



E 课网 — 半导体集成电路在线教育平台

半导体集成电路教育内容和产品提供商



VCS 实验手册



VCS 使用手册（adder 为例）

实验简介：

通过这次实验，熟悉使用 VCS 工具编译仿真，理解逻辑仿真技术，同时理解构成一个验证平台（以下称为 Testbench）最基本的组件 – 激励生成、响应获取和结果比对。

实验目的：

1. 熟悉使用 VCS 工具的使用方法，包括 GUI 界面
2. 熟悉使用 Makefile 脚本使用 VCS 工具进行编译和仿真
3. 对一个典型的基于 Verilog 的 Testbench 有一个大致的感受：

实验准备：

- 使用 VNC 登录服务器，进入你的工作目录中，在 terminal 中键入：
`cd ~/verification/basic/labs/Verilog/adder32`
- 如果在上述准备工作中遇到任何困难，请及时与讲师联系解决。

实验步骤：

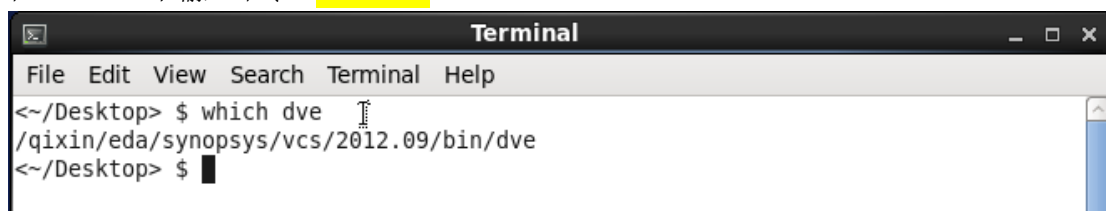
1. 熟悉数字逻辑仿真工具: VCS

a) VCS 简介：

VCS 是 Synopsys 开发的 EDA 仿真工具，可提供高性能仿真引擎、约束条件解算器引擎、Native Testbench (NTB) 支持、广泛的 SystemVerilog 支持、验证规划、覆盖率分析和收敛以及完整的调试环境

b) 确认 Linux 系统环境中的 VCS 可用：

在 Terminal 中输入命令：`which dve`



```
Terminal
File Edit View Search Terminal Help
<~/Desktop> $ which dve
/qixin/eda/synopsys/vcs/2012.09/bin/dve
<~/Desktop> $
```

如果返回上图信息，则说明 VCS 可用，上图为安装路径。如果没有，说明 VCS 不可用，请检查系统配置是否正确，或者联系讲师或助教，及时获得帮助。

c) 编译生产 **simv** 文件：

在 Terminal 中输入命令：

```
vcs -sverilog -debug_all timescale.v full_adder.v full_adder_tb.v -l com.log
```



```
vcs -sverilog -debug_all timescale.v full_adder.v full_adder_tb.v -l com.log
Chronologic VCS (TM)
Version G-2012.09 -- Sun Jun 24 21:50:54 2018
Copyright (c) 1991-2012 by Synopsys Inc.
ALL RIGHTS RESERVED

This program is proprietary and confidential information of Synopsys Inc.
and may be used and disclosed only as authorized in a license agreement
controlling such use and disclosure.

Parsing design file 'timescale.v'
Parsing design file 'full_adder.v'
Parsing design file 'full_adder_tb.v'
Top Level Modules:
    full_adder_tb
TimeScale is 1 ns / 10 ps
Starting vcs inline pass...
2 modules and 0 UDP read.
recompiling module full_adder_tb
Both modules done.
make[1]: Entering directory `/qixin/proj_users/klin/asic_design/sim_syn_lab/
if [ -x ../simv ]; then chmod -x ../simv; fi
g++ -o ../simv -melf_i386 -m32 -WL,-whole-archive -WL,-no-whole-archiv
s_mop.o rmapats.o /qixin/eda/synopsys/vcs/2012.09/linux/lib/libnplex.st
im.so /qixin/eda/synopsys/vcs/2012.09/linux/lib/librterrorinf.so /qixin/eda/s
in/eda/synopsys/vcs/2012.09/linux/lib/libvcsnew.so /qixin/eda/synopsys/vcs/20
opsys/vcs/2012.09/linux/lib/vcs_save_restore_new.o /qixin/eda/synopsys/vcs/20
d -ldl
../simv up to date
```

执行完 VCS 的编译命令之后，在本目录下生成可执行过文件 **simv**。

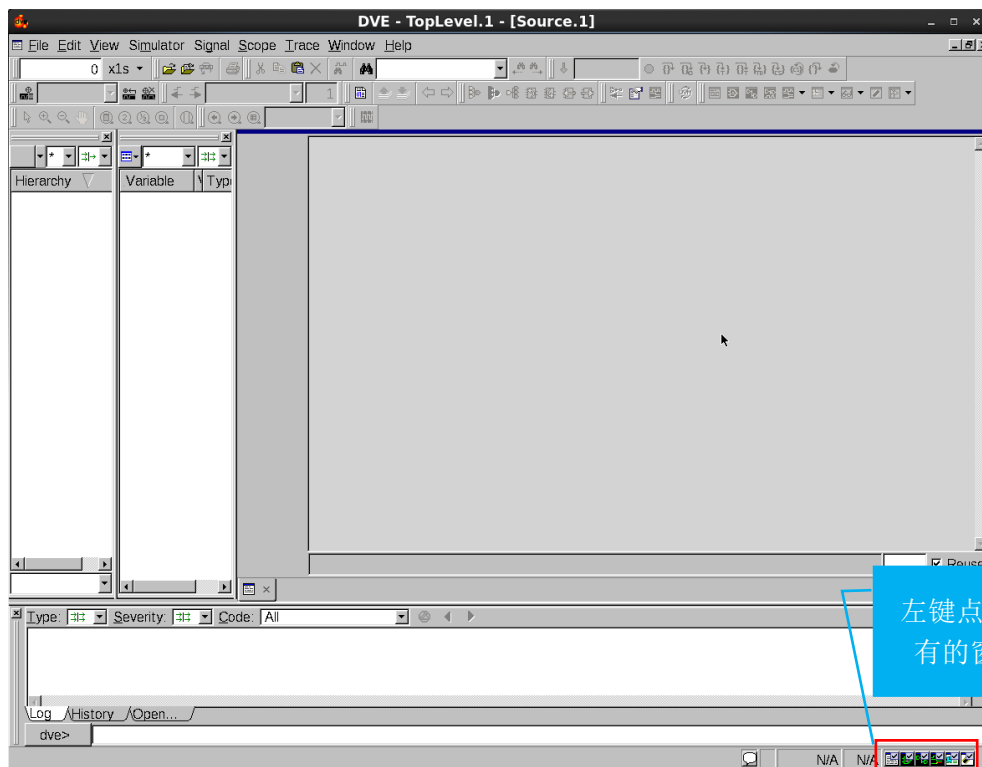
- d) 打开 vcs GUI 界面: (dve 命令是 VCS 仿真工具启动的图形化界面)

在 Terminal 中输入命令:

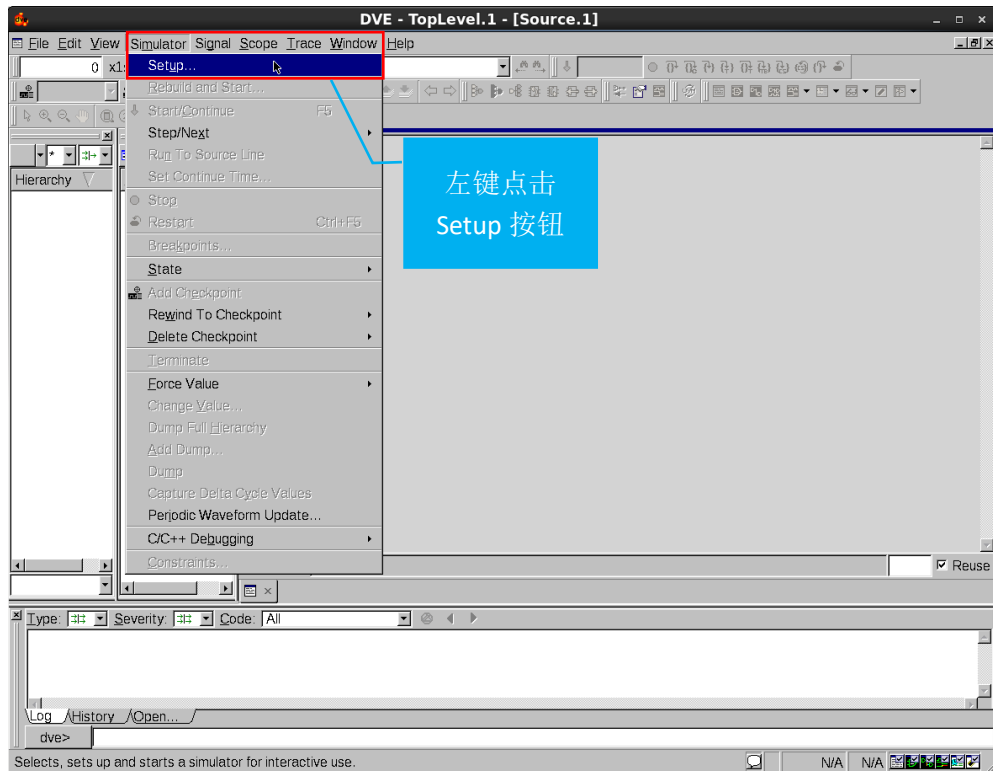
dve & (&是为了让 DVE 在后台运行, Terminal 还能输入命令)

```
$ dve &
```

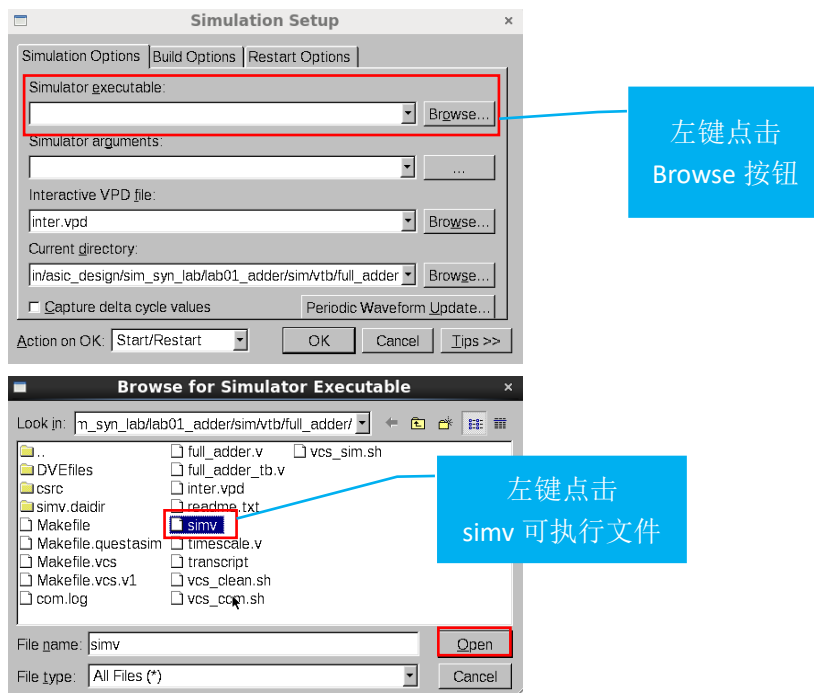
显示如下窗口, 并点选红色方框内的所有窗口。



- e) 载入可执行文件 **simv** 与 **VPD** 文件, setup 一个 simulation



在弹出的对话框中，输入 simv 可执行文件以及 vpd 文件，点 OK



f) 选中需要查看波形的信号，将信号在波形中显示



左键点击加減号“+/-”；
右键选择
full_adder_tb，然后依次
左键点击“Add To
Waves”，“New Wave

RTL 源代码

```

module full_adder tb;

    reg ain, bin, cin; // drive the input port with the reg
    type
    wire sumout, cout; // sample the output port with the w
    ire type

    full_adder u_full_adder(
        // task 1. how to create an instance
        // module head: verillog-2001 format
        /*input wire */ .a_in (ain),
        /*input wire */ .b_in (bin),
        /*input wire */ .c_in (cin), // carry in
        /*output wire */ .sum_out(sumout),
        /*output wire */ .c_out (cout) // carry out
    );

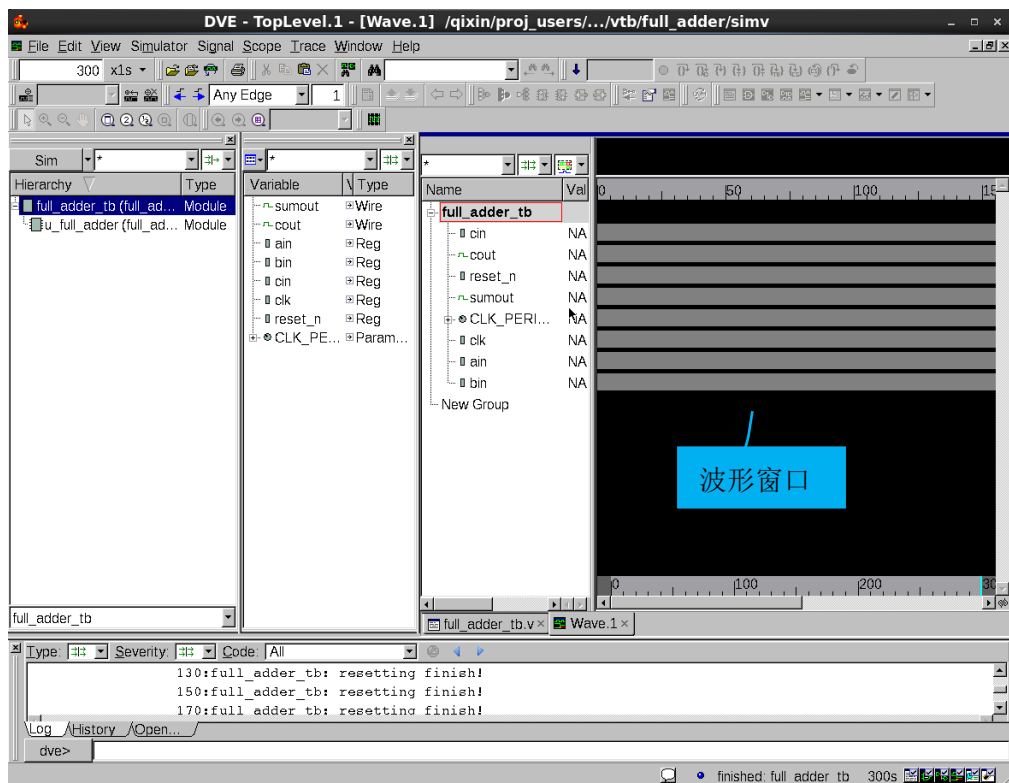
    // behavior of the adder can be synthesizable
    // *assign* means connectivity
    // assign {c_out, sum_out} = a_in + b_in + c_in;

    //task 2. clock and reset generator
    //always @(posedge clk) begin
    //    reset = 1;
    //end

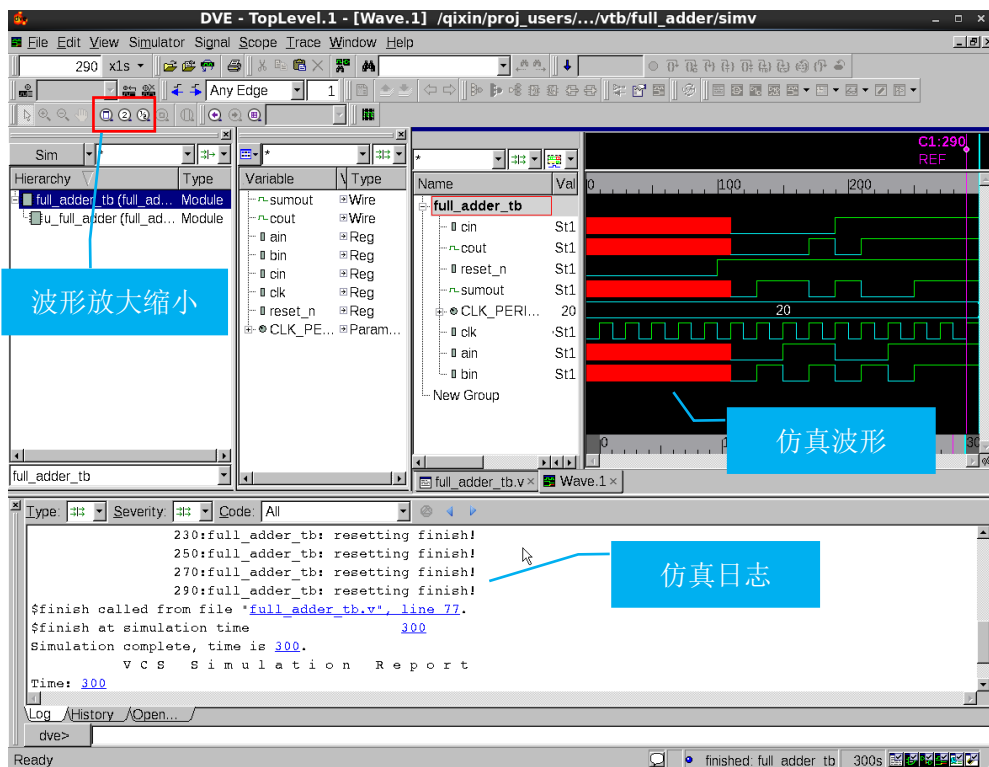
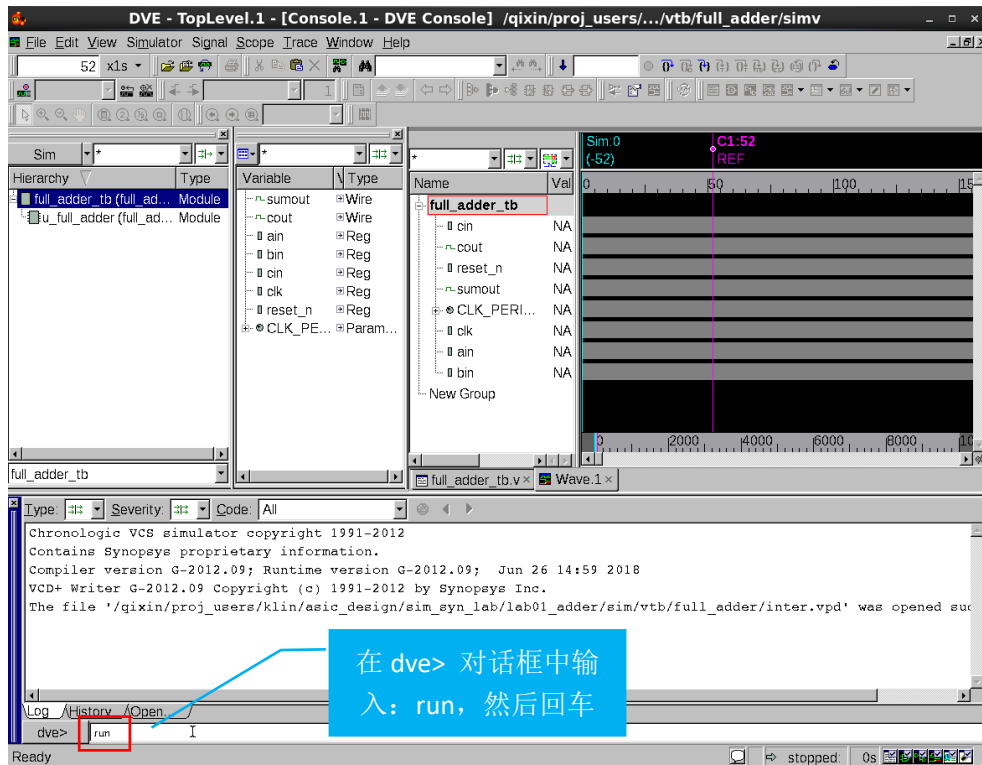
    // Goto design/sim_syn_lab/lab01_adder/sim/vtb/full_adder/full_adder_tb.v 10 Reuse
    full_adder_tb.v x
endmodule

```

finished: full_adder.tb 300s



g) 在命令行里输入 **run** 命令可以跑完整个 simulation



2. 使用命令行模型，调用 vcs 的命令进行编译和仿真

在实际工作中，一般很少用到 GUI 界面，除非是要用波形来 **DEBUG**，通常情况都是通过命令行来进行编译和仿真源代码的，下面介绍命令行的步骤：

a) 删除刚才仿真过程的中间文件：



```
rm -rf simv* csrc* *.tmp *.vpd *.key *.log *hdrs.v DVEfiles
```

b) 编译 RTL 和 Testbench 源代码:

```
vcs -sverilog -debug_all ./full_adder.v ./full_adder_tb.v -l com.log
```

```
$ vcs -sverilog -debug_all full_adder.v full_adder_tb.v -l com.log
Chronologic VCS (TM)
Version G-2012.09 -- Tue Jun 26 16:30:33 2018
Copyright (c) 1991-2012 by Synopsys Inc.
ALL RIGHTS RESERVED

This program is proprietary and confidential information of Synopsys Inc.
and may be used and disclosed only as authorized in a license agreement
controlling such use and disclosure.

Parsing design file 'full_adder.v'
Parsing design file 'full_adder_tb.v'
Top Level Modules:
    full_adder_tb
No TimeScale specified
Starting vcs inline pass...
2 modules and 0 UDP read.
recompiling module full_adder_tb
Both modules done.
if [ -x ../simv ]; then chmod -x ../simv; fi
g++ -o ../simv -melf_i386 -m32 -Wl,-whole-archive -Wl,-no-whole-archive
_mop.o rmapats.o /qixin/eda/synopsys/vcs/2012.09/linux/lib/libnplex_stub
.so /qixin/eda/synopsys/vcs/2012.09/linux/lib/librterrorinf.so /qixin/eda/syn
eda/synopsys/vcs/2012.09/linux/lib/libvcsnew.so /qixin/eda/synopsys/vcs/2012.0
s/vcs/2012.09/linux/lib/vcs_save_restore_new.o /qixin/eda/synopsys/vcs/2012.09
l
../simv up to date
```

```
$ ll
total 696
-rw-rw-r-- 1 klin klin 1531 Jun 26 16:30 com.log
drwxrwxr-x 3 klin klin 4096 Jun 26 16:30 csrc
-rw-rw-r-- 1 klin klin 3567 Jun 24 21:11 full_adder_tb.v
-rw-rw-r-- 1 klin klin 1087 Jun 24 21:09 full_adder.v
-rw-rw-r-- 1 klin klin 676 Jun 26 16:30 Makefile
-rw-rw-r-- 1 klin klin 917 Jun 24 21:06 Makefile.questasim
-rw-rw-r-- 1 klin klin 682 Jun 24 21:49 Makefile.vcs
-rw-rw-r-- 1 klin klin 398 Jun 24 21:13 Makefile.vcs.v1
-rw-rw-r-- 1 klin klin 1013 Jun 24 21:07 readme.txt
-rwxrwxr-x 1 klin klin 645978 Jun 26 16:30 simv
drwxrwxr-x 2 klin klin 4096 Jun 26 16:30 simv.daidir
-rw-rw-r-- 1 klin klin 20 Jun 24 21:08 timescale.v
-rw-rw-r-- 1 klin klin 731 Jun 24 22:46 transcript
-rw-rw-r-- 1 klin klin 45 Jun 24 21:08 vcs_clean.sh
-rw-rw-r-- 1 klin klin 86 Jun 24 21:07 vcs_com.sh
-rw-rw-r-- 1 klin klin 37 Jun 24 21:07 vcs_sim.sh
```

仿真日志
临时文件

可执行文
件和临时
文件

c) 仿真, 执行可执行文件 simv:

```
./simv -l simv.log
```

simv.log 为仿真 log 文件, vcdplus.vpd 为 dump 的波形文件



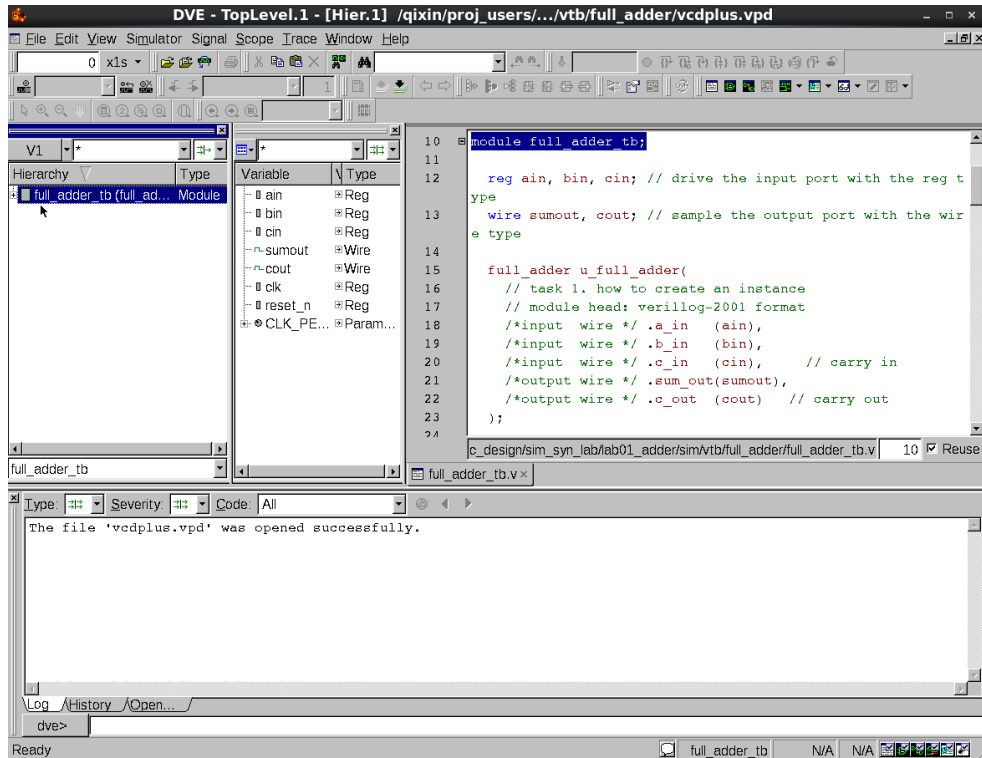
```
$ ./simv -l sim.log
Chronologic VCS simulator copyright 1991-2012
Contains Synopsys proprietary information.
Compiler version G-2012.09; Runtime version G-2012.09; Jun 26 16:34 2018
10:full_adder_tb: resetting ...
30:full_adder_tb: resetting ...
50:full_adder_tb: resetting ...
70:full_adder_tb: resetting ...
90:full_adder_tb: resetting ...
110:full_adder_tb: resetting finish!
130:full_adder_tb: resetting finish!
150:full_adder_tb: resetting finish!
170:full_adder_tb: resetting finish!
190:full_adder_tb: resetting finish!
210:full_adder_tb: resetting finish!
230:full_adder_tb: resetting finish!
250:full_adder_tb: resetting finish!
270:full_adder_tb: resetting finish!
290:full_adder_tb: resetting finish!
$finish called from file "full_adder_tb.v", line 77.
$finish at simulation time 300
VCS Simulation Report
Time: 300
CPU Time: 1.310 seconds; Data structure size: 0.0Mb
Tue Jun 26 16:34:51 2018
```

```
$ ll
total 708
-rw-rw-r-- 1 klin klin 1531 Jun 26 16:37 com.log
drwxrwxr-x 3 klin klin 4096 Jun 26 16:37 csrc
-rw-rw-r-- 1 klin klin 3538 Jun 26 16:36 full_adder_tb.v
-rw-rw-r-- 1 klin klin 1087 Jun 24 21:09 full_adder.v
-rw-rw-r-- 1 klin klin 676 Jun 26 16:30 Makefile
-rw-rw-r-- 1 klin klin 917 Jun 24 21:06 Makefile.questasim
-rw-rw-r-- 1 klin klin 682 Jun 24 21:49 Makefile.vcs
-rw-rw-r-- 1 klin klin 398 Jun 24 21:13 Makefile.vcs.v1
-rw-rw-r-- 1 klin klin 1013 Jun 24 21:07 readme.txt
-rw-rw-r-- 1 klin klin 1298 Jun 26 16:37 sim.log
-rwxrwxr-x 1 klin klin 649569 Jun 26 16:37 simv
drwxrwxr-x 2 klin klin 4096 Jun 26 16:37 simv.daidir
-rw-rw-r-- 1 klin klin 20 Jun 24 21:08 timescale.v
-rw-rw-r-- 1 klin klin 731 Jun 24 22:46 transcript
-rw-rw-r-- 1 klin klin 0 Jun 26 16:37 ucli.key
-rw-rw-r-- 1 klin klin 2510 Jun 26 16:37 vcdplus.vpd
-rw-rw-r-- 1 klin klin 45 Jun 24 21:08 vcs_clean.sh
-rw-rw-r-- 1 klin klin 86 Jun 24 21:07 vcs_com.sh
-rw-rw-r-- 1 klin klin 37 Jun 24 21:07 vcs_sim.sh
```

d) 打开 GUI，进行 DEBUG:

```
dve -vpd vcdplus.vpd &
```

就可以按照上面介绍的打开波形图进行 debug。



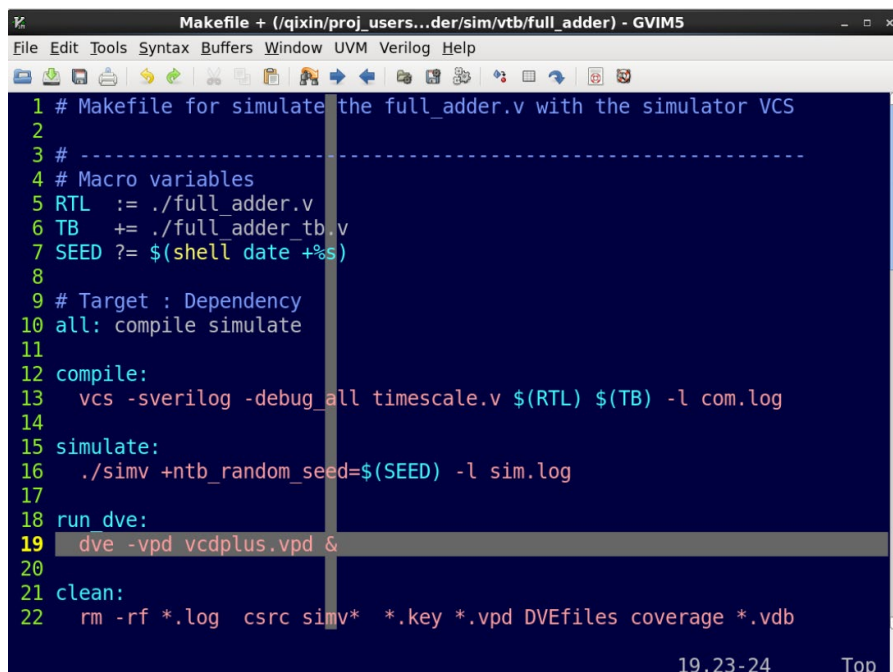
3. 使用 Makefile 脚本自动执行编译仿真命令

虽然使用命令行可以方便执行仿真任务，但是实际工程中，文件量大，每次仿真都敲一遍所有命令，浪费时间，可以将这些语句写入自动化运行的脚本中，在需要的时候直接调用脚本执行命令即可自动完成编译和仿真的过程，这里介绍使用 Makefile 脚本来实现编译仿真

a) 用 gvim 来创建 Makefile 文件:

gvim Makefile

b) 编辑该文件，写入如下命令:





- c) 如上图，当在 terminal 中输入 make 命令时，默认将执行目标 all，而 all 依赖于 compile 和 simulate 的两个命令。首先执行 compile 命令，然后执行 simulate 命令。如果单独执行 compile 和 simulate 命令，可以在 terminal 中输入：

`make compile; make simulate`

- d) 当要打开 GUI，进行 DEBUG 时，在 terminal 中输入：`make run_dve`

- e) DEBUG 结束，要删除中间文件，在 terminal 中输入：`make clean`

说明：在每一次修改源代码之后，重新运行编译和仿真时，都需要将当前的仿真目录删除干净，也就是把上一次编译和仿真生成的文件全部删掉；不然，旧的文件会影响下一次的仿真。