


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42



计算机组成实验指导书-LAB4

 上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	1 of 18
	作者	修改日期	公开	
	WnSN Lab	9/21/2012		

1. OVERVIEW

1.1 实验名称

简单的类 MIPS 单周期处理器实现 –寄存器与内存

1.2 实验目的

1. 理解 CPU 的寄存器与内存
2. 使用 Verilog 语言设计存储器件
3. 使用 ISim 进行行为仿真

1.3 实验范围

本次实验将覆盖以下范围

1. ISE 的使用
2. Register 的实现
3. Data Memory 的实现
4. 有符号扩展的实现


1.4 实验预计时间

150~180 分钟

1.5 实验报告与验收办法

1.6 注意事项

1. 本实验的逻辑设计工具为 Xilinx ISE13.4，但不仅如此，学生可以使用自己喜欢的逻辑设计工具，如 Snyplify 等。

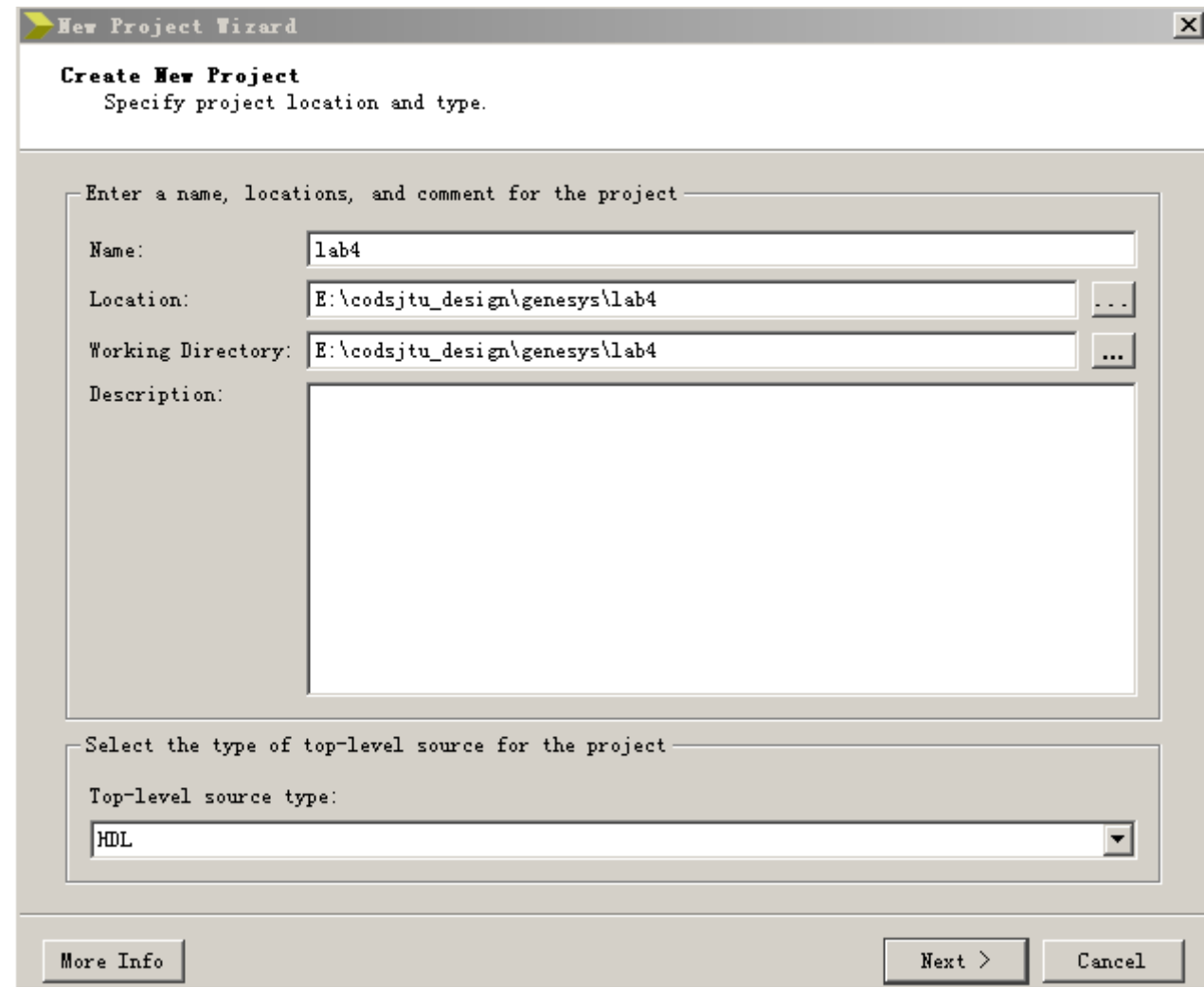
 上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1 作者 WnSN Lab	CSE-COA-LAB-004 修改日期 9/21/2012	0.3	2 of 18 公开

2. 新建工程


2.1 实验描述

2.1.1 新建工程

1. 打开 ISE 工具，新建工程



2. 选择 FPGA 型号、综合和仿真工具、推荐描述语言等配置。

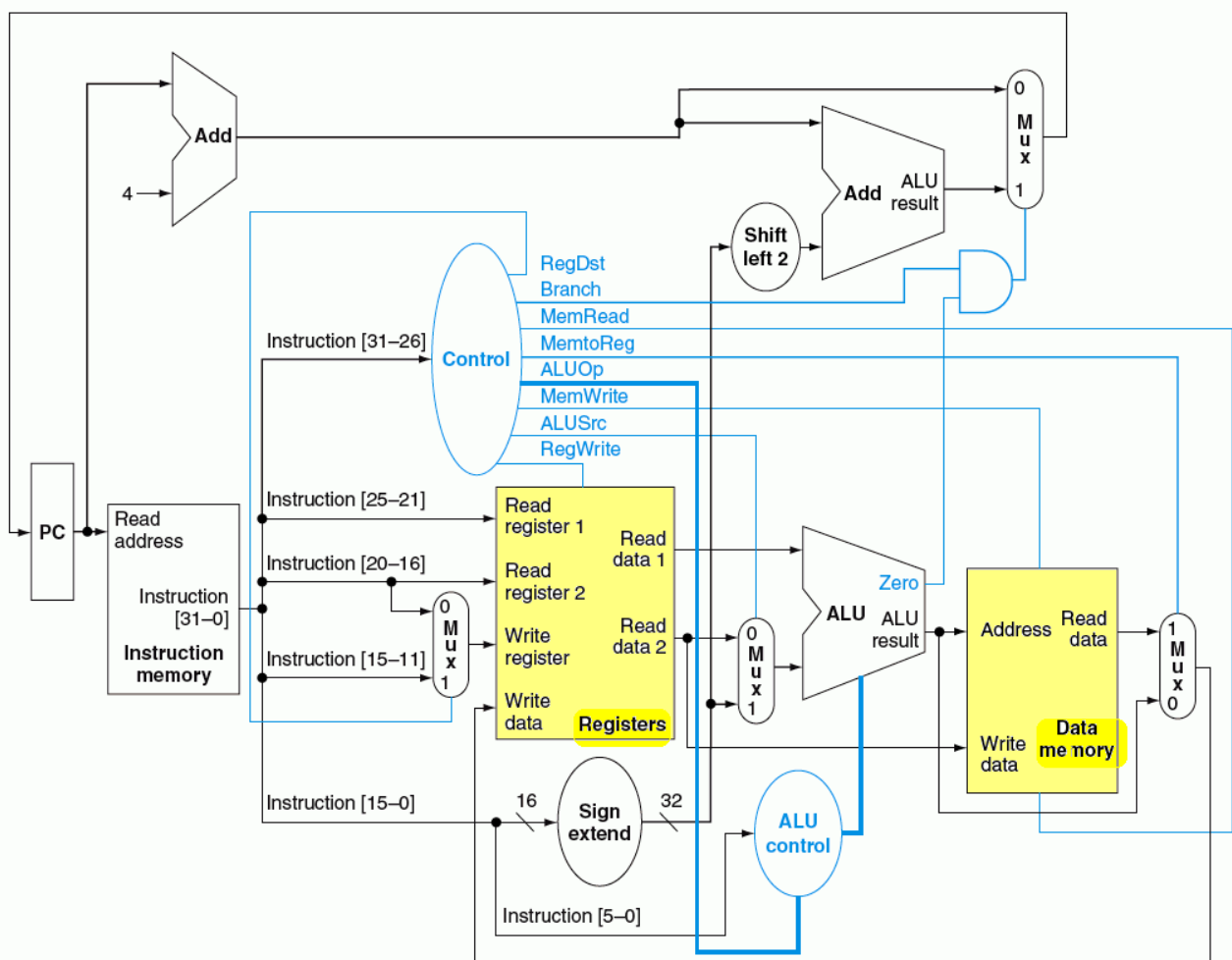
 上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	3 of 18
	作者	修改日期	公开	
	WnSN Lab	9/21/2012		

3. 寄存器单元模块 REGISTER

3.1 实验描述

3.1.1 模块描述

寄存器是指令操作的主要对象，MIPS 中一共有 32 个 32 位的寄存器。

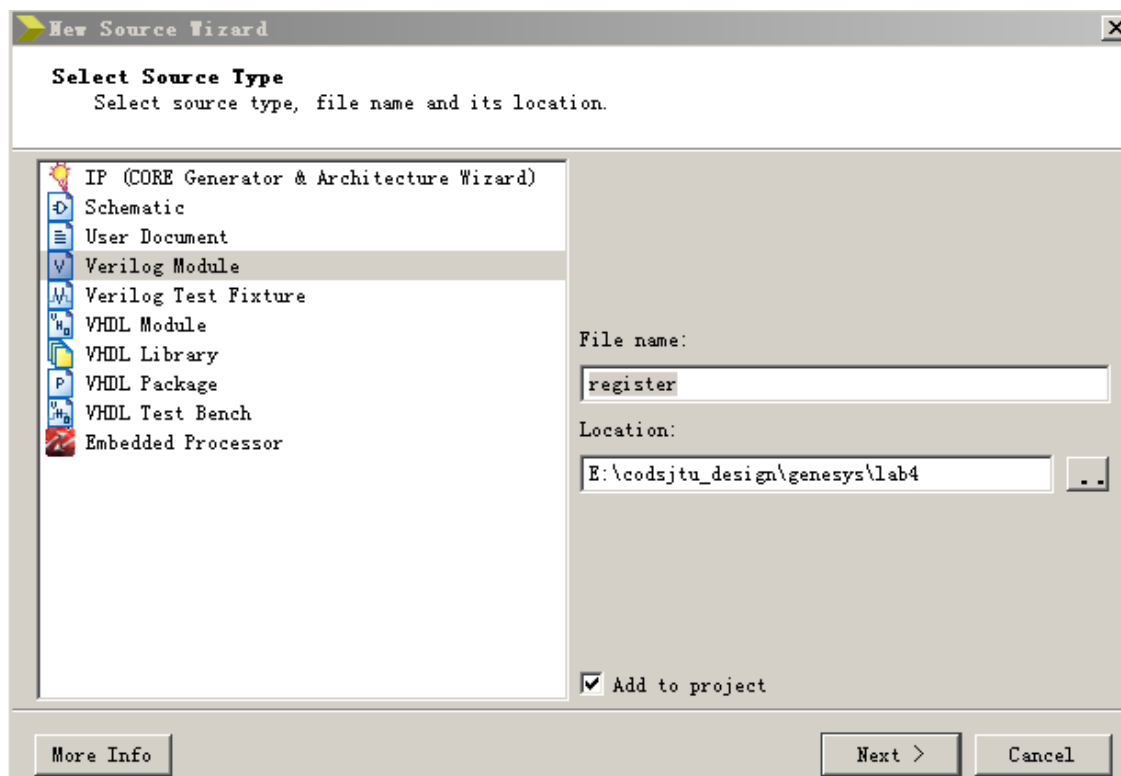


(MIPS 处理器基本架构图寄存器和存储器单元)

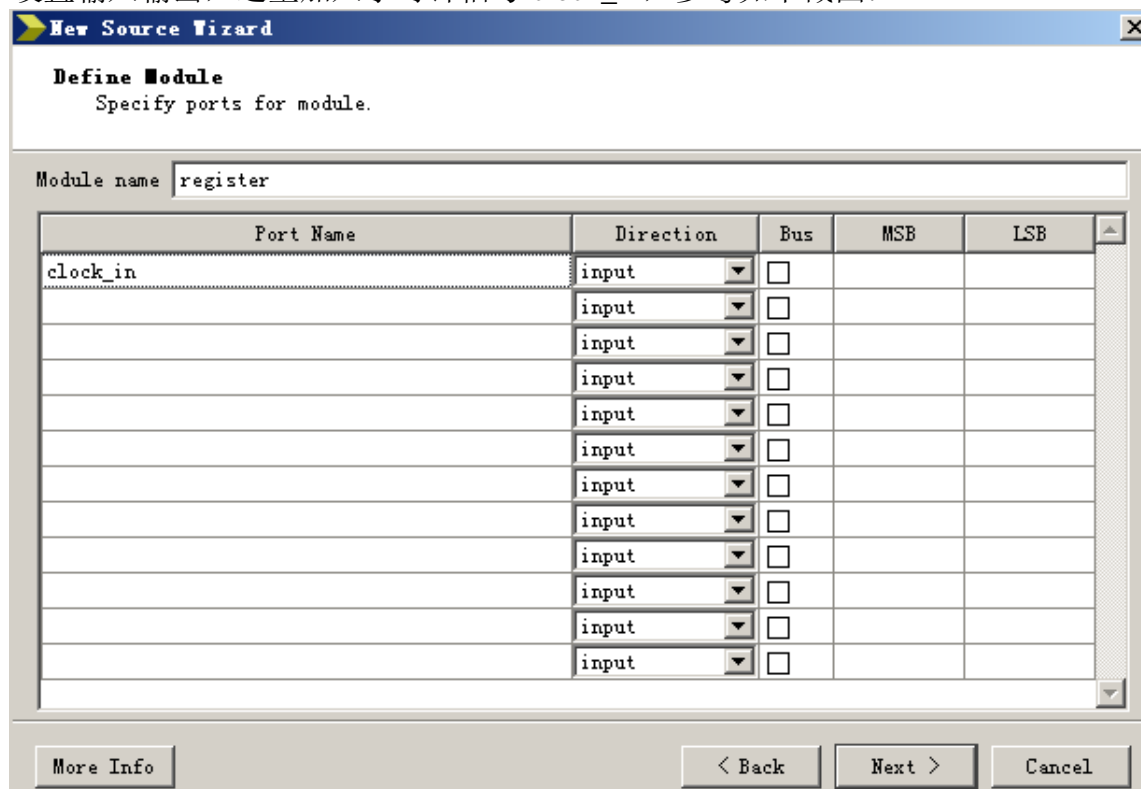
3.1.2 新建模块源文件


1. 创建 Register.v 模块

上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	4 of 18
	作者	修改日期	公开	
	WnSN Lab	9/21/2012		



2. 设置输入输出，这里加入了时钟信号 clock_in，参考如下截图：



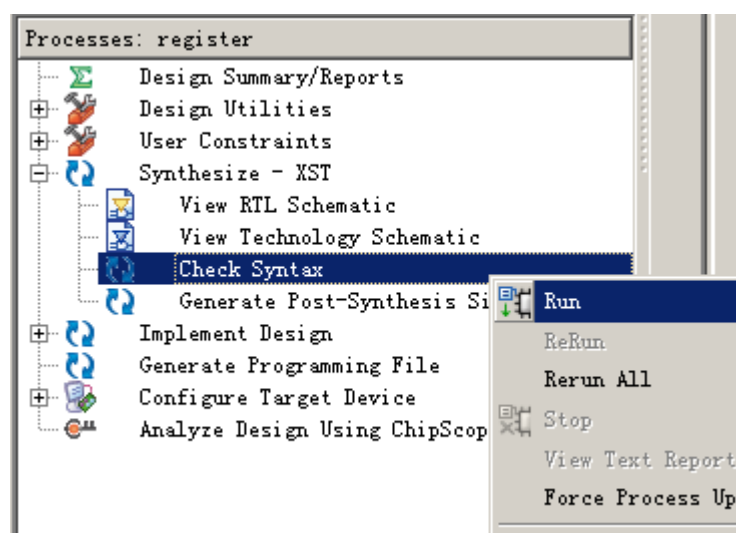
	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	5 of 18
	作者	修改日期	公开	
	WnSN Lab	9/21/2012		

3.1.3 编写功能代码

这里需要注意的是，由于不确定 WriteReg, WriteData, RegWrite 信号的先后次序，我们采用时钟的下降沿作为写操作的同步信号，防止发生错误。

```
21 module register( clock_in, readReg1, readReg2, writeReg,.....
22     input clock_in;
23     //.....
24
25     |
26
27
28     reg [31:0] regFile[31:0]; //registers space: 32*32bits
29
30     //.....
31
32     always @ (readReg1 or readReg2 or ..... )
33     begin
34         //HOW TO DO
35     end
36
37     always @ (negedge clock_in)
38     begin
39         //HOW TO DO
40     end
41
42
43 endmodule
```

写完代码后在综合选项中，如图运行语法检查：



上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	6 of 18
	作者	修改日期	公开	
	WnSN Lab	9/21/2012		

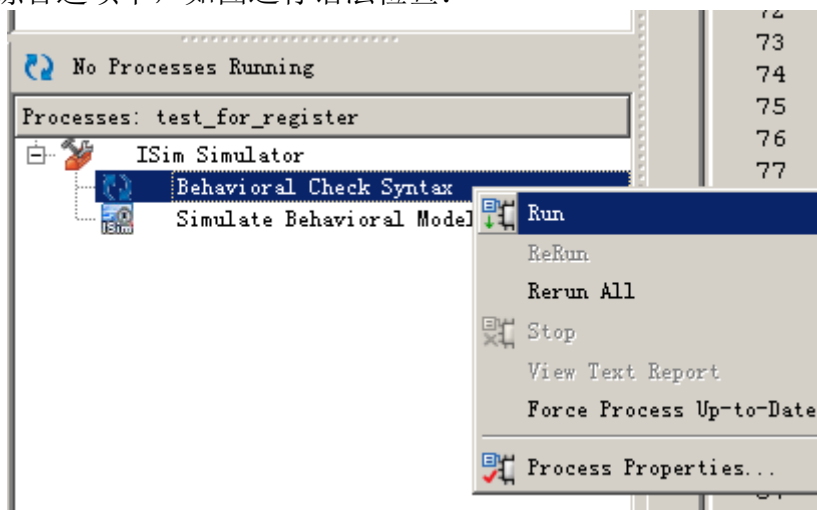
3.1.4 创建 test_for_register.v 测试文件，添加激励信号，进行行为仿真。

1. View:选中 Simulation

2. 添加激励信号如下图，进行行为仿真。使用 clock_in 作为时钟输入，仿真运行周期自定，至少仿真 3 个周期，这里设为 3000ns。时钟周期暂设为 200ns。

```
72     initial begin
73         //.....
74
75         #285; //----- Current Time: 285ns
76         regWrite = 1'b1;
77         writeReg  = 5'b10101;
78         writeData = 32'b11111111111111110000000000000000;
79         //.....
80
81         #200; //----- Current Time: 485ns
82         writeReg = 5'b01010;
83         writeData = 32'b00000000000000000111111111111111;
84         //.....
85
86         #200; //----- Current Time: 685ns
87         regWrite = 1'b0;
88         writeReg  = 5'b00000;
89         writeData = 32'b00000000000000000000000000000000;
90         //.....
91
92         #50; //----- Current Time: 735ns
93         readReg1 = 5'b10101;
94         readReg2 = 5'b01010;
95         //.....
96     end
```

3. 写完代码后在综合选项中，如图运行语法检查：

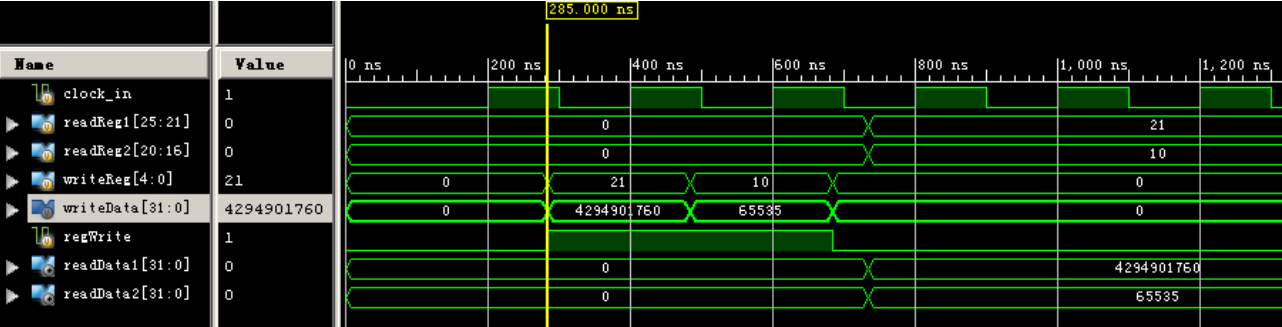
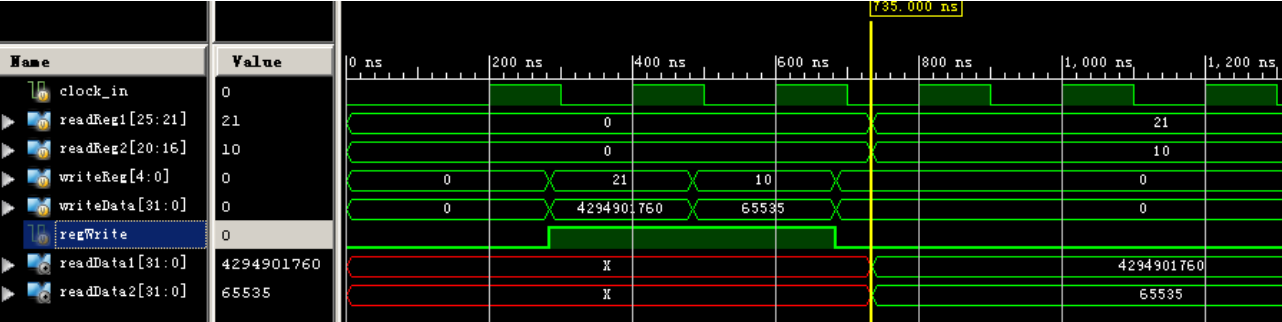


上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	7 of 18
	作者	修改日期	公开	
	WnSN Lab	9/21/2012		

4. ISim 仿真中，观察波形，查看仿真结果，是否满足当初的设计。如果有错，检查修改代码，重新仿真（操作小技巧：小键盘 -， +， 快速缩放波形视野）。

5. 下面给出二个仿真样例：

观察仿真波形时，可依据个人喜好调整信号的查看顺序。



3.2 实验报告

上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	8 of 18
	作者	修改日期	公开	
	WnSN Lab	9/21/2012		

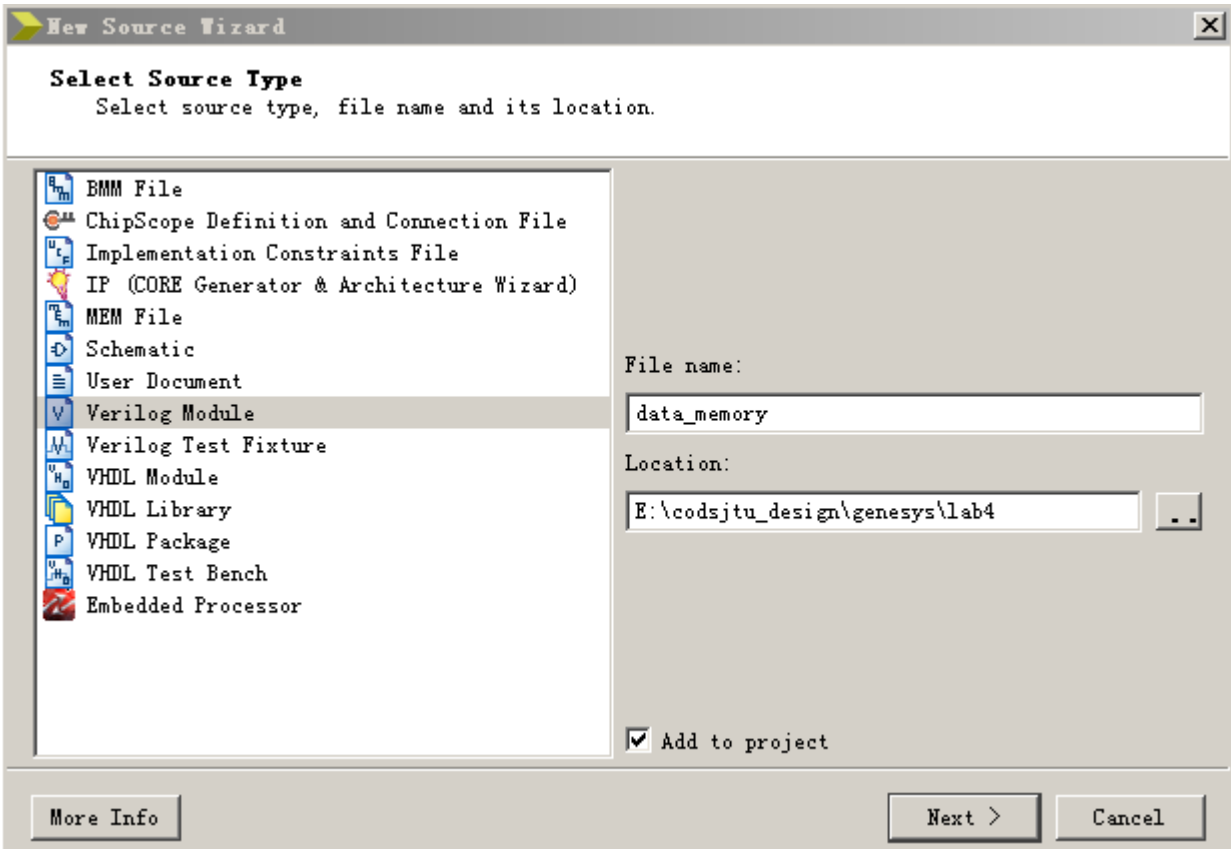
4. 内存单元模块 MEMORY


4.1 实验描述

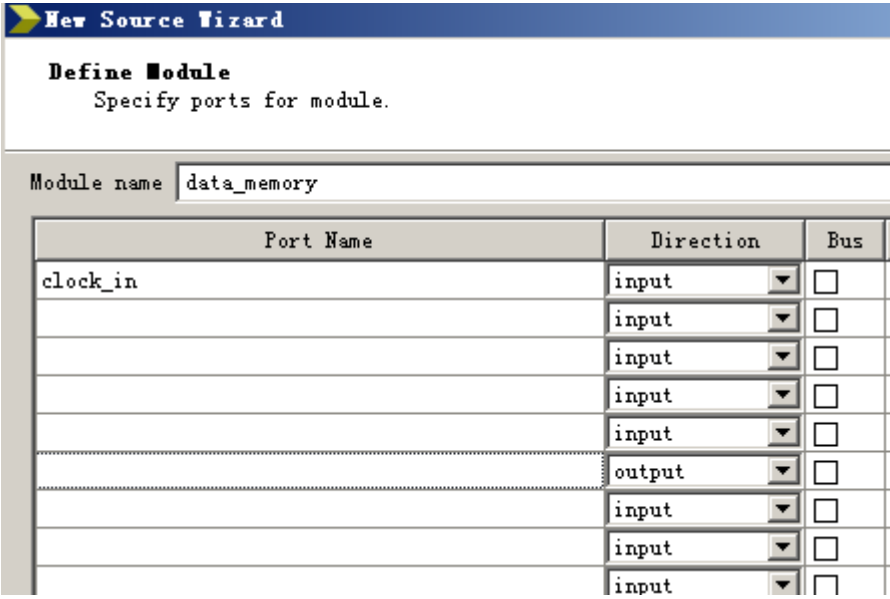
4.1.1 模块描述

内存本模块与 register 类似，由于写数据也要考虑信号同步，因此也需要 clock_in。内存单元的实现，也可用系统 Block Memory 来生成。可参见本实验指导最后附录的部分图示。

4.1.2 新建模块源文件




	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	9 of 18
	作者	修改日期	公开	
	WnSN Lab	9/21/2012		



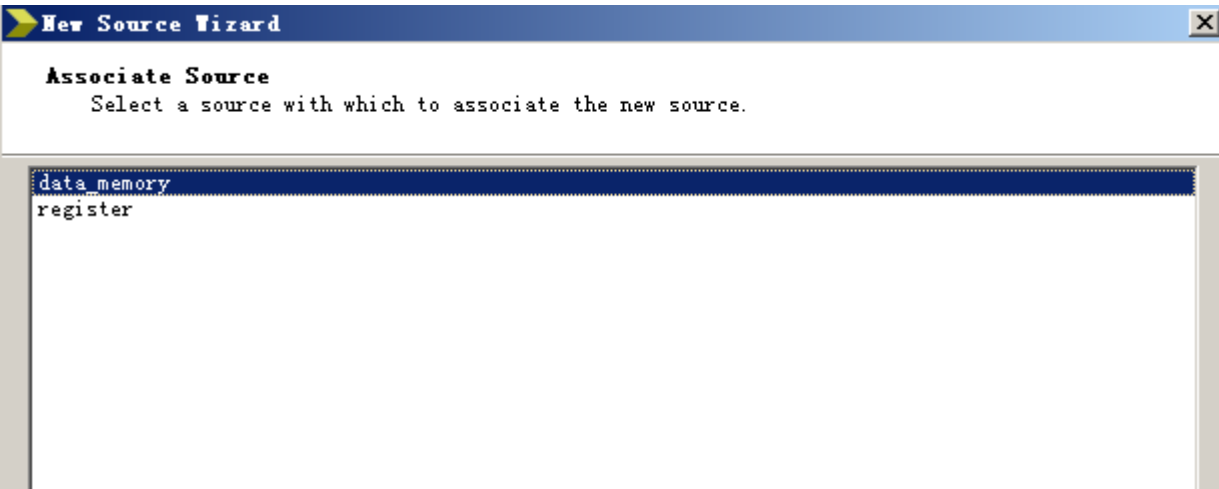
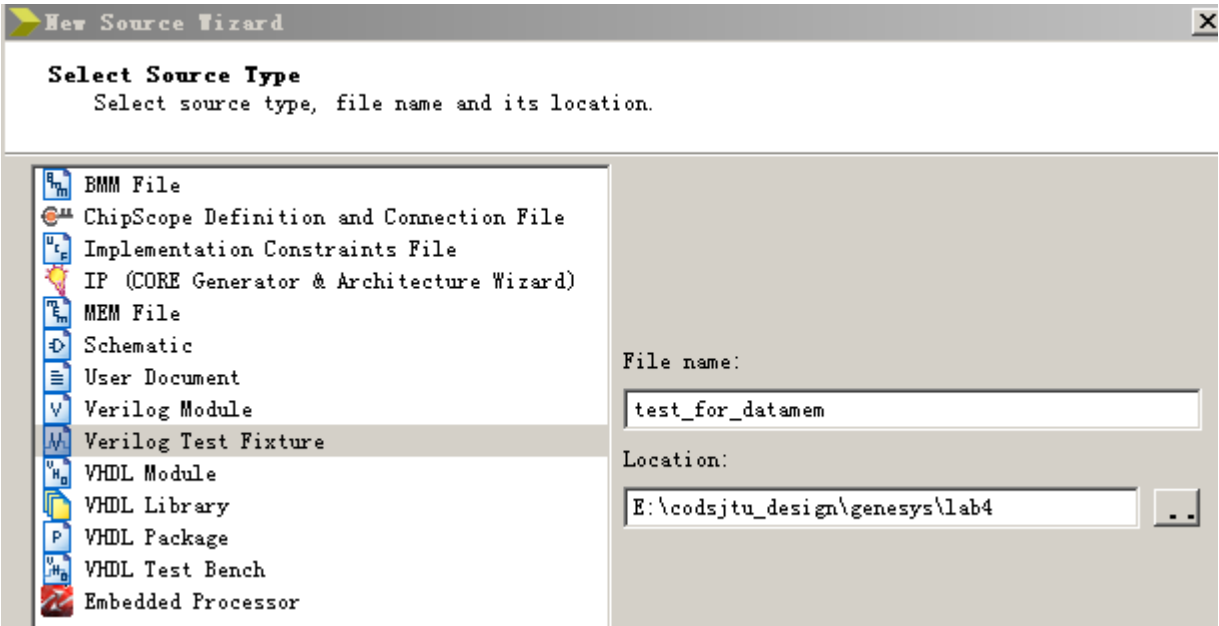
4.1.3 编写功能（从这个模块起，可采用 Verilog 关于模块端口定义的另一种精练的编写方式）

```
20 //////////////////////////////////////////////////
21 module data_memory(
22     input clock_in,
23     input [31:0] ?,
24     input ?,
25     input ?,
26     input ?,
27     output reg[31:0] ?
28 );
29
30 reg [31:0] memFile[0:127]; //memory space: 128*32bits
31 |
32 //.....
33
34 always @( /* conditions? */ )
35 begin
36     //HOW TO DO
37 end
38
39 always @(/* which edge? */)
40 begin
41     //HOW TO DO
42 end
43
44 endmodule
```


 上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1 作者 WnSN Lab	CSE-COA-LAB-004 修改日期 9/21/2012	0.3	10 of 18 公开

4.1.4 仿真

1. 根据之前叙述的方法创建 test_for_datamem.v 测试文件，添加激励信号，进行仿真。



2. 添加激励信号如下图，修改代码进行行为仿真。在 testBench 中设定不同的输入。请覆盖所有的情况，以保证逻辑的正确。

	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	11 of 18
	作者	修改日期	公开	
	WnSN Lab	9/21/2012		

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12

13
14

15

16

24
25

33

38
39
40
41
42

5. 带符号扩展

5.1 实验描述

5.1.1 模块描述

将 16 位有符号数扩展为 32 位有符号数。

补码：

(1) 正数的补码：与原码相同。

+9 的补码是 00001001。

(2) 负数的补码：符号位为 1，其余位为该数绝对值的原码按位取反；然后整个数加 1。
求-7 的补码。

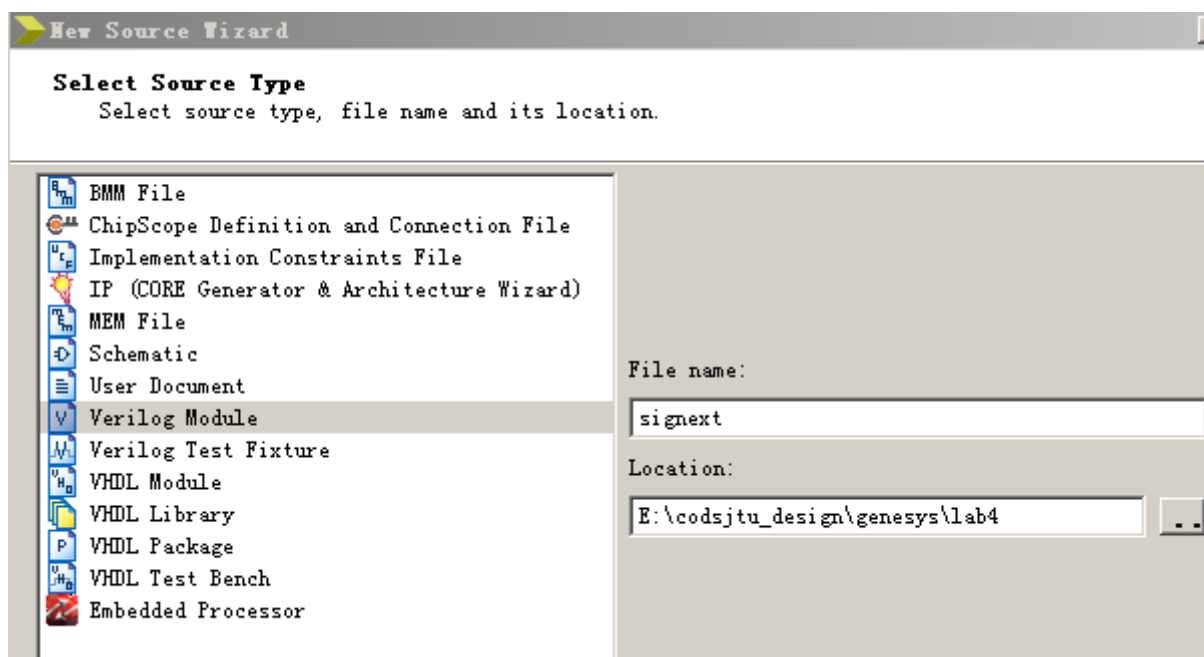
因为给定数是负数，则符号位为“1”。

后七位：+7 的原码（0000111）→按位取反（1111000）→加 1（1111001）

所以-7 的补码是 11111001。


带符号扩展只需要在前面补足符号即可。

5.1.2 新建模块源文件



5.1.3 实现功能

将符号补齐。

	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	13 of 18
	作者	修改日期	公开	
	WnSN Lab	9/21/2012		

```


1      20 //////////////////////////////////////////////////
2      21 module signext(
3      22     input [15:0] inst,
4      23     output [31:0] data
5      24 );
6      25
7      26     assign data = ; //HOW TO DO
8      27
9      28 endmodule

```

5.1.4 仿真

1. 可创建 test_for_signext.v 测试文件。
2. 选取几个输入数据，覆盖不同的情况，保证逻辑正确。
3. 观察波形是否满足设计逻辑。

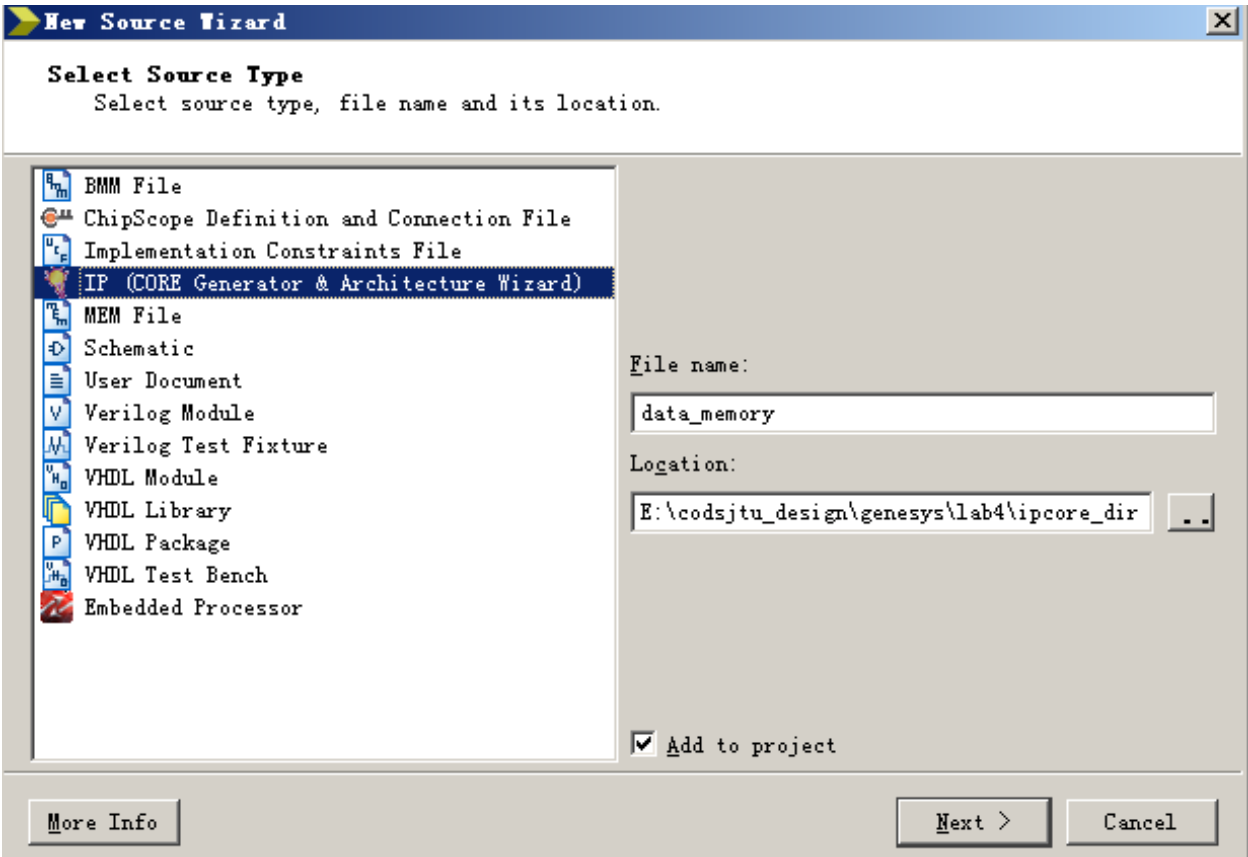
5.2 实验报告

 上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	14 of 18
	作者	修改日期	公开	
	WnSN Lab	9/21/2012		

6. 附录（图示）：

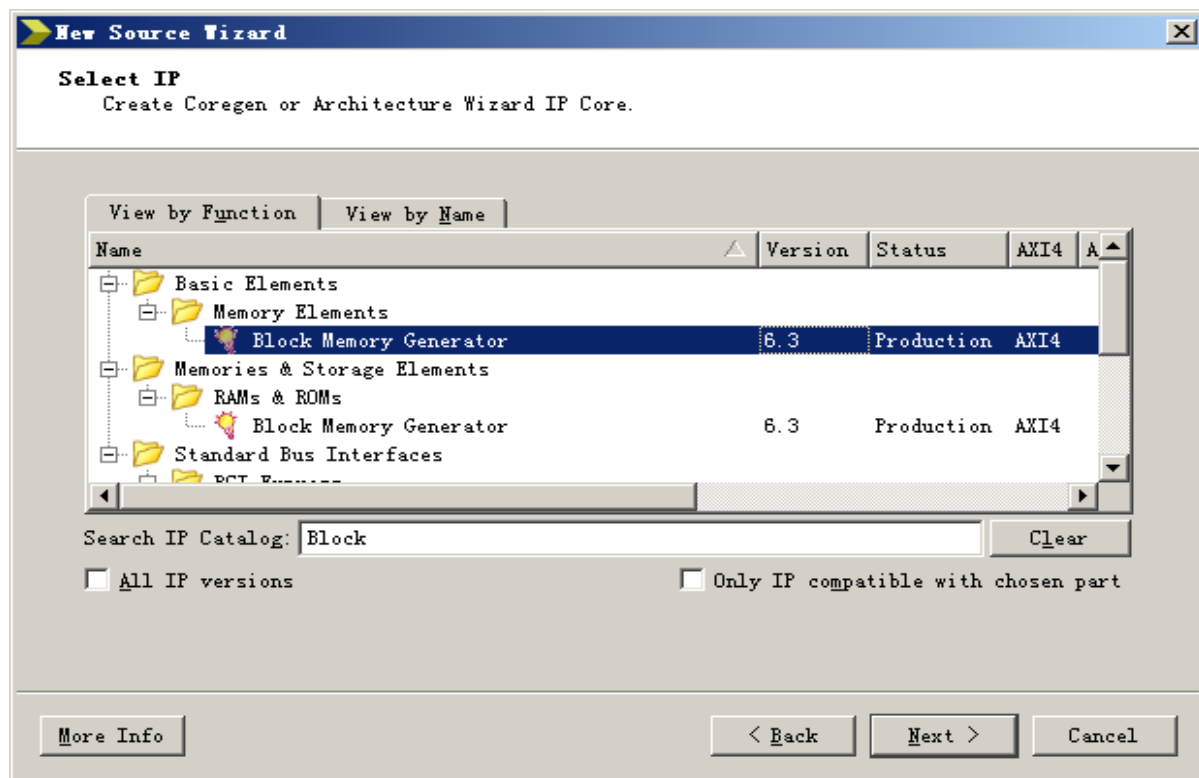
内存既可用类似寄存器的方法来实现，也可用 Block Memory 实现。采用 BRAM 来设计 Data memory 是较方便和有效的。

- 1. 在 new source 对话框中选择选择 IP（CORE Generator & Architecture Wizard）

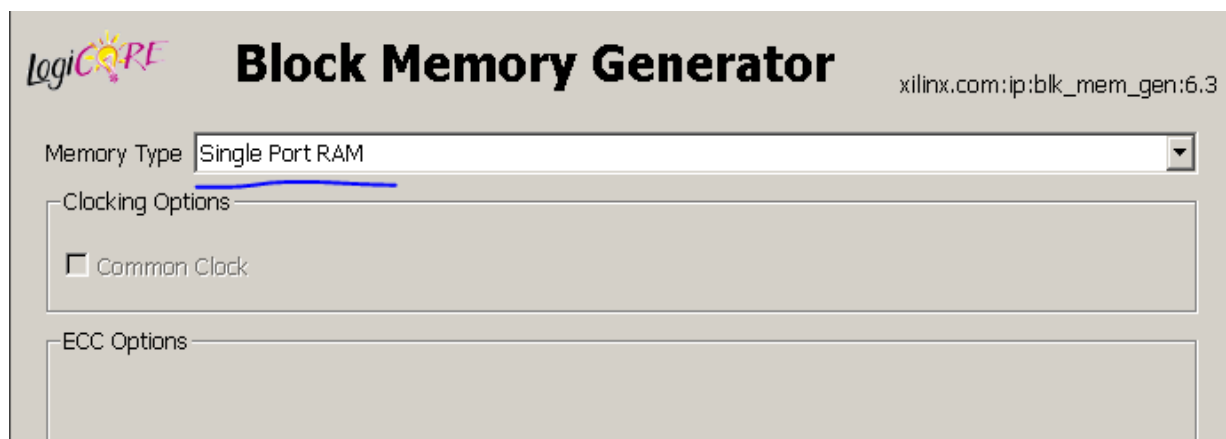



<div>上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering</div>	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	15 of 18
	作者	修改日期	公开	
	WnSN Lab	9/21/2012		

2. IP 中选择 Block Memory Generator

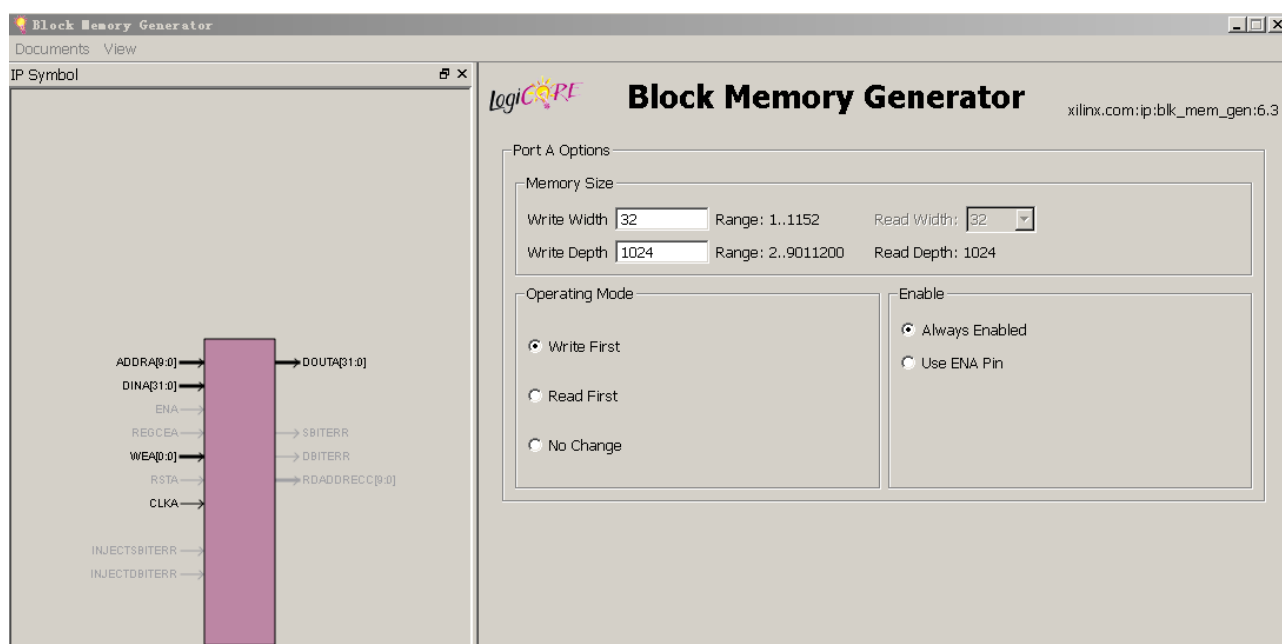


3. 配置 Block Memory 的参数，选择 RAM 的端口

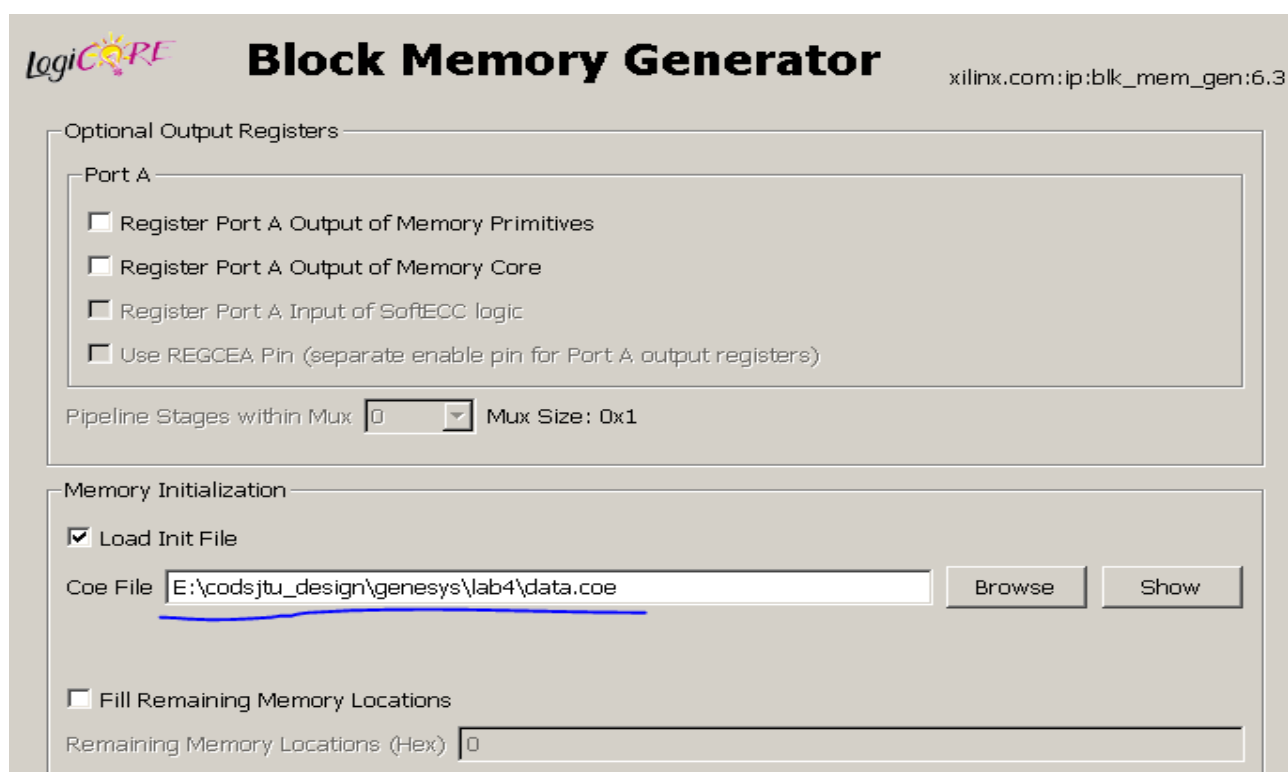



 上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	16 of 18
	作者	修改日期	公开	
	WnSN Lab	9/21/2012		

4. 配置 Block Memory 宽度和深度



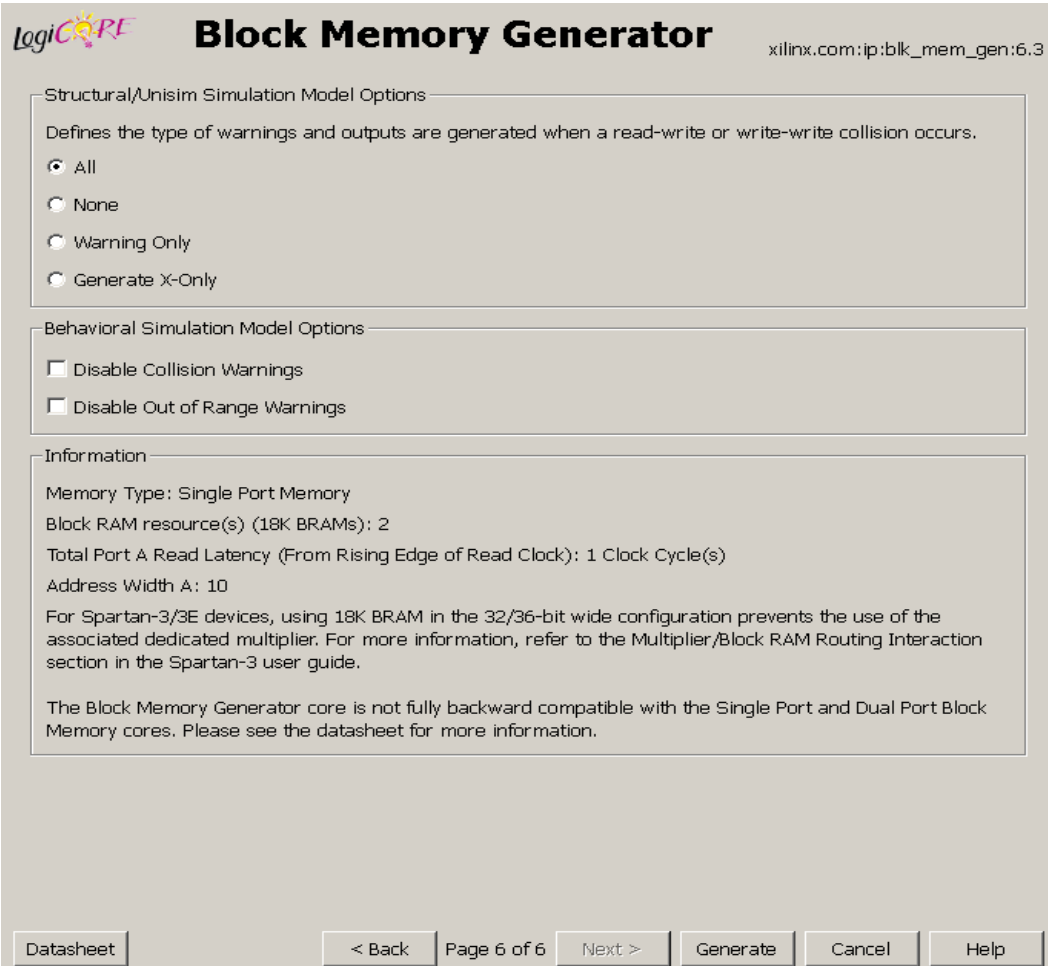
5. 装载 Block Memory 初始化文件以及 coe 文件格式




 上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	17 of 18
	作者 WnSN Lab	修改日期 9/21/2012	公开	

6. COE 文件的编写格式头两句严格必须，如下：

```
memory_initialization_radix=16;
memory_initialization_vector=
00000001,
00000005,
00000008,
00000000,
00000000,
00000000,
00000000,
00000000,
00000000,
00000000,
00000000,
00000000,
00000000,
00000000,
00000000,
00000000
```



7. 点击 Generate，建立 IP 内存。

 上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering	标题	文档编号	版本	页
	计算机组成实验指导书 LAB1	CSE-COA-LAB-004	0.3	18 of 18
	作者	修改日期		
	WnSN Lab	9/21/2012	公开	