


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-LAB5

| | | | | |
|---|-----------------|-----------------|-----|---------|
|  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-005 | 0.3 | 1 of 14 |
| | 作者 | 修改日期 | 公开 | |
| | WnSN Lab | 09/21/2010 | | |

1. OVERVIEW

1.1 实验名称

简单的类 MIPS 单周期处理器实现 – 整体调试

1.2 实验目的

完成单周期的类 MIPS 处理器

1.3 实验范围

本次实验将覆盖以下范围

1. ISE 的使用
2. Xilinx Spartan 3E 实验板的使用
3. 使用 VerilogHDL 进行逻辑设计
4. 仿真测试、下载验证


1.4 实验预计时间

240 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。

| | | | | |
|--|-----------------|-----------------|-----|---------|
|  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-005 | 0.3 | 2 of 14 |
| | 作者 | 修改日期 | 公开 | |
| | WnSN Lab | 09/21/2010 | | |

2. 新建工程

2.1 实验描述

2.1.1 新建工程

1. 启动 ISE 13.4。
2. 新建工程 lab5。

New Project Wizard

Create New Project
Specify project location and type.

Enter a name, locations, and comment for the project

Name: LAB5

Location: E:\codsjtu_design\genesys\LAB5

Working Directory: E:\codsjtu_design\genesys\LAB5


Description:

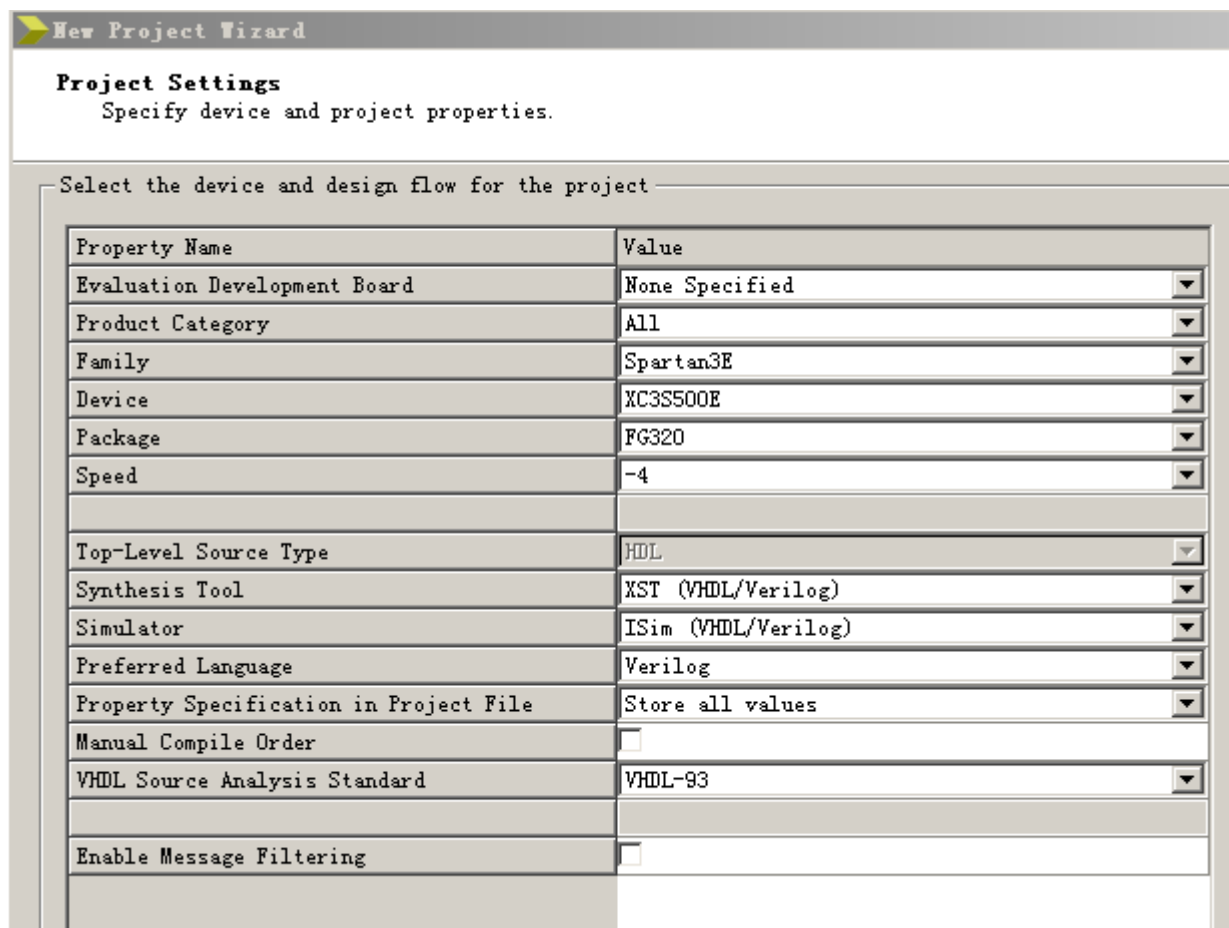
Select the type of top-level source for the project

Top-level source type: HDL

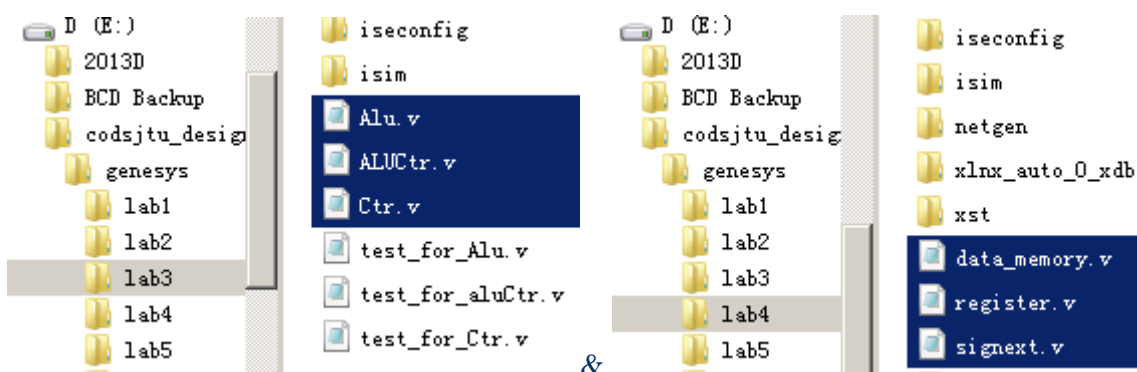
More Info Next > Cancel

3. 选择 FPGA 型号、综合和仿真工具、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-005 | 0.3 | 3 of 14 |
| | 作者 | 修改日期 | 公开 | |
| | WnSN Lab | 09/21/2010 | | |

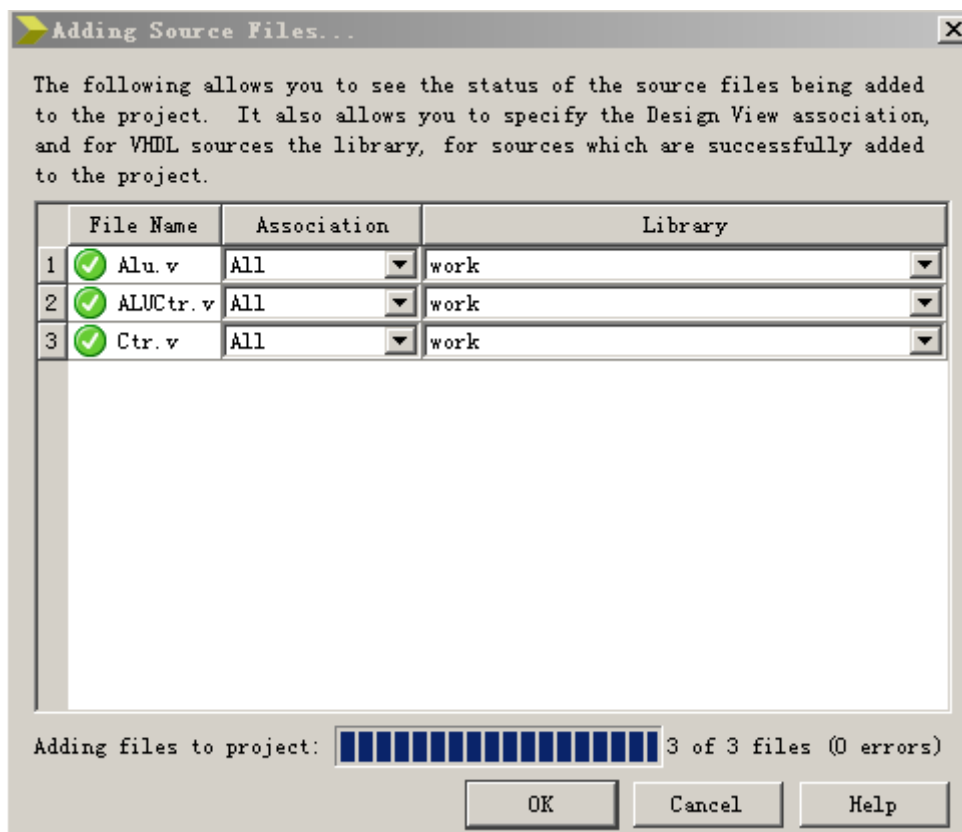


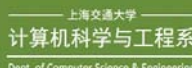
4. 右键点击 Hierarchy 窗口，选择 Add Copy of Source，添加已有的模块。将此前两次实验中的模块文件 (*.v) 拷贝到 lab5 工程目录下。不过需建立顶层模块 Top 和主控制模块 Control 需要重新定义。



5. Adding Source Files... 中，选中全部要添加的文件，保持默认项，点 OK。

| | | | | |
|--|-----------------|-----------------|-----|---------|
| | 标题 | 文档编号 | 版本 | 页 |
| | 计算机组成实验指导书 LAB1 | CSE-COA-LAB-005 | 0.3 | 4 of 14 |
| | 作者 | 修改日期 | 公开 | |
| | WnSN Lab | 09/21/2010 | | |

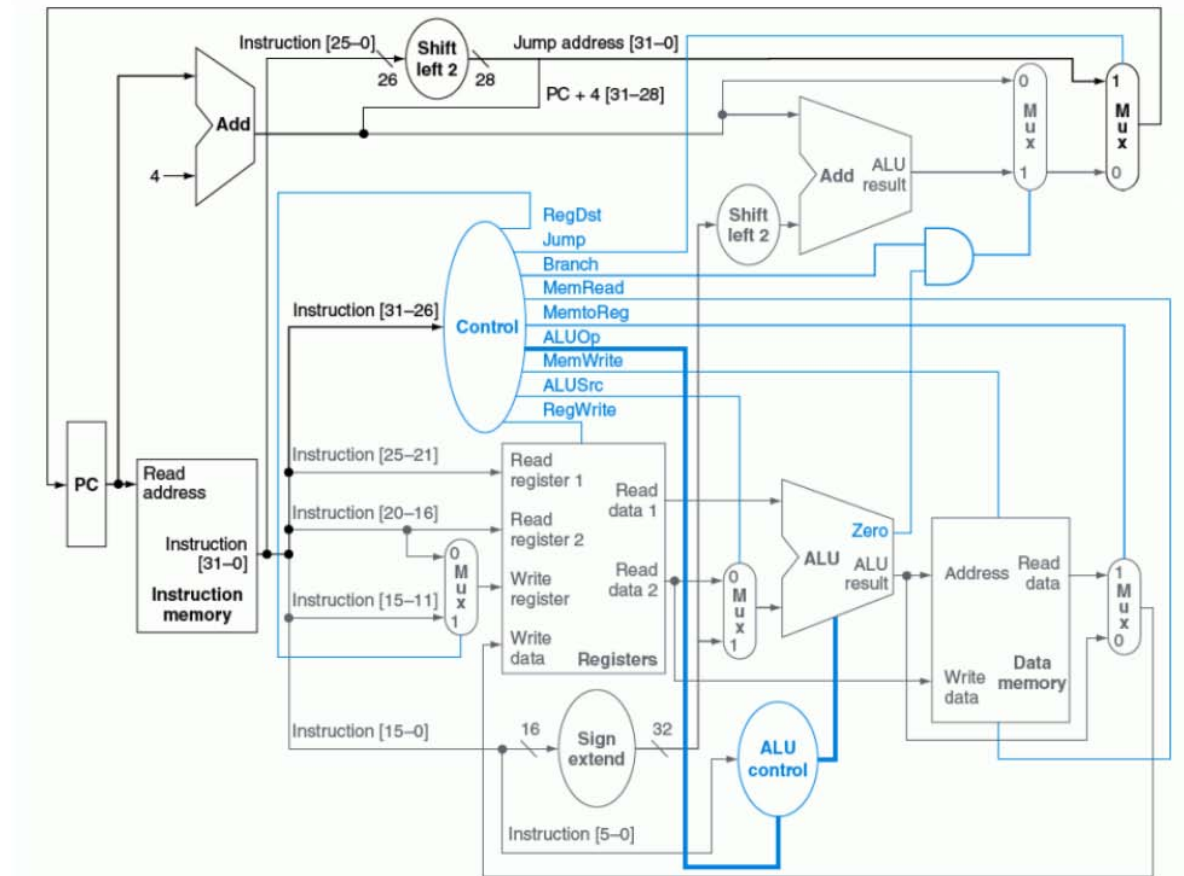


| | | | | |
|--|-----------------|-----------------|-----|---------|
|  上海交通大学 计算机科学与工程系 Dept. of Computer Science & Engineering | 标题 | 文档编号 | 版本 | 页 |
| | 计算机组成实验指导书 LAB1 | CSE-COA-LAB-005 | 0.3 | 5 of 14 |
| | 作者 | 修改日期 | 公开 | |
| | WnSN Lab | 09/21/2010 | | |

3. 顶层模块 Top

3.1 实验描述

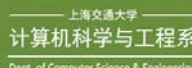
3.1.1 模块描述

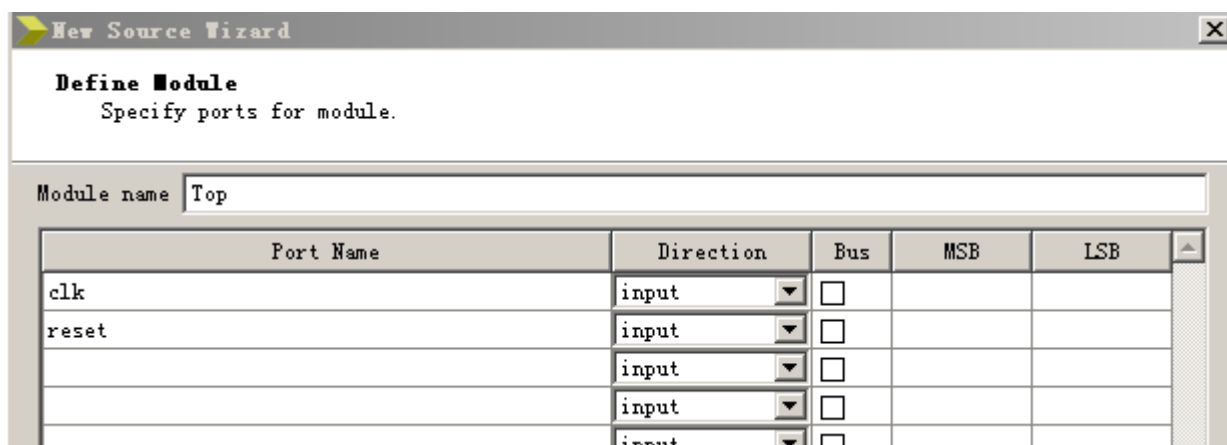


MIPS 单周期处理器原理图

3.1.2 新建模块源文件 Top，将各个模块互联起来。

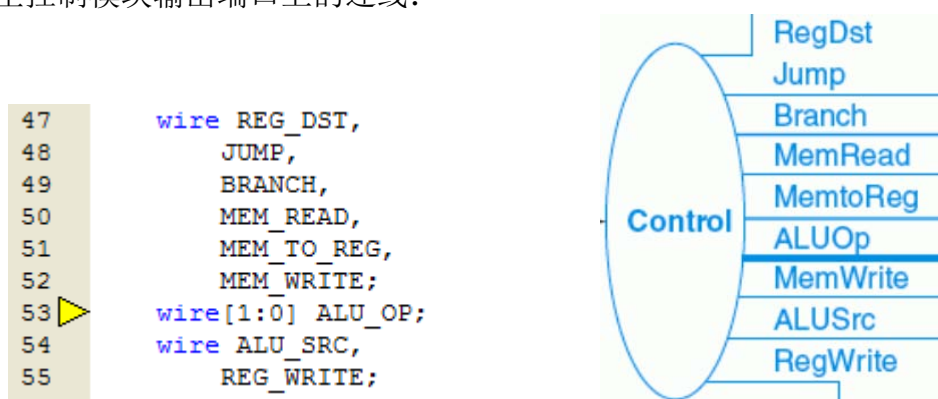
单周期处理器的设计，关键是确定数据通路（信号和数据）以及确定哪些操作需要时钟，哪些不需要时钟，要分析时序并约束。

| | | | | |
|---|-----------------|-----------------|-----|---------|
|  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-005 | 0.3 | 6 of 14 |
| | 作者 | 修改日期 | | |
| | WnSN Lab | 09/21/2010 | 公开 | |



3.1.3 定义信号线

给 top 模块内的每一根连接的信号线命名，并在 top 模块中声明它们。
例如，主控制模块输出端口上的连线：



3.1.4 程序计数器 PC


程序计数器是这个简单 CPU 能够跑起来的关键。定义一个 32 位 reg 类型 PC，在时钟上升沿(下降沿已经被我们用作寄存器的写了)做 $PC \leq PC + 4$ 。

注：简单的讲，在组合逻辑中用阻塞赋值“=”，时序逻辑中用非阻塞赋值“ \leq ”。两者综合出来的电路不一样，具体区别查阅参考书。时序逻辑和组合逻辑不要放在同一个 always 块中。

3.1.5 RESET

PC 置 0x00000000，各寄存器清零，这是 reset 要做的工作。同步或异步，边沿或电平，同学们可以自由实现。

寄存器清零，所以要适当修改上次实验的 registerFile 模块，给模块添加 reset 信号。

| | | | | |
|--|-----------------|-----------------|-----|---------|
|  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-005 | 0.3 | 7 of 14 |
| | 作者 | 修改日期 | 公开 | |
| | WnSN Lab | 09/21/2010 | | |

注：添加 reset 要注意，写在原来“写”的 always 块中。假如新加一个 always 块，当两个“写”always 同时满足时，就将混乱不知赋什么值了。

3.1.6 模块实例化，连接模块

实例化前两次实验中编写的模块，实例化的过程中连接模块的端口。各种变量名比较多复杂，需要定义一套命名规则，方便代码的编写和阅读。实例化有以下两种方法：

1. 严格按照模块定义的端口顺序来连接，不用表明原模块定义时规定的端口名：

模块 模块名(连接端口 1 信号名, 连接端口 2 信号名...)

2. 在连接时用“.”符号，表明原模块是定义时规定的端口名：

模块 模块名(.端口 1 名(信号 1), .端口 2 名(信号 2))

推荐用第 2 种实例化方法。

以主控制模块为例，以下代码实例化一个主控制模块 Ctr，并连接其端口。INST 是定义好的指令存储器输出的连接信号，其他信号线我们在 3.1.3 中已定义。

```
100      Ctr mainCtr(  
101          .opcode(INST[31:26]),  
102          .regDst(REG_DST),  
103          .jump(JUMP),  
104          .branch(BRANCH),  
105          .memRead(MEM_READ),  
106          .memToReg(MEM_TO_REG),  
107          .aLUOp(ALU_OP),  
108          .memWrite(MEM_WRITE),  
109          .aLUSrc(ALU_SRC),  
110          .regWrite(REG_WRITE));  
...
```

实例化 Ctr

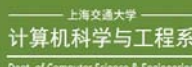
3.1.7 连接其它信号线

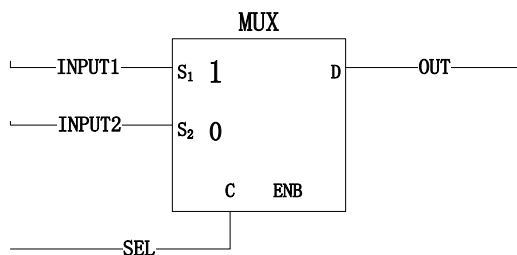
1. MUX

Mux 已经在前几次实验中提到，实现很简单，一个三目运算符

Assign OUT = SEL ? INPUT1 : INPUT2;

OUT, SEL, INPUT1, INPUT2 都是预先定义的信号。

| | | | | |
|---|-----------------|-----------------|-----|---------|
|  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-005 | 0.3 | 8 of 14 |
| | 作者 | 修改日期 | 公开 | |
| | WnSN Lab | 09/21/2010 | | |




2. 左移两位，用移位运算符：左移("<<"), 右移(">>")

3. 加法器，直接用无符号加法运算。

注：verilog 中寄存器类型被解释成无符号数，整数类型(integer)被解释成二进制补码形式的有符号数。因此要综合成无符号算术算符需要使用寄存器类型，而要得到有符号算术算符就需要使用整数。网线类型被解释成无符号数。

4. 与门，使用位运算符&(位与)。注意&和&&的区别。

| | | | | |
|--|-----------------|-----------------|-----|---------|
|  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-005 | 0.3 | 9 of 14 |
| | 作者 | 修改日期 | 公开 | |
| | WnSN Lab | 09/21/2010 | | |

4. 仿真测试

1. 编写二进制测试程序

请编写自己的测试汇编。下面提供一个简易汇编器供参考。

一些相关的基本知识：

指令格式：

| | | | | | | |
|---|--------|---------|-------|-----------|-------|-------|
| R | opcode | rs | rt | rd | shamt | funct |
| | 31 26 | 25 21 | 20 16 | 15 11 | 10 6 | 5 0 |
| I | opcode | rs | rt | immediate | | |
| | 31 26 | 25 21 | 20 16 | 15 | | 0 |
| J | opcode | address | | | | |
| | 31 26 | 25 | | | | 0 |


Mips 基本指令格式

汇编格式：注意汇编中寄存器的顺序跟 Mips 指令格式中的不一样

```
add $1,$2,$3      : $1=$2 + $3
sub $1,$2,$3      : $1=$2 - $3
and $1,$2,$3      : $1=$2 & $3
or $1,$2,$3       : $1=$2 | $3
slt $1,$2,$3      : if($2<$3) $1=1 else $1=0
lw $1,10($2)      : $1=memory[$2+10]
sw $1,10($2)      : memory[$2+10]=$1
beq $1,$2,10       : if($1==$2) goto PC+4+40
                    [10 是 PC+4 后的指令间隔数，故为 PC+4+40]
j 10000           : goto 10000
```

2. 系统任务\$readmemb和\$readmemh。Verilog 中这两个系统任务用来从文件中读取数据到存储器中，代码编写放在存储模块的 initial 初始化块中。格式如下：

```
$readmemx("datafile", memoryName);
$readmemx("datafile", memoryName, startAddr);
```

| | | | | |
|--|-----------------|--------------------|-----|----------|
|  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-005 | 0.3 | 10 of 14 |
| | 作者 WnSN Lab | 修改日期 09/21/2010 | 公开 | |

```
$readmemx("datafile", memoryName, startAddr, endAddr);
```

这里给出 mem_data 和 mem_inst 两个样例文件，分别用来装入 data memory 和 instruction memory。

mem_data

```
1 00000000
2 00000001
3 00000002
4 00000003
5 00000004
6 00000005
7 00000006
8 00000007
9 00000008
```

... ..

mem_inst

```
1 00001000000000000000000000000000 // j a
2 00000000000000000000000000000000 // nop
3 00000000000000000000000000000000 // nop
4 00000000000000000000000000000000 // nop
5 10001100000000010000000100010100 // lw $1, 276($0)
6 10001100000000010000000010000100 // lw $2, 264($0)
7 00000000001000100001100000100000 // add $3, $1, $2
8 0000000000100010001000000100010 // sub $4, $1, $2
```

... ..

3. 其他常用系统任务

\$monitor 提供监控和输出参数列表中的表达式或变量值的功能。

如\$monitor("rxd=%b", rxb);

\$time 返回当前仿真时刻值。

如\$monitor(\$time);

4. 编辑 testbench 文件，进行仿真测试：

- 1) 初始化 register、data memory 和 instruction memory 三大存储模块，这里仅以初始化 data memory 为例，其它类推。该 memory 用于存

| | | | | |
|--|-----------------|-----------------|-----|----------|
| 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-005 | 0.3 | 11 of 14 |
| | 作者 | 修改日期 | 公开 | |
| | WnSN Lab | 09/21/2010 | | |

储二进制代码。如下图，Verilog 中调用了系统任务\$readmemh 将 mem_data 文件中的数据读入到 memFile 数组中。

```

28     );
29
30     reg [31:0] memFile [0:255]; //memory space: 256*32bits
31
32     //initial the instruction or data memory
33     initial
34     begin
35         $readmemh("./Src/mem_data",memFile,10'h0);
36     end
37
38
39     always @ (negedge clock_in)
40     begin
    
```

也可把对存储模块的初始化一起添加到 top 中的 initial 块里，如下：

```

initial
begin
    $readmemh("./Src/regist",regFile,32'h0);
    $readmemh("./Src/mem_data", memFile, 10'h0);
    $readmemb("./Src/mem_inst", InstMemFile, 8'h0);
end
    
```

- 2) 编写 Top 层的 testbench，可定义文件名为 Top_tb。
- 3) 按需要添加时钟激励和其他输入信号的初始化。

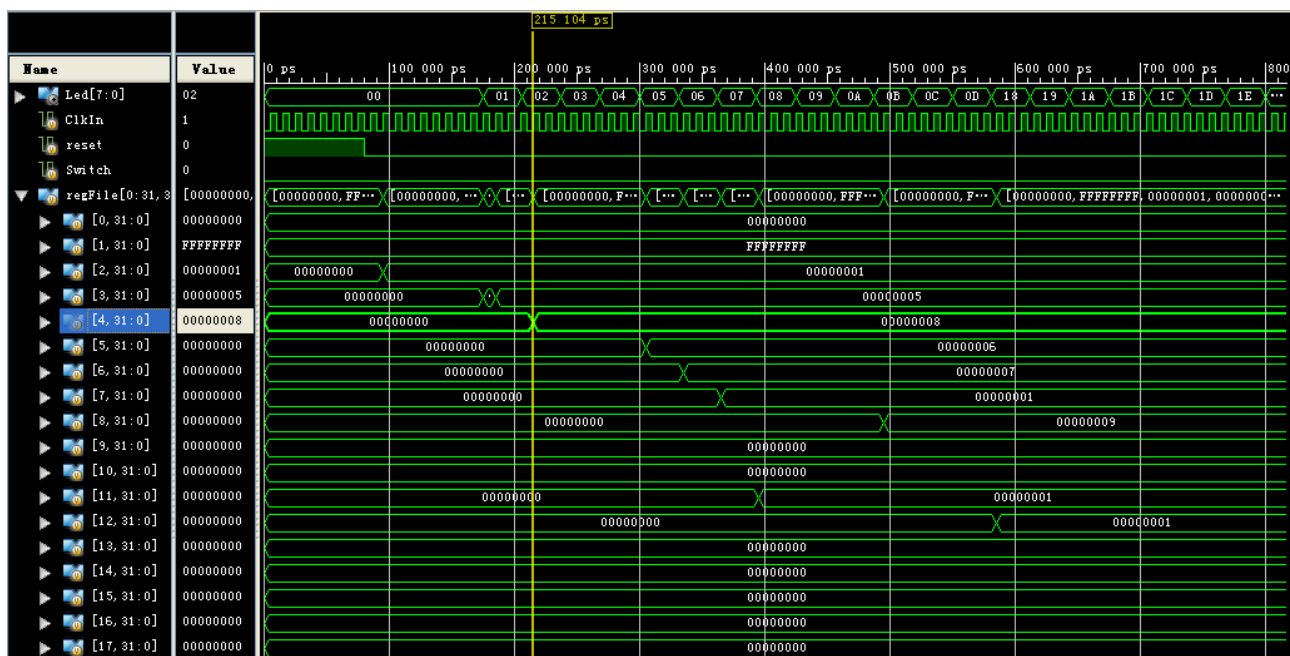
| | | | | |
|--|-----------------|-----------------|-----|----------|
| <div> <div>上海交通大学</div> <div>计算机科学与工程系</div> <div>Dept. of Computer Science & Engineering</div> </div> | 标题 | 文档编号 | 版本 | 页 |
| | 计算机组成实验指导书 LAB1 | CSE-COA-LAB-005 | 0.3 | 12 of 14 |
| | 作者 | 修改日期 | 公开 | |
| | WnSN Lab | 09/21/2010 | | |

```

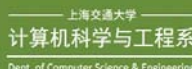
35 // Instantiate the Unit Under Test (UUT)
36 Top uut (
37     .ClkIn(ClkIn),
38     .Rst(Rst),
39     .PC(PC),
40     .Led(Led)
41 );
42
43 initial begin
44     // Initialize Inputs
45     ClkIn = 0;
46     Rst = 1;
47     Switch = 0;
48
49     // Wait 100 ns for global reset to finish
50     #100;
51     Rst = 0;
52
53     // Add stimulus here
54
55 end
56 always
57     #2 clock_in = !clock_in;
58
59 endmodule

```

- 4) 添加 register 模块中的 regfile 寄存器数组到仿真波形窗口，观察各个寄存器的变化情况。
- 5) 在 Console 窗口中输入 restart 和 run 2000ns 命令，重新进行仿真。观察新波形并与你设计的 MIPS 指令运行结果相对比。



仿真波形示例


| | | | | |
|--|-----------------|--------------------|-----|----------|
|  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-005 | 0.3 | 13 of 14 |
| | 作者 WnSN Lab | 修改日期 09/21/2010 | 公开 | |

5. 下载验证

本节上板调试验证，根据自己的情况可选做
注：

1. Memory 没有好的办法去初始化，只有给 memory 添加 reset，在 reset 的时候硬代码写在里面。
2. 没有充足的外设去观察程序运行情况，5 个 pushbutton，4 个 switch，8 个 led。动用你们的思考和愿望...

6. 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-005 | 0.3 | 14 of 14 |
| | 作者 | 修改日期 | 公开 | |
| | WnSN Lab | 09/21/2010 | | |