Exercise

1. Setup EDA Environment

% setdt vcs
% setdt syn

2. Generate **SAIF** (Switching Activity Interchange Format) file.

a) Copy the file **/js1/songch/DA_VLSI/DA_VLSI_Lab4.tgz** to your home directory.
b) Extract the files % **tar zxvf DA_VLSI_Lab4.tgz,** and enter this directory.
c) Modify the file **gcd_rtl_2009.v** (in folder **verilog**). (Step 1 in the following tutorial)
d) Generate a **SAIF** file.
% vcs -PP +lint=all +v2k -debug -line ./verilog/gcd_rtl_2009.v
% ./simv
e) Check if the file **gcd_rtl_top.saif** exists in the directory.

3. Power Analysis

Power Compiler (integrated into the Design Compiler) reads the **SAIF file** and generates switching activity. (You may also use the command **set_switching_activity** to setup a switching activity for every input port except for clock port. **But the method discussed here is recommended in general**)

a) Enter Design Compiler Tcl-Environment.
% dc_shell-t
% dc_shell> source **./rm_setup/dc_setup.tcl**
//For simplicity, we source this file for dc_shell setup this file will source common_setup.tcl
// Make sure you set correct **DESIGN_REF_DATA_PATH** in common_setup.tcl

b) Initializes the name-mapping database.
% dc_shell> **saif_map -start**
// use command “man saif_map” to view more information about “saif_map”

c) Compile your design.

% dc_shell> **analyze -format verilog gcd_rtl_2009.v**

% dc_shell> **elaborate gcd_rtl_top** // the module name of your top design % dc_shell> **link**

% dc_shell> **source -echo timing_constraints.tcl**
// put your commands for constraining design into a script file

% dc_shell> **compile_ultra**

d) Save your compiled design.

% dc_shell> **change_names –rule verilog –hier**

% dc_shell> **write -f verilog –hier –o ./results/gcd_map.v**

% dc_shell> **write –f ddc –hier –o ./results/gcd_map.ddc**

% dc_shell> **report_saif –hier –missing –type rtl**

e) Read SAIF file. // check the annotated

% dc_shell> **report_power** // power analysis

% dc_shell> **read_saif -instance gcd_rtl_2009_test/gcd_rtl –input gcd_rtl_top.saif -auto_map_names**

The SAIF file generated in the simulation

Identify the module instance in the .saif file that corresponds to the **current_design.**
Here, the module instance **gcd_top** in the test bench module **gcd_test** is specified.

4. Gate Level Simulation

a) Check if gate level description file (**gcd_map.v**) exists in your **results** directory.

b) Build a simulator and watch the waveforms.

%set **SEADVCSLIB**=/js1/songch/SAED32_EDK/lib/stdcell_rvt/verilog/saed32nm.v

%vcs -PP +lint=all +v2k -debug -line \
$SEADVCSLIB ./results/gcd_map.v ./verilog/gcd_gate_test.v

%./simv -gui

5. Gate level dynamic power optimization

Use the synthesized gate level list, e.g. gcd_map.v, to generate a SAIF file by gate level simulation. Note: Check correct use of system task $set_gate_level_monitoring("on");.

a) Enter Design Compiler Tcl-Environment.

% dc_shell-t

% dc_shell> source **./rm_setup/dc_setup.tcl**

b) Initializes the name-mapping database.

% dc_shell> **saif_map -start**

// use command “man saif_map” to view more information about “saif_map”

c) Compile your design.

% dc_shell> **read_ddc gcd_map.ddc** //read your synthesized gate level list

% dc_shell> **reset_switching_activity** // reset the switching information annotated before

% dc_shell> **read_saif –auto_map_names –instance **
gcd_gate_test/gcd_gate –input gcd_gate_top.saif

% dc_shell> **report_saif –hier –missing –type rtl**

% dc_shell> **set_max_dynamic_power 0** //power constraints

% dc_shell> **compile_ultra -inc**

d) Save your compiled design.

% dc_shell> **change_names –rule verilog –hier**

% dc_shell> **write -f verilog –hier –o ./results/gcd_map_dp.v**

% dc_shell> **write –f ddc –hier –o ./results/gcd_map_dp.ddc**

e) Read Report.

% dc_shell> **report_timing** // check the timing constraints by timing analysis

% dc_shell> **report_power** // check the power consumption by power analysis % dc_shell> **exit** //quit design compiler

6. Leakage power optimization

Add the cell library with high threshold voltage and the other one with low threshold voltage to the link library through TCL script.

a) Enter Design Compiler Tcl-Environment.

% dc_shell-t

% dc_shell> source **rm_setup/dc_setup.tcl**

% dc_shell> source **addhldb.tcl** //add high/low threshold voltage libraries.

b) Initializes the name-mapping database. %

dc_shell> **saif_map -start**

c) Compile your design.

% dc_shell> **read_ddc gcd_map_dp.ddc** //read your synthesized gate level list

% dc_shell> **set_max_leakage_power 0** //power constraints

% dc_shell> **compile_ultra -inc**

d) Save your compiled design.

% dc_shell> **change_names –rule verilog –hier**

% dc_shell> **write -f verilog –hier –o ./results/gcd_map_lp.v**

% dc_shell> **write –f ddc –hier –o ./results/gcd_map_lp.ddc**

e) Read Report.

% dc_shell> **report_timing** // check the timing constraints by timing analysis %

dc_shell> **report_power** // check the power consumption by power analysis %

dc_shell> **exit** //quit design compiler

Make a comparison between the synthesized results with/without leakage (dynamic) power optimization, and what can you conclude?

Tips:

Use “man command_name”, in dc_shell promote, to view more information about the commands

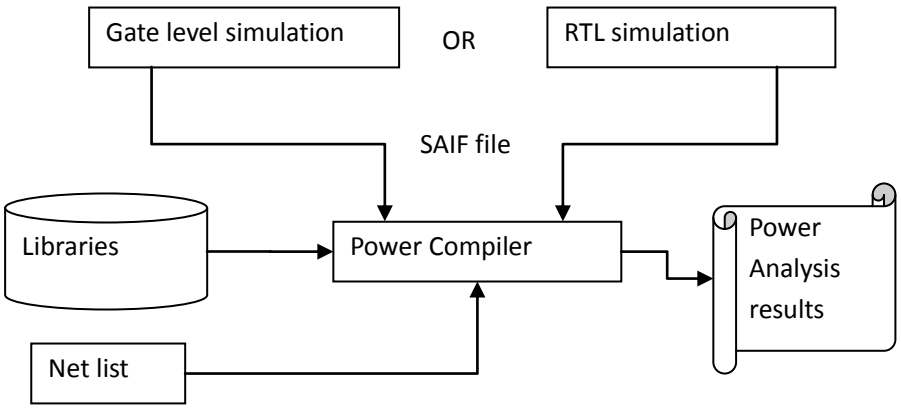
Please refer to **http://staff.ustc.edu.cn/~songch/da-ug.htm** for the following resources

1. Verilog-basics for the Verilog HDL.
2. the Linux tutorial and simple vi manual, respectively, for the simple Linux command and use of **vi/emacs/gedit(recommended)** editor.

Please read the **error** information carefully if any.

Tutorial on Simulation

1. You must read into or set **switching activity information** before running **report_power**. We first show how to generate **SAIF (Switching Activity Interchange Format)** file.



The following example shows the use of system tasks, in test bench module, to generate SAIF file for GCD.

module **gcd_rtl_2009_test**;
… gcd_rtl_top #(16) **gcd_rtl**(…
initial begin
……

$set_gate_level_monitoring("rtl_on");
$set_toggle_region(gcd_rtl_2009_test.gcd_rtl);
// module instance name: *testbench_name.module_instance_name*
$toggle_start();

….
end
…
always @ (posedge clk) begin
if(done_ct==0) begin
done_ct <= 4'h7;
$toggle_stop();

$toggle_report("gcd_rtl_top.saif",1.0e-10,"gcd_rtl_2009_test.gcd_rtl");
$finish();
end
…

For RTL simulation:
$set_gate_level_monitoring("**rtl_on**");
For gate level simulation:
$set_gate_level_monitoring("**on**");

Please refer to **Chapter 21 of** VCS User Guide for more about the system tasks related to SAIF. (http://staff.ustc.edu.cn/~songch/da-ug.htm **VCS Documents -> VCS User Guide**)