University of California, Davis
Department of Electrical and Computer Engineering

EEC180 DIGITAL SYSTEMS II Winter Quarter 2025

LAB 5: Memory System Design / Booth Multiplier Design

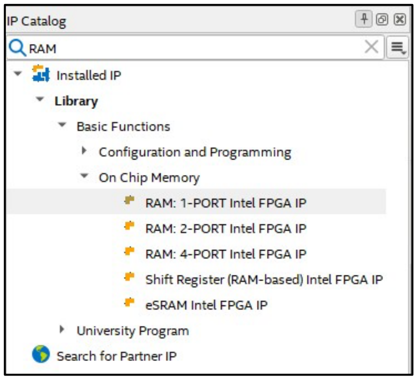
Extra Credit for Part I - Memory IPs & In-System Memory Content Editor

While testing memories with top-level I/O was possible in Part I, the I/O required to provide stimulus and check the design was significant. Typically, designs that need to be tested in hardware will leverage one of many hardware debugging tools provided by the vendor. These tools allow a user to reuse one of the exiting interfaces implemented in the FPGA (such as JTAG) to communicate to a module within the design. One basic tool provided by Quartus is the In-System Memory Content Editor (ISMCE). In this part, we will perform hardware testing using this tool.

1. The In-System Memory Content editor requires some memory IP to be instantiated within your design. IPs are flexible, parameterized, pre-built designs that allow users to skip the development of commonly used modules. By configuring the IP to communicate to the host machine (your personal computer attached to the DE10-Lite), you may communicate to a memory IP using the ISMCE tool. Both IP usage and debugging using a serial connection are common industry design/verification/testing practices.

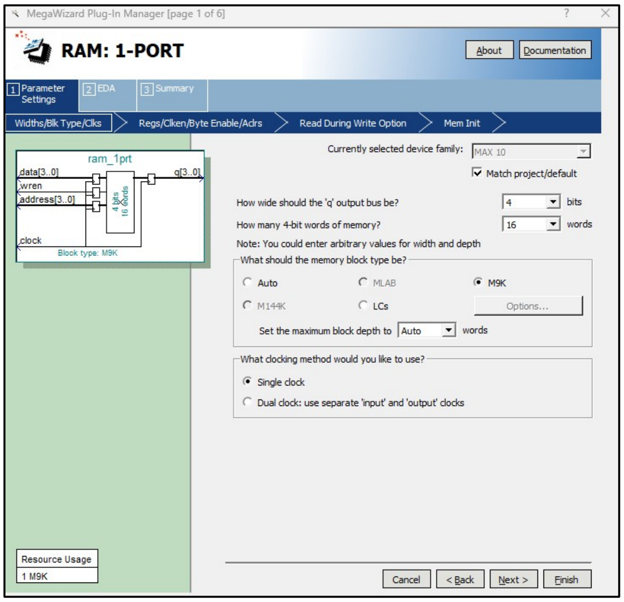
   Before you begin, make sure you have added an IP folder in your lab directory located under **lab5/ip/**.

2. First, define a memory IP by navigating Tools > IP Catalog. In the IP Catalog pane in Quartus search for “RAM 1-Port Intel FPGA IP”. Then, select the IP as shown below:


IP Catalog
RAM
Installed IP
Library
Basic Functions
Configuration and Programming
On Chip Memory
RAM: 1-PORT Intel FPGA IP
RAM: 2-PORT Intel FPGA IP
RAM: 4-PORT Intel FPGA IP
Shift Register (RAM-based) Intel FPGA IP
eSRAM Intel FPGA IP
University Program
Search for Partner IP

3. The IP parameter editor window will open and prompt for a location to save the IP variation you will define. Select the IP folder in the Lab 5 project directory and name the IP variation something descriptive such as *ram_1prt.v.*

4. In the IP editor window, you can specify the exact settings of the RAM from part 1 as shown below:



MegaWizard Plug-In Manager [page 1 of 6]

RAM: 1-PORT

Currently selected device family: MAX 10

☑ Match project/default

How wide should the 'q' output bus be? 4 bits

How many 4-bit words of memory? 16 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?

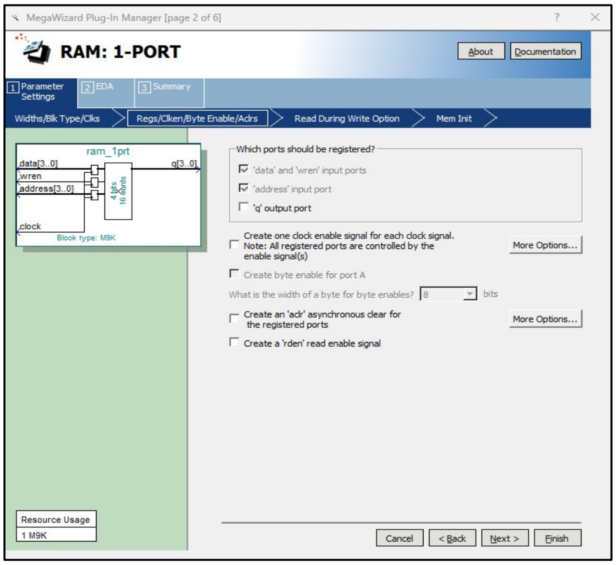
● Single clock

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

Resource Usage

1 M9K

5. After configuring the first page click Next. We will want to unregister the output Q port to maintain the same timing behavior as Part I. To do so uncheck the ‘q’ output port option:



MegaWizard Plug-In Manager [page 2 of 6]

RAM: 1-PORT

Which ports should be registered?

☑ 'data' and 'wren' input ports

☑ '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)

□ Create byte enable for port A

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

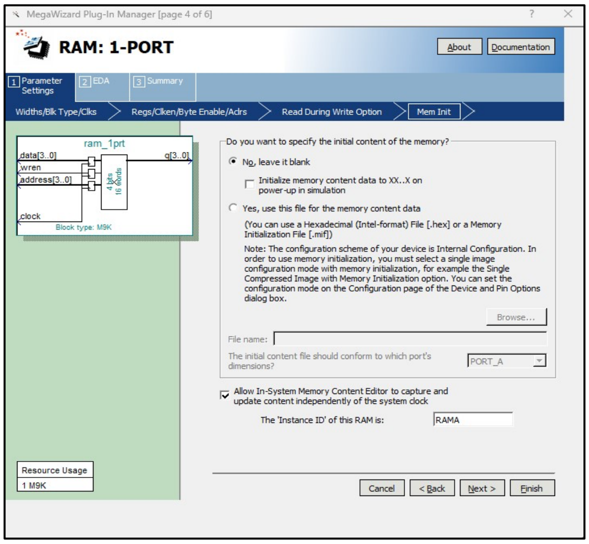
□ Create an 'aclr' asynchronous clear for the registered ports

□ Create a 'rden' read enable signal

Resource Usage

1 M9K

6. Click Next twice, and in the Mem Init window. Now we will want to enable the In-System Memory Content Editor to observe the values within the memory IP. This capability is enabled via JTAG, so you must inform Quartus that you would like to add this feature. To do so, check the "Allow In-System…" box as shown below and provide a name for the instance ID:



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])

Note: The configuration scheme of your device is Internal Configuration. In order to use memory initialization, you must select a single image configuration mode with memory initialization, for example the Single Compressed Image with Memory Initialization option. You can set the configuration mode on the Configuration page of the Device and Pin Options dialog box.

File name:

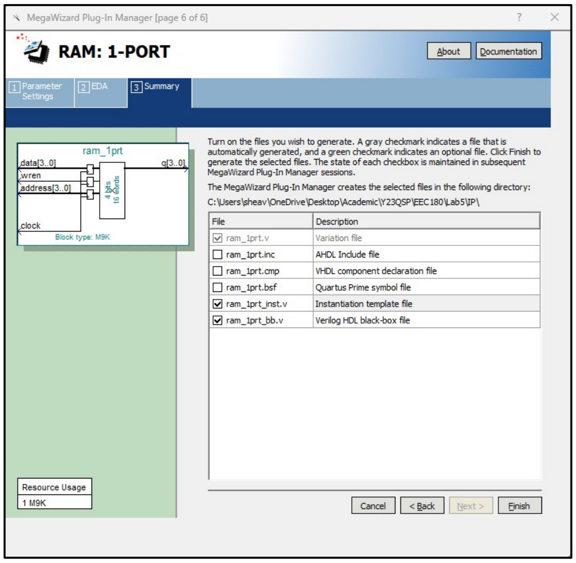
The initial content file should conform to which port's dimensions? PORT_A

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

The 'Instance ID' of this RAM is: RAMA

Resource Usage
1 M9K

7. Skip the EDA tools page and go to Summary. To make instantiating the IP easier, in the Summary page, check the "Instantiation template file" box as shown below, then click Finish.



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:

C:\Users\sheav\OneDrive\Desktop\Academic\Y23QSP\EEC180\Lab5\IP\

File	Description
☑ ram_1prt.v	Variation file
□ ram_1prt.inc	AHDL Include file
□ ram_1prt.cmp	VHDL component declaration file
□ ram_1prt.bsf	Quartus Prime symbol file
☑ ram_1prt_inst.v	Instantiation template file
☑ ram_1prt_bb.v	Verilog HDL black-box file

Resource Usage
1 M9K

8. The main Quartus window will prompt you to add the .qip file that describes the new IP variant to the project. Select "Yes". If you do not see this option add the .qip file located in the IP directory to your project.

9. To instantiate the IP in your project comment one of your existing memory instances, then instantiate the new IP (to see an example of how to instantiate, use the [IP name]_inst.v template by the IP Parameter Editor located in the IP directory).

10. After connecting the IP to your design compile the project and program the DE10-Lite.

11. After programming, keep the DE10-Lite connected to the PC and open the ISMCE tool by navigating Tools > In-System Memory Content Editor.

12. The tool should automatically detect the instantiated IP. If it does not, you can click Scan Chain so that the tool scans the JTAG chain over USB Blaster for available devices. An instance should be observed. To see the initial contents of the RAM click the instance then navigate to Processing and click "Read Data from In-System Memory".

13. To write data you can modify the contents displayed in the window and navigate to Processing "Write Data to In-System Memory".

14. Demonstrate the following to a TA:

    a. Read the initial contents from the memory after programming.
    b. Write data to an address in the memory using the ISMCE tool.
    c. Show that the modified data has been written to the memory using switches and display it on the LEDs.
    d. Write data using the switches.
    e. Read the updated contents from ISMCE.