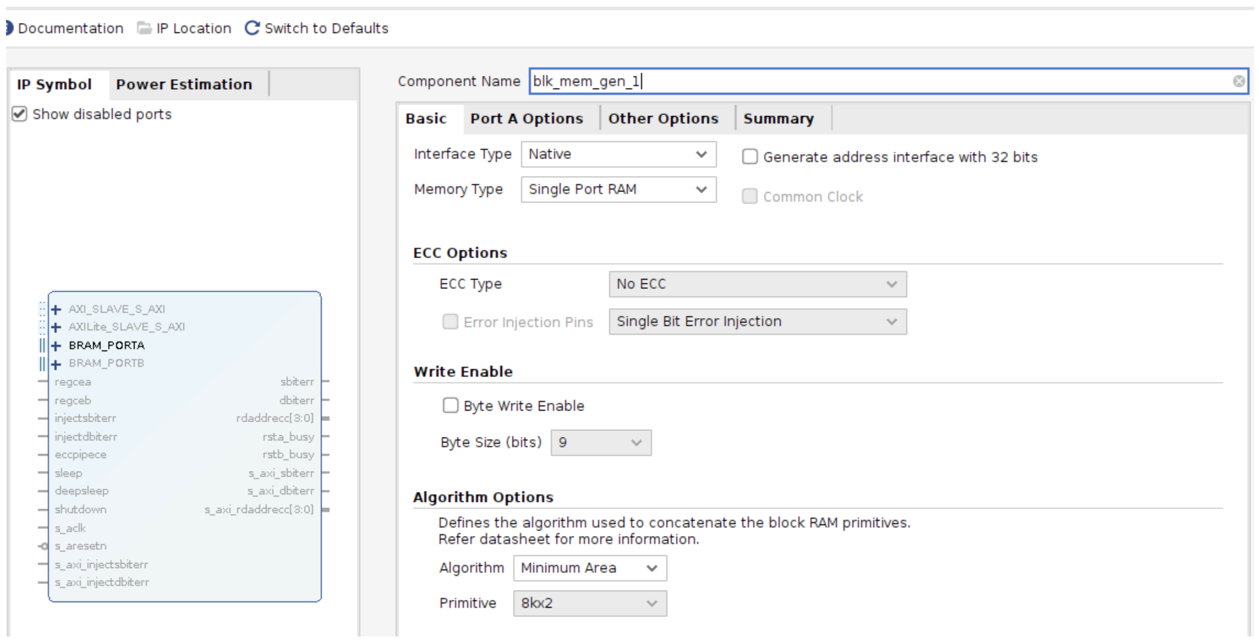
BRAM initialization

—---

In the assignment you will be given .coe format files that will have your input test case. In order to use the test cases in your design , you need to store them first in the FPGA memory blocks (BRAM). Following are the steps that you need to follow for initialization of the memory and check if the initialisation happened correctly.

1. Start Vivado and open your existing project or create a new project.
2. In the **Flow Navigator**, click **IP Catalog**
3. In the **IP Catalog**, search for **Block Memory Generator**
4. Double-click on the **Block Memory Generator** IP to configure it



Component Name: blk_mem_gen_1

Interface Type: Native

Memory Type: Single Port RAM

☐ Generate address interface with 32 bits

☐ Common Clock

ECC Options

ECC Type: No ECC

☐ Error Injection Pins: Single Bit Error Injection

Write Enable

☐ Byte Write Enable

Byte Size (bits): 9

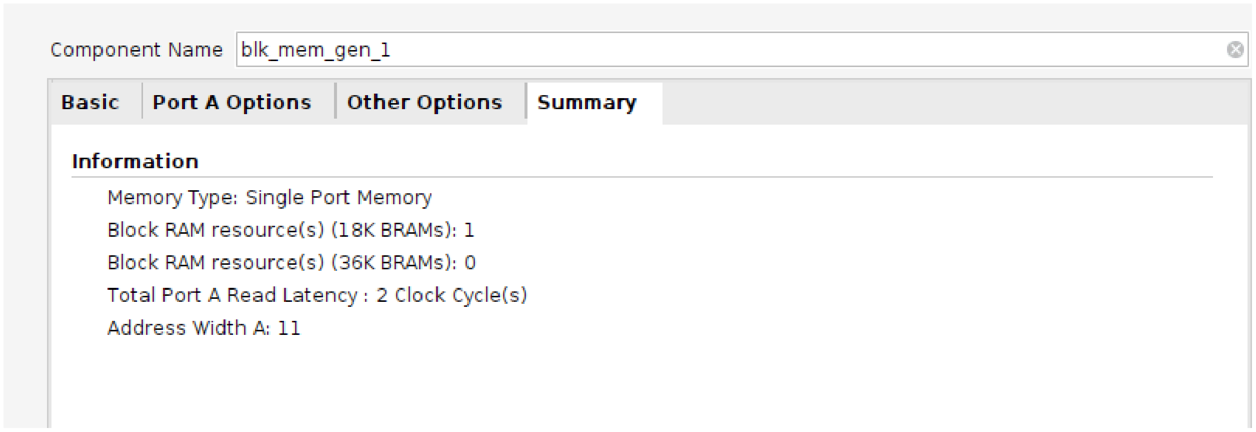
Algorithm Options

Defines the algorithm used to concatenate the block RAM primitives.
Refer datasheet for more information.

Algorithm: Minimum Area

Primitive: 8kx2

5. Choose memory type as **Single Port RAM**
6. In port A option, chose write/read width as 8 bit and read/write depth as 2000
7. In the Other Options tab, select load init file check box, and upload the .coe file test input given by us.
8. On the same tab tick the checkbox , “Fill the remaining memory locations 0”
9. You should see the summary as below , which says we have used 1 18 kbits BRAM(our FPGA board BRAM capacity)



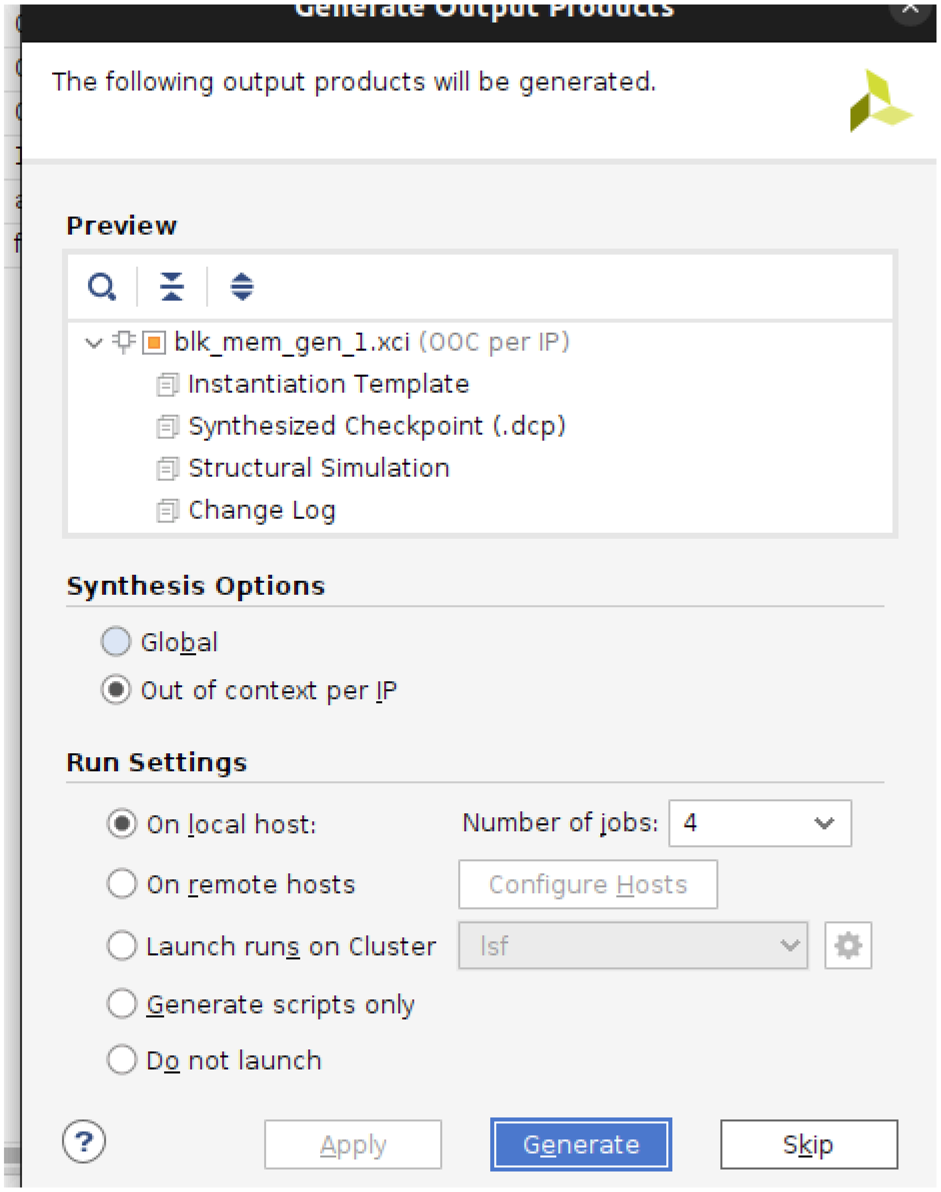
Component Name blk_mem_gen_1

Basic | Port A Options | Other Options | Summary

Information

Memory Type: Single Port Memory
Block RAM resource(s) (18K BRAMs): 1
Block RAM resource(s) (36K BRAMs): 0
Total Port A Read Latency : 2 Clock Cycle(s)
Address Width A: 11

10. Once you are done, click okay and you will get a pop-up window as below



Generate Output Products

The following output products will be generated.

Preview

blk_mem_gen_1.xci (OOC per IP)
- Instantiation Template
- Synthesized Checkpoint (.dcp)
- Structural Simulation
- Change Log

Synthesis Options

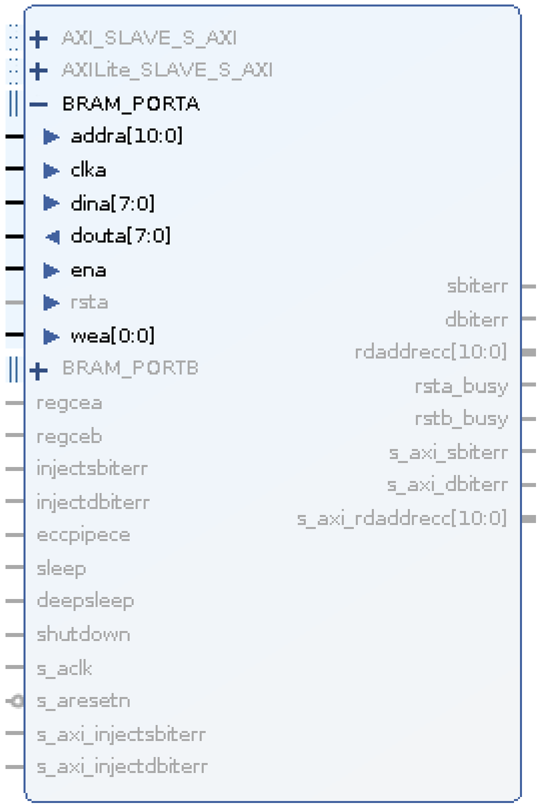
( ) Global
(●) Out of context per IP

Run Settings

(●) On local host: Number of jobs: 4
( ) On remote hosts [Configure Hosts]
( ) Launch runs on Cluster lsf
( ) Generate scripts only
( ) Do not launch

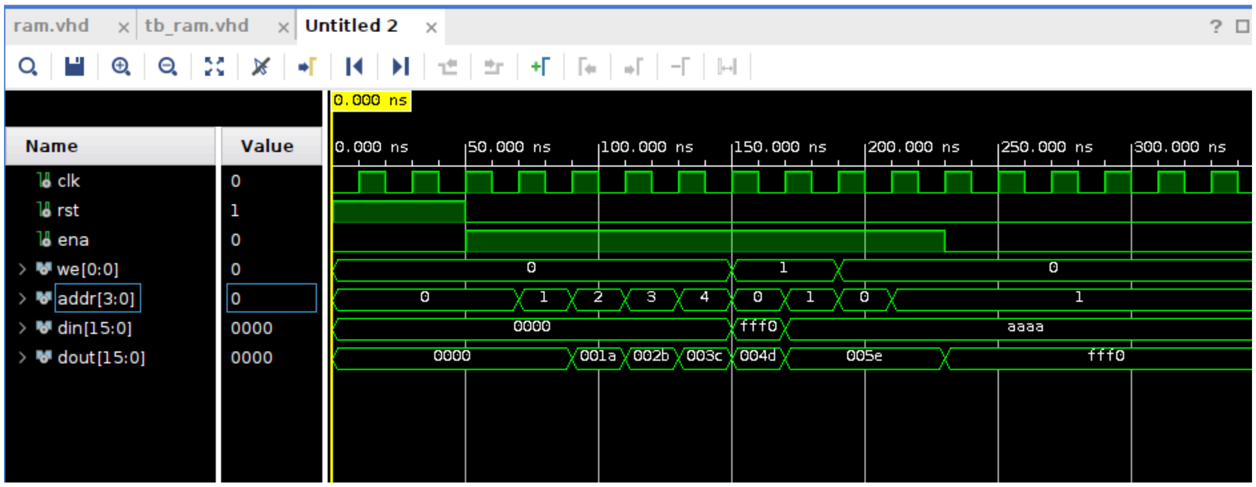
11. Use the global synthesis option and leave other settings as default and then click generate , this will initialize the input data given by us , to memory and you will have preloaded memory (with test input) that you can use during your simulation + synthesis and implementation. In this design file you would see a block named “blk_mem_gen_0”. You will use this in your design files as component (like any usual instantiation).

12. Depending on the data width and depth set , it will generate the memory with corresponding width that you can see by double clicking your generate memory design file (.xci)

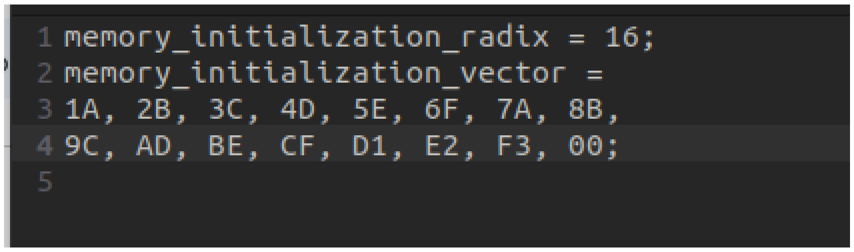


13. I have for example added a 16 element data sample.coe file and added corresponding design file and test bench to show you how we can use the memory data in your program.
You can refer to the code and make changes as per your needs for bit width of addr, din and dout.

With initialization of sample.coe file you would and code given you would see following waveform where I am reading first 5 memory addresses , without modification and then modifying the first 2 memory addresses and again reading it. The wave form will come as per the data in our sample.coe file and test bench operation. **The lag of 2 cycles in is read latency of memory.**



Sample.coe



```
memory_initialization_radix = 16;
memory_initialization_vector =
1A, 2B, 3C, 4D, 5E, 6F, 7A, 8B,
9C, AD, BE, CF, D1, E2, F3, 00;
```

14. After cross checking your data in memory from the test.coe file accordingly you can go ahead with your further design.