Digital Labs

Revision 3.0

Incisive Unified Simulator 9.2/10.2
RTL Compiler 10.1
Encounter Digital Implementation 9.1/11.1

Developed by,

University Support Team
Cadence, Bangalore

Table of Contents

General Notes

There are a number of things to consider before beginning these lab exercises. Please read through this section completely, and perform any needed steps in order to ensure a successful workshop. These labs were designed for use with Incisive Unified Simulator.

Before running any of these labs, ensure that you've set up IUS correctly:

```
%> setenv IUSHOME <IUS-installation-home>
```

The Cadence_Digital_labs directory contains Solutions folder and also Work folder. Inside Work folder you can make modifications of the code locally without affecting your Source code present inside Solutions directory.

Lab directory details:

. /Solutions	Contains a local copy of all the lab experiments including Testbences for simulating the codes.
. /Workarea	It's a place to run Simulation and Synthesis.

To setup the lab environment, please perform the following steps:

1. Ensure the software mentioned above is correctly setup.
2. Source the C-Shell related commands file i.e (cshrc file).

These labs were designed to be run using Cadence Simulator and the synthesis engine.

Lab1: An Inverter

In this lab we will simulate the inverter code modeled using switch level by the help of Incisive unified simulator.In this lab we will see how to perform simulation in command mode using testbench without using GUI window.

1. Change directory to Cadence_Digital_labs/Workarea/Inverter

2. You will need to copy each file present in solutions folder to Workarea/Inverter location by using the below mentioned command :

   cp -rf ../../Solutions/Inverter/* .

3. View the Code of Inverter and also the testbench for the same.

4. **Compile the source Descriptions :**

   (i) Compile the Inverter description with the -messages option:

   ncvlog inverter.v –messages

```
[bsatish@blropt05 Inverter]$ ncvlog inverter.v -messages
ncvlog: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
file: inverter.v
        module worklib.inverter:vlog
                errors: 0, warnings: 0
```

The compiler places the *inverter* description in the *INCA_libs* library.

(ii) Compile the testbench description with the -MESS option:

ncvlog inverter_test.v –MESS

```
[bsatish@blropt05 Inverter]$ ncvlog inverter_test.v -MESS
ncvlog: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
file: inverter_test.v
        module worklib.inv_test:vlog
                errors: 0, warnings: 0
```

The compiler places the *inverter_test* description in the *INCA_libs* library.

Note: You can abbreviate options down to their shortest unique string and use upper or Lower case.

*List all "inverter" library objects (**ncls inverter**).*

```
[bsatish@blropt05 Inverter]$ ncls inverter
ncls: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
        module worklib.inverter:vlog (VST)
```

What library objects does the compiler create?
Answer: The compiler creates Verilog VST objects.

5. **Elaborate the top level Design**

(i) Elaborate the testbench

ncelab inv_test -messages

The elaborator places the *inv_test* code and snapshot in the *INCA_libs* library.

```
[bsatish@blropt05 Inverter]$ ncelab inv_test -messages
ncelab: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
        Elaborating the design hierarchy:
                Caching library 'worklib' ....... Done
        Building instance overlay tables: ..................... Done
        Generating native compiled code:
                worklib.inv_test:vlog <0x505a8bd2>
                        streams:   3, words:  1862
        Loading native compiled code:     ..................... Done
        Building instance specific data structures.
        Design hierarchy summary:
                                Instances  Unique
                Modules:                2       2
                Primitives:             2       2
                Registers:              1       1
                Scalar wires:           4       -
                Initial blocks:         1       1
                Pseudo assignments:     1       1
                Simulation timescale:  1ns
        Writing initial simulation snapshot: worklib.inv_test:vlog
```

List all “inv_test" library objects (**ncls inv_test**).

What library objects does the elaborator create?
Answer: The elaborator creates SIG (signature), HDLCODE (code),
and SSS (snapshot) objects.

```
[bsatish@blropt05 Inverter]$ ncls inv_test
ncls: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
        module worklib.inv_test:vlog (VST)
        module worklib.inv_test:vlog (SIG) <0x505a8bd2>
        module worklib.inv_test:vlog (COD) <0x505a8bd2>
        snapshot worklib.inv_test:vlog (SSS)
```

6. Simulate the Top-Level Design

(i) Simulate the testbench:

ncsim inv_test

The simulator displays results similar to the following:

```
[bsatish@blropt05 Inverter]$ ncsim inv_test
ncsim: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
ncsim> run
time=10 ns Input=0 Output=1
time=20 ns Input=1 Output=0
time=30 ns Input=x Output=x
time=40 ns Input=z Output=x
ncsim: *W,RNQUIE: Simulation is complete.
ncsim> exit
```

Lab Summary:

In this lab we saw how to compile, elaborate and simulate the tesbench for Inverter module.

Lab 2: A Buffer

In this lab, you will simulate a design using the Incisive simulator. You will:

* Create the *cds.lib* and *hdl.var* files
* Compile, elaborate, and simulate the design and testbench

Perform this lab in the *Buffer* directory. This directory contains the following files (which you should briefly examine) describing a simple Buffer and its testbench:

File(s) Description

Buffer.v Buffer code
Buffer_test.v Testbench

1. Change directory to Cadence_Digital_labs/Workarea/Buffer.
2. You will need to copy each file present in solutions folder to Workarea/Buffer location by using the below mentioned command :

cp -rf ../../Solutions/Buffer/* .

3. View the Code of Inverter and also the testbench for the same.

4 **Set Up the Design Environment**

Using your favorite text editor, create the *cds.lib* file and make the following entries:

Define Buffer_lib ./Buffer.lib

Create the local library directory:

mkdir Buffer.lib

Create the *hdl.var* file and make the following entry:

Define WORK Buffer_lib

```
[bsatish@blropt05 Buffer]$ ls
Buffer.lib  buffer.v  buffer_test.v  cds.lib  hdl.var
```

5. **Compile the Source Descriptions**

(i). Compile the buffer description with the `-messages` option:

ncvlog buffer.v –messages

```
[bsatish@blropt05 Buffer]$ ncvlog buffer.v -messages
ncvlog: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
file: buffer.v
        module Buffer_lib.inverter:vlog
                errors: 0, warnings: 0
        module Buffer_lib.buffer:vlog
                errors: 0, warnings: 0
```

The compiler places the *Buffer* description in the *Buffer_lib* library.

(ii). Compile the testbench description with the -MESS option:

ncvlog buffer_test.v –MESS

```
[bsatish@blropt05 Buffer]$ ncvlog buffer_test.v -MESS
ncvlog: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
file: buffer_test.v
        module Buffer_lib.buf_test:vlog
                errors: 0, warnings: 0
```

The compiler places the *buffer_test* description in the *Buffer_lib* library.

Note: You can abbreviate options down to their shortest unique string and use upper or lower case.

*List all "mux" library objects (**ncls buffer**).*

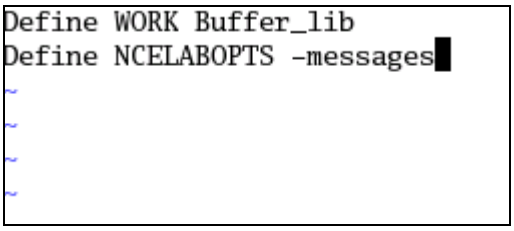
```
[bsatish@blropt05 Buffer]$ ncls buffer
ncls: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
        module Buffer_lib.buffer:vlog (VST)
```

What library objects does the compiler create?
Answer: The compiler creates Verilog VST objects.

6. Elaborate the Top-Level Design

1. Append this line to the *hdl.var* file:

Define NCELABOPTS –messages



2. Elaborate the testbench:

ncelab buf_test

The elaborator places the *buffer_test* code and snapshot in the *Buffer_lib* library.

```
[bsatish@blropt05 Buffer]$ ncelab buf_test
ncelab: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
        Elaborating the design hierarchy:
                Caching library 'Buffer_lib' ....... Done
        Building instance overlay tables: ..................... Done
        Generating native compiled code:
                Buffer_lib.buf_test:vlog <0x505a8bd2>
                        streams:   3, words:  1862
        Loading native compiled code:     ..................... Done
        Building instance specific data structures.
        Design hierarchy summary:
                                Instances  Unique
                Modules:                4       3
                Primitives:             4       2
                Registers:              1       1
                Scalar wires:           5       -
                Initial blocks:         1       1
                Pseudo assignments:     1       1
                Simulation timescale:  1ns
        Writing initial simulation snapshot: Buffer_lib.buf_test:vlog
```

7. **Simulate the Top-Level Design**

(i) Simulate the testbench:

ncsim buf_test

The simulator displays results similar to the following:

```
[bsatish@blropt05 Buffer]$ ncsim buf_test
ncsim: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
ncsim> run
time=10 ns Input=0 Output=0
time=20 ns Input=1 Output=1
time=30 ns Input=x Output=x
time=40 ns Input=z Output=x
ncsim: *W,RNQUIE: Simulation is complete.
ncsim> exit
```

Lab Summary

In this lab, you simulated a design using the Incisive simulator. You:
* Created the *cds.lib* and *hdl.var* files
* Compiled, elaborated, and simulated the design and testbench

Lab 3: Transmission Gate

In this lab, you will simulate a design using the Incisive simulator. You will:

* Create the *cds.lib* and *hdl.var* files
* Compile, elaborate, and simulate the design and testbench

Perform this lab in the *TG* directory. This directory contains the following files (which you should briefly examine) describing a simple Transmission gate and its testbench:

File(s) Description

tg.v -- Transmission Gate code
tg_test.v -- Testbench

1. Change directory to Cadence_Digital_labs/Workarea/TG.

2. You will need to copy each file present in solutions folder to Workarea/TG location by using the below mentioned command :

   cp -rf ../../Solutions/TG/* .

3. View the Code of tranmission gate and also the testbench for the same.

4. **Set Up the Design Environment**

   Using your favorite text editor, create the *cds.lib* file and make the following entries:

   Define tg_lib ./tg.lib

   Create the local library directory:

   mkdir tg.lib

   Create the *hdl.var* file and make the following entry:

   Define WORK tg_lib

```
[bsatish@blropt05 TG]$ ls
cds.lib  hdl.var  tg.lib  tg.v  tg_test.v
```

5. **Compile the Source Descriptions**

(i). Compile the transmission gate description with the `-messages` option:

ncvlog tg.v –messages

```
[bsatish@blropt05 TG]$ ncvlog tg.v -messages
ncvlog: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
file: tg.v
        module tg_lib.trangate:vlog
                errors: 0, warnings: 0
```

The compiler places the *Transmission Gate* description in the *tg_lib* library.

(ii). Compile the testbench description with the -MESS option:

ncvlog tg_test.v –MESS

```
[bsatish@blropt05 TG]$ ncvlog tg_test.v -MESS
ncvlog: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
file: tg_test.v
        module tg_lib.trangate_test:vlog
                errors: 0, warnings: 0
```

The compiler places the *tg_test* description in the *tg_lib* library.

Note: You can abbreviate options down to their shortest unique string and use upper or lower case.

6. **Elaborate the Top-Level Design**

(i) Append this line to the *hdl.var* file:

Define NCELABOPTS –messages

```
Define WORK tg_lib
Define NCELABOPTS -messages
~
~
~
~
~
```

(ii) Elaborate the testbench:

ncelab trangate_test -access +rwc

The elaborator places the *tg_test* code and snapshot in the *tg_lib* library.

```
[bsatish@blropt05 TG]$ ncelab trangate_test -access +rwc -MESS
ncelab: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
        Elaborating the design hierarchy:
                Caching library 'tg_lib' ....... Done
        Building instance overlay tables: ..................... Done
        Generating native compiled code:
                tg_lib.trangate_test:vlog <0x00b3b171>
                        streams:   5, words:  3364
        Loading native compiled code:     ..................... Done
        Building instance specific data structures.
        Design hierarchy summary:
                                Instances  Unique
                Modules:                2       2
                Primitives:             2       2
                Registers:              3       3
                Scalar wires:           4       -
                Initial blocks:         1       1
                Pseudo assignments:     3       3
                Simulation timescale:  1ns
        Writing initial simulation snapshot: tg_lib.trangate_test:vlog
```

7. **Simulate the top level design:**

(i) Simulate the testbench: with –gui option:

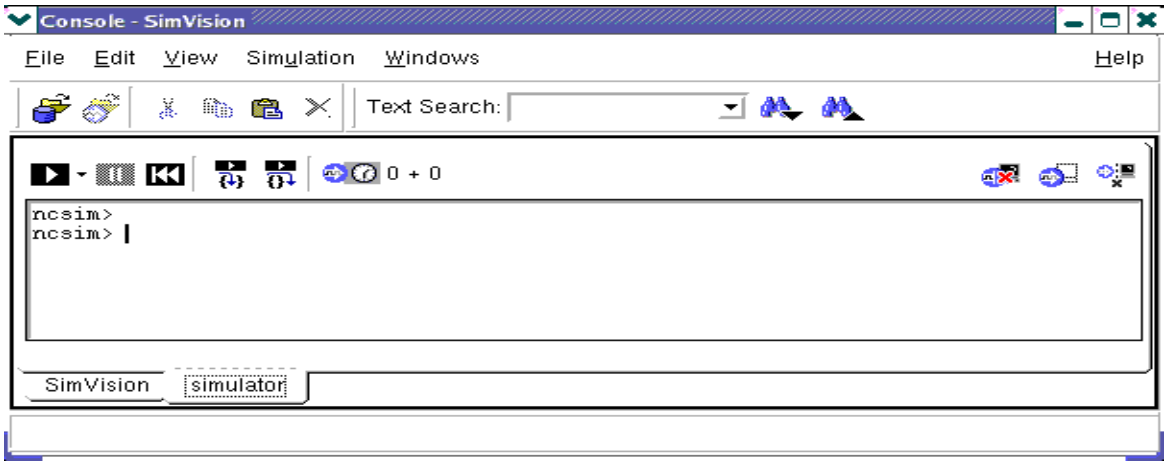
ncsim trangate_test –gui

The -gui option opens the **Console** and **Design Browser** windows.

Tour the Graphical Interface

1. Examine the Console window.

a. You can use the Menu Bar to run or step the simulation, set scopes and stops, show the value of objects, and start other graphical tools.
b. You can use the Tool Bar to run, interrupt, reset, step, or next the simulation, and shut down the interface or the simulation, or disconnect the simulation.
c. You can use the command line interface to the simulation in the I/O Region.



2. Examine the Design Browser window.

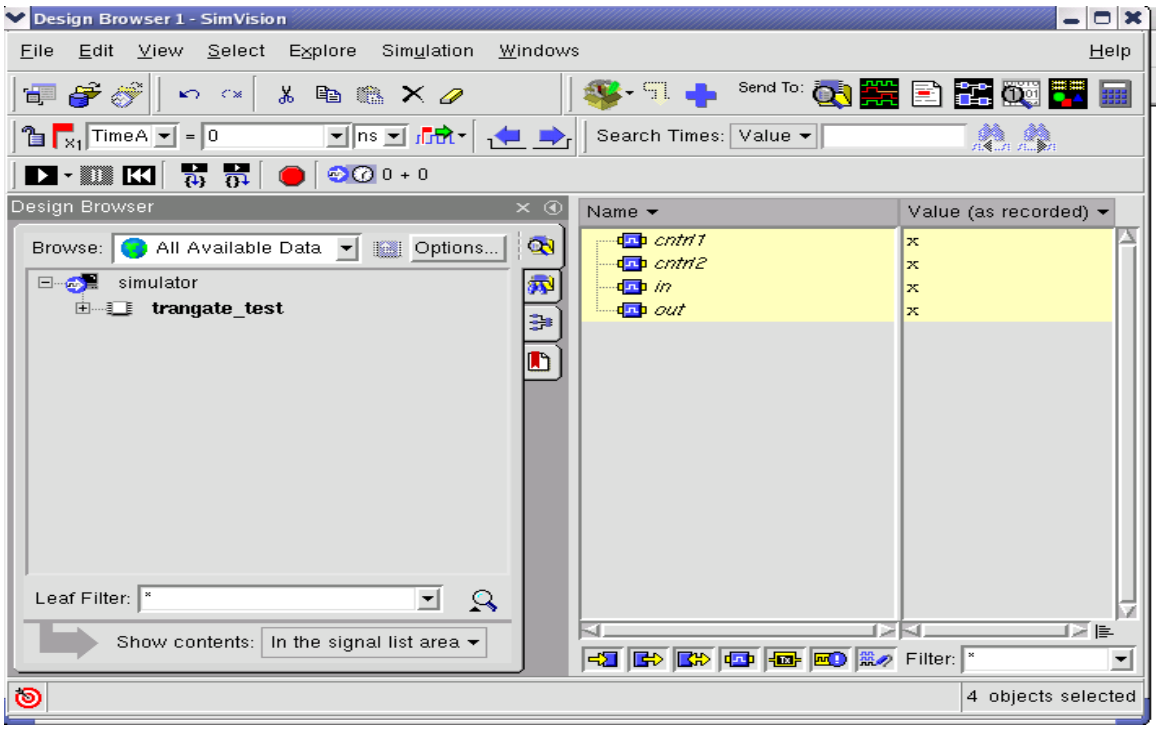
Open an existing Design Browser window or select the Windows— new — Design Browser menu item or the Design Browser button.

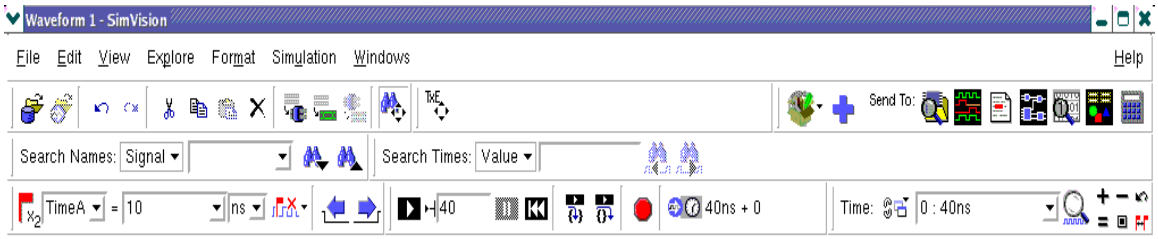
a. Display the objects of a scope and their value in the Objects List pane (Select any displayed scope in the Scope Tree pane).
b. Display the component instances of the scope (double-click the scope in the Scope Tree pane).

3. Tour the **Waveform** window.

Open an existing Waveform window or select **the Windows - New - Waveform** menu item or the **Waveform** button.The simulator creates a default SHM database and sets a probe onany selected signals and opens a Waveform window displaying the selected signals.

a. In the Design Browser window select all signals at the *trangate_test* scope.
b. Add the selected signal(s) to the Waveform window (select the **Waveform** button or the **Add Selected** button or drag and drop the signals into the Waveform window).**Note:** To add additional signals simply select them in any window and click the ***Waveform*** button again.





Examine the Design and Testbench Hierarchy

In this section of the lab you visit the Source Browser, Schematic Tracer and Waveform window.

1. In the Design Browser window select the top-level (mem_test) scope and select the Source Browser button to send it to the target Source Browser window. As no such window yet exists, this opens a Source Browser window displaying the source of the Top-level unit, and makes it the default Source Browser target window.

2. In the Source Browser window ensure that just the top-level scope is selected (navigate up as needed and Select—This Scope) and send it to the target Schematic Tracer

window. As no such window yet exists, this opens a Schematic Tracer window displaying the top-level unit, and makes it the default Schematic Tracer target window, in which you:

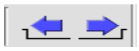
a. Ensure that the top-level scope is still selected, and select the fill Module  button, to display the testbench content.
b. Select the Edit - Select - All menus item, and again select the fill Module button, to Expand the second level content.
c. Select the Zoom Full  button to fit all displayed elements.

3. In the Source Browser window ensure that just the top-level scope is selected and send it  tothe target Waveform window. As no such window yet exists, this opens a Waveform window displaying the signals of the top-level unit, and makes it the default Waveform target window.

a. In the left sidebar, select the Design Browser  tab toexpand the sidebar area and display the embedded Design Browser.

--- Run the simulation until the next breakpoint, or for the Duration entered in the time field (i.e 40ns).

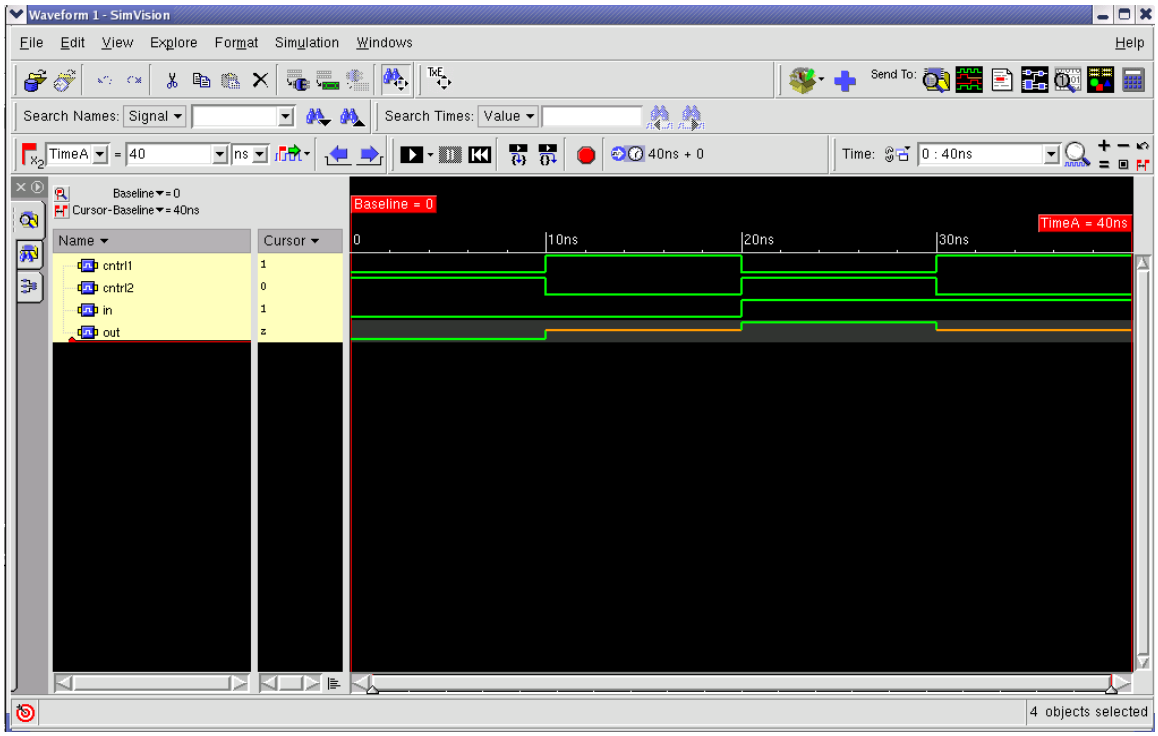
--- Current Time range.

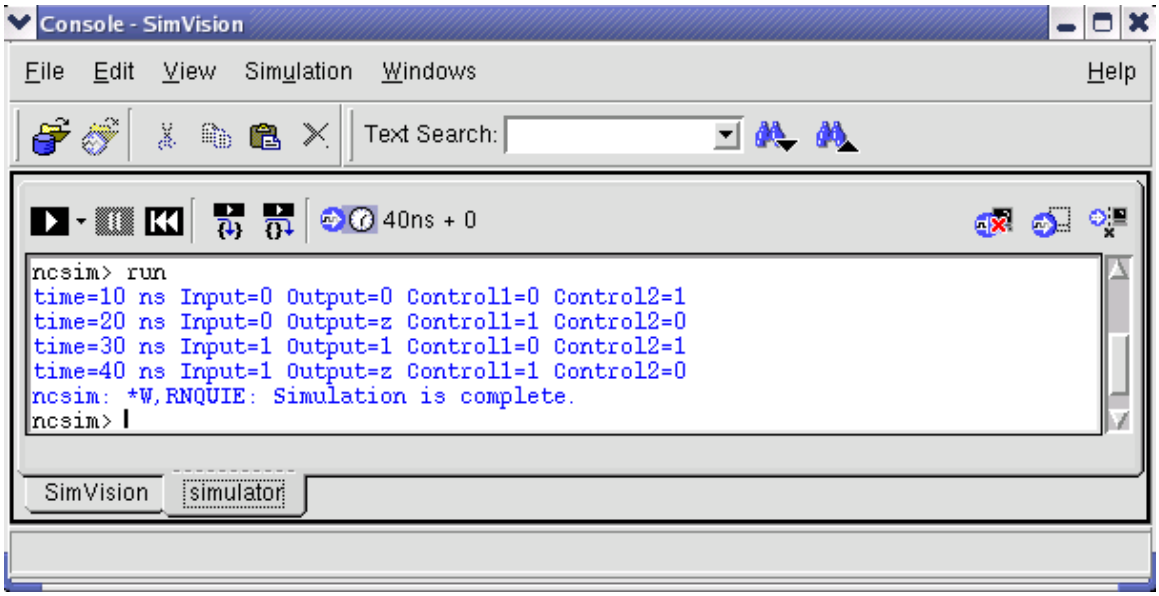


--- Move primary cursor to previous edge of select signal.

--- Reset the simulation back to time “ 0 “.

Once the simulation is done you can see the following waveform window and console window with the outputs.


Waveform 1 - SimVision
File Edit View Explore Format Simulation Windows Help
Search Names: Signal
Search Times: Value
Send To:
TimeA = 40 ns
40ns + 0
Time: 0 : 40ns
Baseline = 0
Cursor-Baseline = 40ns
Name
Cursor
cntrl1
cntrl2
in
out
Baseline = 0
TimeA = 40ns
0
10ns
20ns
30ns
4 objects selected


Console - SimVision
File Edit View Simulation Windows Help
Text Search:
40ns + 0
ncsim> run
time=10 ns Input=0 Output=0 Control1=0 Control2=1
time=20 ns Input=0 Output=z Control1=1 Control2=0
time=30 ns Input=1 Output=1 Control1=0 Control2=1
time=40 ns Input=1 Output=z Control1=1 Control2=0
ncsim: *W,RNQUIE: Simulation is complete.
ncsim>
SimVision
simulator

Lab 4: Basic / Universal Gates

In this lab, you will simulate a design using the Incisive simulator. You will:

* Create the *cds.lib* and *hdl.var* files
* Compile, elaborate, and simulate the design and testbench

Perform this lab in the *Gates directory*. This directory contains the following subdirectories (which you should briefly examine) describing a simple basic/universal gates and its testbench:

Subdirectories Descriptions:

AND , NAND , NOR , OR , XOR , XNOR

1 Change directory to Cadence_Digital_labs/Workarea/Gates.

2 You will need to copy each file present in Solutions folder to Workarea/Gates location by using the below mentioned command :

cp -rf ../../Solutions/Gates/* .

3 Change directory to AND folder.

4 View the Code of AND gate and also the testbench for the same.

5. This procedure is same for all the other logic gates present inside Gates folder.

6. **Set Up the Design Environment**

Using your favorite text editor, create the *cds.lib* file and make the following entries:

Define and_lib ./and.lib

Create the local library directory:

mkdir and.lib

Create the *hdl.var* file and make the following entry:

Define WORK and_lib

```
[bsatish@blropt05 AND]$ ls
and.lib  and.v  and_test.v  cds.lib  hdl.var
[bsatish@blropt05 AND]$
```

7. **Compile the Source Descriptions**

(i). Compile the transmission gate description with the `-messages` option:

ncvlog and.v –messages

```
[bsatish@blropt05 AND]$ ncvlog and.v -messages
ncvlog: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
file: and.v
        module and_lib.andgate:vlog
                errors: 0, warnings: 0
```

The compiler places the *And Gate* description in the *and_lib* library.

(ii). Compile the testbench description with the -MESS option:

ncvlog and_test.v –MESS

```
[bsatish@blropt05 AND]$ ncvlog and_test.v -MESS
ncvlog: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
file: and_test.v
        module and_lib.and_test:vlog
                errors: 0, warnings: 0
```

The compiler places the *and_test* description in the *and_lib* library.

Note: You can abbreviate options down to their shortest unique string and use upper or lower case.

8. **Elaborate the Top-Level Design**

(i) Append this line to the *hdl.var* file:

Define NCELABOPTS –messages

```
Define WORK tg_lib
Define NCELABOPTS -messages
~
~
~
~
~
```

(ii) Elaborate the testbench:

ncelab andgate -access +rwc -MESS

The elaborator places the *andgate* code and snapshot in the *and_lib* library.

```
[bsatish@blropt05 AND]$ ncelab andgate -access +rwc -MESS
ncelab: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
        Elaborating the design hierarchy:
        Building instance overlay tables: .................... Done
        Loading native compiled code:     .................... Done
        Building instance specific data structures.
        Design hierarchy summary:
                                Instances  Unique
                Modules:                1       1
                Primitives:             6       2
                Scalar wires:           6       -
                Simulation timescale:  1ns
        Writing initial simulation snapshot: and_lib.andgate:vlog
```

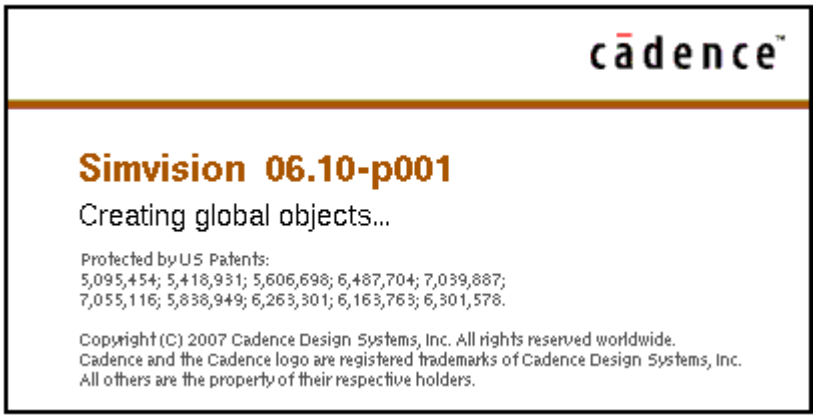
7. **Simulate the top level design:**

(ii) Simulate the design: with -gui option:

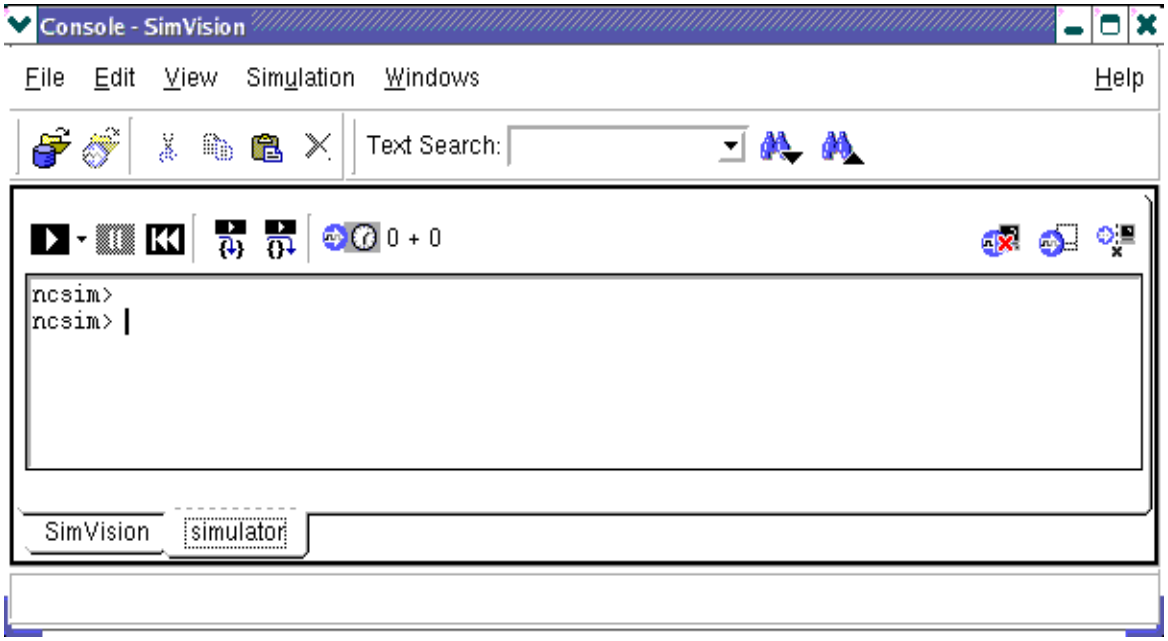
ncsim andgate –gui

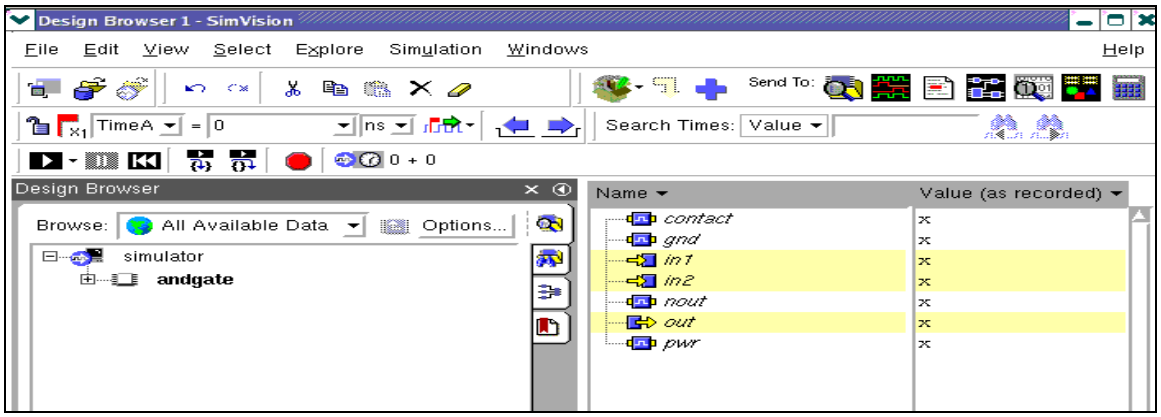
```
[bsatish@blropt05 AND]$ ncsim andgate -gui
ncsim: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
simvision: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
```

Simvision tool from Cadence Design Systems automatically pops up:

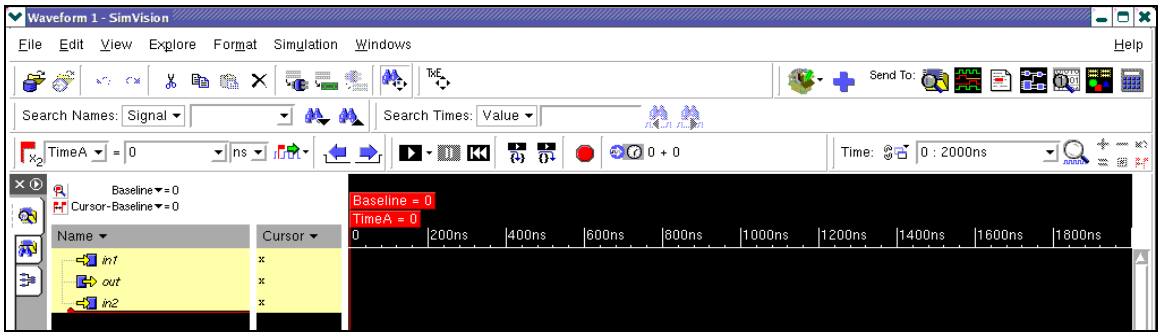


The -gui option opens the **Console** and **Design Browser** windows.

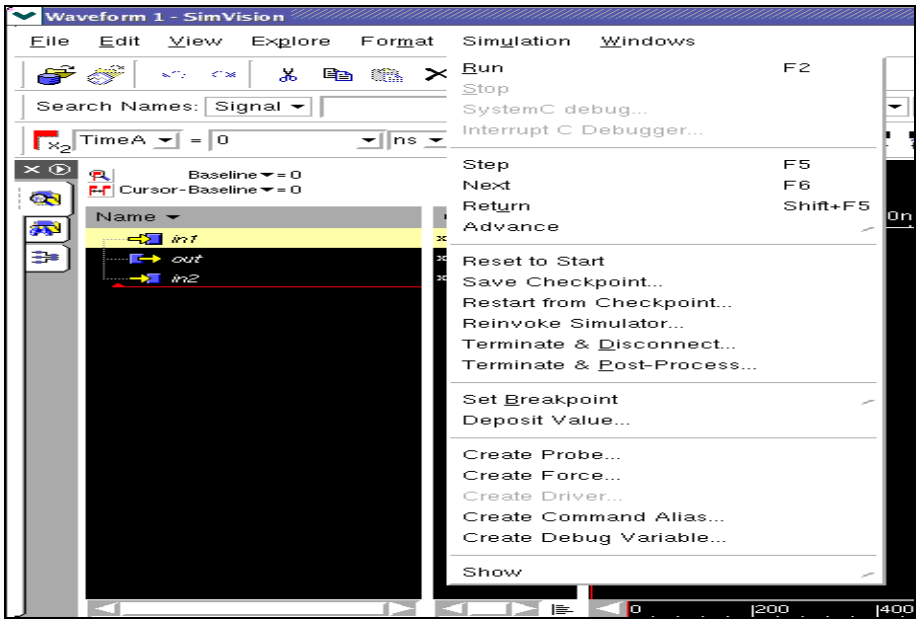




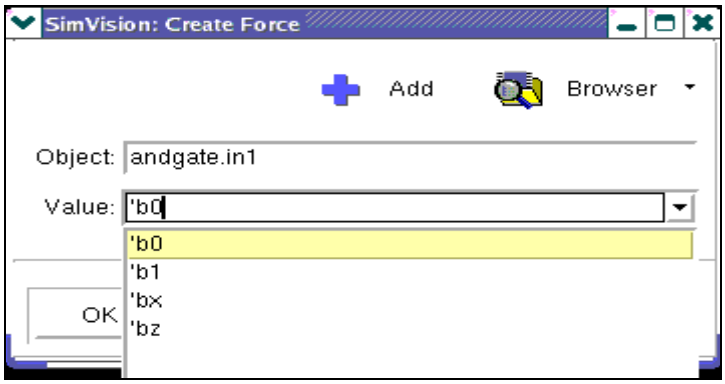
Selcet the input and output ports as shown above and click  button to open up the waveform window.

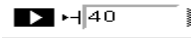


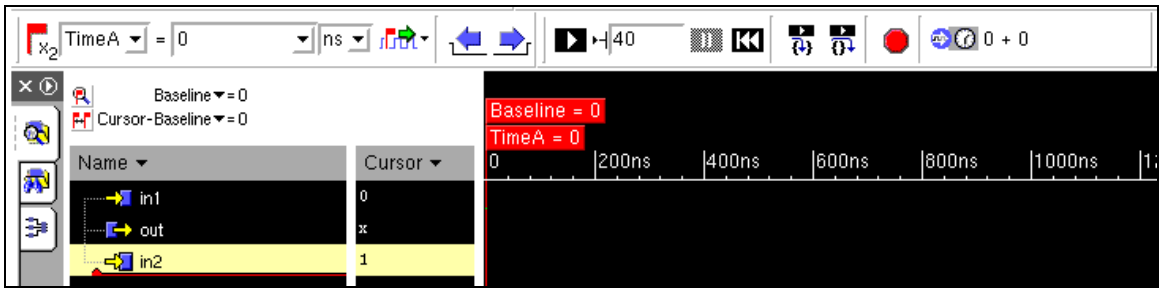
Select “in1” and go to Simulation tab and selcet “Create Force “option



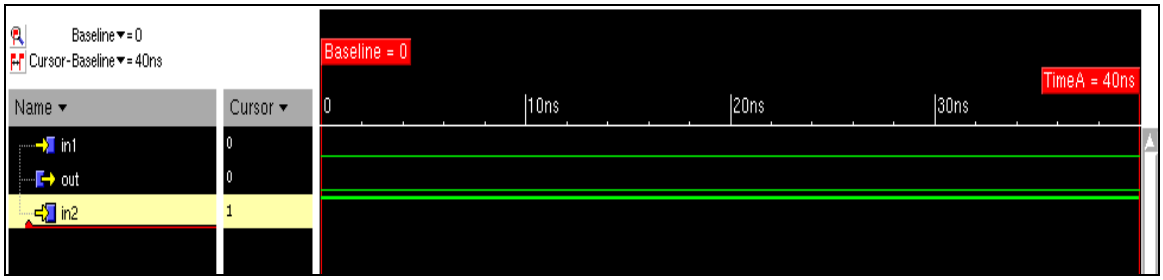
Following Create force window comes up .Now apply binary input logic to “in1”from the available binary logic – “0,1,X,Z”.Selcet any logic and click “Ok”.



In the same way apply another input “in2”.
Once the inputs are applied, select the time range for simulation and run the simulation using the following Icon 



Output Waveform is as shown below for AND gate :



Its also possible to reset the simulation with this button.  .

Once the reset is done and want to perform simulation with another set of inputs ,then follow the same procedure as mentioned ablove.

Lab 5: Flip-Flops

In this lab, you will simulate a design using the Incisive simulator. You will:

* Create the *cds.lib* and *hdl.var* files
* Compile, elaborate, and simulate the design and testbench

Perform this lab in the *Flipflops* directory. This directory contains the following files (which you should briefly examine) describing a Flipflops and its testbenches:

File(s) Description:

SR_ff.v d_ff.v jk_ff.v ms_ff.v t_ff.v

1. Change directory to Cadence_Digital_labs/Workarea/Flipflops

2: You will need to copy each file present in Solutions folder to Workarea/Flipflops location by using the below mentioned command :

cp -rf ../../Solutions/Flipflops/* .

3. View the Code of SR-Flipflop and also the testbench for the same.

4. **This procedure is same for all the other flipflops present inside Flipflops folder.**

5. **Set Up the Design Environment**

Using your favorite text editor, create the *cds.lib* file and make the following entries:

Define ff_lib ./ff.lib

Create the local library directory:

mkdir ff.lib

Create the *hdl.var* file and make the following entry:

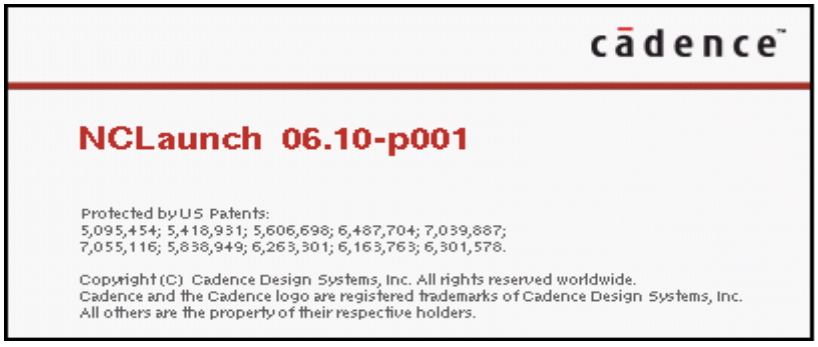
Define WORK ff_lib

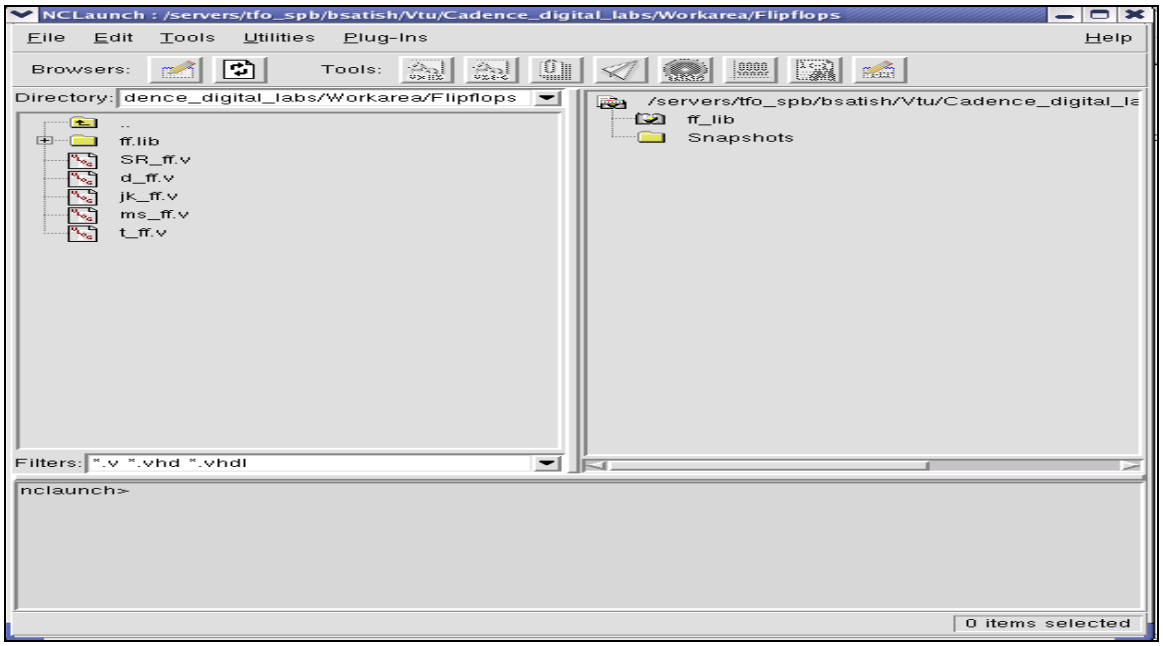
```
[bsatish@blropt05 Flipflops]$ ls
SR_ff.v  cds.lib  d_ff.v  ff.lib  hdl.var  jk_ff.v  ms_ff.v  t_ff.v
```

6. **Use the following command to invoke user friendly GUI:**

nclaunch &

Following windows appears.This is the GUI of nclaunch.
Rightside of window has ff_lib (worklib) and snapshots directories listed.
ff_lib is the directory where all the compiled codes are stored while snapshot will have output of elaboration which in turn goes for simulation.



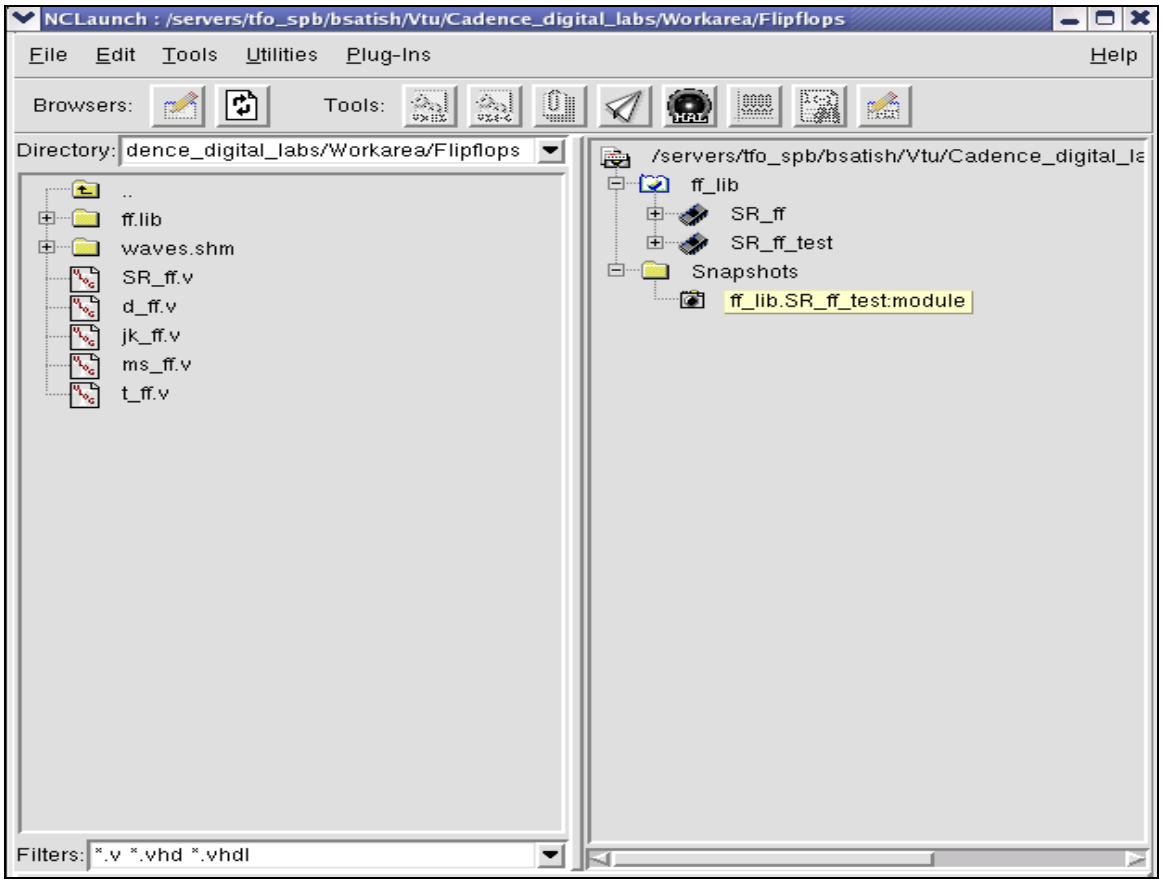


Once the above window appears select the flipflop of your choice for ex: I am selecting SR-ff.v .Once the flipflop is selected click the following buttons shown below for compilation, elaboration and simulation.

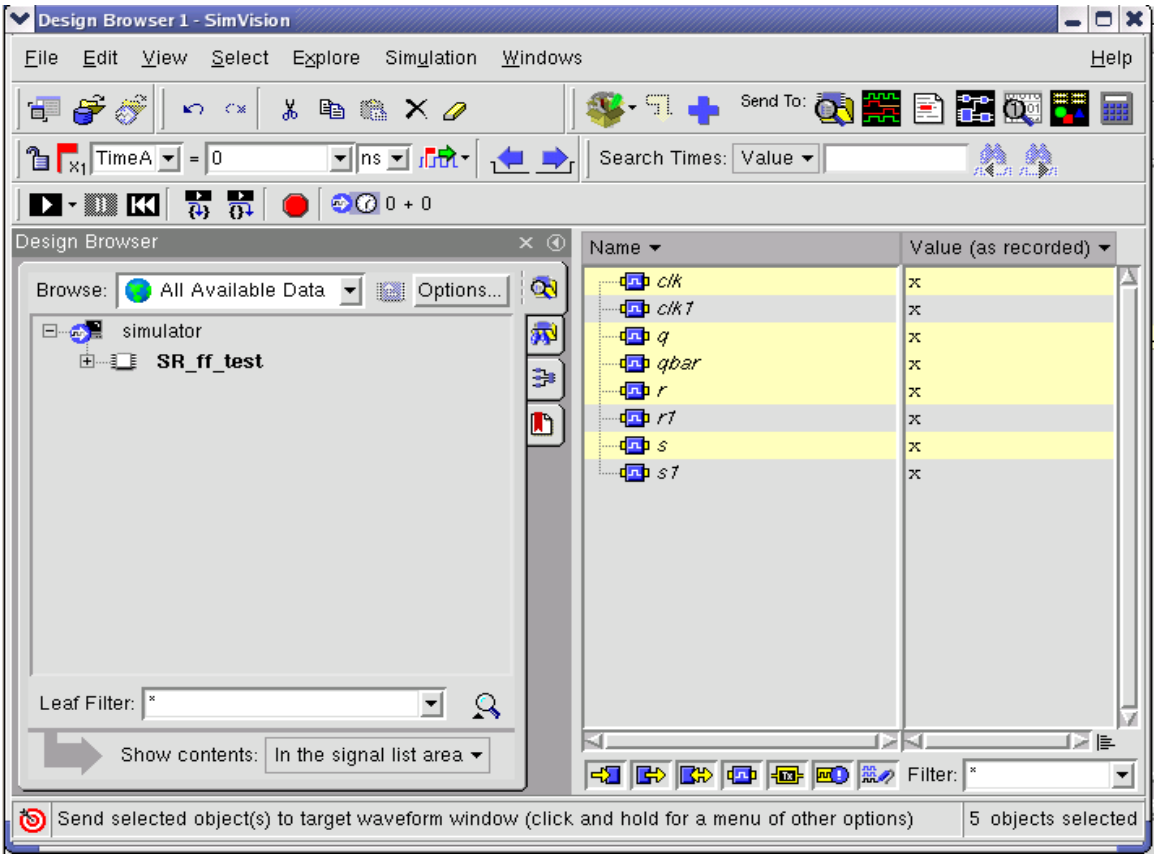
-- This Icon is for compiling the verilog code, once the compilation is done it Creates VST and you can see **SR_ff** and **SR_ff_test** under ff_lib directory.

-- Select **SR_ff_test** Icon to elaborate the testbench ,once the elaboration is done It creates snapshot and stores inside snapshots directory.

-- Select the snapshot **ff_lib.SR_ff_test: module** and click the following Icon to Invoke the waveform window (Simvision) for performing the simulation.

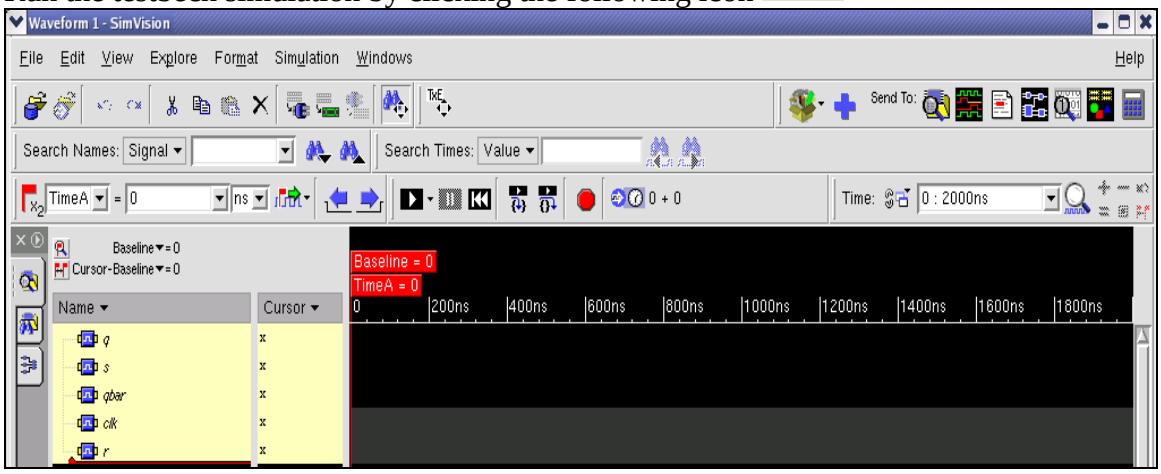


Once the Design browser window (i.e Simvision) comesup select **SR_ff_test** also the required inputs and outputs ports and click the  button to open the waveform window.

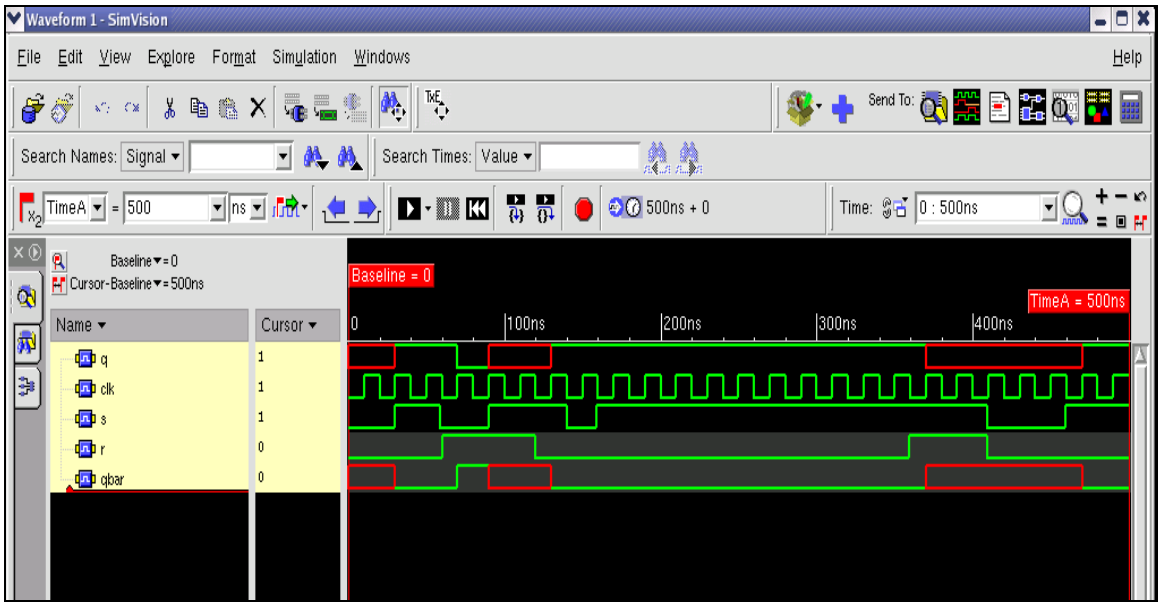


Waveform window appears as shown below:

Run the testbech simulation by clicking the following icon 



We will obtain the required output of SR – Flip flop in the waveform window as shown below:



The equivalent command terminal output can be observed in the Simvision console window and also in the nclaunch console terminal.The o/p is as shown below:

```
nclaunch> ncvlog -cdslib /servers/tfo_spb/bsatish/Vtu/Cadence_digital_labs/Workarea/Flipflops/cds.lib -l
ncvlog: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
ncvlog: Memory Usage - 7.4M program + 4.9M data = 12.3M total
ncvlog: CPU Usage - 0.0s system + 0.0s user = 0.0s total (0.0s, 65.1% cpu)
nclaunch> ncelab -cdslib /servers/tfo_spb/bsatish/Vtu/Cadence_digital_labs/Workarea/Flipflops/cds.lib -l
ncelab: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
        Elaborating the design hierarchy:
                Caching library 'ff_lib' ..... Done
        Building instance overlay tables: ..... Done
        Generating native compiled code:
                ff_lib.SR_ff:module <0x4ebc7132>
                        streams:   7, words:  2029
                ff_lib.SR_ff_test:module <0x2a6287f1>
                        streams:  11, words:  7555
        Loading native compiled code:     ..... Done
        Building instance specific data structures.
        Design hierarchy summary:
                            Instances  Unique
                Modules:            2       2
                Registers:          4       4
                Scalar wires:       8       -
                Always blocks:      3       3
                Initial blocks:     3       3
                Cont. assignments:  5       5
                Pseudo assignments: 3       3
                Timing checks:      4       3
        Writing initial simulation snapshot: ff_lib.SR_ff_test:module
ncelab: Memory Usage - 15.8M program + 9.1M data = 24.9M total
ncelab: CPU Usage - 0.0s system + 0.0s user = 0.1s total (0.1s, 70.0% cpu)
nclaunch> ncsim -gui -cdslib /servers/tfo_spb/bsatish/Vtu/Cadence_digital_labs/Workarea/Flipflops/cds.li
nclaunch> ncsim: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
txe: 06.10-p001: (c) Copyright 1995-2006 Cadence Design Systems, Inc.
simvision: 06.10-p001: (c) Copyright 1995-2007 Cadence Design Systems, Inc.
ncsim: Memory Usage - 14.2M program + 12.4M data = 26.6M total
ncsim: CPU Usage - 0.2s system + 0.1s user = 0.3s total (154.9s, 0.2% cpu)
```

Lab6: NCO(10 Bit number controlled oscillator)

There are a number of things to consider before beginning the lab exercises. Please read through this section completely, and perform any needed steps in order to ensure a successful workshop.

The NCO directory contains rclabs folder. Inside rclabs folder you will see many other directories but for IUS, change the directory to Simulation and for Synthesis and P&R select work directory

Lab directory details:

Simulation	Contains the lab experiments including Testbences for simulating the codes.
work	It's a place to run Synthesis and P&R for NCO.

In this lab, you will simulate the design using the Incisive simulator. You will Perform this lab in the *Simulation* directory. This directory contains the following files (which you should briefly examine) describing the NCO and its testbenches:

File(s) Description:

mem.v mux_2to1.v phase_inc.v testbench.v top.v

Compile , Elaborate and Simulate using Irun Utility:

Run the below command

irun mem.v mux_2to1.v phase_inc.v testbench.v top.v –access +rwc –mess –gui

1. Now the Console & Design Browser window opens

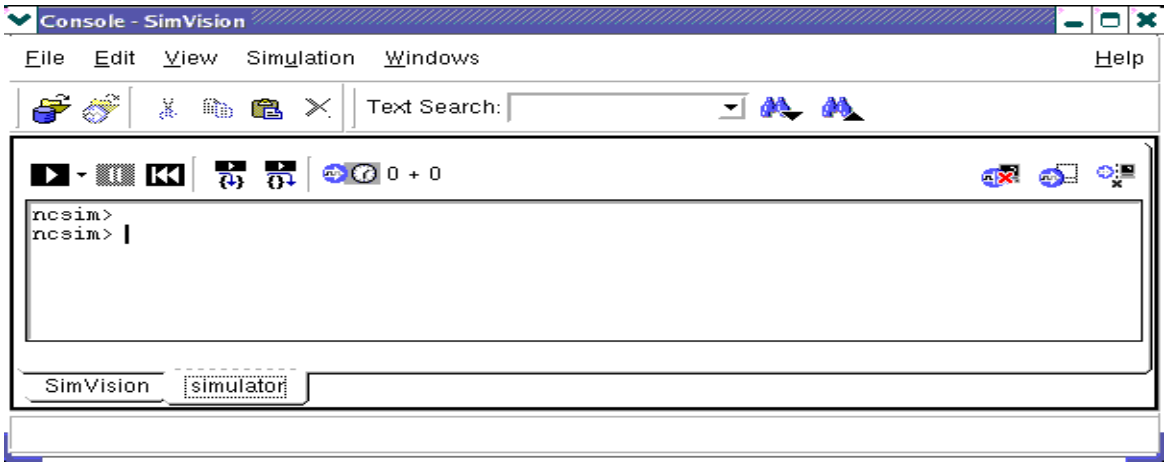
2. Before proceeding to the next step analyze the messages in the terminal window

The -gui option opens the **Console** and **Design Browser** windows.

Tour the Graphical Interface

1. Examine the Console window.

a. You can use the Menu Bar to run or step the simulation, set scopes and stops, show the value of objects, and start other graphical tools.
b. You can use the Tool Bar to run, interrupt, reset, step, or next the simulation, and shut down the interface or the simulation, or disconnect the simulation.
c. You can use the command line interface to the simulation in the I/O Region.



2. Examine the Design Browser window.

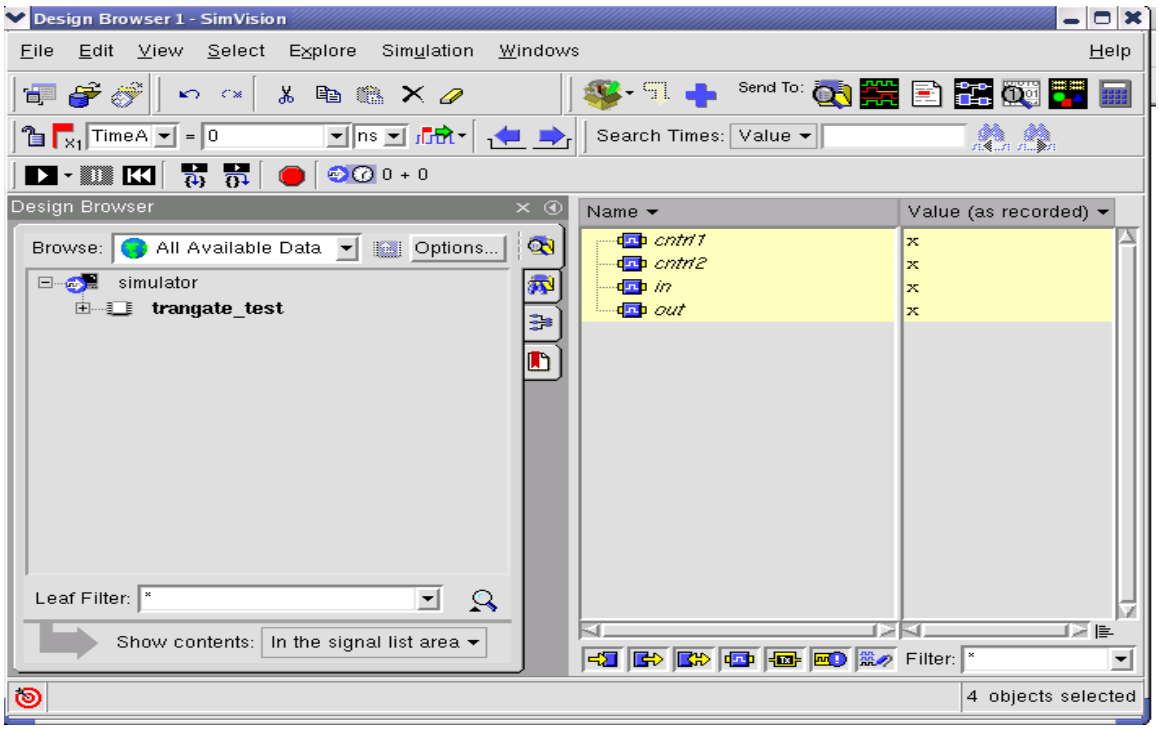
Open an existing Design Browser window or select the Windows— new — Design Browser menu item or the Design Browser button.

a. Display the objects of a scope and their value in the Objects List pane (Select any displayed scope in the Scope Tree pane).
b. Display the component instances of the scope (double-click the scope in the Scope Tree pane).

3. Tour the **Waveform** window.

Open an existing Waveform window or select **the Windows - New - Waveform** menu item or the **Waveform**  button.The simulator creates a default SHM database and sets a probe onany selected signals and opens a Waveform window displaying the selected signals.

a. In the Design Browser window select all signals at the *testbench* scope.
b. Add the selected signal(s) to the Waveform window (select the **Waveform** button or the **Add Selected** button or drag and drop the signals into the Waveform window).**Note:** To add additional signals simply select them in any window and click the ***Waveform*** button again.



Examine the Design and Testbench Hierarchy

In this section of the lab you visit the Source Browser, Schematic Tracer and Waveform window.

1. In the Design Browser window select the top-level (mem_test) scope and select the Source Browser button to send it to the target Source Browser window. As no such window yet exists, this opens a Source Browser window displaying the source of the Top-level unit, and makes it the default Source Browser target window.

2. In the Source Browser window ensure that just the top-level scope is selected (navigate up as needed and Select—This Scope) and send it to the target Schematic Tracer window. As no such window yet exists, this opens a Schematic Tracer window displaying the top-level unit, and makes it the default Schematic Tracer target window, in which you:

a. Ensure that the top-level scope is still selected, and select the fill Module button, to display the testbench content.
b. Select the Edit - Select - All menus item, and again select the fill Module button, to Expand the second level content.
c. Select the Zoom Full button to fit all displayed elements.

3. In the Source Browser window ensure that just the top-level scope is selected and send it  tothe target Waveform window. As no such window yet exists, this opens a Waveform window displaying the signals of the top-level unit, and makes it the default Waveform target window.

a. In the left sidebar, select the Design Browser  tab toexpand the sidebar area and display the embedded Design Browser.

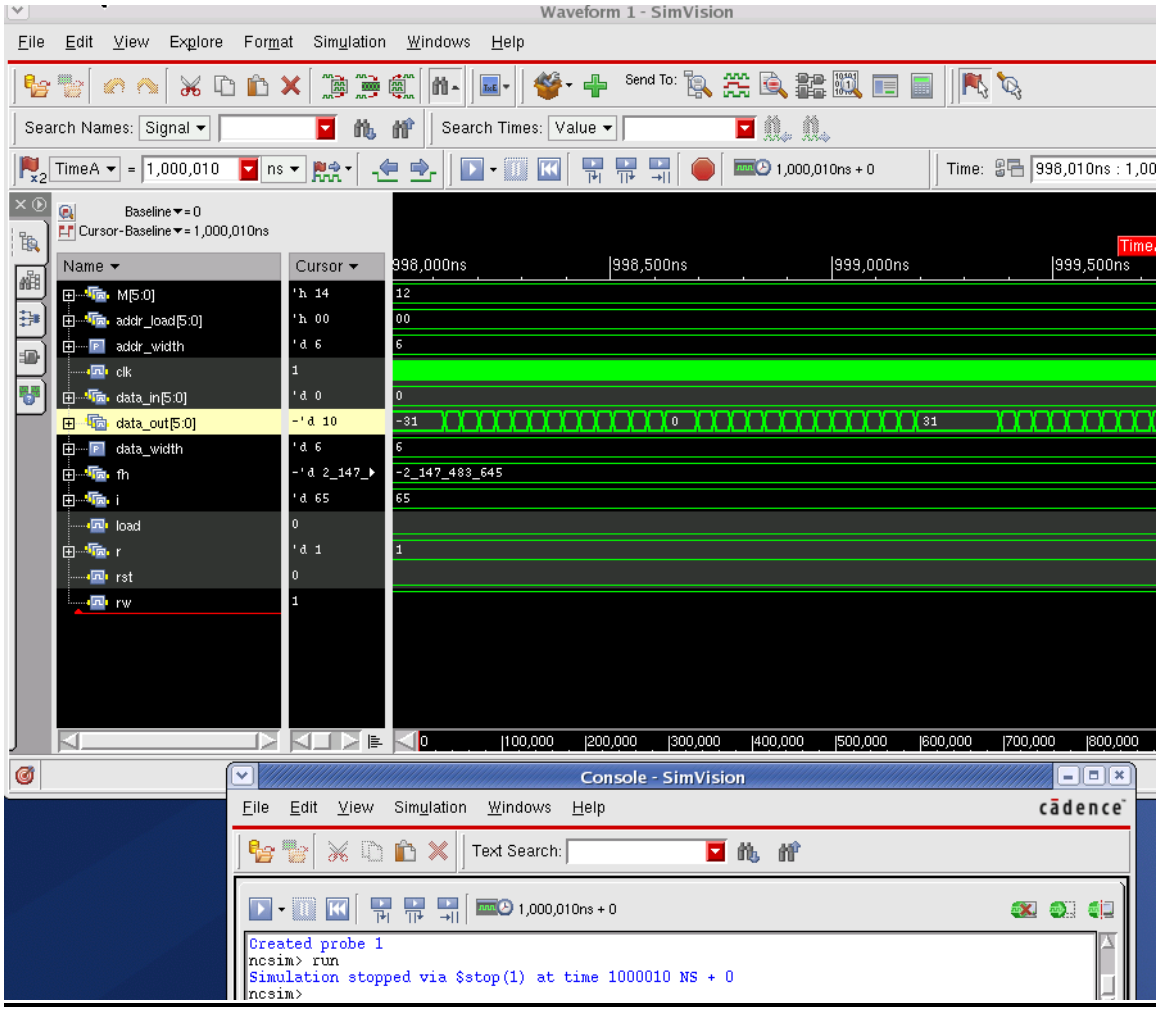
--- Run the simulation until the next breakpoint, or for the Duration entered in the time field (i.e 40ns).

--- Current Time range.

 --- Move primary cursor to previous edge of select signal.

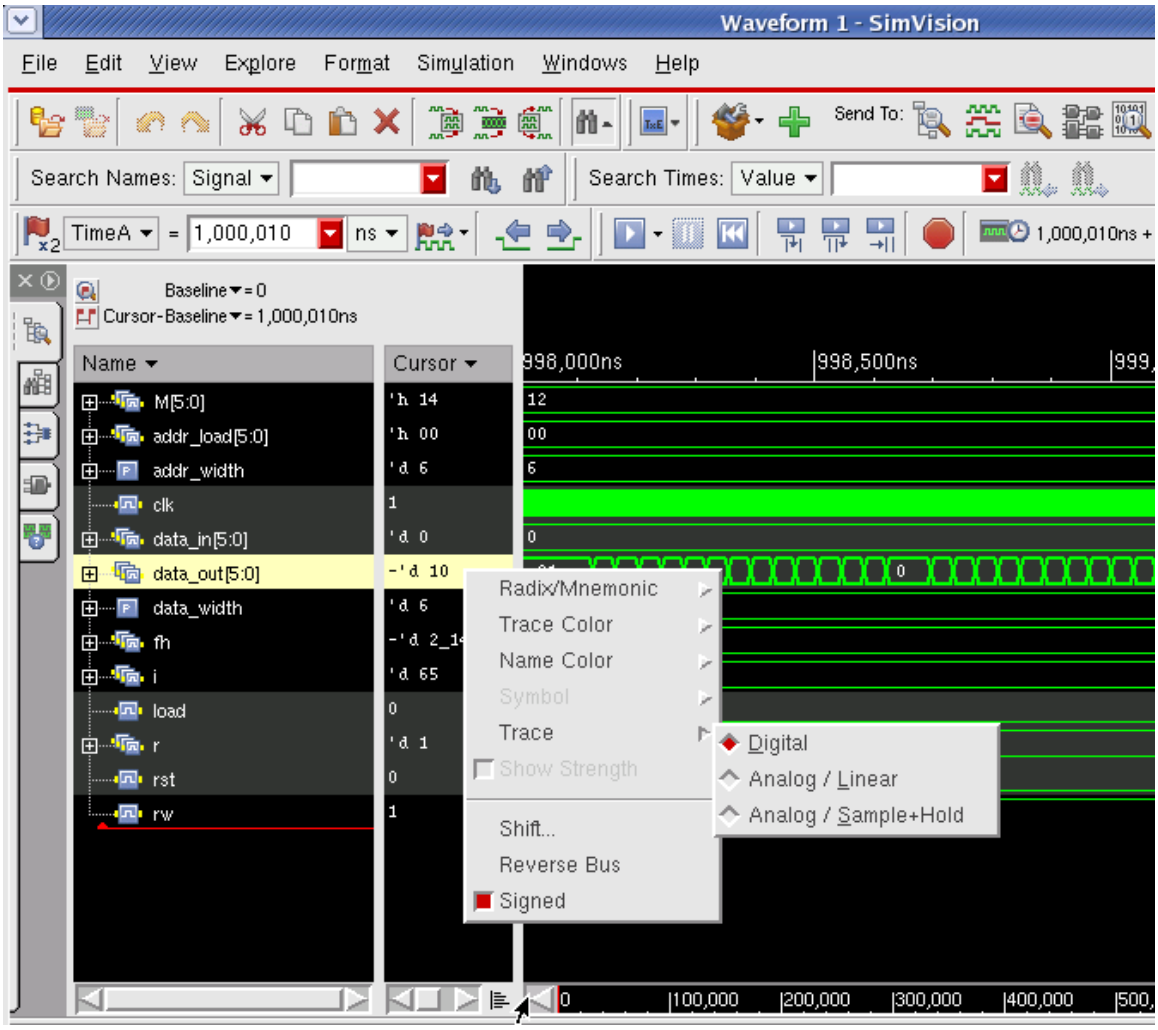
 --- Reset the simulation back to time " 0 ".

Once the simulation is done you can see the following waveform window and console window with the outputs.

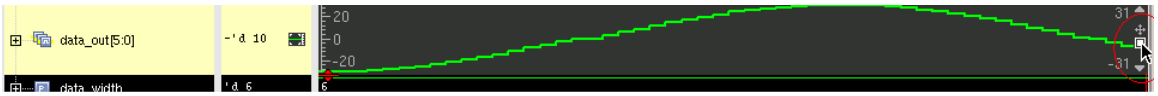


Further follow the following steps:

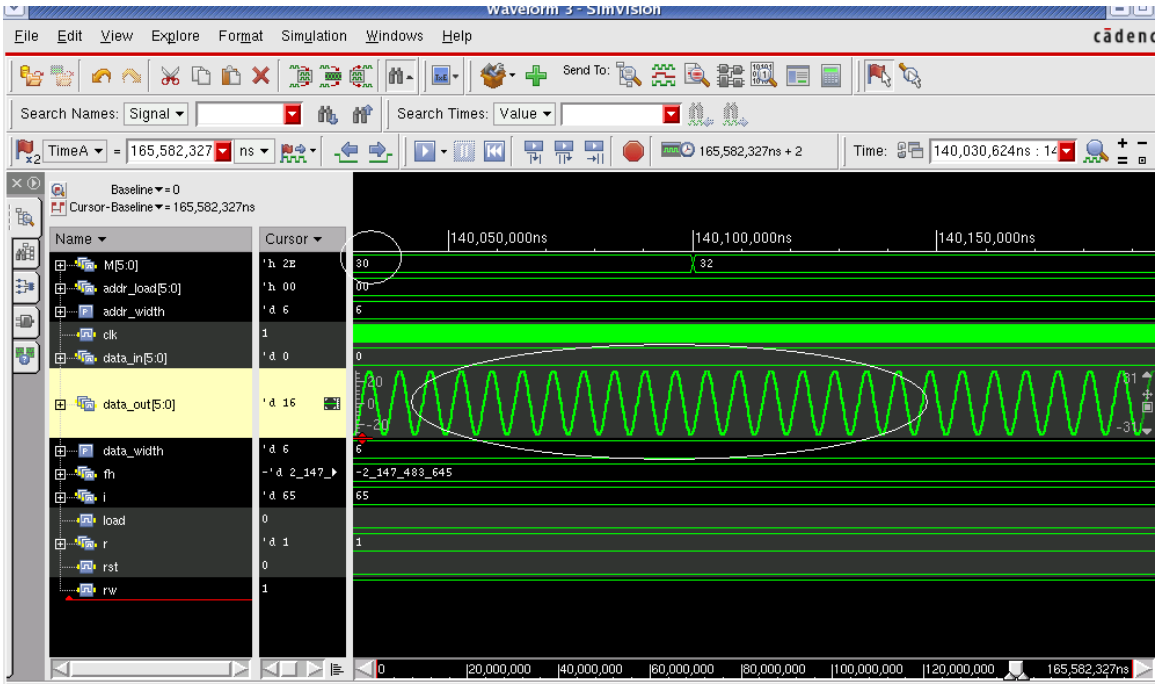
1. Highlight the output pin(data_out[5:0]) and right click on it
2. Click on Trace and select Analog/Sample+Hold



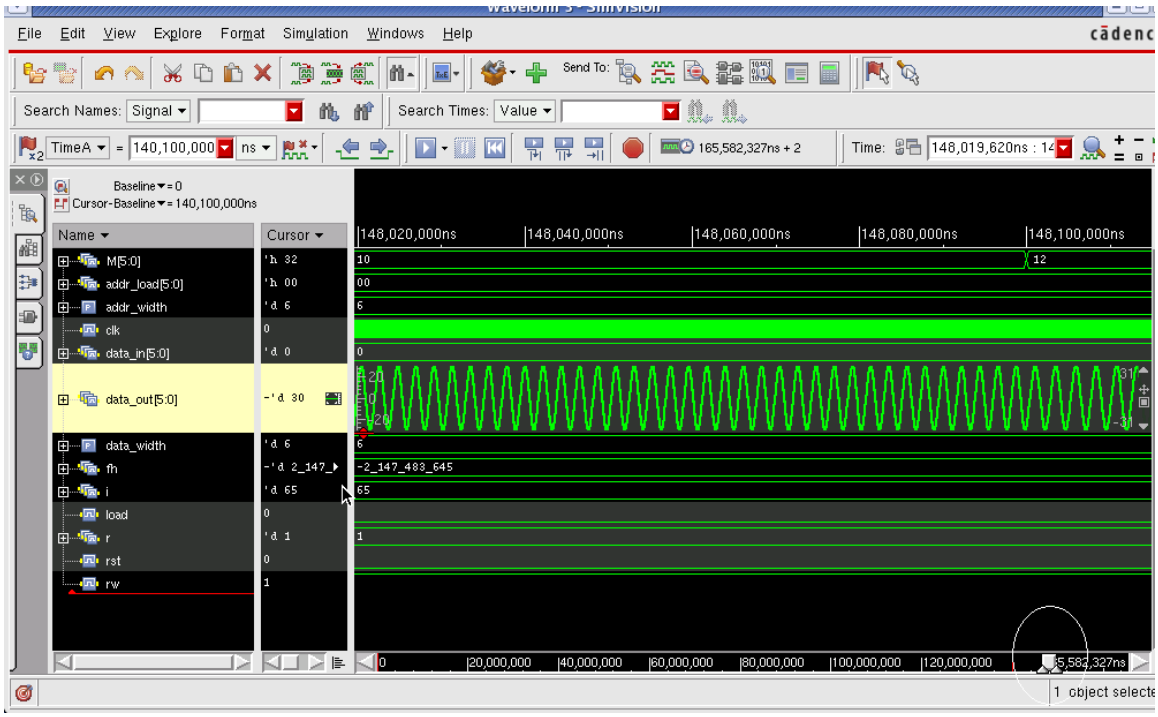
3. Now select the symbol which is highlighted by red circle



5 . Now we can see based on the numerical value (hexadecimal) corresponding output wave form can be seen.



6. Drag and observe the waveform for different numerical values



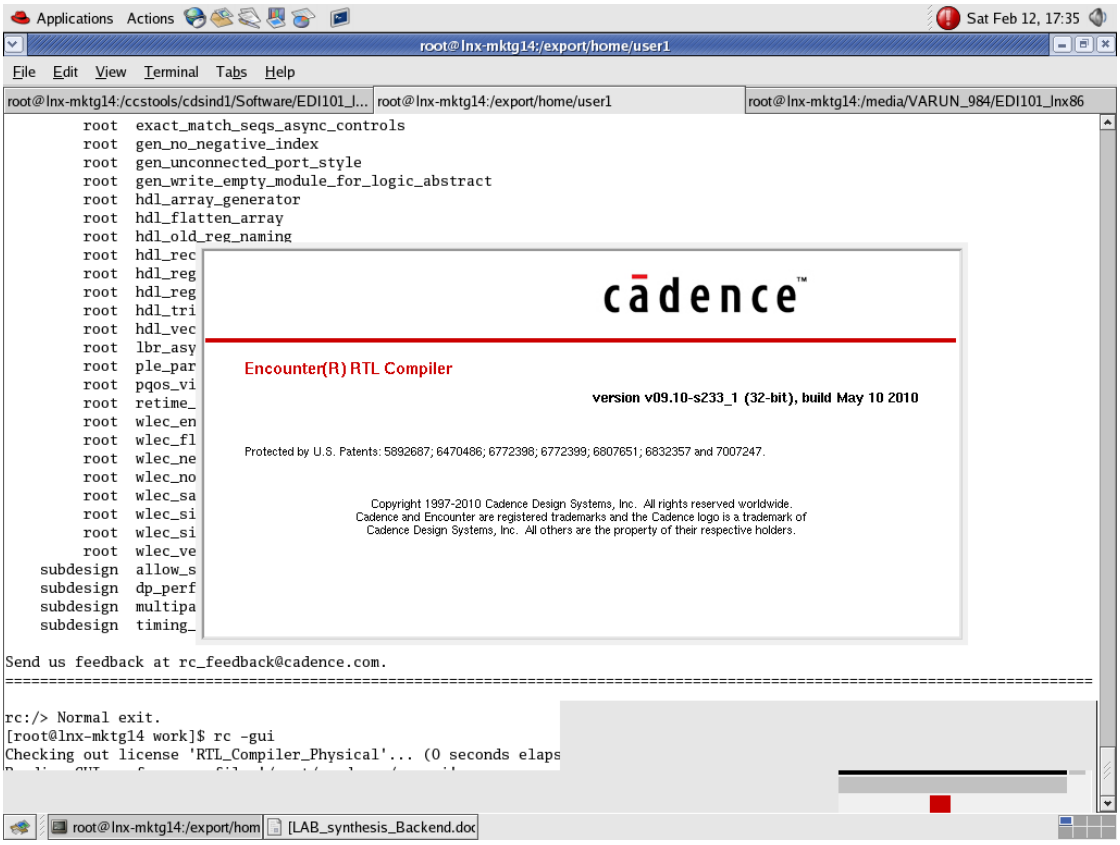
Lab7: Automatic layout generation followed by post layout extraction and simulation of the circuit studied in Lab 6

In this lab we will do the Synthsesis and Physical Design of NCO Design for which simulation is done in Lab6. Synthesis will be done using RTL Compiler and Physical Design will be done using Encounter Digital Implementation System.

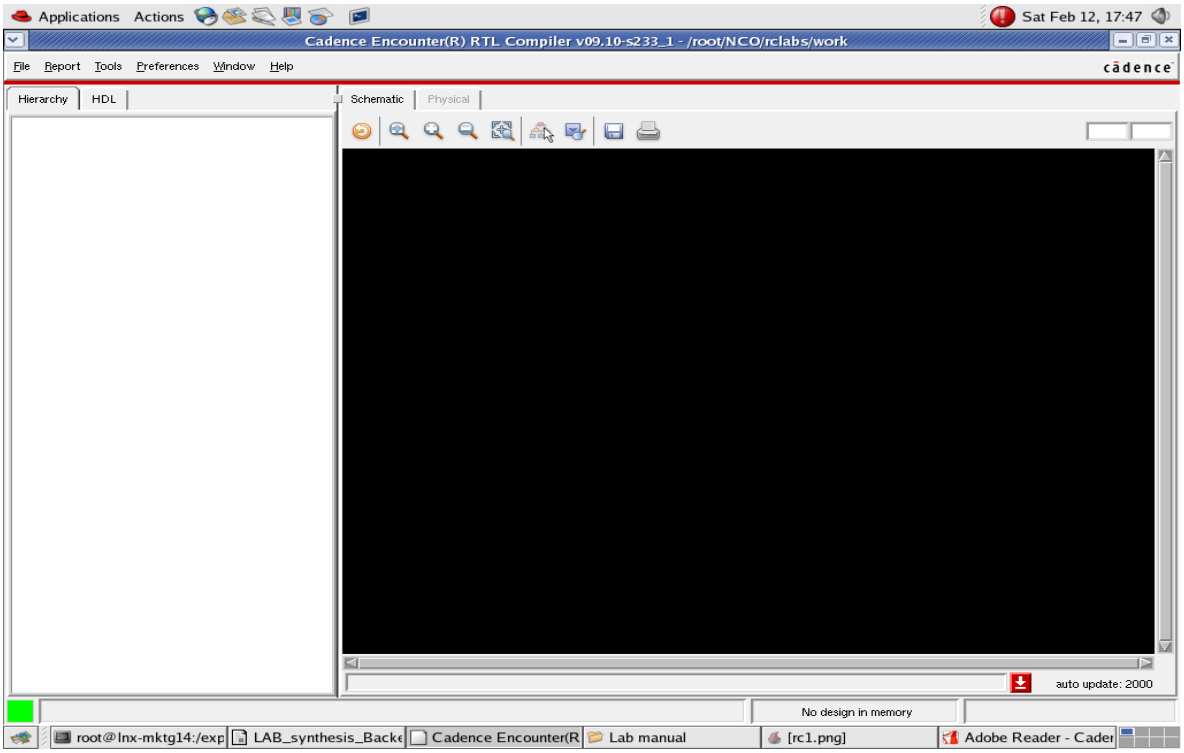
Go to directory /NCO/rclabs/work.

Lets do the Synthesis first.

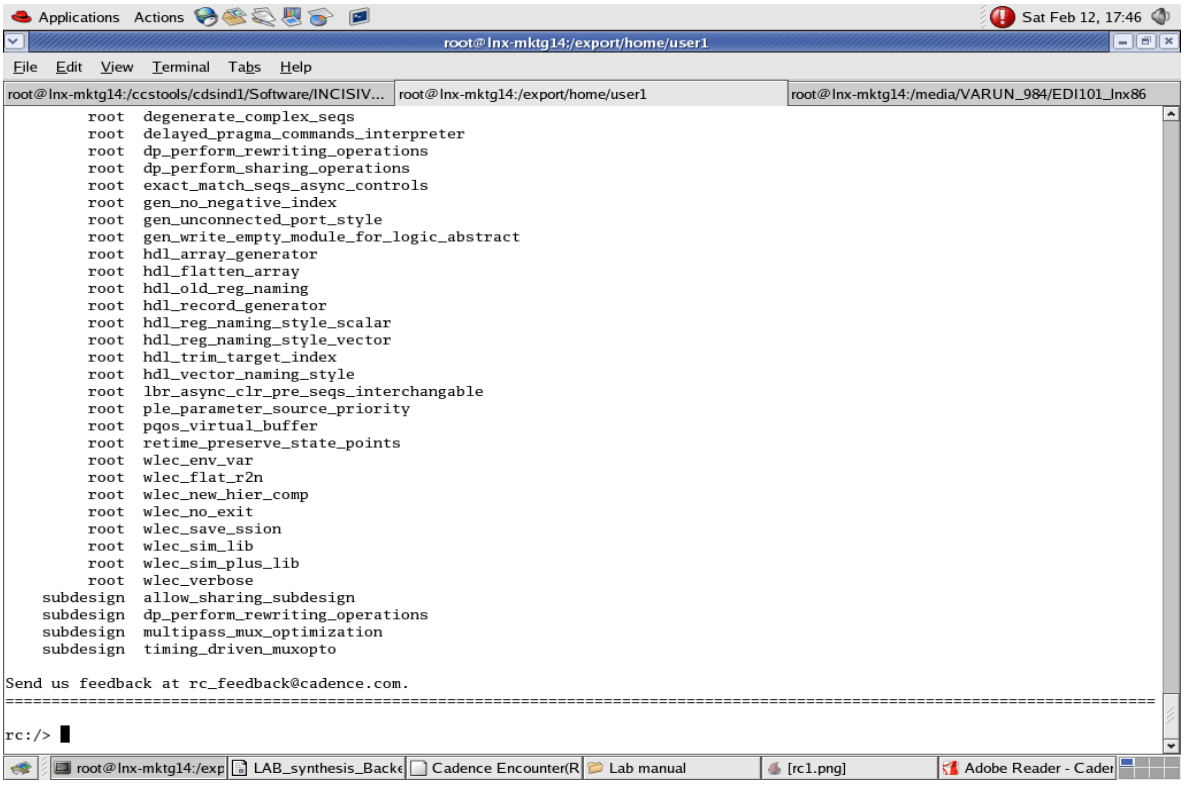
1. Invoke RTL Compiler by typing “rc -gui” on your terminal window.
   The below picture can be seen after typing the above command.



The tool window look like the below image:



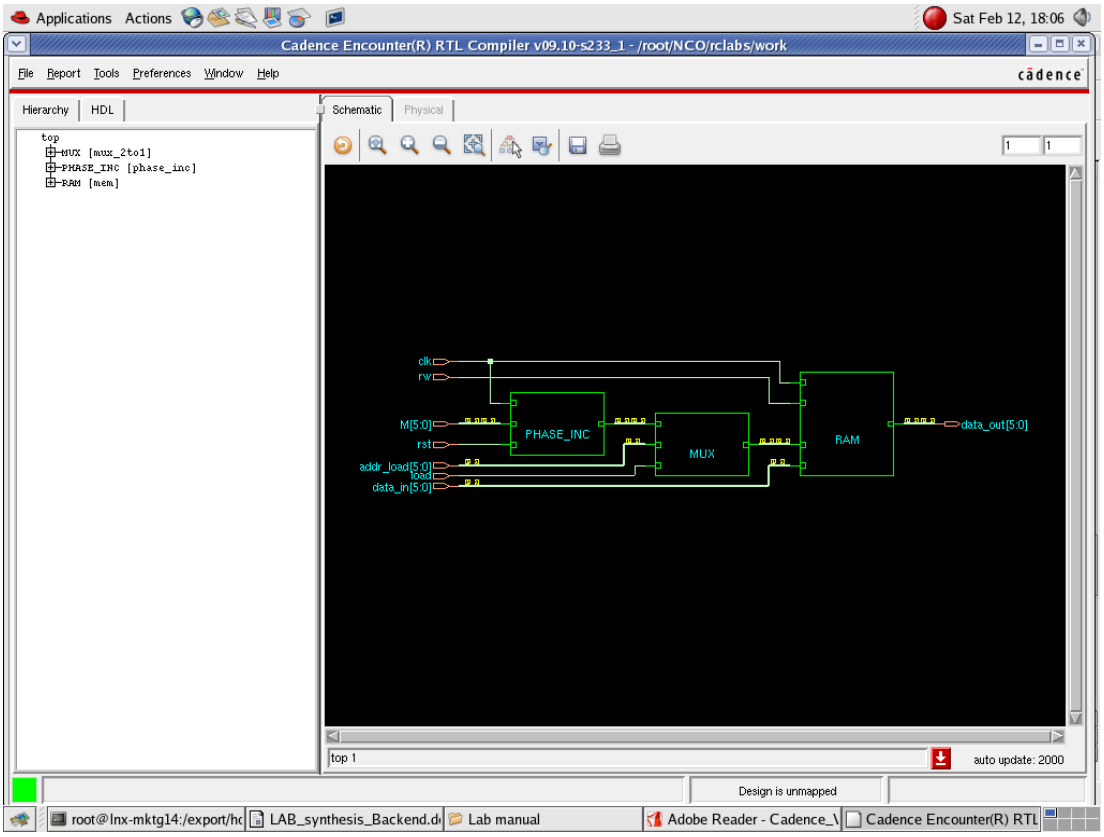
The terminal will look like the below image after the tool is invoked.



2. Give the path of the library w.r.t to the directory you are in using the command:
   "set_attribute lib_search_path ../library"
3. Give the path of the RTL files with respect to the directory you are in using the below command:
   "set_attribute hdl_search_path ../rtl"
4. Read the library from the directory specified in giving the path for the library files in step 2 using the command:
   "set_attribute library slow_normal.lib"
   "slow_normal.lib" is the name of the library file in the directory "library". There is another library there in that directory with name "slow_highvt.lib". Any one of these two libraries could be used at a time.
5. Read the RTL files from the directory specified in the path in step 3. The RTL files are in the directory name "rtl":
   "read_hdl {mem.v mux_2to1.v phase_inc.v top.v}.
6. Now Elaborate the design using "elaborate"command.
7. Give the command to see the circuit in Tool window:

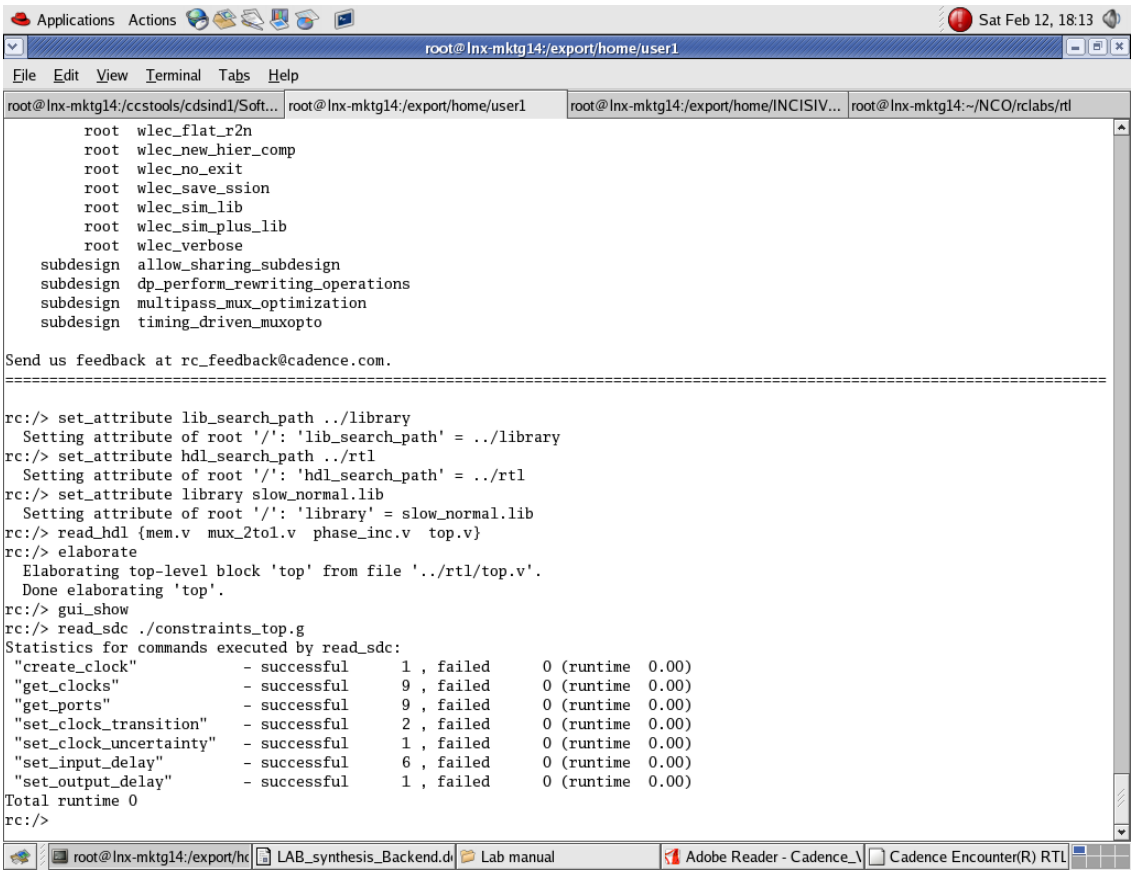
The terminal window after the step 7 will look like

The Tool window looks like image on next page

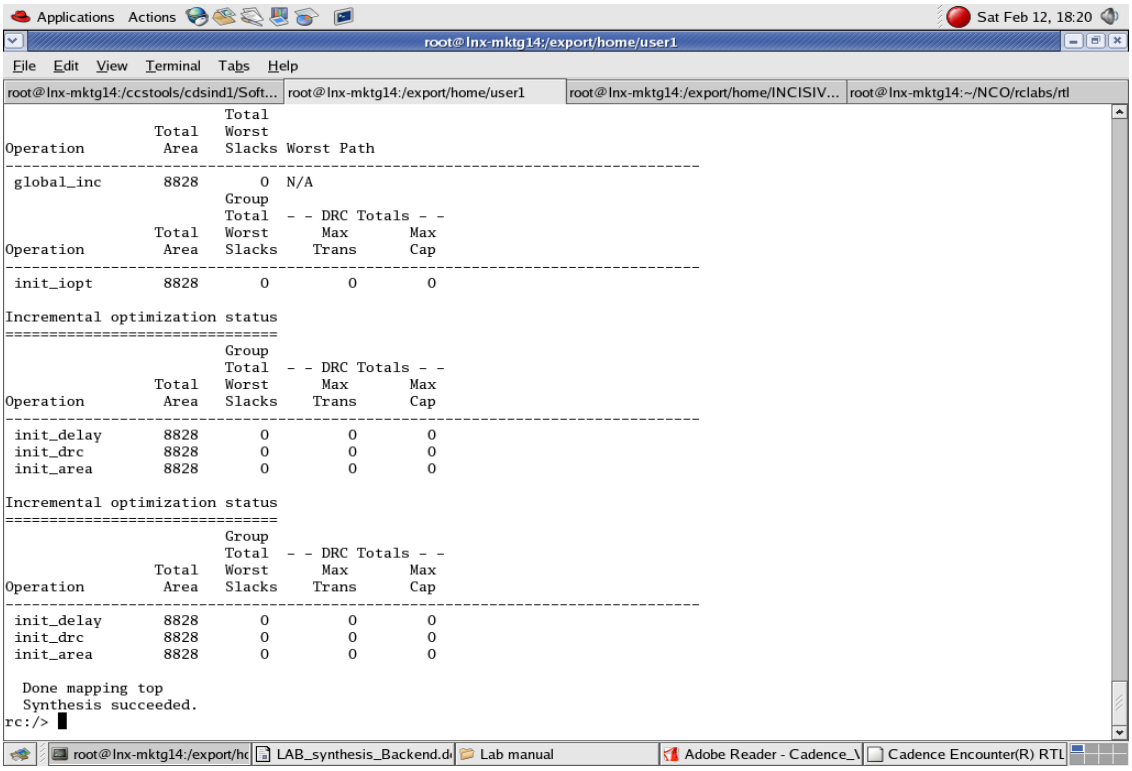


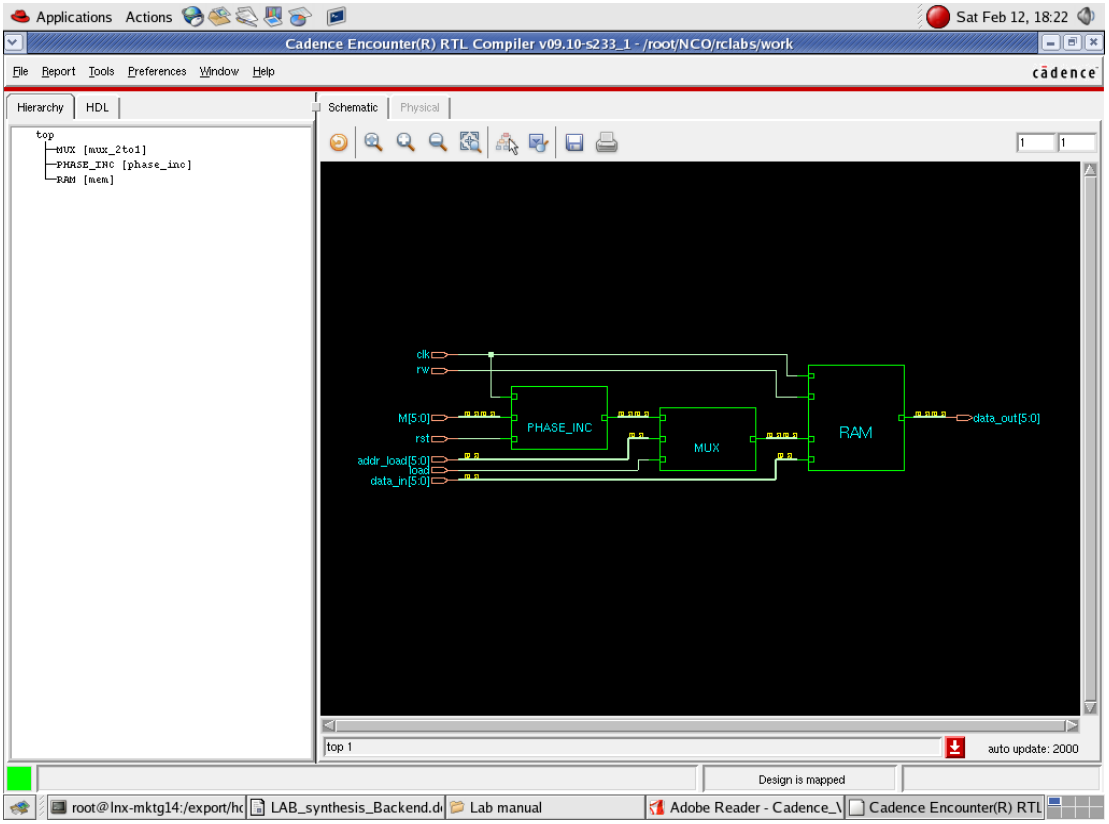
8. Give the standard delay constraints using:
   "read_sdc ./constraints_top.g".

The terminal window looks like the image on next page.



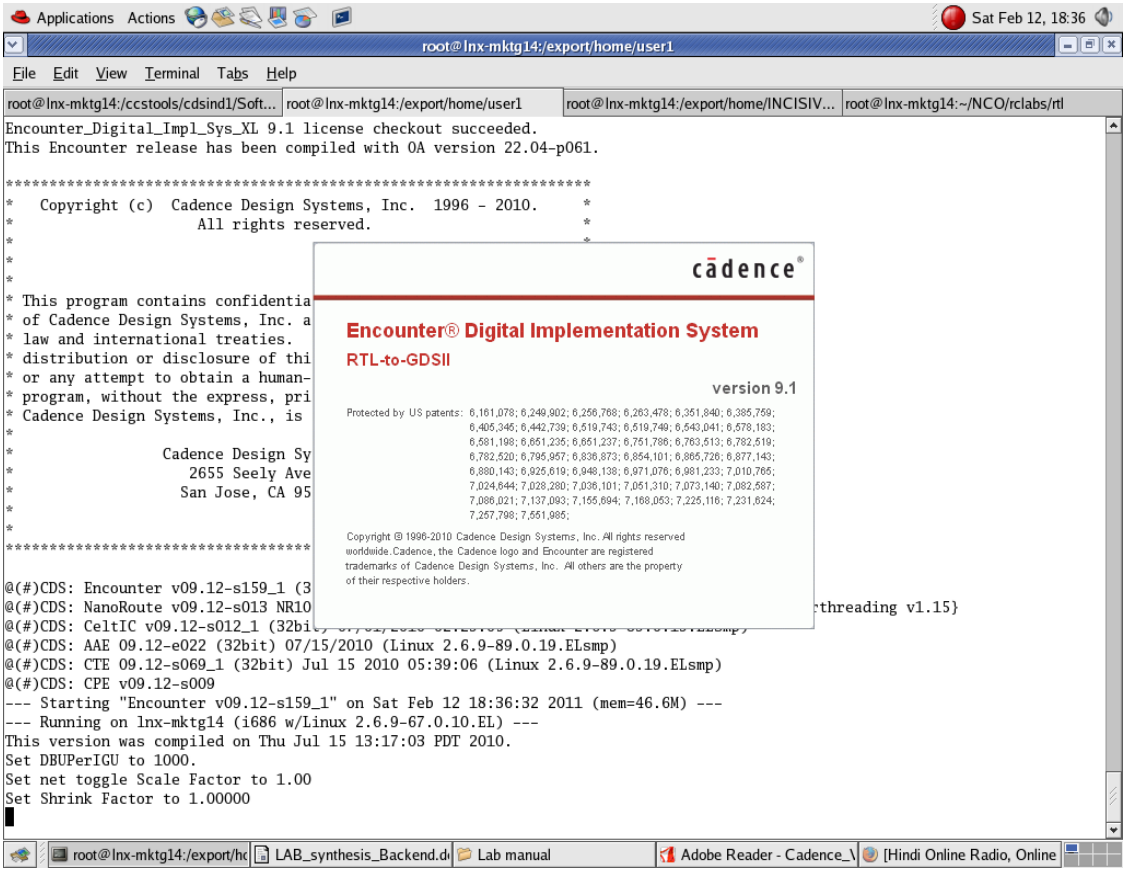
9. Synthesize the circuit using the command:
   "synthesize -to_mapped -effort medium".
   The terminal window and the synthesized circuit in tool window will appear to be as on next page:


Applications Actions
Sat Feb 12, 18:20
root@lnx-mktg14:/export/home/user1
File Edit View Terminal Tabs Help
root@lnx-mktg14:/ccstools/cdsind1/Soft...
root@lnx-mktg14:/export/home/user1
root@lnx-mktg14:/export/home/INCISIV...
root@lnx-mktg14:~/NCO/rclabs/rtl
Total
Total Worst
Operation Area Slacks Worst Path
global_inc 8828 0 N/A
Group
Total - - DRC Totals - -
Total Worst Max Max
Operation Area Slacks Trans Cap
init_iopt 8828 0 0 0
Incremental optimization status
Group
Total - - DRC Totals - -
Total Worst Max Max
Operation Area Slacks Trans Cap
init_delay 8828 0 0 0
init_drc 8828 0 0 0
init_area 8828 0 0 0
Incremental optimization status
Group
Total - - DRC Totals - -
Total Worst Max Max
Operation Area Slacks Trans Cap
init_delay 8828 0 0 0
init_drc 8828 0 0 0
init_area 8828 0 0 0
Done mapping top
Synthesis succeeded.
rc:/>
root@lnx-mktg14:/export/hc
LAB_synthesis_Backend.d
Lab manual
Adobe Reader - Cadence_\
Cadence Encounter(R) RTL

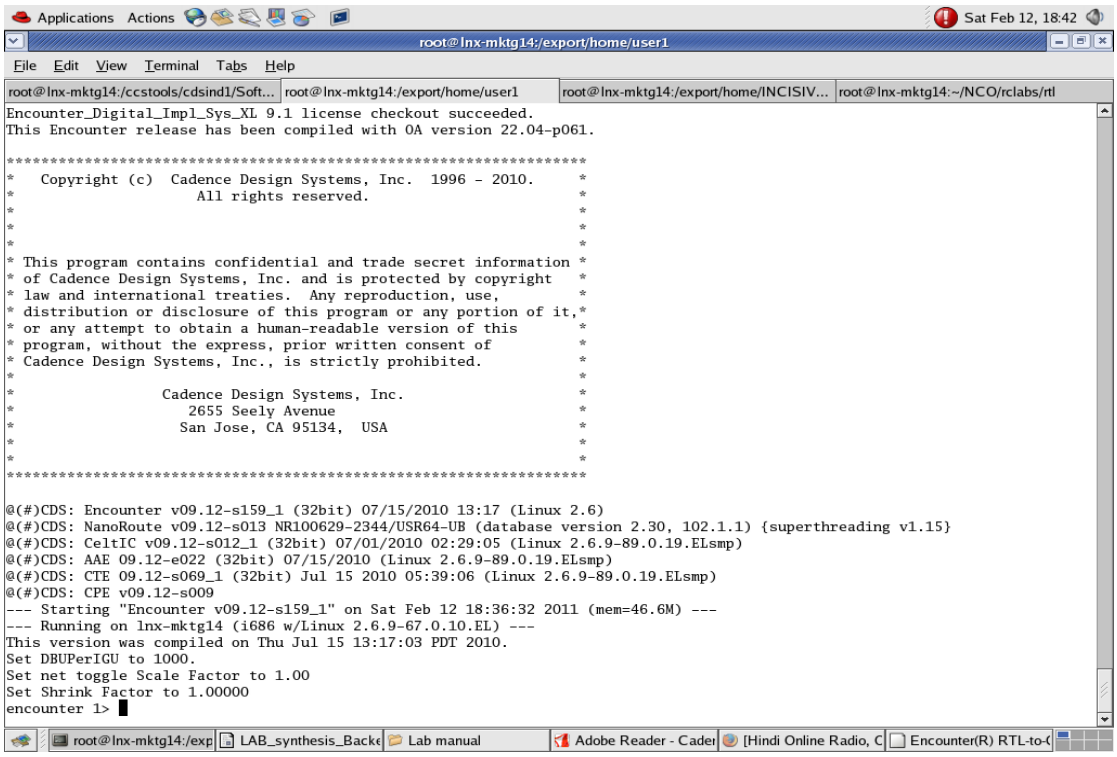


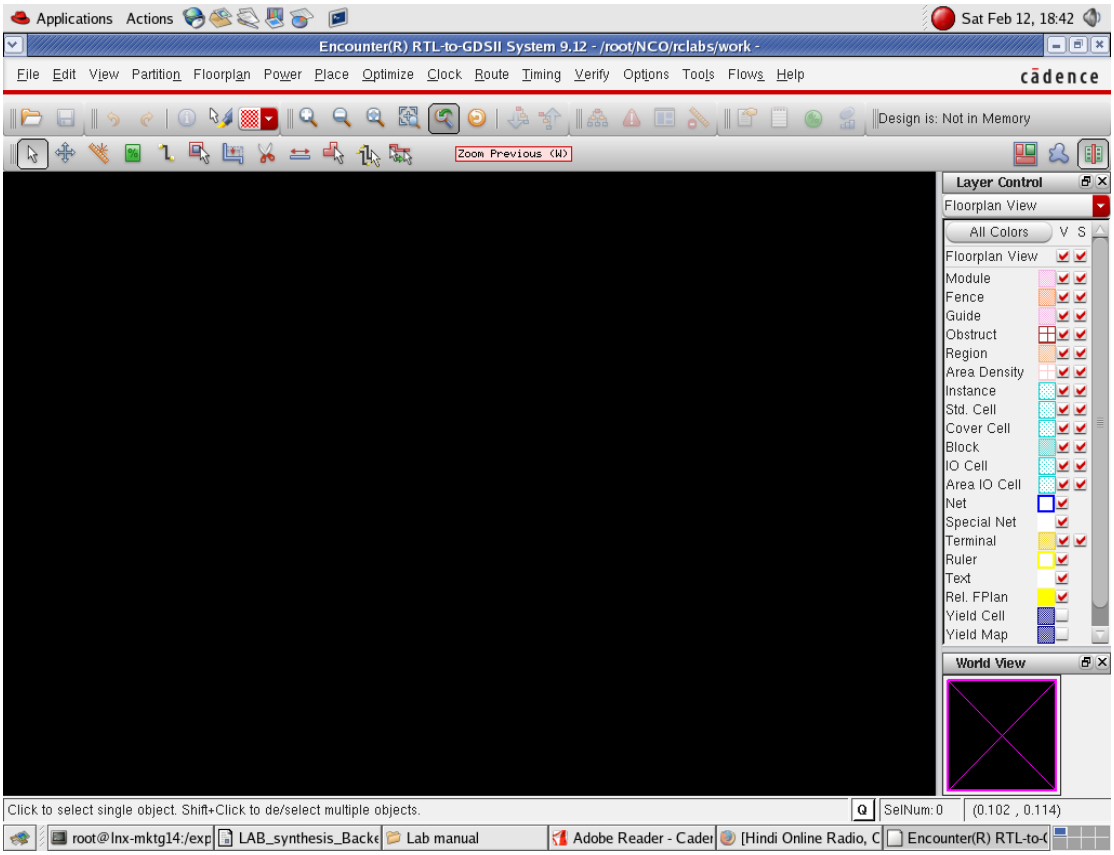
10. Write the hdl code in terms of library components for the synthesized circuit using the command:
    "write_hdl > nco.hdl"
    "nco.hdl" is the name of file in which the code gets write.
11. Similarly write the constraint file using
    "write_sdc > nco.sdc".
12. Timing could be check using "report timing".
13. Similarly for Gates "report gates".
14. Check area using "report area".
15. Check Power dissipation using "report power".

    After the Synthesis Physical Design can be done by invoking the tool "Encounter Digital implementation".
16. Go to Directory /NCO/rclabs/work.
17. Invoke the tool using "encounter" or "velocity".
    The tool starts as below image:

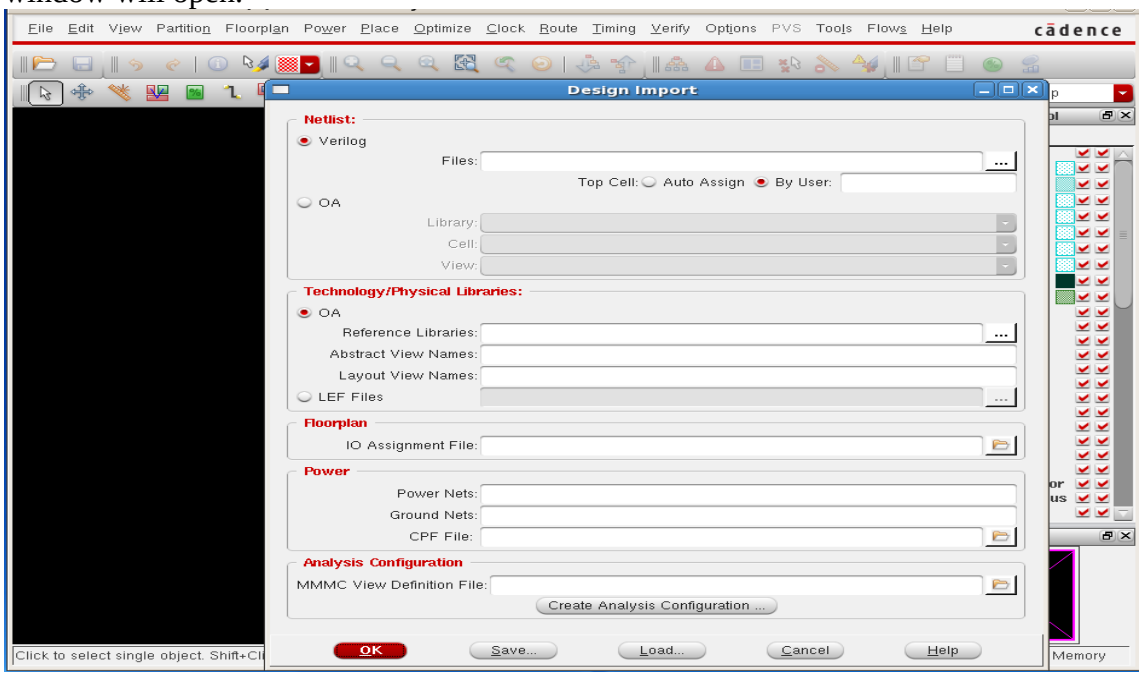


The terminal window and tool window can be seens as similar to images on next page

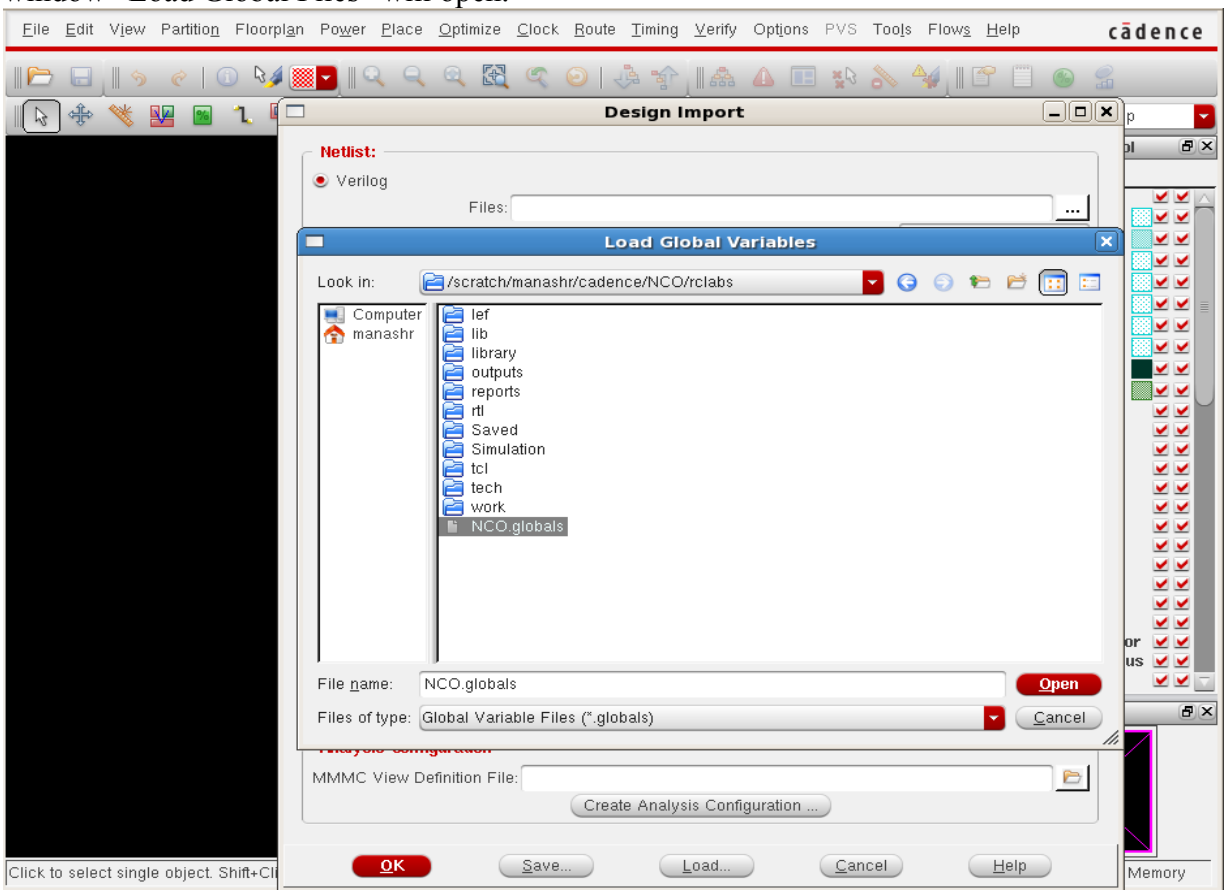

Applications Actions
Sat Feb 12, 18:42
root@lnx-mktg14:/export/home/user1
File Edit View Terminal Tabs Help
root@lnx-mktg14:/ccstools/cdsind1/Soft...
root@lnx-mktg14:/export/home/user1
root@lnx-mktg14:/export/home/INCISIV...
root@lnx-mktg14:~/NCO/rclabs/rtl
Encounter_Digital_Impl_Sys_XL 9.1 license checkout succeeded.
This Encounter release has been compiled with OA version 22.04-p061.
* Copyright (c) Cadence Design Systems, Inc. 1996 - 2010. *
* All rights reserved. *
* This program contains confidential and trade secret information *
* of Cadence Design Systems, Inc. and is protected by copyright *
* law and international treaties. Any reproduction, use, *
* distribution or disclosure of this program or any portion of it, *
* or 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. *
* Cadence Design Systems, Inc. *
* 2655 Seely Avenue *
* San Jose, CA 95134, USA *
@(#)CDS: Encounter v09.12-s159_1 (32bit) 07/15/2010 13:17 (Linux 2.6)
@(#)CDS: NanoRoute v09.12-s013 NR100629-2344/USR64-UB (database version 2.30, 102.1.1) {superthreading v1.15}
@(#)CDS: CeltIC v09.12-s012_1 (32bit) 07/01/2010 02:29:05 (Linux 2.6.9-89.0.19.ELsmp)
@(#)CDS: AAE 09.12-e022 (32bit) 07/15/2010 (Linux 2.6.9-89.0.19.ELsmp)
@(#)CDS: CTE 09.12-s069_1 (32bit) Jul 15 2010 05:39:06 (Linux 2.6.9-89.0.19.ELsmp)
@(#)CDS: CPE v09.12-s009
--- Starting "Encounter v09.12-s159_1" on Sat Feb 12 18:36:32 2011 (mem=46.6M) ---
--- Running on lnx-mktg14 (i686 w/Linux 2.6.9-67.0.10.EL) ---
This version was compiled on Thu Jul 15 13:17:03 PDT 2010.
Set DBUPerIGU to 1000.
Set net toggle Scale Factor to 1.00
Set Shrink Factor to 1.00000
encounter 1>
root@lnx-mktg14:/exp
LAB_synthesis_Backe
Lab manual
Adobe Reader - Cade
[Hindi Online Radio, C
Encounter(R) RTL-to-(



18. Go to the Tool window and click on the File and select Import Design. A new window will open.



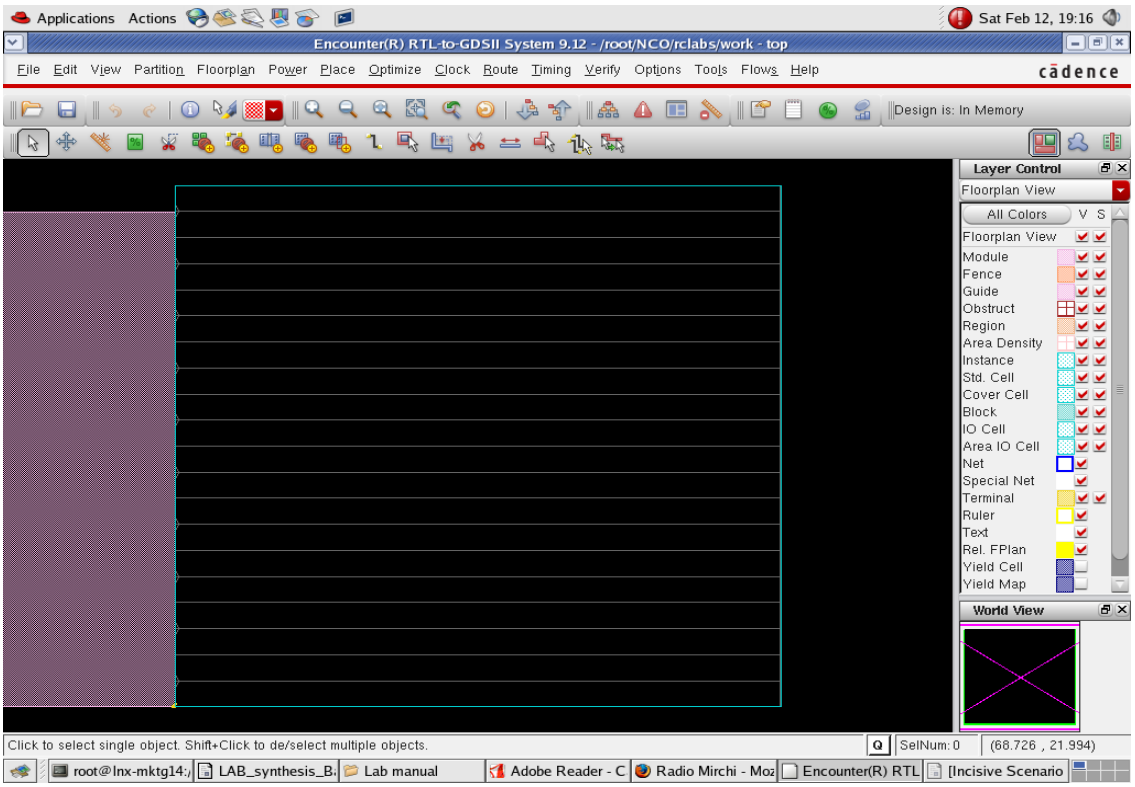
19. Select the verilog files, power nets, lef files and view files using load button. A new window “Load Global Files” will open.



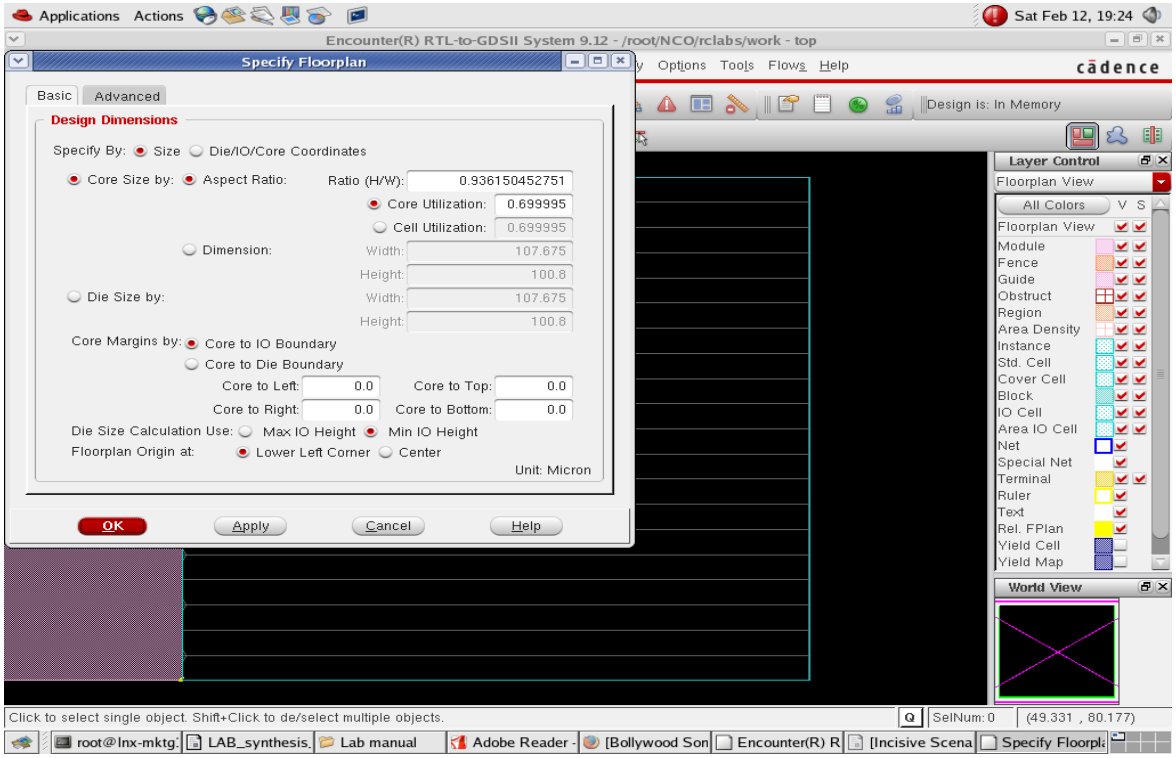
20. Click on the back button and select the “NCO_globals” file and click the Open button.

21. Click on Auto assign after top cell.

22. Select OK. The tool window will look like image on next page.

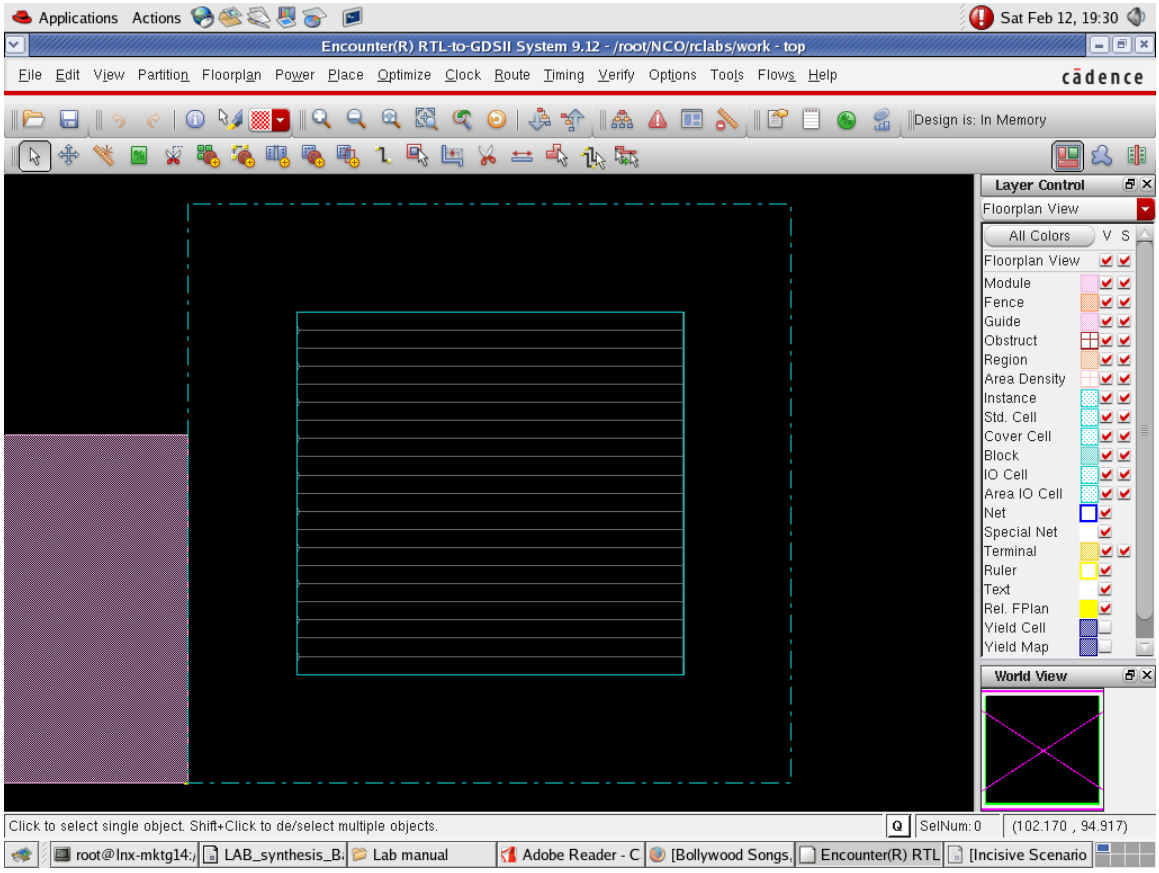


The pink colour blocks are the standard cells. This is floorplan view of the design.



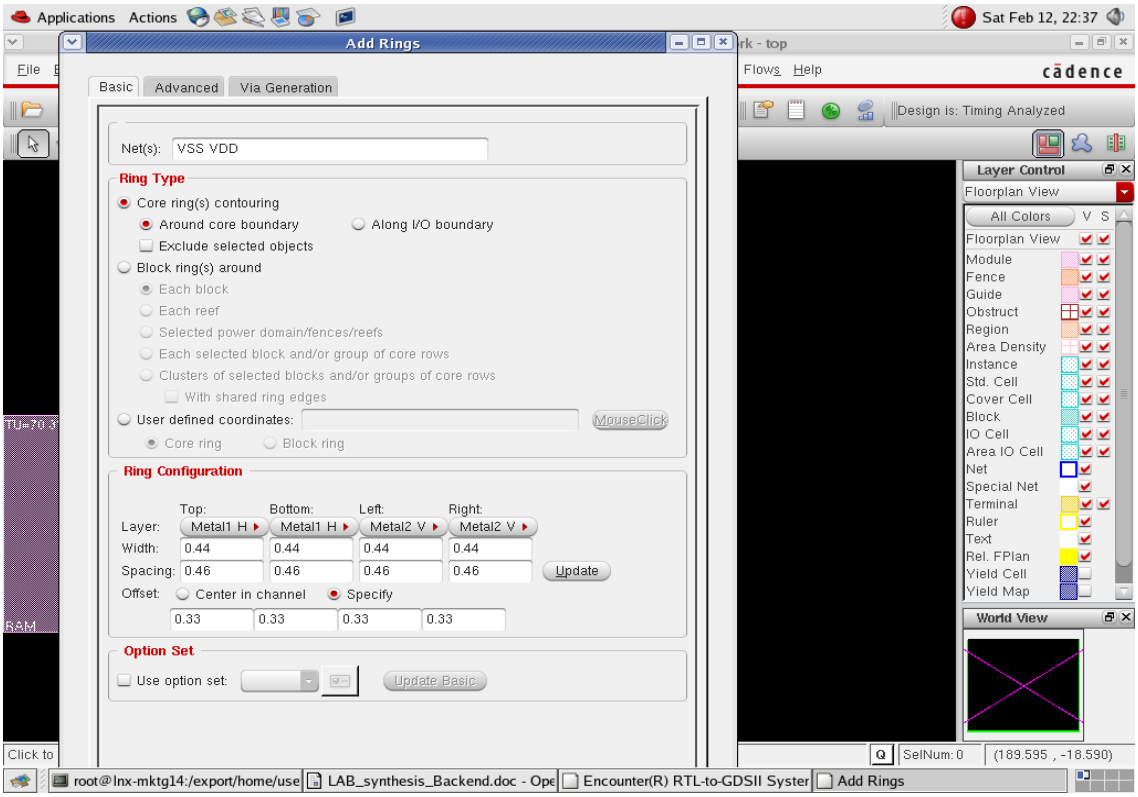
23.Click on Floorplan and select “Specify Floorplan”.

Select the Aspect Ratio as per the requirement. Give some dimension in “Core to left”, “Core to right”, “Core to top”,“Core to bottom”. e.g. give 30 to each. This is to create the space for Power rings which will be created in power planning. Click OK and the Tool window will be look like as below.



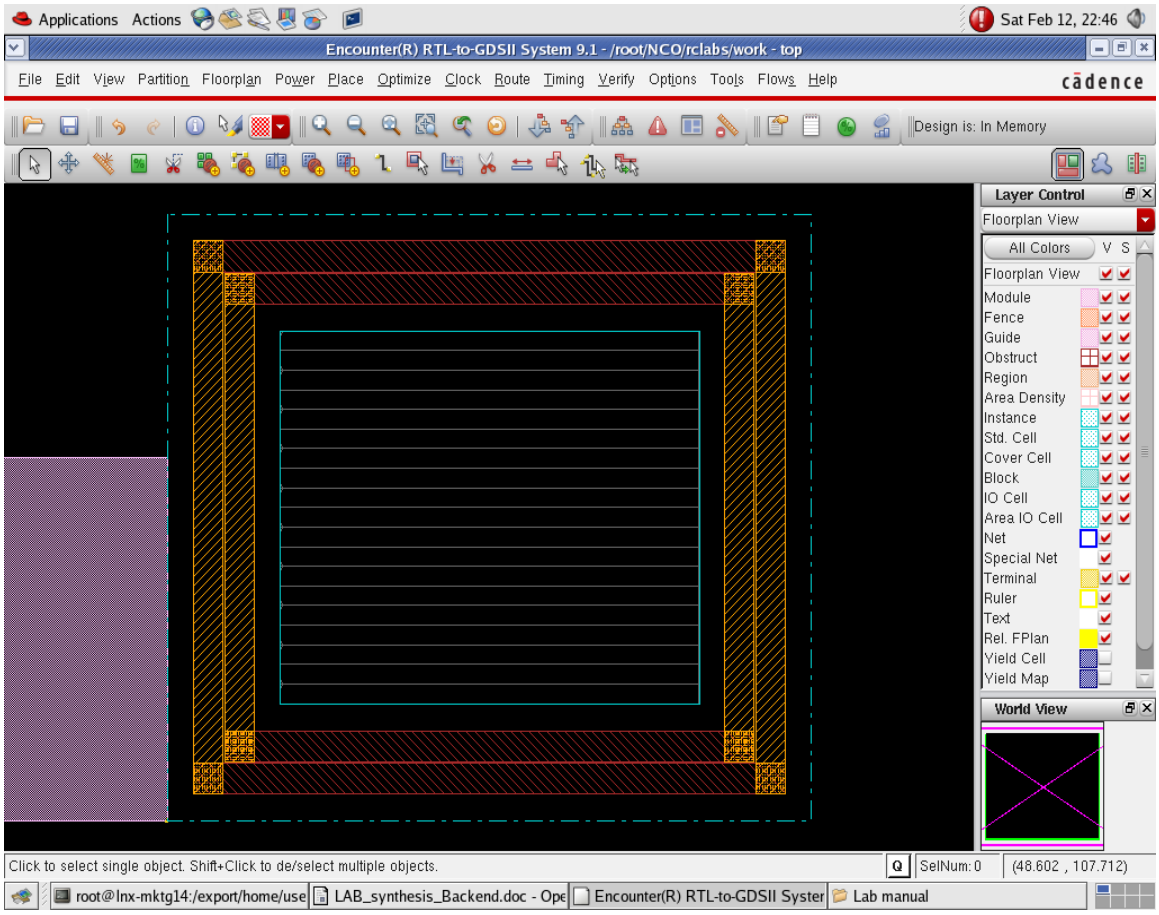
The core dimensions are changed.
24.Click on Floorplan and select Automatic Floorplan and select Plan Design. Click Ok. This will automatically put the Macros if there are any in the design.

25.Next step is to do power planning. Click on power, select power planning and click on Add Rings.

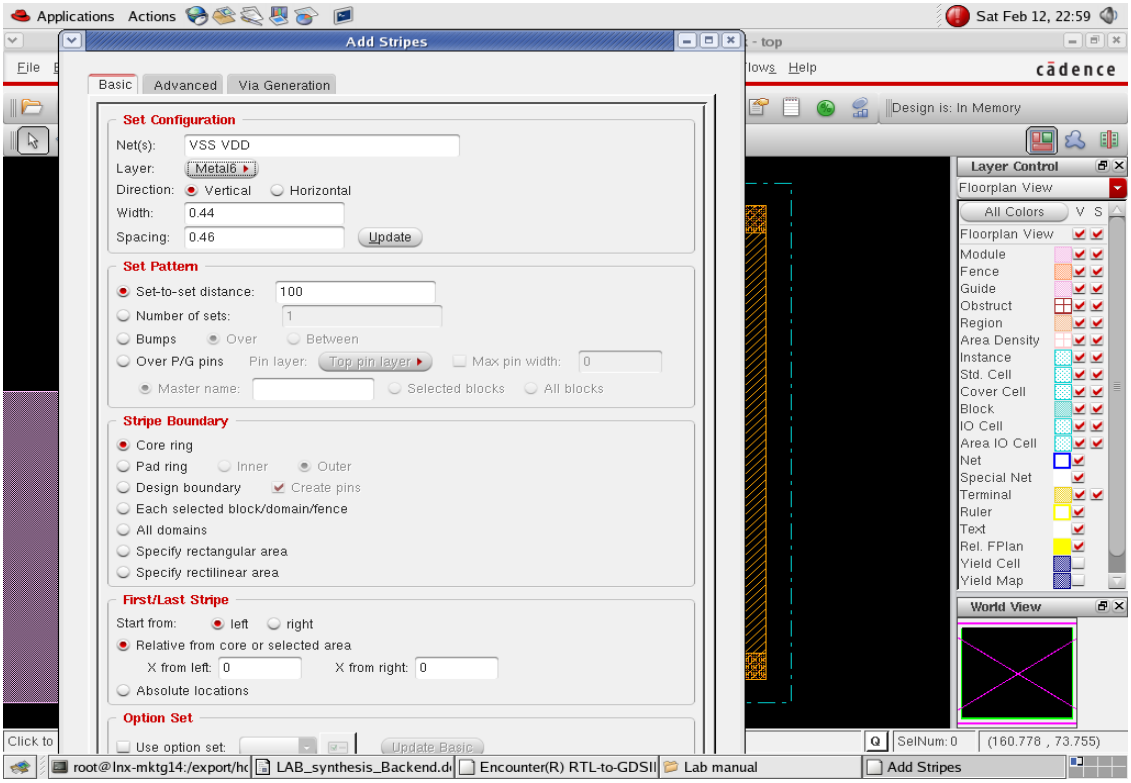


26. Select the top and bottom layer as Metal5, Left and Right as Metal6. Set the width as per the requirement and taking the space between core boundary and I/O pad considerations. Select the option for offset as “center in channel” and click OK.

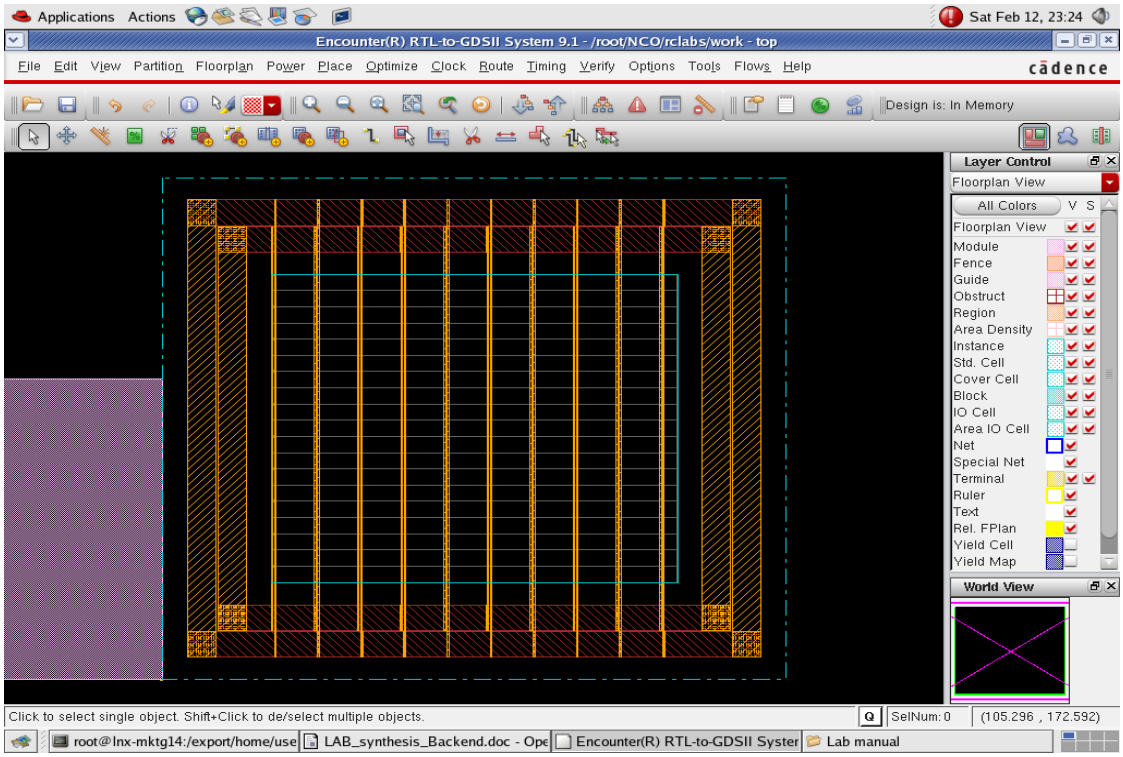
The power ring will get created in between the channel. The image on the next page is showing the power ring created.



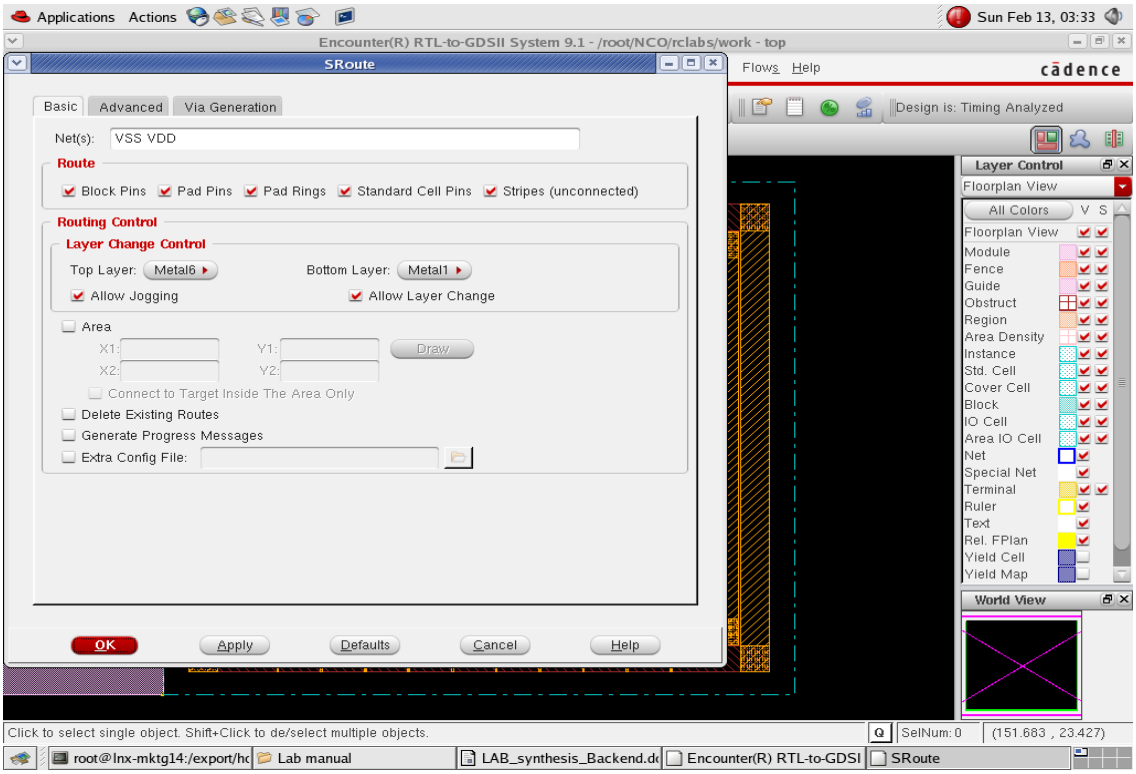
27. The next step in power planning is to create power strips. Select Power, click Power Planning and click Add Stripe.



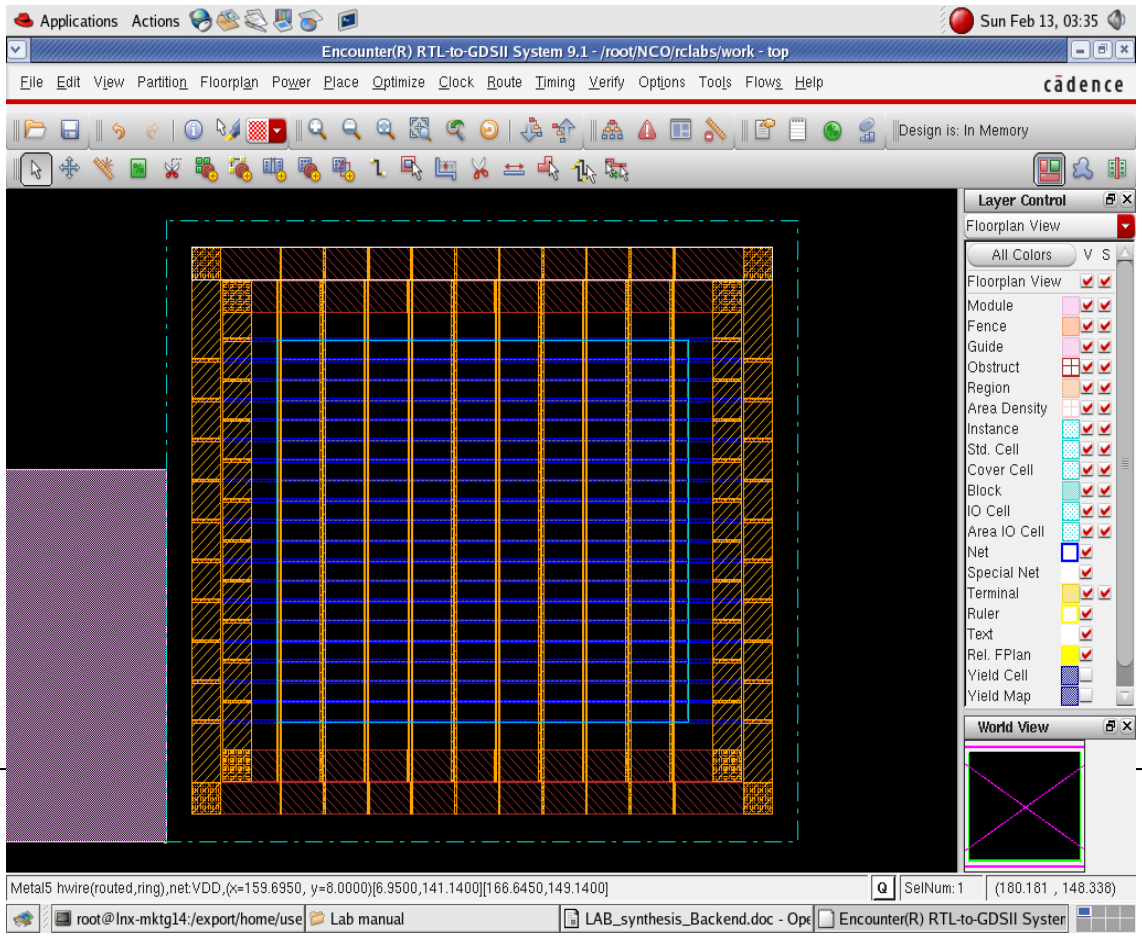
28. For adding the stripes, select metal layer as Metal 6 and chose direction as vertical(if direction chosen is horizontal, chose metal layer as Metal 5). Click OK and the design will get the vertical thin strips of type Metal 6.



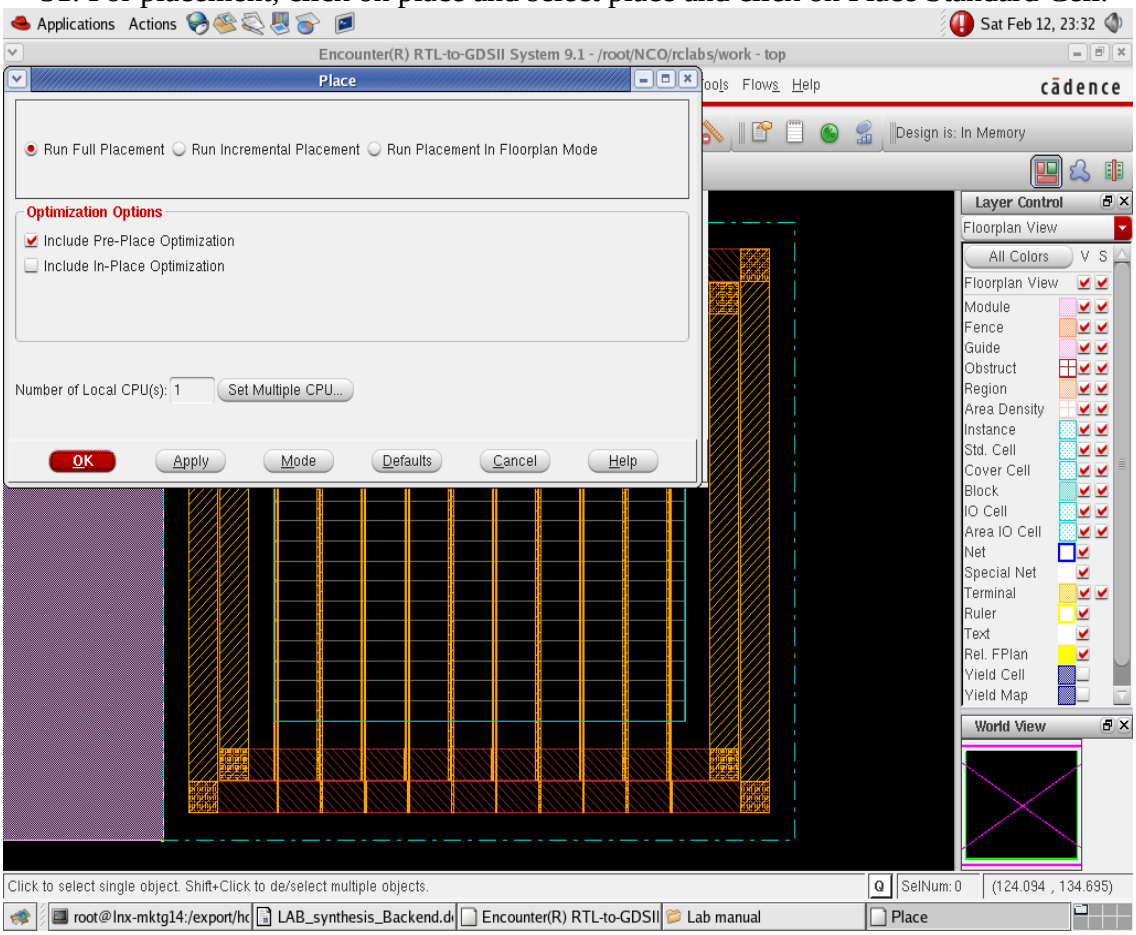
29. After the power planning, go to Route and click Special Route. A new Window Sroute will appear.



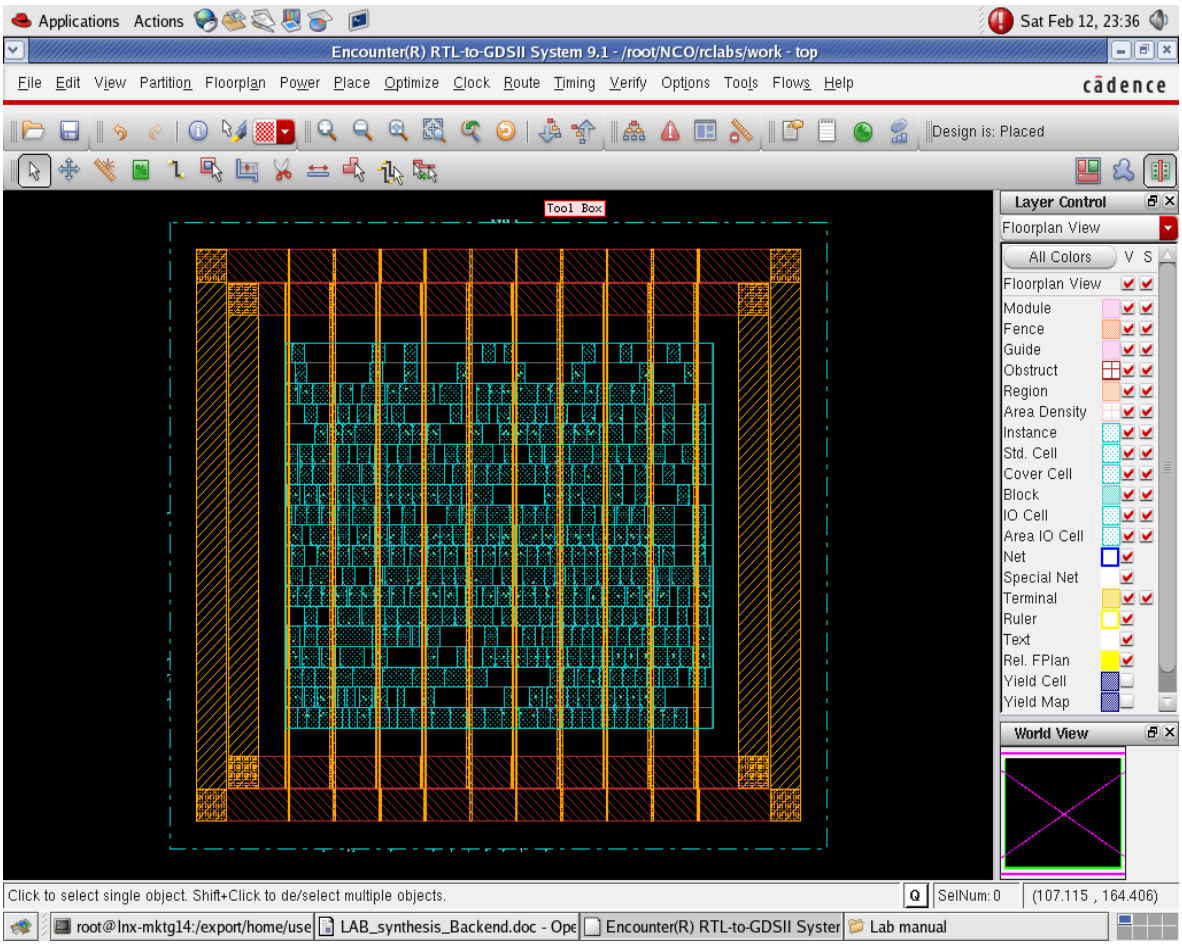
30. Click OK with all default settings. This is done to provide power to standard cells. The horizontal blue coloured metal1 stripes created as a result of Special Route.



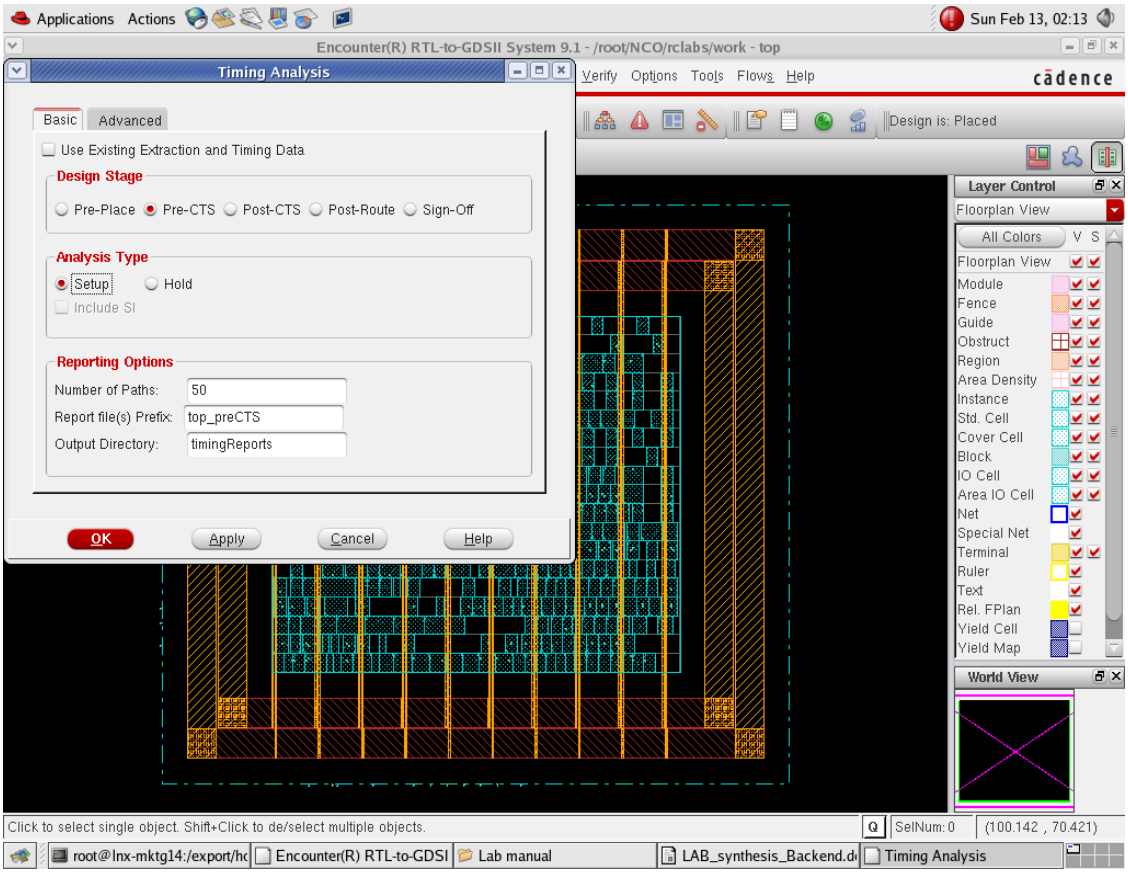
31. For placement, click on place and select place and click on Place Standard Cell.



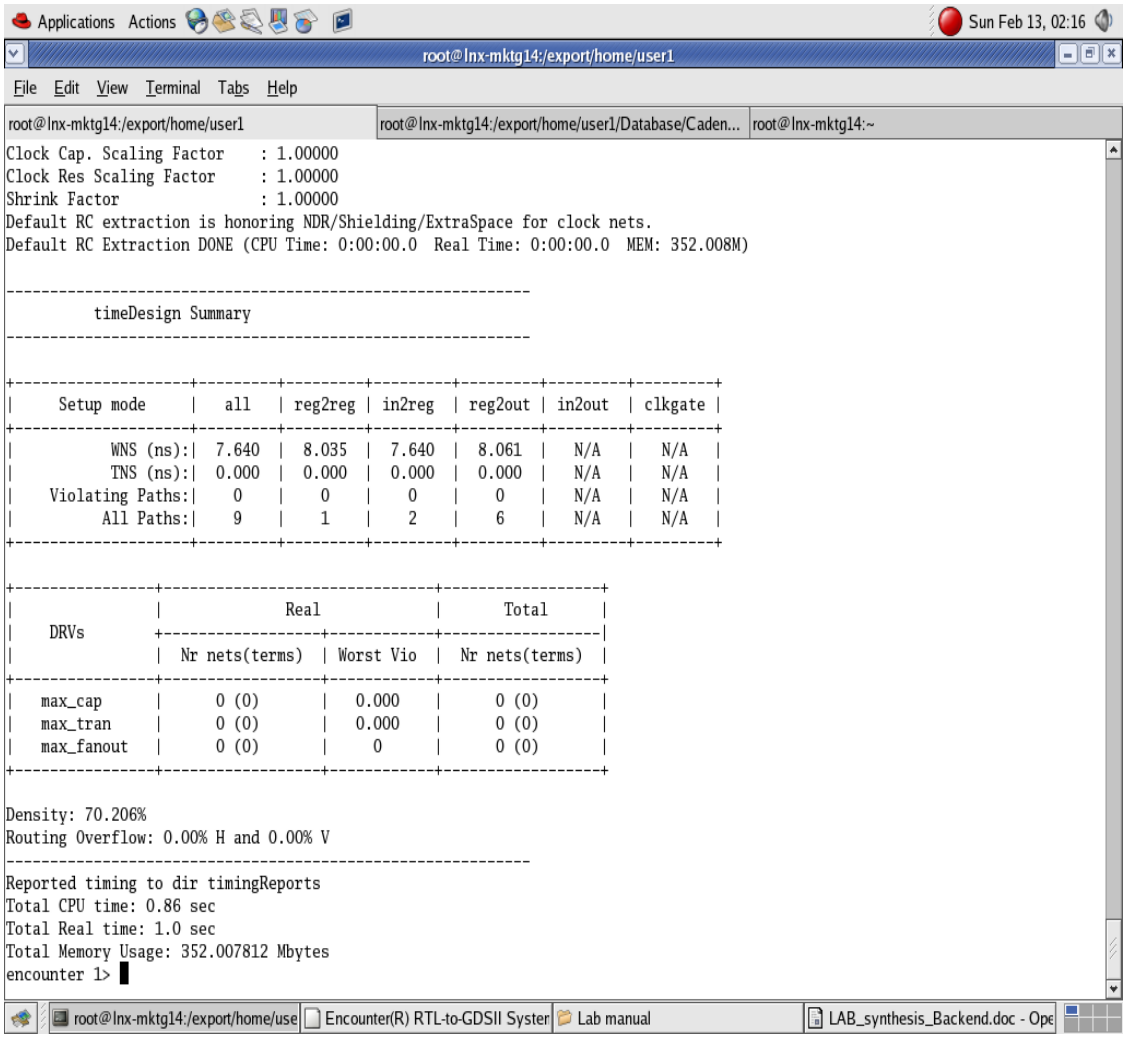
32. Click OK on Place window and in physical view the blue coloured standard cells can be seen as a result of placement of standard cells.

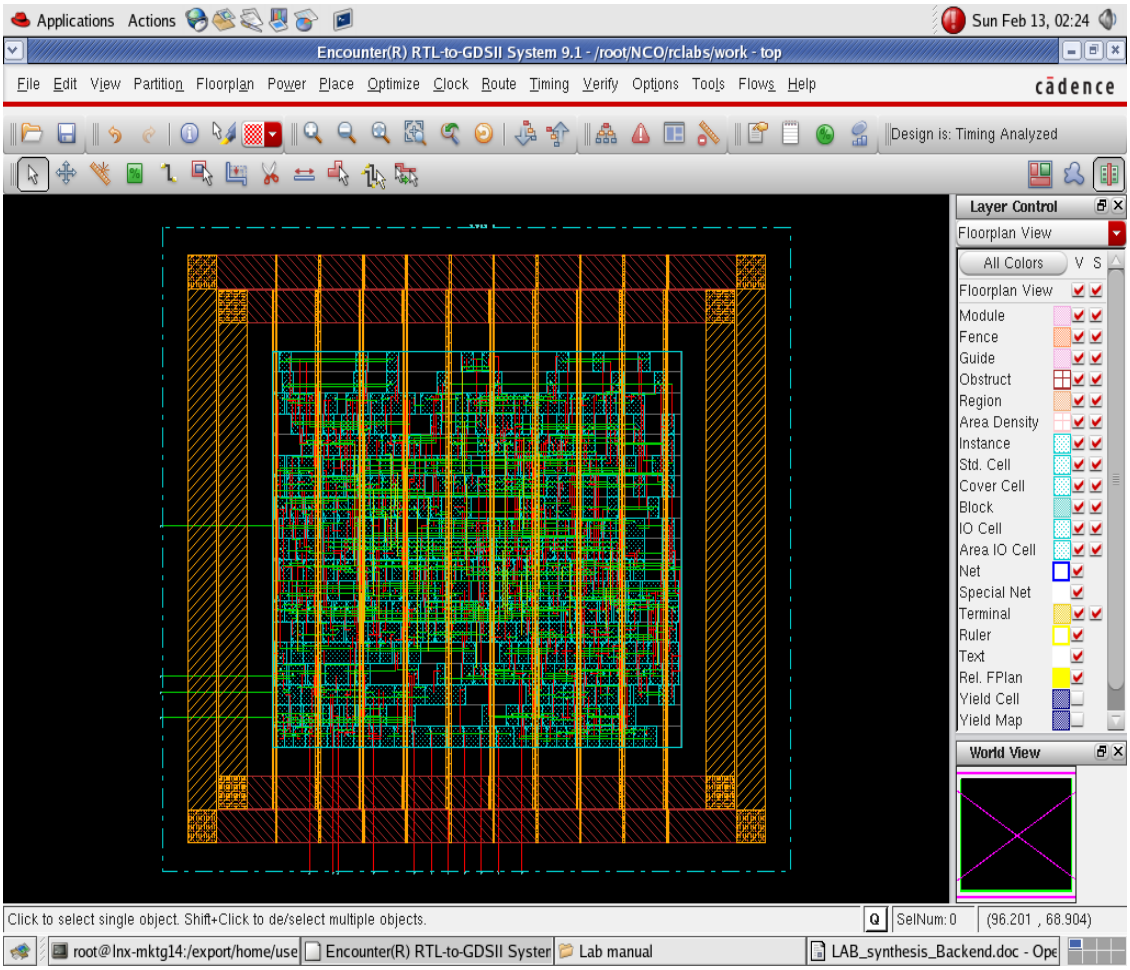


33. Before CTS, timing analysis has to be done for any setup violations. Click on Timing, and select Report Timing. A Timing analysis window will get open. In the window select the “Pre-CTS” as Design Stage and select the “Setup” as Analysis Type.

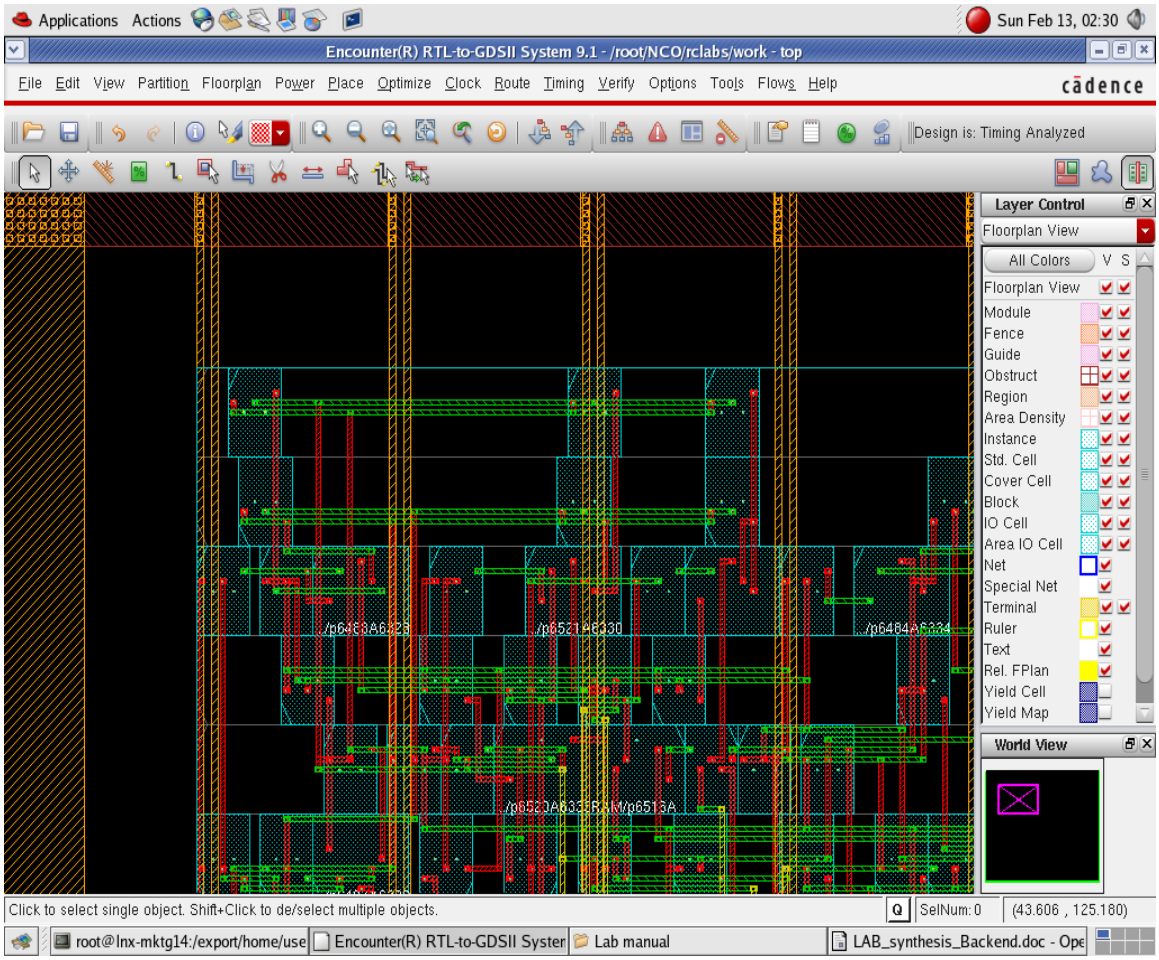

Applications Actions
Sun Feb 13, 02:13
Encounter(R) RTL-to-GDSII System 9.1 - /root/NCO/rclabs/work - top
Timing Analysis
Basic
Advanced
Use Existing Extraction and Timing Data
Design Stage
Pre-Place
Pre-CTS
Post-CTS
Post-Route
Sign-Off
Analysis Type
Setup
Hold
Include SI
Reporting Options
Number of Paths:
50
Report file(s) Prefix:
top_preCTS
Output Directory:
timingReports
OK
Apply
Cancel
Help
Verify Options Tools Flows Help
cadence
Design is: Placed
Layer Control
Floorplan View
All Colors
Module
Fence
Guide
Obstruct
Region
Area Density
Instance
Std. Cell
Cover Cell
Block
IO Cell
Area IO Cell
Net
Special Net
Terminal
Ruler
Text
Rel. FPlan
Yield Cell
Yield Map
World View
Click to select single object. Shift+Click to de/select multiple objects.
SelNum: 0
(100.142 , 70.421)
root@lnx-mktg14:/export/hc
Encounter(R) RTL-to-GDSI
Lab manual
LAB_synthesis_Backend.d
Timing Analysis

34. Click OK to complet the Timing analysis. The timing information will get display on terminal in tabular form. In the table displayed on the terminal under “timeDesign Summary”, check for any negative value under WNS(Worst Negative Slack) and TNS(Total Negative Slack). The terminal will look as the image below and Tool window as on next page.



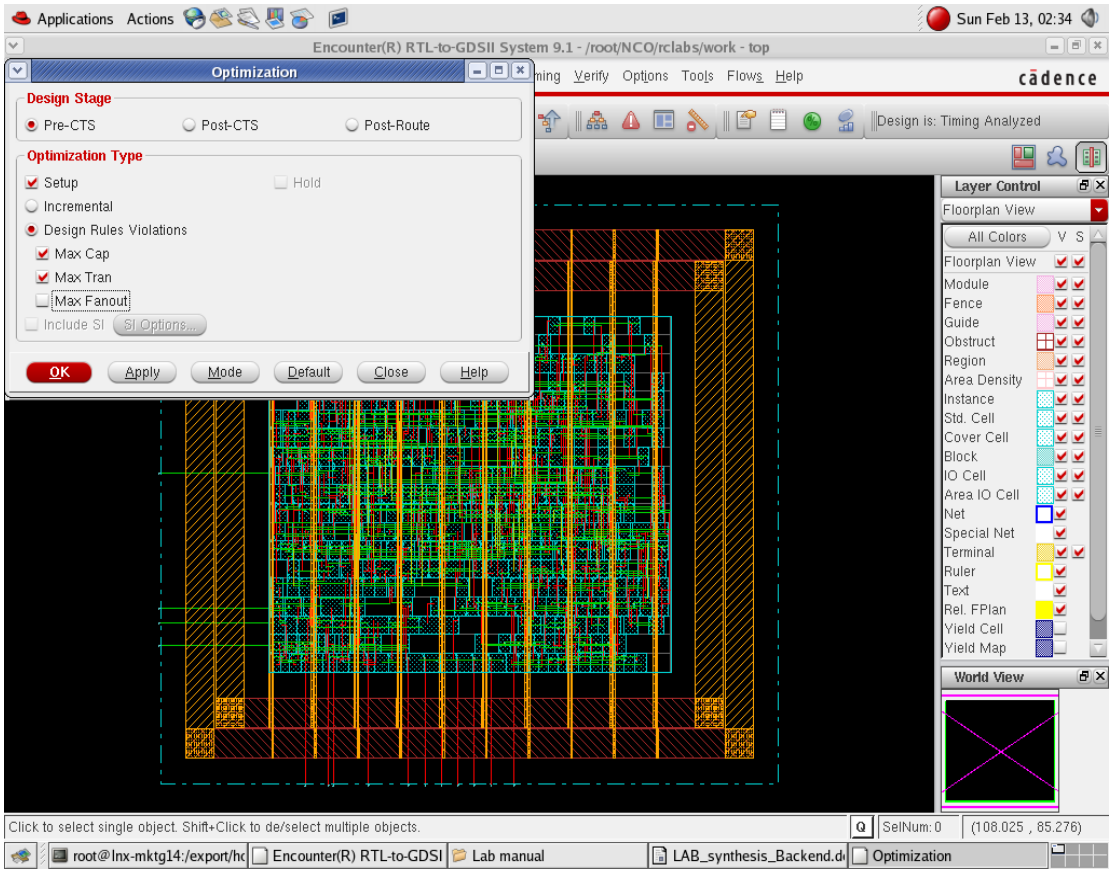


The multi-coloured lines visible in the tool window are the connections between standard cells using metal layers. If any part of this design is Zoom-in, metal layers can be viewed easily.



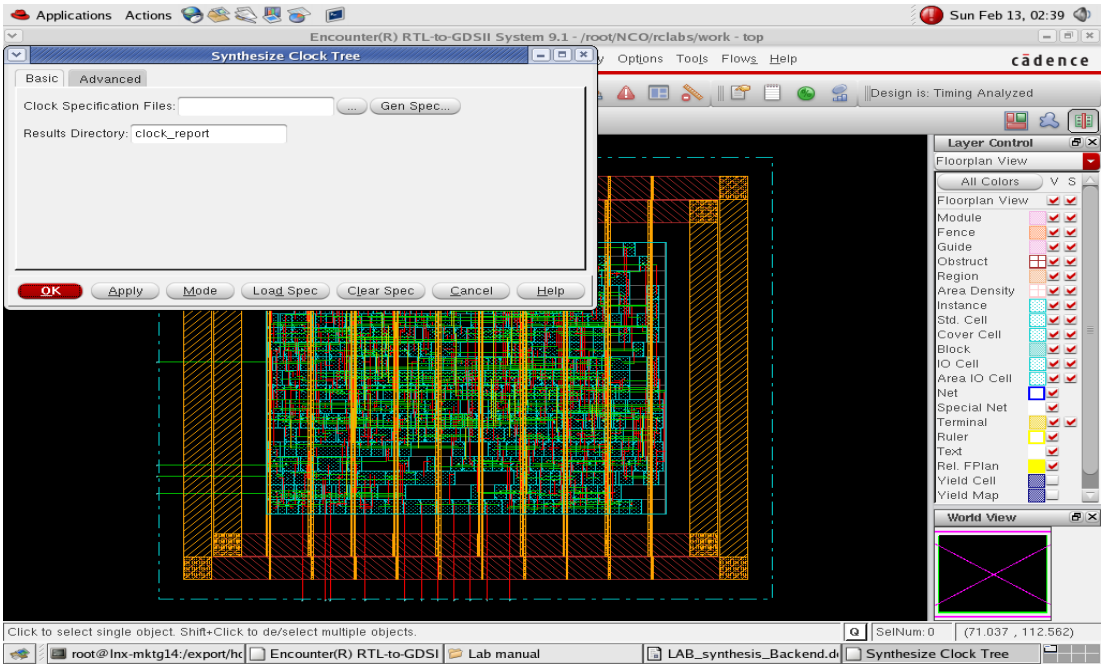
Different colours show different metal

35. If there is any of the negative slack value under WNS or TNS, click Optimize in Tool window and Select Optimize Design. A new window “Optimization” will get open. Select “Pre-CTS” as Design Stage and “Setup” as optimization type and click OK. The tool will optimize the design and the optimized timing results will be displayed over terminal again.

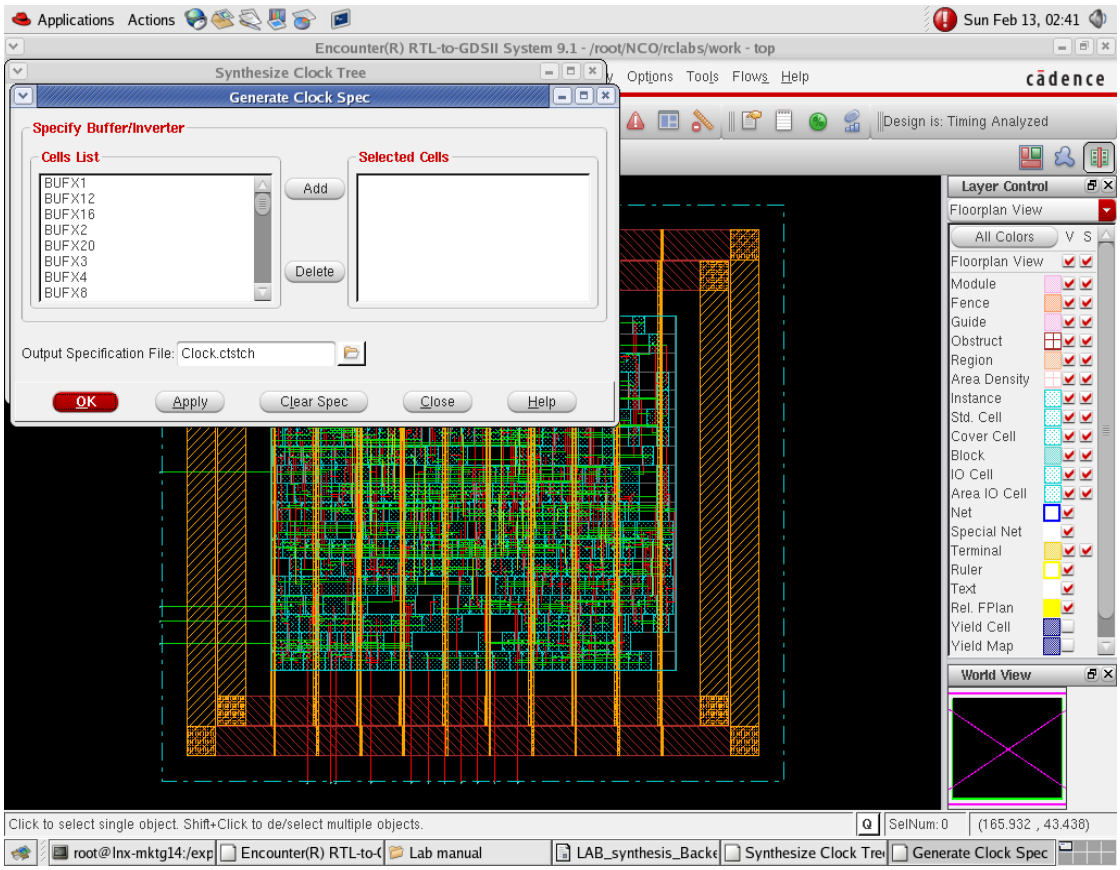


In this case we did not get any negative slack, so this step is skipped here.

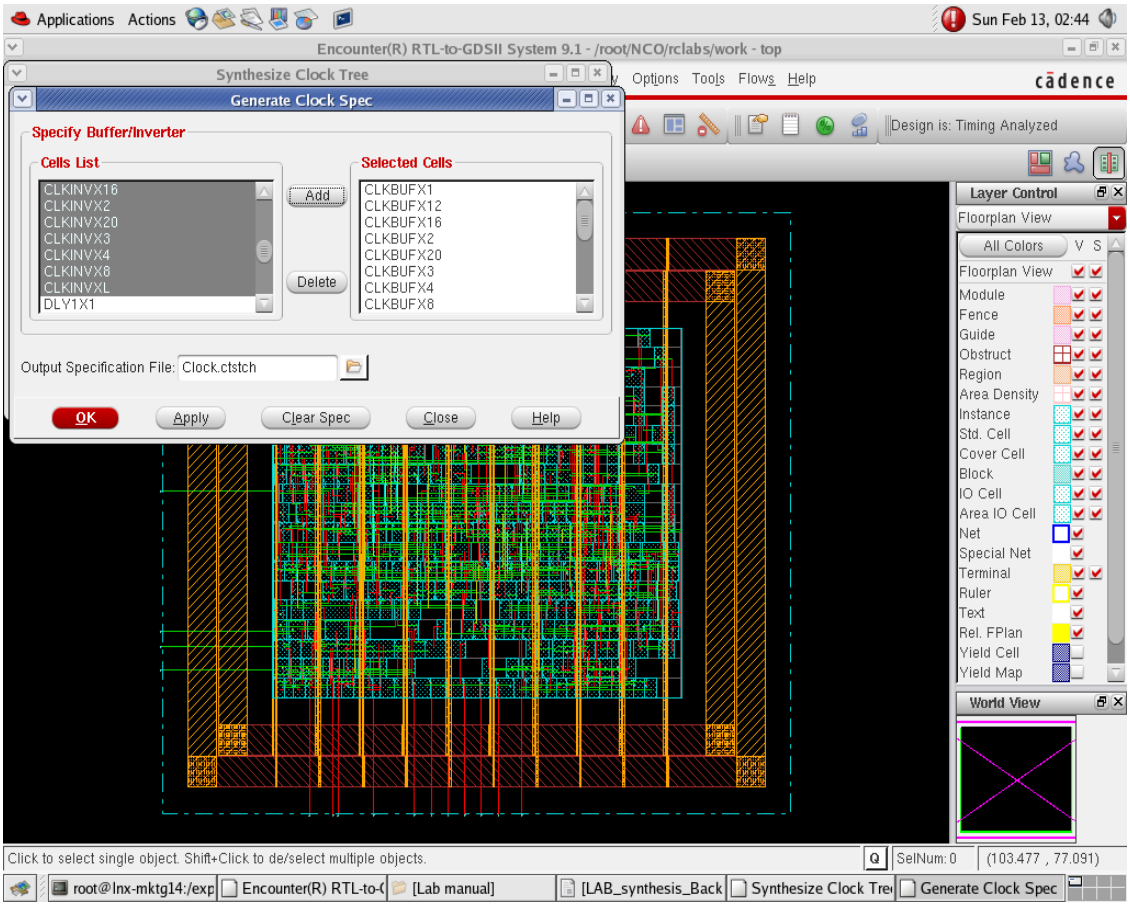
36. Go to Clock, click “Synthesize Clock Tree”, a new window “Synthesize Clock Tree” will get open.



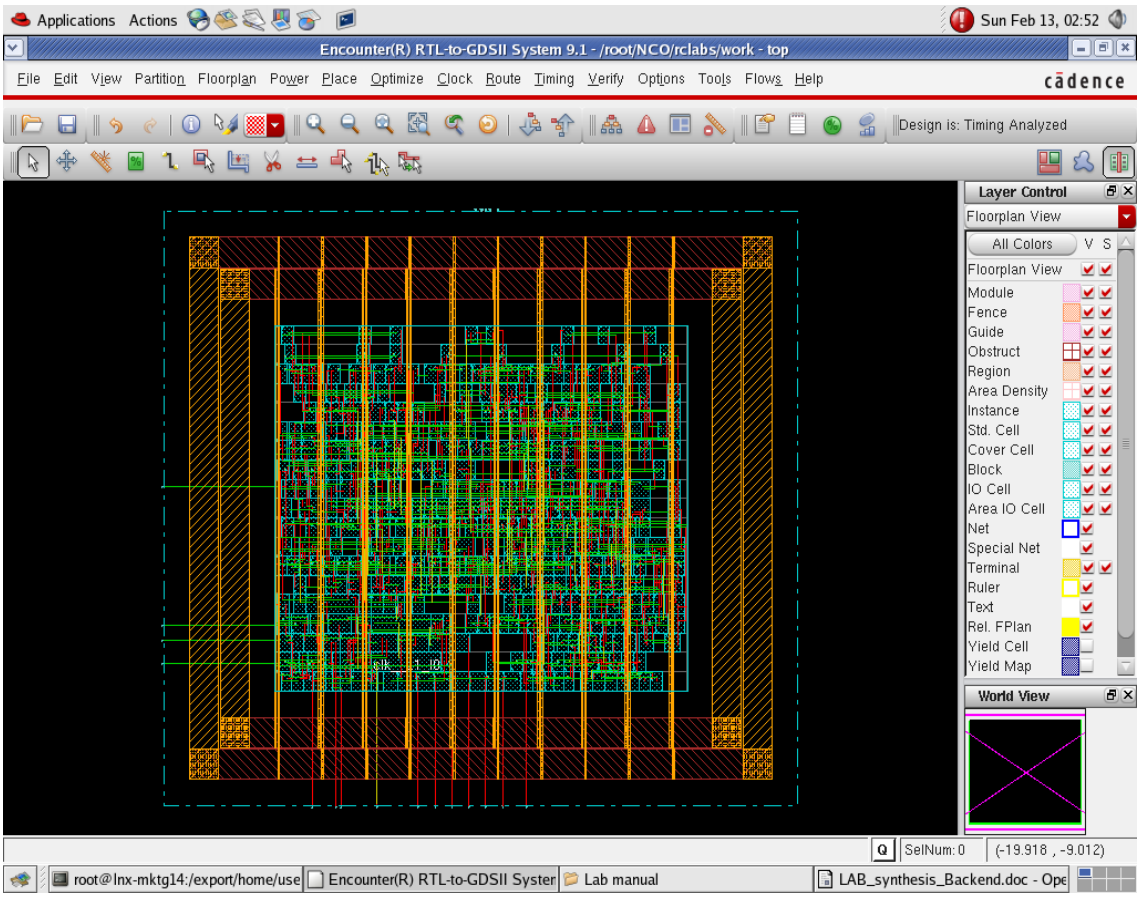
37. Click on Gen Spec and a new window “Generate Clock Spec” will open.



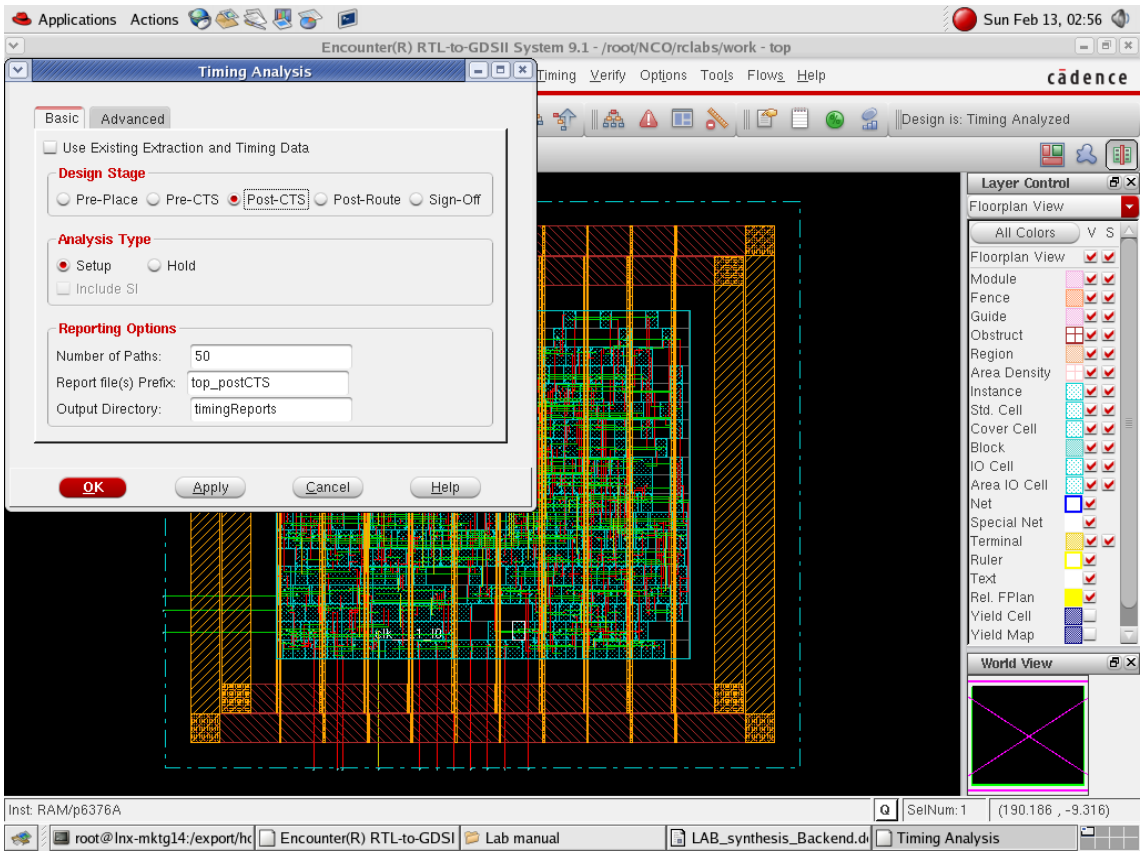
38. From Cells List, Select all clocks starting with “CLK” and click on Add button to add them to the Selected Cells. Select a name for Output specification.



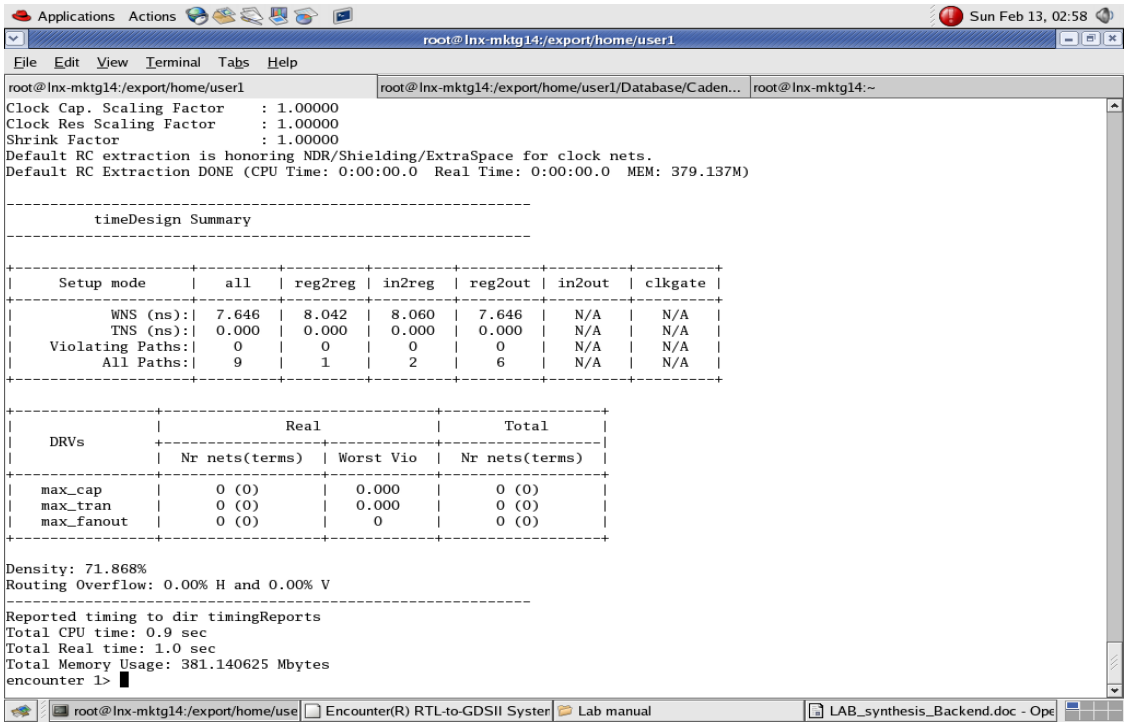
39. Click OK. Then specify a name for Results Directory. and click OK. The tool window looks like the image below.



40. Again Perform the Timing by clicking on Timing and selecting Report Timing. Select “Post-CTS” under Design Stage and do the select “Set-up” as Analysis Type.

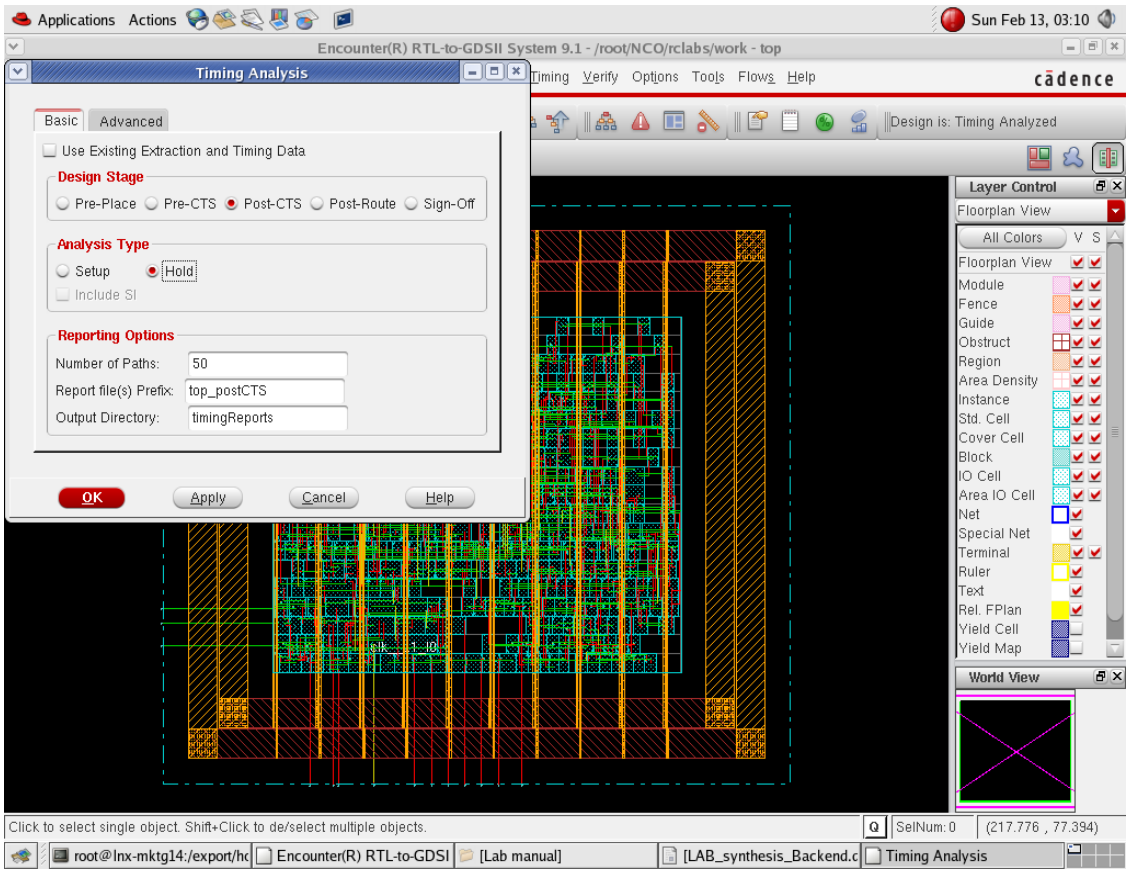


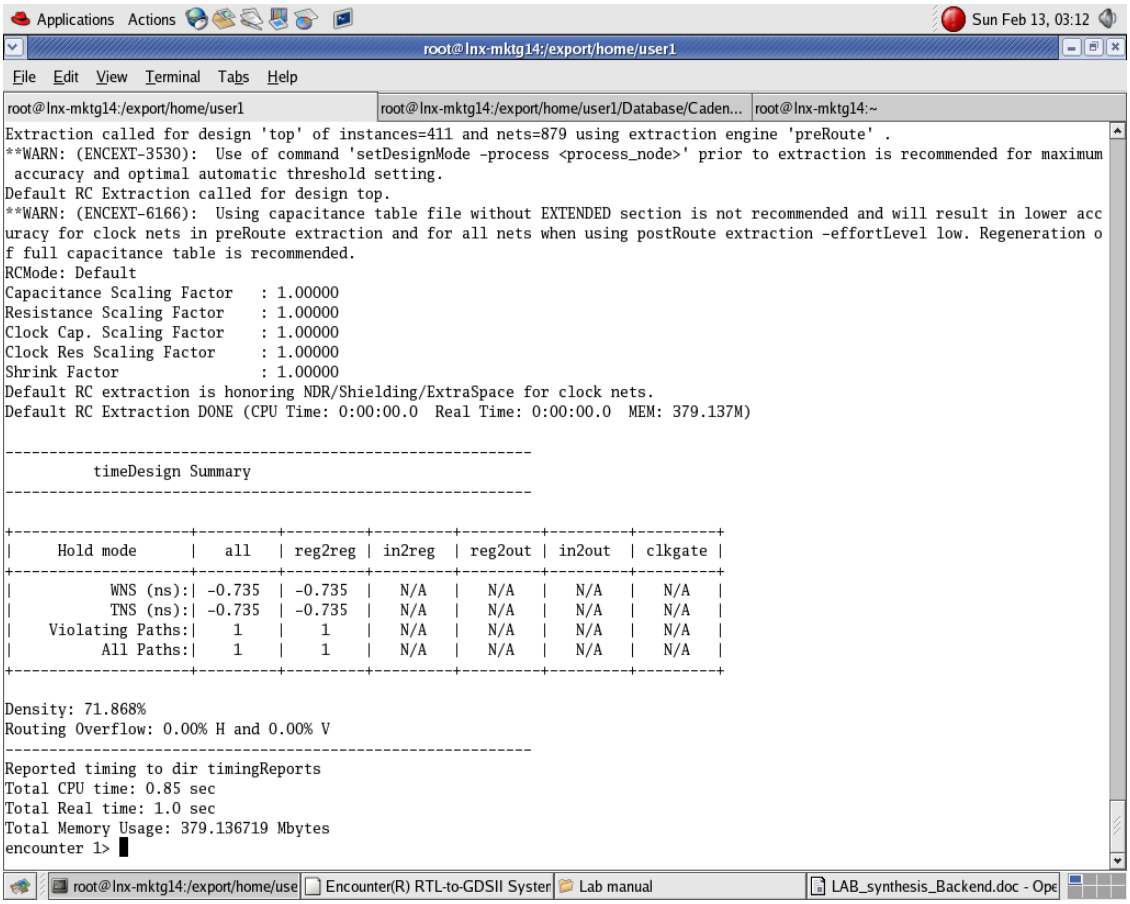
41. Click Ok to perform the timing. The timing information will be displayed over the terminal window. Again check for any negative slacks under WNS or TNS.



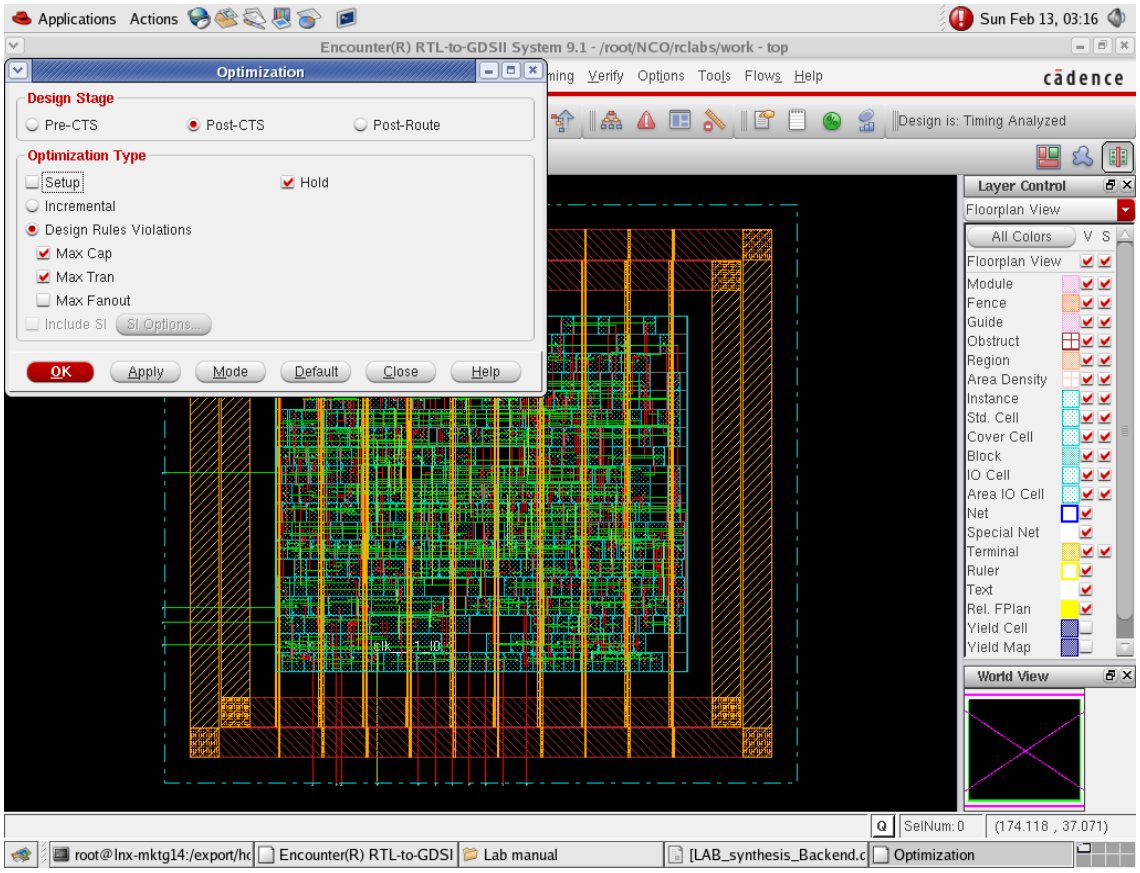
42. If there is any negative value found for either of WNS or TNS then perform the Optimization Technique to reduce the negative slack. No negative slack is found in the terminal image on previous page so this step is skipped here.

43. Timing Analysis for “Setup” as Analysis Type is done. Repeat Step 27 for performing timing for “Post CTS” as Design Stage and “Hold” as Analysis Type. The tool will show the timing results in the terminal window.

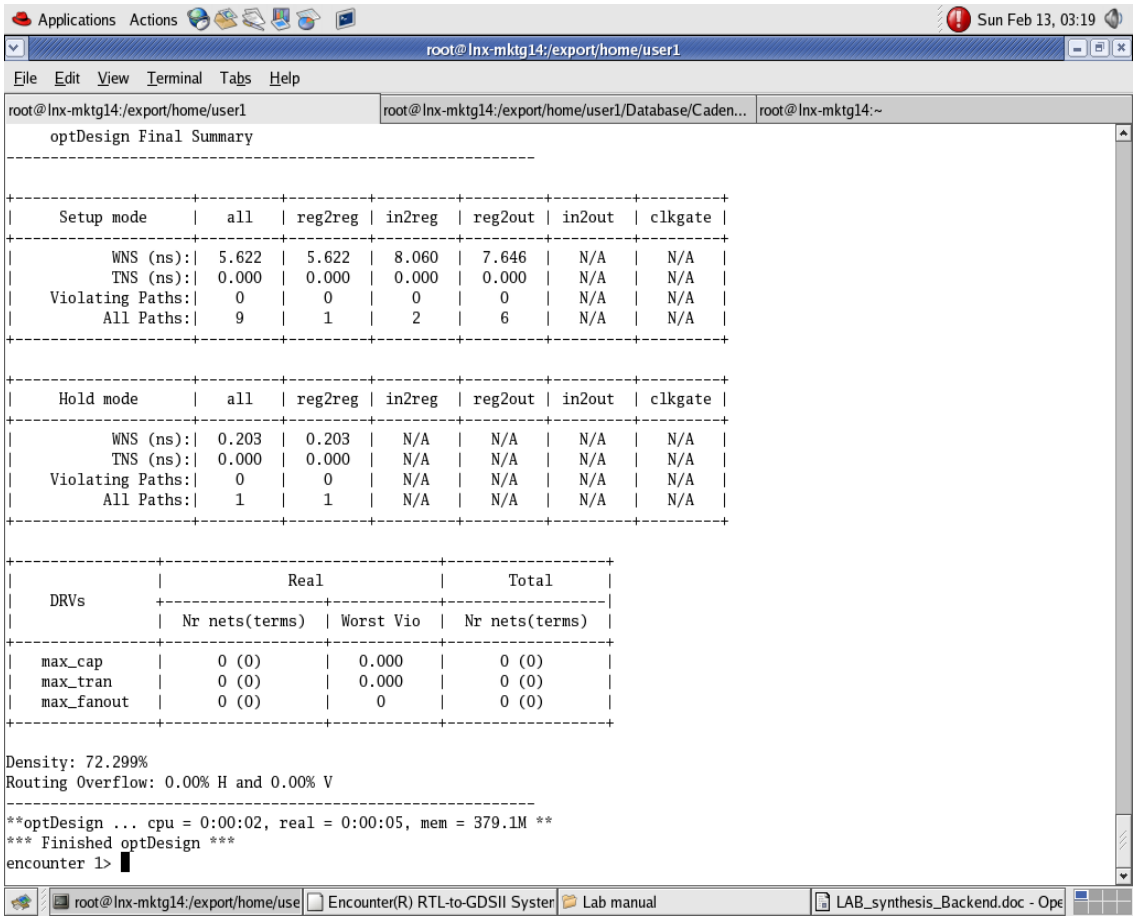

Applications Actions
Sun Feb 13, 03:10
Encounter(R) RTL-to-GDSII System 9.1 - /root/NCO/rclabs/work - top
Timing Verify Options Tools Flows Help
cadence
Design is: Timing Analyzed
Timing Analysis
Basic
Advanced
Use Existing Extraction and Timing Data
Design Stage
Pre-Place
Pre-CTS
Post-CTS
Post-Route
Sign-Off
Analysis Type
Setup
Hold
Include SI
Reporting Options
Number of Paths:
50
Report file(s) Prefix:
top_postCTS
Output Directory:
timingReports
OK
Apply
Cancel
Help
Layer Control
Floorplan View
All Colors
V S
Floorplan View
Module
Fence
Guide
Obstruct
Region
Area Density
Instance
Std. Cell
Cover Cell
Block
IO Cell
Area IO Cell
Net
Special Net
Terminal
Ruler
Text
Rel. FPlan
Yield Cell
Yield Map
World View
Click to select single object. Shift+Click to de/select multiple objects.
SelNum: 0
(217.776 , 77.394)
root@lnx-mktg14:/export/hc
Encounter(R) RTL-to-GDSI
[Lab manual]
[LAB_synthesis_Backend.c
Timing Analysis



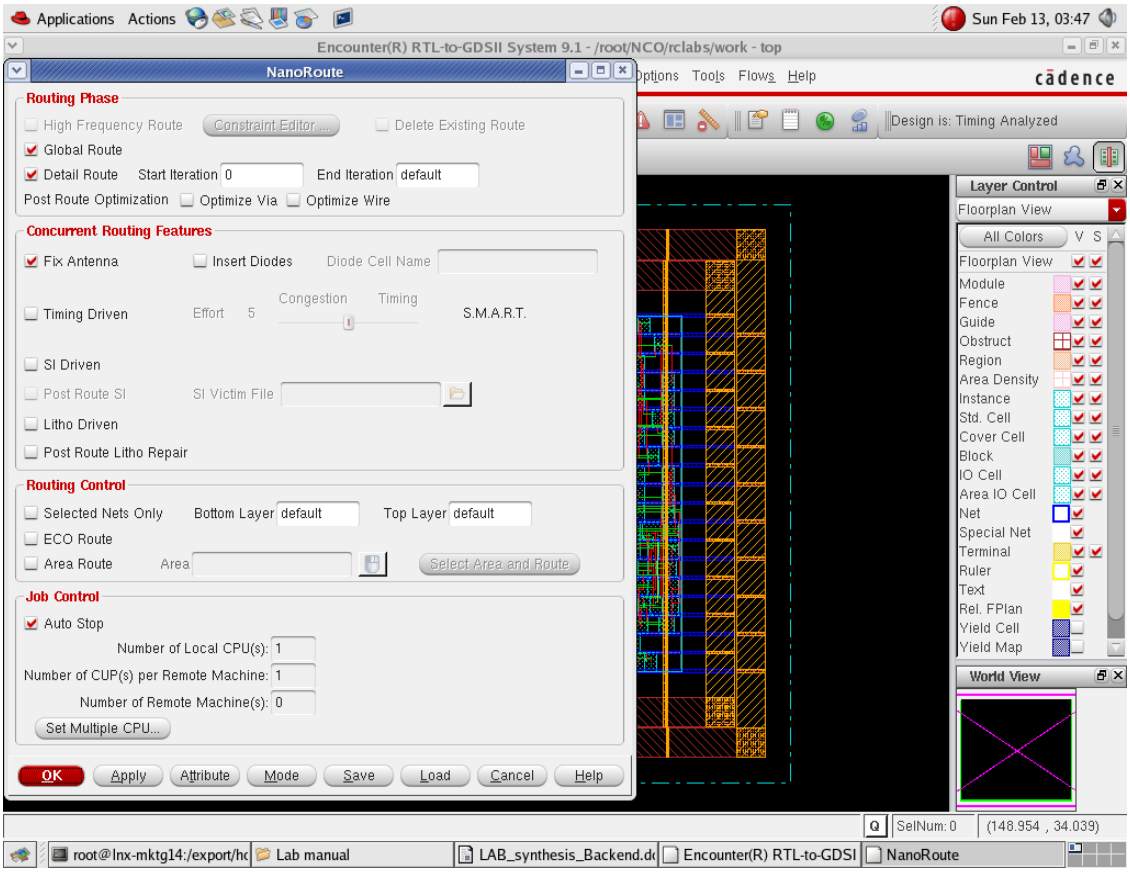
44. After Timing Analysis is performed, the timeDesign Summary is showing the negative slack values for both TNS and WNS. Perform the Optimization. Go to Optimize and click on Optimize Design. Select “Post-CTS” as and “HOLD” as the Optimization Type



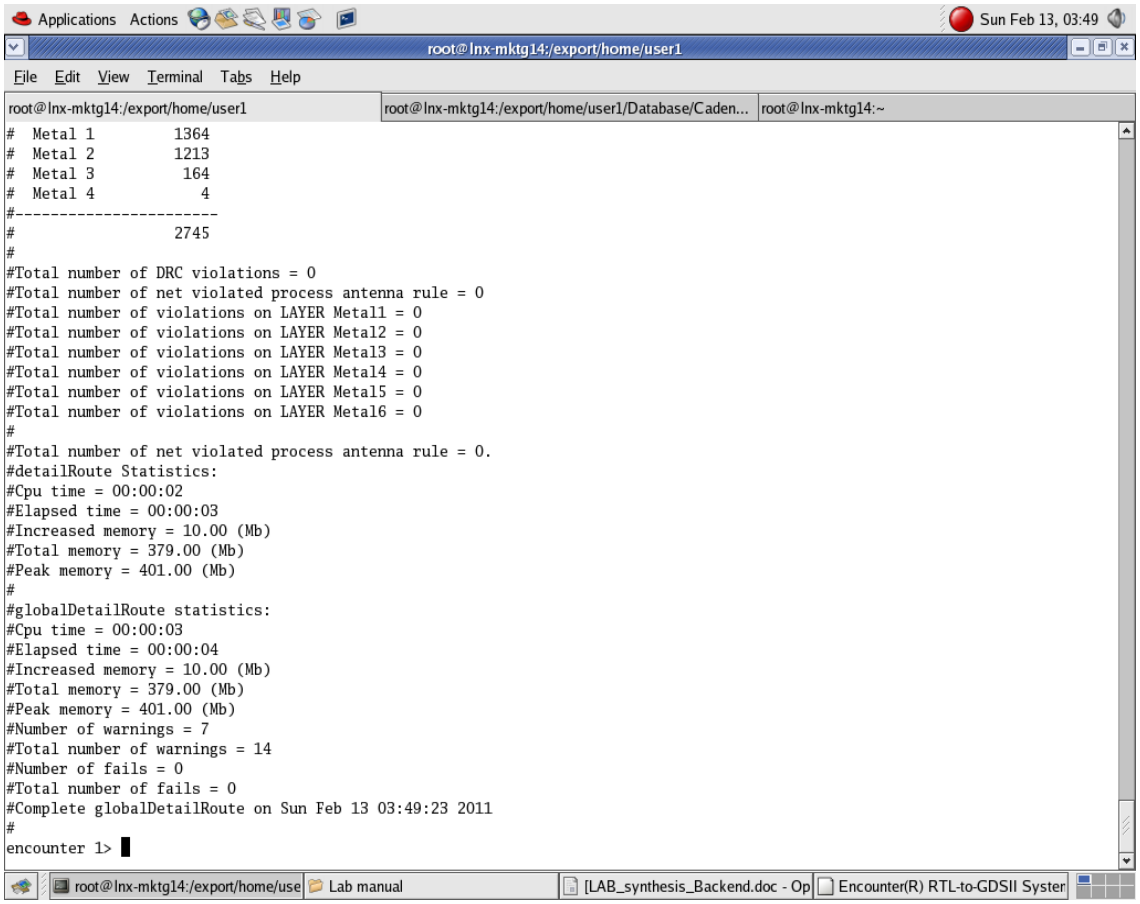
45. Click OK to perform the Optimization and Tool will perform the optimization and displays the optimized results in the terminal window under timeDesign Summary. The results of Optimization can be seen on the next page in tabular form for both Setup and Hold mode. As compare to the Timng Results performed for Hold mode in Step 30, the design has been optimized and tabular results shows that all slack values are now positive values and no more negative values for slack.



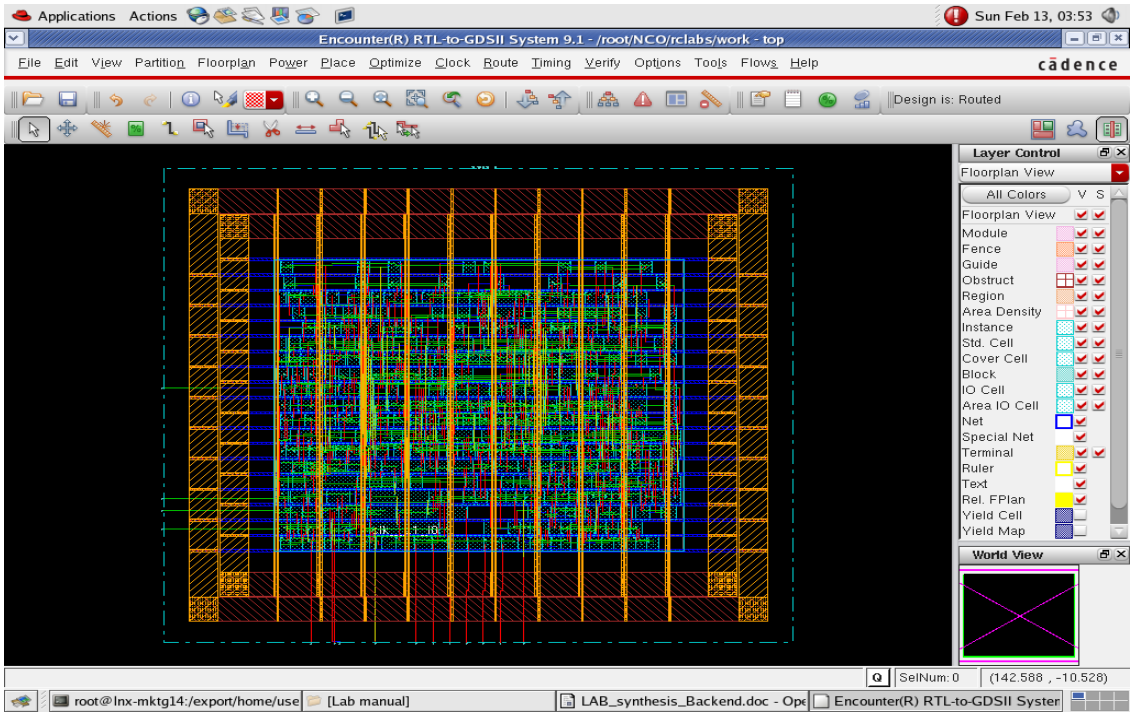
46. Perform Routing by clicking Route, and select NanoRoute and then click Route. A window NaoRoute will open.



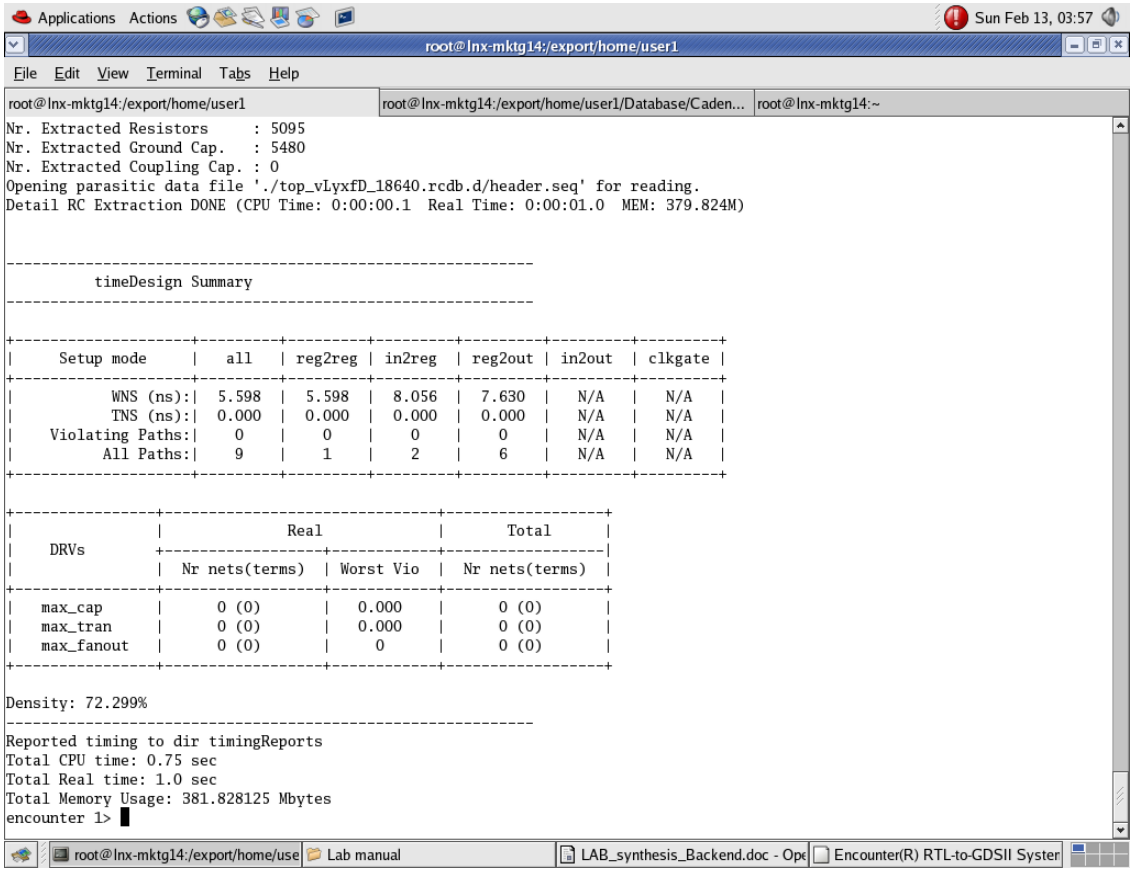
47. Click Ok to Perform Routing. The tool will Perform the Routing and the Routing statistics can be seen on terminal window including DRC violations.



48. After routing tool window looks like the below image.

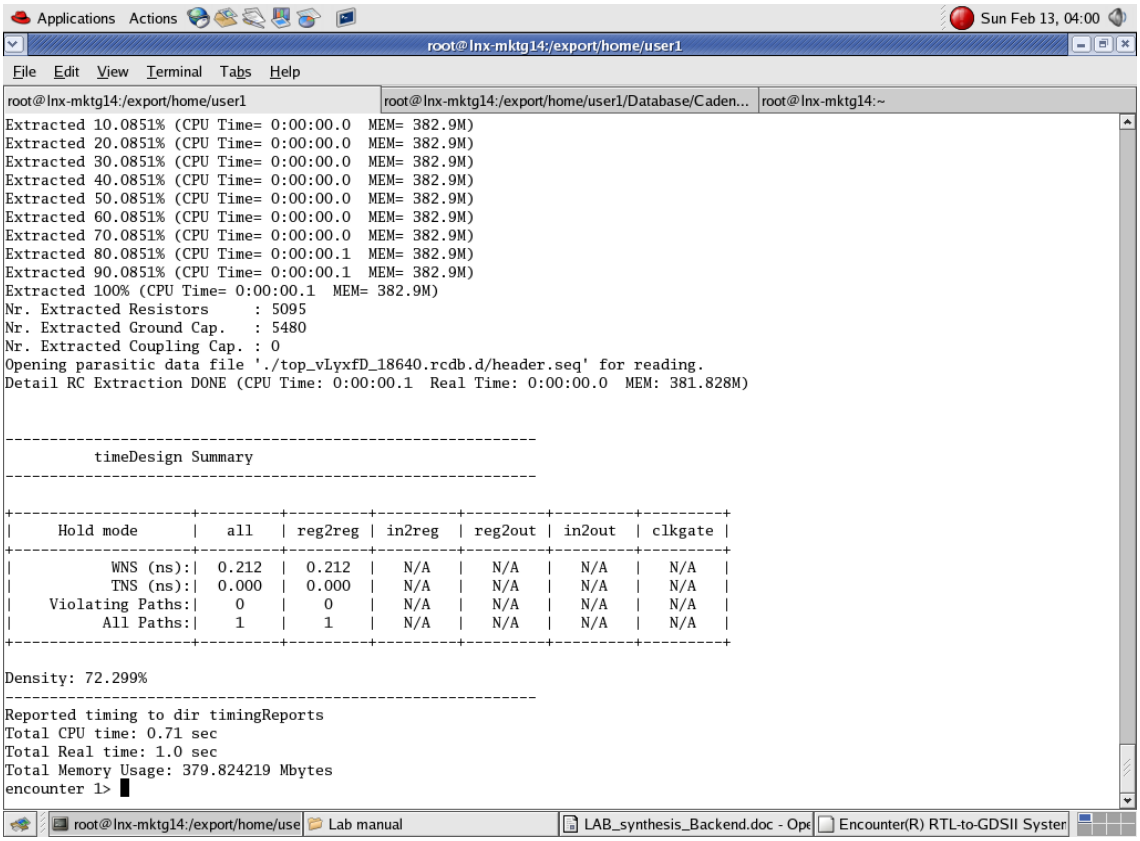


49. Perform the timing again. Go to Timing, seelct Report Timing and a Timing Analysis window will get open. Select “Post-Route” as the Design Stage and “Setup” as Analysis Type. Click Ok. The timing results will be displayed in terminal window for Set up mode.



Since there is no negative value of slack so design does not require optimization for Set-up mode in Post-Route stage.

50. Repeat Step 36 for “Post-Route” as Design Stage and “Hold” as the Analysis Type. Click OK. The timing results can be seen in the terminal window for hold mode.



As there is no negative value of slack, the optimization is not require to perform. The final view of the circuit is as below: