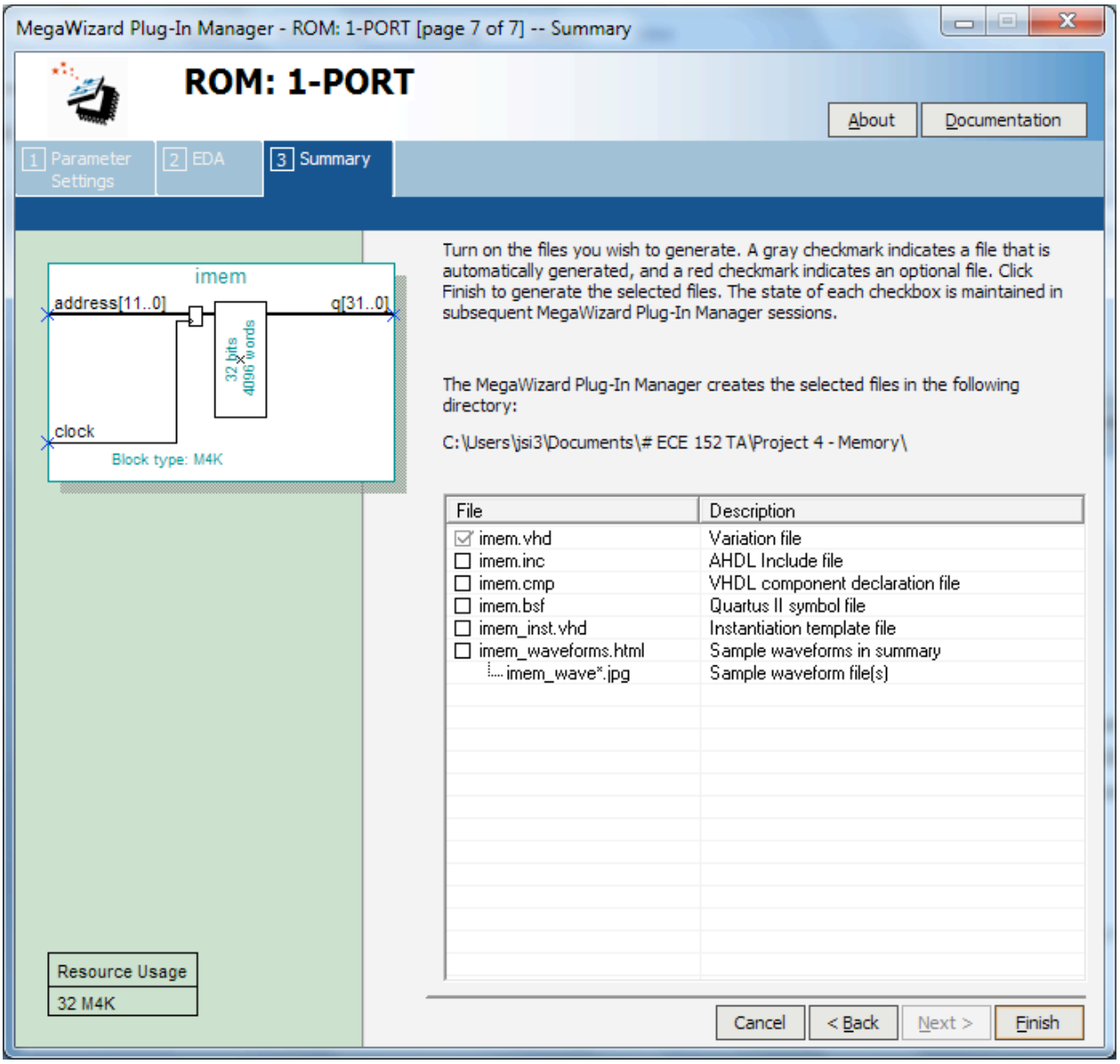
On Page 5, you select the Memory Initialization File (.mif) to load initial values into the memory with. A Memory Initialization File is available for you to download at http://people.ee.duke.edu/~sorin/ece152/project/testGiveMeN.imem.mif. Download, browse to, and select this file on your computer, then click Next.



On Page 6, click Next.



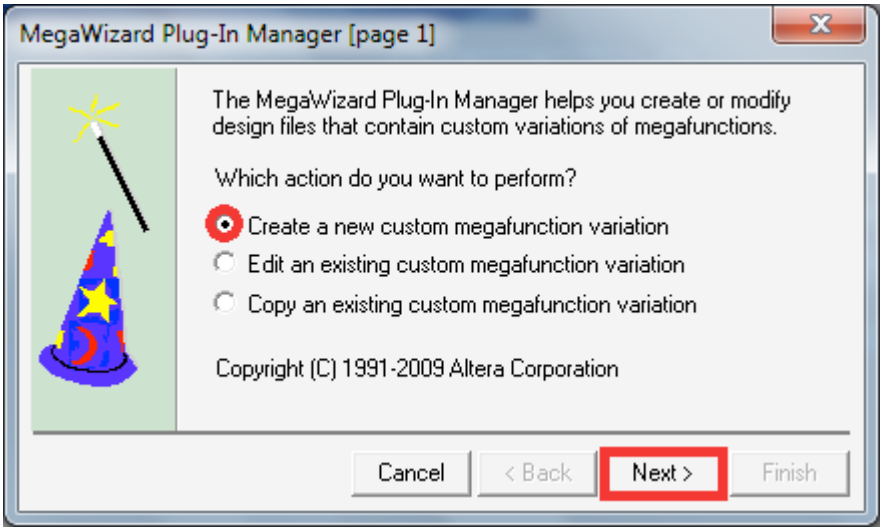
On Page 7, uncheck everything but the “imem.vhd” file, then click Finish.



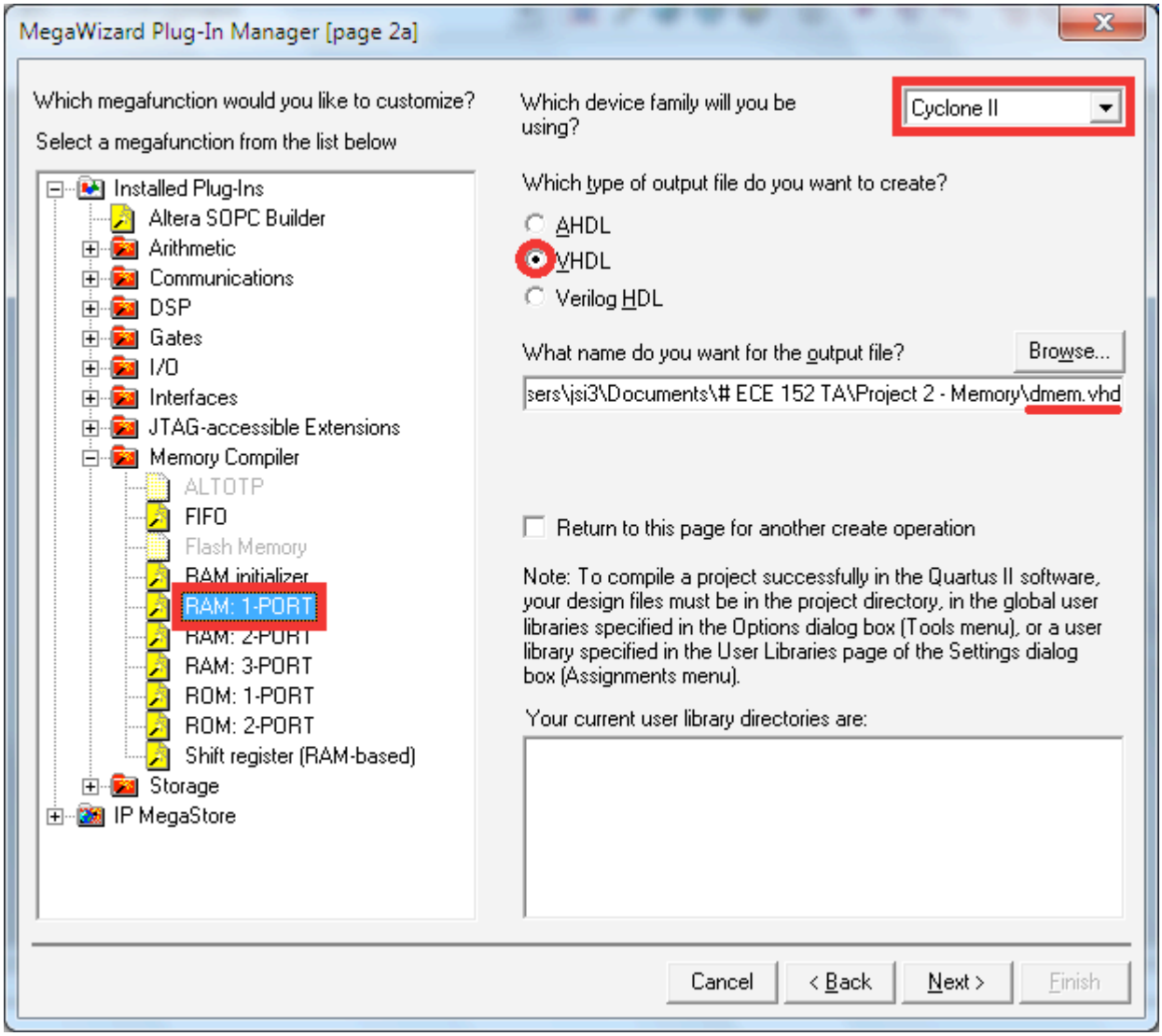
Your instruction memory is now created. Click Yes to automatically add the memory to your project, or click No to manually add just the .vhd file to your project later.

Project Part 2b: Data Memory

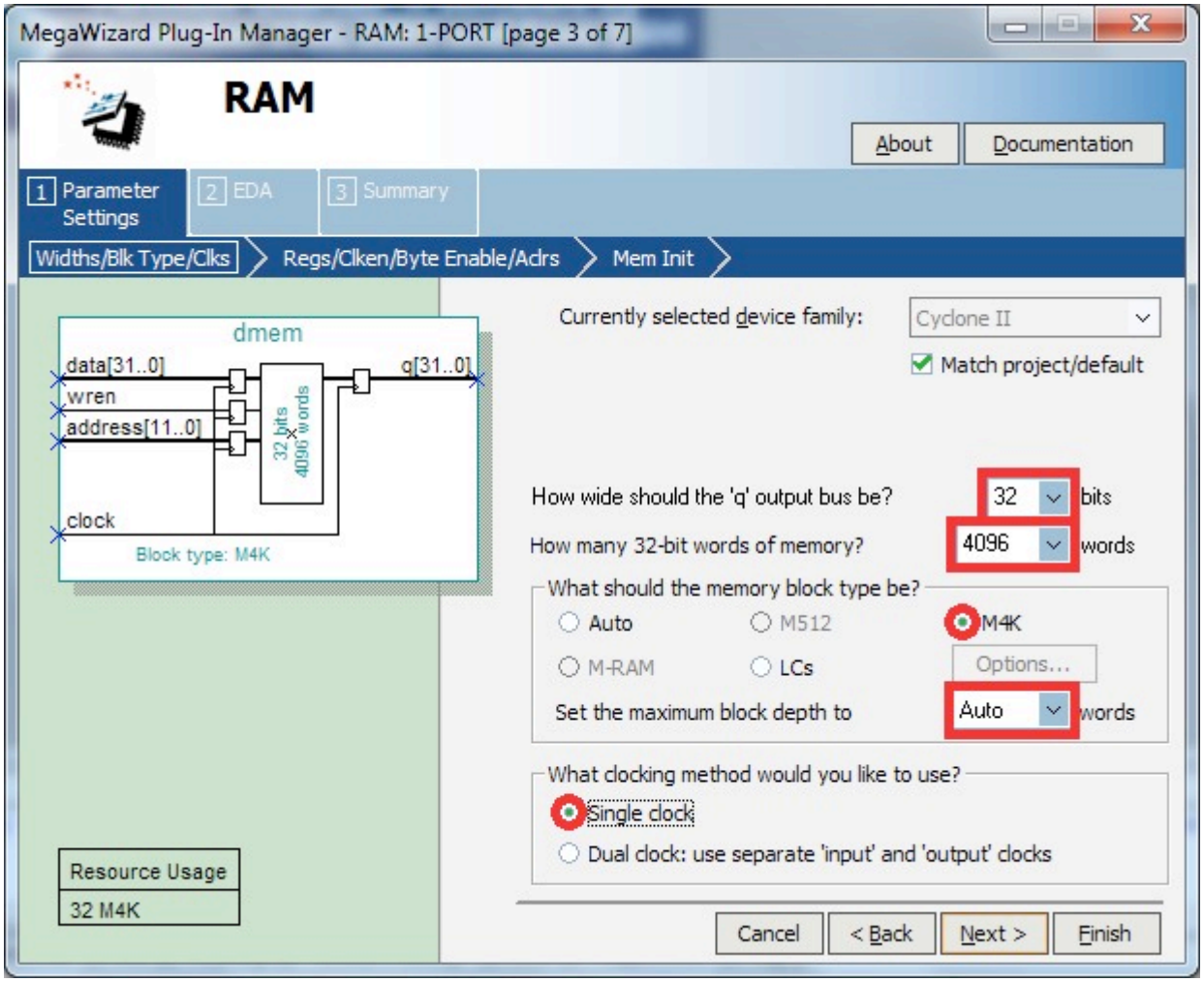
To make the data memory (dmem.vhd), start by going to Tools ➧MegaWizard Plug-In Manager. On Page 1, select "Create" and click Next.



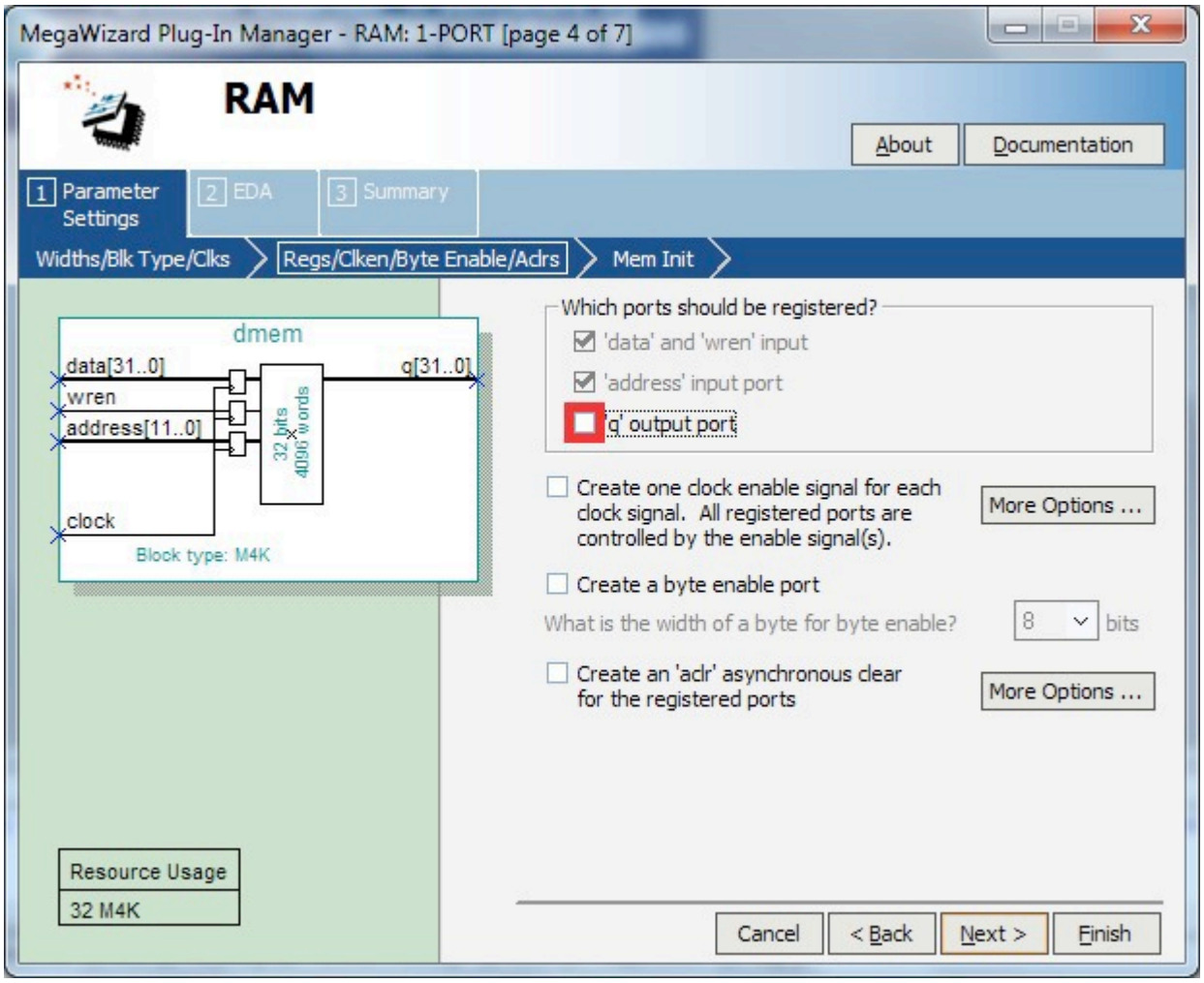
On Page 2a, select “Cyclone II” as the device family, select “VHDL” as the output file type, and set the output file name to “dmem.vhd” in your working directory. On the left side, select “Installed Plug-Ins” ➧ “Memory Compiler” ➧ “RAM: 1-PORT”. Then click Next.



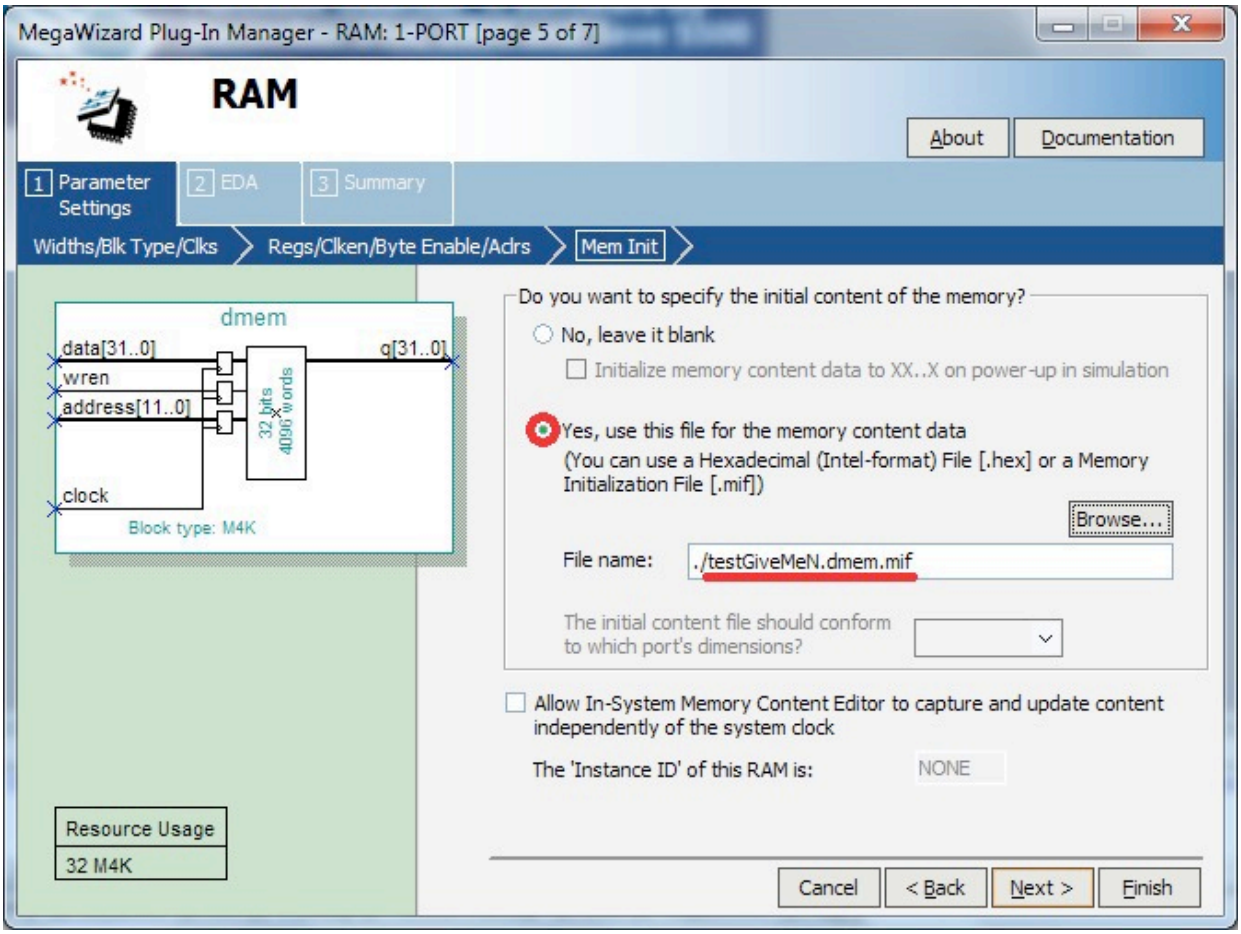
On Page 3, set the output bus width to 32 bits, set the memory size to 4096 words, set the memory block type to “M4K”, set the block depth to “Auto”, and set the clocking method to “Single clock”. Then click Next.



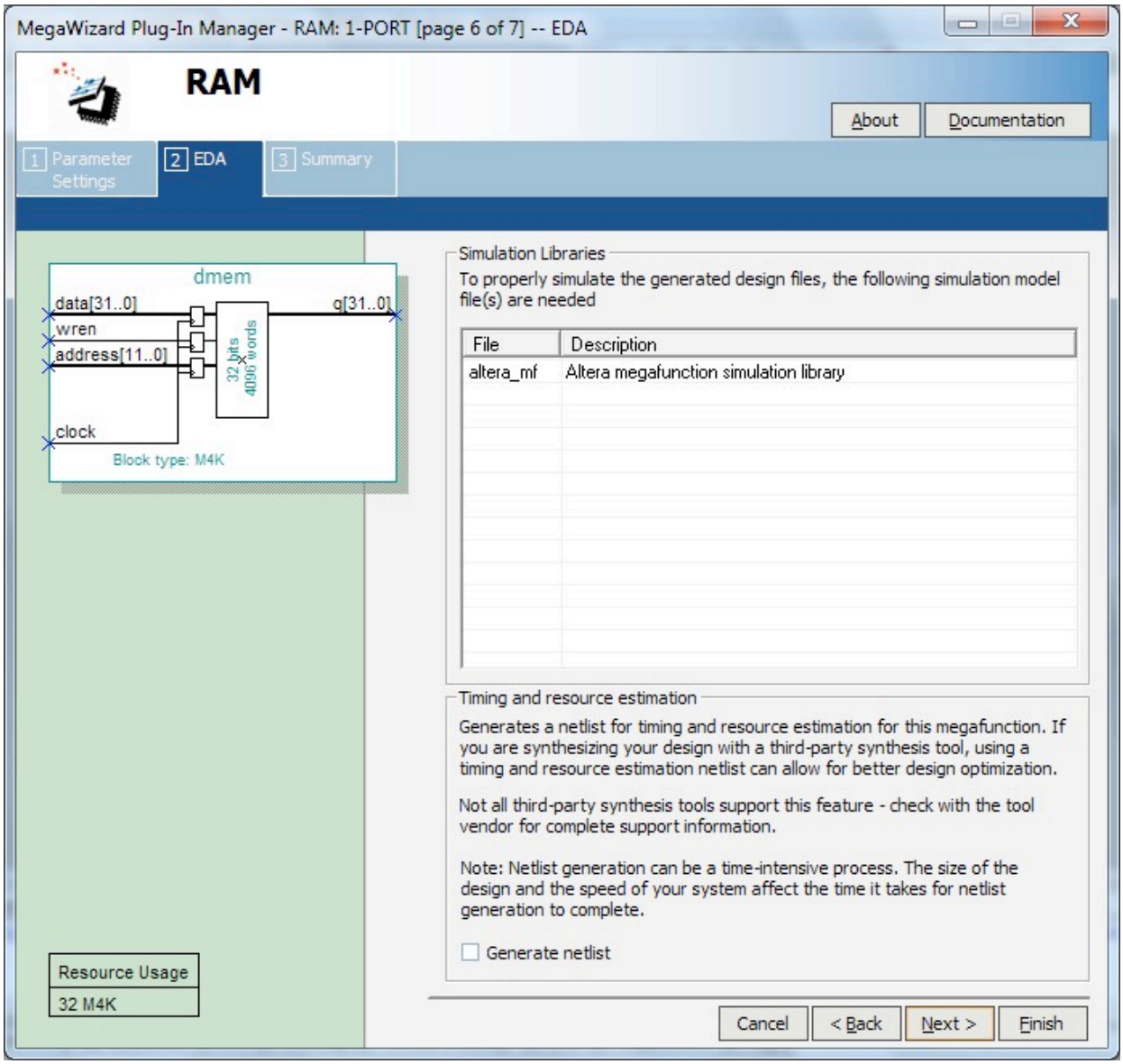
On Page 4, uncheck the register on the output port. Then click Next.



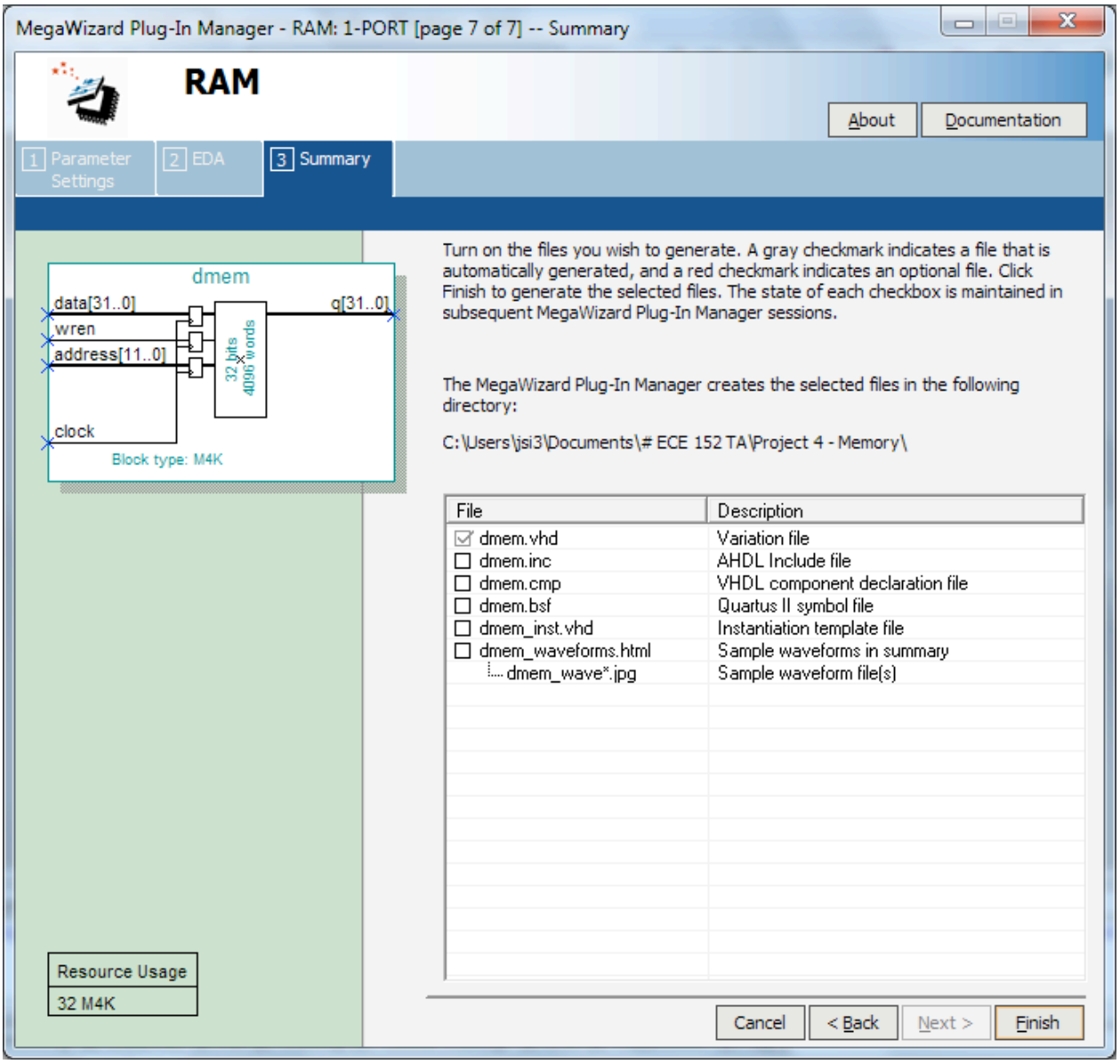
On Page 5, you select the Memory Initialization File (.mif) to load initial values into the memory with. A Memory Initialization File is available for you to download at http://people.ee.duke.edu/~sorin/ece152/project/testGiveMeN.dmem.mif. Download, browse to, and select this file on your computer, then click Next.



On Page 6, click Next.



On Page 7, uncheck everything but the “dmem.vhd” file, then click Finish.



Your data memory is now created. Click Yes to automatically add the memory to your project, or click No to manually add just the .vhd file to your project later.

Project Part 2c: Initializing the Memories

To change the Memory Initialization File for your memories, open the imem.vhd or dmem.vhd file and go to line 89 or 94, respectively, and rename the "init_file" path to the new file.

```
 81
 82   BEGIN
 83       q    <= sub_wire0(31 DOWNTO 0);
 84
 85       altsyncram_component : altsyncram
 86       GENERIC MAP (
 87           clock_enable_input_a => "BYPASS",
 88           clock_enable_output_a => "BYPASS",
 89           init_file => "testGiveMeN.imem.mif",
 90           intended_device_family => "Cyclone II",
 91           lpm_hint => "ENABLE_RUNTIME_MOD=NO",
 92           lpm_type => "altsyncram",
 93           numwords_a => 4096,
 94           operation_mode => "ROM",
 95           outdata_aclr_a => "NONE",
 96           outdata_reg_a => "UNREGISTERED",
 97           ram_block_type => "M4K",
 98           widthad_a => 12,
 99           width_a => 32,
100           width_byteena_a => 1
101       )
102       PORT MAP (
103           clock0 => clock,
104           address_a => address,
105           q_a => sub_wire0
106       );
107
```

After implementing your memories, you should test them thoroughly to verify that they work correctly. One test waveform is provided for your instruction memory at http://people.ee.duke.edu/~sorin/ece152/project/test_imem.vwf and for your data memory at http://people.ee.duke.edu/~sorin/ece152/project/test_dmem.vwf. In addition, this assignment will be graded by running additional tests that are not provided, so do not assume that you can ignore bugs that do not manifest themselves on the one test that is provided.

Submitting This Assignment

To submit this assignment, create a Quartus Archive (Project ➧Archive Project) named project2.qar of all the files needed to implement your design. Make sure that your top-level files are named imem.vhd and dmem.vhd. Submit the archive on the assignment page in Sakai, with the net IDs of the group members. There should be only one submission per group.