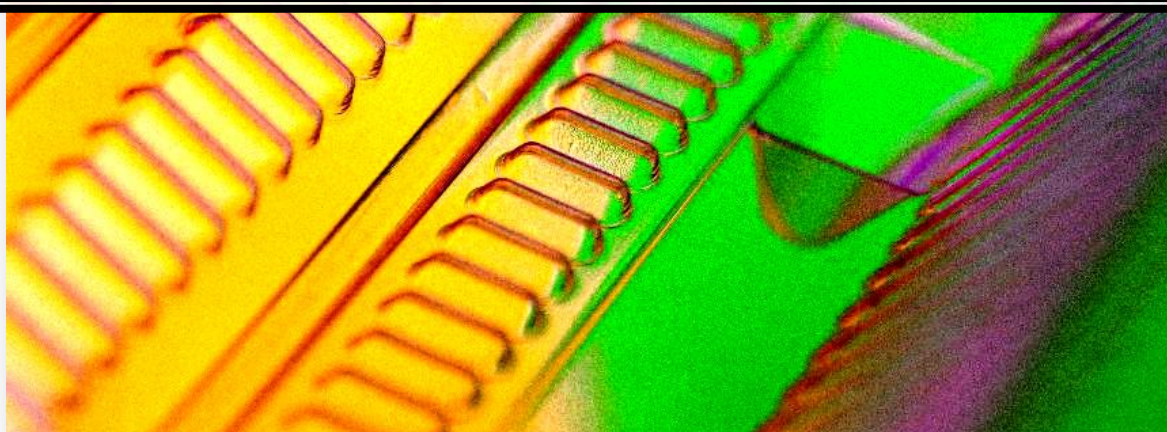


zrtech

**FPGA/CPLD 开发套件实验教程**

**--仿真，调试，设计篇**



**WWW.ZR-TECH.COM**

## 实验四、调试利器 Debussy

### 实验目的：

本节给大家介绍 Debussy 的使用方法，使用户掌握采用 Debussy 对设计进行调试排错的方法。

### 实验原理：

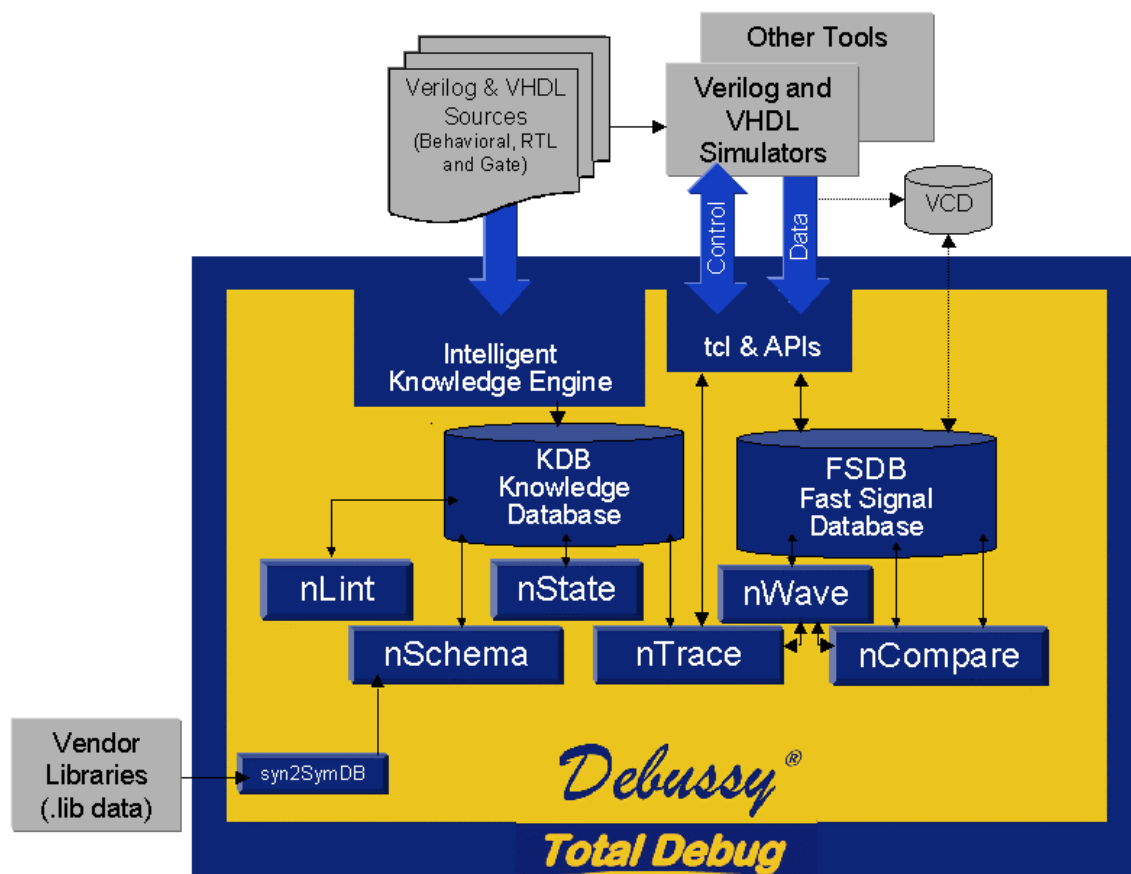
#### 1. Debussy简介

Debussy 是 NOVAS Software, Inc(思源科技)发展的 HDL Debug & Analysis tool，其主要不是用来跑模拟或看波形，它最强大的功能是：能够在 HDL source code、schematic diagram、waveform、state bubble diagram 之间，即时做 trace，协助工程师 debug。可能大家会觉得：只要有 simulator 如 ModelSim 就可以做 debug 了，我何必再学这个软件呢？大家别急，学完这课后你自然就知道了答案。

下图所示为整个 Debussy 的原理架构，可归纳几个结论：

Debussy 有五个主要单元(component)

- nTrace：超文本连接方式的源代码追踪及分析
- nSchema：原理图显示及分析
- nWave：波形显示及分析
- nState：有限状态机的显示及分析
- nCompare：分析仿真结果，比较其相异处



Debussy 本身不含模拟器(simulator), 必须呼叫外部模拟器(如 Verilog-XL or ModelSim)产生 FSDB file, 其显示波形的单元 "nWave"透过读取 FSDB file, 才能显示波形或讯号值的变化。Debussy v5.0 以后的新版本, 还提供了 nLint -- check coding style. & synthesizable, 这蛮有用的, 可以协助工程师了解如何写好 coding style, 并养成习惯。

## 实验结果：

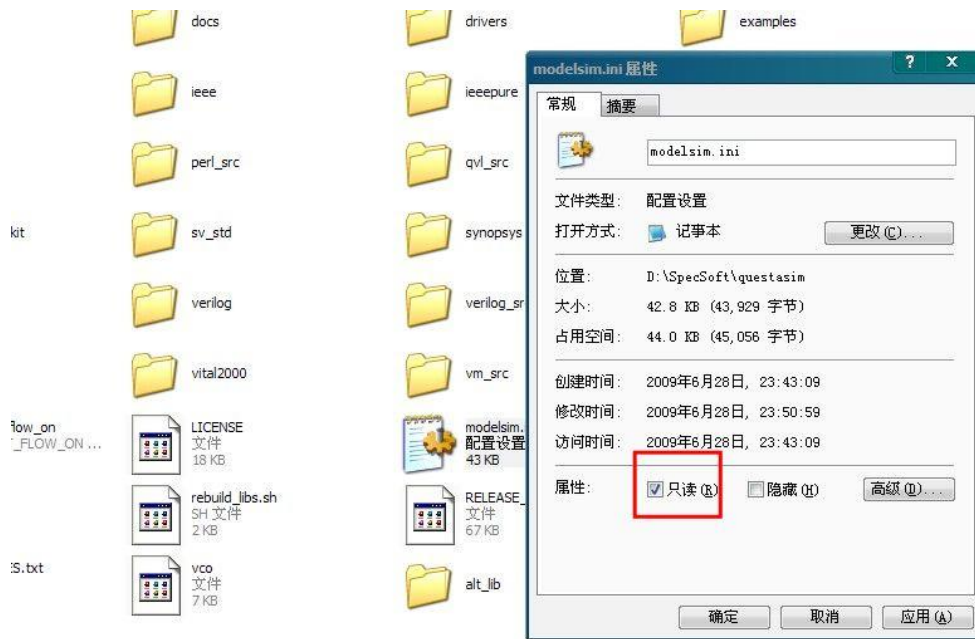
利用 modelsim 对计数器的 Verilog 测试程序功能仿真，并将仿真结果打印在屏幕上，并存储到文件。

## 具体步骤：

### 1.Modelsim的设置

1. 将 debussy 安装目录下的\share\PLI\modelsim\_pli\WINNT 中的 novas.dll 拷贝到 modelsim 安装目录下的 win32 文件夹中。然后再 modelsim.ini 中的[vsim]标签下添加 Veriuser = novas.dll。如下图所示。

注意：在更改 modelsim.ini 前先将属性的“只读”去除。



[vsim]

```
; vopt flow
; Set to turn on automatic optimization of a design.
; Default is on
VoptFlow = 0
VoptFlow = 0
Veriuser = novas.dll

; vopt automatic SDF
; If automatic design optimization is on, enables automatic compilation
; of SDF files.
; Default is on, uncomment to turn off.
; VoptAutoSDFCompile = 0
```

2.建工程，加入源程序和测试程序，注意在测试代码中加入以下语句：

initial

begin

\$fsdbDumpfile("\*\*\*.fsdb");

\$fsdbDumpvars;

end

\*\*\*.fsdb 中\*\*\*可以是任意合法的字母数字组合。如：wave\_test.fsdb。如下图所示：

```
always #10 clk=~clk;

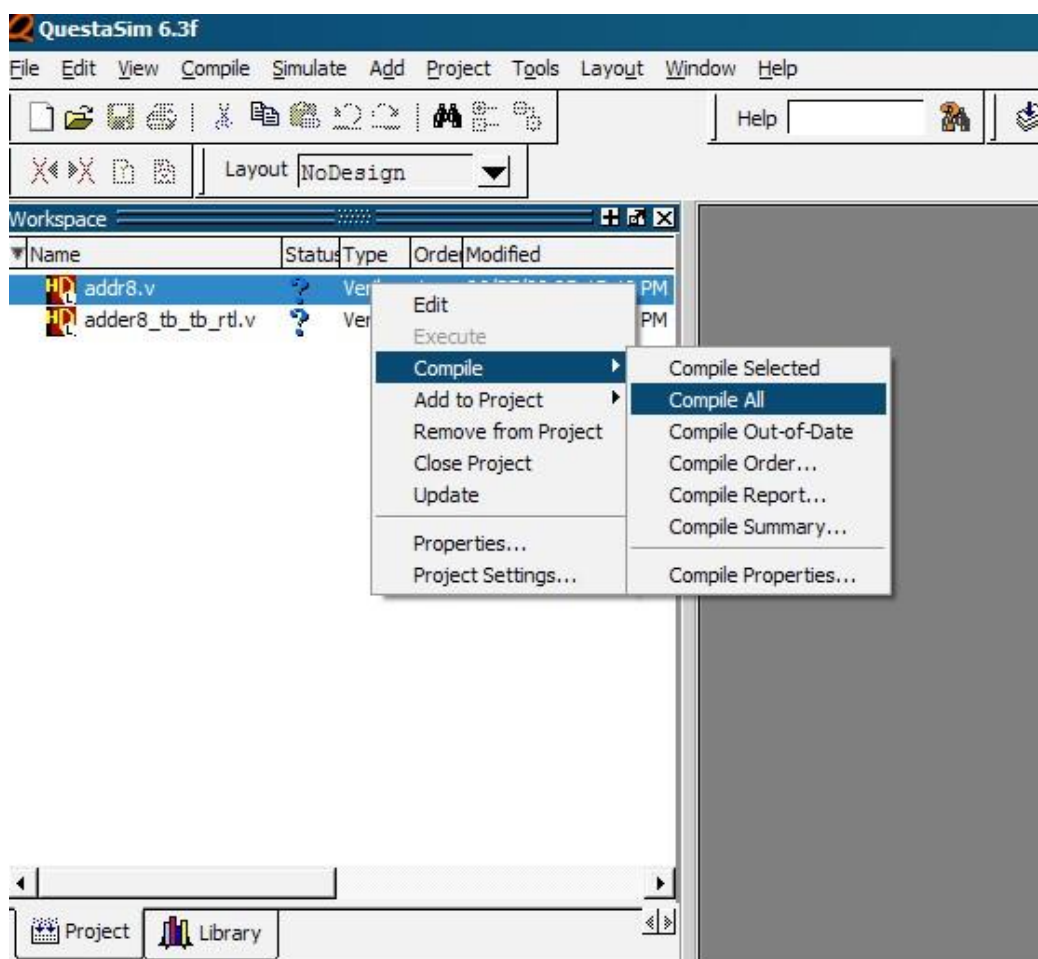
initial
begin

    $fsdbDumpfile("wave_test.fsdb");
    $fsdbDumpvars;

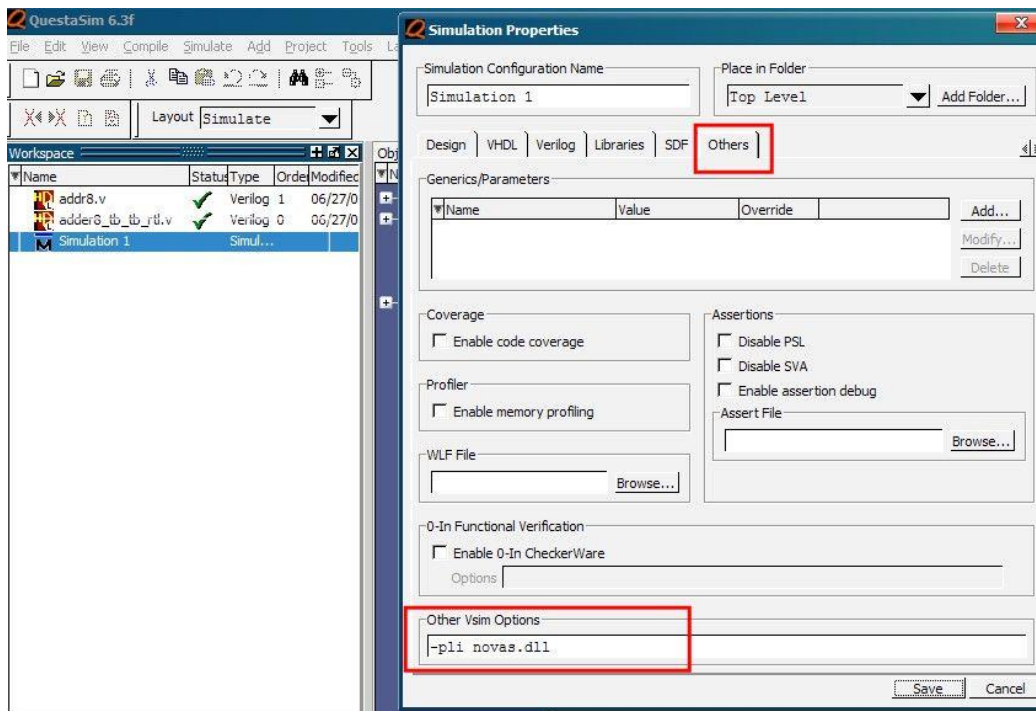
end

initial
```

编译源代码和测试代码，在 Project 标签中添加 Add to Project->Simulation Configuration..出现下面的对话框：



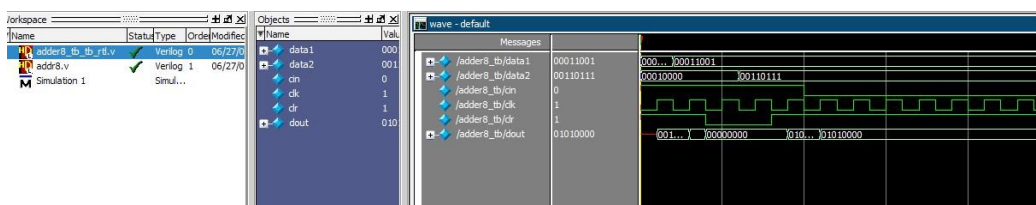




在 others 标签下的 Other Vsim Options 标签下加入 "-pli novas.dll" 。

### 3. 仿真程序

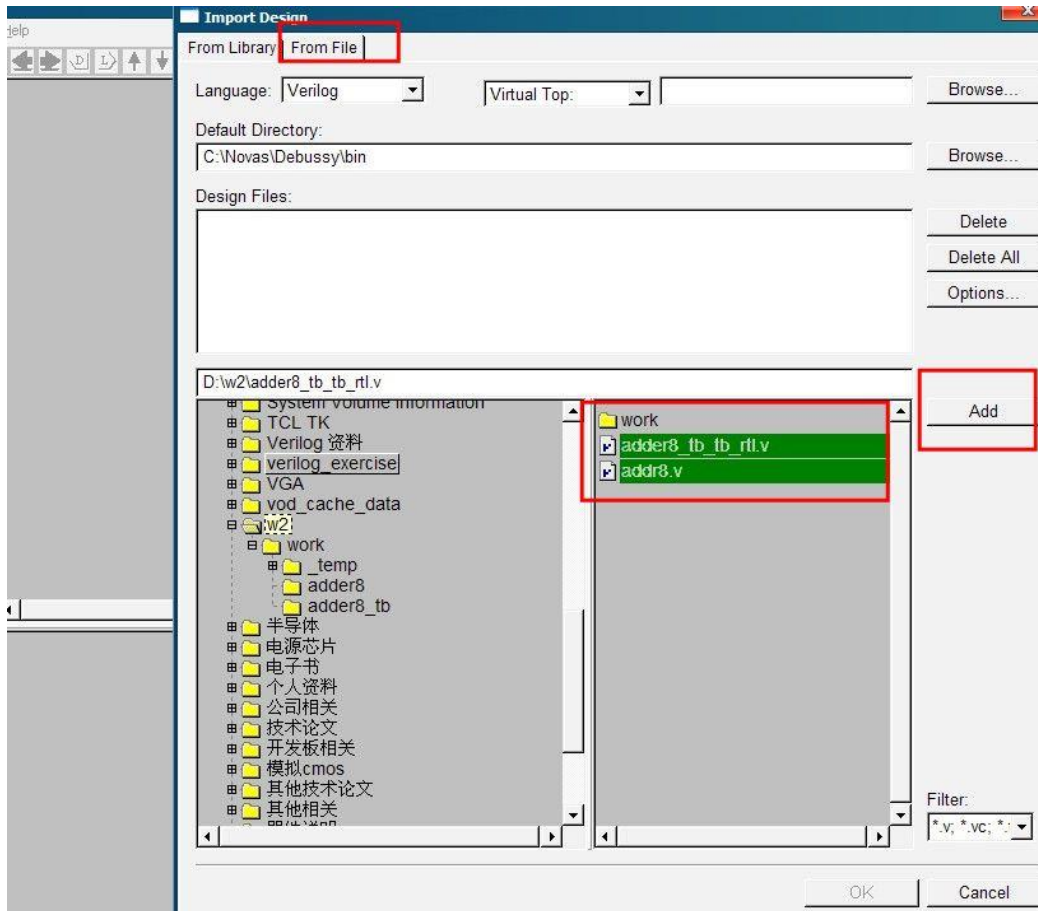
按照一般的仿真步骤，加入波形，完成仿真。



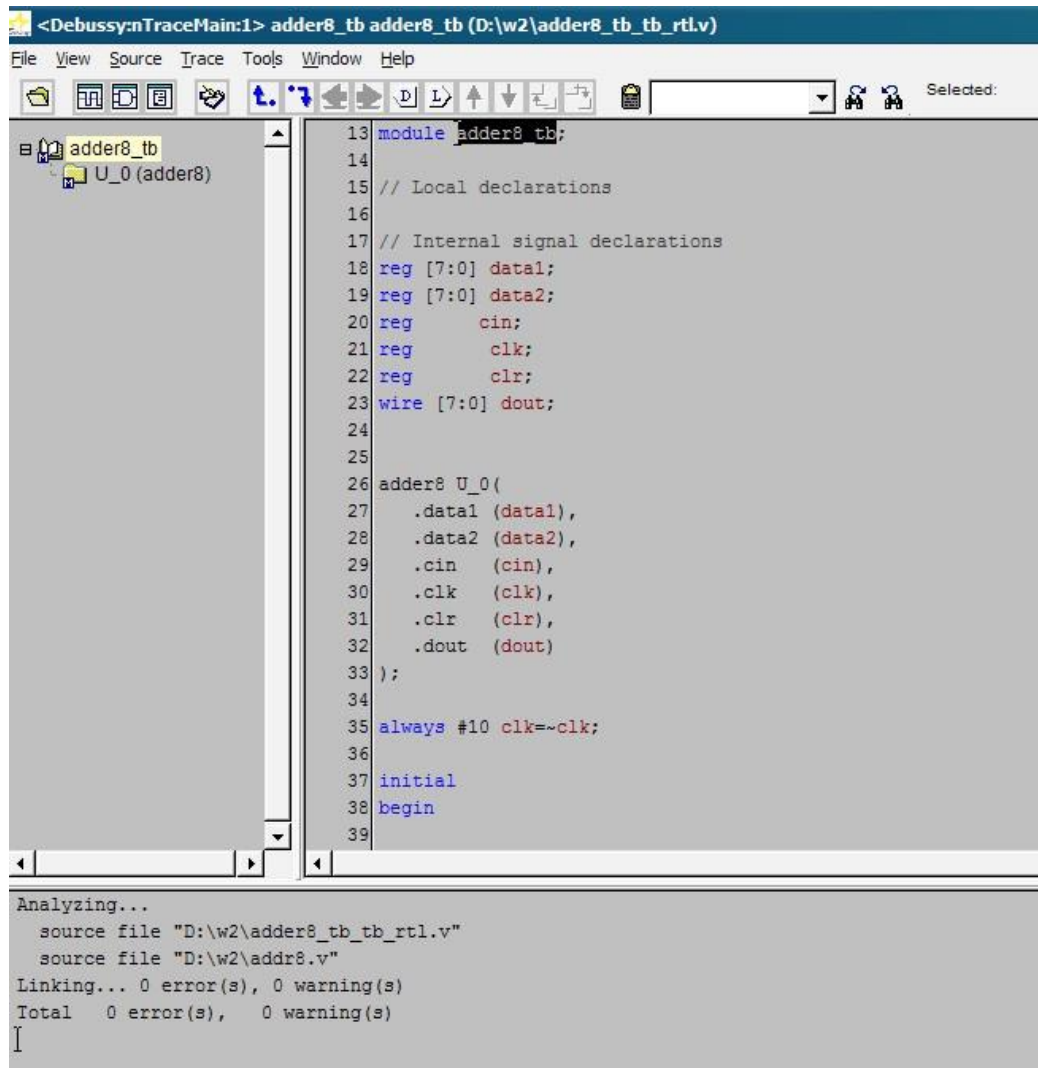
仿真完成后，会在工程目录下面生成一个 fsdb 的文件。

### 3. Debussy 使用

(1) 开启界面，File→import design 加入源代码，选择 “From File” 如下图所示：




导入成功后可以看到以下的主界面，nTrace 视窗中，含有三个区域，Hierarchical Brower、Source code window、Message window。：



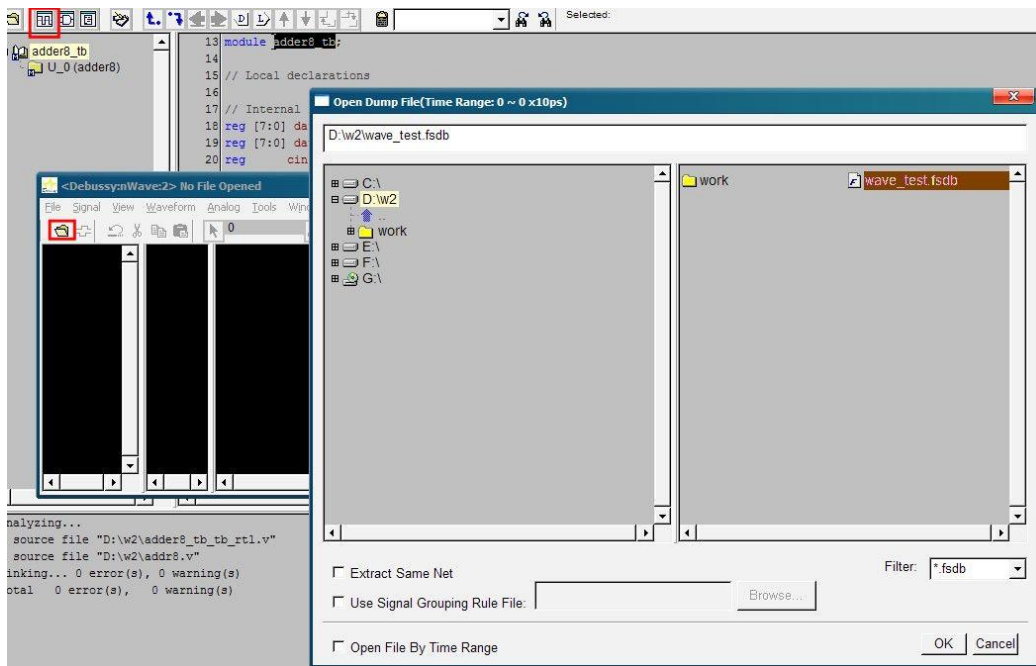
双击左侧目录下得文件名可以轻易地追踪出 project 内的所有 design 彼此之间的关联性。

除了追踪 designs 之间的关联性，也可以用同样的方法追踪出 signal's drivers and loads。

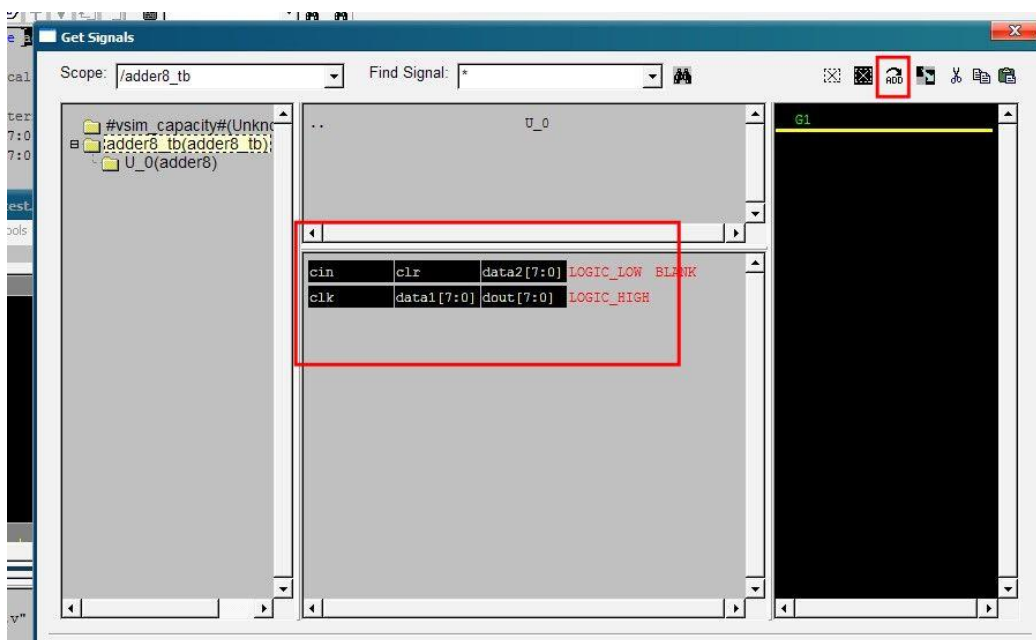
(2) 导入 fsdb 文件。

点击  以产生一个新的波形，界面开启后，点击 file—》open，导入 fsdb。界面如下图所示：

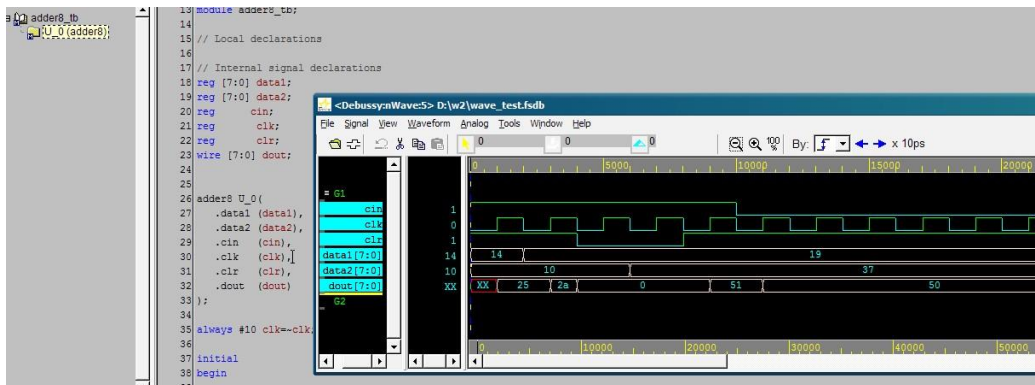




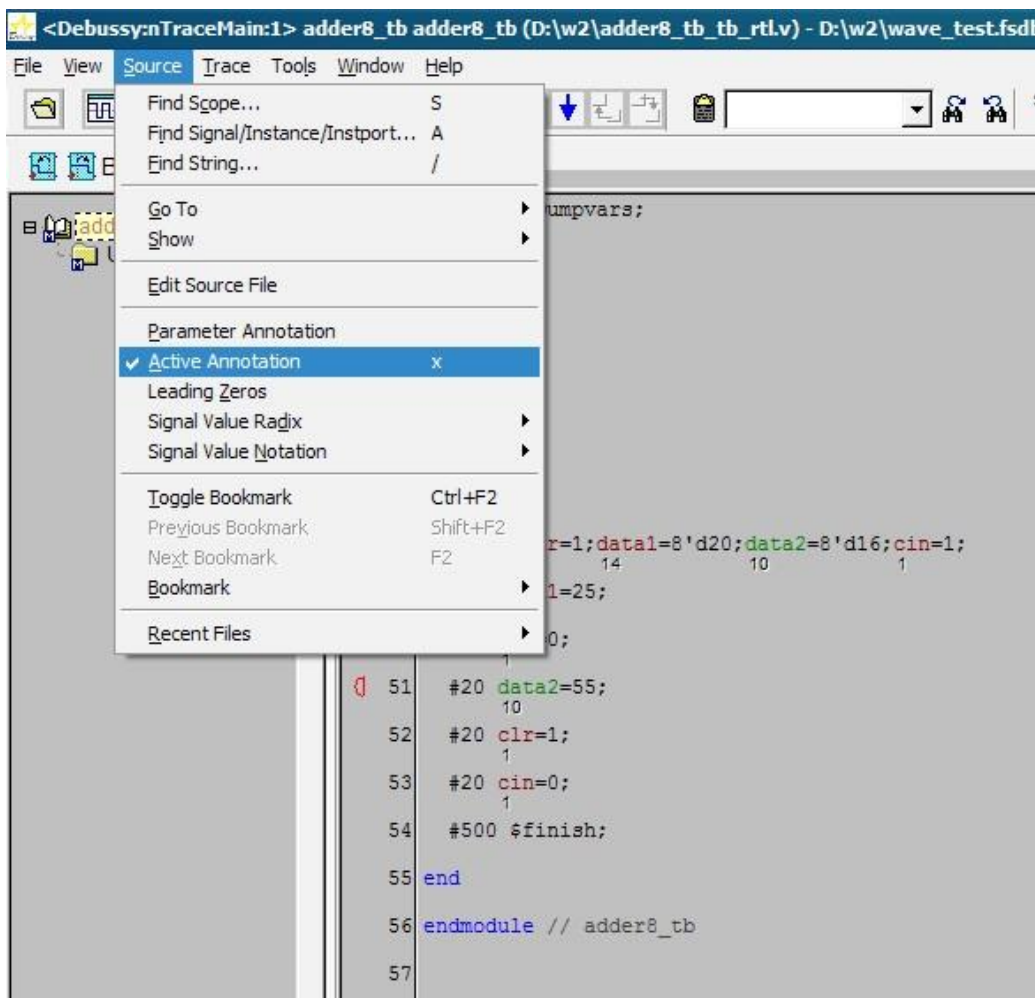
导入 fsdb 成功后，接著按 Get Signals icon，此时就会看到有信号可以让你选择了，选择后，按右上侧的那个“ADD”按钮即可加入信号。如果你看不到信号，把想观察的 design，直接从(nTrace) Hierarchy browser 拖进 nWave 就可以。



使用(nWave) View \ Zoom \ Zoom All 可以查看所有的波形了：

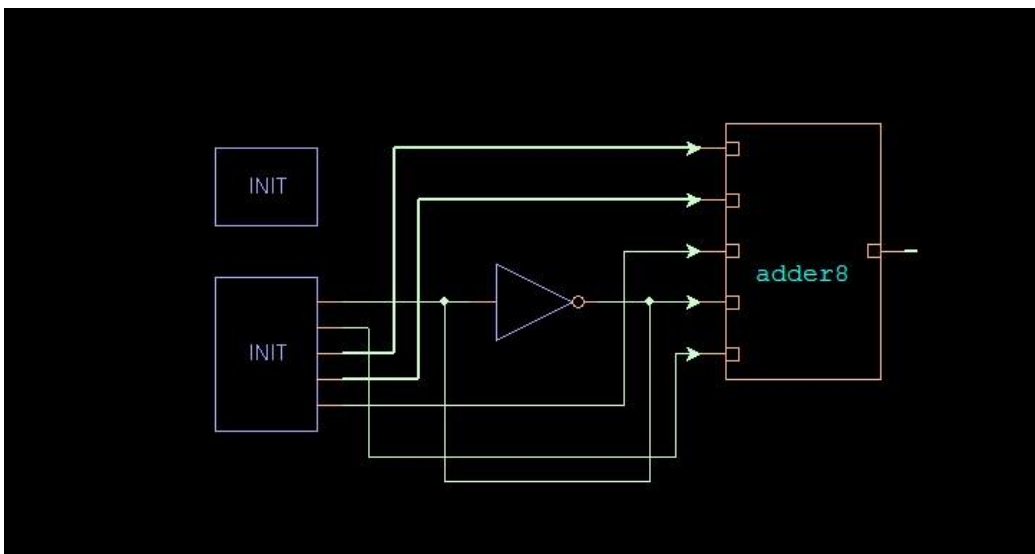


(3) 启动 Active Annotation 功能：



能够在 nWave 选择讯号触发缘, 同时在 nTrace 的 source code 的所有讯号符号下方, 直接看到数值的变换。在 nWave 双击想观察其触发状况的信号, 此时就会跳转到 test bench 中。摸索几次就会发现其中的奥妙之处。

(4) 开启 nSchema, 双击上图内的元件符号可进一步看到其内容/ 细部电路组成。在 hierarchy browser、source code 与 schematic window 之间, 可以用滑鼠中键互相拖曳 designs/signals。



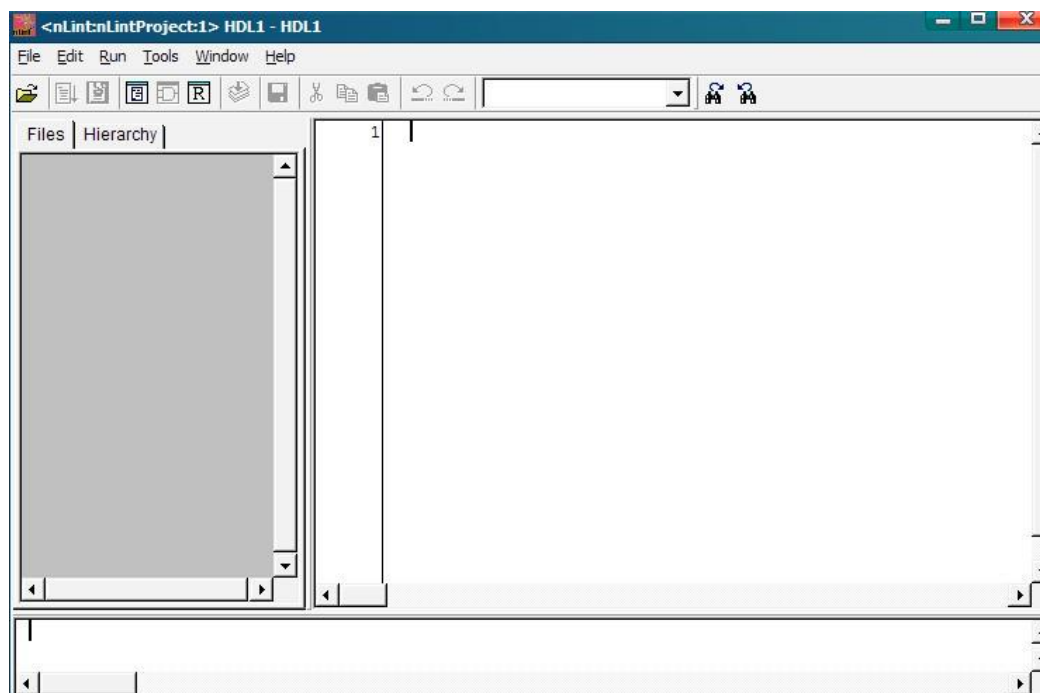
#### 4. nLint 的使用

##### (1) 开启图形界面的 nLint

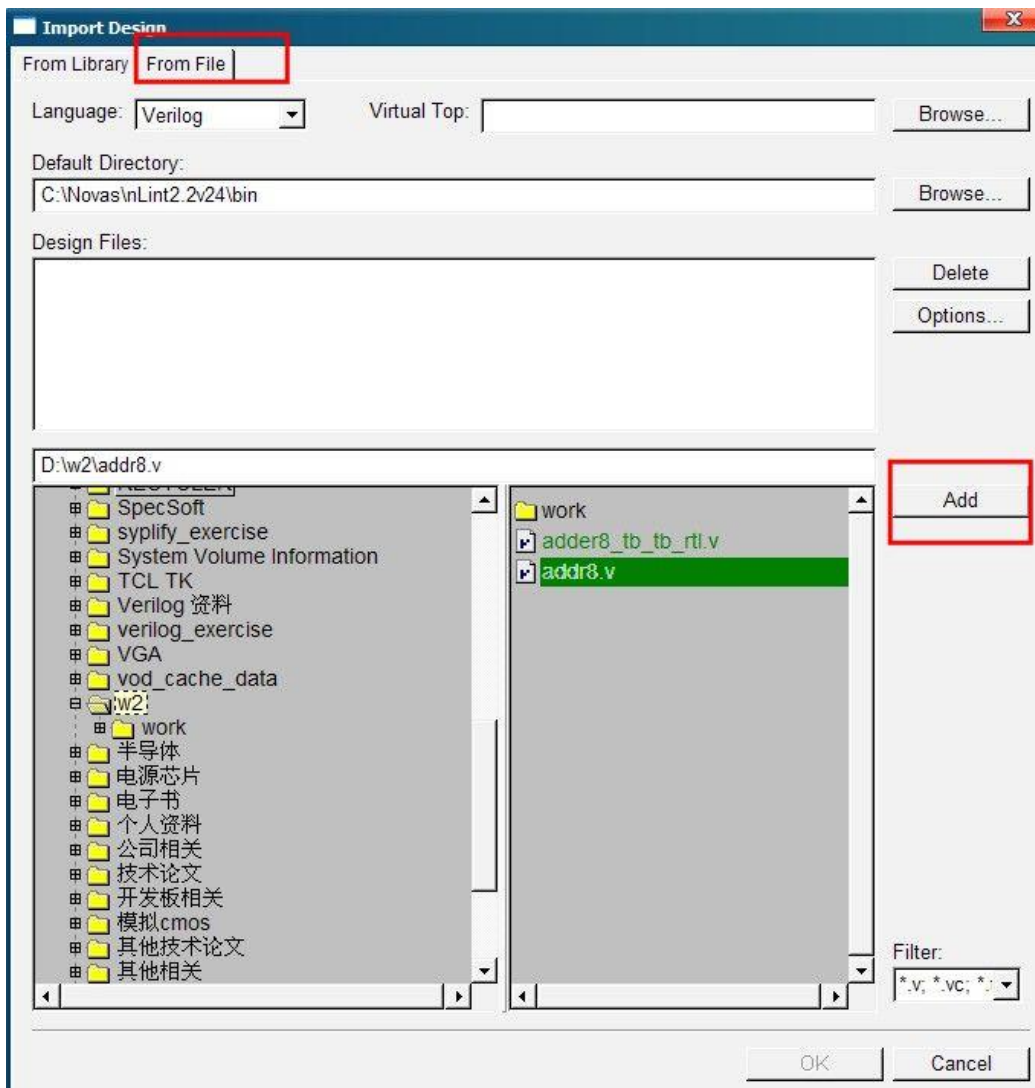
nLint 的图形界面必须使用脚本才能唤醒。如下图, 在安装目录的 bin 文件下建立一个 .bat 文件。输入: "nlint -gui", 双引号不需要输入。注意 "nlint" 和 "-" 之间有个空格。



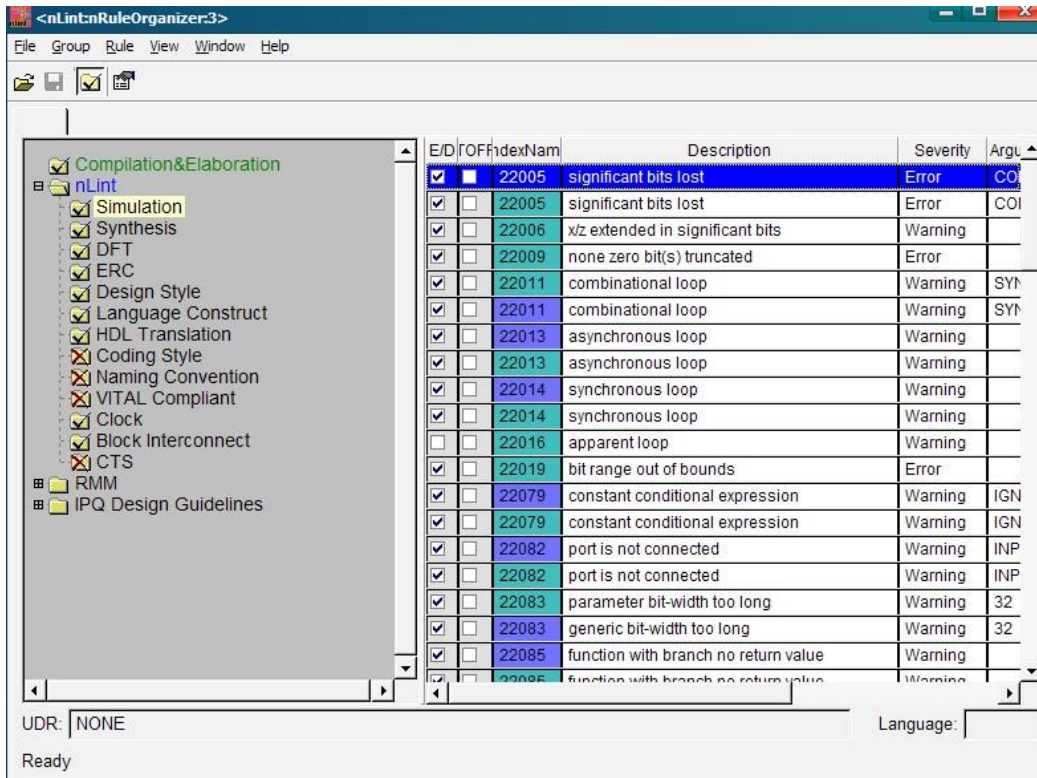
双击 bat 文件后，出现 nlint 的界面。



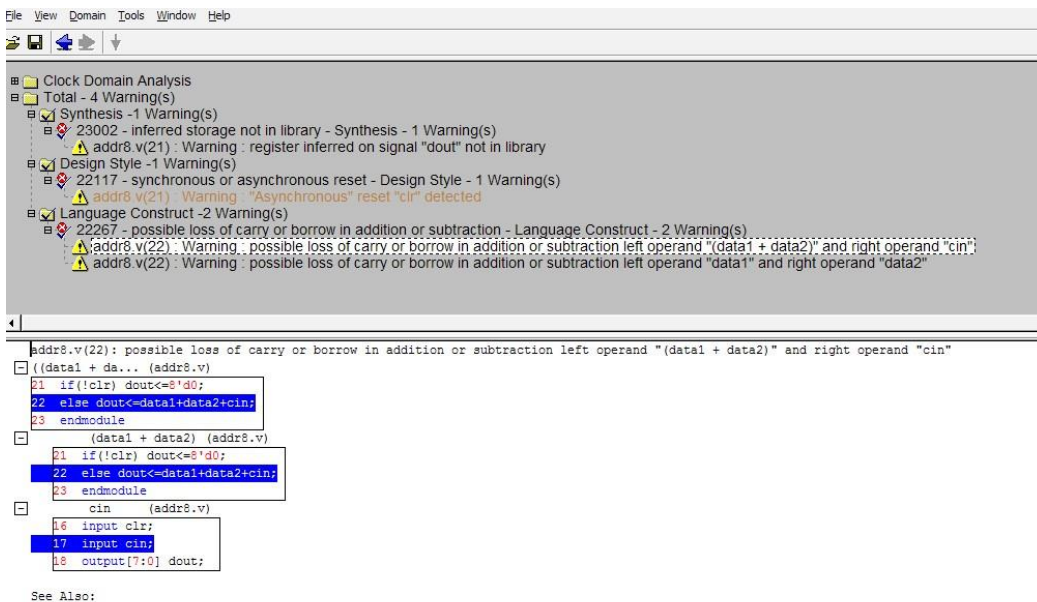
使用 debussy 相似的文件导入方式导入需要检查的源文件。



文件导入后执行 Run\Compile，接着执行 Tools\ Rule Organizer，或按选择 check 哪些 rules 不想检查的项目，就把 E/D 那一栏的核选框取消。



执行 Run \ Lint，会自动跳出 ReportViwer 显示检查结果。



展开 report，选择第一个黄色警示标示，其出错的原因与原始码部份，会即时显示在下方的栏框中。如果要修改 source code，



在该错误选项上, 按滑鼠右键, 选择 Show violation \ to default editor。

## 1、先在 Quartus II 里生成一个例子程序

具体步骤略, 可参考 Quartus 的具体步骤, 例程代码见 counter..v。

```
module counter (clk, reset, enable, count);  
  
    input clk, reset, enable;  
    output [3:0] count;  
    reg [3:0] count;  
  
    always @ (posedge clk)  
    if (reset == 1'b1)  
        count <= 0;  
    else if (enable == 1'b1)  
        count <= count + 1;  
    end  
endmodule
```

## 2 . 编写对应的testbench文件

```
module counter_test;
reg clk, reset, enable;
wire [3:0] count;
integer file_id;
out_file = $fopen ( " counter.data " );

counter counter _t(
.clk (clk),
.reset (reset),
.enable (enable),
.count (count)
);

initial begin
    clk = 0;
    reset = 0;
    enable = 0;
end

always
    #5    clk = ! clk;

initial    begin
    $display("\t\ttime,\tclk,\treset,\tenable,\tcount");
    $fdisplay(file_id, "\t\ttime,\tclk,\treset,\tenable,\tcount");
    $monitor("%0d,\t%0b,\t%0b,\t%0b,\t%0d", $time, clk, reset, enable, count);
    $fmonitor(file_id, "%0d,\t%0b,\t%0b,\t%0b,\t%0d", $time, clk, reset, enable, count);
end

initial
    #100    $finish;
endmodule
```

从以上测试代码可以看出, 我们不但对计数器进行了测试, 而且将测试的结果打印在显示器, 且存在了文件" counter.data " 中

### 3. 打开modelsim进行功能仿真

ModelSim 流程, 我们再复习一下:

- 直接选择工程目录下的源文件与测试文件进行编译
- 启动仿真器, 指定顶层设计单元
- 添加待观察信号

- 查看和调试结果

这些步骤和上一节完全相同, 我们就不再赘述了。

## 实验总结：

编写 Testbench 的目的是对硬件描述语言写的电路进行仿真, 从而验证电路的功能与性能是否与预期的设计相符。对于复杂时序电路的仿真, modelsim 加测试激励要比采用 vwf 波形仿真高效很多。testbench 刚开始写起来估计会有些不太习惯, 但是多写写就会熟练很多了。如果有的朋友实在不想编写测试激励, 却想体验一下 modelsim 的强大功能, 也可以在 quartus 中将 vwf 转换成对应的测试激励文件。

## 课后作业：

将本节实验继续完成下去, 实现综合后仿真与时序仿真。观察实验结果。

## 文档内部编号: FEC1001T03

编号说明:

首一字母: F-FPGA系列

首二字母: L-理论类 E-实验类 T-专题类

首三字母: C-普及类 Q-逻辑类 S-软核类

数字前两位: 代表年度

数字后两位: 同类文档顺序编号

尾字母/数字: C目录, T正文, 数字表示章节号

## 修订记录

版本号	日期	描述	修改人
1.0	11.1.2	初稿完成	左超