CADENCE TUTORIAL

San Diego State University,
Department of Electrical and Computer Engineering

Amith Dharwadkar and Ashkan Ashrafi

Contents

1) Introduction…… 3
2) Connecting to the Volta server ……..4
3) Creating a work directory ..……5
4) RTL Simulation ..……8
5) NETLIST Simulation ..……..12
6) Layout Design..……..16

ABSTRACT

This tutorial is aimed at introducing a user to the CADENCE tool. It gives step by step approach to performing a RTL simulation, gate level synthesis/simulation and finally layout design using SOC ENCOUNTER's auto place and route with TSMC 0.13μm standard cell library. The tutorial however does not discuss installation and environment setup for CADENCE. The entire tutorial is organized into five chapters beginning with connecting to Volta server on which CADENCE resides. It then explains RTL simulation, gate-level synthesis, post-synthesis simulation and layout design using encounter.

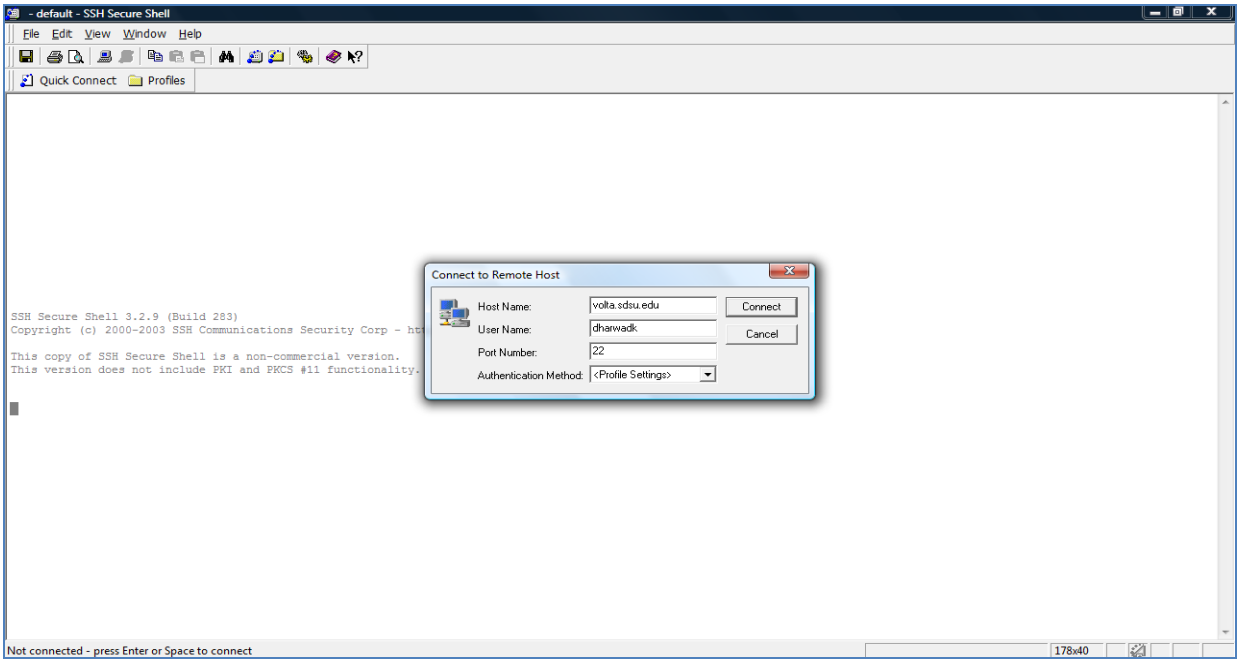
CHAPTER 1

Connecting to the Volta server

There are two ways of connecting to the Volta server on which CADENCE is installed.

- **Connecting remotely** – X-Win32 or Secure Shell Client can be used to establish a connection to the Volta server. X-Win32/SSH client can be downloaded from SDSU college website - http://scc.sdsu.edu/downloads.php.

SSH client installation steps are given here. X-Win32 can be installed in the same manner and a connection established with the Volta server.
Upon SSH client installation, click on the SSH client icon to invoke SSH. From the pop-up pane click on the File menu tab and select 'Quick Connect' option to connect to the Volta server.



Specify the **hostname** and the **username** and click on **Connect** button.
HostName – volta.sdsu.edu
Username and Password – Enter the username and password provided to you.

- **Connecting from a school computer** – The second approach to connect to the VOLTA server is to use the LINUX machines available on-campus. Login to the Linux machines using the username and password provided to you. Invoke the command prompt. Verify if a connection has been established with the Volta server. If the connection is not established, execute the following command at the prompt,

  1. Type **`ssh -X volta.sdsu.edu`** at the command prompt.
  2. Enter **username** and **password** to login to the Volta server.

CHAPTER 2

Creating a work directory

The first step in using CADENCE effectively is to create a work directory/folder. A work directory enables the user to organize the code files. The user can create folder for every project that is implemented on CADENCE. The following steps need to be followed in order to create a work folder,

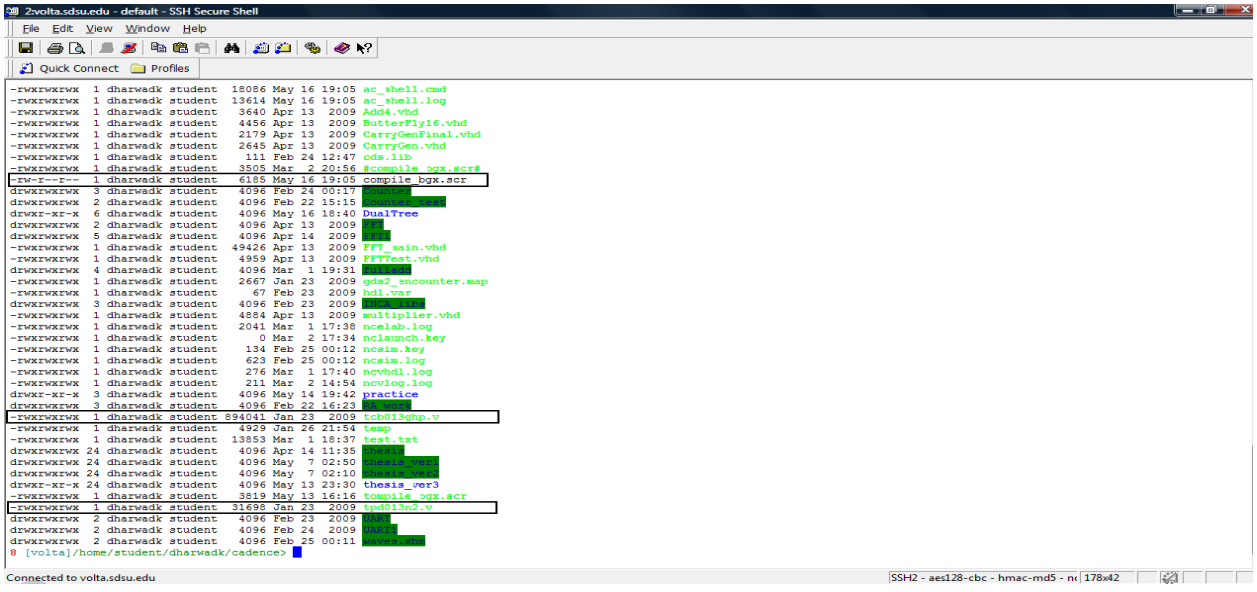
- Ensure that a folder called **cadence** is available in your home directory. Every user on the Volta server is assigned a home directory to which he/she has exclusive access. The home directory generally has the same name as the Username. Following commands could be used to verify the presence of cadence directory/folder in the home directory.

  ```
  >> cd /home/student/<Enter username>
  >> ls
  ```

  See if the cadence folder is present in this folder.

- Change to cadence directory using the command
  `>> cd cadence`

- This directory should have the following three files for CADENCE to compile verilog files without any errors. The files are,
  1. `Compile_bgx.scr`
  2. `Tpd013n2.v`
  3. `Tcb013ghp.v`



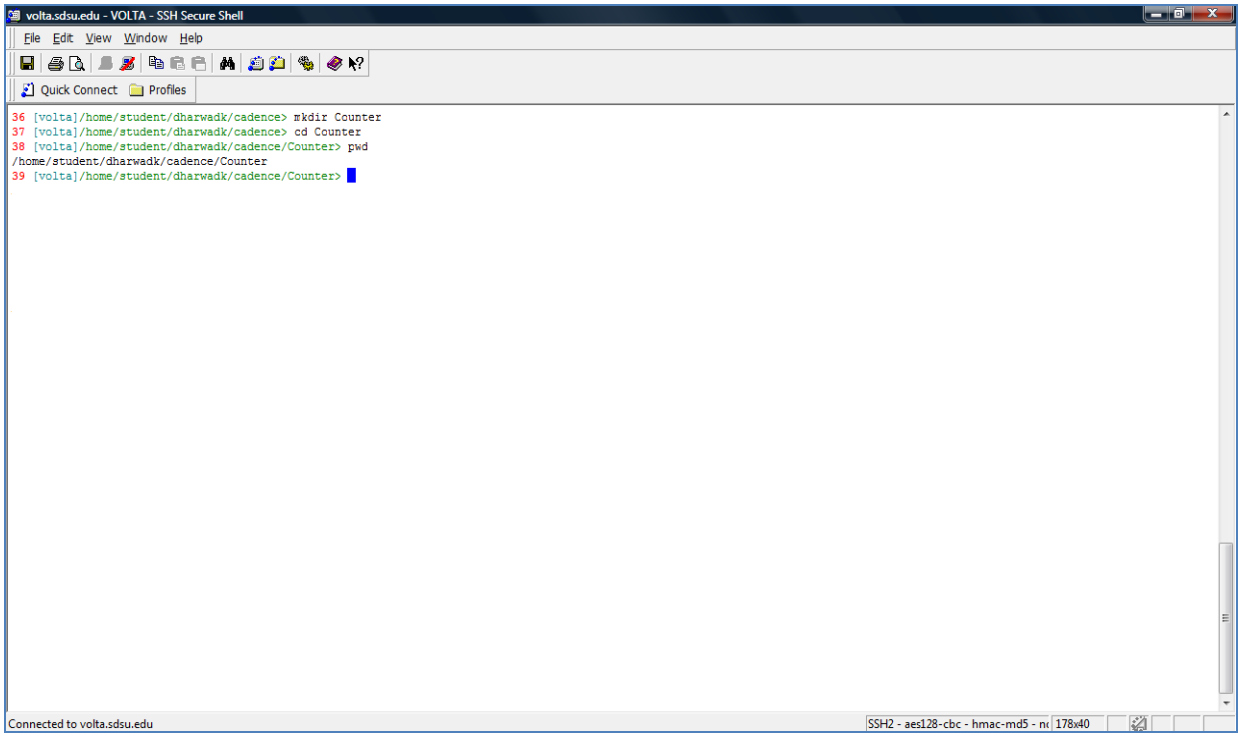
`Compile_bgx.scr` is a script that **bgx_shell** uses in order to compile the user created Verilog files. The `compile_bgx.scr` script contains module path and all other details necessary for

generating an object file. The remaining two files contain delay information for TSMC 0.13um standard cells that would be used in our design.

Notes - If these files are not present then they need to be copied into cadence folder.

- Once the cadence folder is created and all the necessary files copied, the work folder has to be created in the cadence directory. The work directory will house all the Verilog files and test benches. The following command can be used to do the same.

>> **`mkdir counter`** // this creates a folder named counter
>> **`cd counter`** // Changes the directory to counter.



- Now create a folder named **encounter** in the work directory. This folder will house files that are necessary for proper execution of the SOC ENCOUNTER tool. The SOC ENCOUNTER tool could be used to generate the layout for compiled/synthesized Verilog/netlist file.

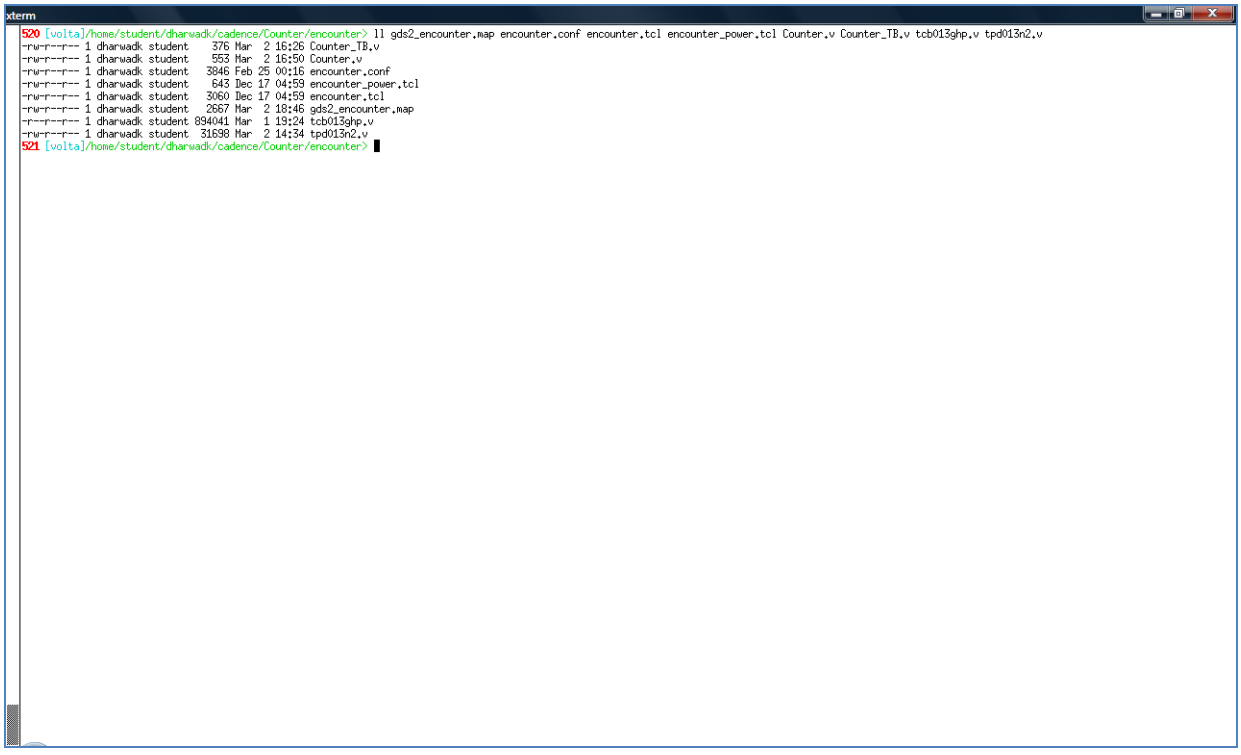
  Commands to be executed for creating the encounter folder,

  >> **`mkdir encounter`**
  >> **`cd encounter`**

  Note – Before executing these commands ensure that you are in the work directory (which is "Counter" in the current example).

- Copy the files `1)encounter.conf  2)encounter.tcl 3)encounter_power.tcl 4)gds2_encounter.map` into these folder. These files are necessary for proper functioning of the ENCOUNTER tool and should be made available to you before you start working with Cadence.
- Also copy `1)Tpd013n2.v 2)Tcb013ghp.v` into encounter folder. These contain delay information and definition of TSMC 0.13um standard cells/gates. These files are used when verifying the synthesized code using the Verilog-XL compiler.
- Copy both the Verilog as well as the testbench file into the encounter folder. You could use a SSH client to copy the already created files or create new Verilog and testbench files using a VI editor.

Note – Steps to Create verilog and testbench files are beyond the scope of this tutorial.



CHAPTER 3

RTL Simulation using "Verilog-XL" compiler

A good design practice dictates that a designer first carry out RTL simulation in order to validate the design and then eventually move over to gate level synthesis. A RTL simulation lets us know if the behavior of the component is as desired.

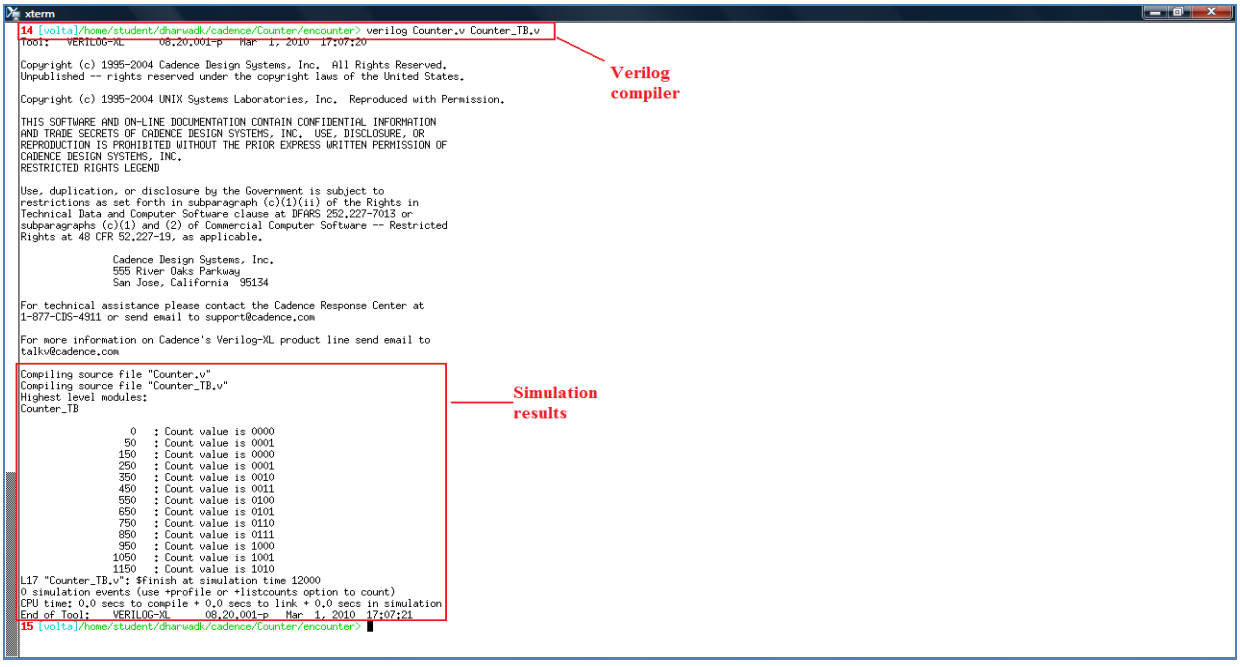
In order to carry out RTL simulation we can use either

1) Verilog-XL compiler.
2) NCVERILOG and NCSIM(simvision).

This tutorial describes the use of Verilog-XL compiler of CADENCE in order to carry out RTL simulation. The following command has to be executed to invoke the compiler,

```
>> verilog main_file.v test_bench.v
```

The user has to pay attention when specifying the files names. The files have to be specified in a particular order such that the lower-level modules are compiled before the higher-level modules. The testbench would be the last item to be compiled. Upon execution of the above statement the simulation results are displayed on the terminal. (For the simulation results to be displayed on the terminal the designer has to include "display" or "monitor" tasks in the testbench).



GATE LEVEL SYNTHESIS

Gate level synthesis involves implementing the behavior of the circuit (described by a Verilog model) using standard gates. In gate level synthesis, the Verilog file is synthesized into a netlist file which includes standard gates and their delays. In any design process the simulation of this gate-level netlist will eventually shows the success or failure of the design. Along with the netlist file the synthesis script also generates timing-report, area report and power report. The timing report is a critical piece of information. It gives details about critical path which indicates whether the circuit meets the timing/frequency requirements.

Steps to carry out gate level synthesis,

The **`compile_bgx.scr`** script is used to generate the gate level netlist. The following changes need to be made to the script in order to generate the gate level netlist file(which will have a .vh extension) and .sdc file which would contain the timing information for standard gates present in the netlist file. The .vh and the .sdc files are eventually used by SOC ENCOUNTER to generate the layout.

- Specify the Verilog file containing the definition of the top-level module.
- Specify the correct directory.
- Specify the name of the top-level module.
- Specify the frequency for which the design has to be implemented. Based on the frequency information provided, the synthesis tool tries to optimize the design so as to meet the frequency requirements.

```
xterm
#*************************************************************/
#* Compile Script for Cadence BuildGates                     */
#* TSMC 013um Standard Cell Library                          */
#* Verilog RTL                                               */
#*                                                           */
#* bgx_shell -f compile_bgx.scr                              */
#*                                                           */
#* Dr. Ashkan Ashrafi  SDSU Department of ECE                */
#* Jan. 23rd, 2009                                           */
#*                                                           */
#*************************************************************/

# All verilog files, separated by spaces
set my_verilog_files      {./Counter/encounter/Counter.v}
set my_report_path        {./Counter/encounter}
#set my_copy_all_files_for_encounter {./KH_work/binCounter/encounter/.}
#set my_verilog_files       {./fulladd/encounter/fulladdmain.v}
#set my_report_path         {./fulladd}
# Top-level Module
set my_toplevel_module  Counter

# The name of the clock pin. If no clock-pin
# exists, pick anything
set my_clock_pin          clk

# Target frequency in MHz for optimization
set my_clock_freq_MHz     100

# Delay of input signals (Clock-to-Q, Package etc.)
set my_input_delay_ns     0

# Reserved time for output signals (Holdtime etc.)
set my_output_delay_ns    0

#*********************************************
# No changes necessary beyond this point
#*********************************************

set TSMCHOME $env(TSMCHOME)

read_tlf $TSMCHOME/digital/Front_End/timing_power/tcb013ghp_211a/tcb013ghptc.tlf

read_verilog $my_verilog_files

set_global target_technology tcb013ghptc
set_global fix_multiport_nets true
set_global hdl_verilog_out_unconnected_style full
set_global hdl_write_multi_line_port_maps false
set_global hdl_tree_height_reduction true
set_global hdl_common_subexpression_elimination true
set_global aware_adder_architecture fcla
# set_global aware_mux_dissolve_size 14

"compile_bgx.scr" 85L, 3039C                                                    38,0-1        Top
```

Note – It is advisable to spend some time trying to understand the `compile_bgx.scr` *script. This script has details about the netlist file, .sdc file and timing, power and area reports. It*

specifies the path where the reports are generated. The .vh and .sdc files generated on running this script have to be copied into "encounter" folder in the work directory for the SOC ENCOUNTER tool to generate the layout.

- Additional changes have been made to the script to copy the netlist and .sdc file into encounter folder. In addition to that the reports generated are directly copied into encounter folder. Add the following lines of code to the script.

```
do_build_generic -module $my_toplevel_module


set_current_module $my_toplevel_module
set_top_timing_module $my_toplevel_module

set period [expr 1000.0/$my_clock_freq_MHz]
set_clock vclk -period $period
if {[get_names [find -inputs $my_clock_pin]] == $my_clock_pin} {
    set_clock_root -clock vclk $my_clock_pin
}

set_input_delay -clock vclk $my_input_delay_ns [get_names [find -inputs -no_clocks *]]
set_external_delay -clock vclk $my_output_delay_ns [get_names [find -outputs *]]
set_drive_cell -cell INVD8 [get_names [find -inputs *]]

# Disable this command to skip flattening
# do_dissolve_hierarchy -hierarchical

do_optimize -dont_reclaim_area -flatten -effort high -force -minimize multiple_output -priority area

write_verilog -hier $my_toplevel_module.vh

# Write SDC file
write_sdc $my_toplevel_module.sdc

# Added/Modified this code to make encounter work correctly. Start- Amith.
cp $my_toplevel_module.vh $my_toplevel_module.sdc $my_report_path/.

# The line below is commented out if precaution is taken to manually go and place the
# file in the work directory. tpd013n2.v file too needs to go in here.
#cp $TSMCHOME/digital/Front_End/verilog/tcb013ghp_211a/tcb013ghp.v $my_report_path/.

report_timing > $my_report_path/timing.rep
report_area   > $my_report_path/cell.rep
report_power  > $my_report_path/power.rep

# End - Amith
exit
"compile_bgx.scr" 97L, 3498C                                                    96,1          Bot
```

The compile_bgx.scr script is executed using the command,

>> **bgx_shell -f compile_bgx.scr**

```
45 [volta]/home/student/dharwadk/cadence> bgx_shell -f compile_bgx.scr

*****************************************************************
*                                                               *
*   Copyright (c)  Cadence Design Systems, Inc.  1994 - 2005.   *
*                    All rights reserved.                       *
*                                                               *
* Cadence acknowledges trademarks or registered trademarks of   *
* other organizations for their respective products and         *
* services.                                                     *
*                                                               *
* This program contains confidential, trade secret information *
* of Cadence Design Systems, Inc.  Any reproduction, use or     *
* disclosure of this program, including any attempt to obtain   *
* a human-readable version of this program, without the         *
* express, prior written consent of Cadence Design Systems,     *
* Inc. is strictly prohibited.                                  *
*                                                               *
* U.S. GOVERNMENT END USERS:  This program is a "commercial     *
* item" as that term is defined at 48 C.F.R. 2.101 (OCT 1995), *
* consisting of "commercial computer software" and "commercial *
* computer software documentation" as such terms are used in    *
* 48 C.F.R. 12.212 (SEPT 1995).  Consistent with 48 C.F.R. 212 *
* and 48 C.F.R. 227.7202-1 through 227.7202-4 (JUNE 1995), all *
* U.S. Government End Users acquire the program with only       *
* those rights set forth herein.                                *
*                                                               *
*                Cadence Design Systems, Inc.                   *
*                   2655 Seely Avenue                           *
*                 San Jose, CA 95134,  USA                      *
*                                                               *
*      bgx_shell    v05.17-s013+1    Feb  5 2007 03:22:26       *
*                                                               *
* Options  -BGX -License                                        *
*                                                               *
* Host volta                     Mon Mar 01 18:29:19 PST 2010 *
*                                                               *
*****************************************************************
            Loading command line script 'compile_bgx.scr'...
```

```
xterm
|                    Path Group Options Report                    |
|-------------------------------------------------------------------|
| PathGroup | Effort | All | Target |  Critical | Worst | TEFS |
|           |        | Pts |  Slack | Endpoints | Slack |      |
|-----------+--------+-----+--------+-----------+-------+------|
|   default |        |     |        |       0/9 |  +inf | 0.00 |
+-------------------------------------------------------------------+
              Increasing depth for slew propagation. Resetting timing <TOPT-523>.
+-------------------------------------------------------------------+
|                    Path Group Options Report                    |
|-------------------------------------------------------------------|
| PathGroup | Effort | All | Target |  Critical | Worst | TEFS |
|           |        | Pts |  Slack | Endpoints | Slack |      |
|-----------+--------+-----+--------+-----------+-------+------|
|   default |        |     |        |       0/9 |  +inf | 0.00 |
+-------------------------------------------------------------------+
              Increasing depth for slew propagation. Resetting timing <TOPT-523>.
+-------------------------------------------------------------------+
|                    Path Group Options Report                    |
|-------------------------------------------------------------------|
| PathGroup | Effort | All | Target |  Critical | Worst | TEFS |
|           |        | Pts |  Slack | Endpoints | Slack |      |
|-----------+--------+-----+--------+-----------+-------+------|
|   default |        |     |        |       0/9 |  +inf | 0.00 |
+-------------------------------------------------------------------+
    +-------------------------------------------------------------------+
    |                                Counter                            |
    |-------------------------------------------------------------------|
    |  Cell area  |  Total area | Worst slack | Local slack | CPU(s) Mem(M) |
    |-------------+-------------+-------------+-------------+---------------|
    |      171.44 |      171.44 |        +INF |             |      4    103 |
    +-------------------------------------------------------------------+
              Restarting optimization loop. <TOPT-534>.
+-------------------------------------------------------------------+
|                    Path Group Options Report                    |
|-------------------------------------------------------------------|
| PathGroup | Effort | All | Target |  Critical | Worst | TEFS |
|           |        | Pts |  Slack | Endpoints | Slack |      |
|-----------+--------+-----+--------+-----------+-------+------|
|   default |        |     |        |       0/9 |  +inf | 0.00 |
+-------------------------------------------------------------------+
+-------------------------------------------------------------------+
|                    Path Group Options Report                    |
|-------------------------------------------------------------------|
| PathGroup | Effort | All | Target |  Critical | Worst | TEFS |
|           |        | Pts |  Slack | Endpoints | Slack |      |
|-----------+--------+-----+--------+-----------+-------+------|
|   default |        |     |        |       0/9 |  +inf | 0.00 |
+-------------------------------------------------------------------+
              Command do_optimize -dont_reclaim_area -flatten -effort high
              -force -minimize multiple_output -priority area finished at Mon
              Mar  1 18:29:23 2010
               using 0:0:1 Real time. Current peak memory: 103.793MB
              <COMMON-076>.
46 [volta]/home/student/dharwadk/cadence>
```

- Observe that files `1) Counter.vh 2) Counter.sdc` are generated and placed in the encounter folder. `Counter.vh` is the netlist file where as `Counter.sdc` is the timing file which has all the timing details for the standard gates present in the netlist file.

CHAPTER 4

NETLIST Simulation

This chapter discusses simulation of the netlist file. It is also known as post-synthesis simulation. There are two approaches to doing this. We can either use the Verilog-XL compiler or use NCVERILOG/NCSIM. Chapter 2 talked about using Verilog-XL for carrying out RTL simulation. The same procedure could be used for simulating the netlist file. However, we will have to include TSMC 0.13um standard cell library files **`tcb013ghp.v`** and **`tpd013n2.v`** when invoking the compiler. These files contain delay information and definition for standard gates present in the netlist file. The command to invoke the Verilog-XL compiler to compile and simulate the netlist file is,

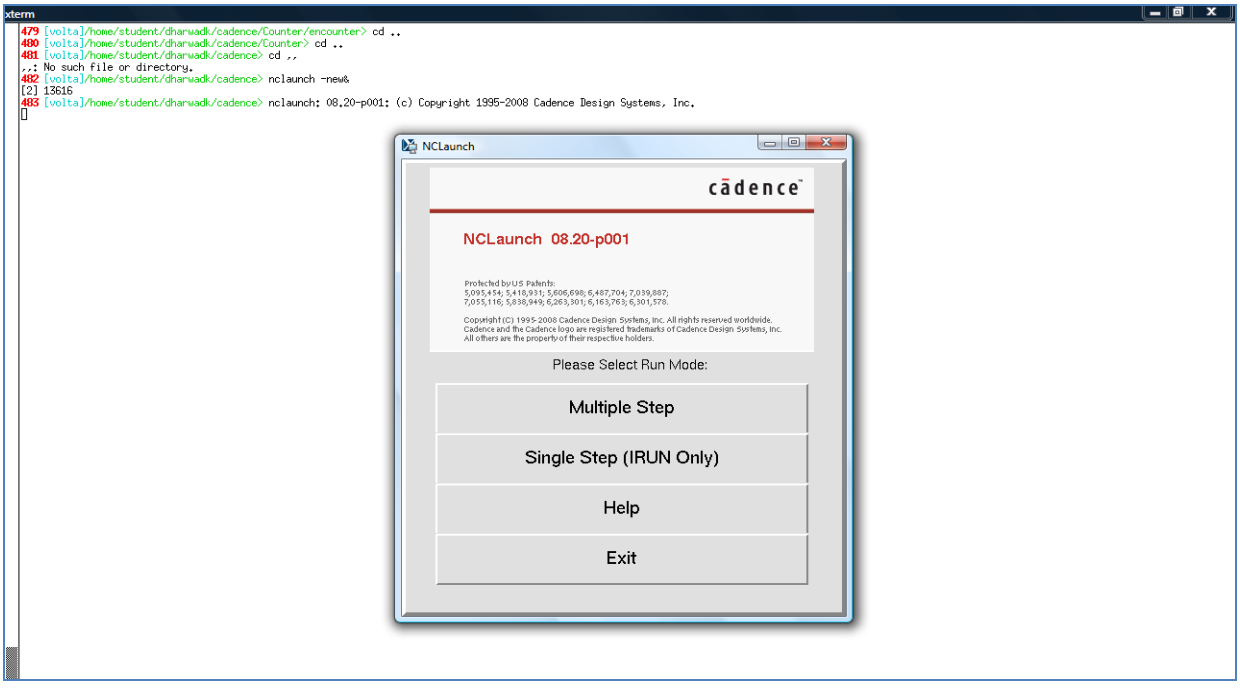
```
>> verilog tcb013ghp.v Counter.vh Counter_TB.v
```

Observe the output and see of it tallies with the desired output.
The second method to simulate the netlist file is to use NCVERILOG/NCSIM. This approach is explained in detail so as to familiarize users with NCVERILOG. The GUI for NCVERILOG can be invoked using the command

```
>> nclaunch -new&
```

This command has to be executed from the encounter subfolder. **–new** option is entered when invoking NCVERILOG for the first time for any given project(in this case it would be Counter project). On subsequent calls to NCVERILOG the following command has to be used

```
>> nclaunch &
```



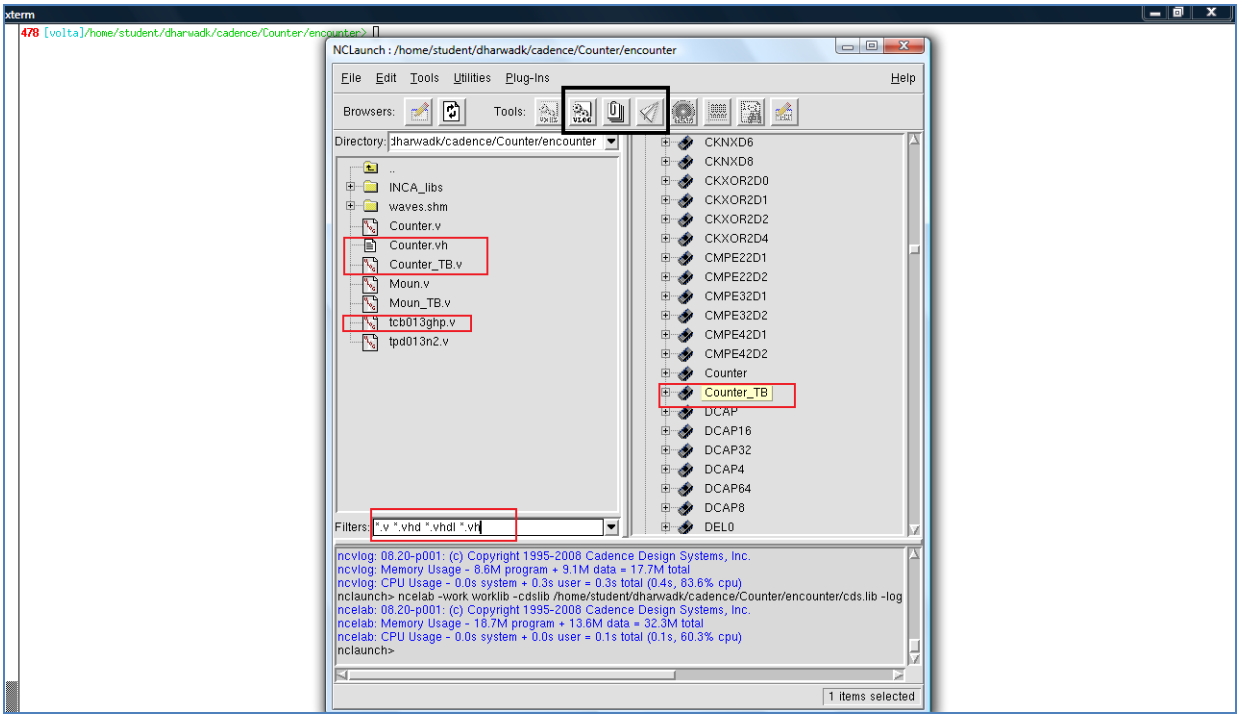
Select **Mutiple Step** option. Click on **Create cds.lib file** button and press **OK**. This will setup the environment for compiling and simulating the netlist file using NCVERILOG. The same approach could be followed for RTL simulation. The only difference between RTL and NETLIST simulation are the files that are compiled and simulated. In netlist simulation the netlist file(.vh) file is compiled and simulated where in RTL simulation the Verilog file is compiled and simulated.

The figure below highlights various regions in the tool that need to be explored by the user. When compiling/simulating a Verilog/netlist file care has to be taken while selecting the files. The order of selection is important. Always start with the file that has the lower-level components/modules and gradually move onto the files containing higher level modules. The testbench should always be the last file to be selected.

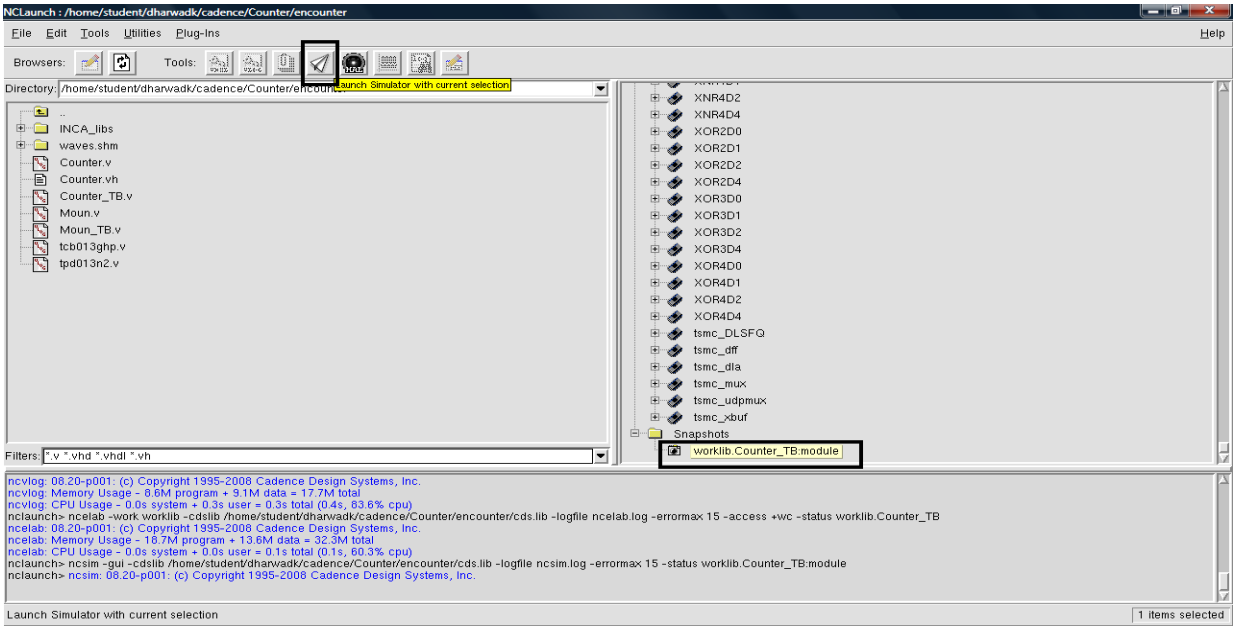
Note – The files in the left pane of the window may not be arranged in order. It is the job of the user to select the files in the right order and subject them for compilation/simulation.

Upon selecting the files, click on the **Tools** menu item and select the **Verilog Compiler** option. Press **OK** in order to compile the netlist file and generate a **worklib** folder in the right pane. On expanding the **worklib** folder in the right pane users should be able to see two links. One carries the name of the top module and another is the testbench module.

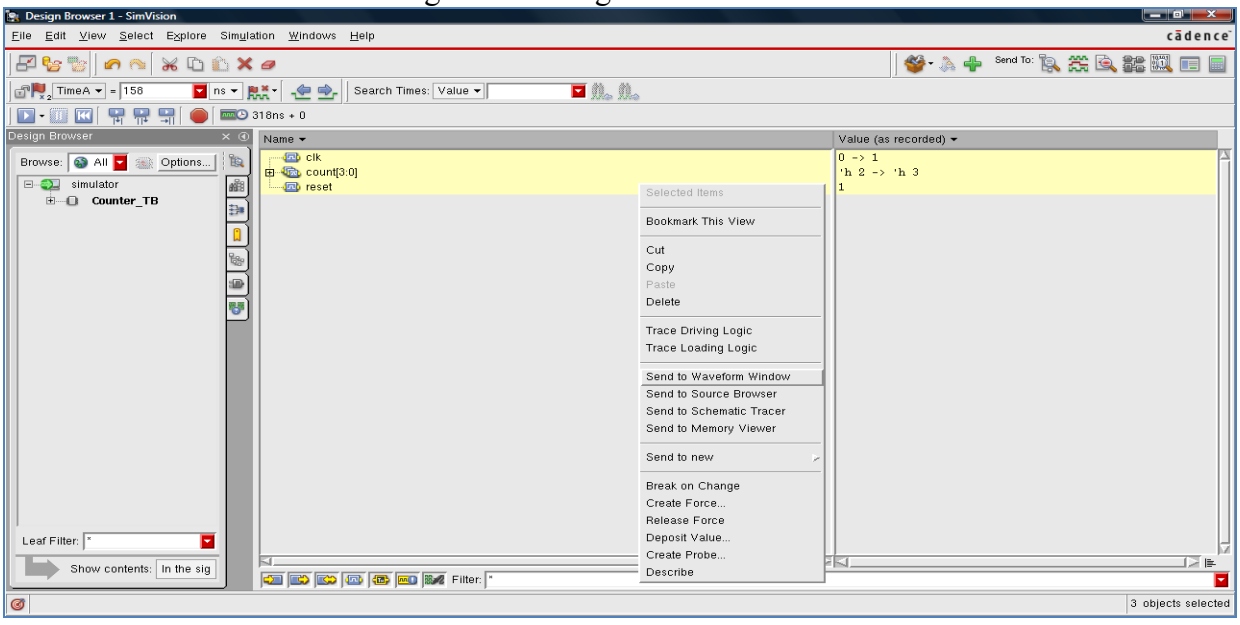
Select/Click on the testbench module link in the right pane and select the **ELABORATE** option from the **Tools** menu. Once the elaboration process is complete, in the right pane under the **snapshot** folder a snapshot is created for the testbench module.



The next step is to simulate the compiled netlist file and observe the waveforms. NCSIM is used for simulation. Simvision is used to observe the waveforms. In order to invoke simvision select the snapshot and click on the arrow button as shown in the figure below.



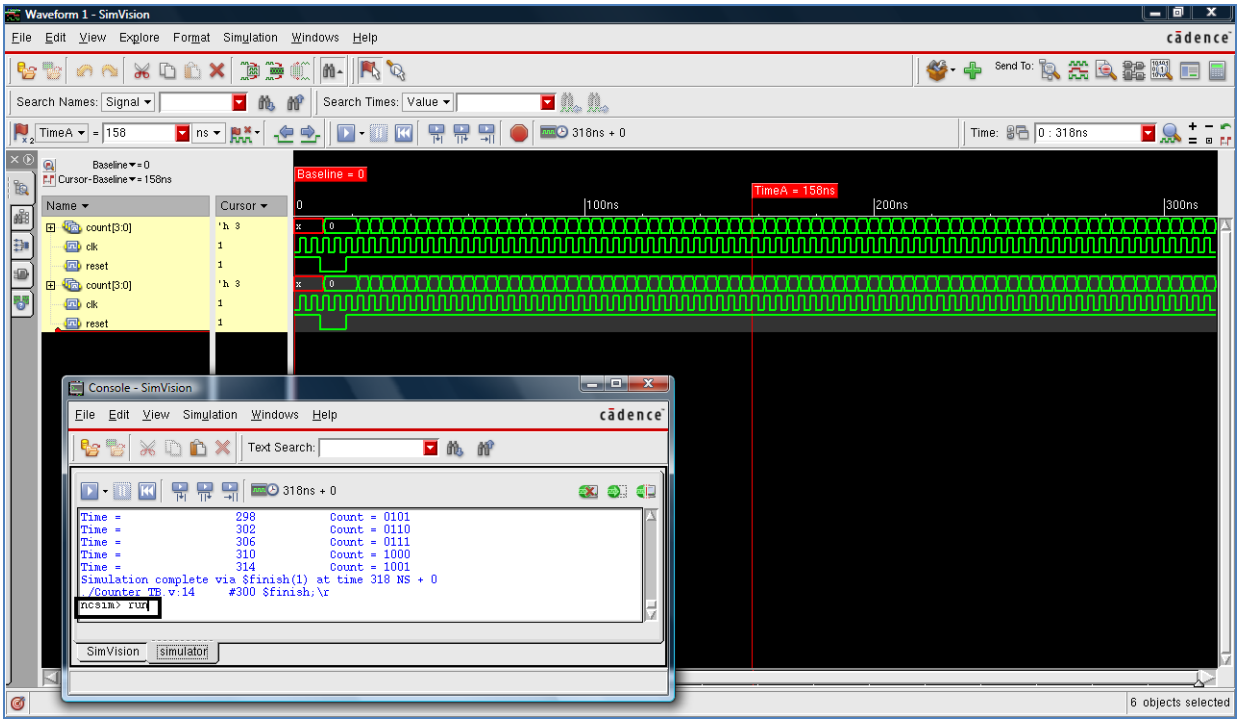
The Simvision tool, upon being invoked, opens in a new terminal. From the left pane select the module for which the waveforms need to be generated. From the right pane highlight all the nets for which waveforms need to be generated. Right click and select “Send to waveform window”.



This will invoke the waveform window. In order to observe the waveforms run the simulation by typing the command

>> **run**

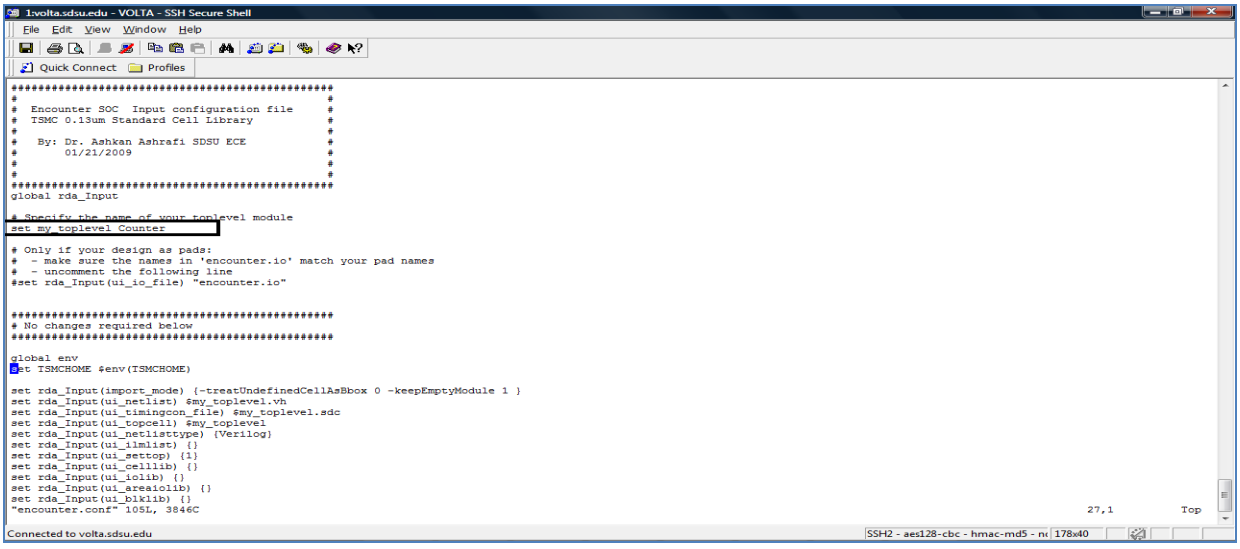
in the simulation console.



CHAPTER 5

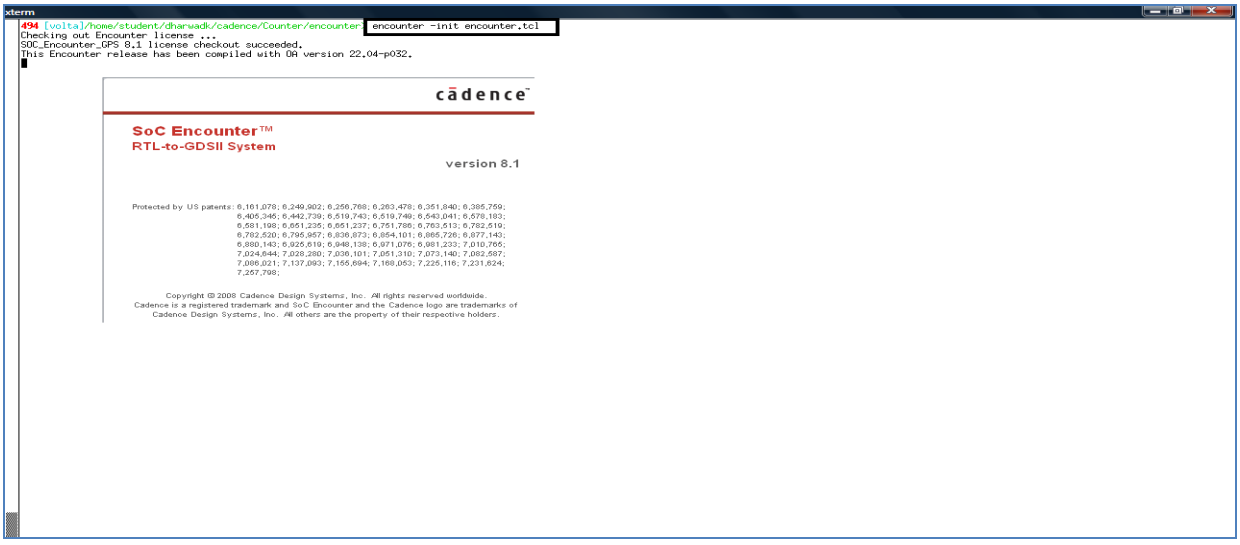
Layout Design

We started with RTL simulation, then synthesized the Verilog code into gate-level netlist, performed post-synthesis simulation and finally verified the behavior of the circuit by generating waveforms. The last step in the design is to auto place and route the standard gates. We use ENCOUNTER tool for this purpose. The **`encounter.tcl`** script invokes the ENCOUNTER tool and does auto place and route. The **`encounter.tcl`** script reads the **`encounter.conf`** which has all the details about layouting (number of metal layers, VDD, GND). The **`encounter.conf`** file has to be modified so as to refer to the synthesized netlist file.

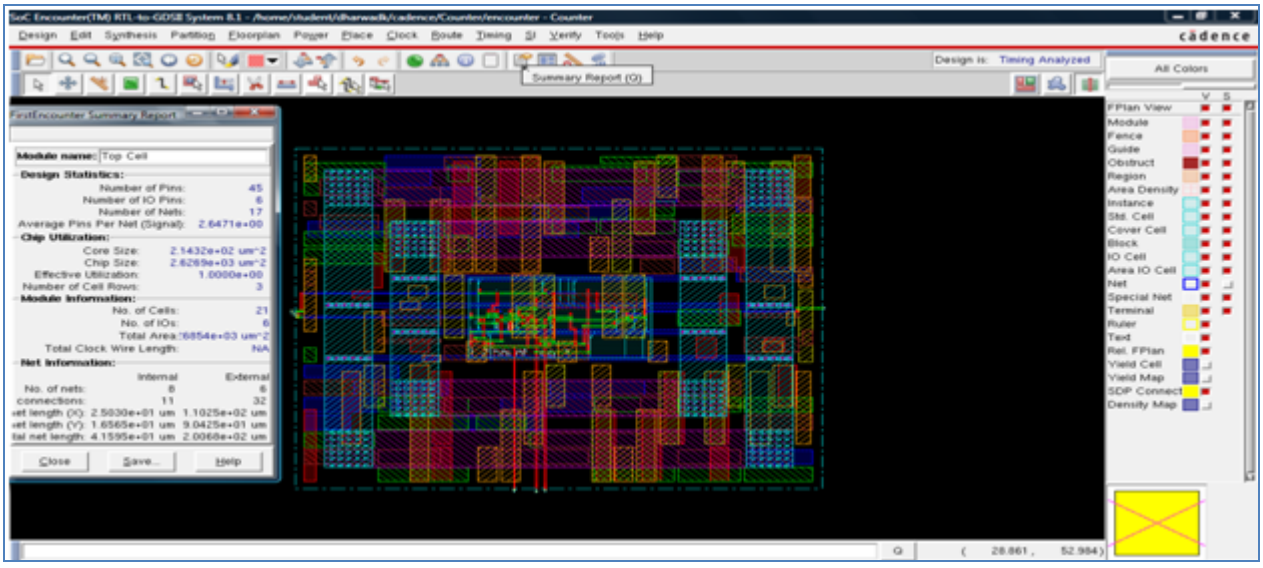


Before invoking the SOC ENCOUNTER tool ensure that **`gds2_encounter.map`** file is present in the encounter folder. The SOC ENCOUNTER tool can be invoked using the command,

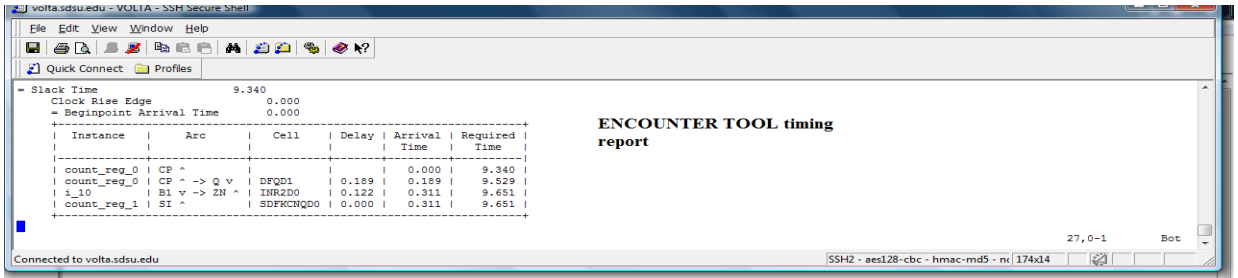
```
>> encounter -init encounter.tcl
```



The figure given below is that of the layout generated by the auto place and route functionality of SOC ENCOUNTER tool for our circuit.

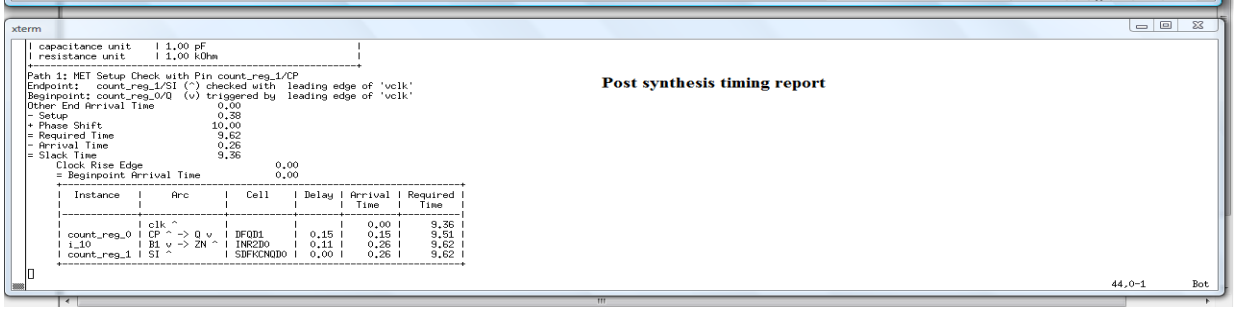


On the left side is the summary report obtained by clicking on the "summary report" button on the top. The summary report carries details about area utilization. The encounter.tcl script also generates a timing report bearing the name **`timing_final.rep`**. This has details of the critical path, slack time etc. The post synthesis (generation of the netlist file using compile_bgx.scr script) too resulted in a timing report getting generated. Upon comparing both the results it can be observed that the report generated by SOC ENCOUNTER will incorporate delays introduced due to interconnections and parasitic capacitances along with propagation delays through standard gates. The post-synthesis timing report however, considers only propagation delays through standard gates.



```
- Slack Time                    9.340
    Clock Rise Edge                     0.000
    = Beginpoint Arrival Time           0.000
    +-------------------------------------------------------------------------+
    |  Instance    |      Arc       |   Cell     | Delay | Arrival | Required |
    |              |                |            |       |  Time   |   Time   |
    |--------------+----------------+------------+-------+---------+----------|
    | count_reg_0  | CP ^           |            |       |   0.000 |    9.340 |
    | count_reg_0  | CP ^ -> Q v    | DFQD1      | 0.189 |   0.189 |    9.529 |
    | i_10         | B1 v -> ZN ^   | INR2D0     | 0.122 |   0.311 |    9.651 |
    | count_reg_1  | SI ^           | SDFKCNQD0  | 0.000 |   0.311 |    9.651 |
    +-------------------------------------------------------------------------+
```

ENCOUNTER TOOL timing report



```
| capacitance unit     | 1.00 pF                          |
| resistance unit      | 1.00 kOhm                        |
+-----------------------------------------------------------+
Path 1: MET Setup Check with Pin count_reg_1/CP
Endpoint:   count_reg_1/SI (^) checked with  leading edge of 'vclk'
Beginpoint: count_reg_0/Q  (v) triggered by  leading edge of 'vclk'
Other End Arrival Time          0.00
- Setup                         0.38
+ Phase Shift                  10.00
= Required Time                 9.62
- Arrival Time                  0.26
= Slack Time                    9.36
    Clock Rise Edge                     0.00
    = Beginpoint Arrival Time           0.00
    +-------------------------------------------------------------------------+
    |  Instance    |      Arc       |   Cell     | Delay | Arrival | Required |
    |              |                |            |       |  Time   |   Time   |
    |--------------+----------------+------------+-------+---------+----------|
    |              | clk ^          |            |       |    0.00 |     9.36 |
    | count_reg_0  | CP ^ -> Q v    | DFQD1      |  0.15 |    0.15 |     9.51 |
    | i_10         | B1 v -> ZN ^   | INR2D0     |  0.11 |    0.26 |     9.62 |
    | count_reg_1  | SI ^           | SDFKCNQD0  |  0.00 |    0.26 |     9.62 |
    +-------------------------------------------------------------------------+
```

Post synthesis timing report