Área Científica
de Electrónica

Microelectronica

Standard Cell Placement and Routing Tutorial

AustriaMicroSystems C35B3 (HIT-Kit 3.70)

Marcelino Santos, Sílvia Gomes

Junho 2007

Table of contends

Starting Encounter for the first time

At first, open a terminal program and source the configuration file cad.init.

```
> source cad.init
```

Then create a directory for the project files and change to that directory.

```
> mkdir cds_encounter_tutorial
> cd cds_encounter_tutorial
```

Then a script from AustriaMicroSystems is used to initialize the Cadence Custom IC Design tools with the process technology C35B3 (*-t c35b3*).

- `ams_encounter -t c35b3`
- `encounter`

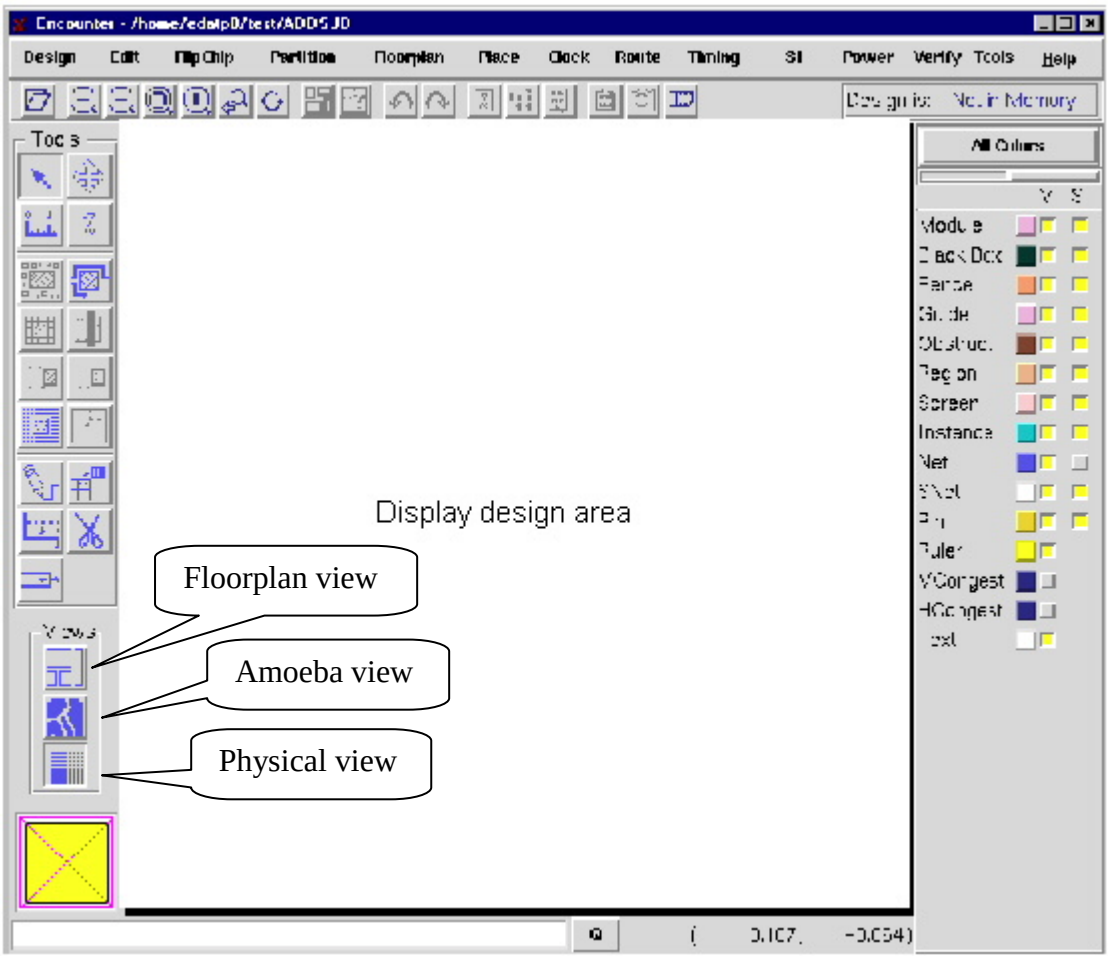

Fig.1 – Encounter graphical interface.

The Unix shell from which the tool is started is called the Encounter console. The console displays the "encounter>" prompt. This is where you can enter all Encounter text commands and where the tool displays messages.

The main window includes three different design views (see Fig.1) that you can toggle during a session:

- The Floorplan view, the Amoeba view, and the Physical view. The Floorplan view displays the hierarchical module and block guides, connection flight lines, and floorplan objects, including block placement, and power/ground nets.

- The Amoeba view displays the outline of the modules and sub modules after placement, showing physical locality of the module.

- The Physical view displays the detailed placements of the module's blocks, standard cells, nets, and interconnects.

In the left bottom of the main window you have a satellite window, which identifies the location of the current view in the design display area, relative to the entire design. The chip area is identified by a yellow box; the satellite view is identified by the pink cross box. When you display an entire chip in the design display area, the satellite cross box encompasses the chip area yellow box. When you zoom and pan through the chip in the design display area, the satellite cross box identifies where you are relative to the entire chip.

• To move to an area in the design display area, click and drag on the satellite cross box.
• To select a new area in the design display area, click and drag on the satellite cross box.
• To resize an area in the satellite window, click with the Shift key and drag a corner of the cross box.
• To define a chip area in the satellite window, right-click and drag on an area.

Starting Encounter again

If you want to start a new Encounter session in a directory where the Cadence tool has already been initialized, only the following commands are necessary:

```
> source cad.init
> cd cds_encounter_tutorial
> encounter
```

Design import

Importing the design into Encounter involves specifying the following setup information:
• **Design libraries and files**. This includes information on the technological process and the cell library in the LEF (Layout Exchange Format) format. LEF files provide information such as metal and via layers and via generate rules which are used for routing tasks. They also provide the minimum information on cell layouts for placement and routing.
• **Gate-level netlist**. This relates to the (Verilog) netlist to be placed and routed.
• **Timing libraries**. This includes information on the cell timings (delays, setup/hold times, etc.).
• **Power information**. This relates to the power nets to use in the layout.

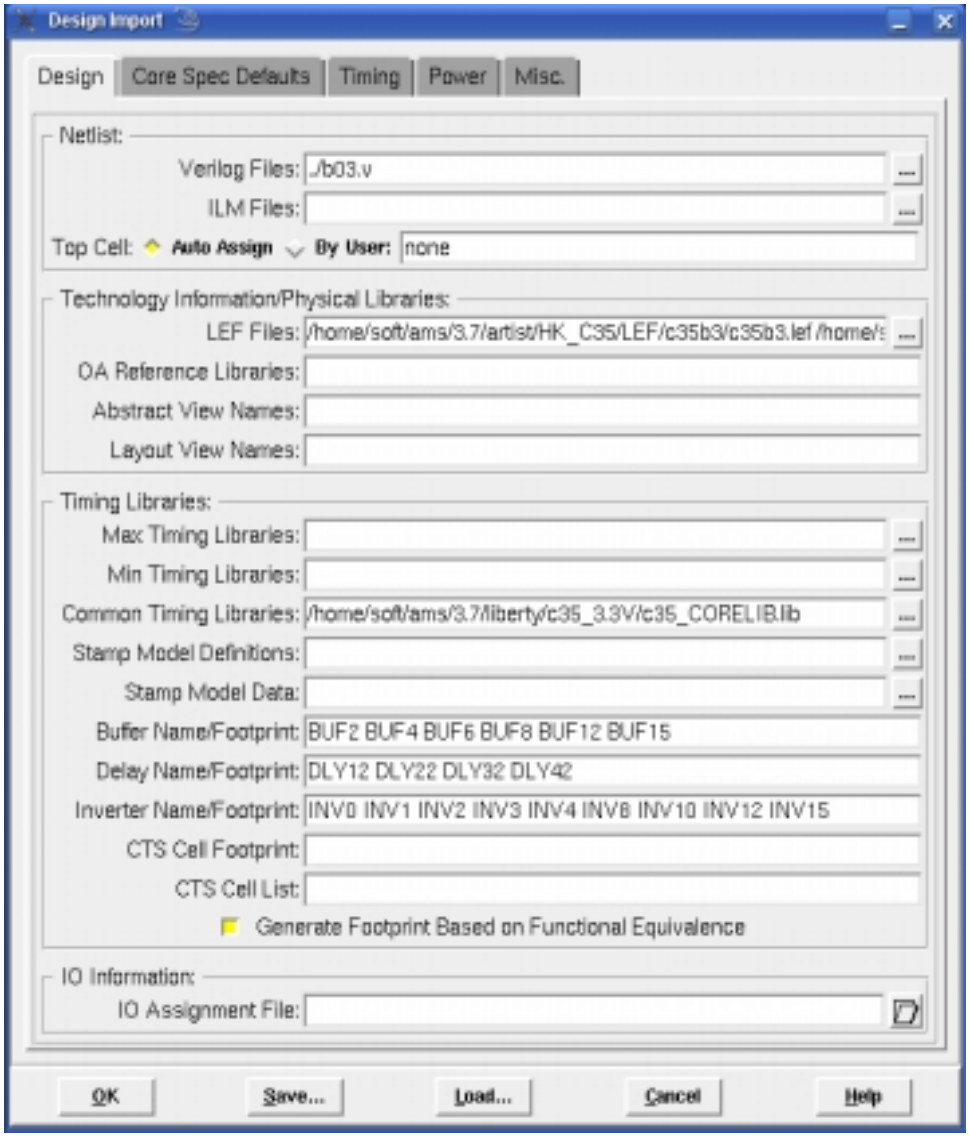

Fig.2 – Import design dialogue box.

To start the design import, select **Design -> Design Import...** in the main menu. Then, click on the **Load...** button and load the file c35b3_std.conf. This file defines a basic import configuration. There are a number of additions and changes required to the initial configuration. The new configuration will then be saved for future uses.

The first information to add is the netlist. Click on the **...** button on the right of the Verilog Files field. You get a new dialog window with only one pane. Click on the top-right icon to get the full window. Remove the VERILOG/none line in the left pane. Select the Verilog netlist file ../b03.v, add it to the left pane and close the window.

In the Design Import window, select the Auto Assign box to let the tool extract the top cell name from the file. If the Verilog file includes more than one design (more than one top module name), you need to give the name of the top module to use explicitly.

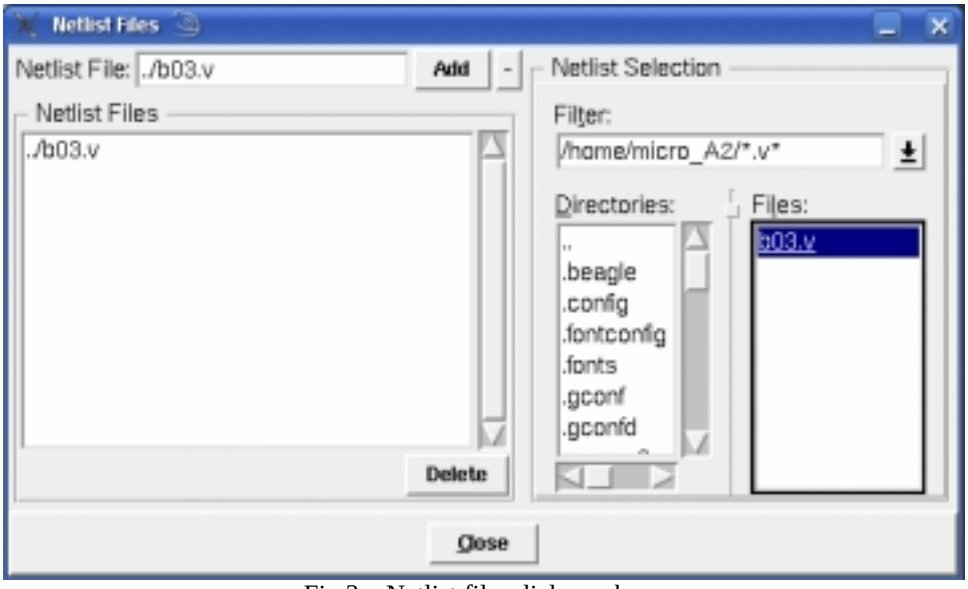

Fig.3 – Netlist files dialogue box.

In the "LEF files" and "Common Timing Libraries" fields, you should remove the reference to the IO pad library since you will not include pads in this design. Remove the paths:
`/soft/ams/v3.7/artist/HK_C35/LEF/c35b3/IOLIB_4M.lef`
`/soft/ams/v3.7/liberty/c35_3.3V/c35_IOLIB_4M.lib`

In the Power tag, you can keep only the vdd! and gnd! power nets if you won't add periphery cells.

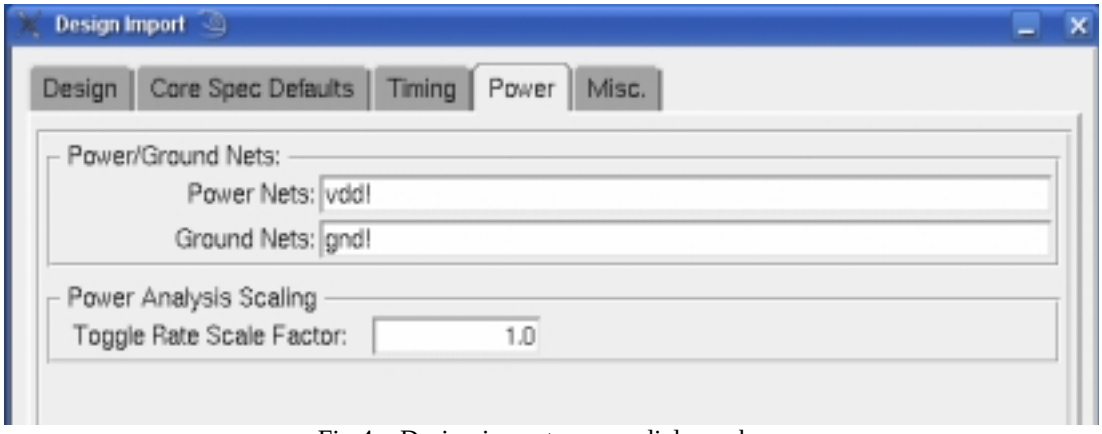

Fig.4 – Design import: power dialogue box.

In the Timing tag, you can specify the timing constraints that you used for synthesis.

Select the file ../b03.sdc (in the Synopsys Design Constraints format):

```
set sdc_version 1.6

set_operating_conditions WORST-IND -library c35_CORELIB
set_max_area 0
create_clock -name clock  -period 10  -waveform {0 5}
```

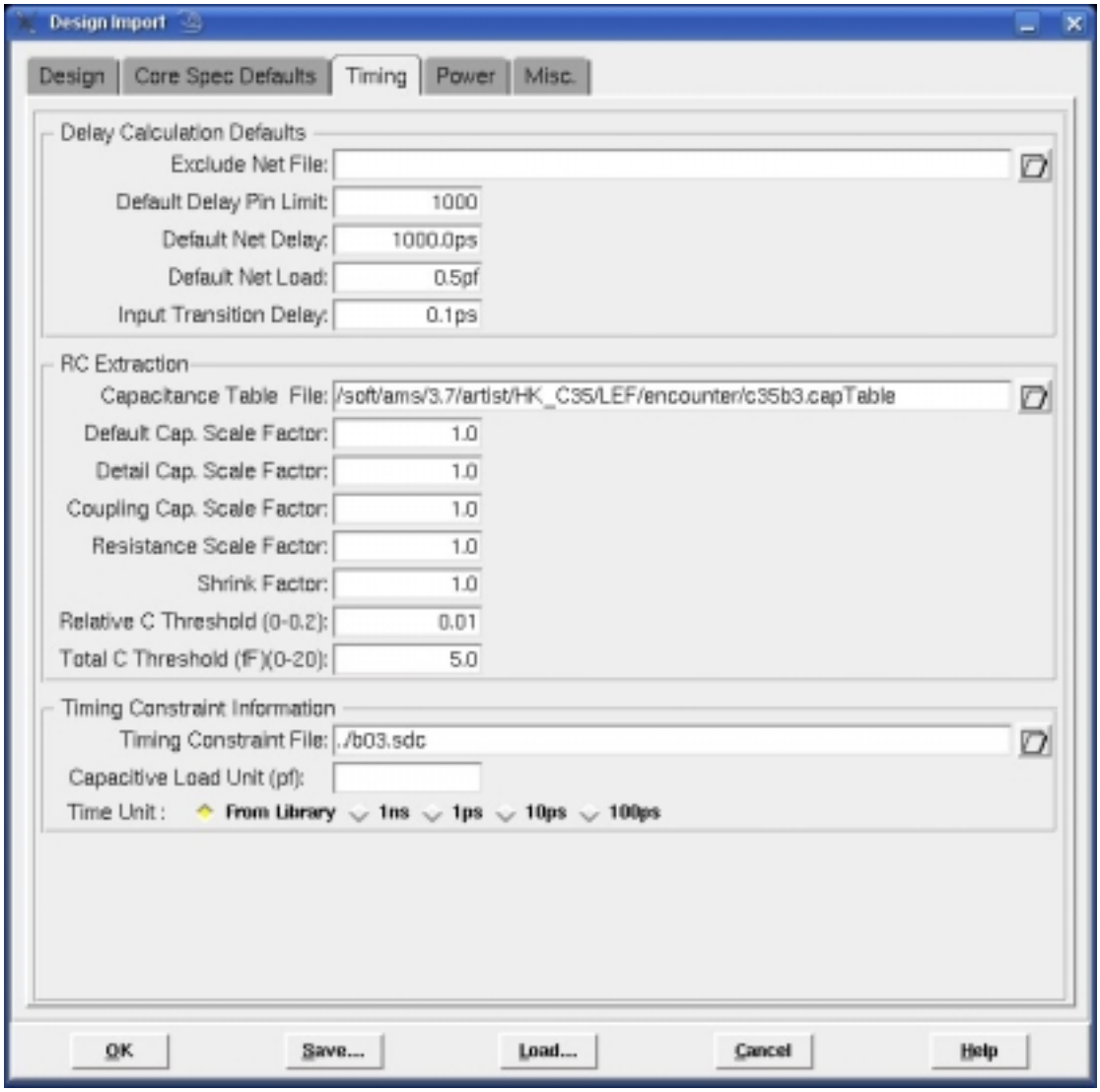

Fig.5 – Design import: timing dialogue box.

Not all the constraints in the file are actually supported in Encounter. In our case, only the clock definition is actually imported.

The rest of the settings can be left as is. You can now save the updated configuration in the file ./b03.conf by clicking on the **Save...** button.

Finally, click on **OK**. The configuration is then read in.

In the case of the AMS design kit, a number of similar errors are reported when the LEF file is loaded. The error message is like:

```
**ERROR: Macro '...' obs coordinate y value... isn't on manufacturing
grid.
It's likely result in placement/routing that can't be manufactured
```

The errors can be safely ignored. The violating coordinates belong to obstructions in the cell LEF file that describe 45 degree shapes. These shapes are not on the manufacturing grid, but this is no problem for the place & route flow. Finally the cells are replaced by the complete layouts anyhow.

To reload a configuration, select **Design -> Design Import...** in the main menu. Then, click on the **Load...** button and load the configuration file ./b03.conf.

The grid should be defined as a multiple of 1.4 micron.
Select **Design -> Preferences…** in the main menu and in the Floorplan tag check the User-defined Grid followed by the Snap All Corners to Grid option. Define the two boxes of the User-defined Grid as 1.4 micron (as showed in figure 6).

Click on **Apply** and **OK**.

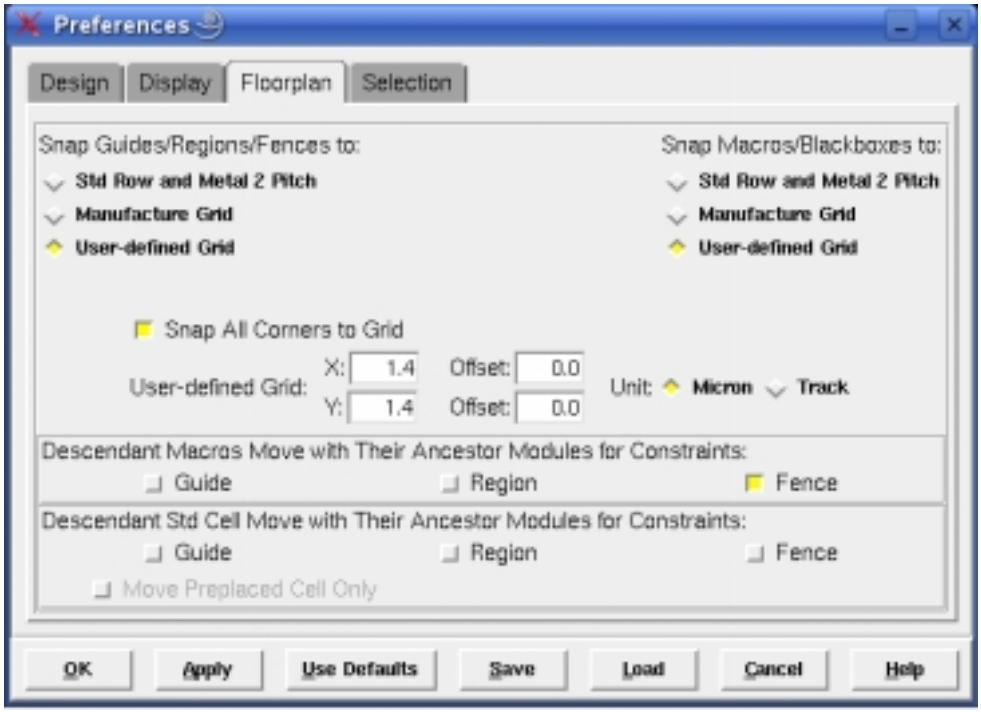

Fig.6 – Preferences dialogue box.

Global net connections

This step assigns pins or nets to global power and ground nets.
Select **Floorplan -> Global Net Connections...** in the main menu.

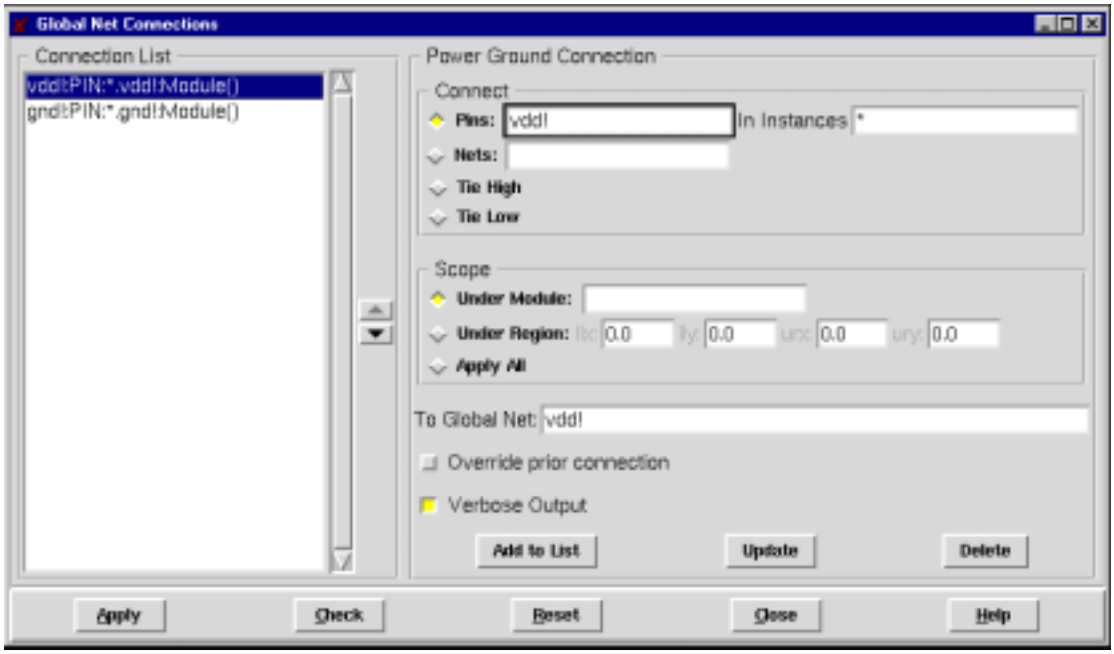

Fig.7 – Global net connections.

The left pane (Connection List) is initially empty. For each VDD and ground net:
• Check the Pins field and enter the pin name (vdd! or gnd!).
• Fill the To Global Net field with either vdd! or gnd!.
• Click on Add to List. The left pane now includes the related global net connection.
Click on **Apply** and **Close**.

Operating conditions definition

The operating conditions define the temperature, process and voltage conditions for the design. They impact the timing calculations and optimizations.

Select the **Timing -> Specify Analysis Conditions -> Specify Operating Condition/PVT...** in the main menu.

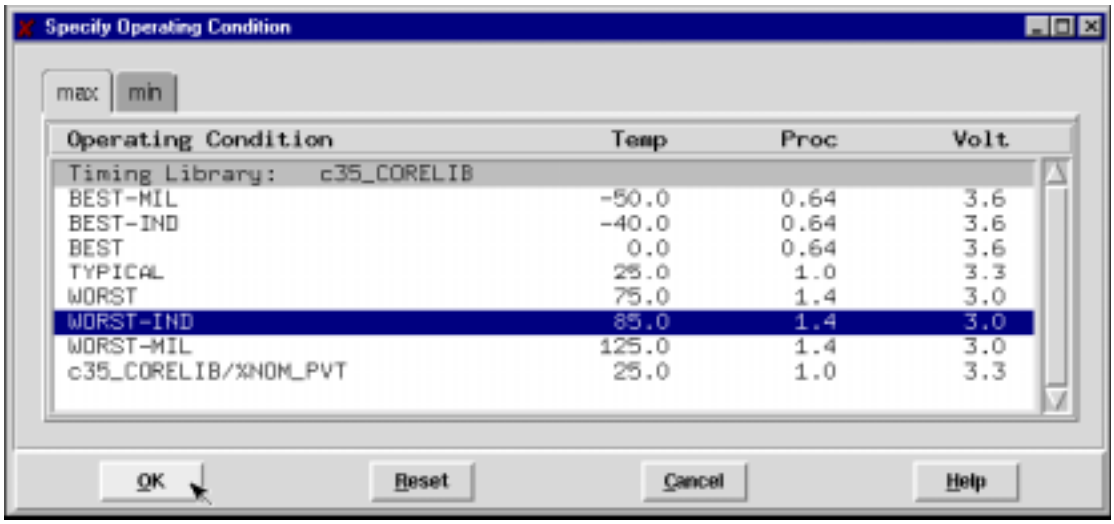

Fig.8

In the max tab, select the WORST-IND operating condition. In the min tab, select the BEST-IND operating condition.

Click **OK**.
The max operating conditions will be used to meet setup timing constraints, while the min operating conditions will be used to meet hold timing constraints.

Floorplan Specification

The floorplan defines the actual form, or aspect ratio, the layout will take, the global and detailed routing grids, the rows to host the core cells and the I/O pad cells (if required), and the location of the corner cells (if required).

Select **Floorplan -> Specify Floorplan...** in the main menu.

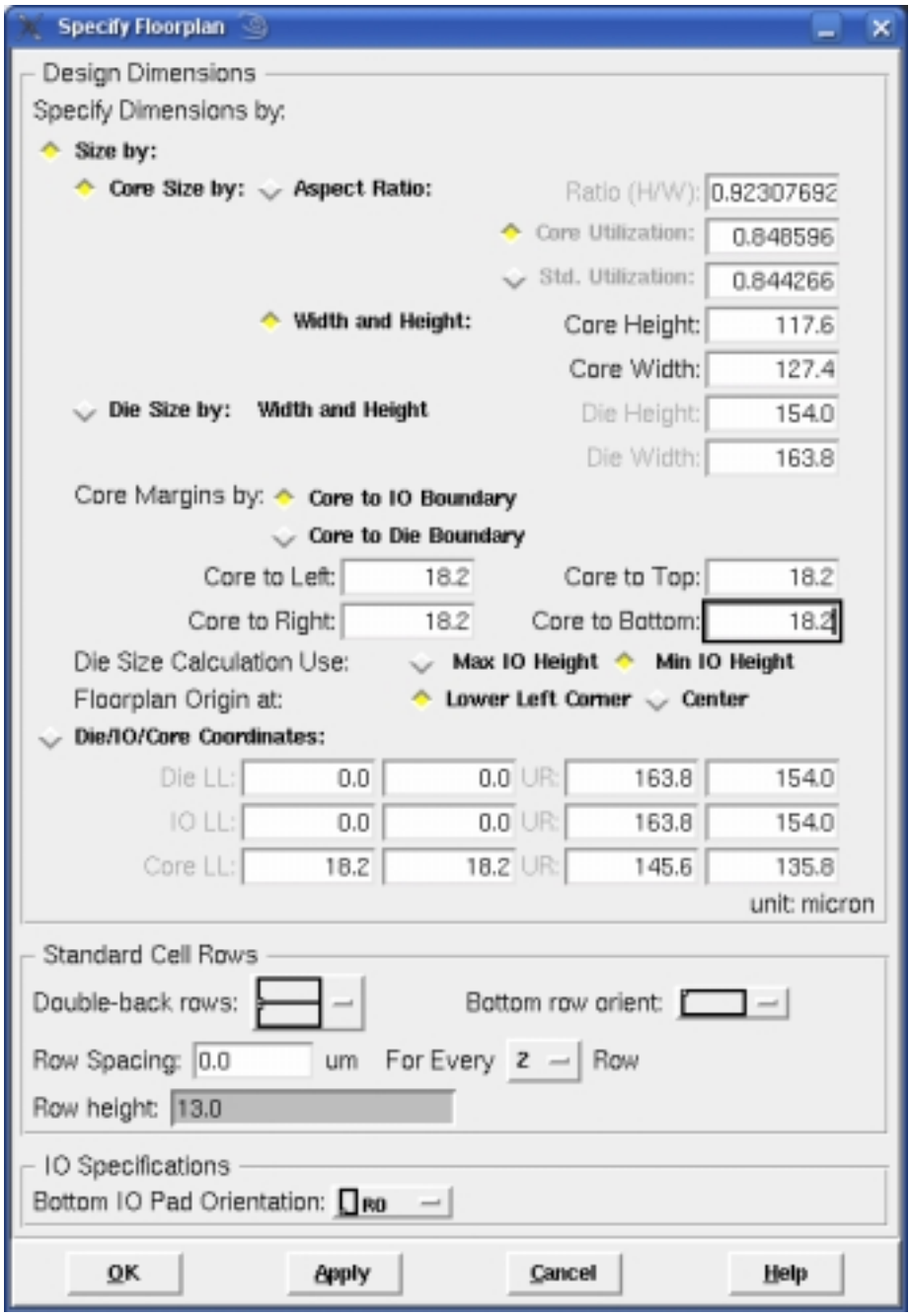

Fig.9

Define the core width and height with the values showed in figure 9 (check the Width and Height box).

Alternatively the core size can be defined as an aspect ratio of 1.
Setting the core utilization to 85%, means that 15% of the core area will be free for possible future buffer insertions or cell replacements. The other utilization ratio (Std. Utilization) refers to the density of the standard cells.

Define I/O to core distances of 18.2 micron. This is enough since there won't be any periphery cells (I/O pads) in the layout. The cell rows will be abutted and flipped each two row to share VDD and ground power rails.

Click **OK**.

The display design area pane now shows the defined floorplan with the required number of rows.

It is a good idea to save the design at that stage to allow restarting here quickly without needing to redo all the previous steps. Select **Design -> Save Design...** in the main menu and save the current state in the file cds_encouter_tutorial/b03.enc. The data is actually saved in the directory cds_encounter_tutorial/b03.enc.dat.

To restore design data, select **Design -> Restore Design...** in the main menu and select the .enc file to read in the cds_encounter_tutorial directory.

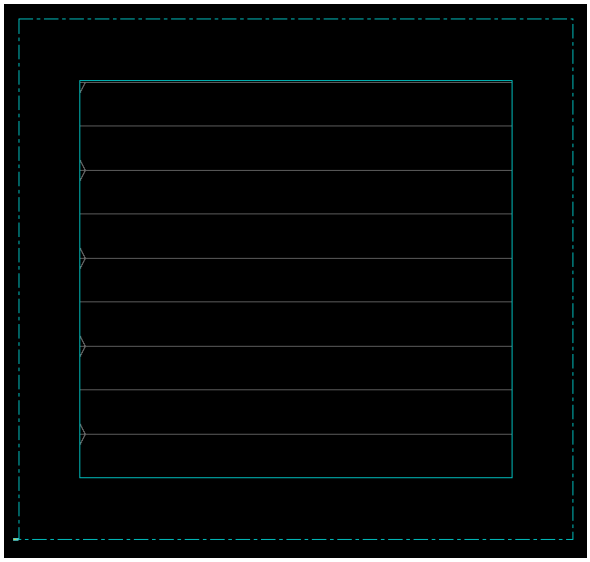

Fig.10

CAP cell placement

The AMS design kit requires that physical-only termination cells must be placed at each row ends. The so called CAP cells are used to properly bias the P+ and N+ substrates.

Select **Place -> Filler -> Add End Cap...** in the main menu. Then, select the ENDACPL cell as the cell to place at the beginning of each row and the ENDCAPR cell as the cell to place at the end of each row.

Click **OK**. The added cells are now visible in the Physical view (type <f> to redraw and fit the design in the window).

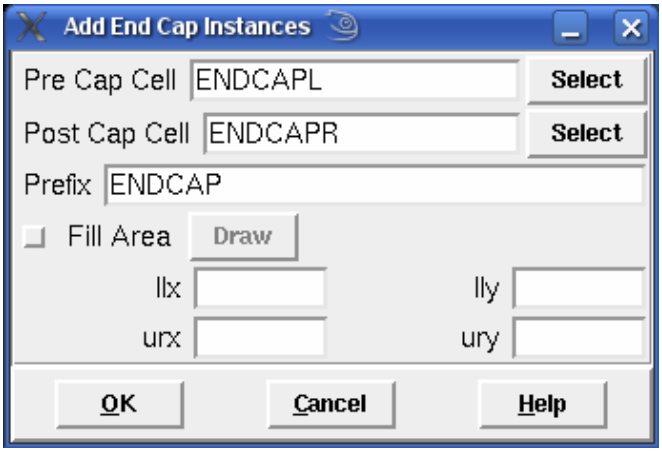

Fig.11

Power ring/stripe creation and routing

This step generates the VDD and ground power rings around the core and optionally adds a number of vertical and/or horizontal power stripes across the core. Stripes ensure a proper power distribution in large cores.

Select **Floorplan -> Power Planning -> Add Rings...** in the main menu.

The Net(s) field defines the number and the kinds of rings from the core. In our case, there will be first a ground ring around the core and a VDD ring around the ground ring. The net names should be consistent with the power net names in the cell LEF file.

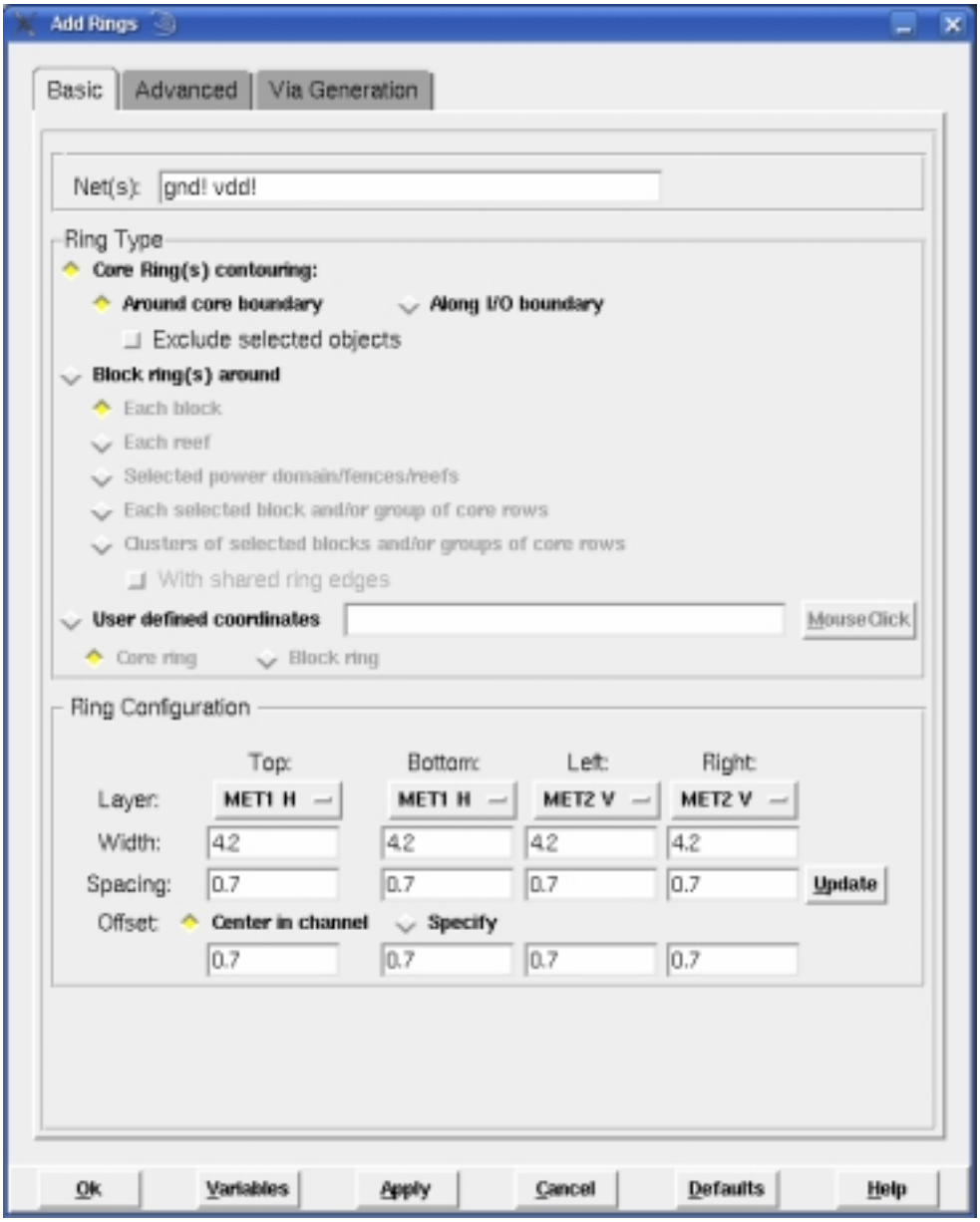

Fig.12

The ring configuration defines ring widths of 4.2 micron spaced by 0.7 micron. The rings will be placed in the centre of the channel between the core and the chip boundary (or the IO pads, if any).

It is possible to extend the ring segments to reach the core boundary. Click on the Advanced tab and click on the segments you'd like to extend (see Fig.13). Other power and ground side trunks can be defined by selecting only horizontal or vertical segments.

Click **OK** to generate the rings.

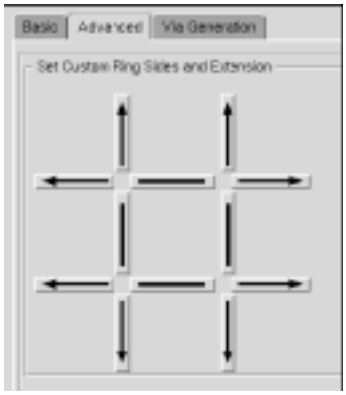

Fig.13

To add power stripes, select **Floorplan -> Power Planning -> Add Stripes...** in the main menu.

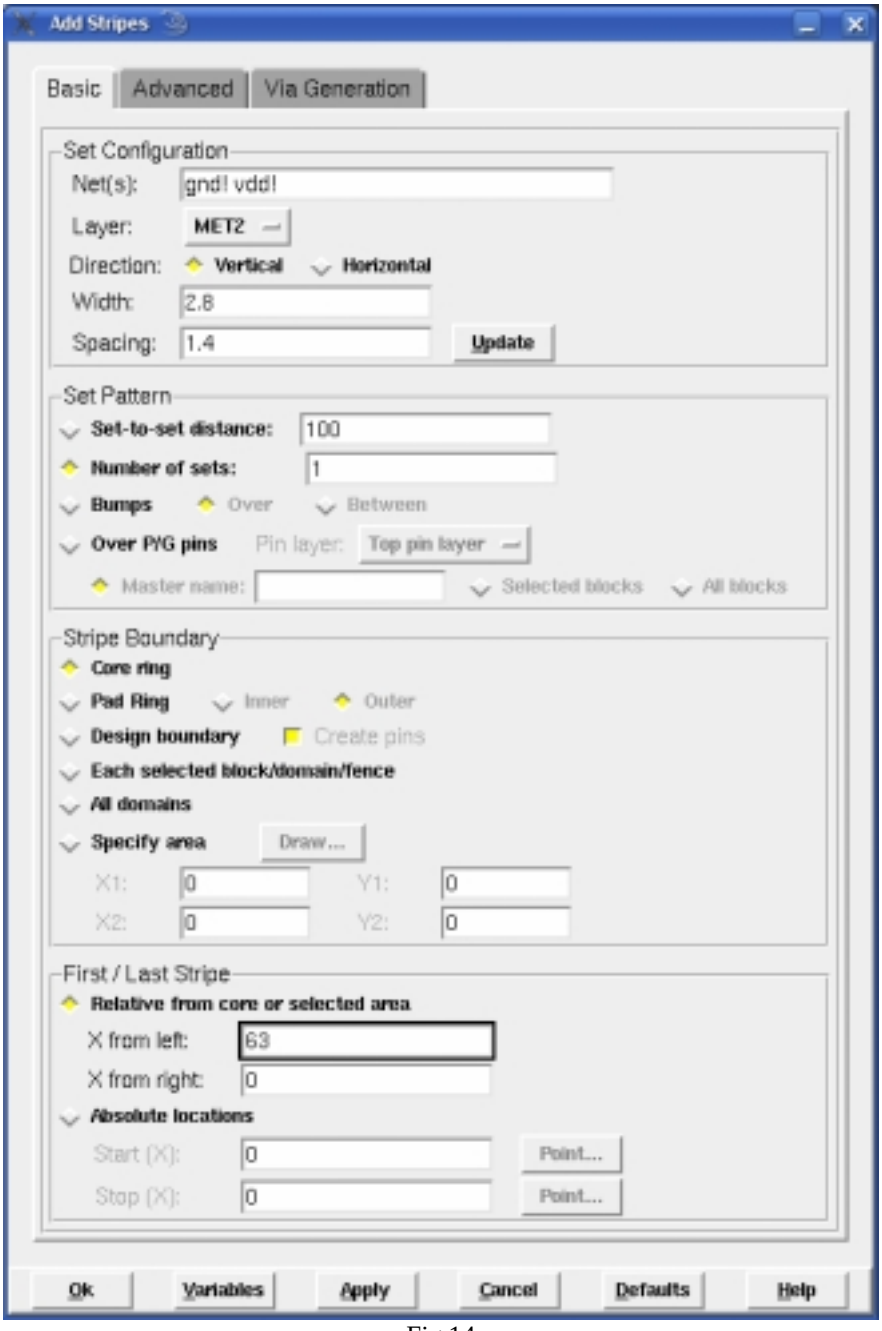

Fig.14

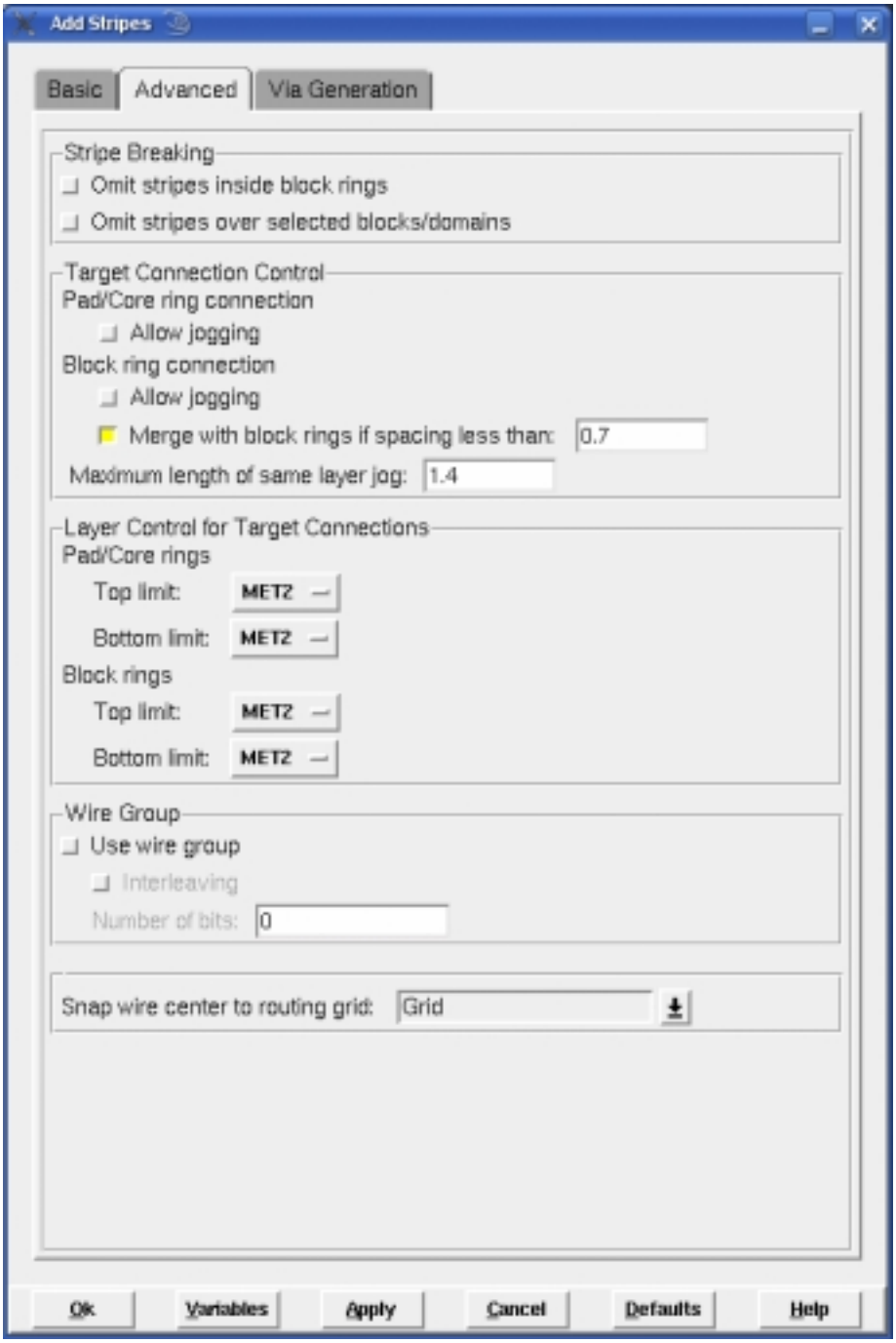

Fig.15

The Net(s) field defines the pattern. Here a single pattern will be generated (select Number of sets and insert 1). Each stripe will be 2.8 micron wide and the space between them will be 1.4 micron.

Check Relative from core or selected area and enter the value 63 (micron) in the X from left field. The two stripes will be vertically placed near the centre of the core. It is possible to measure sizes by using the ruler (press ESC to remove all rulers).

Click on the Advanced tab and on the Maximum length of same layer jog insert 1.4. On bottom of the dialogue box, on Snap wire center to routing grid, select Grid.

Click on **Apply** and **OK** to generate the stripes.

Now, it is possible to route the power grid. Select **Route -> SRoute...** in the main menu. All default values are fine. Click **OK** to do the routing. The design now looks like below:

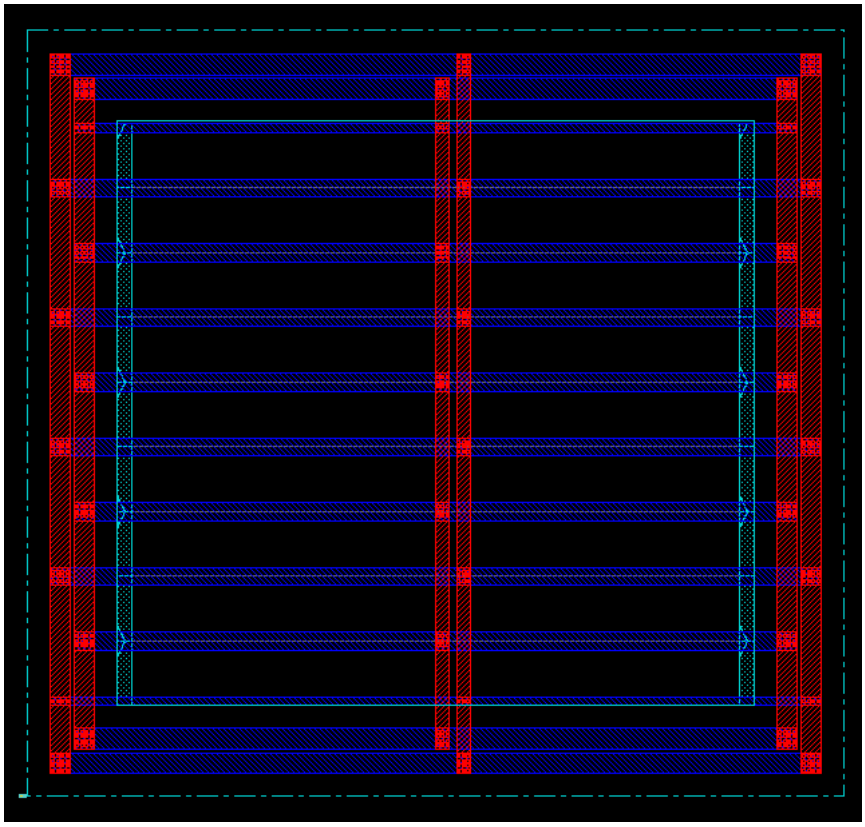

Fig.16

It is recommended to save the new stage of the design. Select **Design -> Save Design...** in the main menu and save the current state in the file cds_encounter_tutorial/b03.enc.

Core cell placement

This steps places the cells of the imported Verilog netlist in the rows.

Select **Place -> Place...** in the main menu.

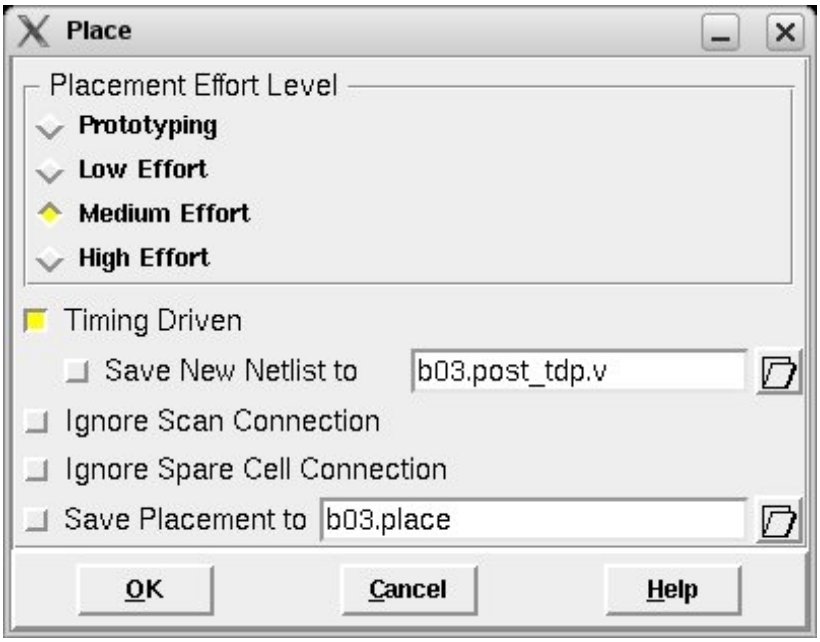

Fig.17

Select the Timing Driven option. This is only valid if timing constraints have been properly defined on the design. This option will optimize the placement of the cells that are on the critical path.
Click **OK** to do the placement. It may take some time to complete, especially when the placement is timing driven.

Clock tree synthesis (optional)

As the paths that will propagate the clock signal in the design are not necessarily balanced, some registers may receive the active clock edge later than others (clock skew) and may therefore violate the assumed synchronous design operation. For example, the original clock tree we can get from the previously placed design is:

```
# FirstEncounter(TM) Clock Synthesis Technology File Format
# for automatic gated clock tree synthesis
#   AutoCTSRootPin    <clock root pin name>
#   MaxDelay     <number>
#   MinDelay     <number>
#   SinkMaxTran          <number>
#   BufMaxTran   <number>
#   MaxSkew           <number>
#   NoGating          <rising|falling|NO>
#   MaxDepth     <number>
#   Buffer           <cell_1> <cell_2> <cell_3> ....
#   LeafPin
#   +  <pin name> <rising|falling>
#   +  ....
#   LeafPort
#   +  <port name> <rising|falling>
#   +  ....
#   ExcludedPin
#   +  <pin name>
#   +  ....
#   ExcludedPort
#   +  <port name>
#   +  ....
#   End
#
AutoCTSRootPin    clock
NoGating          rising
MaxDelay          3ns
MinDelay          0ns
MaxSkew           150ps
SinkMaxTran       400ps
BufMaxTran        400ps
Buffer            BUF2 INV0
End
```

You will see the following data in the b03.cts.auto file:

- The clock root name is clock.
- The max and min insertion delays are specified as 3 and 0 ns.
- The transition on clock nets is specified as 400 ps, while the clock skew is 150 ps.

The NoGating option instructs the clock tree generation tool to stop at nonbuffer/noninverter pins and to use the specified edge as the trigger edge.
If you have more internal clocks, add their information to the specification file.

In order to load the clock tree specification file select **Clock -> Specify Clock Tree...** in the main menu and select the file ../b03.cts.auto that has been previously created.

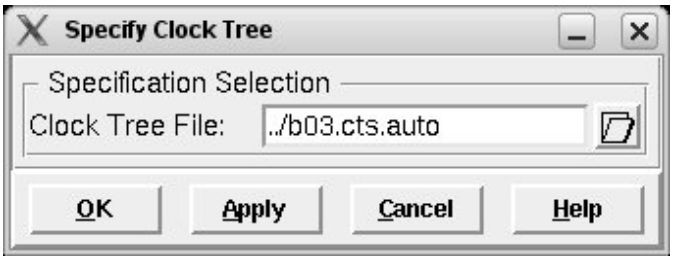

Fig.18

To create the clock tree, select **Clock -> Synthesize Clock Tree ...** in the main menu.
Define the result directory as b03_cts and the base file name as b03_cts.
Click **OK** to create the clock tree.

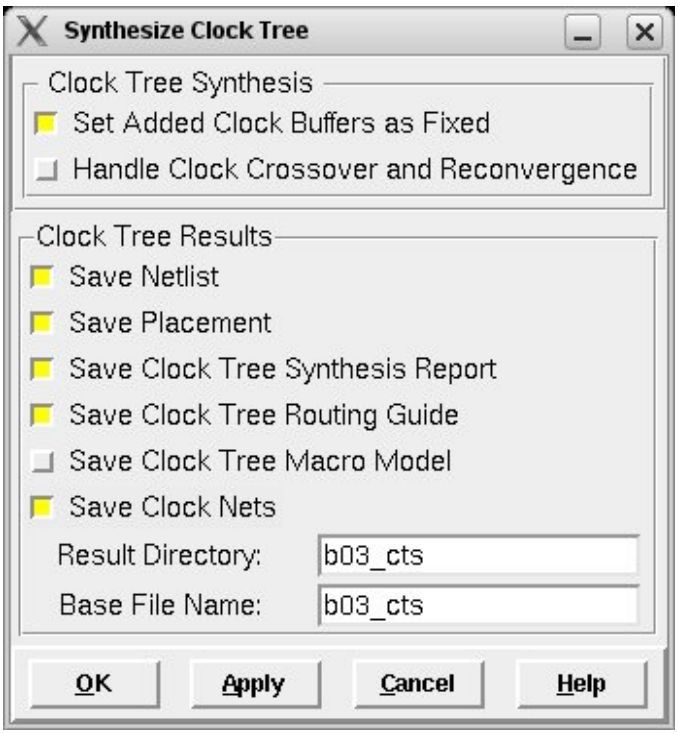

Fig.19

Now it is possible to run timing optimization, based on the generated clock tree. Select **Timing -> Optimization...** in the main menu and check postCTS. Click **OK** to do the optimization.

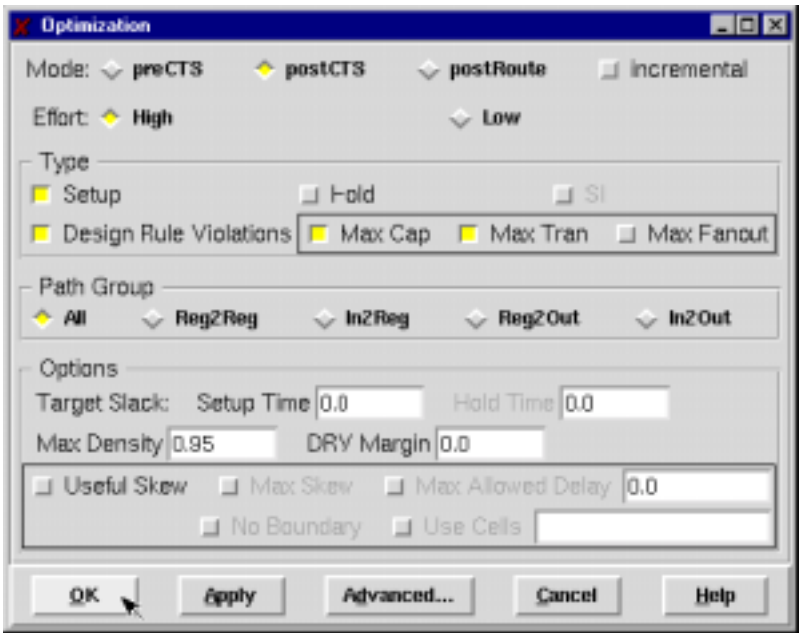

Fig.20

The console will display the results of the timing optimization:

```
------------------------------------------------------------
                    optDesign
------------------------------------------------------------
Setup mode
  Worst Slack: 0.000ns
  TNS: 0.000ns  Violating Paths: 0
  Pathgroup Slacks
    reg2reg: 0.000ns
    in2reg: 0.000ns
    reg2out: 0.000ns
    in2out: 0.000ns
Density: 96.954%
Routing Overflow: 0.00% H and 2.39% V
Real DRV (fanout, cap, tran): (0, 0, 0)
Total DRV (fanout, cap, tran): (0, 0, 0)
------------------------------------------------------------
**optDesign ... cpu = 0:0:0, real = 0:0:1, mem = 97.6M **
*** Finished optDesign ***
```

Design routing

This step generates all the wires required to connect the cells according to the imported gate-level netlist.

To route the design, select **Route -> Nanoroute...** in the main menu and check the Insert Diodes box. On the Diode Cell Name insert FILLANT1, FILLANT2, FILLANT5 and FILLANT10.
Click **OK** to do the routing.

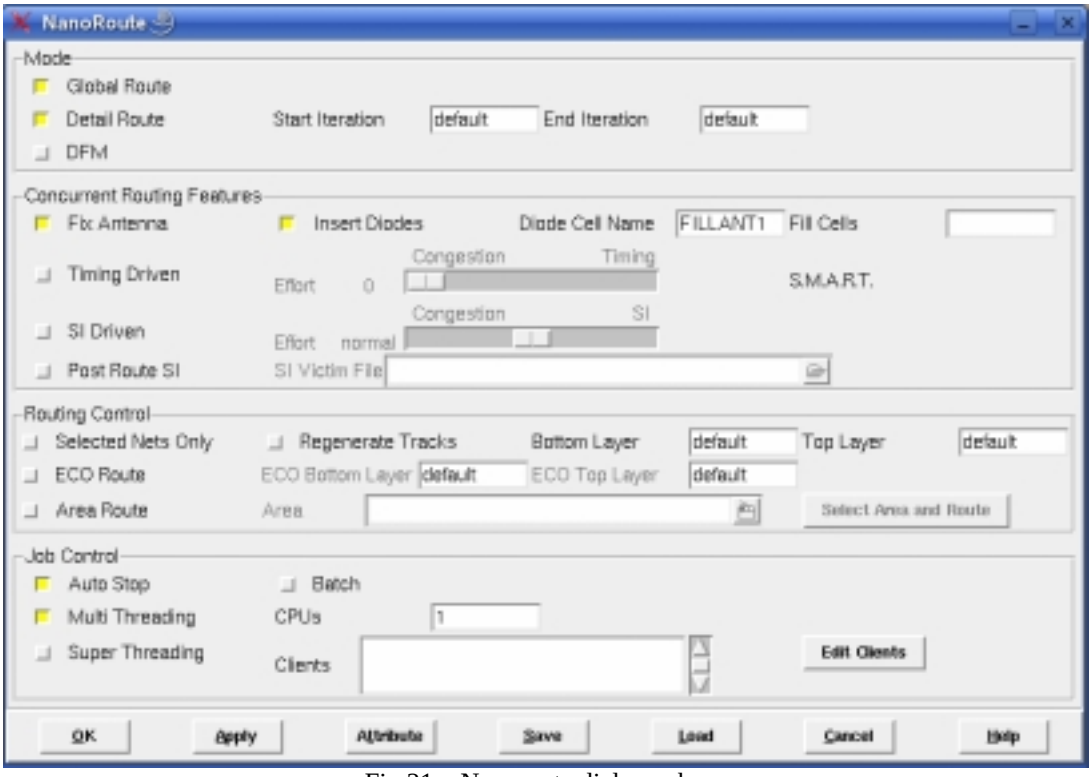

Fig.21 – Nanoroute dialogue box.

Filler cells

Filler cells will fill remaining holes in the rows and ensure the continuity of power/ground rails and N+/P+ wells in the row.

To fill the holes with filler cells, select **Place -> Filler -> Add Filler...** in the main menu. Select the cells FILL1, FILL2, FILL5, FILL10, FILL25 and FILLRT1, FILLRT2, FILLRT5, FILLRT10, FILLRT25 and click **OK** to place the filler cells.

Do not add filler cells now if you intend to synthesize a clock tree. Clock tree synthesis may need some free space to insert clock buffers to balance the clock paths.

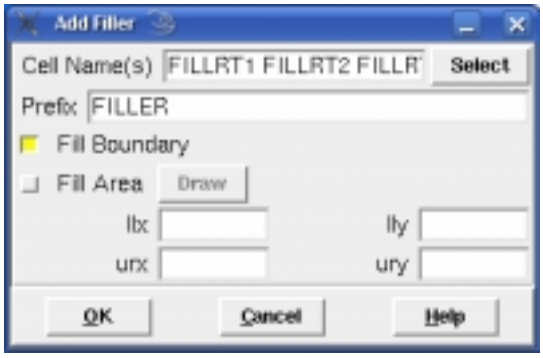

Fig.22

The placement should then look like below (redraw <f> the image if it was not updated):

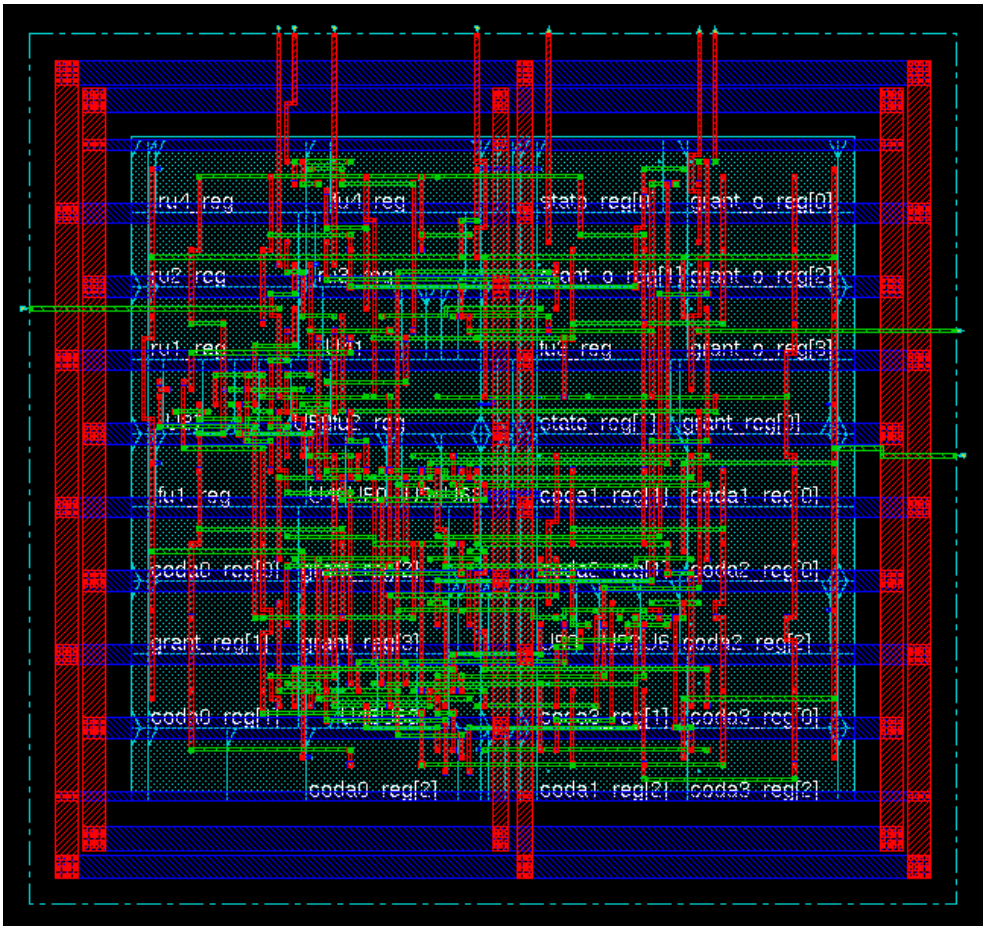

Fig.23

Timing analysis

Select **Timing -> Timing Analysis...** in the main menu. Define the path for the slack report file. Click **OK**.

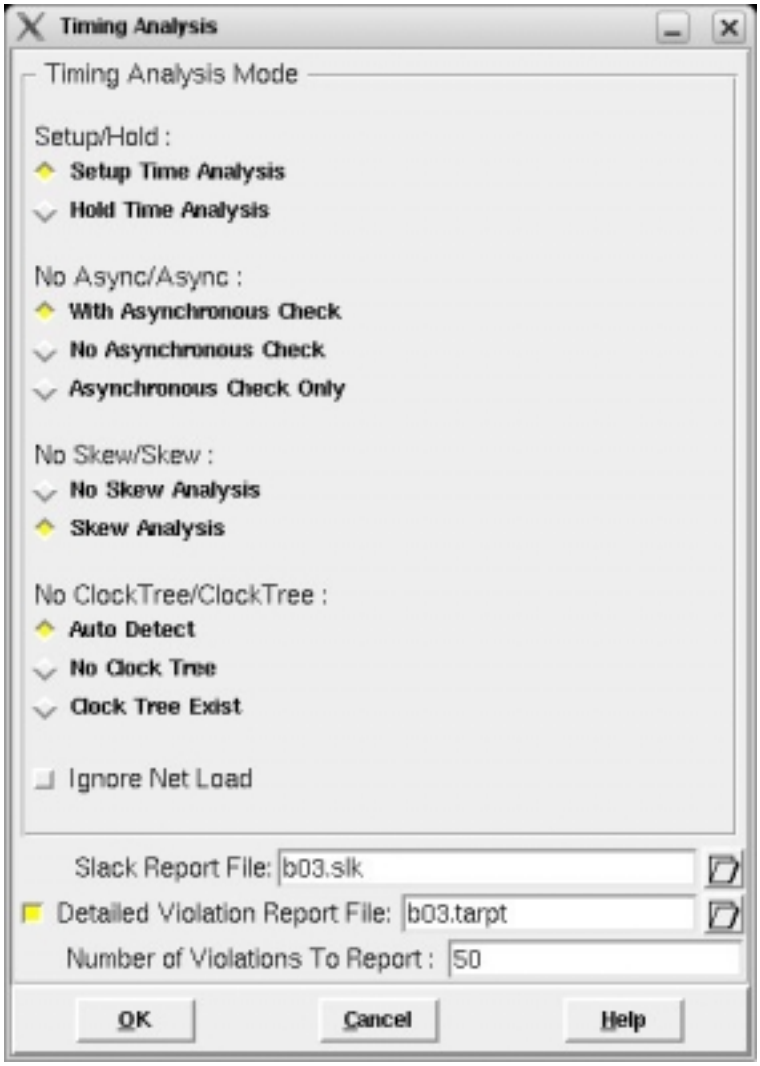

Fig.24

In the console window you get a summary of the timing analysis and the slack report file includes more detailed information.

Right after the placement it is also possible to get a first report on the design timing with the same procedure.

Design checks

The Verify menu has a number of items to check that the design has been properly placed and routed. Select **Verify -> Verify Connectivity...** in the main menu.

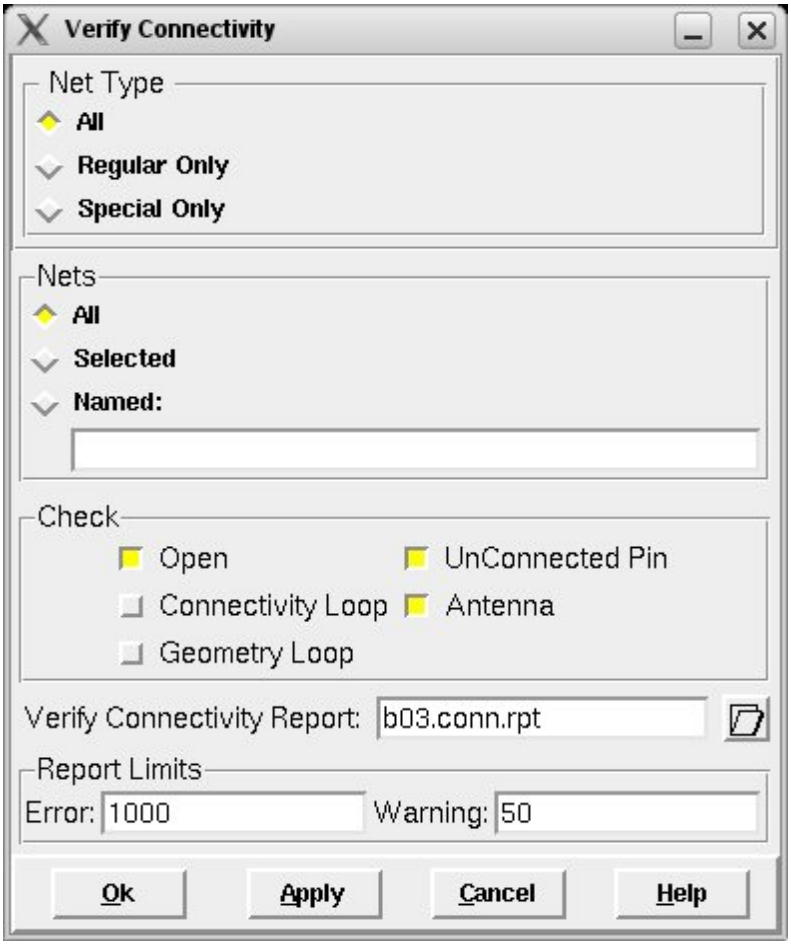

Fig. 25

Define the report file as ./RPT/b03.conn.rpt. Click **OK**. The console displays the results. Confirm that there are no violations or warnings.

Select **Verify -> Verify Geometry...** in the main menu.

Fig. 26

In the Advanced tab, define the report file as b03.rpt. Click **OK**. The console displays the results.

Report generation

A number of reports have been already generated in the previous steps. They should be located in the working directory. The **Tools** menu includes some additional reports, try:
Tools -> Netlist Stats…
Tools -> Gate Count Report...
Tools -> Summary Report...
in order to obtain different types of statistical information.

Post-route timing data extraction

This step generates the post-route SDF (Standard Delay Format) file that includes both the actual interconnect and cell timing delays.

The parasitics must be first extracted. Select **Timing -> Extract RC...** in the main menu.

The generated cap file includes the wired capacitance, pin capacitance, total capacitance, net length, wire cap per unit length and the fanout of each net in the design.

The generated SPEF (Standard Parasitics Exchange Format) file includes RC values in a SPICE-like format.

The SDF file may be then generated by selecting **Timing -> Calculate Delay...** in the main menu.

The checked Ideal Clock switch means that flip-flops are considered as having 0ps rising and falling transition times.

Post-route netlist generation

This step generates a Verilog netlist of the routed design. The netlist may be different from the imported netlist as cells may have been added or replaced during clock tree synthesis and timing-driven optimizations.

Select **Design -> Save -> Netlist...** in the main menu.

The generated file should be saved into the HDL/GATE directory.

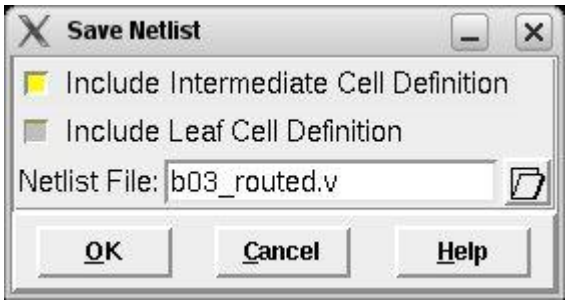

Fig.27

GDS2 file generation

The placed and routed design can be exported in different formats for further processing outside the Encounter tool. The GDS2 binary format is a standard format for integrating the block in the top-level layout, doing DRC/LVS checking, or delivering the layout to the foundry.

To export the design in the GDS2 format, select **Design -> Save -> GDS...** in the main menu.

The GDS map file has been copied by the AMS setup script in the DEX directory. The generated GDS2 file is written in the same directory.

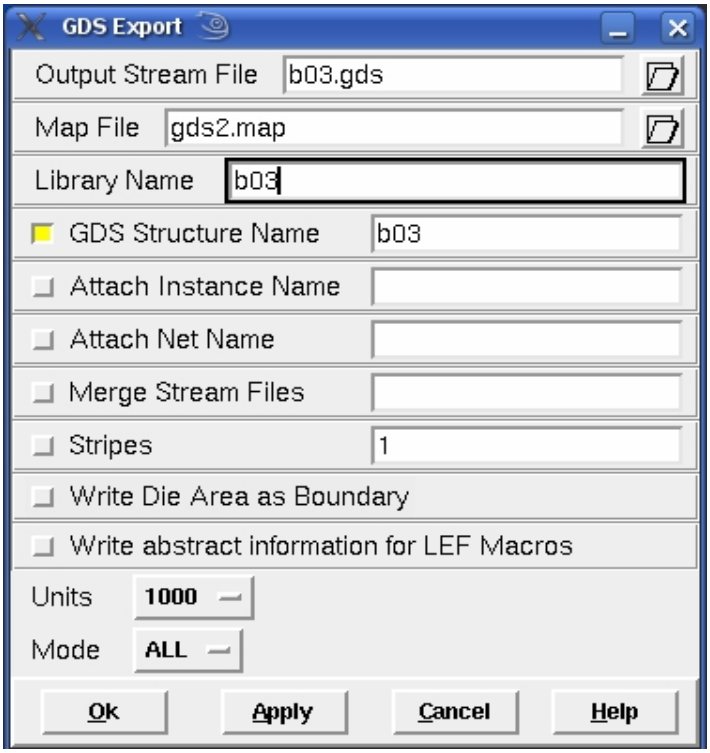

Fig.28

Design import in Virtuoso

The GDS2 file can be imported in Virtuoso for further processing. To do that, it is recommended to start the Cadence IC environment in the LAY directory.

Start Cadence IC:

```
ams_cds -t c35b3 -m fb
```

Create a new library called b03 with the attached technology file TECH_C35B3.

In the DFII shell, select **File -> Import -> Stream...**.

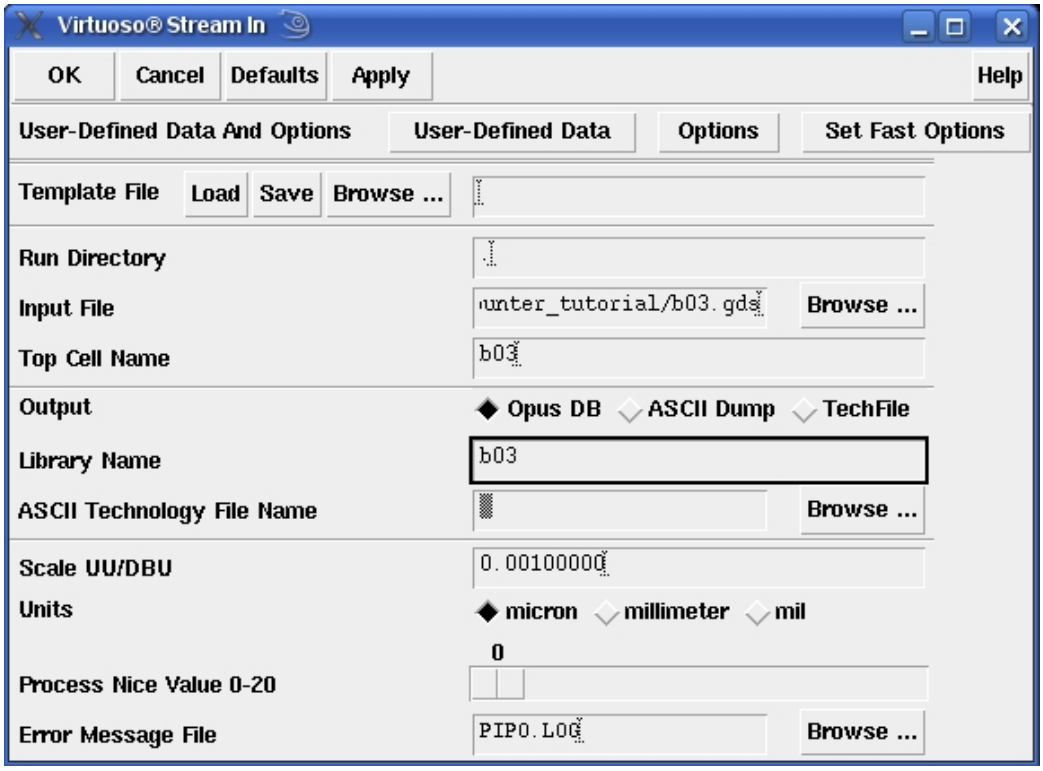

Fig.29

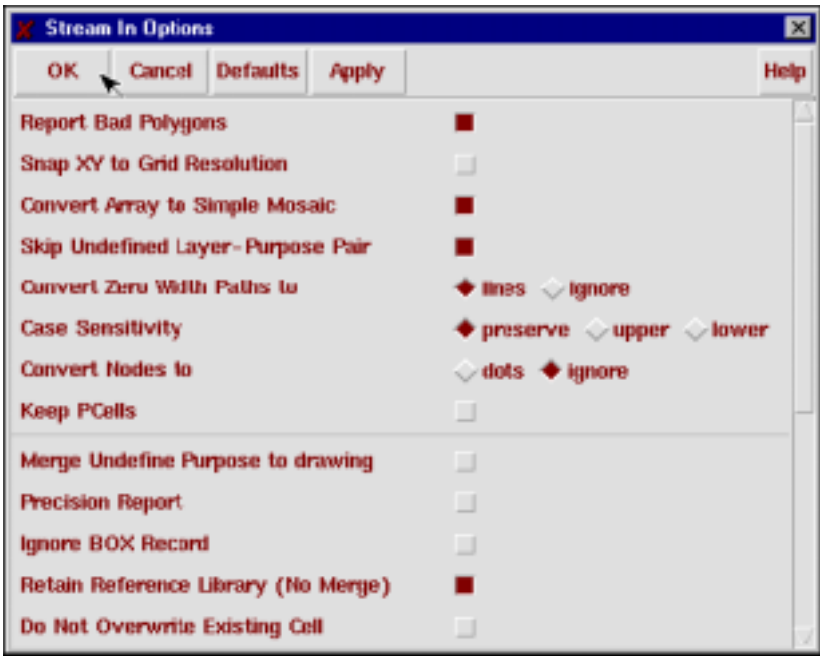

Fig.30

In the Options tab, check the Retain Reference Library (No Merge) option. Since the cell layouts are available in the CORELIB library, it is possible to display the full layout of the block.

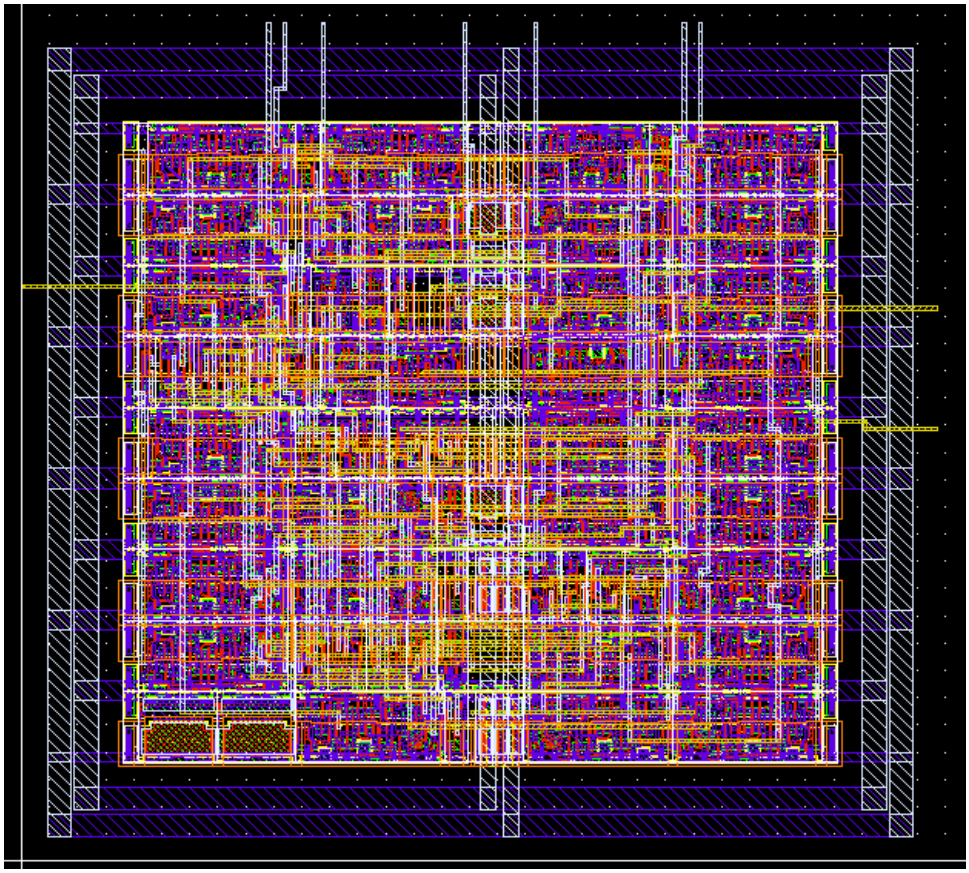

Fig.31 – b03 layout (in Virtuoso console).

Using scripts (optional)

It is much more convenient to capture the placement and routing flow in a script. Cadence Encounter also supports the Tcl language for building scripts.

An example of such a script for placement and routing is presented next.

To run a Tcl script, execute the following command in a Unix shell:

```
>encounter script_name.tcl
```

The example given below may be modified to define design information and to control the flow to some extent.

```
#
# Cadence Encounter Tcl script for adder-subtractor.
#
# Process: AMS 0.35u CMOS (C35), Hit-Kit 3.70
#
#-------------------------------------------------------------------------
# It is assumed that a project directory structure has already been
# created using 'create_project' and that this synthesis script is
# executed from the project root directory $PROJECT_DIR
#-------------------------------------------------------------------------
set PROJECT_DIR [pwd]
#-------------------------------------------------------------------------
# Design related information (can be changed)
#-------------------------------------------------------------------------
set DESIGN addsub_nbits8
```

```
set TIM_LIBRARY C35_CORELIB
set TIM_OC_MAX WORST ;# TYPICAL | WORST | WORST-IND
set TIM_OC_MIN BEST ;# TYPICAL | BEST | BEST-IND
# Floorplan settings
#
set FP_ASPECT_RATIO 1
set FP_ROW_DENSITY 0.85 ;# percent
set FP_CORE2IO 16 ;# micron
# Power ring and settings
#
set PR_WIDTH 4 ;# micron
set PR_SPACING 1 ;# micron
set PR_LAYER_TB MET1 ;# top and bottom layer
set PR_LAYER_LR MET2 ;# left and right layer
# Power stripe settings
#
set ST_NUM_SETS 1 ;# number of sets
set ST_SPACING 1 ;# micron
set ST_LAYER_V $PR_LAYER_LR
set ST_WIDTH 2 ;# micron
set ST_XOFS_R 60 ;# micron
set ST_XOFS_L 60 ;# micron
# Placement settings
#
set PL_EFFORT -high ;# -low | -medium | -high
# Clock tree synthesis settings
#
set CTS_BUFFER BUF2
set CTS_INV INV0
#--------------------------------------------------------------------------
# Flags that drive the script behavior (can be changed)
#
# ADD_STRIPES (0 | 1)
# if 1, add stripes
# PLACE_TIMING (0 | 1)
# if 1, do a timing driven placement
# CLOCK_TREE (0 | 1)
# if 1, create a clock tree
# CTS_CREATE_SPEC (0 | 1)
# if 1, create a clock tree specification file with default values
# ROUTE_TIMING (0 | 1)
# if 1, do a timing driven routing
# OPT (string)
# can be used to have different generated file names
#--------------------------------------------------------------------------
set ADD_STRIPES 1
set PLACE_TIMING 1
set CLOCK_TREE 1
set CTS_CREATE_SPEC 0
set ROUTE_TIMING 1
set OPT "_cts"
#--------------------------------------------------------------------------
# File names
#--------------------------------------------------------------------------
set CONF_FILE_NAME ${DESIGN}.conf
set DESIGN_NAME ${DESIGN}${OPT}
set SAVE_DESIGN_FP_NAME ${DESIGN_NAME}-fplan.enc
set SAVE_DESIGN_PR_NAME ${DESIGN_NAME}-pring.enc
set SAVE_DESIGN_PL_NAME ${DESIGN_NAME}-placed.enc
set SAVE_DESIGN_PF_NAME ${DESIGN_NAME}-placed_filled.enc
set SAVE_DESIGN_CT_NAME ${DESIGN_NAME}-cts.enc
set SAVE_DESIGN_RO_NAME ${DESIGN_NAME}-routed.enc
set TIM_RCDB_NAME ${DESIGN_NAME}.rcdb
set SDF_FILE_NAME ${DESIGN_NAME}-routed.sdf
set SPEF_FILE_NAME ${DESIGN_NAME}-routed.spef
set RPT_CHECK_TA_NAME ${DESIGN_NAME}-checkta.rpt
set RPT_REPORT_TA_NAME ${DESIGN_NAME}-ta.rpt
set RPT_SLACK_NAME ${DESIGN_NAME}-slack.rpt
```

```
set RPT_GATE_COUNT_NAME ${DESIGN_NAME}-gate_count.rpt
set RPT_NOTCH_NAME ${DESIGN_NAME}-notch.rpt
set RPT_CONN_NAME ${DESIGN_NAME}-conn.rpt
set RPT_GEOM_NAME ${DESIGN_NAME}-geom.rpt
set RPT_DENSITY_NAME ${DESIGN_NAME}-density.rpt
set VLOG_NETLIST_SIM_NAME ${DESIGN_NAME}-routed.v
set VLOG_NETLIST_LVS_NAME ${DESIGN_NAME}-routed_lvs.v
set CTS_SPEC_NAME ${DESIGN_NAME}-spec.cts
set CTS_RGUIDE_NAME ${DESIGN_NAME}-guide.cts
set CTS_RPT_NAME ${DESIGN_NAME}-cts.rpt
set GDS_FILE_NAME ${DESIGN_NAME}.gds
#-------------------------------------------------------------------------
# Absolute paths
#-------------------------------------------------------------------------
set CONF_FILE ${PROJECT_DIR}/PAR/CONF/${CONF_FILE_NAME}
set SAVE_DESIGN_FP_FILE ${PROJECT_DIR}/PAR/DB/${SAVE_DESIGN_FP_NAME}
set SAVE_DESIGN_PR_FILE ${PROJECT_DIR}/PAR/DB/${SAVE_DESIGN_PR_NAME}
set SAVE_DESIGN_PL_FILE ${PROJECT_DIR}/PAR/DB/${SAVE_DESIGN_PL_NAME}
set SAVE_DESIGN_PF_FILE ${PROJECT_DIR}/PAR/DB/${SAVE_DESIGN_PF_NAME}
set SAVE_DESIGN_CT_FILE ${PROJECT_DIR}/PAR/DB/${SAVE_DESIGN_CT_NAME}
set SAVE_DESIGN_RO_FILE ${PROJECT_DIR}/PAR/DB/${SAVE_DESIGN_RO_NAME}
set SDF_FILE ${PROJECT_DIR}/PAR/TIM/${SDF_FILE_NAME}
set SPEF_FILE ${PROJECT_DIR}/PAR/TIM/${SPEF_FILE_NAME}
set TIM_RCDB_FILE ${PROJECT_DIR}/PAR/TIM/${TIM_RCDB_NAME}
set RPT_CHECK_TA_FILE ${PROJECT_DIR}/PAR/RPT/${RPT_CHECK_TA_NAME}
set RPT_REPORT_TA_FILE ${PROJECT_DIR}/PAR/RPT/${RPT_REPORT_TA_NAME}
set RPT_SLACK_FILE ${PROJECT_DIR}/PAR/RPT/${RPT_SLACK_NAME}
set RPT_GATE_COUNT_FILE ${PROJECT_DIR}/PAR/RPT/${RPT_GATE_COUNT_NAME}
set RPT_NOTCH_FILE ${PROJECT_DIR}/PAR/RPT/${RPT_NOTCH_NAME}
set RPT_CONN_FILE ${PROJECT_DIR}/PAR/RPT/${RPT_CONN_NAME}
set RPT_GEOM_FILE ${PROJECT_DIR}/PAR/RPT/${RPT_GEOM_NAME}
set RPT_DENSITY_FILE ${PROJECT_DIR}/PAR/RPT/${RPT_DENSITY_NAME}
set VLOG_NETLIST_SIM_FILE ${PROJECT_DIR}/HDL/GATE/${VLOG_NETLIST_SIM_NAME}
set VLOG_NETLIST_LVS_FILE ${PROJECT_DIR}/HDL/GATE/${VLOG_NETLIST_LVS_NAME}
set CTS_SPEC_FILE ${PROJECT_DIR}/PAR/CTS/${CTS_SPEC_NAME}
set CTS_RGUIDE_FILE ${PROJECT_DIR}/PAR/CTS/${CTS_RGUIDE_NAME}
set CTS_RPT_FILE ${PROJECT_DIR}/PAR/RPT/${CTS_RPT_NAME}
set GDS_FILE ${PROJECT_DIR}/PAR/DEX/${GDS_FILE_NAME}
set GDS_MAP_FILE ${PROJECT_DIR}/PAR/DEX/gds2.map
#-------------------------------------------------------------------------
# Procedures
#-------------------------------------------------------------------------
# make_clock_tree
#
proc make_clock_tree create_spec {
global CTS_BUFFER CTS_INV CTS_SPEC_FILE CTS_RGUIDE_FILE CTS_RPT_FILE
if { $create_spec || ![file exists $CTS_SPEC_FILE] } {
createClockTreeSpec \
-bufFootprint $CTS_BUFFER \
-invFootprint $CTS_INV \
-output $CTS_SPEC_FILE
}
specifyClockTree -clkfile $CTS_SPEC_FILE
ckSynthesis \
-rguide $CTS_RGUIDE_FILE \
-report $CTS_RPT_FILE
optDesign -postCTS -setup -drv -outDir PAR/RPT
} ;# make_clock_tree
#-------------------------------------------------------------------------
# Load configuration file
#-------------------------------------------------------------------------
loadConfig $CONF_FILE 0
commitConfig
#-------------------------------------------------------------------------
# Set operating conditions
#-------------------------------------------------------------------------
setOpCond \
-maxLibrary $TIM_LIBRARY -max $TIM_OC_MAX \
```

```
-minLibrary $TIM_LIBRARY -min $TIM_OC_MIN
#-------------------------------------------------------------------------
# Set user grids
#-------------------------------------------------------------------------
setPreference ConstraintUserXGrid 0.1
setPreference ConstraintUserXOffset 0.1
setPreference ConstraintUserYGrid 0.1
setPreference ConstraintUserYOffset 0.1
setPreference SnapAllCorners 1
setPreference BlockSnapRule 2
#-------------------------------------------------------------------------
# Define global Power nets - make global connections
#-------------------------------------------------------------------------
clearGlobalNets
globalNetConnect vdd! -type pgpin -pin vdd! -inst * -module {} -verbose
globalNetConnect gnd! -type pgpin -pin gnd! -inst * -module {} -verbose
#globalNetConnect vdd3o! -type pgpin -pin vdd3o! -inst * -module {} -verbose
#globalNetConnect vdd3r1! -type pgpin -pin vdd3r1! -inst * -module {} -verbose
#globalNetConnect vdd3r2! -type pgpin -pin vdd3r2! -inst * -module {} -verbose
#globalNetConnect gnd3o! -type pgpin -pin gnd3o! -inst * -module {} -verbose
#globalNetConnect gnd3r! -type pgpin -pin gnd3r! -inst * -module {} -verbose
#-------------------------------------------------------------------------
# Initialize floorplan
#-------------------------------------------------------------------------
floorPlan -r $FP_ASPECT_RATIO \
$FP_ROW_DENSITY \
$FP_CORE2IO $FP_CORE2IO $FP_CORE2IO $FP_CORE2IO
fit
saveDesign $SAVE_DESIGN_FP_FILE
#-------------------------------------------------------------------------
# Add CAP cells
#-------------------------------------------------------------------------
addEndCap -preCap ENDCAPL -postCap ENDCAPR -prefix ENDCAP
#-------------------------------------------------------------------------
# Create and route power rings and power stripes
#-------------------------------------------------------------------------
addRing \
-around core \
-nets { gnd! vdd! } \
-width_bottom $PR_WIDTH -width_top $PR_WIDTH \
-width_left $PR_WIDTH -width_right $PR_WIDTH \
-spacing_bottom $PR_SPACING -spacing_top $PR_SPACING \
-spacing_left $PR_SPACING -spacing_right $PR_SPACING \
-layer_bottom $PR_LAYER_TB -layer_top $PR_LAYER_TB \
-layer_left $PR_LAYER_LR -layer_right $PR_LAYER_LR \
-center 1 \
-tl 1 -tr 1 -bl 1 -br 1 -lt 1 -lb 1 -rt 1 -rb 1 \
-stacked_via_bottom_layer MET1 -stacked_via_top_layer MET4 \
-threshold 0.7
if { $ADD_STRIPES } {
addStripe \
-nets { gnd! vdd! } \
-number_of_sets $ST_NUM_SETS \
-spacing $ST_SPACING \
-layer $ST_LAYER_V \
-width $ST_WIDTH \
-xleft_offset $ST_XOFS_L
}
sroute \
-jogControl { preferWithChanges differentLayer } \
-nets { gnd! vdd! }
saveDesign $SAVE_DESIGN_PR_FILE
#-------------------------------------------------------------------------
# Core cell placement
#-------------------------------------------------------------------------
if { $PLACE_TIMING } {
amoebaPlace $PL_EFFORT -timingdriven
} else {
```

```
amoebaPlace $PL_EFFORT
}
setDrawMode place
saveDesign $SAVE_DESIGN_PL_FILE
#-------------------------------------------------------------------------
# Create clock tree (optional)
#-------------------------------------------------------------------------
if { $CLOCK_TREE } {
make_clock_tree $CTS_CREATE_SPEC
saveDesign $SAVE_DESIGN_CT_FILE
}
#-------------------------------------------------------------------------
# Add filler cells
#-------------------------------------------------------------------------
addFiller -cell FILL25 FILL10 FILL5 FILL2 FILL1 -prefix FILLER
saveDesign $SAVE_DESIGN_PF_FILE
#-------------------------------------------------------------------------
# Route design (Nanoroute)
#-------------------------------------------------------------------------
if { $ROUTE_TIMING } {
setNanoRouteMode -quiet -timingEngine CTE
setNanoRouteMode -quiet -routeWithTimingDriven true
setNanoRouteMode -quiet -routeTdrEffort 0
}
globalDetailRoute
optDesign -postRoute -setup -drv -outDir PAR/RPT
saveDesign $SAVE_DESIGN_RO_FILE
setDrawMode place
#-------------------------------------------------------------------------
# Verifications
#-------------------------------------------------------------------------
fillNotch -report $RPT_NOTCH_FILE
verifyConnectivity \
-type all \
-error 1000 \
-warning 50 \
-report $RPT_CONN_FILE
verifyGeometry \
-allowSameCellViols \
-allowRoutingBlkgPinOverlap \
-allowRoutingCellBlkgOverlap \
-report $RPT_GEOM_FILE
verifyMetalDensity \
-detailed \
-report $RPT_DENSITY_FILE
#-------------------------------------------------------------------------
# Extract parasitics
#-------------------------------------------------------------------------
setExtractRCMode \
-detail \
-rcdb $TIM_RCDB_FILE \
-relative_c_t 0.01 \
-total_c_t 5.0 \
-reduce 5
extractRC
#-------------------------------------------------------------------------
# Generate RC and timing files
#-------------------------------------------------------------------------
rcOut -spef $SPEF_FILE
delayCal -sdf $SDF_FILE
#-------------------------------------------------------------------------
# Generate reports
#-------------------------------------------------------------------------
reportGateCount -outfile $RPT_GATE_COUNT_FILE
# Timings
#
setCteReport
setAnalysisMode -setup -async -skew -noClockTree -sequentialConstProp
```

```
reportAnalysisMode
buildTimingGraph
checkTA -verbose > $RPT_CHECK_TA_FILE
reportTA \
-format { hpin arc cell delay arrival required slew fanout load } \
-late \
-max_points 10 \
-net \
> $RPT_REPORT_TA_FILE
#--------------------------------------------------------------------------
# Save netlist
#--------------------------------------------------------------------------
saveNetlist -excludeLeafCell $VLOG_NETLIST_SIM_FILE
saveNetlist -physical $VLOG_NETLIST_LVS_FILE
#--------------------------------------------------------------------------
# Save GDS2
#--------------------------------------------------------------------------
streamOut $GDS_FILE \
-mapFile $GDS_MAP_FILE \
-libName ADDSUB \
-structureName $DESIGN_NAME \
-stripes $ST_NUM_SETS \
-units 1000 \
-mode ALL
```