

目录

目录.....	1
前言---必读.....	2
1、Modesim 软件安装	2
1.1 Modelsim 软件版本选择.....	2
1.2 Modelsim 软件安装.....	2
1.3 Modelsim 软件破解.....	4
2、VIVADO 联合 Modelsim 库编译	8
2.1 Vivado 库编译.....	8
方法一、使用 GUI 编译仿真库	8
方法二、使用 TCL 命令编译仿真库.....	11
2.2 设置 Modelsim 安装路径.....	12
2.3 测试.....	14
3、Modelsims 添加 VIVADO 仿真库	16

前言----必读

Modelsim 对于 Vivado 而言是第三方仿真软件，相比较于 Vivado 自带的仿真软件，Modelsim 的执行效率更高（毕竟 Modelsim 是一款高级的仿真软件）。通常情况下使用 Vivado 自带的仿真软件就可以满足需求，使用者可以根据实际需求，选择安装第三方仿真软件。

注意：提供的软件安装仅用于测试使用以及学习交流！不能用于其他任何商业用途！

1、Modesim 软件安装

1.1 Modelsim 软件版本选择


不同的 Vivado 版本支持使用的 Modesim 版本不同，具体可查看 Xilinx 提供的文档 UG973-vivado-release-notes-install-license。

我们使用的 Vivado 软件版本是 vivado2017.4，推荐使用版本是 ModelSim SE/DE/PE (10.6b)，经过安装发现，使用低于推荐的版本，在 Vivado 库编译时会出错，使用高于推荐版本，在 Vivado 库编译时不会出错。这里，我们安装的软件版本是 Modelsim -se-win64-10.6d。

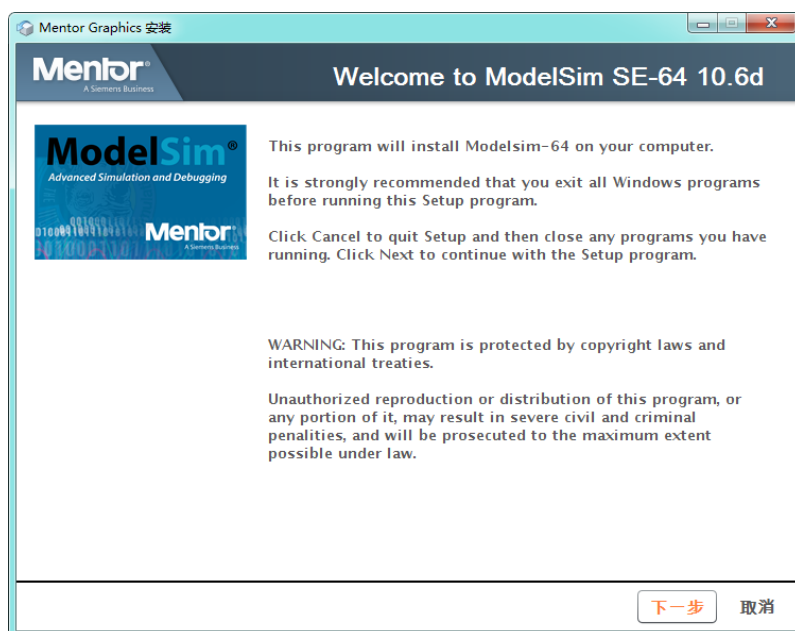
1.2 Modelsim 软件安装

安装软件前，请先关闭杀毒软件。

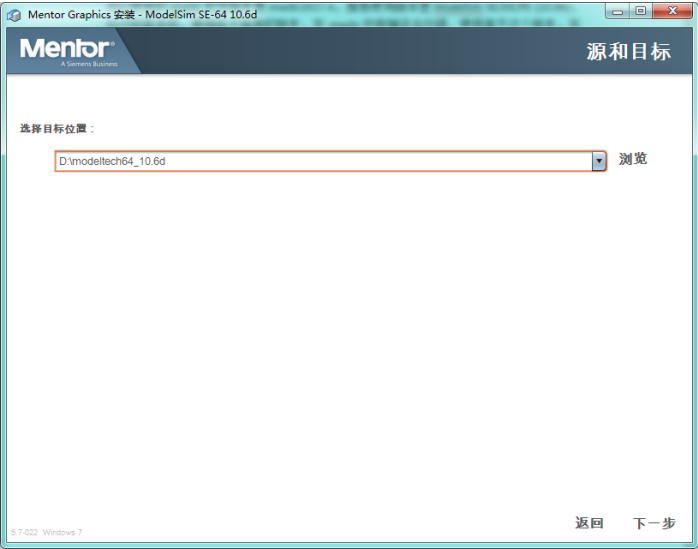
1、解压压缩包，双击 modelsim-win64-10.6d-se.exe 安装。

 modelsim-win64-10.6d-se.exe

2、点击下一步



3、自定义安装路径，这里安装在 D 盘，选择下一步。



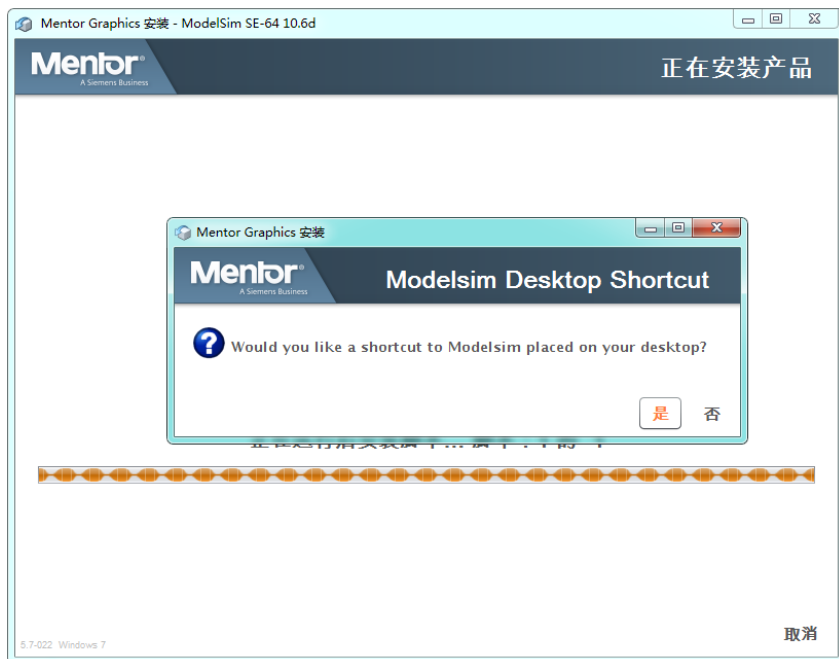
4、创建目标目录，选择是。



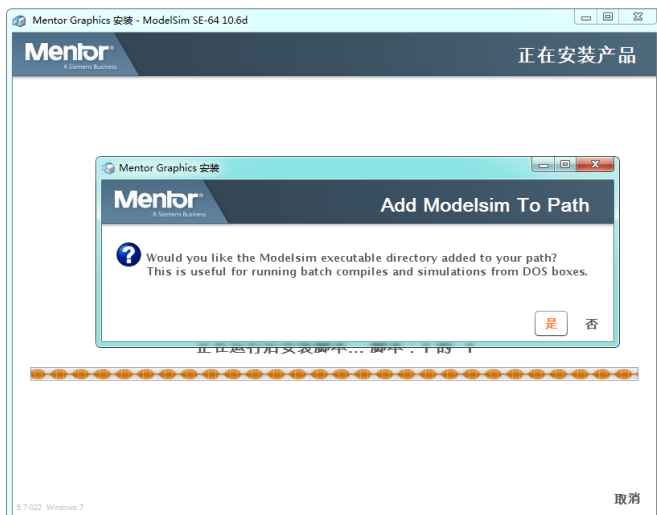
5、选择同意，进行安装。



6、生成桌面快捷方式，选择是。



7、选择是，

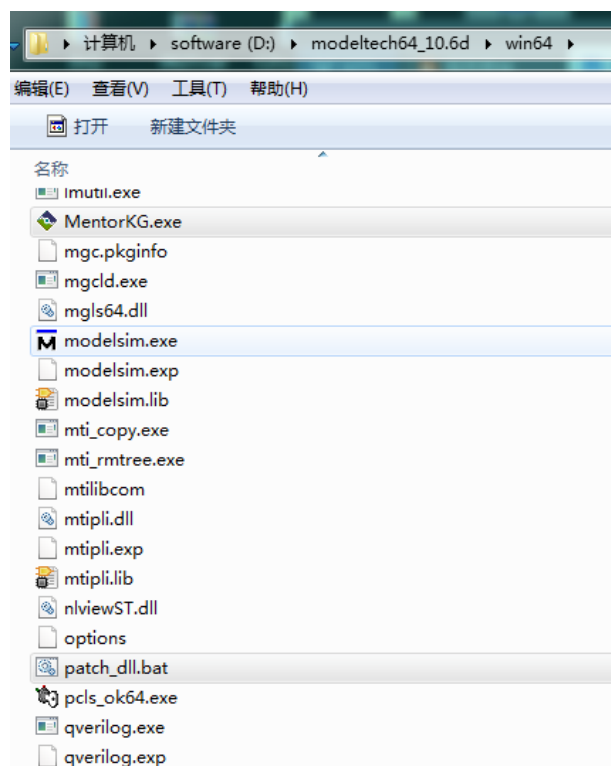


8、出现选择是否重启界面，可以选择重启，也可以不重启。这里选择不重启。

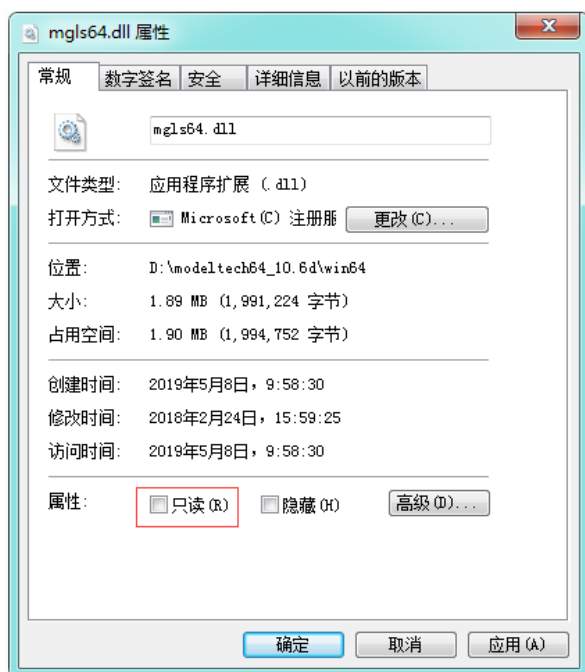
9、点击完成。

1.3 Modelsim 软件破解

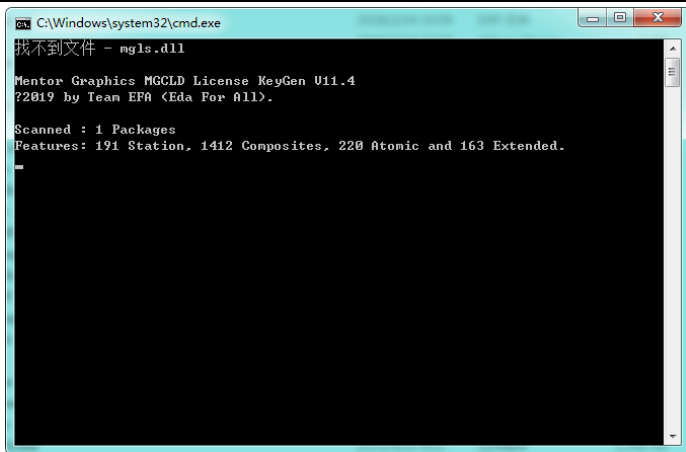
1、将"MentorKG.exe"和"patch_dll.bat"复制到"D:\modeltech64_10.6d\win64 下面。



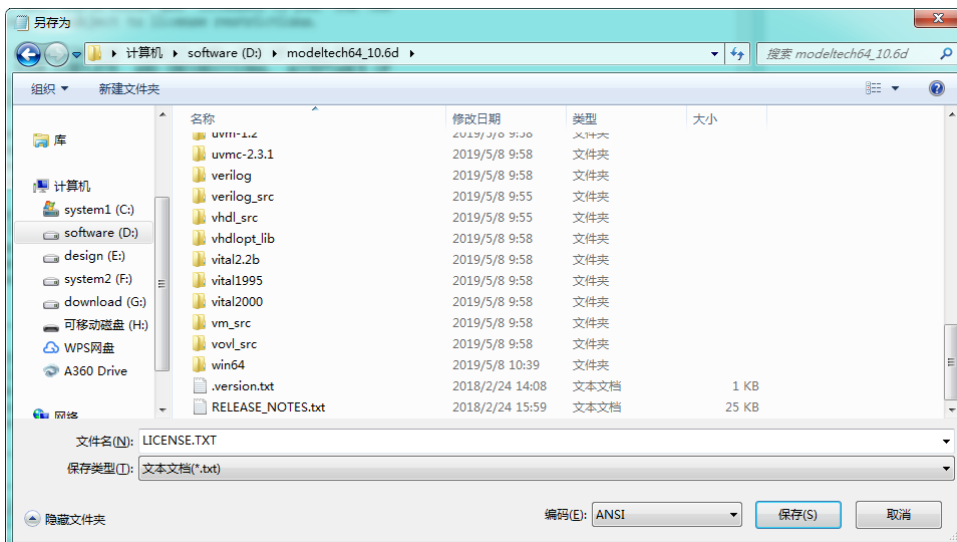
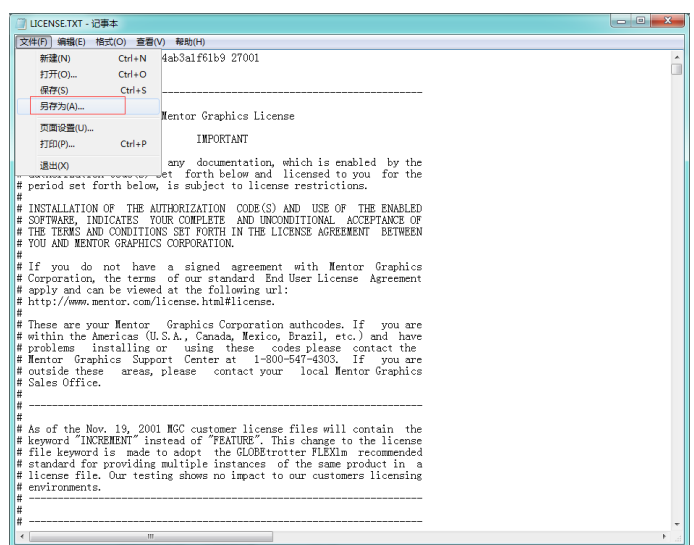
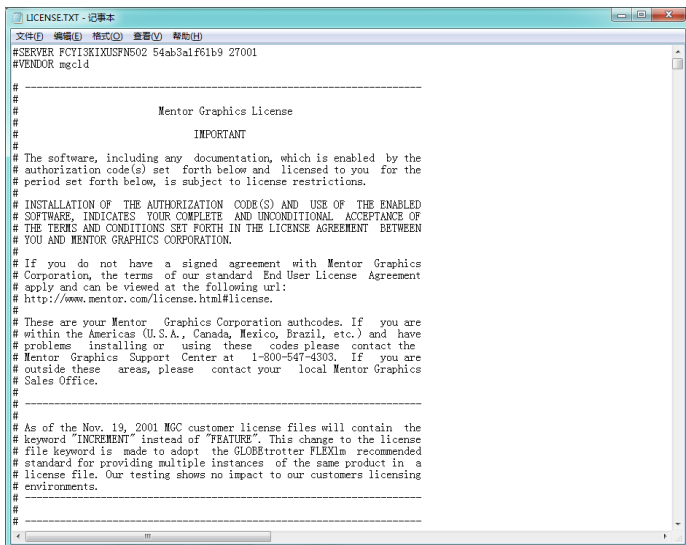
2、去掉文件"D:\modeltech64_10.6d\win64\" (mgls64.dll) 的只读属性。



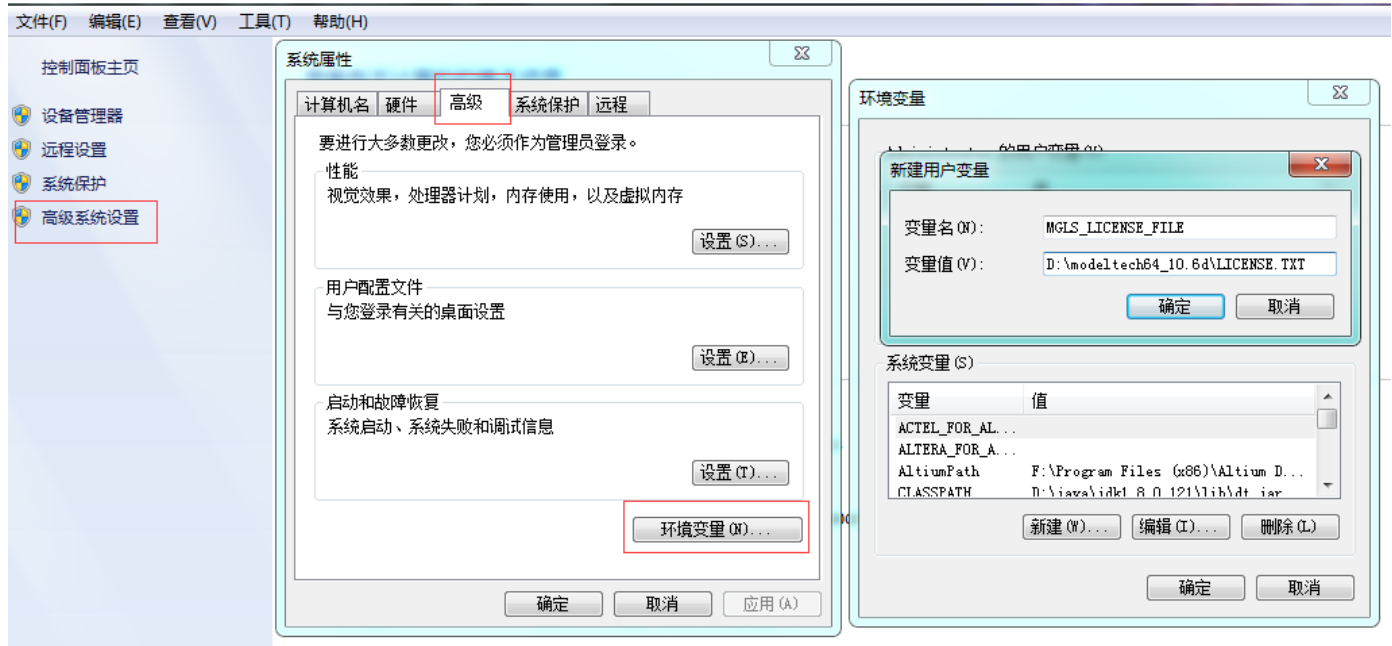
3、以管理员的方式运行 "D:\modeltech64_10.6d\win64\patch_dll.bat"，将产生的文件另存为："D:\modeltech64_10.6d\LICENSE.TXT"。



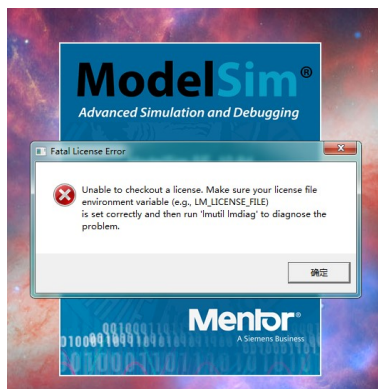
等待一会, 会自动生成 LICENSE.TXT。(这里提示找不到 mgls.dll 文件, 不用管, 破解后软件可以正常使用)



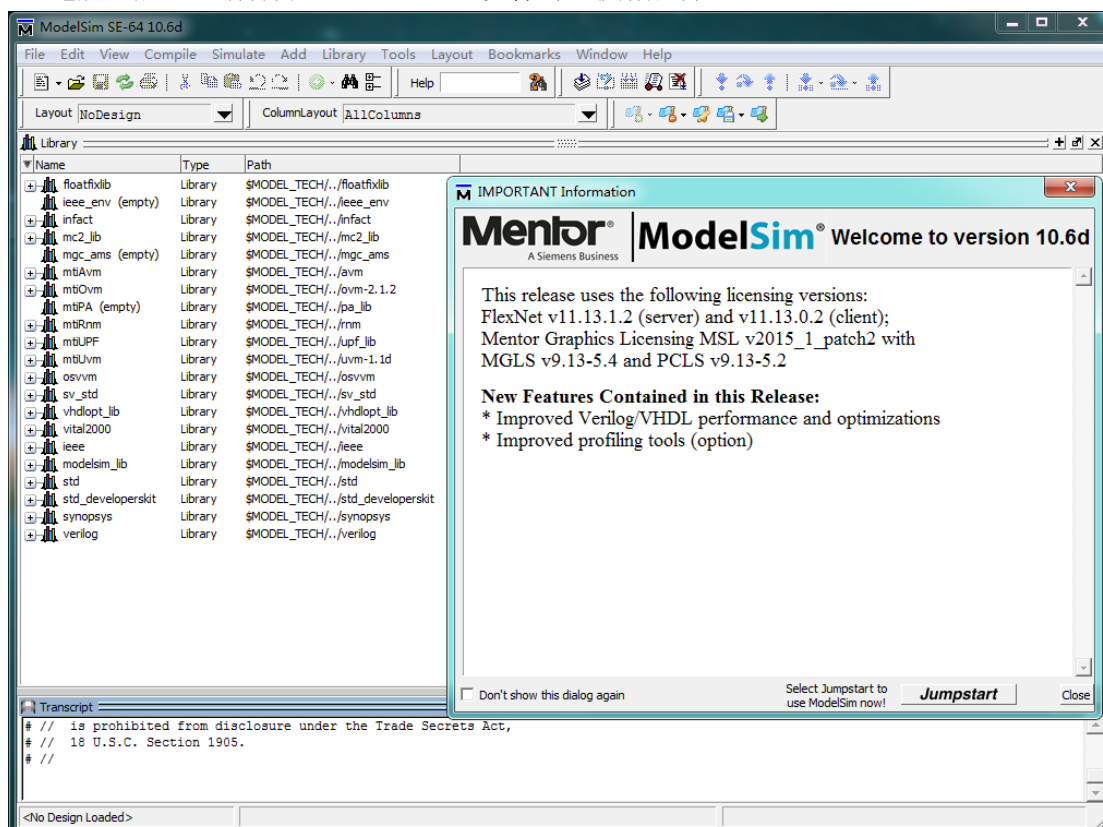
4、新建环境变量 MGLS_LICENSE_FILE 指向"D:\modeltech64_10.6d\LICENSE.TXT".



5、直接打开 Modelsim, 会出现如下情况。



6、电脑重启。重新打开 Modelsim, 可以看到, 破解成功。



2、VIVADO 联合 Modelsim 库编译

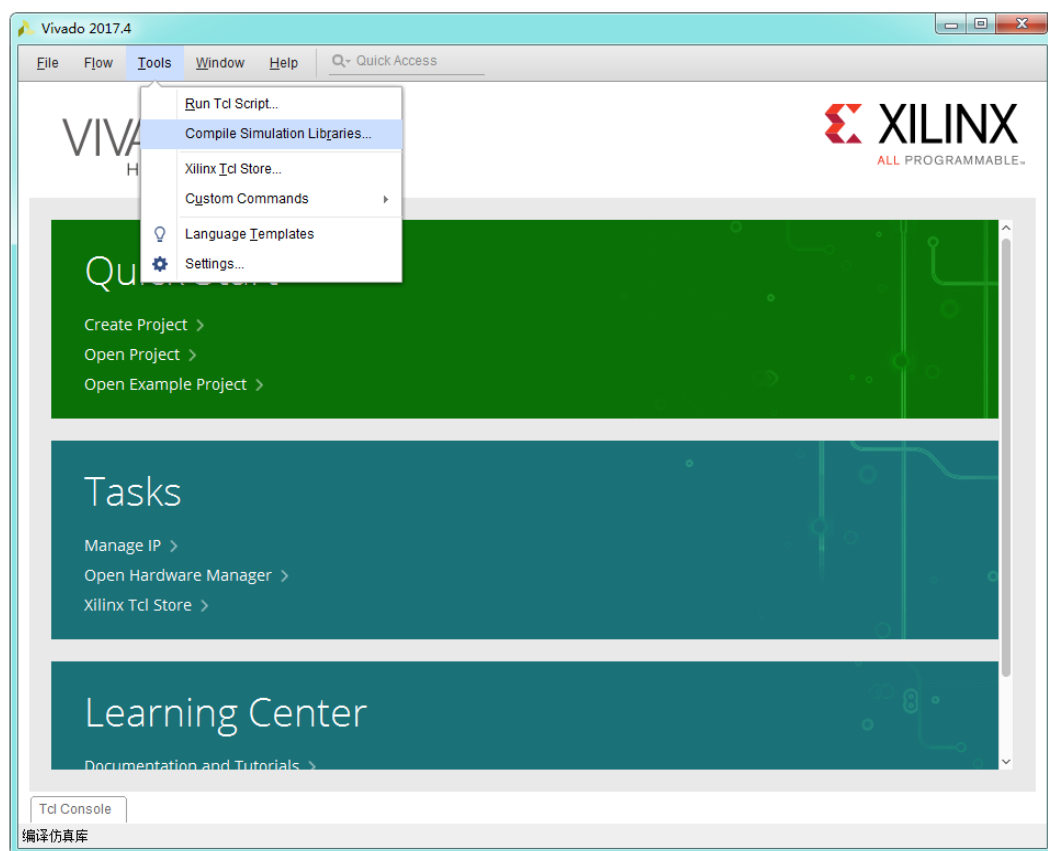
Vivado 要使用 Modelsim 作为第三方软件，需要将两者进行关联。

2.1 Vivado 库编译

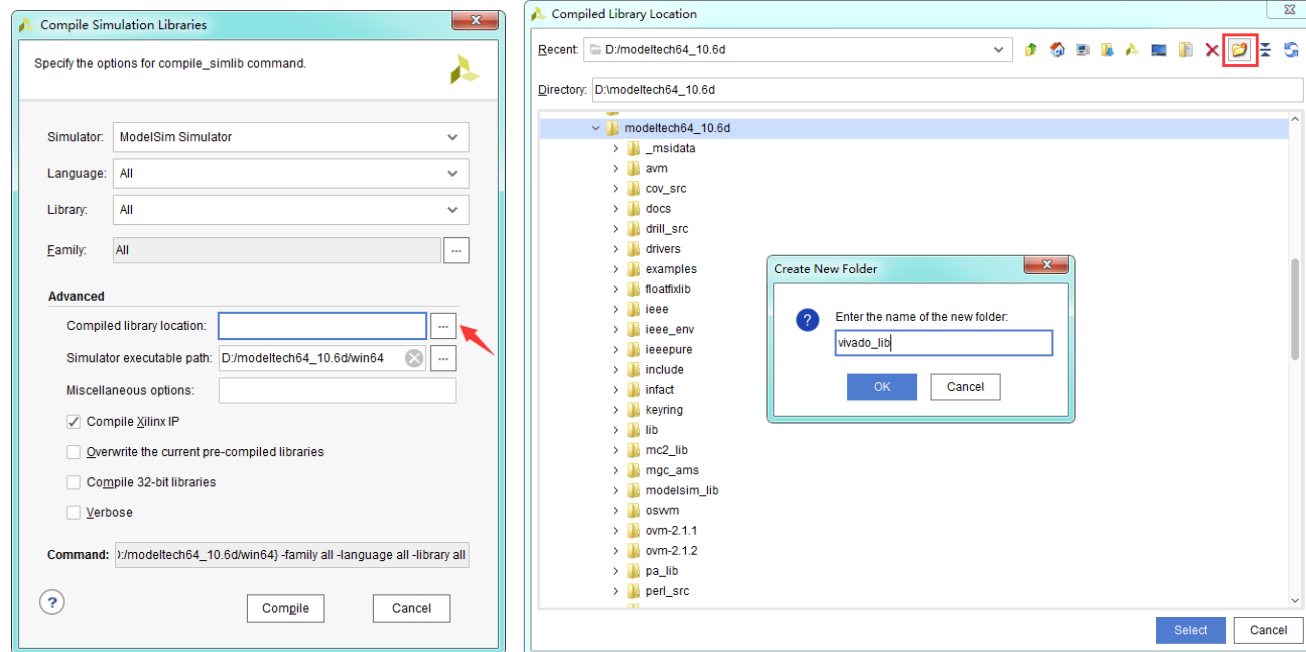
根据 Xilinx 提供的文档 UG900-vivadio-logic-simulation，对 Vivado 进行库编译有两种方法，一种是通过 GUI 编译仿真库，另外一种是通过 TCL 命令编译仿真库。

方法一、使用 GUI 编译仿真库

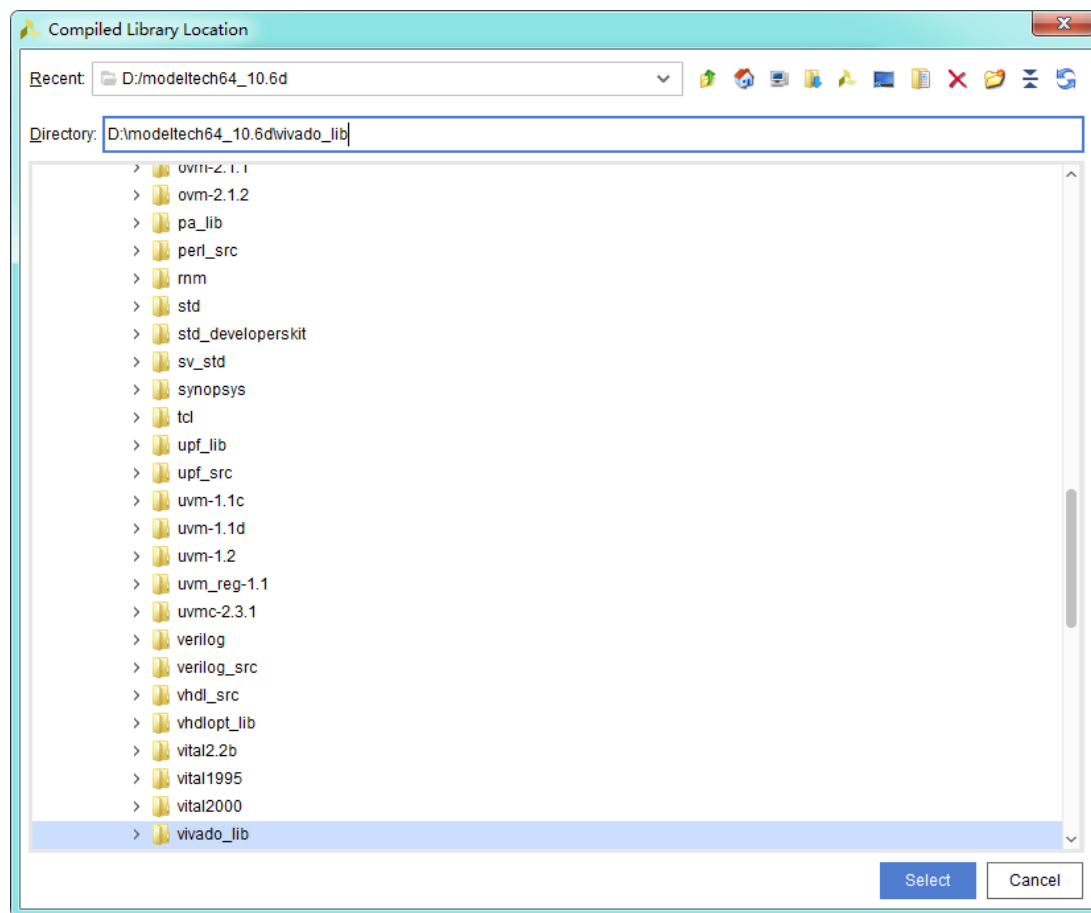
1、在 Vivado 菜单中选择，Tool→Compile Simulation Libraries...



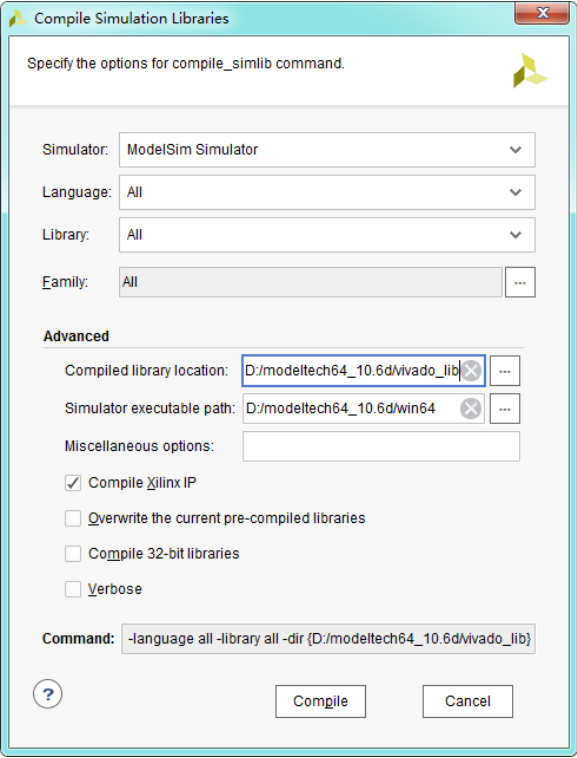
2、启动编译界面如下，在红色箭头指向的位置是默认的仿真库的编译路径，其他设置默认不变。单击红色箭头处，选择放置仿真库的文件夹。这里，我们新建一个文件夹 vivado_lib，用于存放编译的库文件。



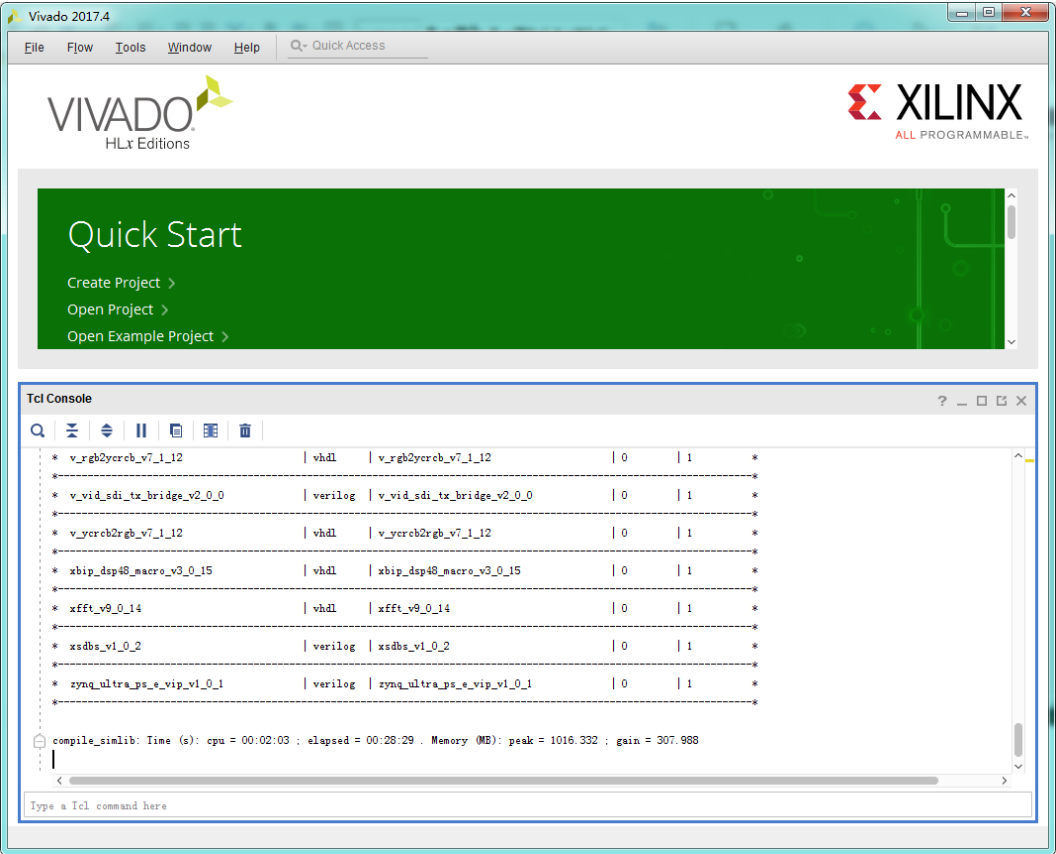
单击 select。



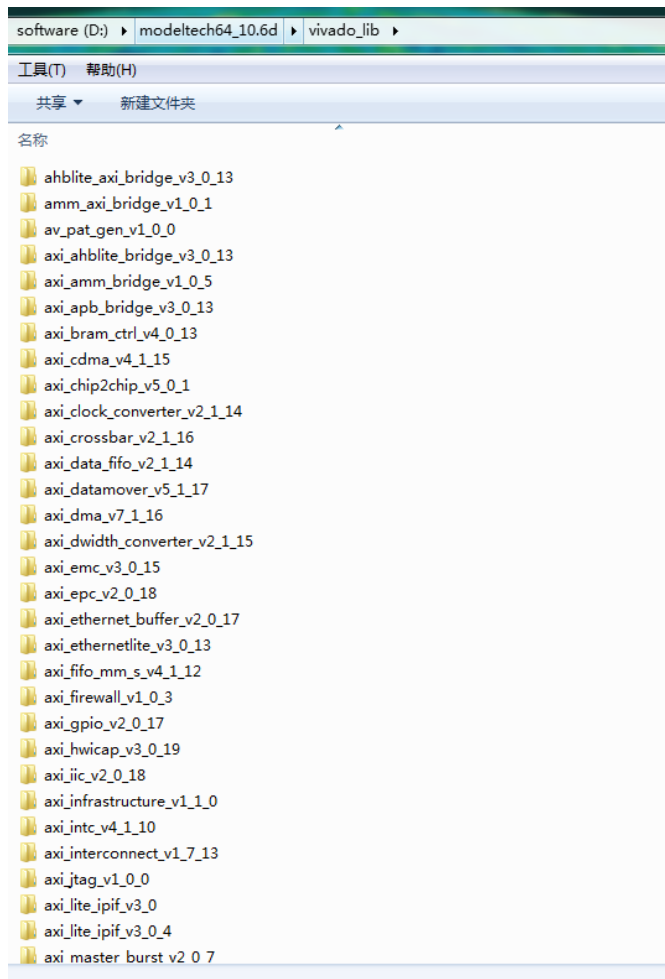
3、单击 compile，开始编译，等待库编译完成。



4、编译完成如图所示。

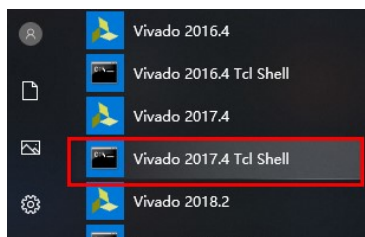


编译后的库文件



方法二、使用 TCL 命令编译仿真库

打开 vivado 2017.4 Tcl Shell.



使用 TCL 脚本：`compile_simlib` 编译仿真库，下面的命令就可以实现（更多内容参考 [ug835](#)）

```
compile_simlib -directory <library_output_directory>-simulator <agr>  
simulator_exec_path<sim install location>
```

例如：a) 仿真库编译到 D:/modeltech64_10.6d/vivado_lib; 仿真工具使用 Modelsim; ModelSim 安装在 D:/modeltech64 10.6d/win64; 那么完整的 tcl 命令就是：

```
compile_simlib -directory D:/modeltech64_10.6d/vivado_lib -simulator
modelsim -simulator exec path D:/modeltech64_10.6d/win64
```

```
Vivado 2017.4 Tcl Shell - E:\Xilinx2017\Vivado\2017.4\bin\vivado.bat -mode tcl

***** Vivado v2017.4 (64-bit)
**** SW Build 2086221 on Fri Dec 15 20:55:39 MST 2017
**** IP Build 2085800 on Fri Dec 15 22:25:07 MST 2017
** Copyright 1986-2017 Xilinx, Inc. All Rights Reserved.

Vivado% compile_simlib -directory D:/modeltech64_10.6d/vivado_lib -simulator modelsim -simulator_exec_path D:/modeltech64_10.6d/win64_
```

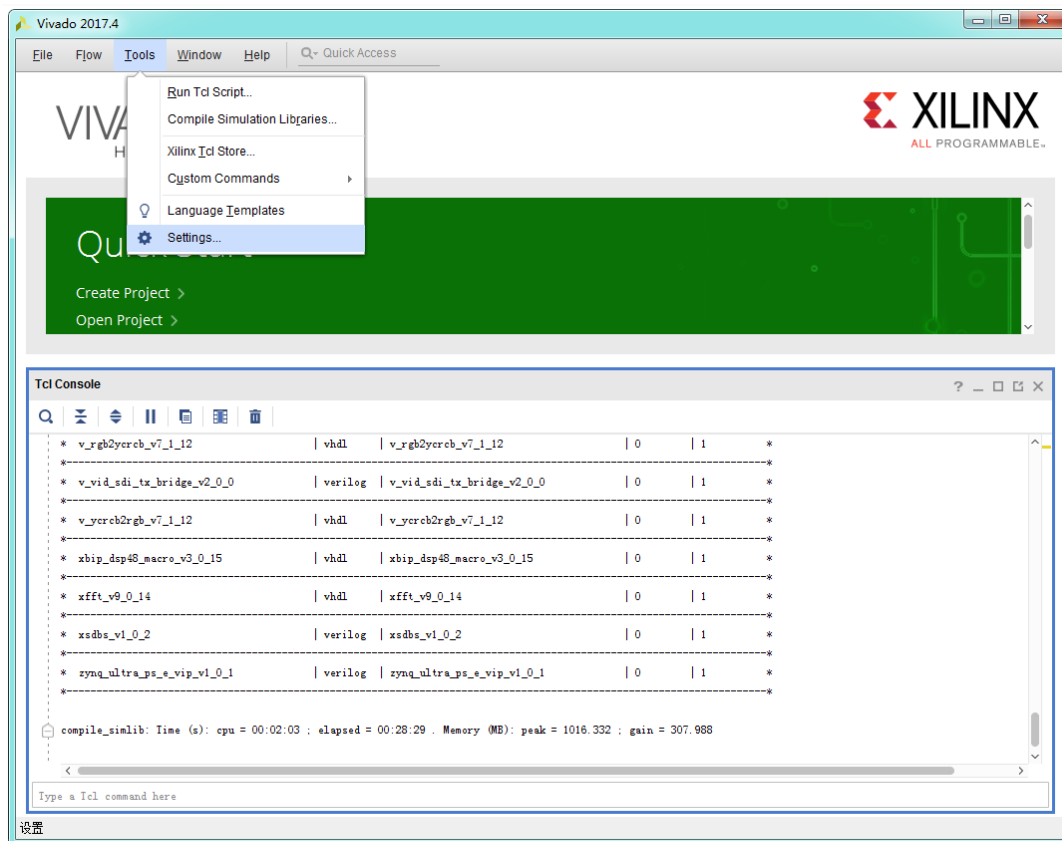
当等待一段时间，编译结束，显示没有库编译错误，编译完成。

```
-----*
* unfast          ! vhdl    ! unfast          ! 0      !
* -----*
* unisim          ! verilog ! unisims_ver    ! 0      !
* -----*
* unimacro        ! verilog ! unimacro_ver    ! 0      !
* -----*
* unfast          ! verilog ! unfast_ver      ! 0      !
* -----*
* simprim         ! verilog ! simprims_ver    ! 0      !
* -----*
* -----*

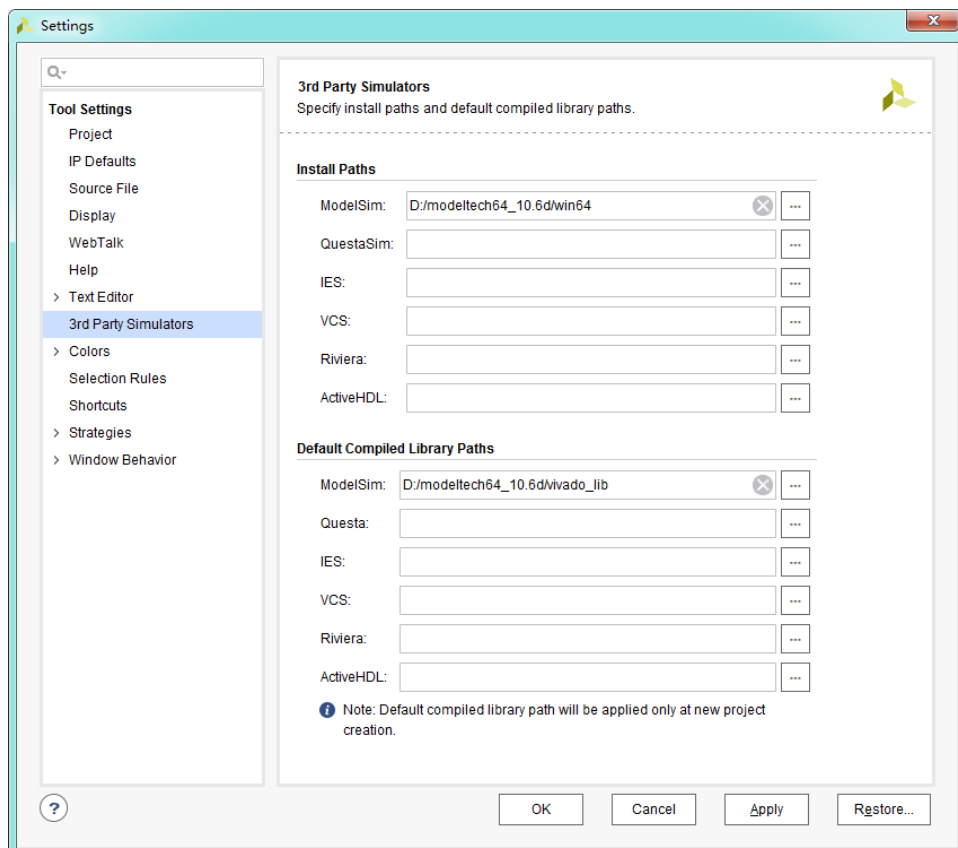
compile_simlib: Time (s): cpu = 00:00:01 ; elapsed = 00:04:02 . Memory (MB): peak = 191.508 ; gain = 2.648
Vivado%
```

2.2 设置 Modelsim 安装路径

1、在 Vivado 菜单中选择，Tool→Setting...



2、选择 3rd Party Simulators，Install Paths 中选择 Modelsim 安装路径，Default Compiled Library Paths 中选择编译库路径。至此，库编译完成。

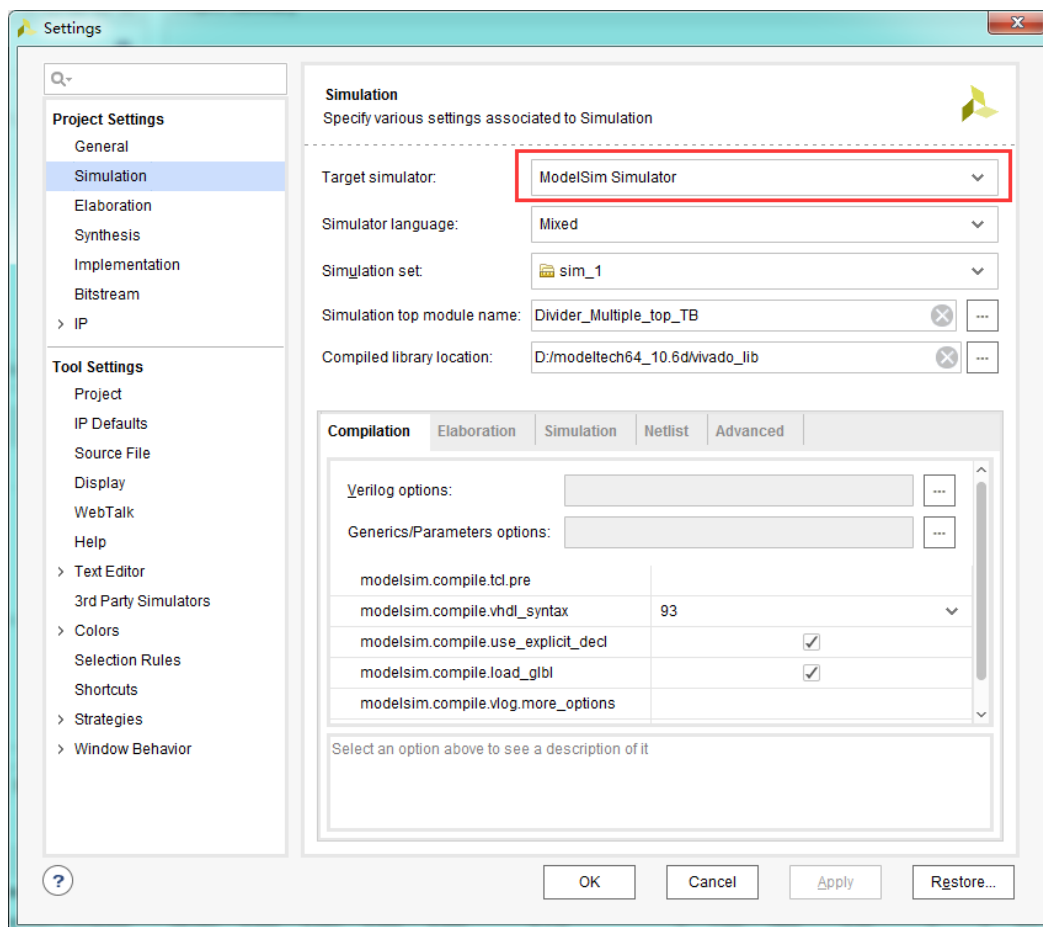


2.3 测试

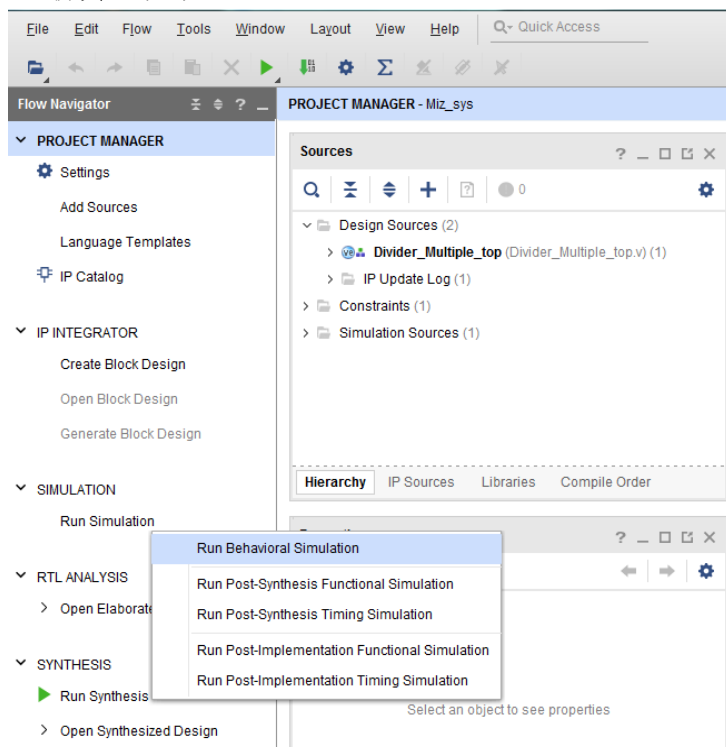
测试可以使用自己写的测试工程，这里不提供测试工程。

1、打开一个写好的测试工程。

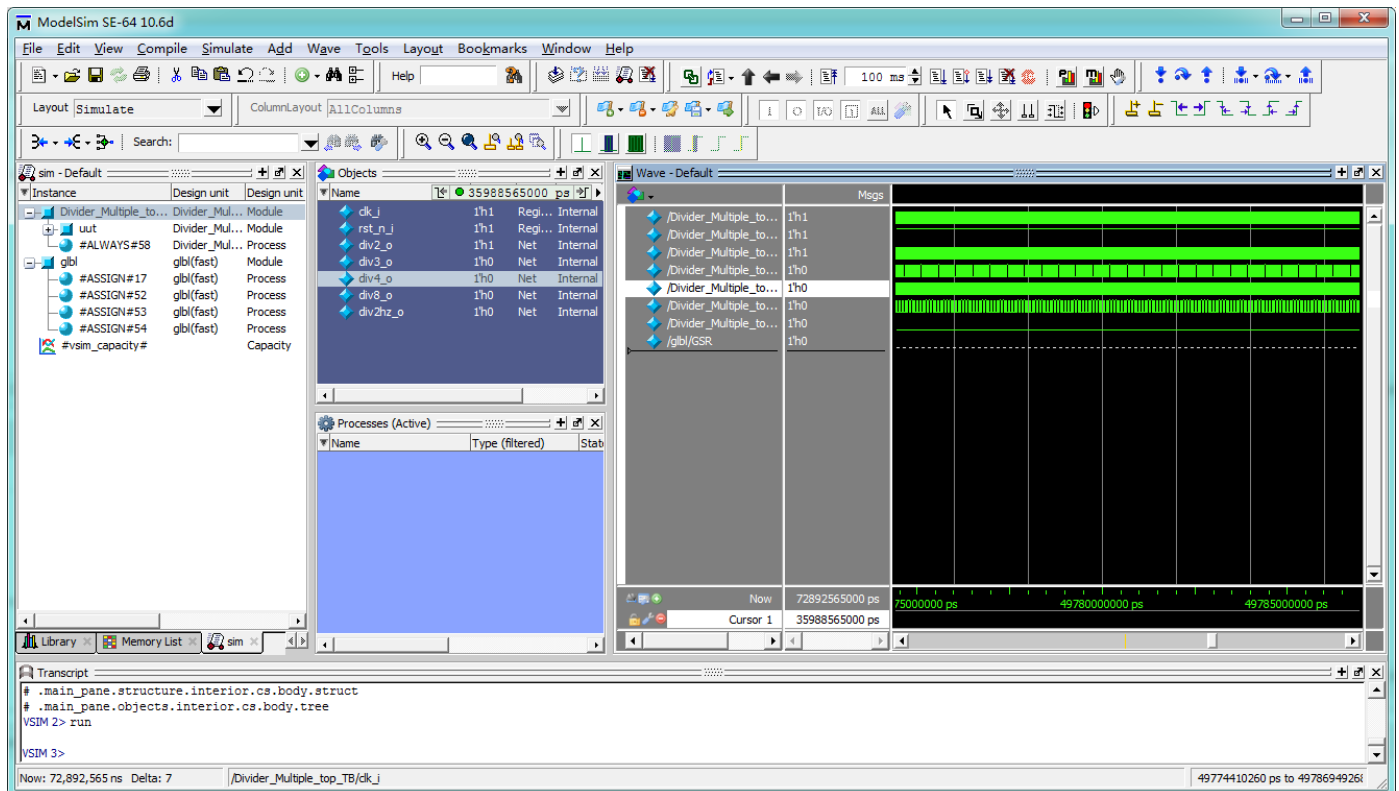
2、单击 settings→Simulation, Target simulator 是 ModelSim Simulator (这里可以选择其他仿真方式)。



3、仿真，单击 Run Simulation→Run Behavioral Simulation.

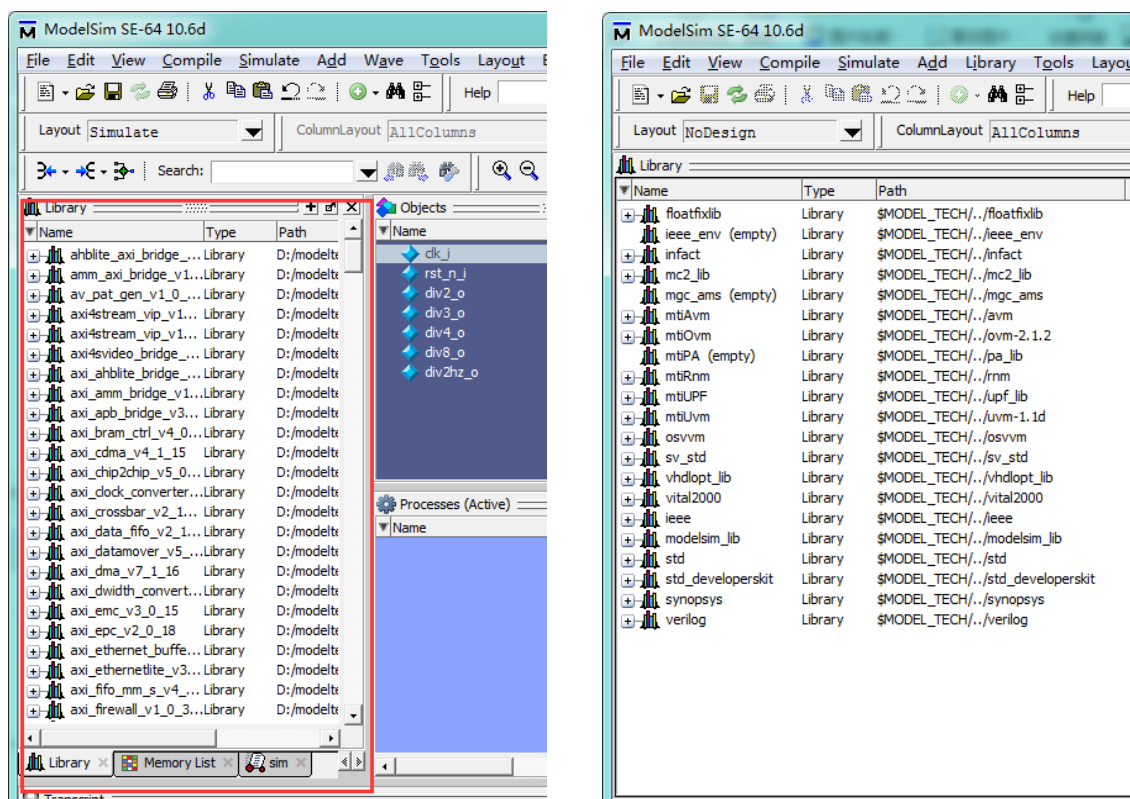


4、查看仿真波形。测试通过。



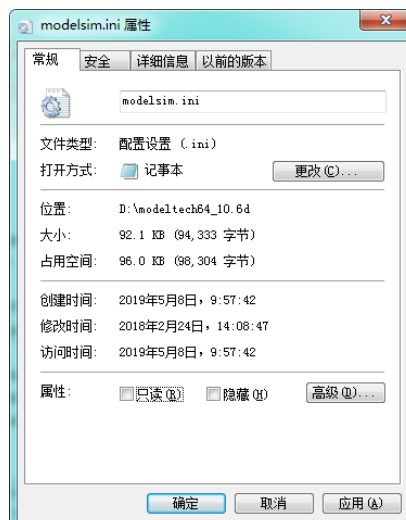
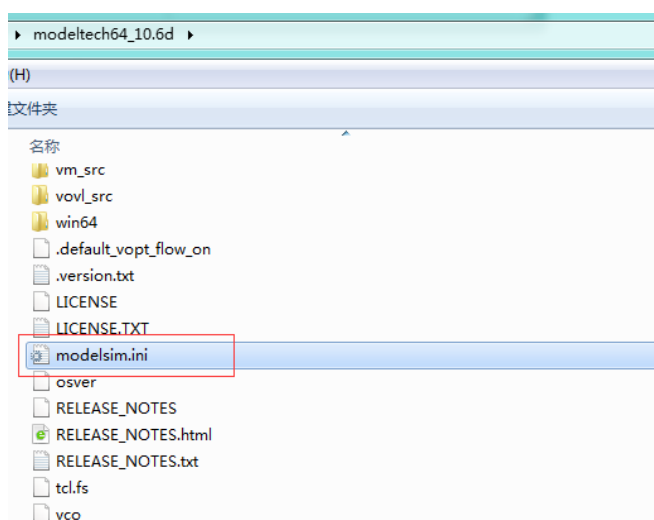
3、Modelsim 添加 VIVADO 仿真库

上面的测试中, 通过 Vivado 打开 Modelsim, 可以发现仿真的时候包含 Vivado 中编译的库。然而, 单独打开 Modelsim, 并不包含这些仿真库。这是因为没有在 Modelsim 添加 VIVADO 仿真库。



在 Modelsim 添加 VIVADO 仿真库不是必须的, 使用者可以不添加, 如果需要单独使用 Modelsim 对 VIVADO 中的涉及的 IP 仿真, 则需要在 Modelsim 中添加 VIVADO 仿真库。

1、在 modelsim 安装路径下有一个 modelsim.ini 文件, 选中, 右键取消其只读属性, 然后用记事本或 notpad++软件将其打开, 找到 “modelsim_lib = \$MODEL_TECH/./modelsim_lib” 处, 准备添加 ip 库路径。



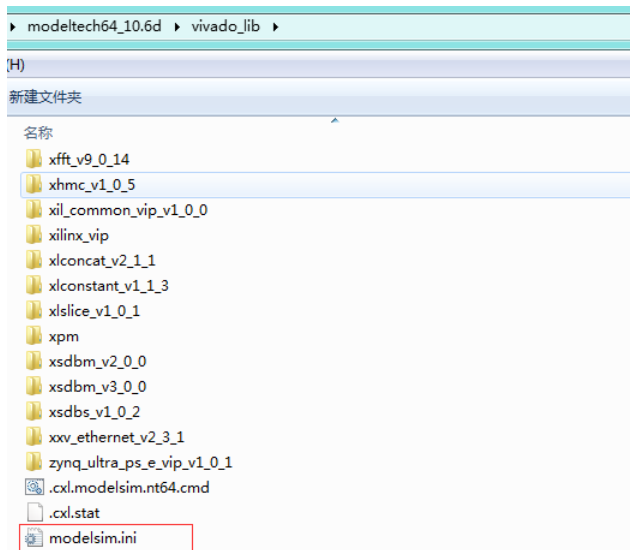

```

;
; For compatibility with previous releases, logical library name vital2000 maps
; to library vital2000 (a different library than library ieee, containing the
; same packages).
; A design should not reference VITAL from both the ieee library and the
; vital2000 library because the vital packages are effectively different.
; A design that references both the ieee and vital2000 libraries must have
; both logical names ieee and vital2000 mapped to the same library, either of
; these:
;   $MODEL_TECH/./ieee
;   $MODEL_TECH/./vital2000
;
verilog = $MODEL_TECH/./verilog
std_developerskit = $MODEL_TECH/./std_developerskit
synopsys = $MODEL_TECH/./synopsys
modelsim_lib = $MODEL_TECH/./modelsim_lib
;

sv_std = $MODEL_TECH/./sv_std
mtiAvm = $MODEL_TECH/./avm
mtiRnm = $MODEL_TECH/./rnm
mtiOvm = $MODEL_TECH/./ovm-2.1.2
mtiUvm = $MODEL_TECH/./uvm-1.1d
mtiUPF = $MODEL_TECH/./upf_lib
mtiPA = $MODEL_TECH/./pa_lib
floatfixlib = $MODEL_TECH/./floatfixlib
mc2_lib = $MODEL_TECH/./mc2_lib
osvnm = $MODEL_TECH/./osvnm

```

2、找到 Vivado 编译 ip 库的文件夹，目录下也会有一个 modelsim.ini 文件，如下图所示：



打开 vivado_lib 中的 modelsim.ini,将下面的内容复制

```

;   $MODEL_TECH/./vital2000
;
; added mapping for ADMS

;vhdl_psl_checkers = $MODEL_TECH/./vhdl_psl_checkers // Source files only for this release
;verilog_psl_checkers = $MODEL_TECH/./verilog_psl_checkers // Source files only for this release
;mvc_lib = $MODEL_TECH/./mvc_lib

; Automatically perform logical->physical mapping for physical libraries that
; appear in -L/-Lf options with filesystem path delimiters (e.g. '.' or '/').
; The tail of the filesystem path name is chosen as the logical library name.
; For example, in the command "vopt -L ./path/to/lib1 -o opttop top",
; vopt automatically performs the mapping "lib1 -> ./path/to/lib1".
; See the User Manual for more details.
;
; AutoLibMapping = 0

secureip = D:/modeltech64_10.6d/vivado_lib/secureip
unisim = D:/modeltech64_10.6d/vivado_lib/unisim
unimacro = D:/modeltech64_10.6d/vivado_lib/unimacro
unifast = D:/modeltech64_10.6d/vivado_lib/unifast
unisims_ver = D:/modeltech64_10.6d/vivado_lib/unisims_ver
unimacro_ver = D:/modeltech64_10.6d/vivado_lib/unimacro_ver
unifast_ver = D:/modeltech64_10.6d/vivado_lib/unifast_ver
simprims_ver = D:/modeltech64_10.6d/vivado_lib/simprims_ver
xpm = D:/modeltech64_10.6d/vivado_lib/xpm
xilinx_vip = D:/modeltech64_10.6d/vivado_lib/xilinx_vip
ahblite_axi_bridge_v3_0_13 = D:/modeltech64_10.6d/vivado_lib/ahblite_axi_bridge_v3_0_13
av_pat_gen_v1_0_0 = D:/modeltech64_10.6d/vivado_lib/av_pat_gen_v1_0_0
axis_infrastructure_v1_1_0 = D:/modeltech64_10.6d/vivado_lib/axis_infrastructure_v1_1_0
axis_protocol_checker_v1_1_15 = D:/modeltech64_10.6d/vivado_lib/axis_protocol_checker_v1_1_15
axis_protocol_checker_v1_2_1 = D:/modeltech64_10.6d/vivado_lib/axis_protocol_checker_v1_2_1
axi_ahblite_bridge_v3_0_13 = D:/modeltech64_10.6d/vivado_lib/axi_ahblite_bridge_v3_0_13
axi_amm_bridge_v1_0_5 = D:/modeltech64_10.6d/vivado_lib/axi_amm_bridge_v1_0_5
axi_chip2chip_v5_0_1 = D:/modeltech64_10.6d/vivado_lib/axi_chip2chip_v5_0_1
axi_infrastructure_v1_1_0 = D:/modeltech64_10.6d/vivado_lib/axi_infrastructure_v1_1_0

```

```

...
videoaxis4s_bridge_v1_0_9 = D:/modeltech64_10.6d/vivado_lib/videoaxis4s_bridge_v1_0_9
viterbi_v9_1_8 = D:/modeltech64_10.6d/vivado_lib/viterbi_v9_1_8
v_ccm_v6_0_14 = D:/modeltech64_10.6d/vivado_lib/v_ccm_v6_0_14
v_cfa_v7_0_13 = D:/modeltech64_10.6d/vivado_lib/v_cfa_v7_0_13
v_cresample_v4_0_13 = D:/modeltech64_10.6d/vivado_lib/v_cresample_v4_0_13
v_dual_splitter_v1_0_8 = D:/modeltech64_10.6d/vivado_lib/v_dual_splitter_v1_0_8
v_enhance_v8_0_14 = D:/modeltech64_10.6d/vivado_lib/v_enhance_v8_0_14
v_gamma_v7_0_14 = D:/modeltech64_10.6d/vivado_lib/v_gamma_v7_0_14
v_osd_v6_0_15 = D:/modeltech64_10.6d/vivado_lib/v_osd_v6_0_15
v_rgb2ycrcb_v7_1_12 = D:/modeltech64_10.6d/vivado_lib/v_rgb2ycrcb_v7_1_12
v_vid_sdi_tx_bridge_v2_0_0 = D:/modeltech64_10.6d/vivado_lib/v_vid_sdi_tx_bridge_v2_0_0
v_ycrcb2rgb_v7_1_12 = D:/modeltech64_10.6d/vivado_lib/v_ycrcb2rgb_v7_1_12
xbip_dsp48_macro_v3_0_15 = D:/modeltech64_10.6d/vivado_lib/xbip_dsp48_macro_v3_0_15
xfft_v9_0_14 = D:/modeltech64_10.6d/vivado_lib/xfft_v9_0_14
xsdb_v1_0_2 = D:/modeltech64_10.6d/vivado_lib/xsdb_v1_0_2
zynq_ultra_ps_e_vip_v1_0_1 = D:/modeltech64_10.6d/vivado_lib/zynq_ultra_ps_e_vip_v1_0_1
;[DefineOptionset]
; Define optionset entries for the various compilers, vmake, and vsim.
; These option sets can be used with the "-optionset <optionsetname>" syntax.
; i.e.

```

3、粘贴到 modelsim 安装路径下的 modelsim.ini 中,并保存。

```

; For compatibility with previous releases, logical library name vital2000 maps
; to library vital2000 (a different library than library ieee, containing the
; same packages).
; A design should not reference VITAL from both the ieee library and the
; vital2000 library because the vital packages are effectively different.
; A design that references both the ieee and vital2000 libraries must have
; both logical names ieee and vital2000 mapped to the same library, either of
; these:
; $MODEL_TECH/./ieee
; $MODEL_TECH/./vital2000
;
verilog = $MODEL_TECH/./verilog
std_developerskit = $MODEL_TECH/./std_developerskit
synopsys = $MODEL_TECH/./synopsys
modelsim_lib = $MODEL_TECH/./modelsim_lib
secureip = D:/modeltech64_10.6d/vivado_lib/secureip
unisim = D:/modeltech64_10.6d/vivado_lib/unisim
unimacro = D:/modeltech64_10.6d/vivado_lib/unimacro
unifast = D:/modeltech64_10.6d/vivado_lib/unifast
unisims_ver = D:/modeltech64_10.6d/vivado_lib/unisims_ver
unimacro_ver = D:/modeltech64_10.6d/vivado_lib/unimacro_ver
unifast_ver = D:/modeltech64_10.6d/vivado_lib/unifast_ver
simprims_ver = D:/modeltech64_10.6d/vivado_lib/simprims_ver
xpm = D:/modeltech64_10.6d/vivado_lib/xpm
xilinx_vip = D:/modeltech64_10.6d/vivado_lib/xilinx_vip
ahblite_axi_bridge_v3_0_13 = D:/modeltech64_10.6d/vivado_lib/ahblite_axi_bridge_v3_0_13
av_pat_gen_v1_0_0 = D:/modeltech64_10.6d/vivado_lib/av_pat_gen_v1_0_0
axis_infrastructure_v1_1_0 = D:/modeltech64_10.6d/vivado_lib/axis_infrastructure_v1_1_0
axis_protocol_checker_v1_1_15 = D:/modeltech64_10.6d/vivado_lib/axis_protocol_checker_v1_1_15
axis_protocol_checker_v1_2_1 = D:/modeltech64_10.6d/vivado_lib/axis_protocol_checker_v1_2_1
axi_ahblite_bridge_v3_0_13 = D:/modeltech64_10.6d/vivado_lib/axi_ahblite_bridge_v3_0_13
axi_amm_bridge_v1_0_5 = D:/modeltech64_10.6d/vivado_lib/axi_amm_bridge_v1_0_5
axi_chip2chip_v5_0_1 = D:/modeltech64_10.6d/vivado_lib/axi_chip2chip_v5_0_1
axi_infrastructure_v1_1_0 = D:/modeltech64_10.6d/vivado_lib/axi_infrastructure_v1_1_0

```

...

```
spdif_v2_0_18 = D:/modeltech64_10.6d/vivado_lib/spdif_v2_0_18
srio_gen2_v4_1_2 = D:/modeltech64_10.6d/vivado_lib/srio_gen2_v4_1_2
switch_core_top_v1_0_4 = D:/modeltech64_10.6d/vivado_lib/switch_core_top_v1_0_4
tcc_decoder_3gppmm_v2_0_15 = D:/modeltech64_10.6d/vivado_lib/tcc_decoder_3gppmm_v2_0_15
tcc_encoder_3gpp_v5_0_12 = D:/modeltech64_10.6d/vivado_lib/tcc_encoder_3gpp_v5_0_12
tmr_comparator_v1_0_1 = D:/modeltech64_10.6d/vivado_lib/tmr_comparator_v1_0_1
tmr_sem_v1_0_3 = D:/modeltech64_10.6d/vivado_lib/tmr_sem_v1_0_3
tri_mode_ethernet_mac_v9_0_10 = D:/modeltech64_10.6d/vivado_lib/tri_mode_ethernet_mac_v9_0_10
tsn_temac_v1_0_2 = D:/modeltech64_10.6d/vivado_lib/tsn_temac_v1_0_2
videoaxi4s_bridge_v1_0_5 = D:/modeltech64_10.6d/vivado_lib/videoaxi4s_bridge_v1_0_5
viterbi_v9_1_8 = D:/modeltech64_10.6d/vivado_lib/viterbi_v9_1_8
v_ccm_v6_0_14 = D:/modeltech64_10.6d/vivado_lib/v_ccm_v6_0_14
v_cfa_v7_0_13 = D:/modeltech64_10.6d/vivado_lib/v_cfa_v7_0_13
v_cresample_v4_0_13 = D:/modeltech64_10.6d/vivado_lib/v_cresample_v4_0_13
v_dual_splitter_v1_0_8 = D:/modeltech64_10.6d/vivado_lib/v_dual_splitter_v1_0_8
v_enhance_v8_0_14 = D:/modeltech64_10.6d/vivado_lib/v_enhance_v8_0_14
v_gamma_v7_0_14 = D:/modeltech64_10.6d/vivado_lib/v_gamma_v7_0_14
v_osd_v6_0_15 = D:/modeltech64_10.6d/vivado_lib/v_osd_v6_0_15
v_rgb2ycrcb_v7_1_12 = D:/modeltech64_10.6d/vivado_lib/v_rgb2ycrcb_v7_1_12
v_vid_sdi_tx_bridge_v2_0_0 = D:/modeltech64_10.6d/vivado_lib/v_vid_sdi_tx_bridge_v2_0_0
v_ycrcb2rgb_v7_1_12 = D:/modeltech64_10.6d/vivado_lib/v_ycrcb2rgb_v7_1_12
xbip_dsp48_macro_v3_0_15 = D:/modeltech64_10.6d/vivado_lib/xbip_dsp48_macro_v3_0_15
xfft_v9_0_14 = D:/modeltech64_10.6d/vivado_lib/xfft_v9_0_14
xsdb_v1_0_2 = D:/modeltech64_10.6d/vivado_lib/xsdb_v1_0_2
zynq_ultra_ps_e_vip_v1_0_1 = D:/modeltech64_10.6d/vivado_lib/zynq_ultra_ps_e_vip_v1_0_1

sv_std = $MODEL_TECH/../sv_std
mtiAvM = $MODEL_TECH/../avm
mtiRnm = $MODEL_TECH/../rnm
mtiOvm = $MODEL_TECH/../ovm-2.1.2
mtiUvm = $MODEL_TECH/../uvm-1.1d
mtiUPF = $MODEL_TECH/../upf_lib
mtiPA = $MODEL_TECH/../pa_lib
floatfixlib = $MODEL_TECH/../floatfixlib
mc2_lib = $MODEL_TECH/../mc2_lib
osvmm = $MODEL_TECH/../osvmm

; added mapping for ADMS
```

4、重新打开 modesim，库都加进来了。

