FPGA on-chip ROM Read and Write Experiment

Technical Email: alinx@aithtech.com Sales Email: rachel.zhou@alinx.com

1 Experiment Introduction

This lab will introduce you how to use the internal ROM of the FPGA and the data read operation of the program.

2 Experiment Principle

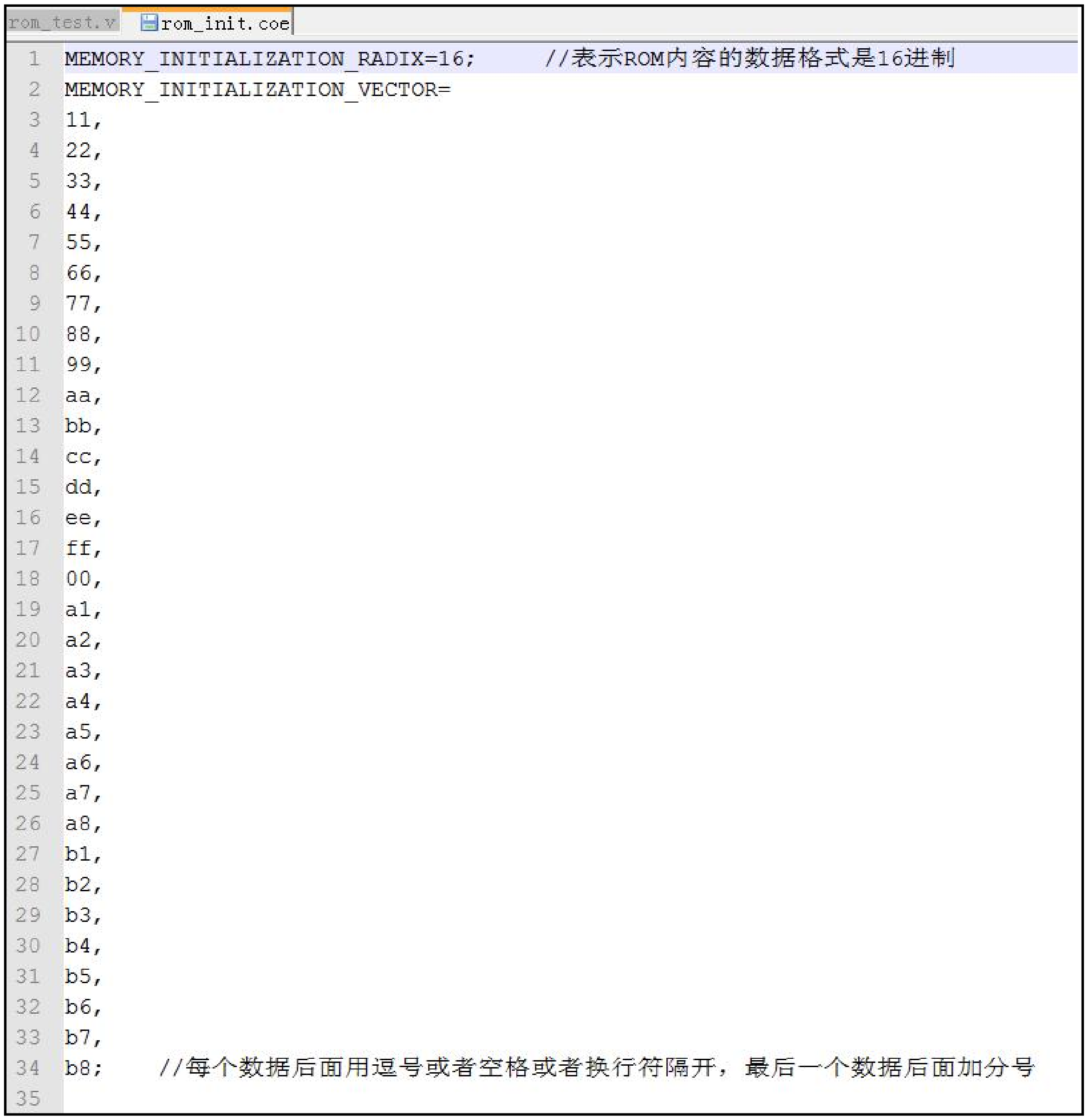
Xilinx has provided the IP core of “ROM” for us in VIVADO. We only need to instantiate a ROM through IP core and read the data stored in ROM according to the read timing of ROM. In the experiment, “VIVADO” integrated online logic analyzer “ila”, we can observe the ROM read timing and data read from the ROM.

3 Programming

3.1 Create a ROM initialization file

Since it is a ROM, then we must prepare the data in advance, and then when the FPGA is actually running, we can directly read the pre-stored data in these ROMs. The on-chip ROM of Xilinx FPGA supports initial data configuration. As shown below, we can create a file called “rom_init.coe”, note that the suffix must be ".coe", the name of the front you can of course.

The format of the ROM initialization file is very simple, as shown in the following figure. The first behavior defines the data format, and 16 represents the data format of the ROM in hexadecimal. From the 3rd line to the 34th line, it is the initialization data of this 32*8bit size ROM. Each line of numbers is followed by a comma, and the last line of numbers ends with a semicolon.



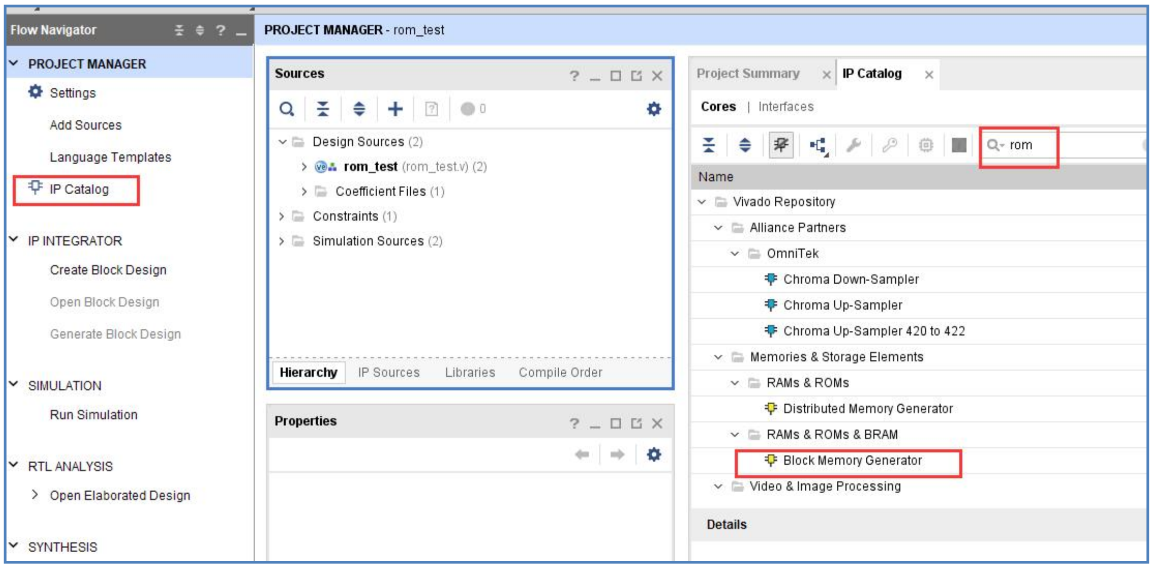
```
MEMORY_INITIALIZATION_RADIX=16;      //表示ROM内容的数据格式是16进制
MEMORY_INITIALIZATION_VECTOR=
11,
22,
33,
44,
55,
66,
77,
88,
99,
aa,
bb,
cc,
dd,
ee,
ff,
00,
a1,
a2,
a3,
a4,
a5,
a6,
a7,
a8,
b1,
b2,
b3,
b4,
b5,
b6,
b7,
b8;     //每个数据后面用逗号或者空格或者换行符隔开，最后一个数据后面加分号
```

After “rom_init.coe” is written, save it. Next we will design and configure the ROM IP core.

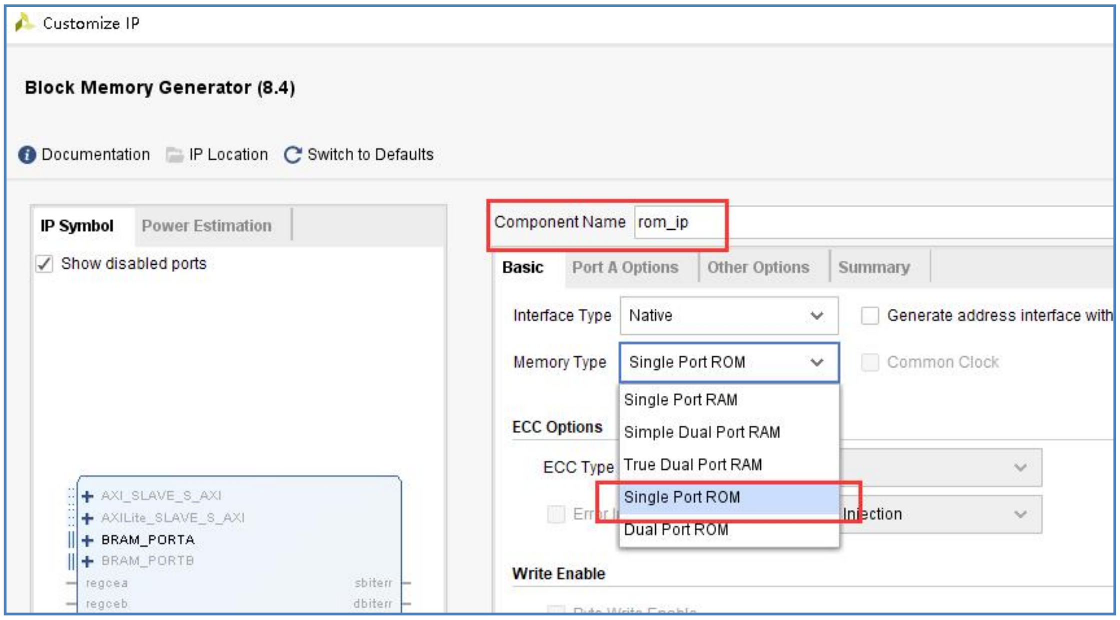
3.2 ”ROM IP” addition and configuration

Create a new “rom_test” project before adding the “ROM IP”, and then add the “ROM IP” in the project as follows:

1. Click on the “IP Catalog” in the figure below, search for “rom” in the pop-up interface on the right, find the “Block Memory Generator”, and double-click to open it.

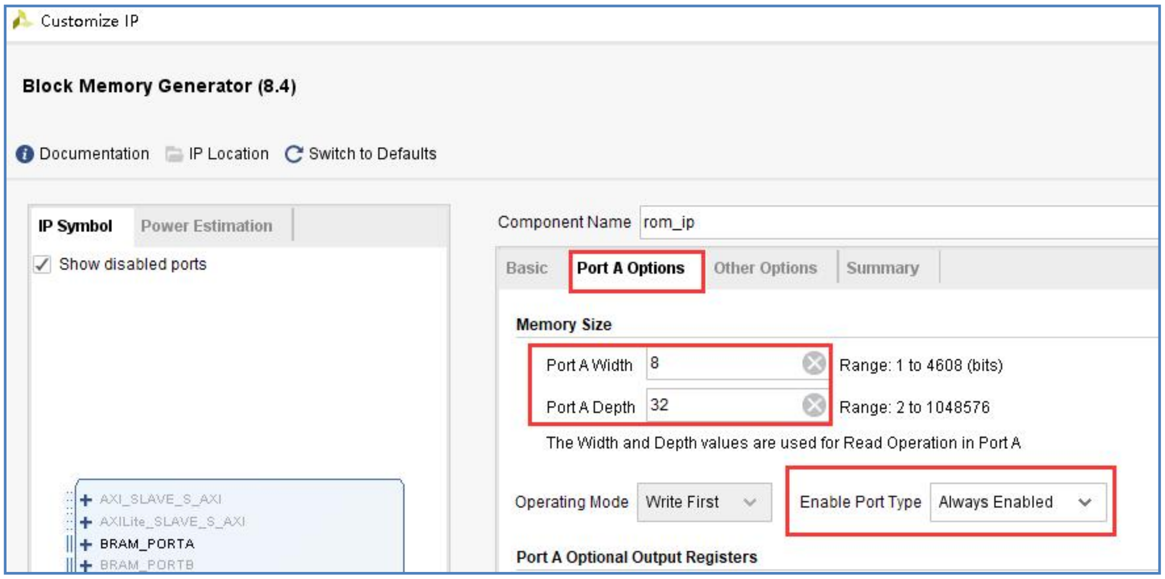


2. Change the “Component Name” to “rom_ip”. In the Basic section, change the “Memory Type” to “Single Prot ROM”



Component Name: rom_ip
Interface Type: Native
Memory Type: Single Port ROM

3. Switch to the “Port A Options” field, change the “ROM” bit width “Port A Width” to 8, change the ROM depth “Port A Depth” to 32, and enable the “Enable Port Type” to “Always Enable”


Customize IP

Block Memory Generator (8.4)

Component Name rom_ip

Memory Size

Port A Width 8 Range: 1 to 4608 (bits)

Port A Depth 32 Range: 2 to 1048576

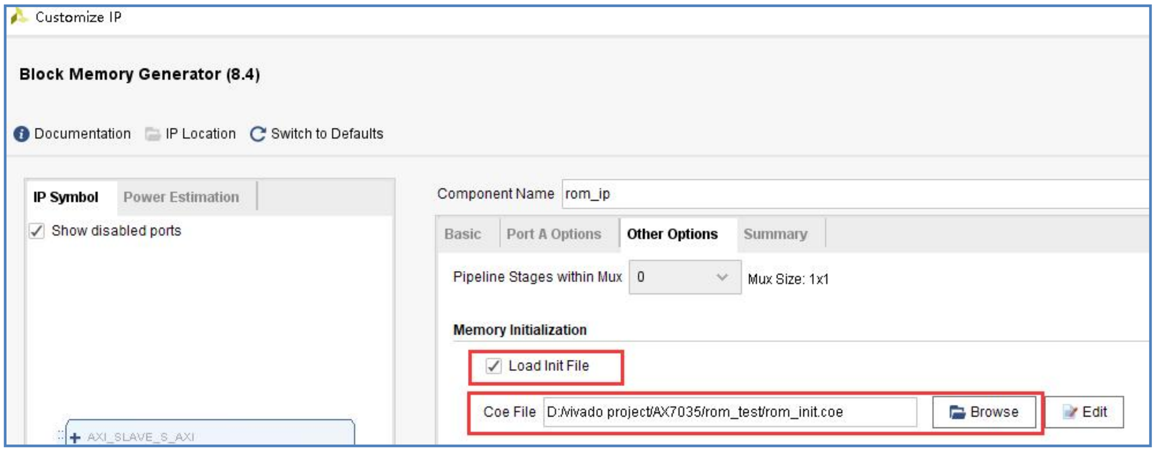
The Width and Depth values are used for Read Operation in Port A

Operating Mode Write First

Enable Port Type Always Enabled

Port A Optional Output Registers

4. Switch to the “Other Options” section, check “Load Init File”, click “Browse”, and select the “.coe” file you created earlier.


Customize IP

Block Memory Generator (8.4)

Component Name rom_ip

Pipeline Stages within Mux 0 Mux Size: 1x1

Memory Initialization

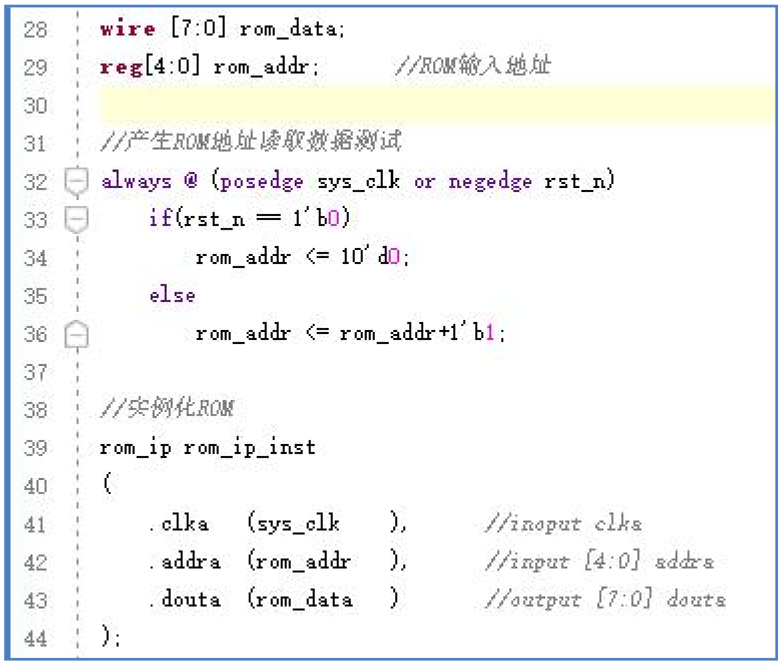
☑ Load Init File

Coe File D:/vivado project/AX7035/rom_test/rom_init.coe

5. Click ok, click “Generate” to generate the “ip” core.

3.3 ROM test program

The programming of the ROM is very simple. In the program, we only need to change the address of the ROM every clock, and the ROM will output the internal storage data of the current address. The instantiation and programming of the ROM IP are as follows:

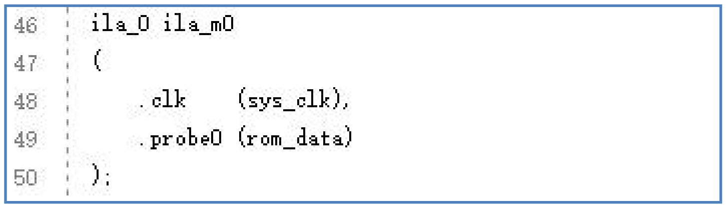


```
wire [7:0] rom_data;
reg[4:0] rom_addr;      //ROM输入地址

//产生ROM地址读取数据测试
always @ (posedge sys_clk or negedge rst_n)
    if(rst_n == 1'b0)
        rom_addr <= 10'd0;
    else
        rom_addr <= rom_addr+1'b1;

//实例化ROM
rom_ip rom_ip_inst
(
    .clka   (sys_clk    ),      //inoput clka
    .addra  (rom_addr   ),      //input [4:0] addra
    .douta  (rom_data   )       //output [7:0] douta
);
```

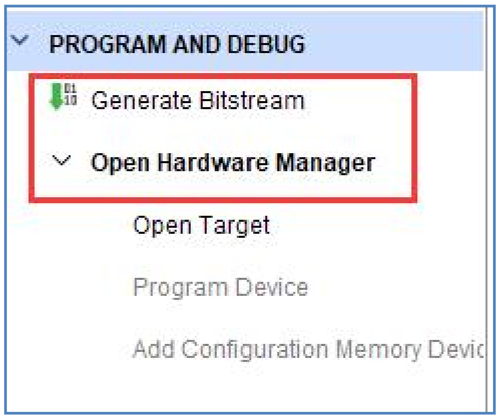
In order to see the data in the ROM in real time, we have added the “ila” tool to observe the data signal of the ROM. Please refer to the "I2C Interface EEPROM Experiment.pdf" tutorial for how to generate ila.



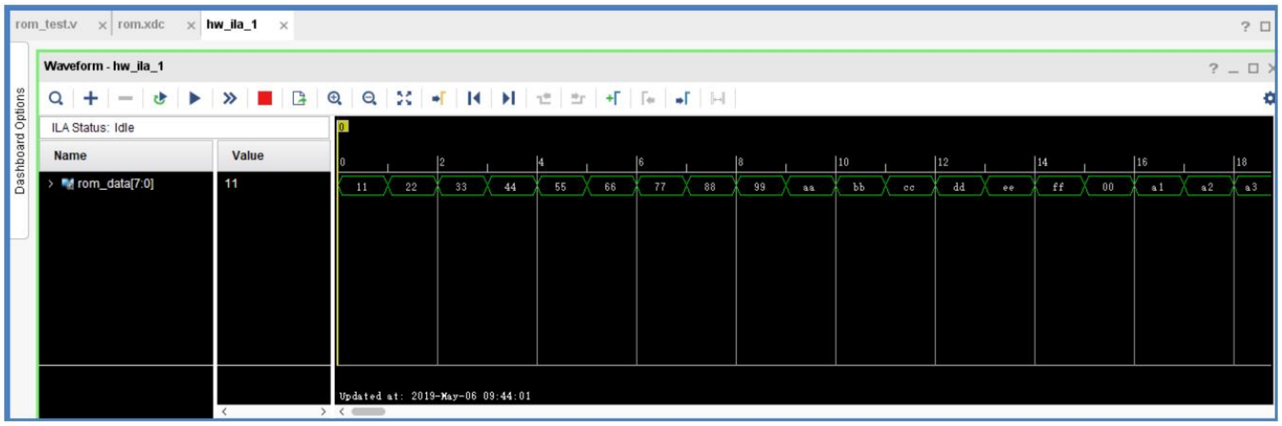
```
ila_0 ila_m0
(
    .clk    (sys_clk),
    .probe0 (rom_data)
);
```

4 Experiment Result

Add the “XDC” file, generate the “bit” file, and open the “Hardware Manager” to download the bit file to the FPGA.



In the “Waveform “window we can see that the “rom_data” data is the coe file we stored in the ROM.


rom_test.v
rom.xdc
hw_ila_1
Waveform - hw_ila_1
Dashboard Options
ILA Status: Idle
Name
Value
rom_data[7:0]
11
0
2
4
6
8
10
12
14
16
18
11
22
33
44
55
66
77
88
99
aa
bb
cc
dd
ee
ff
00
a1
a2
a3
Updated at: 2019-May-06 09:44:01