FPGA on-chip RAM read and write Experiment

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

1 Experiment Introduction

This document describes how to use the internal RAM of the FPGA and the program to read and write data to the RAM.

2 Experiment Principle

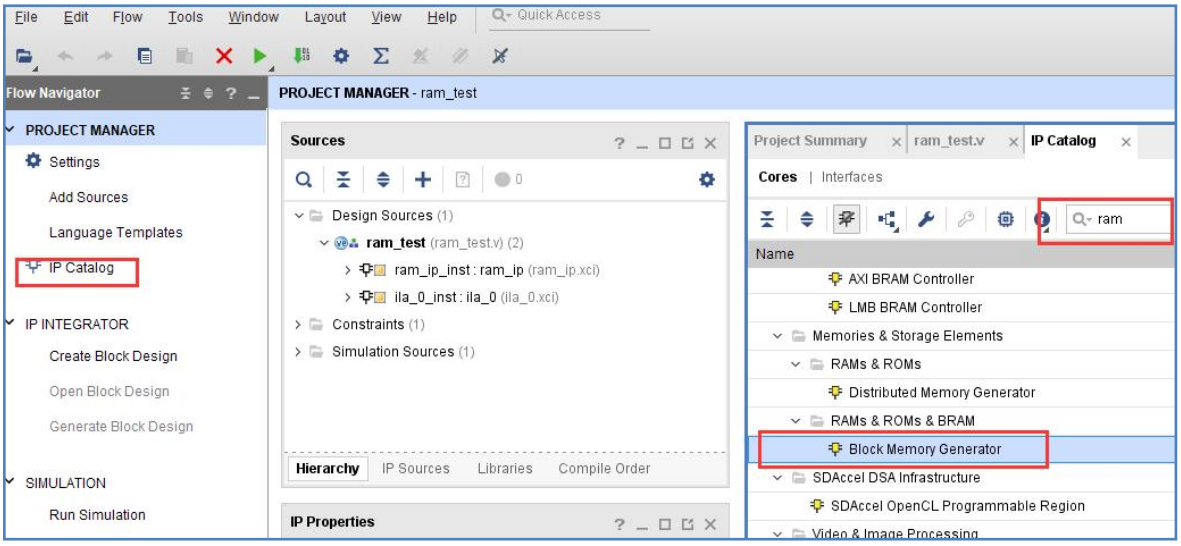
Xilinx provides the IP core of RAM in VIVADO. We only need to instantiate a RAM through the IP core to write and read the data stored in the RAM according to the read and write timing of the RAM. In the experiment, VIVADO integrated online logic analyzer ila, we can observe the RAM read and write timing and data read from RAM.

3 Programming

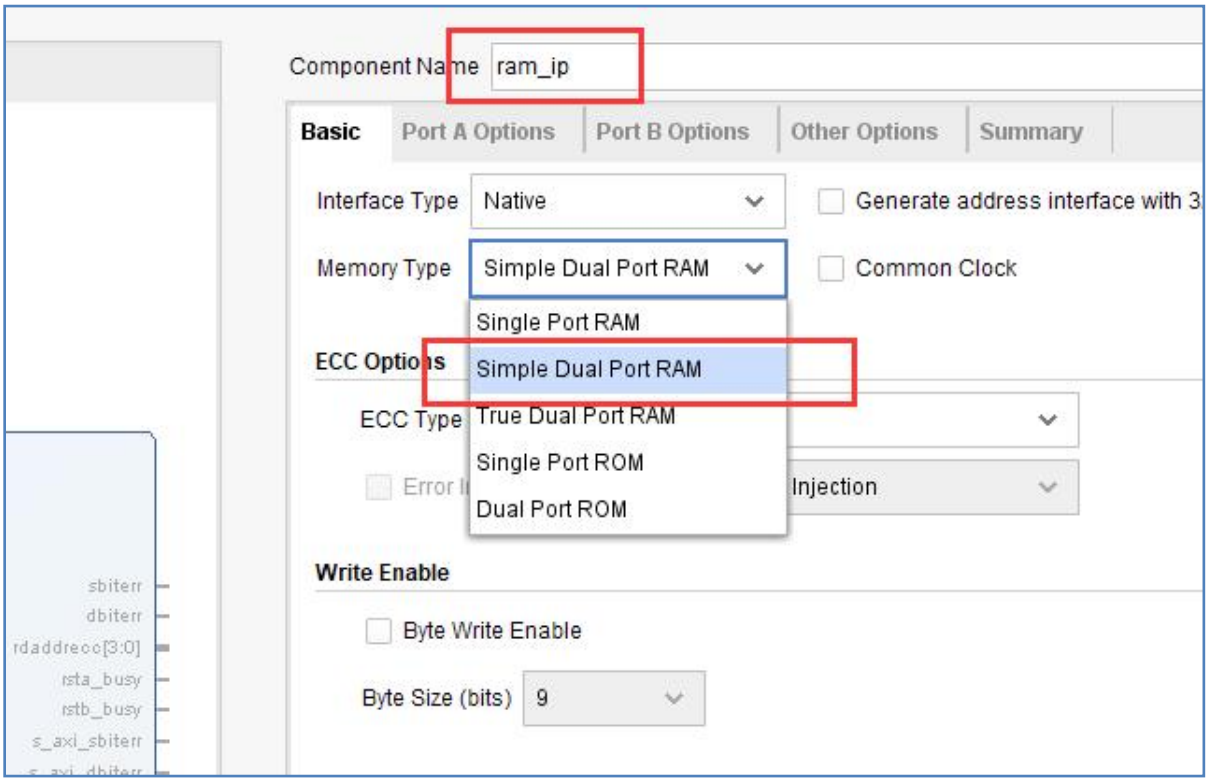
3.1 RAM IP addition and configuration

Create a new “ram_test” project before adding “RAM IP”, then add “RAM IP” to the project as follows:

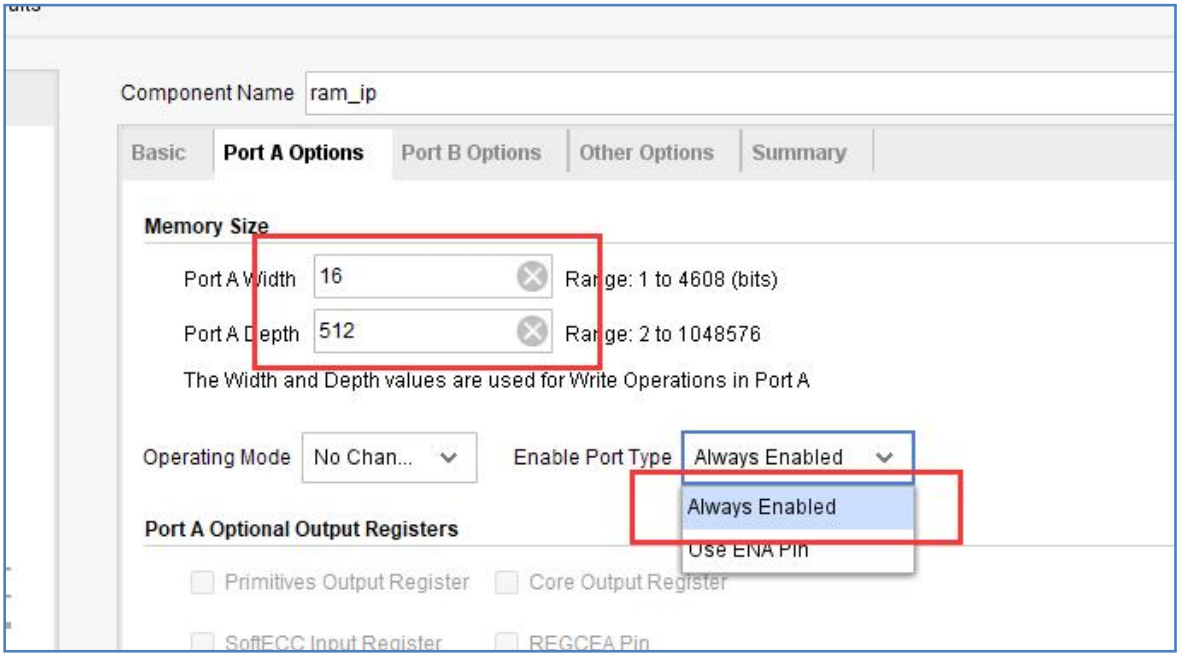
1. Click on the “IP Catalog” in the figure below, search for “ram” in the pop-up interface on the right, find the “Block Memory Generator”, and double-click to open it.



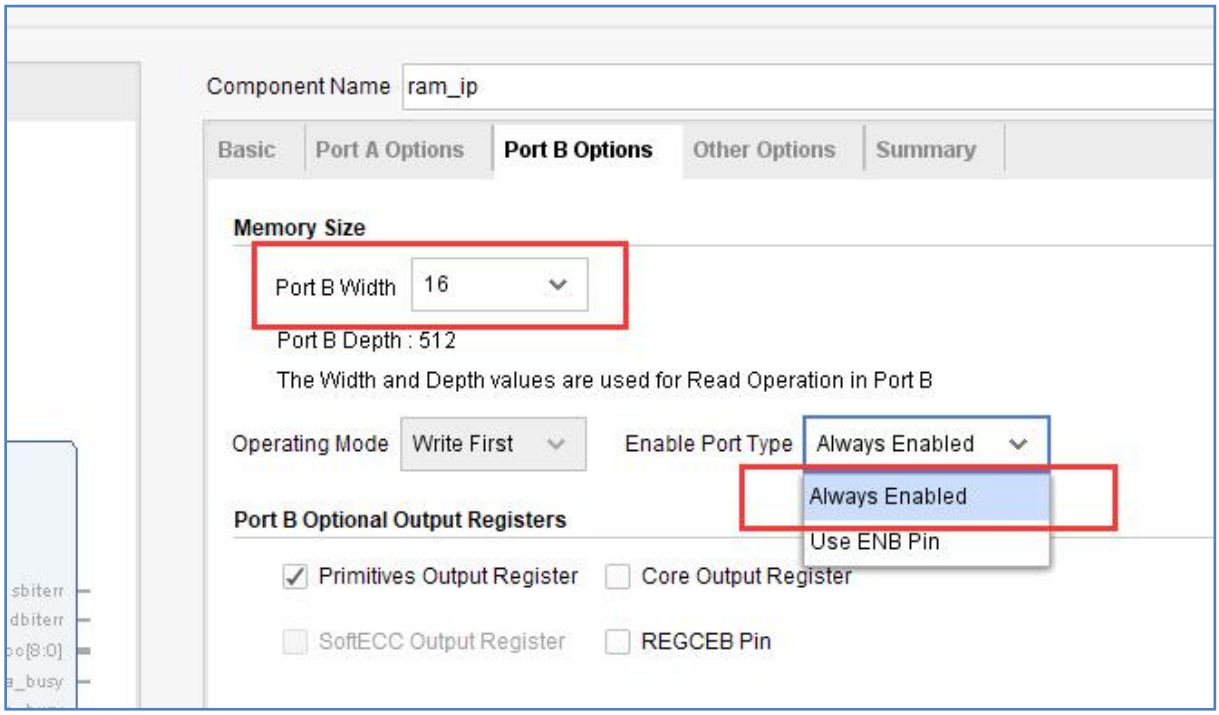
2. Change the “Component Name” to “ram_ip”. Under the “Basic” column, change the “Memory Type” to “Simple Dual Prot RAM”. Generally speaking, "Simple Dual Port RAM" is the most commonly used because it is two ports with independent input and output signals.



3. Switch to the “Port A Options” field, change the “RAM” bit width “Port A Width” to 16, change the “RAM” depth “Port A Depth” to 512, and enable the “Enable Port Type” to “Always Enable”.



4. Switch to the “Port B Options” field, change the “RAM” bit width “Port B Width” to 16, and “Enable Port Type” to “Always Enable”.



5. In the “Other Options” section, here is not the need to initialize the “RAM” data like “ROM”, we can write in the program, so the configuration can be default, just click OK.

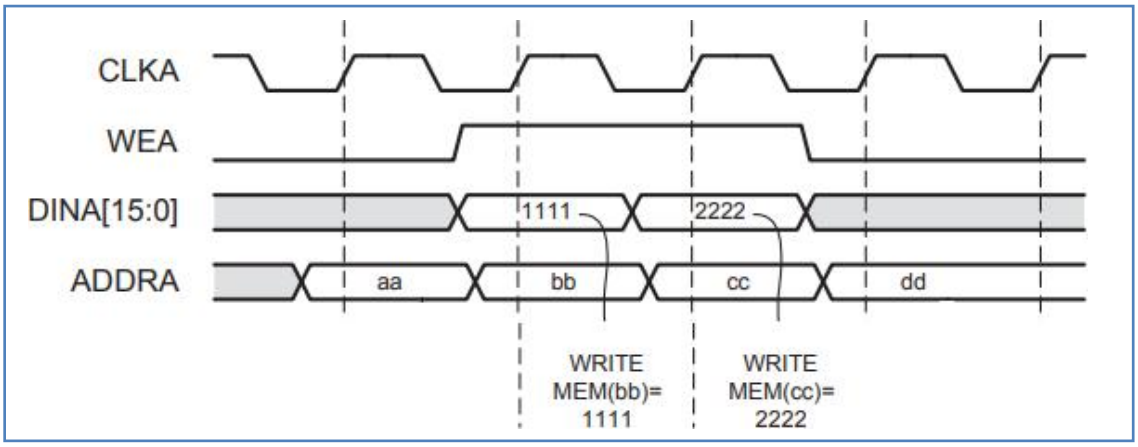
6. Click "Generate" to generate the RAM IP.

3.2 RAM port definition and timing

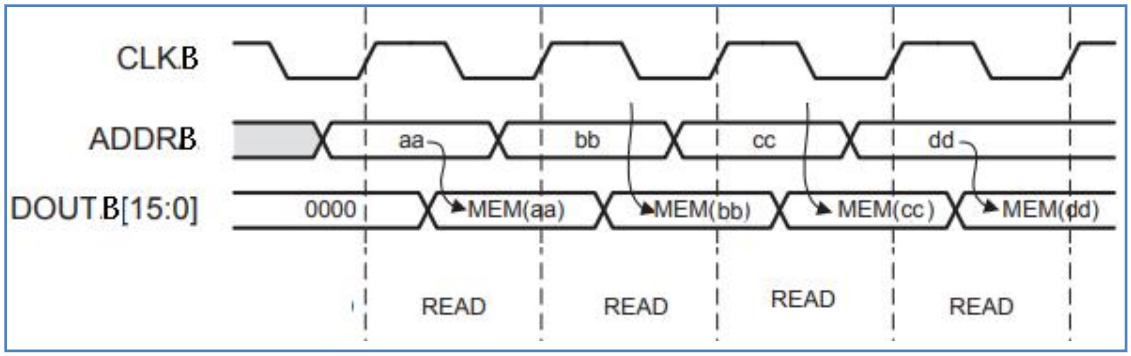
The description of the Simple Dual Port RAM module port is as follows:

Signal Name	Direction	Description
clka	in	Port A clock input
wea	in	Port A enabled
addra	in	Port A address input
dina	in	Port A data input
clkb	in	Port B clock input
addrb	in	Port B address input
doutb	out	Port B data output

The data writing and reading of the “RAM” are all operated on the rising edge of the clock. When the port A data is written, the “wea” signal needs to be set high. The following figure shows the timing chart of the input written to the “RAM”.



RAM write timing



RAM read timing

3.3 RAM test program

Add a first instantiation of RAM IP, instantiation and programming of RAM IP as follows:

```
43  //------------------------------------------------------------------
44  //实例化RAM
45  ram_ip ram_ip_inst (
46    .clka     (clk        ),    // input clka
47    .wea      (wea        ),    // input [0 : 0] wea
48    .addra    (w_addr     ),    // input [8 : 0] addra
49    .dina     (w_data     ),    // input [15 : 0] dina
50    .clkb     (clk        ),    // input clkb
51    .addrb    (r_addr     ),    // input [8 : 0] addrb
52    .doutb    (r_data     )     // output [15 : 0] doutb
53  );
```

After the program is powered on, 0~511 data will be written into the RAM.

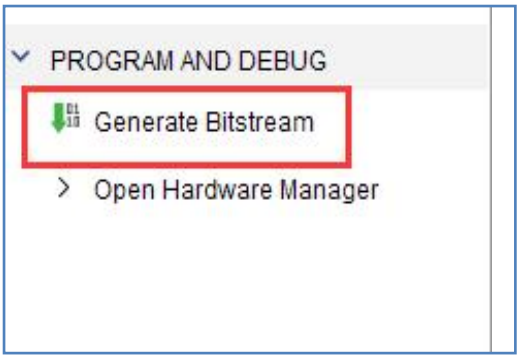
```
23  ///产生RAM写入的数据
24  always@(posedge clk or negedge rst_n)
25  begin
26    if(rst_n==1'b0) begin
27        wea <= 1'b0;
28       w_addr <= 9'd0;
29        w_data <= 16'd0;
30    end
31    else begin
32       if(w_addr==511) begin    //ram写入完毕
33          wea <= 1'b0;
34       end
35       else begin
36          wea<=1'b1;              //ram写使能
37            w_addr <= w_addr + 1'b1;
38            w_data <= w_data + 1'b1;
39        end
40    end
41  end
42
```

After continuously reading the RAM data, in order to see the data value read in the RAM in real time, we have added the ila tool to observe the RAM data signal. Please refer to the "I2C Interface EEPROM Experiment.pdf" tutorial for how to generate ila.

```
56  ila_0 ila_0_inst (
57      .clk(clk),
58      .probe0(r_data),
59      .probe1(r_addr)
60  );
```

4 Experiment Result

Download the “bit” file to the FPGA. Next we use “ila” to observe whether the data read from “RAM” is the data of our initialization file “coe”.



"Waveform" window we can see that "r_addr" is constantly being added from 0 to 1ff. With the change of "r_addr"," r_data" is also changing. The data of "r_data" is exactly 512 data we write to RAM. It should be noted here that when "r_addr" has a new address, the data corresponding to "r_data" will be delayed by two clock cycles, and the data will appear two clock cycles later than the address.