



Model*Sim*详解

李永

西安交大SOC设计中心

2004.8

Model*Sim* 仿真工具

- 由Model技术公司开发
- 工业上最通用的仿真器之一
- 可在Verilog 和 VHDL仿真
 - OEM版本允许Verilog仿真 或者 VHDL 仿真



Model Technology
A MENTOR GRAPHICS COMPANY

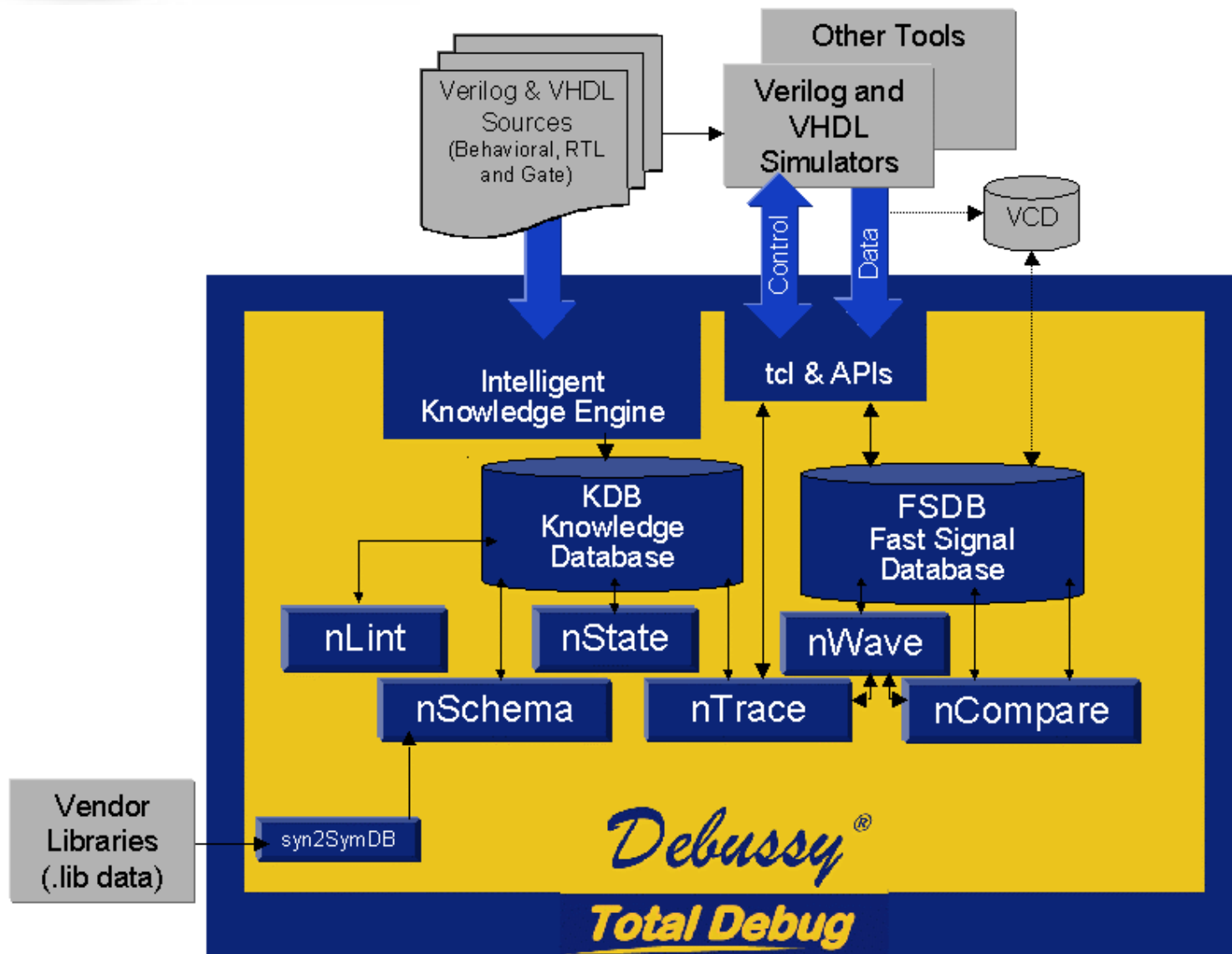
ModelSim 产品

- ModelSim/VHDL 或者 ModelSim/Verilog
 - OEM
- ModelSim/LNL
 - 许可 Verilog 或者 VHDL , 但是不同时许可
- ModelSim/PLUS
 - 设计者能立刻混合仿真Verilog 和 VHDL
- ModelSim/SE
 - 首要的版本
 - PLUS的所有功能连同附加功能



Debussy是一个Verilog\VHDL调试工具

- Debussy是一套很好的Verilog\VHDL调试工具，可以帮助设计者快速理解复杂的设计（设计小组中别人开发的复杂、不熟悉的代码或者IP），查找和定位设计中存在的bug，提高效率，缩短产品上市时间。
- Debussy主要有以下几个模块：
 - 1、nTrace：超文本连接方式的源代码追踪及分析
 - 2、nSchema：原理图显示及分析
 - 3、nWave：波形显示及分析
 - 4、nState：有限状态机的显示及分析
 - 5、nCompare：分析仿真结果，比较其相异处



课程安排

- Modesim GUI的基本仿真流程
- Modelsim使用中一些问题讨论
 - 1)命令行模式和批处理模式
 - 2)建库
 - 3)vcd 格式文件输出
 - 4)时序仿真
 - 5)PLI函数与debussy支持
- Debussy 工具介绍

Model 技术公司的 ModelSim

main主窗口:

structure结构窗口

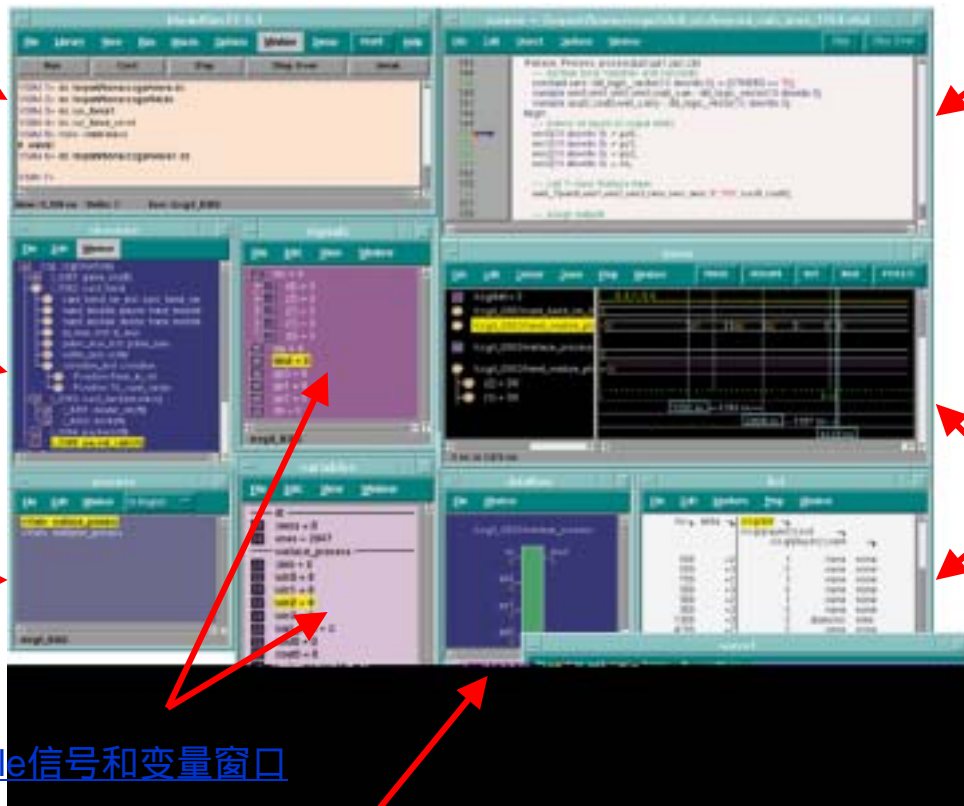
process处理窗口:

Signal&variable信号和变量窗口

dataflow数据流窗口

source源窗口

Wave&list
波形和列表窗口



仿真的目的与分类

- 仿真的目的
在软件环境下，验证电路的行为和设想中的是否一致
- 仿真的分类：
 - a) 功能仿真：在RTL层进行的仿真，其特点是不考虑构成电路的逻辑和门的时间延迟，着重考虑电路在理想环境下的行为和设计构想的一致性；
 - b) 时序仿真：又称为后仿真，是在电路已经映射到特定的工艺环境后，将电路的路径延迟和门延迟考虑进对电路行为的影响后，来比较电路的行为是否还能够一定条件下满足设计构想。

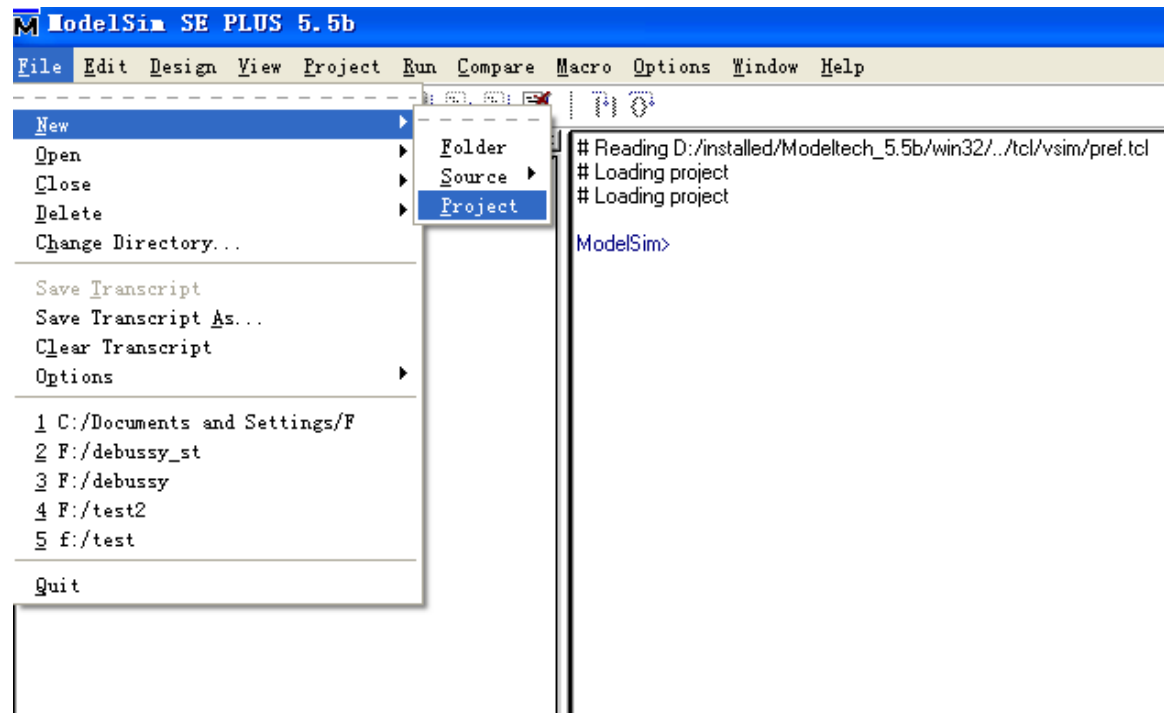
Model *Sim* 实现方法

- 交互式的命令行 (Cmd)
 - 唯一的界面是控制台的命令行, 没有用户界面
- 图形用户界面 (GUI)
 - 能接受菜单输入和命令行输入
- 批处理模式
 - 从DOS或UNIX命令行运行批处理文件

GUI 基本仿真步骤

- 1 ⇒ 建立工程（同时指定了工作库）
- 2 ⇒ 编译源代码
- 3 ⇒ 启动仿真器
- 4 ⇒ 加入波形
- 5 ⇒ 执行仿真

1 ⇒ 建立ModelSim工程

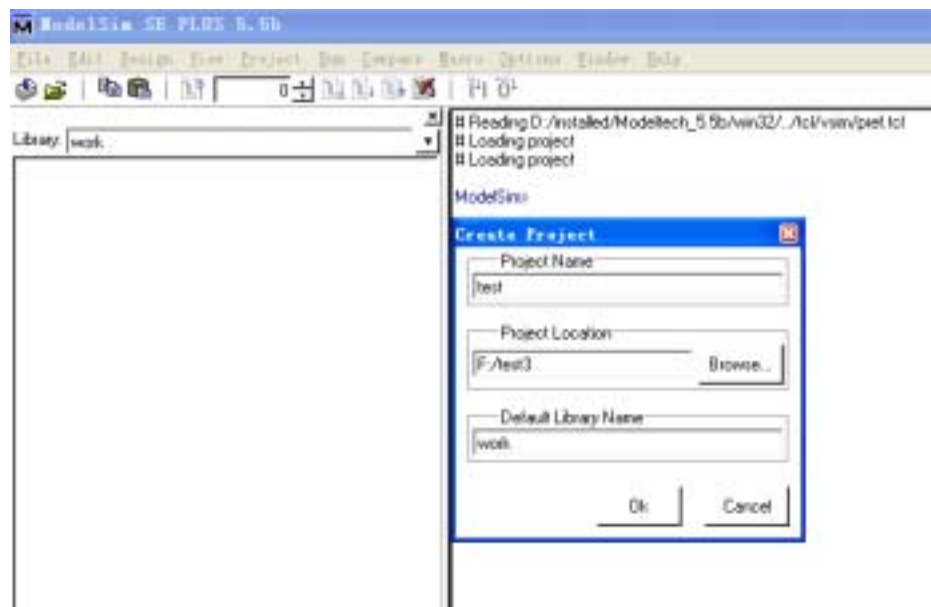


GUI) 从主菜单里面:

file -> new -> project

指定ModelSim 工作库

- 1) 指定工程名
- 2) 指定工程目录
- 3) 指定工作库



库的两个类型

Working (缺省值 **work**)

包含当前被编译的设计单元

编译前必须建立一个working库

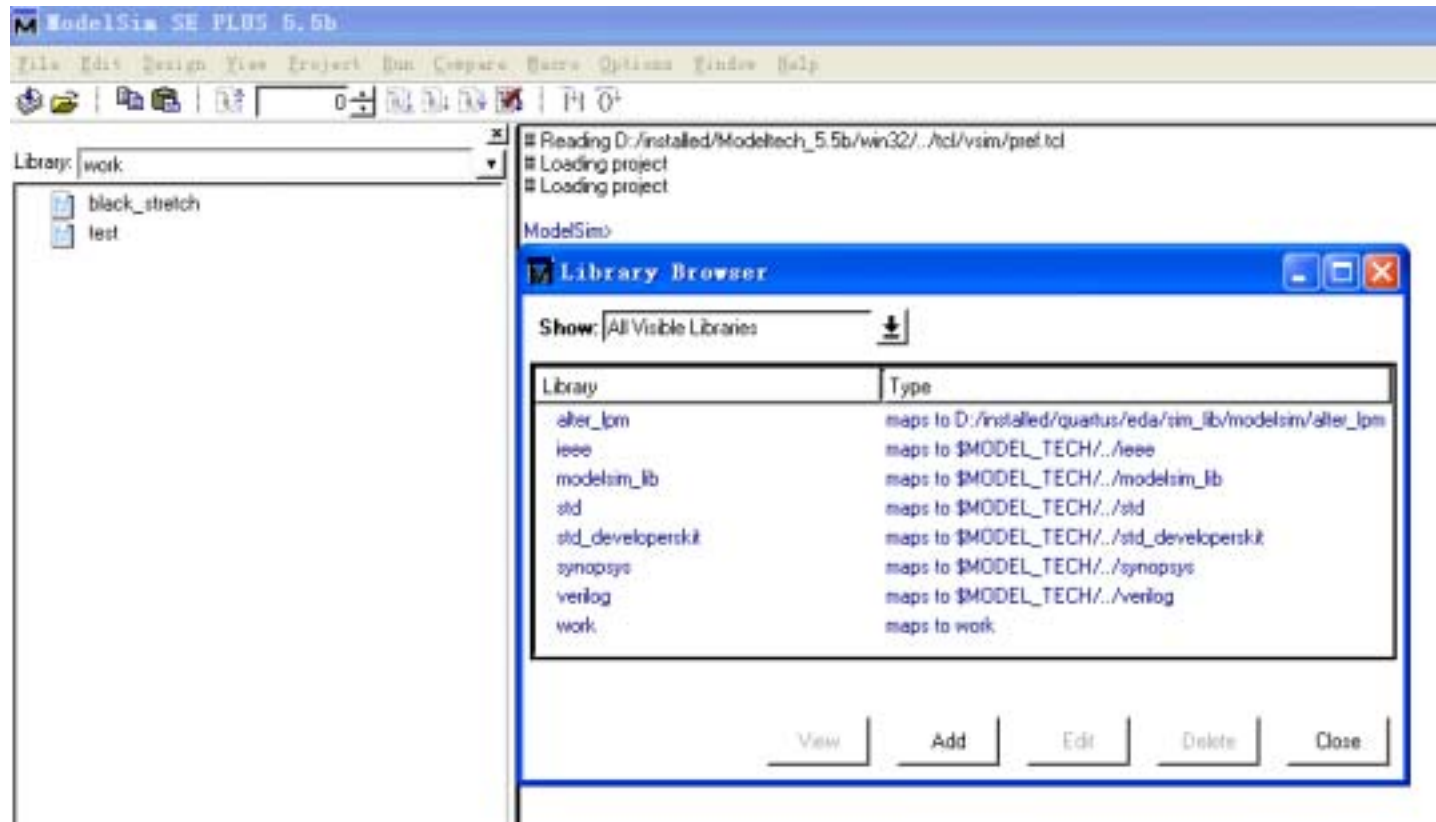
每个编译只允许一个

Resource

包含能被当前编译引用的设计单元

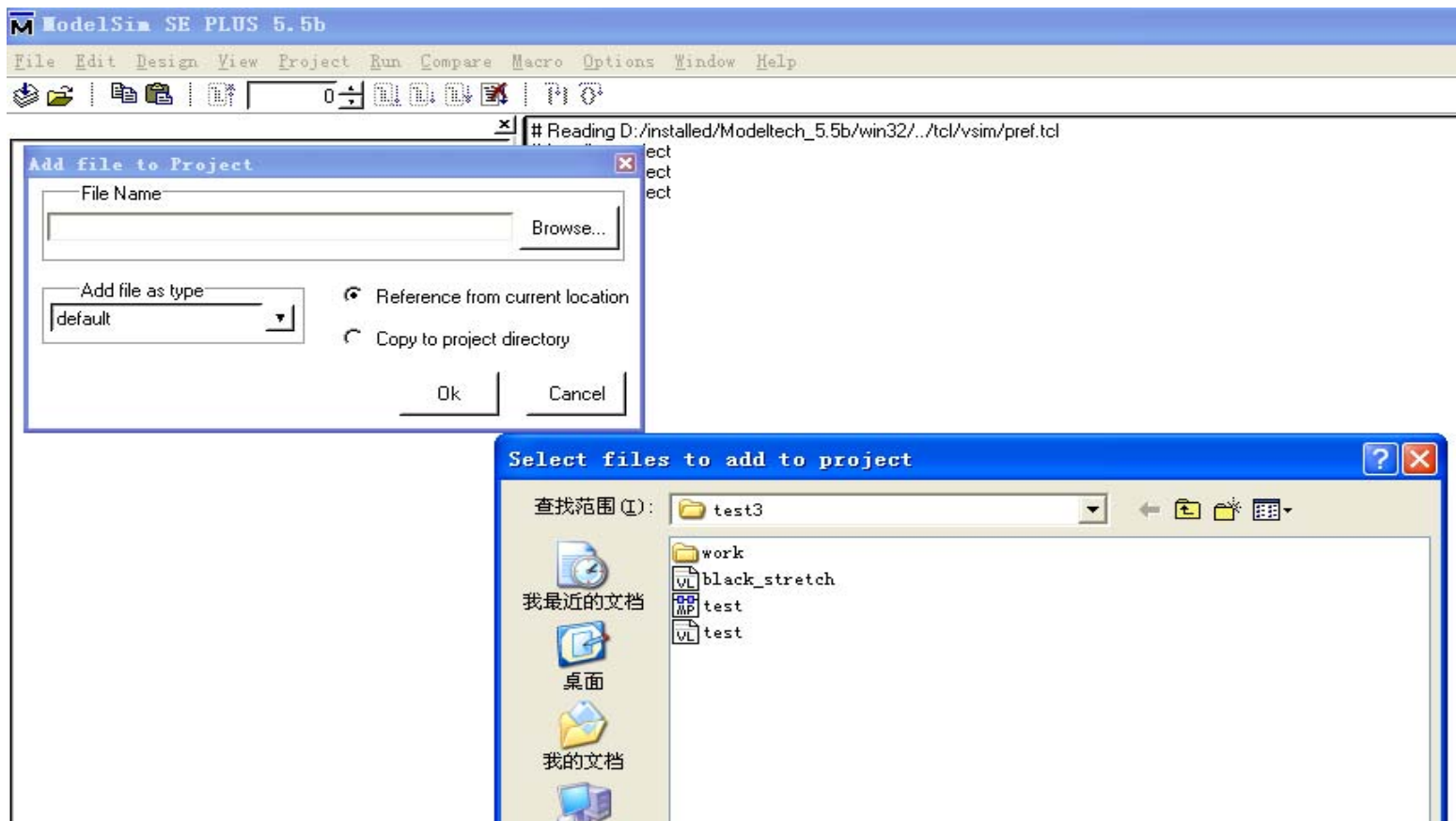
在编译期间允许多个

Library browser



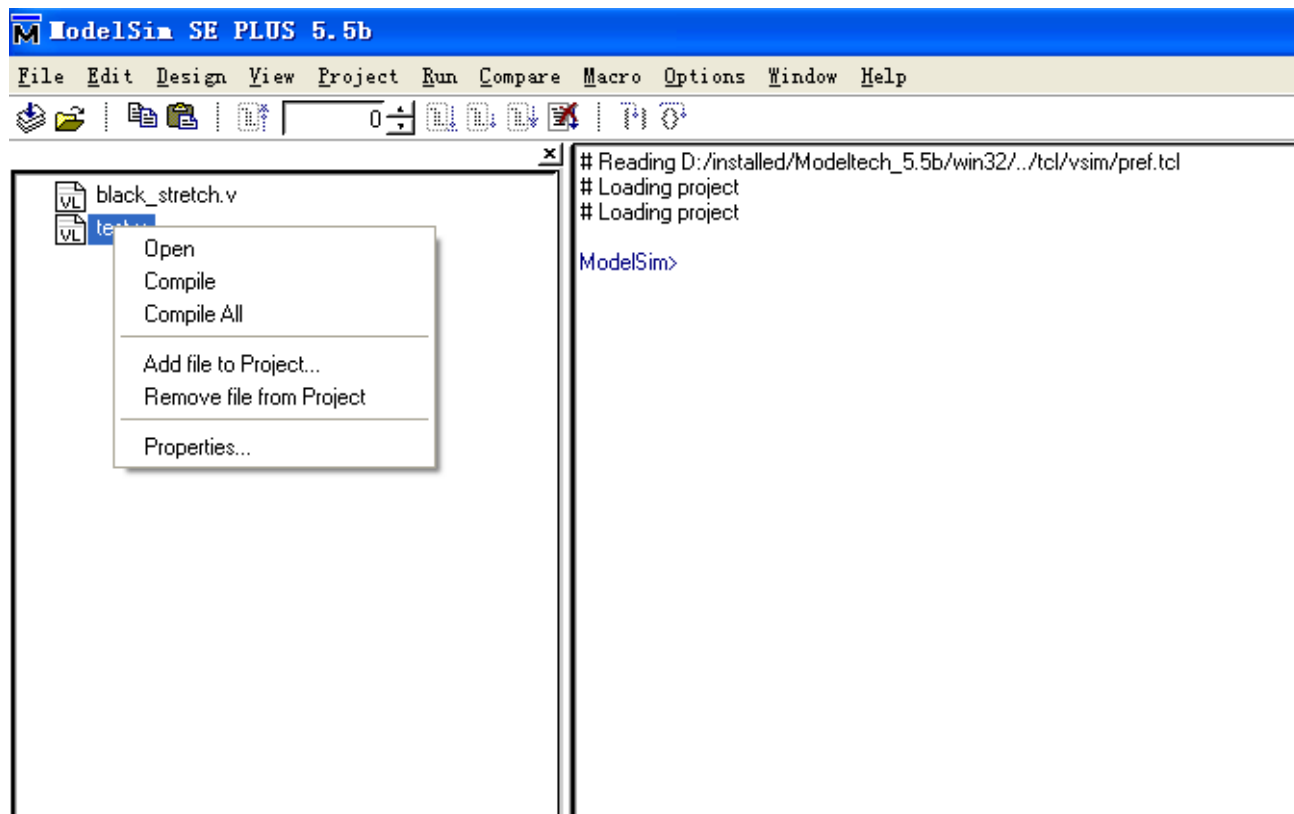
GUI) 从主菜单里面:
design -> library browser

2 ⇒ 加入源代码



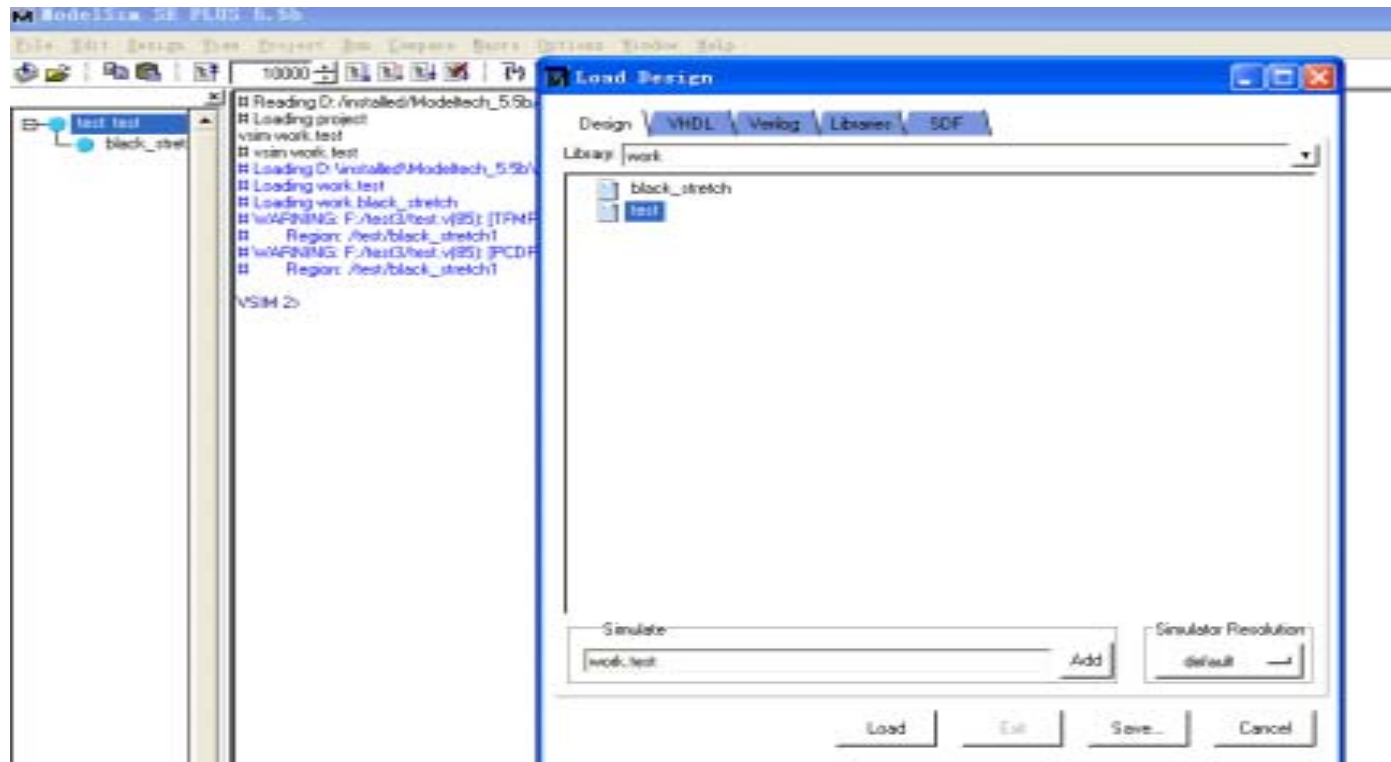
- GUI) *project -> add file to project*

compile



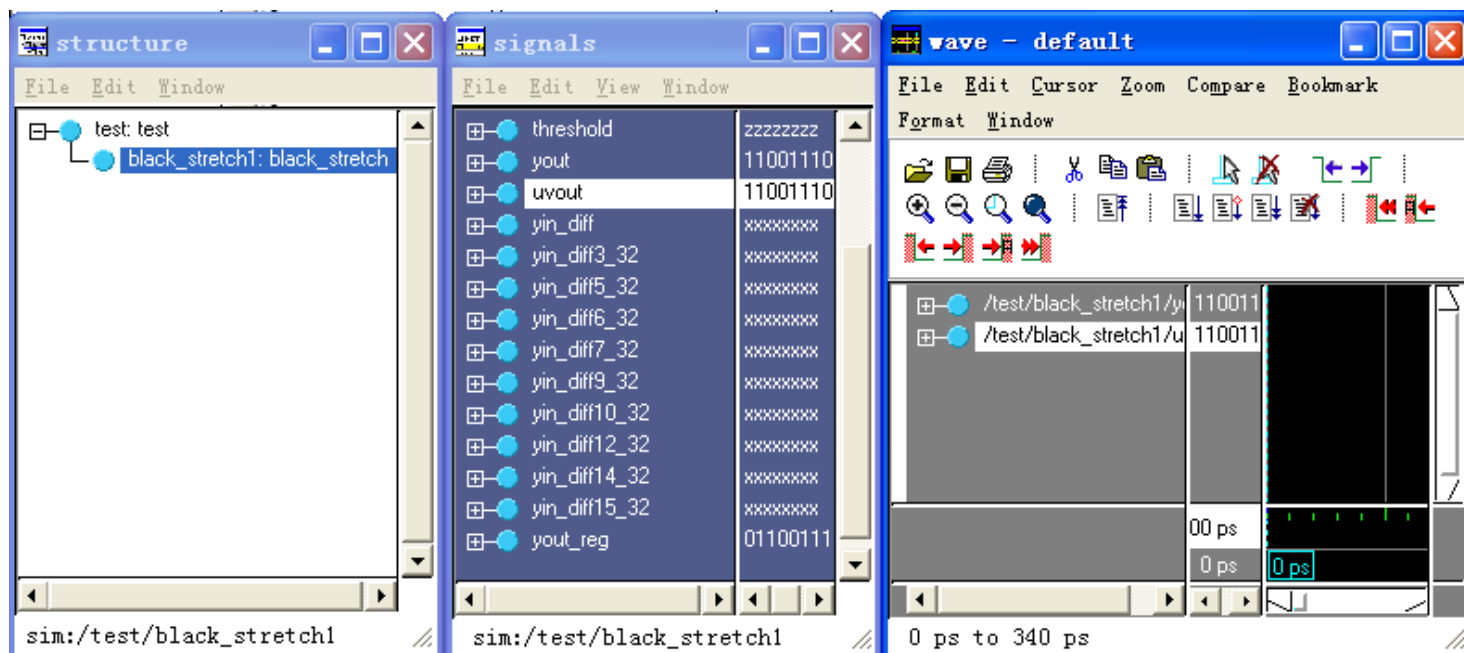
- GUI) *project* -> *compile order*
- GUI) *project* -> *compile all*
- 可通过在workspace 的project窗口上点击右键来完成相应操作

3 ⇒ 启动仿真器 (load design)



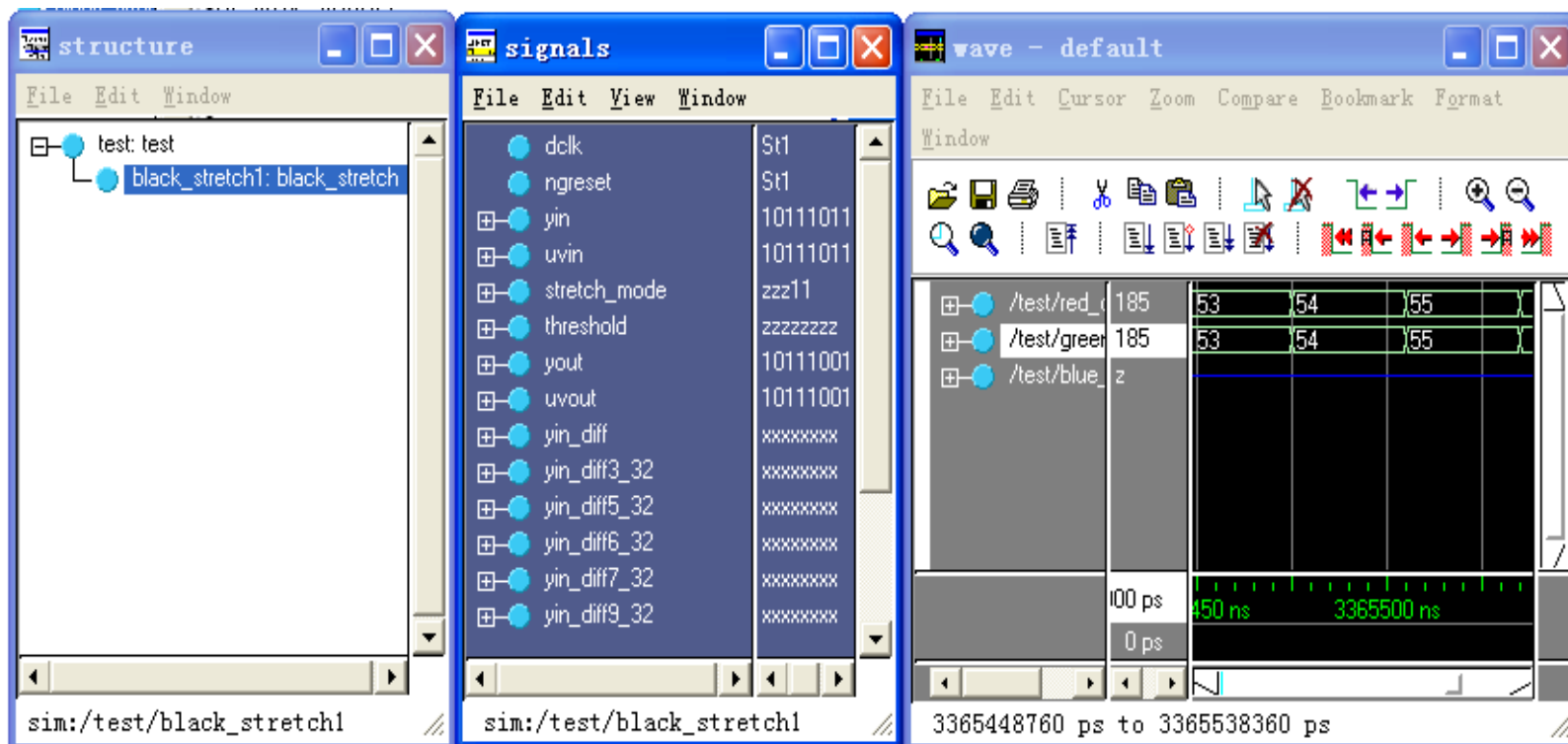
- GUI) *design -> load design*

4 ⇒ 添加波形



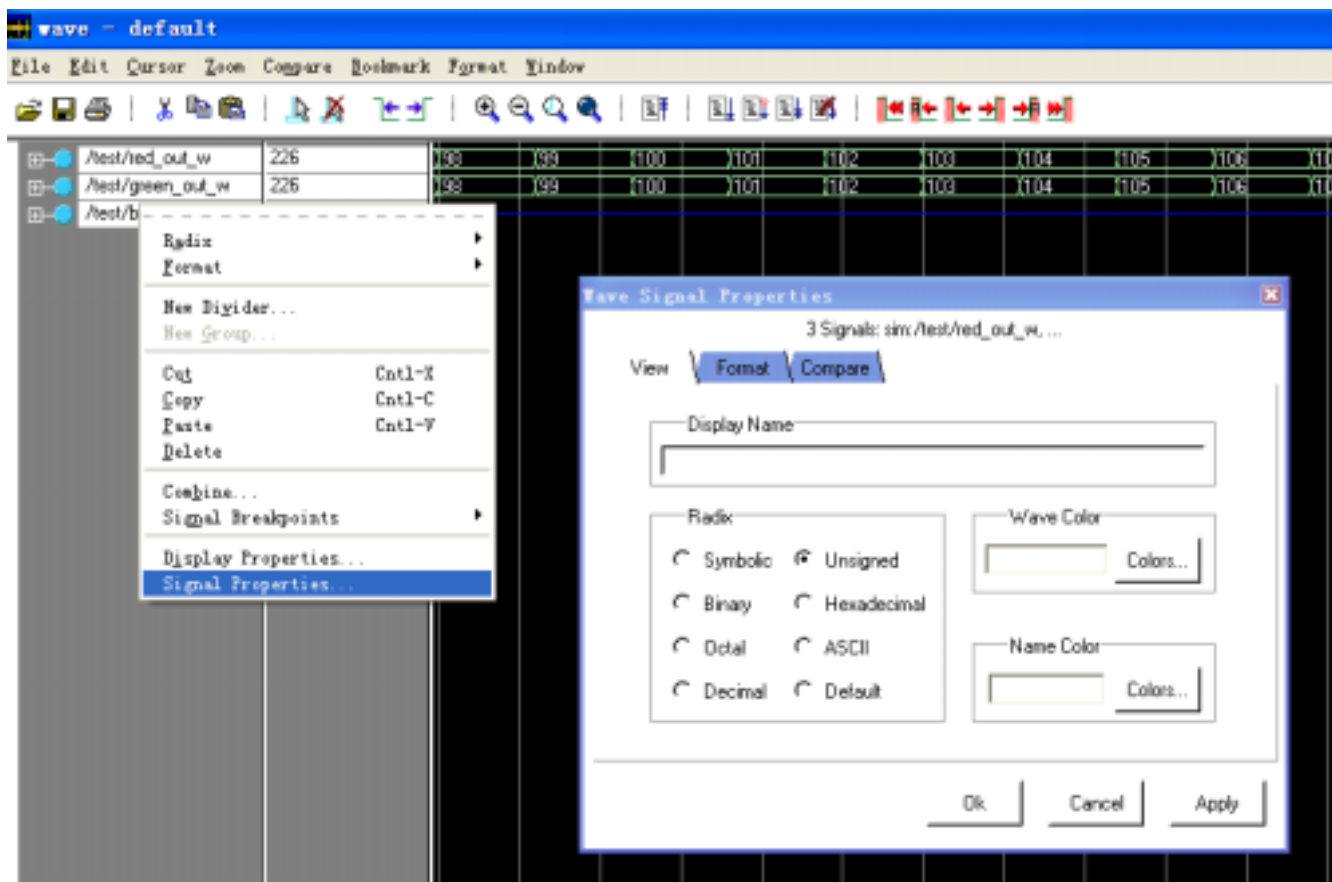
- GUI) **view** -> **structure** 选择需要显示的module
- GUI) **view** -> **signals** 选择所要显示的信号
- GUI) **view** -> **wave** 将所选择的信号调入波形窗口

5 ⇒ 运行 (run)



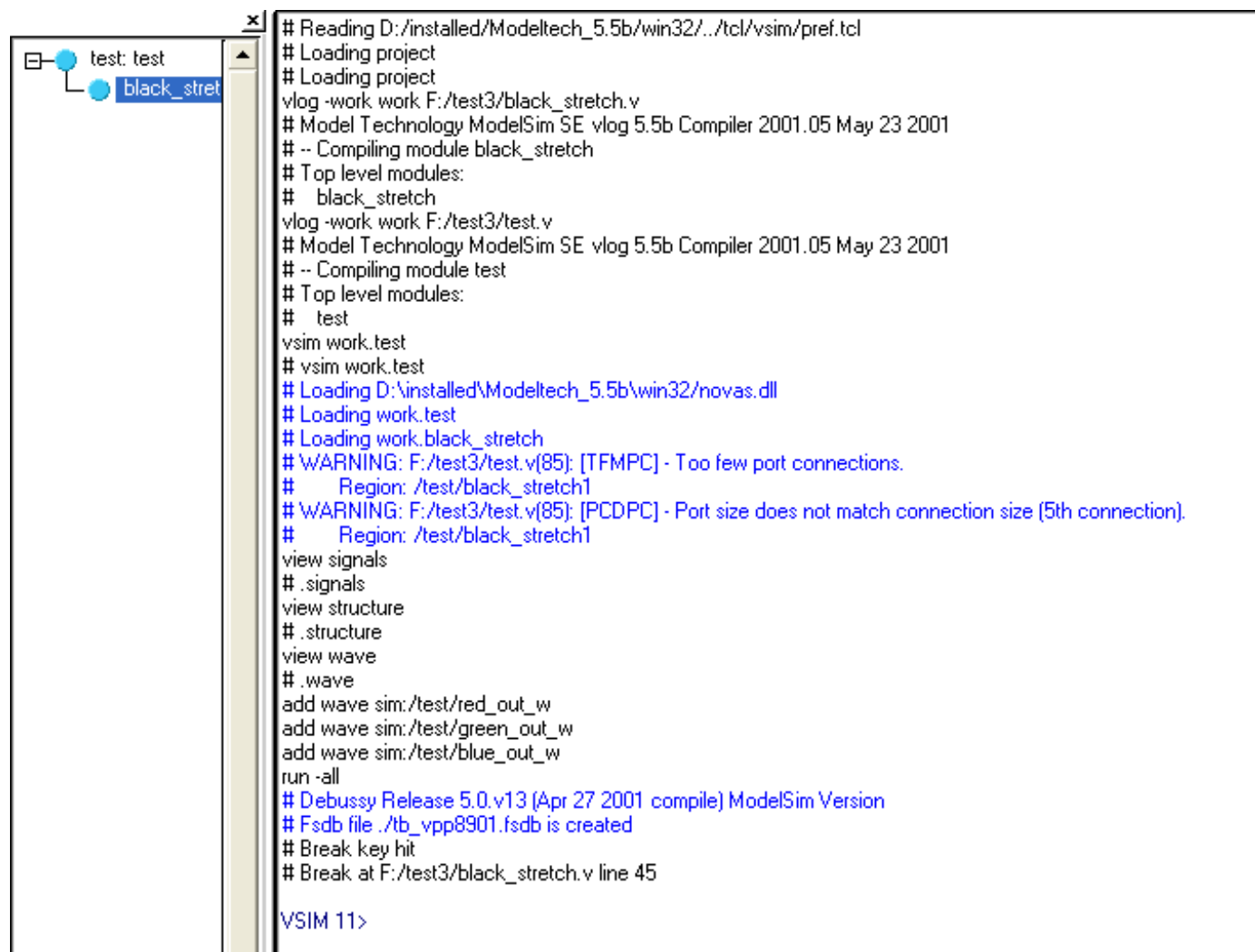
- GUI) `run -> run -a11` 也可以通过点击wave窗口的
- 点击wave窗口 停止仿真

Wave窗口操作



通过在wave窗口可以通过菜单、工具栏和在选中信号后点右键来完成相应的调试工作。

Modelsim窗口命令记录

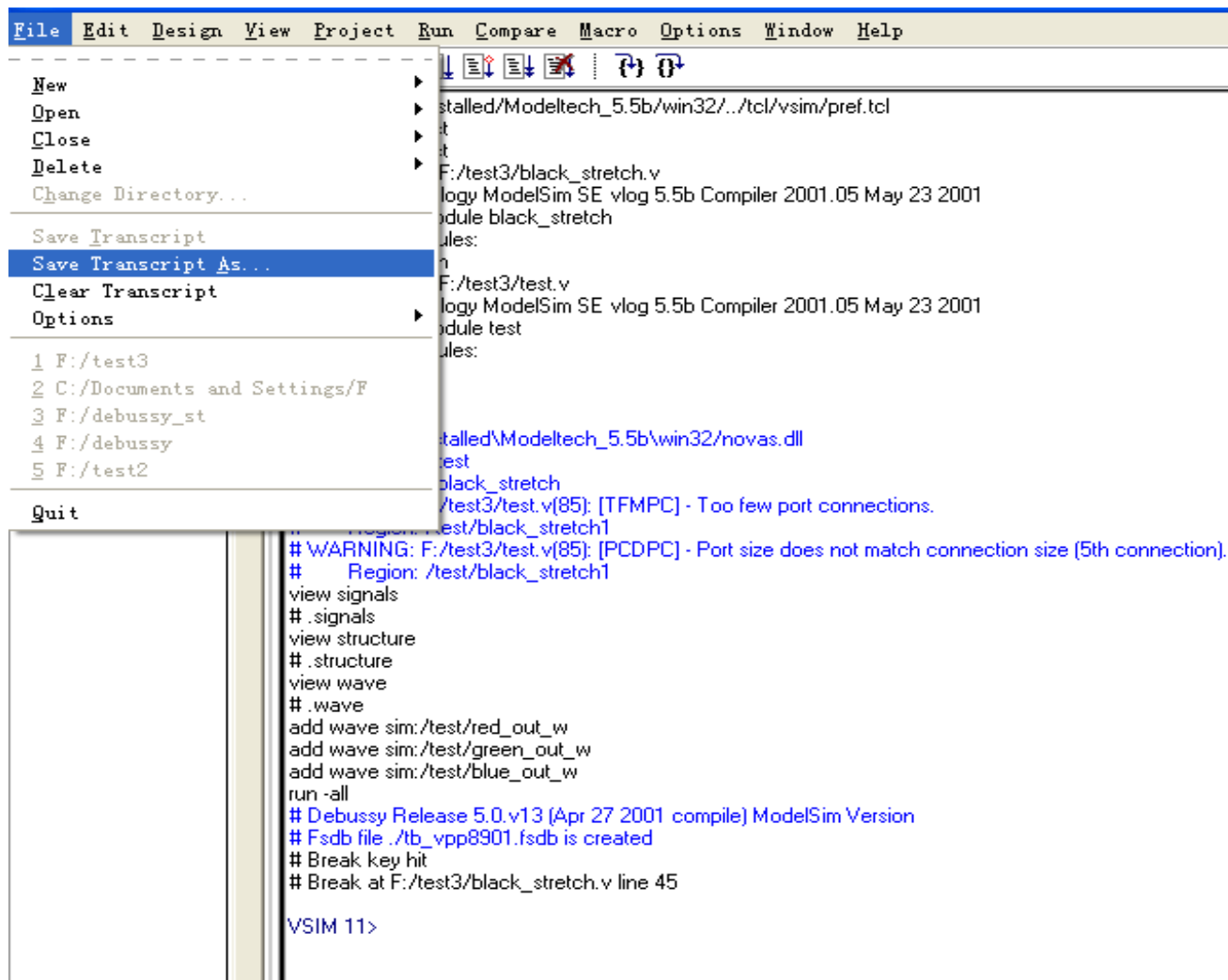


The screenshot shows the Modelsim command window with a project tree on the left and a command log on the right. The project tree shows a hierarchy: test: test (expanded) -> black_stret. The command log contains the following text:

```
# Reading D:/installed/Modeltech_5.5b/win32/.../tcl/vsim/pref.tcl
# Loading project
# Loading project
vlog -work work F:/test3/black_stretch.v
# Model Technology ModelSim SE vlog 5.5b Compiler 2001.05 May 23 2001
# -- Compiling module black_stretch
# Top level modules:
#   black_stretch
vlog -work work F:/test3/test.v
# Model Technology ModelSim SE vlog 5.5b Compiler 2001.05 May 23 2001
# -- Compiling module test
# Top level modules:
#   test
vsim work.test
# vsim work.test
# Loading D:/installed/Modeltech_5.5b/win32/novas.dll
# Loading work.test
# Loading work.black_stretch
# WARNING: F:/test3/test.v(85): [TFMPC] - Too few port connections.
#   Region: /test/black_stretch1
# WARNING: F:/test3/test.v(85): [PCDPC] - Port size does not match connection size (5th connection).
#   Region: /test/black_stretch1
view signals
# .signals
view structure
# .structure
view wave
# .wave
add wave sim:/test/red_out_w
add wave sim:/test/green_out_w
add wave sim:/test/blue_out_w
run -all
# Debussy Release 5.0.v13 (Apr 27 2001 compile) ModelSim Version
# Fsdb file ./tb_vpp8901.fsdb is created
# Break key hit
# Break at F:/test3/black_stretch.v line 45

VSIM 11>
```

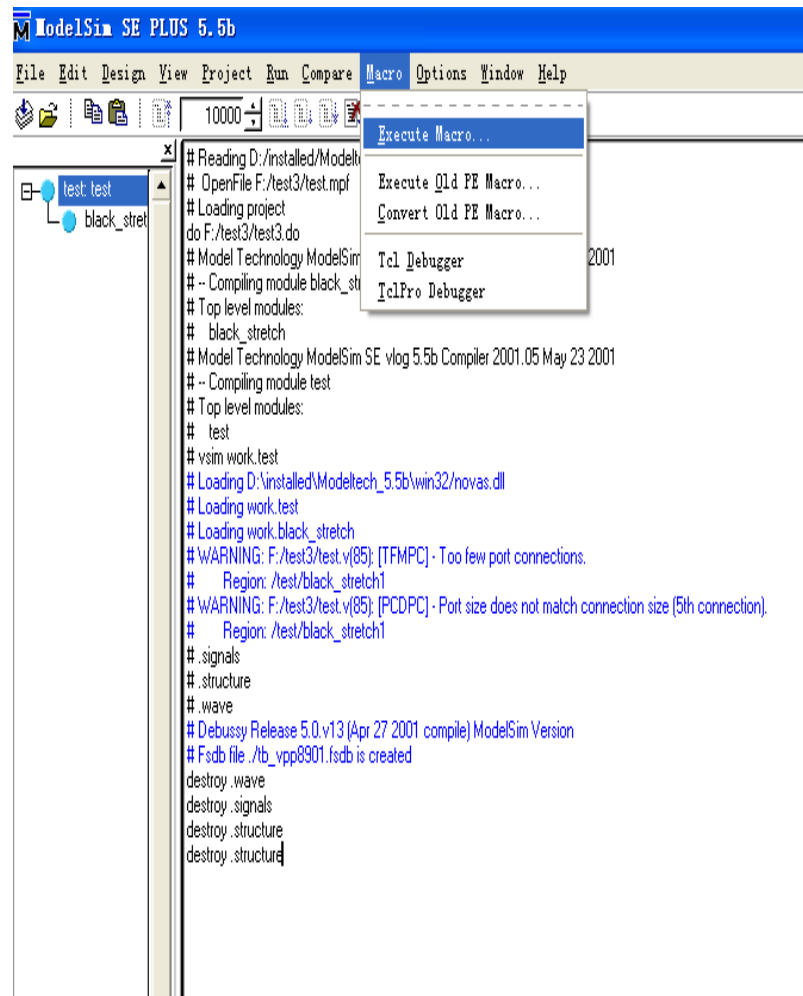
保存运行命令



将命令存储成.do格式，并调用

```
# test3.do
```

```
vlog -work work
    F:/test3/black_stretch.v
vlog -work work F:/test3/test.v
vsim work.test
view signals
view structure
view wave
add wave sim:/test/red_out_w
add wave sim:/test/green_out_w
add wave sim:/test/blue_out_w
run -all
```



工程文件test.mpf

- [Library]
- std = \$MODEL_TECH/./std
- ieee = \$MODEL_TECH/./ieee
- verilog = \$MODEL_TECH/./verilog
- std_developerskit = \$MODEL_TECH/./std_developerskit
- synopsys = \$MODEL_TECH/./synopsys
- modelsim_lib = \$MODEL_TECH/./modelsim_lib
- alter_lpm =
D:/installed/quartus/eda/sim_lib/modelsim/alter_lpm
- work = work
- [vcom]
- ; Turn on VHDL-1993 as the default. Default is off (VHDL-1987).
- ; VHDL93 = 1.
- ; RangeCheck = 1
- [vlog]
- ; Turn off inclusion of debugging info within design units.
- [vsim]
- Veriuser = novas.dll
- ; Default run length
- RunLength = 100

1)当下次再次打开这个工程时，modelsim从mpf文件中读取关于库的定位、启动文件的定位、ModelSim其他缺省设定等信息

2)在建立工程时，modelsim从初始文件 modelsim.ini中获取各种信息，包括[Library] [vcom] [vlog] [vsim] [lmc]，因此有必要修改modelsim.ini文件以改变一些缺省信息。

3)缺省modelsim.ini文件存在modelsim的安装目录下，为只读文件。

4)modelsim.ini文件的搜索顺序为：

- a)环境变量所指的文件架
- b)当前的工作目录
- c)modelsim的安装目录

工程文件test.mpf (续)

; Maximum iterations that can be run without advancing simulation time

IterationLimit = 5000

License = license.dat

; VSIM Startup command

; Startup = do startup.do

; File for saving command transcript

TranscriptFile = transcript

; File for saving command history

CommandHistory = cmdhist.do.

[lmc]

; ModelSim's interface to Logic Modeling's SmartModel SWIFT software

libsm = \$MODEL_Tech/libsm.sl

[project]

Project_Version = 1

Cur_Top_DUs = work.test

Modelsim使用中一些问题讨论

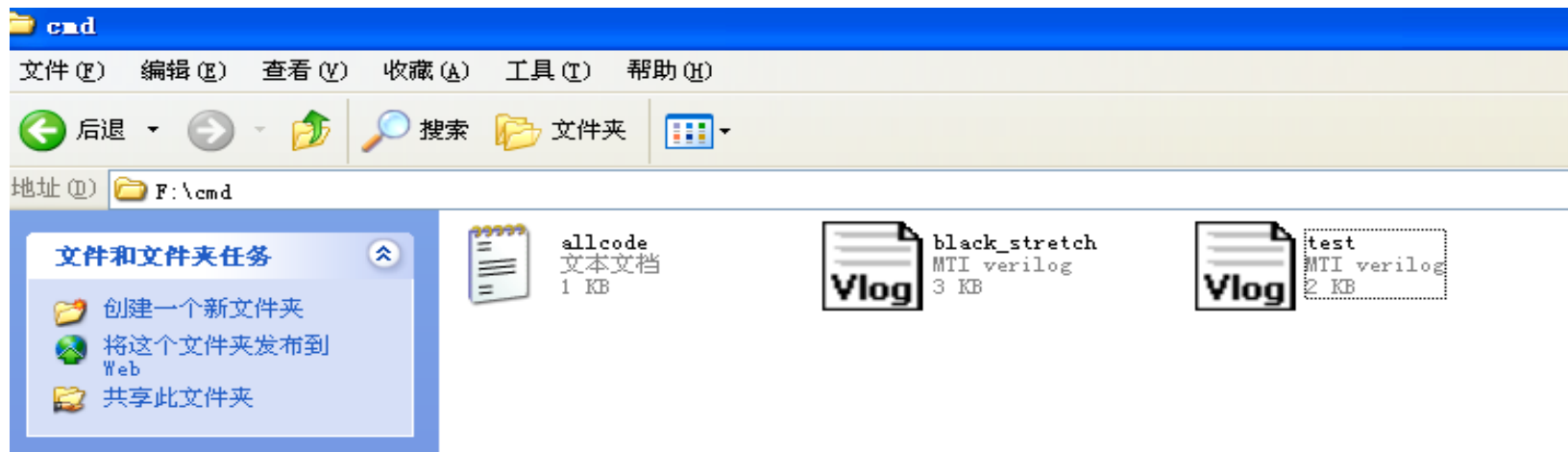
- 1) 命令行模式和批处理模式
- 2) 建库
- 3) vcd 格式文件输出
- 4) 时序仿真
- 5) PLI函数与debussy支持

Command-line mode (命令行模式)

This an operational mode that has only an interactive command line; no interactive , windows are opened. To run **vsim** in this manner, invoke it with the **-c** option as the first argument from either the UNIX prompt or the DOS prompt in Windows 95/98/2000/NT.

在前台运行，不显示modelsim的可视化界面。

运行前工作目录下的文件

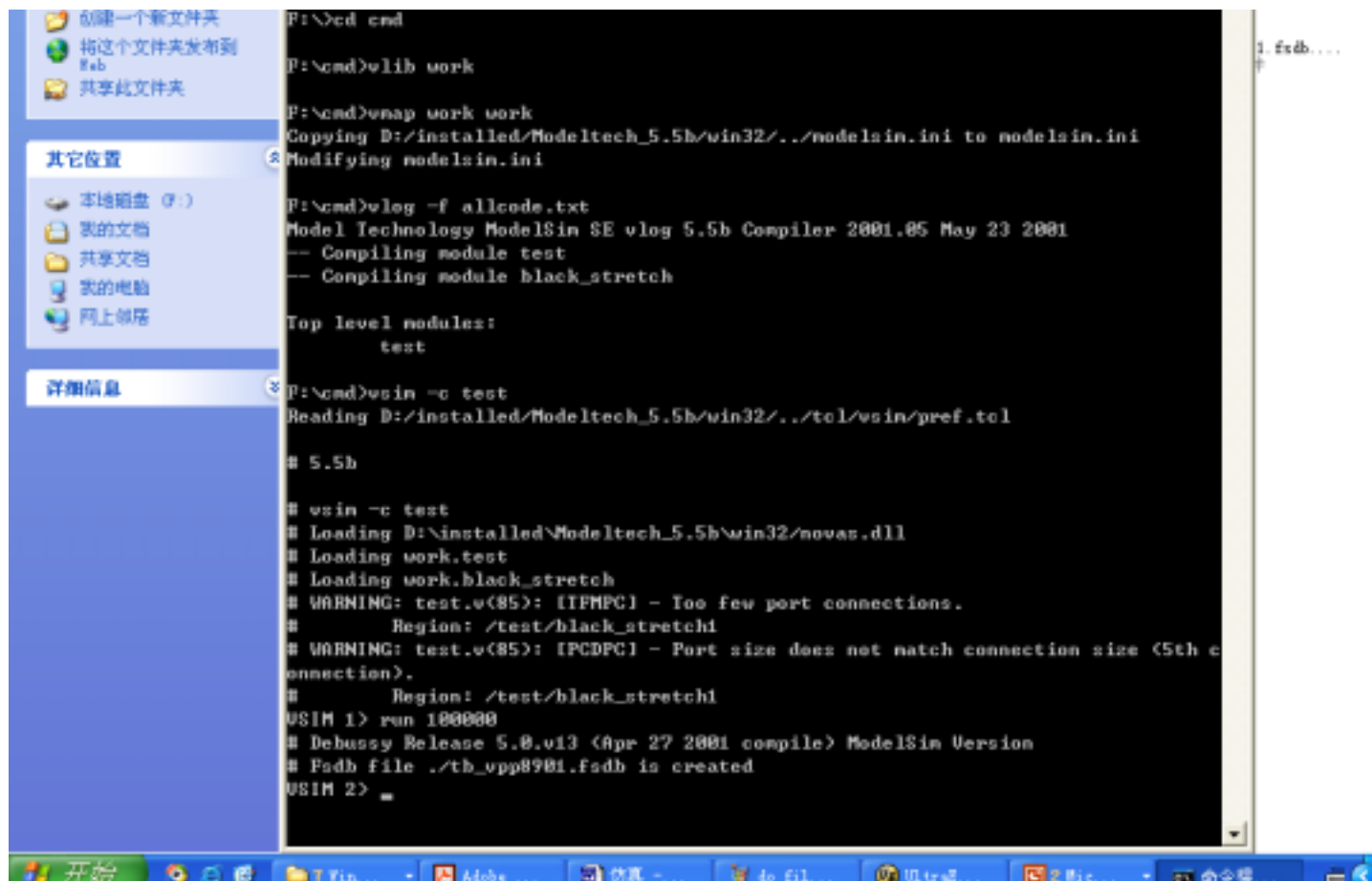


Allcode.txt 文件中的内容

1 : test.v

2 : black_stretch.v

DOS下的命令行模式



```
F:\>cd cmd
F:\cmd>mkdir work
F:\cmd>cd work
F:\cmd>copy D:\installed\Modeltech_5.5b\win32\..\modelsim.ini .
F:\cmd>copy D:\installed\Modeltech_5.5b\win32\..\modelsim.ini .
F:\cmd>vlog -f allcode.txt
Model Technology ModelSim SE vlog 5.5b Compiler 2001.05 May 23 2001
-- Compiling module test
-- Compiling module black_stretch

Top level modules:
    test

F:\cmd>vsim -c test
Reading D:\installed\Modeltech_5.5b\win32\..\tol\vsim/pref.tol

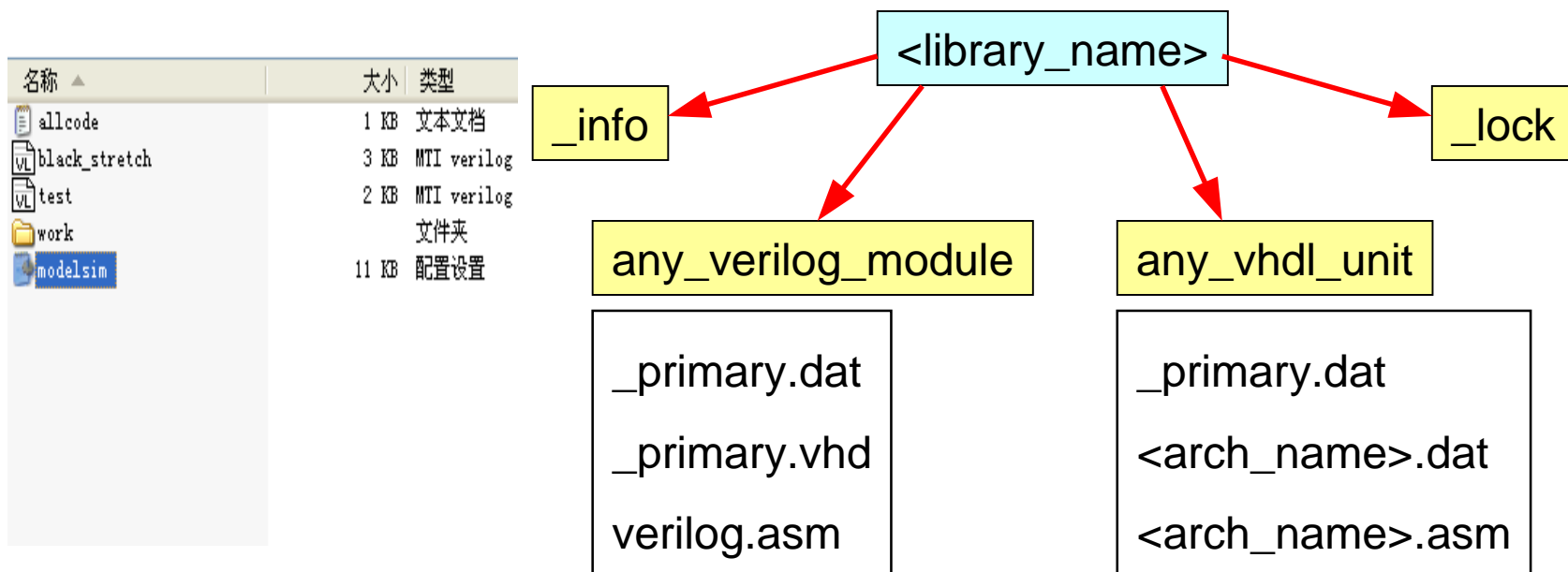
# 5.5b

# vsim -c test
# Loading D:\installed\Modeltech_5.5b\win32\novas.dll
# Loading work.test
# Loading work.black_stretch
# WARNING: test.v(85): [IFMPC] - Too few port connections.
#       Region: /test/black_stretch1
# WARNING: test.v(85): [PCDPC] - Port size does not match connection size (5th c
#       Region: /test/black_stretch1
USIM 1> run 100000
# Debussy Release 5.0.v13 (Apr 27 2001 compile) ModelSim Version
# Fldb file ./th_opp8981.fldb is created
USIM 2>
```

Vlib和Vmap

- The **vlib** command creates a design library. You must use **vlib** rather than operating system commands to create a library directory or index file.
- The **vmap** command defines a mapping between a logical library name and a directory by modifying the *modelsim.ini* file. With no arguments, vmap reads the appropriate *modelsim.ini* file(s) and prints the current logical library to physical directory mappings. Returns nothing.

Vlib 和 Vmap 命令后结果



- 运行完Vlib 后会产生 work库目录，目录里存放_info文件，用于记录各种库中的各种模块。
- 运行完Vmap 会将modelsim安装目录下的modelsim.ini 复制到当前工作目录，并将库和目录对应起来，在[Library]在增加work = work语句。

Vlog命令

- **vlog**
- The **vlog** command is used to invoke VLOG, the Model Technology Verilog compiler.
- **Syntax**
- **vlog**
 - [-93] [-help] [-compat] [-
 - [+define+<macro_name>[=<macro_text>]]
 - [+delay_mode_path] [+delay_mode_unit]
 - [-fast[=<secondary_name>]
 - [-hazards] [+incdir+<directory>] [-incr]
 - [-line <number>] [-lint] [+mindelays] [+maxdelays]
 - [+nocheckCLUP] [+nocheckDNET] [+nocheckOPRD]
 - [-nodebug[=ports | =pli]] [-noincr] [+nolibcell] [-
 - [+nowarn<CODE>] [-O0 | -O1 | -O4 | -O5] [-quiet] [-R
 - [-source] [+typdelays] [-u] [-v <library_file>]
 - [-version] [-work <library_name>] [-y

Vlog 后 Work目录下的信息

名称	大小	类型
_info	1 KB	文件
test		文件夹
black_stretch		文件夹

名称	大小	类型
verilog	10 KB	ASM 文件
_primary	1 KB	MTI vhd1
_primary	2 KB	DAT 文件

- Where
 - _primary.dat - Verilog module 或 VHDL entity的编码格式
 - _primary.vhd - Verilog 端口的VHDL entity陈述
 - verilog.asm 和 <arch_name>.asm – 执行代码文件

Vsim

- The **vsim** command is used to invoke the VSIM simulator, or to view the results of a previous simulation run (when invoked with the **-view** switch).
- **Syntax**
- vsim
- [-c] [-coverage] [-do "<command_string>" |
- [-f <filename>] [-g<Name>=<Value> ...] [-
- [-help] [-i] [-installcolormap] [-keeploaded] [-
- [-keepstdout] [-l <filename>] [-lib
- [-multisource_delay min | max | latest] [-
- [+no_tchk_msg] [+notimingchecks] [-quiet]
- [-restore <filename>]
- [-sdfmin | -sdftyp | -sdfmax
- [-sdfnoerror] [-sdfnowarn] [+sdf_verbosel] [-t
- [-tag <string>] [-title <title>] [-trace_foreign
- [-view [<dataset_name>=<WLF_filename>] [-wlf
- <size>]

Vsim (续)

```
[-wlftlim <duration>] [-wlnocompress]
[-absentisempty] [-foreign <attribute>] [-
[-noglitch] [+no_glitch_msg] [-std_input
[-std_output <filename>] [-strictvital]
[-vcdread <filename>] [-vital2.2b]
[+alt_path_delays] [-hazards] [-L
[-Lf <library_name> ...] [+maxdelays] [+mindelays]
[+multisource_int_delays] [+no_neg_tchk]
[+nosdfwarn] [+nosdferror] [+nowarn<CODE>]
[-pli "<object list>"] [+<plusarg>]
[+pulse_e_style_ondetect] [+pulse_e_style_onevent]
[+pulse_int_e/<percent>] [+pulse_int_r/<percent>]
[+sdf_nocheck_celltype] [+transport_int_delays]
[+typdelays] [<library_name>.<design_unit>]
```

Batch mode

Batch mode

- Batch mode is an operational mode that provides neither an interactive command line, nor interactive windows.
- 其所有操作都在后台进行，用户看不到modelsim的界面，也不需要交互式输入命令。当工程很大，文件比较多时，用批处理比较方便。
- 增加了两个文件，一个是runallcode.bat(批处理文件)，一个是runallcode.do（modelsim的脚本文件）。

runallcode.bat

vsim -c -do runallcode.do

runallcode.do

vlib work

vmap work work

vlog -work work -f allcode.txt

vsim work.test

run 100000000

quit -f

运行批处理文件前后比较

名称 ▲	大小	类型
allcode	1 KB	文本文档
black_stretch	3 KB	MTI verilog
runallcode	1 KB	MS-DOS 批处理文件
runallcode.do	1 KB	DO 文件
test	2 KB	MTI verilog

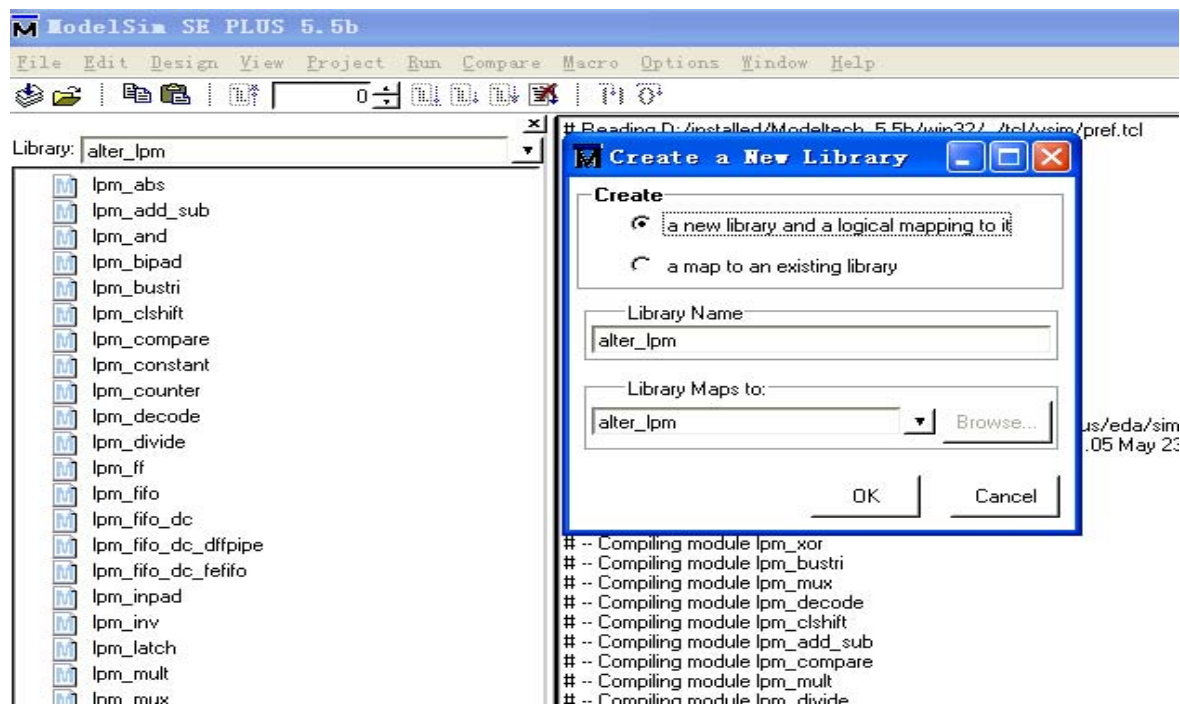
名称 ▲	大小	类型
allcode	1 KB	文本文档
black_stretch	3 KB	MTI verilog
runallcode	1 KB	MS-DOS 批处理文件
runallcode.do	1 KB	DO 文件
test	2 KB	MTI verilog
work		文件夹
1tb_vpp8901.fsdb	46 KB	FSDB 文件
modelsim	11 KB	配置设置
transcript	1 KB	文件

- 直接运行runallcode.bat文件，会在后台调用modelsim，执行runallcode.do文件，完成操作。
- 因为批处理和命令行格式没有调用modelsim的窗口，因此必须采用数据转存，将仿真数据存储成VCD格式或则FSDB格式，作off-line分析。

仿真Alter 宏函数、LPM

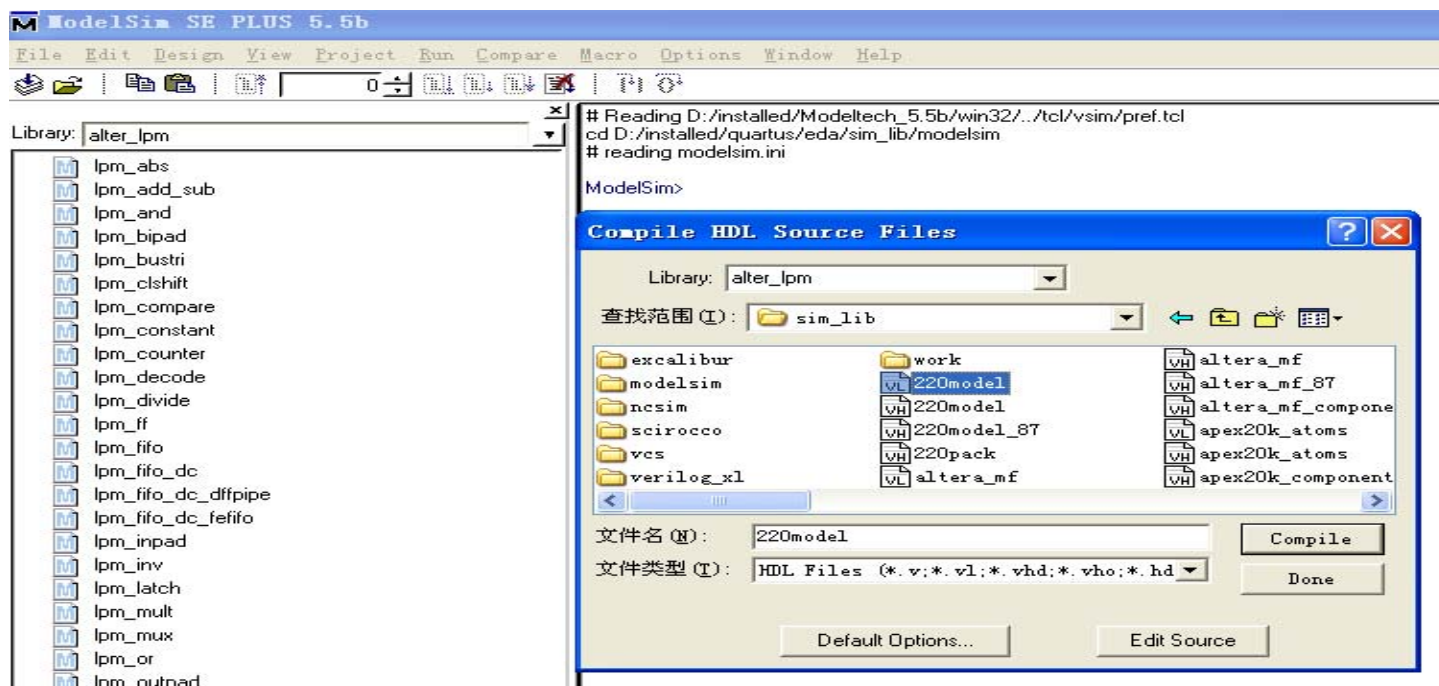
- Alter 提供的宏函数、LPM和通过MegaWizard插件等功能为FPGA设计提供了许多方便，在quartus\eda\sim_lib提供了220model.v和alter_mf.v，里面存在各种模型。
- 在仿真LPM，最直接的想法是将220model.v复制到工作目录，或者从220model.v中找出所需要的module，然后和源文件一起编译；或则为其建一个库alter_lpm（可以建立在任何目录下，我们这里库目录为D:/installed/quartus/eda/sim_lib/modelsim/alter_lpm），将220model.v中的module都编译到库里。
- 如果我们经常用到这个库的话，我们可以在modelsim的安装目录下将modelsim.ini 进行修改，在[Library]增加alter_lpm = D:/installed/quartus/eda/sim_lib/modelsim/alter_lpm，这样当我们在新建工程时，alter_lpm库就会和其他的资源库一起出现在library里，不需要再从新编译。

在D:/installed/quartus/eda/sim_lib/modelsim/目录下建立alter_lpm 库



- 先将工作目录转换到D:/installed/quartus/eda/sim_lib/modelsim
- GUI) *design -> create a new library*

在alter_lpm库下编译model220.v



- GUI) *design -> compile*
- 在library中选择alter_lpm，然后选择220model.v文件，并点击编译，那么在alter_lpm库中便存在了编译好的许多modele。

在modlesim.ini文件添加alter_lpm库

- [Library]
- std = \$MODEL_TECH/./std
- ieee = \$MODEL_TECH/./ieee
- verilog = \$MODEL_TECH/./verilog
- std_developerskit = \$MODEL_TECH/./std_developerskit
- synopsys = \$MODEL_TECH/./synopsys
- modelsim_lib = \$MODEL_TECH/./modelsim_lib
- alter_lpm = D:/installed/quartus/eda/sim_lib/modelsim/alter_lpm

- [vcom]
- ; Turn on VHDL-1993 as the default. Default is off (VHDL-1987).
- ; VHDL93 = 1

Value Change Dump (VCD) Files

- The VCD file format is specified in the IEEE 1364 standard. It is an ASCII file containing header information, variable definitions, and variable value changes. VCD is in common use for Verilog designs, and is controlled by VCD system task calls in the Verilog source code.

- 可以通过在源代码中添加语句来实现
initial begin

```
$Dumpfile("./test.fsdb");
```

```
$Dumpvars(0,test);
```

```
#600000000 $finish;
```

```
end
```

- 也可以通过modelsim的命令来实现

```
vcd add (CR-200)
```

```
$dumpvars
```

转储层次信号

```
vcd file (CR-210)
```

```
$dumpfile
```

打开一个文件准备转储波形数据

```
vcd off (CR-216)
```

```
$dumpoff
```

停止转储

```
vcd on (CR-217)
```

```
$dumpon
```

开始转储

SDF (Standard Delay Format) timing annotation.

- 可以采用\$sdf_annotate 系统任务可进行反标：

initial begin

```
$sdf_annotate("black_stretch.sdf", black_stretch1);
```

end

- 也可以采用vsim命令来进行反标

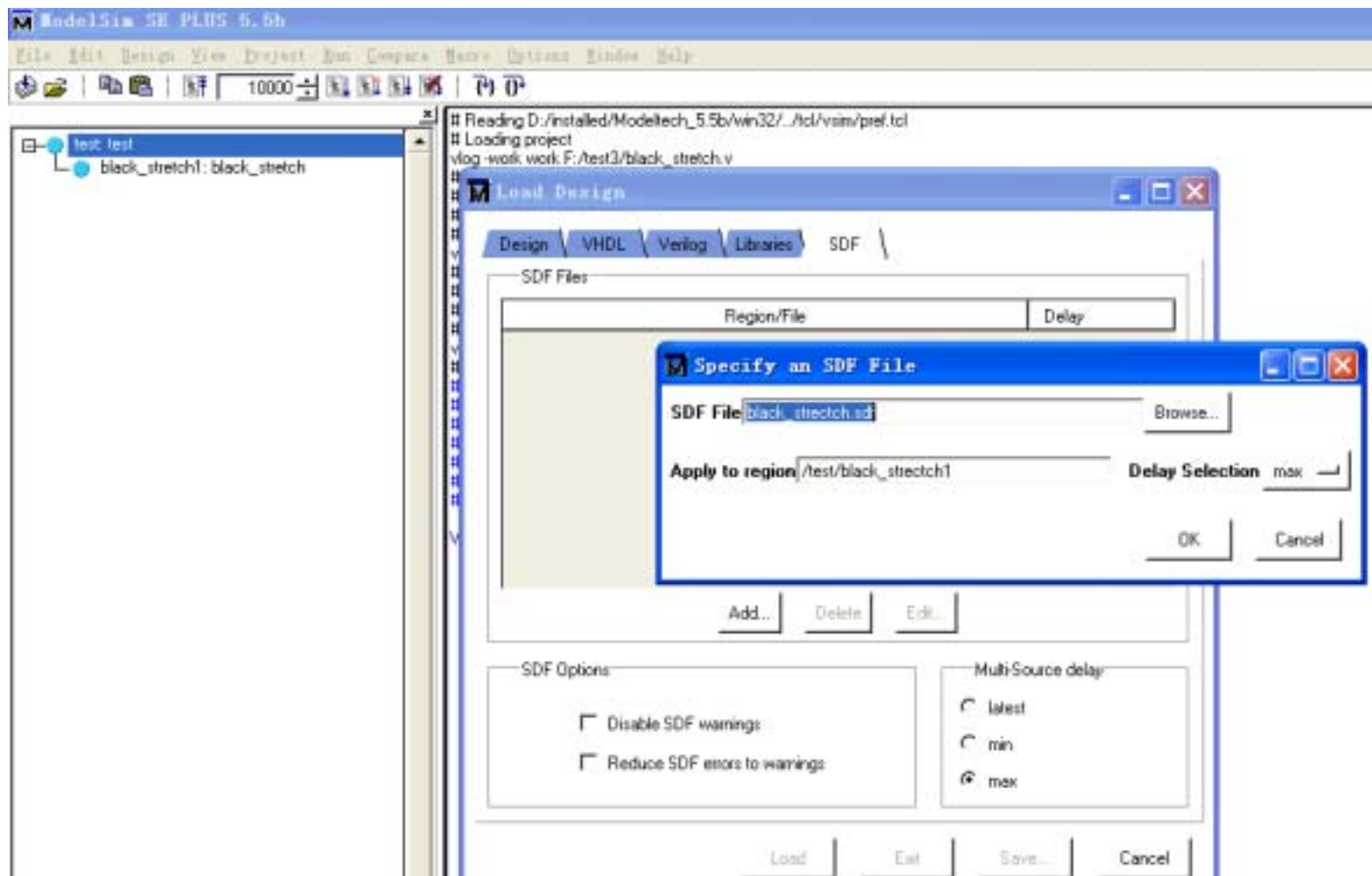
```
vsim -sdfmax
```

- 还可以通过GUI窗口来实现延时文件的反标

GUI) *design -> load design*

点击load design 窗口的 sdf 选项，添加sdf文件

GUI窗口添加SDF文件



Debussy : 仿真辅助调试工具 :

- a)看仿真波形无疑是代码排错的主要手段，在Modelsim中的波形窗口在大的仿真中有如下缺陷：a、只能显示出在仿真前设置好的信号波形，如果在仿真完成后想观察其他的信号，唯一的办法就是添加需要观察的信号从新开始仿真。b、波形只是简单显示，和代码没有对应和关联关系，不能借助波形直观的调试代码；c、如果观察的信号太多，由于其是实时全信号显示，在仿真时间较长后，仿真速度明显减慢，屏幕的刷新速度也明显减慢。
- b)这些缺点不单Modelsim有，其他的优秀仿真工具也有，而且历史由来以久，因此很早人们就提出了“先转储后观察调试”方法，在verilog语言中以\$dumpXXX开头的系统函数就是做波形转储用的。就是先将波形先存在文件中，等仿真结束后在调出来显示观察和调试。
- c)这种观察功能很多EDA工具都有，并不足为奇；但Debussy的独特之处在于，它不但能显示波形，而且还能非常智能化的将波形中的任何一个变化和引起这个变化的RTL代码联系起来，使代码排错的效率大幅度提高。在原来IC所的一个大型项目中，由于引进了Debussy，使调试效率至少提高了3倍。

Debussy 的数据转存函数

- Debussy 提供的新的波形文件格式FSDB(Fast Signal DataBase)，相比于VCD格式，压缩量大，比vcd文件小5-50倍，加载速度快。
- Debussy提供了PLI (for Verilog) 和FLI (for VHDL) 接口，我们可以在仿真时直接导出FSDB文件。例如，在Verilog的测试模块中加入如下语句：

```
initial
begin
    $fsdbDumpfile("test.fsdb");
    $fsdbDumpvars (0,test);
end
```

PLI简介

- PLI = Verilog Program Language Interface，也称为Verilog PLI。简单来说，PLI提供一种接口，将用户编写的C或C++程序连接到verilog仿真器上，实现verilog仿真器的功能扩展和定制。
- PLI接口主要提供以下三种功能。
 1. PLI接口允许用户编写自定义的system task和system function。用户写出相应的PLI程序并连接到仿真器后，就可以在自己写的verilog程序中使用这些system task和function了。一旦这些task/function在仿真过程中被调用，仿真器就会找到对应的用户编写的PLI程序来执行，从而实现仿真器的定制。
 2. 这个接口还允许用户在自己的PLI程序中与仿真器中实例化的verilog硬件进行交互，比如读一个wire的值，向一排reg写值，设置一个cell的delay，等等。不夸张地说，对于PLI程序而言，仿真器中的verilog实例完全是透明的，用户想对这些硬件做什么操作都可以。（当然，硬件结构不能修改）有了这个功能，用户就可以在自定义的task/function中对硬件执行某些用verilog语言难以完成的操作。
 3. 某些特定的操作需要对仿真过程中一些信号的变化做出响应。虽然我们可以用always来监控少量信号的变化，但如果需要监测大量信号，这种机制并不现实。PLI接口提供了一种函数回调机制解决这个问题。用户可以将某个wire/reg等信号挂上一个PLI程序中的C函数，以后每当该信号变化，这个C函数都会被调用，从而很方便地实现信号监测。事实上，很多打波形的system task都是用这个方法实现的。

Linking Debussy PLI routines with MTI ModelSim (5.2 or later)

1. Locate pli.lib and veriusers.c, which Debussy provides in the following directory:
`<install_dir>/share/PLI/modelsim_pli/WINNT/pli.lib`
`<install_dir>/share/PLI/modelsim_pli/WINNT/veriusers.c`
2. If you are using PLI applications with your ModelSim, you need to combine your existing veriusers.c and the Debussy-provided veriusers.c into a single file. If you are not using PLI applications with your ModelSim you can use the Debussy-provided veriusers.c file.

3. Generate PLI dynamic link library (novas.dll) with the following steps:

```
>cl -c -I<modelsim_install_dir>/modeltech/include veriusers.c
>link -dll -out:novas.dll -nodefaultlib:libcmt -export:init_usertfs veriusers.obj pli.lib shell32.lib
<modelsim_install_dir>\modeltech\WINNT\mtipli.lib
```

Debussy also provides a batch file "make.bat" for user to make a .dll file. You can modify it to fit your file directory. The file is under:

```
<install_dir>\share\PLI\modelSim\WINNT
```

4. Set the dynamic link library path to the directory containing the library produced.

For instance, you have novas.dll under <user>/novas/PLI, add <user>/ novas/PLI to your LD_LIBRARY_PATH in your windows environment.

5. Edit your modelsim.ini file to instruct ModelSim to load the shared object produced.

*Add this Veriusers entry to the [VSIM] section:

```
[VSIM] .
```

```
Veriusers = novas.dll
```

MODESIM 调用debussy的系统函数

- Copy

X:\Novas\Debussy\share\PLI\modelsim_pli\WINNT\novas.dll
to X:\modelsim\win32\

- Edit X:\modelsim\modelsim.ini and add block box text

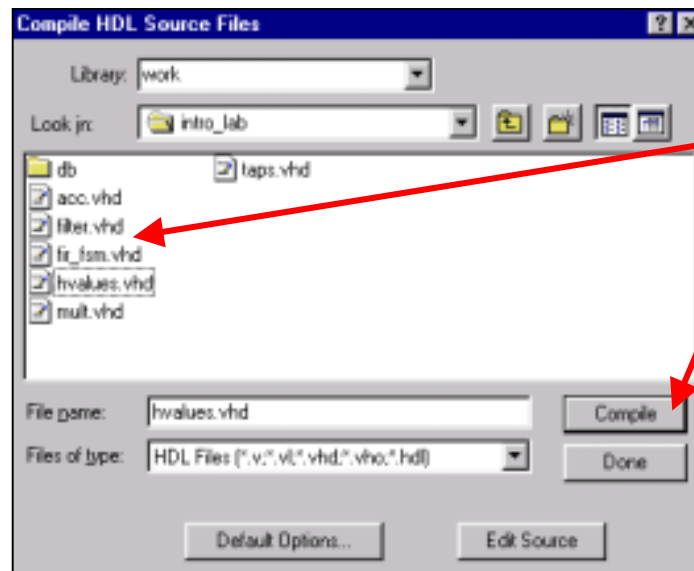
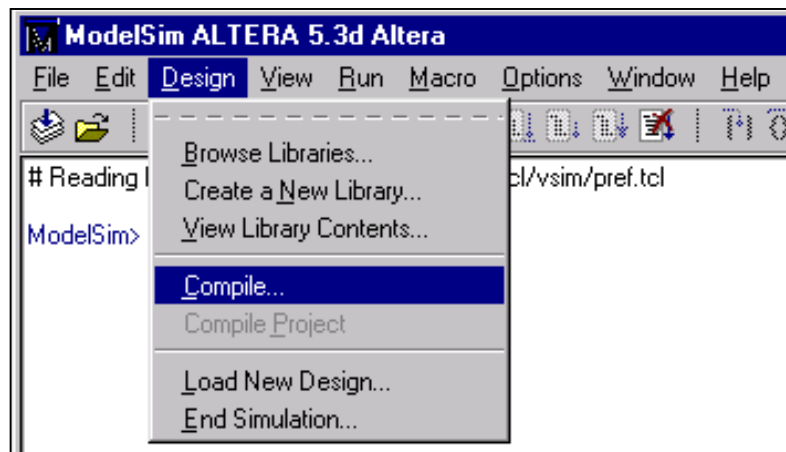
```
[vsim]
; Simulator resolution
; Set to fs, ps, ns, us, ms, or sec with option
Resolution = ns

; User time unit for run commands
; Set to default, fs, ps, ns, us, ms, or sec. T
; unit specified for Resolution. For example, i
; then UserTimeUnit defaults to ps.
UserTimeUnit = default
```

```
;set to Debussy simulation Environment
Veriuser = novas.dll
```

```
; Default run length
RunLength = 100
```

编译 (UI)



点亮一个或多个
文件并点击
Compile