

## SMART 平台介绍与仿真

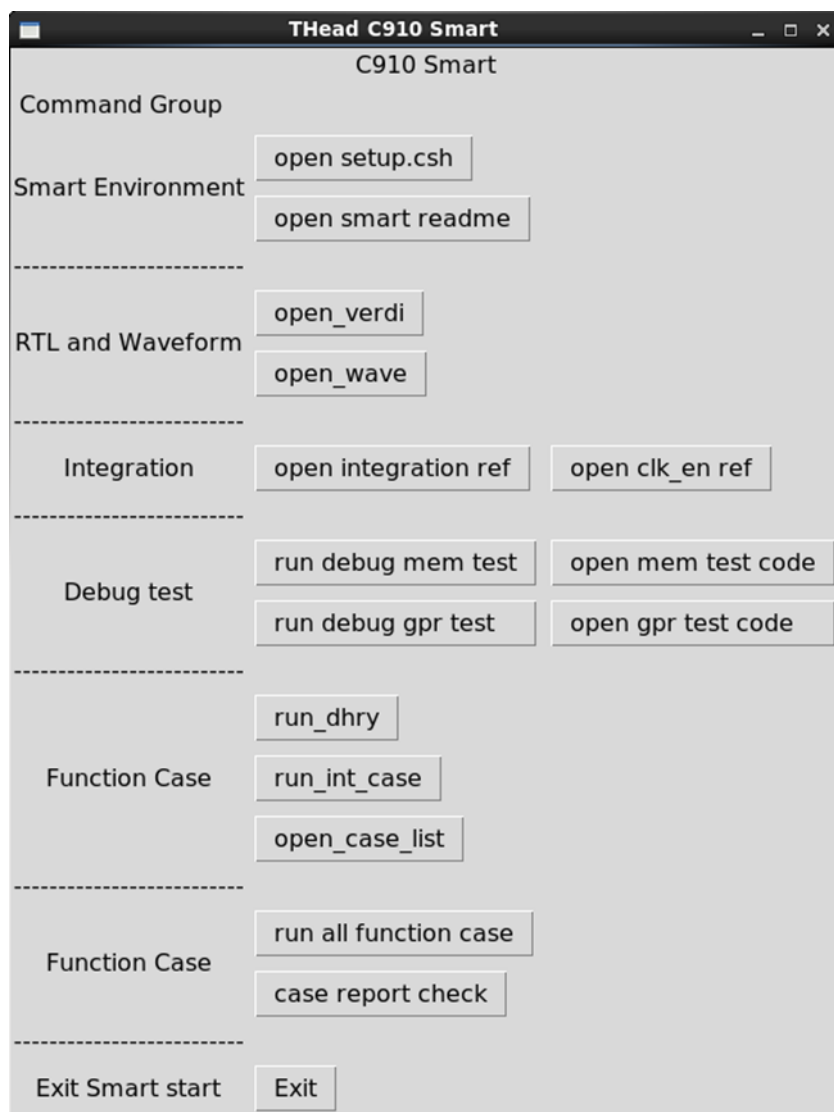
### 1、实验目的

通过在 SMART 平台上进行程序仿真，了解和掌握 SMART 平台的主要功能和使用方式。

### 2、实验步骤（包括实验结果，数据记录、截图等）

（1）配置好 SMART 平台环境。

- ① 登录服务器 admin，在 admin 节点上配置环境变量
- ② 用已经配置好的.cshrc 文件替换原有的.cshrc 文件
- ③ 将当前目录切换到已经复制到自己账户路径下的 SMART 平台根目录 smart9\_release/，使用指令 `source ./setup.csh` 和 `./run_smart`，打开图形界面



（2）仿真各例程,包括 dhrystone 测试、memory 和 gpr 的读写测试、中断测试，对输出结果截图。

- ① 仿真 Dhrystone 程序

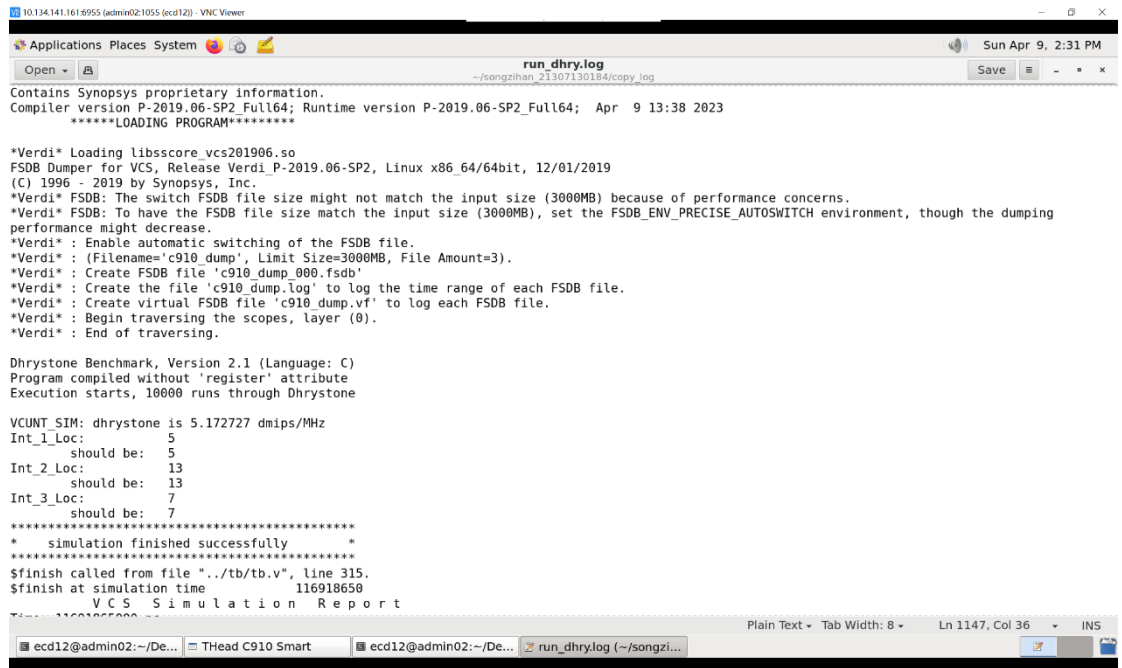
在图形界面处点击 **run\_dhry**，开始仿真 Dhrystone 程序。

新开一个终端，可使用指令 **bjobs** 命令查看当前运行的任务。

一段时间后再次运行 **bjobs** 命令，得到提示 “No unfinished job found”，说明相关任务都已经运行完毕。

进入 **workdir** 目录，可以找到 **run.log** 文件，输入命令 **gedit run.log**，查看

dhystone 测试结果:



```
run_dhry.log
Contains Synopsys proprietary information.
Compiler version P-2019.06-SP2_Full64; Runtime version P-2019.06-SP2_Full64; Apr 9 13:38 2023
*****LOADING PROGRAM*****

*Verdi* Loading libsscore_vcs201906.so
FSDump Dumper for VCS, Release Verdi_P-2019.06-SP2, Linux x86_64/64bit, 12/01/2019
(C) 1996 - 2019 by Synopsys, Inc.
*Verdi* FSDump: The switch FSDump file size might not match the input size (3000MB) because of performance concerns.
*Verdi* FSDump: To have the FSDump file size match the input size (3000MB), set the FSDump_ENV_PRECISE_AUTOSWITCH environment, though the dumping
performance might decrease.
*Verdi* : Enable automatic switching of the FSDump file.
*Verdi* : (Filename='c910_dump', Limit Size=3000MB, File Amount=3).
*Verdi* : Create FSDump file 'c910_dump_000.fsdump'
*Verdi* : Create the file 'c910_dump.log' to log the time range of each FSDump file.
*Verdi* : Create virtual FSDump file 'c910_dump.vf' to log each FSDump file.
*Verdi* : Begin traversing the scopes, layer (0).
*Verdi* : End of traversing.

Dhystone Benchmark, Version 2.1 (Language: C)
Program compiled without 'register' attribute
Execution starts, 10000 runs through Dhystone

VCUNT_SIM: dhystone is 5.172727 dmips/MHz
Int_1_Loc: 5
should be: 5
Int_2_Loc: 13
should be: 13
Int_3_Loc: 7
should be: 7
*****
* simulation finished successfully
*****
$finish called from file "../tb/tb.v", line 315.
$finish at simulation time: 116918650
VCS Simulation Report
```

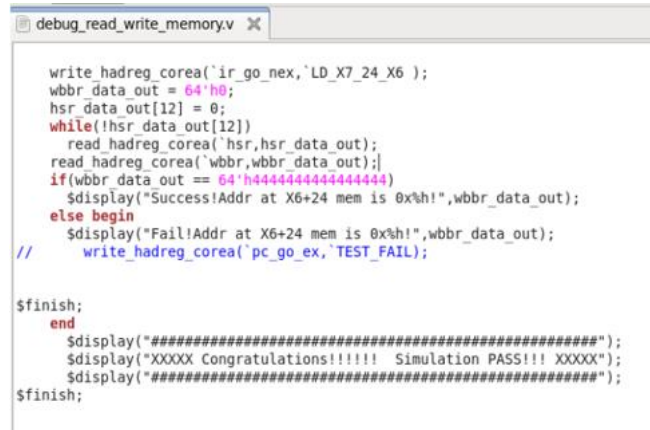
将 run.log 移动到目录 copy\_log 下, 并将其改名为 run\_dhry.log。

## ② 仿真 memory 读写调试程序

在图形界面处点击 **run debug mem test**, 开始仿真 memory 读写调试程序。

在新开的终端中使用指令 **bjobs** 命令, 查看当前运行的任务。

在图形界面处点击 **open mem test code**, 可以阅读其读写测试用例:



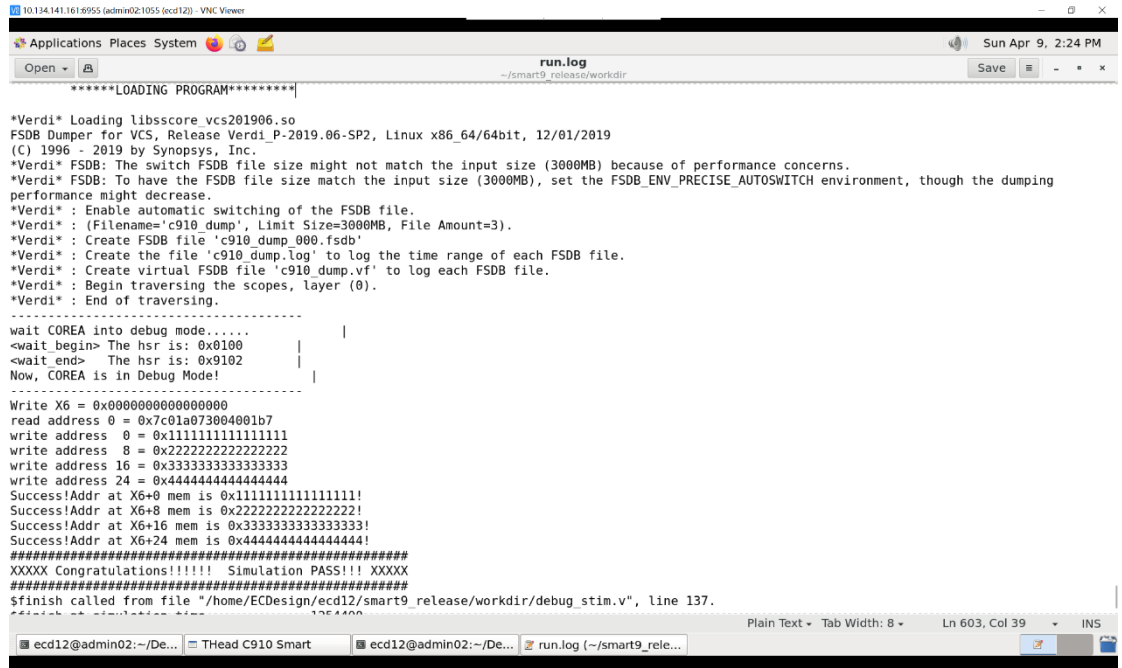
```
debug_read_write_memory.v

write hadreg_core('ir_go_nex,'LD_X7_24_X6 );
wbr data_out = 64'h0;
hbr data_out[12] = 0;
while(!hbr_data_out[12])
  read_hadreg_core('hbr,hbr_data_out);
read_hadreg_core('wbr,wbr_data_out);
if(wbr_data_out == 64'h4444444444444444)
  $display("Success!Addr at X6+24 mem is 0x%h!",wbr_data_out);
else begin
  $display("Fail!Addr at X6+24 mem is 0x%h!",wbr_data_out);
//   write_hadreg_core('pc_go_ex,'TEST_FAIL);

$finish;
end
$display("#####");
$display("XXXXX Congratulations!!!!!! Simulation PASS!!! XXXXX");
$display("#####");
$finish;
```

一段时间后再次运行” bjobs” 命令, 得到提示 “No unfinished job found”, 说明相关任务都已经运行完毕。

进入 workdir 目录, 可以找到 run.log 文件, 输入命令 **gedit run.log**, 查看 memory 测试结果:



```
*****LOADING PROGRAM*****
*Verdi* Loading libsscore_vcs201906.so
FSDB Dumper for VCS, Release Verdi_P-2019.06-SP2, Linux x86_64/64bit, 12/01/2019
(C) 1996 - 2019 by Synopsys, Inc.
*Verdi* FSDB: The switch FSDB file size might not match the input size (3000MB) because of performance concerns.
*Verdi* FSDB: To have the FSDB file size match the input size (3000MB), set the FSDB_ENV_PRECISE_AUTOSWITCH environment, though the dumping
performance might decrease.
*Verdi* : Enable automatic switching of the FSDB file.
*Verdi* : (Filename='c910_dump', Limit Size=3000MB, File Amount=3).
*Verdi* : Create FSDB file 'c910_dump.000.fsd'
*Verdi* : Create the file 'c910_dump.log' to log the time range of each FSDB file.
*Verdi* : Create virtual FSDB file 'c910_dump.vf' to log each FSDB file.
*Verdi* : Begin traversing the scopes, layer (0).
*Verdi* : End of traversing.
-----
wait COREA into debug mode..... |
<wait begin> The hsr is: 0x0100 |
<wait end> The hsr is: 0x9102 |
Now, COREA is in Debug Mode! |
-----
Write X6 = 0x0000000000000000
read address 0 = 0x7c01a073004001b7
write address 0 = 0x1111111111111111
write address 8 = 0x2222222222222222
write address 16 = 0x3333333333333333
write address 24 = 0x4444444444444444
Success!Addr at X6+0 mem is 0x1111111111111111!
Success!Addr at X6+8 mem is 0x2222222222222222!
Success!Addr at X6+16 mem is 0x3333333333333333!
Success!Addr at X6+24 mem is 0x4444444444444444!
#####
XXXXX Congratulations!!!!!! Simulation PASS!!! XXXXX
#####
$finish called from file "/home/ECDesign/ecd12/smart9_release/workdir/debug_stim.v", line 137.
-----
```

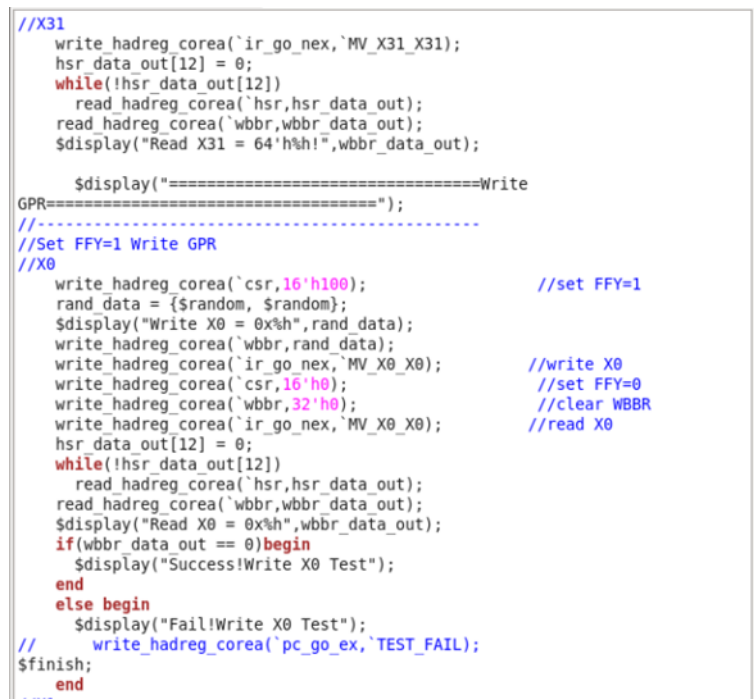
将 run.log 移动到目录 copy\_log 下，并将其改名为 run\_mem.log。

### ③ 仿真 gpr 读写调试程序

在图形界面处点击 **run debug gpr test**，开始仿真 gpr 读写调试程序。

在新开的终端中使用指令 **bjobs** 命令，查看当前运行的任务。

在图形界面处点击 **open gpr test code**，可以阅读其读写测试用例：



```
//X31
write_hadreg_corea(`ir_go_nex,`MV_X31_X31);
hsr_data_out[12] = 0;
while(!hsr_data_out[12])
    read_hadreg_corea(`hsr,hsr_data_out);
read_hadreg_corea(`wbbr,wbbr_data_out);
$display("Read X31 = 64'h%h!",wbbr_data_out);

$display("=====Write
GPR=====");
//-----
//Set FFY=1 Write GPR
//X0
write_hadreg_corea(`csr,16'h100); //set FFY=1
rand data = {$random};
$display("Write X0 = 0x%h",rand data);
write_hadreg_corea(`wbbr,rand data);
write_hadreg_corea(`ir_go_nex,`MV_X0_X0); //write X0
write_hadreg_corea(`csr,16'h0); //set FFY=0
write_hadreg_corea(`wbbr,32'h0); //clear WBRR
write_hadreg_corea(`ir_go_nex,`MV_X0_X0); //read X0
hsr_data_out[12] = 0;
while(!hsr_data_out[12])
    read_hadreg_corea(`hsr,hsr_data_out);
read_hadreg_corea(`wbbr,wbbr_data_out);
$display("Read X0 = 0x%h",wbbr_data_out);
if(wbbr_data_out == 0)begin
    $display("Success!Write X0 Test");
end
else begin
    $display("Fail!Write X0 Test");
// write_hadreg_corea(`pc_go_ex,`TEST_FAIL);
$finish;
end
//X1
```

一段时间后再次运行 **bjobs** 命令，得到提示 **"No unfinished job found"**，说明相关任务都已经运行完毕。

进入 **workdir** 目录，可以找到 **run.log** 文件，输入命令 **gedit run.log**，查看 gpr 测试结果：

```
10.134.141.161:8955 (admin02:1055 [ecd12]) - VNC Viewer
Applications Places System
Sun Apr 9, 2:51 PM
run_gpr.log
~/songzihan_21307130184/copy_log
Save
Contains Synopsys proprietary information.
Compiler version P-2019.06-SP2_Full64; Runtime version P-2019.06-SP2_Full64; Apr 9 14:33 2023
*****LOADING PROGRAM*****

*Verdi* Loading libscore_vcs201906.so
FSD8 Dumper for VCS, Release Verdi_P-2019.06-SP2, Linux x86_64/64bit, 12/01/2019
(C) 1996 - 2019 by Synopsys, Inc.
*Verdi* FSD8: The switch FSD8 file size might not match the input size (3000MB) because of performance concerns.
*Verdi* FSD8: To have the FSD8 file size match the input size (3000MB), set the FSD8_ENV_PRECISE_AUTOSWITCH environment, though the dumping
performance might decrease.
*Verdi* : Enable automatic switching of the FSD8 file.
*Verdi* : (Filename='c910_dump', Limit Size=3000MB, File Amount=3).
*Verdi* : Create FSD8 file 'c910_dump.000.fsd8'
*Verdi* : Create the file 'c910_dump.log' to log the time range of each FSD8 file.
*Verdi* : Create virtual FSD8 file 'c910_dump.vf' to log each FSD8 file.
*Verdi* : Begin traversing the scopes, layer (0).
*Verdi* : End of traversing.

-----
wait COREA into debug mode.....
<wait_begin> The hsr is: 0x0100
<wait_end> The hsr is: 0x9102
Now, COREA is in Debug Mode!
-----

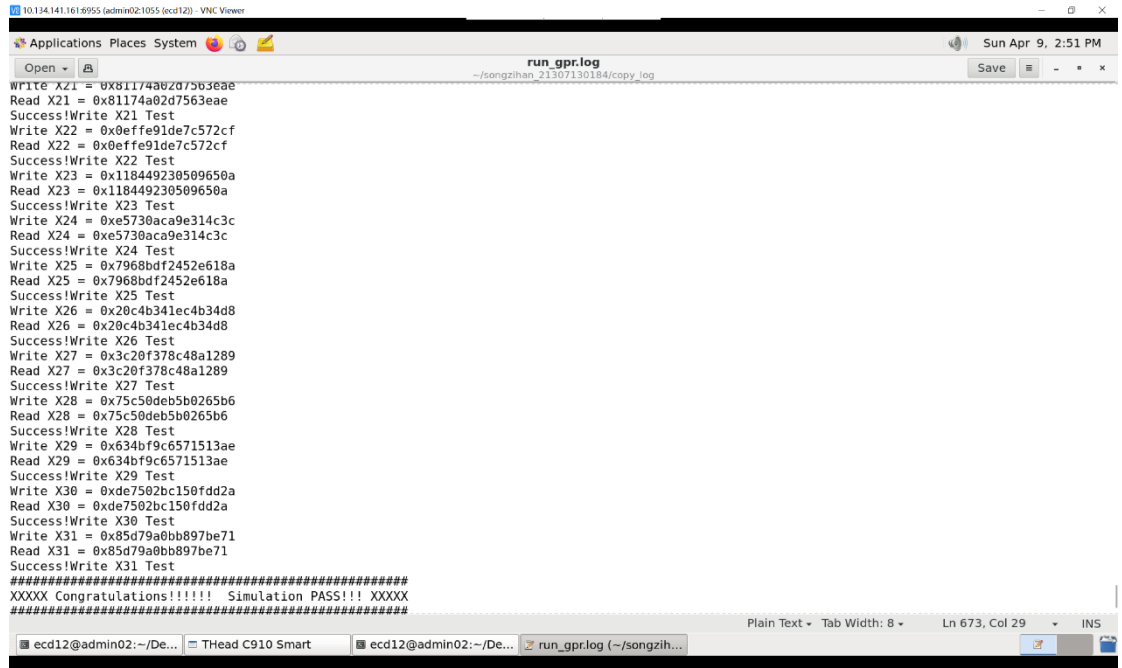
=====Read GPR=====
Read X0 = 64'h0000000000000000!
Read X1 = 64'h0000000000000000!
Read X2 = 64'h0000000000000000!
Read X3 = 64'h00000000000070013!
Read X4 = 64'h0000000000000001!
Read X5 = 64'h0000000000000000!
Read X6 = 64'h0000000000000000!
Read X7 = 64'h0000000000000000!
Read X8 = 64'h0000000000000000!
Read X9 = 64'h0000000000000000!
Read X10 = 64'h0000000000000000!
Read X11 = 64'h0000000000000000!

Plain Text Tab Width: 8 Ln 1, Col 1 INS
ecd12@admin02:~/De... THead C910 Smart ecd12@admin02:~/De... run_gpr.log (~songzih...
```

```
10.134.141.161:8955 (admin02:1055 [ecd12]) - VNC Viewer
Applications Places System
Sun Apr 9, 2:51 PM
run_gpr.log
~/songzihan_21307130184/copy_log
Save
Read X20 = 64'h0000000000000000!
Read X21 = 64'h0000000000000000!
Read X22 = 64'h0000000000000000!
Read X23 = 64'h0000000000000000!
Read X24 = 64'h0000000000000000!
Read X25 = 64'h0000000000000000!
Read X26 = 64'h0000000000000000!
Read X27 = 64'h0000000000000000!
Read X28 = 64'h0000000000000000!
Read X29 = 64'h0000000000000000!
Read X30 = 64'h0000000000000000!
Read X31 = 64'h0000000000000000!

=====Write GPR=====
Write X0 = 0x12153524c0895e81
Read X0 = 0x0000000000000000
Success!Write X0 Test
Write X1 = 0x8484d609b1f05663
Read X1 = 0x8484d609b1f05663
Success!Write X1 Test
Write X2 = 0x06b97b0d46df998d
Read X2 = 0x06b97b0d46df998d
Success!Write X2 Test
Write X3 = 0xb2c2846589375212
Read X3 = 0xb2c2846589375212
Success!Write X3 Test
Write X4 = 0x00f3e30106d7cd0d
Read X4 = 0x00f3e30106d7cd0d
Success!Write X4 Test
Write X5 = 0x3b23f1761e8dcd3d
Read X5 = 0x3b23f1761e8dcd3d
Success!Write X5 Test
Write X6 = 0x76d457ed462df78c
Read X6 = 0x76d457ed462df78c
Success!Write X6 Test
Write X7 = 0x7cfe9f9e33724c6
Read X7 = 0x7cfe9f9e33724c6

Plain Text Tab Width: 8 Ln 628, Col 32 INS
ecd12@admin02:~/De... THead C910 Smart ecd12@admin02:~/De... run_gpr.log (~songzih...
```



```
run_gpr.log
~/songzihan_21307130184/copy_log

Write X21 = 0x81174a02d7563eae
Read X21 = 0x81174a02d7563eae
Success!Write X21 Test
Write X22 = 0x0effe91de7c572cf
Read X22 = 0x0effe91de7c572cf
Success!Write X22 Test
Write X23 = 0x118449230509650a
Read X23 = 0x118449230509650a
Success!Write X23 Test
Write X24 = 0xe5730aca9e314c3c
Read X24 = 0xe5730aca9e314c3c
Success!Write X24 Test
Write X25 = 0x7968bdf2452e618a
Read X25 = 0x7968bdf2452e618a
Success!Write X25 Test
Write X26 = 0x20c4b341ec4b34d8
Read X26 = 0x20c4b341ec4b34d8
Success!Write X26 Test
Write X27 = 0x3c20f378c48a1289
Read X27 = 0x3c20f378c48a1289
Success!Write X27 Test
Write X28 = 0x75c50deb5b0265b6
Read X28 = 0x75c50deb5b0265b6
Success!Write X28 Test
Write X29 = 0x634bf9c6571513ae
Read X29 = 0x634bf9c6571513ae
Success!Write X29 Test
Write X30 = 0xde7502bc150fdd2a
Read X30 = 0xde7502bc150fdd2a
Success!Write X30 Test
Write X31 = 0x85d79a0bb897be71
Read X31 = 0x85d79a0bb897be71
Success!Write X31 Test
#####
XXXXX Congratulations!!!!!! Simulation PASS!!! XXXXX
#####
```

将 run.log 移动到目录 copy\_log 下，并将其改名为 run\_gpr.log。

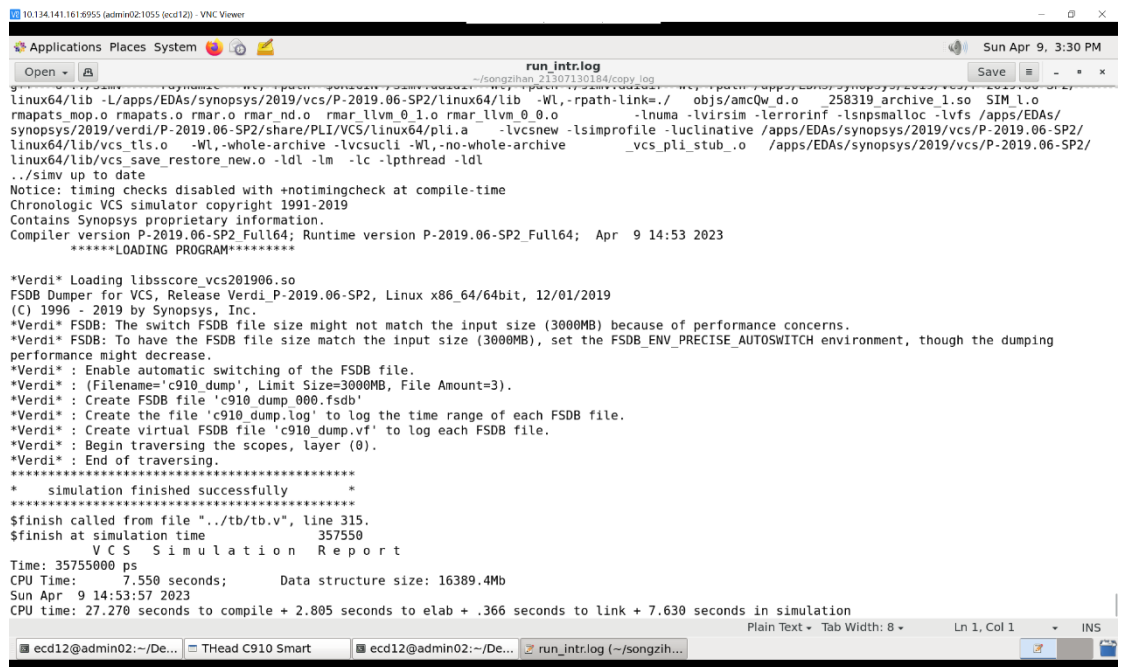
#### ④ 仿真 PLIC 的中断测试程序

在图形界面处点击 **run\_int\_case**，开始仿真 gpr 读写调试程序。

在新开的终端中使用指令 **bjobs** 命令，查看当前运行的任务。

一段时间后再次运行 **bjobs** 命令，得到提示 “No unfinished job found”，说明相关任务都已经运行完毕。

进入 **workdir** 目录，可以找到 **run.log** 文件，输入命令 **gedit run.log**，查看中断测试程序测试结果：



```
run_intr.log
~/songzihan_21307130184/copy_log

linux64/lib -L/apps/EDAs/synopsys/2019/vcs/P-2019.06-SP2/linux64/lib -Wl,-rpath-link=/obj/amcQw.d.o_258319.archive.1.so SIM.l.o
rmapats.mop.o rmapats.o rmar.o rmar.nd.o rmar.llvm.0.l.o rmar.llvm.0.0.o -lnuma -lvirsim -lerrorinf -lsnpsmalloc -lvfs /apps/EDAs/
synopsys/2019/verdi/P-2019.06-SP2/share/PLI/VCS/linux64/pli.a -lvcsnew -lsimprofile -luclnative /apps/EDAs/synopsys/2019/vcs/P-2019.06-SP2/
linux64/lib/vcs_tls.o -Wl,-whole-archive -lvcsucli -Wl,-no-whole-archive _vcs_pli_stub.o /apps/EDAs/synopsys/2019/vcs/P-2019.06-SP2/
linux64/lib/vcs_save_restore_new.o -ldl -lm -lc -lpthread -ldl
../simv up to date
Notice: timing checks disabled with +notimingcheck at compile-time
Chronologic VCS simulator copyright 1991-2019
Contains Synopsys proprietary information.
Compiler version P-2019.06-SP2 Full64; Runtime version P-2019.06-SP2 Full64; Apr 9 14:53 2023
*****LOADING PROGRAM*****

*Verdi* Loading libscore_vcs201906.so
FSDump Dumper for VCS, Release Verdi_P-2019.06-SP2, Linux x86_64/64bit, 12/01/2019
(C) 1996 - 2019 by Synopsys, Inc.
*Verdi* FSDump: The switch FSDump file size might not match the input size (3000MB) because of performance concerns.
*Verdi* FSDump: To have the FSDump file size match the input size (3000MB), set the FSDump_ENV_PRECISE_AUTOSWITCH environment, though the dumping
performance might decrease.
*Verdi* : Enable automatic switching of the FSDump file.
*Verdi* : (Filename='c910_dump', Limit Size=3000MB, File Amount=3).
*Verdi* : Create FSDump file 'c910_dump.000.fsdump'
*Verdi* : Create the file 'c910_dump.log' to log the time range of each FSDump file.
*Verdi* : Create virtual FSDump file 'c910_dump.vf' to log each FSDump file.
*Verdi* : Begin traversing the scopes, layer (0).
*Verdi* : End of traversing.
*****
* simulation finished successfully
*****
$finish called from file "../tb/tb.v", line 315.
$finish at simulation time 357550
V C S S i m u l a t i o n R e p o r t
Time: 35755000 ps
CPU Time: 7.550 seconds; Data structure size: 16389.4Mb
Sun Apr 9 14:53:57 2023
CPU time: 27.270 seconds to compile + 2.805 seconds to elab + .366 seconds to link + 7.630 seconds in simulation
```

将 run.log 移动到目录 copy\_log 下，并将其改名为 run\_intr.log。

#### (3) 自行编写 C 程序，并在命令行执行，对输出结果截图。

在 case 目录下建立一个新的文件夹，命名为 szh，在该文件夹里用 vi 编辑器编写 c 程序并保存，命名为 szh.c。返回/smart9\_release 目录下，使用命令 source setup.csh 对环境变量进行配置。

之后进入/workdir 目录下，执行../tools../tools/run\_case ../case/szh/szh.c  
自行编写一个对 0-9 累加求和的函数：

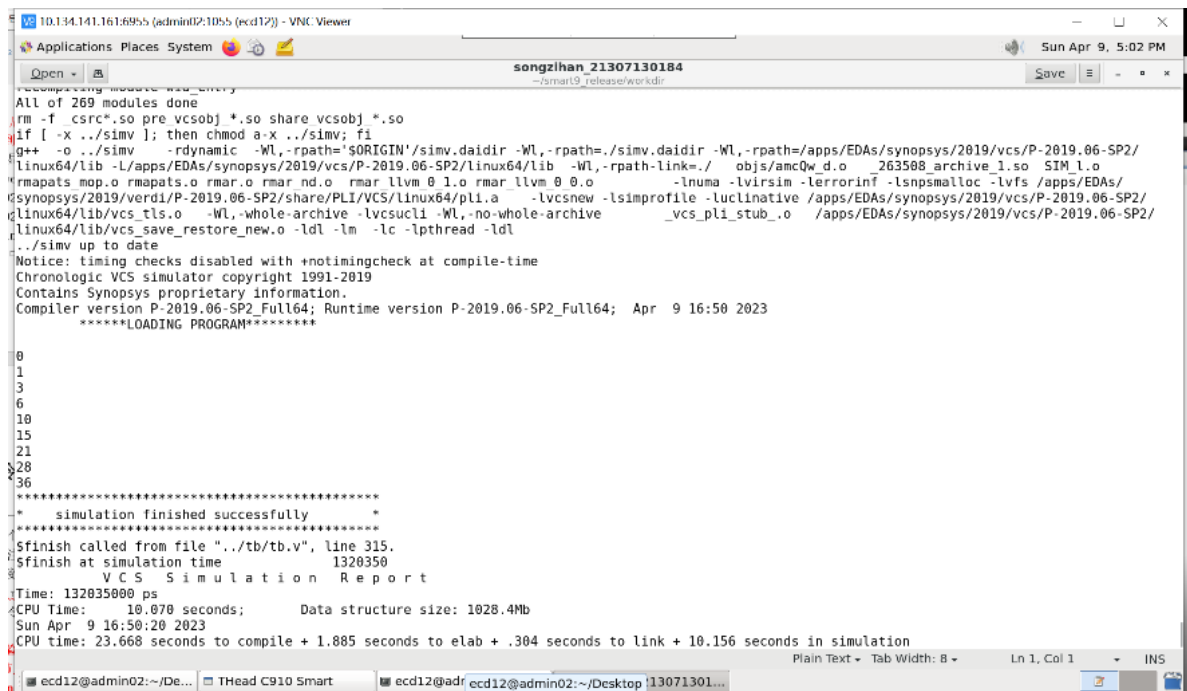
```
#include<stdio.h>
int main (void)
{
    int a[10];
    int sum = 0;
    for(int i=0;i<9;i++)
    {
        sum += i;
        printf("%d\n",sum);
    }
    return 0;
}
```

"~/smart9\_release/case/szh/szh.c" 12L, 137C

在新开的终端中使用指令 bjobs 命令，查看当前运行的任务。

一段时间后再次运行 bjobs 命令，得到提示 “No unfinished job found”，说明相关任务都已经运行完毕。

进入 workdir 目录，可以找到 run.log 文件，输入命令 gedit run.log，查看中断测试程序测试结果：



```
Compiling module 'std'...
All of 269 modules done
rm -f _csrc*.so pre vcsojb *.so share vcsojb *.so
if [ -x ../simv ]; then chmod a-x ../simv; fi
g++ -o ../simv -rdynamic -Wl,-rpath='$(ORIGIN)/simv.daidir' -Wl,-rpath=../simv.daidir -Wl,-rpath=/apps/EDAs/synopsys/2019/vcs/P-2019.06-SP2/
linux64/lib -L/apps/EDAs/synopsys/2019/vcs/P-2019.06-SP2/linux64/lib -Wl,-rpath-link=../objs/amcQw_d.o _263508_archive_1.so SIM_1.o
rmapats_mop.o rmapats.o rmar.o rmar.nd.o rmar.llvm 0 1.o rmar.llvm 0 0.o -lnuma -lvirsim -lerrorinf -lsnpsmalloc -lvfs /apps/EDAs/
synopsys/2019/verdi/P-2019.06-SP2/share/PLI/VCS/linux64/pli.a -lvcsnew -lsimprofile -luclintative /apps/EDAs/synopsys/2019/vcs/P-2019.06-SP2/
linux64/lib/vcs_tls.o -Wl,-whole-archive -lvcsucli -Wl,-no-whole-archive _vcs_pli_stub.o /apps/EDAs/synopsys/2019/vcs/P-2019.06-SP2/
linux64/lib/vcs_save_restore_new.o -ldl -lm -lc -lpthread -ldl
../simv up to date
Notice: timing checks disabled with +notimingcheck at compile-time
Chronologic VCS simulator copyright 1991-2019
Contains Synopsys proprietary information.
Compiler version P-2019.06-SP2_Full64; Runtime version P-2019.06-SP2_Full64; Apr 9 16:50 2023
*****LOADING PROGRAM*****

0
1
3
6
10
15
21
28
36
*****
* simulation finished successfully *
*****
$finish called from file "../tb/tb.v", line 315.
$finish at simulation time 13203500
VCS Simulation Report
Time: 132035000 ps
CPU Time: 10.070 seconds; Data structure size: 1028.4Mb
Sun Apr 9 16:50:20 2023
CPU time: 23.668 seconds to compile + 1.885 seconds to elab + .304 seconds to link + 10.156 seconds in simulation
Plain Text - Tab Width: 8 - Ln 1, Col 1 - INS
```

将 run.log 移动到目录 copy\_log 下，并将其改名为 run\_szh.log。

### 3、实验分析和总结



①实验中对两个不同的读写程序——memory 程序和 gpr 程序进行了仿真，观察二者代码可以发现，memory 读写调试程序进行的是读写上的计算仿真验证，而 gpr 则对 X0 到 X31 进行了遍历的读写调试。所以使用 **bjobs** 命令可以发现，gpr 的运行时间要远远大于 memory 的运行时间。

②实验中，在图形界面进行测试后，可以用 **bjobs** 命令查看测试是否进行完成。因为 **bjobs** 命令检测的是当前正在运行的也就是说未完成的程序，在测试程序运行过程中，运行该命令可以看到当前测试的相关信息，而测试完成后运行 **bjobs** 则会显示 “No unfinished job found”。例如：

```
admin:/home/ECDesign/ecd12>[51]bjobs
JOBID  USER  STAT  QUEUE  FROM_HOST  EXEC_HOST  JOB_NAME  SUBMIT_TIME
574627 ecd12  RUN   normal  admin      32*asicskl0 */workdir/ Apr 12 08:10
```

由此可以得到当前测试的运行状态、任务 ID、提交时间等信息。

③每次仿真之后，workdir 目录下都会生成一个新的对应的 run.log 文件，并且覆盖之前的 run.log 文件。因此如果要保存 run.log 文件，需要将其移动到新的文件夹 copy\_log 下并重命名。

```
admin:/home/ECDesign/ecd12/songzihan_21307130184/copy_log>[65]ll
total 4281
drwxr-xr-x 2 ecd12 ECDesign 4096 Apr  9 17:07 ./
drwxr-xr-x 3 ecd12 ECDesign 4096 Apr  9 14:05 ../
-rw-r--r-- 1 ecd12 ECDesign 84006 Apr  9 13:56 run_dhry.log
-rw-r--r-- 1 ecd12 ECDesign 46015 Apr  9 14:34 run_gpr.log
-rw-r--r-- 1 ecd12 ECDesign 41665 Apr  9 14:53 run_intr.log
-rw-r--r-- 1 ecd12 ECDesign 42537 Apr  9 14:20 run_mem.log
-rw-r--r-- 1 ecd12 ECDesign 39299 Apr  9 16:50 run_szh.log
admin:/home/ECDesign/ecd12/songzihan_21307130184/copy_log>[66]
```

#### 4、实验收获、存在问题、改进措施或建议等

本次实验我在 SMART 平台上进行了包含自己所编写的程序在内的一系列程序仿真，进一步理解了 SMART 平台的基础结构和功能，熟悉了其结构和使用方式。