Cadence First Encounter Tutorial

Files for this tutorial can be downloaded from:
www.cs.wright.edu/~emmert/tutorials/enc_files.tar.gz

Dr. J M Emmert

Configuration File

- This file contains information used to setup your design for synthesis from RTL to the layout or gds levels of circuit abstraction
- Some (not all) of the file options:
  - Configuration file name: map.conf
  - RTL verilog input filename: map.v
    - This file can come from Cadence RTL compiler, Synopsys DC Ultra, or other HDL compiler
  - Top level design name: CHIP
    - This is your top level design name
  - gds libraries if available for gds extraction
  - Timing library files: *.tlf files
  - IO file that defines the asic pin locations: map.io
  - Timing file generated by the synthesis tool: map.sdc
  - Library exchange format or lef library file names
  - Footprints for buffers and inverters
    - For example, {buf} would include all cells whose names started with "buf"
  - The names for the vdd and vss pins

```
################################################
#                                              #
#  FirstEncounter Input configuration file : map.conf          #
#                                              #
################################################
global rda_Input
set cwd .
set rda_Input(import_mode) {-treatUndefinedCellAsBbox 0}
set rda_Input(ui_netlist) "map.v"
set rda_Input(ui_netlisttype) {Verilog}
set rda_Input(ui_rtllist) ""
set rda_Input(ui_ilmdir) ""
set rda_Input(ui_ilmlist) ""
set rda_Input(ui_ilmspef) ""
set rda_Input(ui_settop) {0}
set rda_Input(ui_topcell) {CHIP}
set rda_Input(ui_celllib) ""
set rda_Input(ui_iolib) ""
set rda_Input(ui_areaiolib) ""
set rda_Input(ui_blklib) ""
set rda_Input(ui_kboxlib) ""
set rda_Input(ui_gds_file) “vtvtlib25.gds"
set rda_Input(ui_oa_oa2lefversion) {}
set rda_Input(ui_view_definition_file) ""
set rda_Input(ui_timelib,max) ""
set rda_Input(ui_timelib,min) ""
set rda_Input(ui_timelib) “vtvtlib25.tlf"
set rda_Input(ui_smodDef) ""
set rda_Input(ui_smodData) "“
.
.
.
.
```

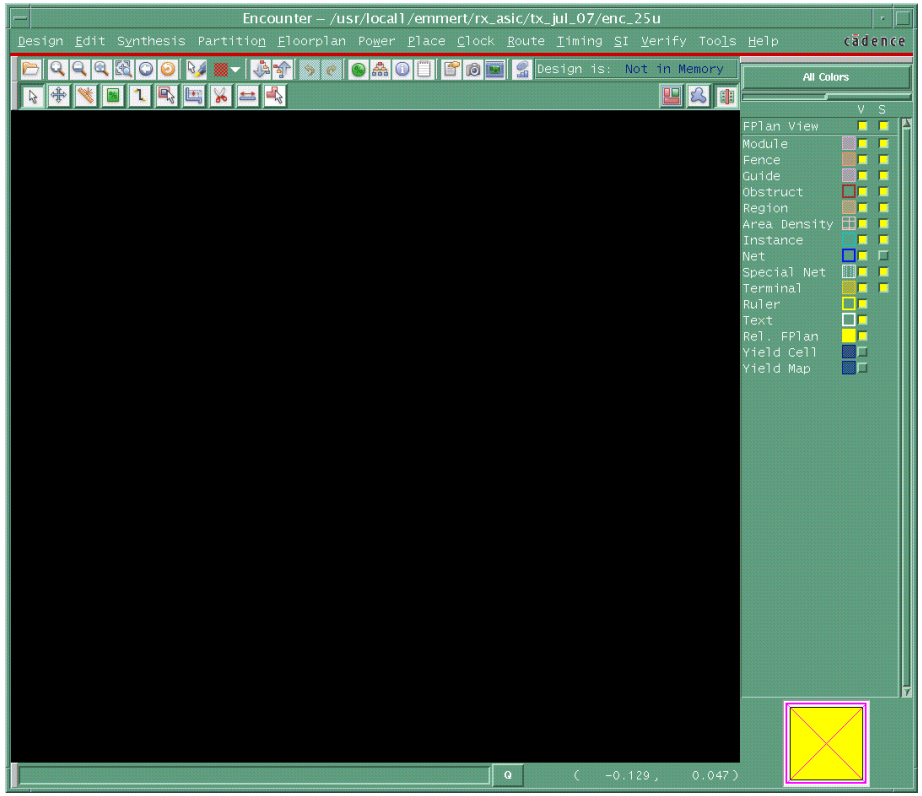
Starting Encounter

- To start the tool, first you must source the environment file

  `source set_cadence_soc_env <CR>`

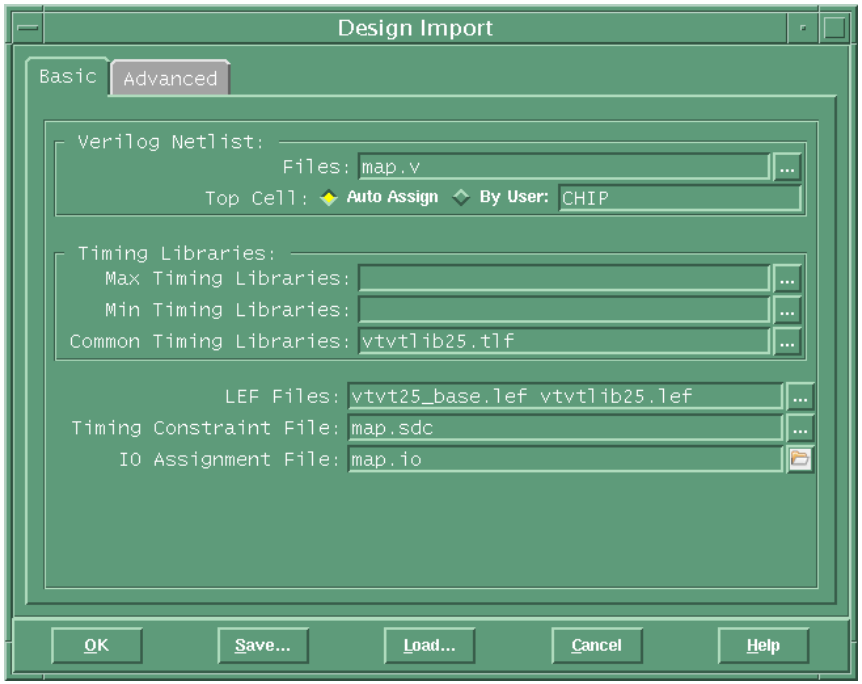
  – This file sets up the paths and license file access to run First Encounter
- Then on the command line type

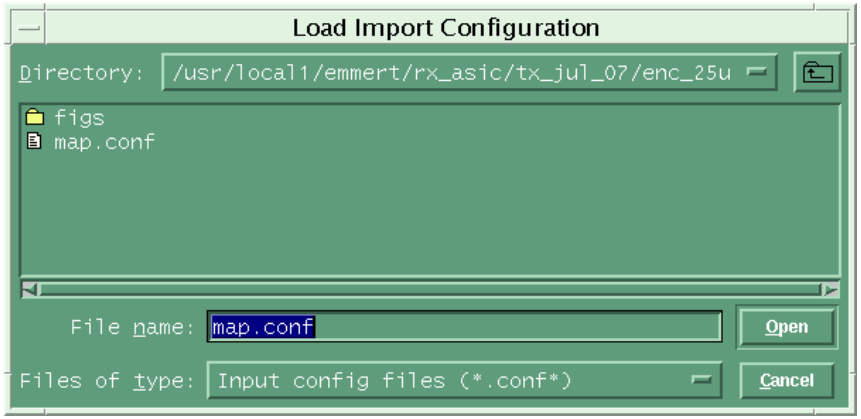
  `encounter <CR>`



Read In Your Design

- From the Design tab
  `Design → Import Design`
- From the Design Import window
  `Load`
- Type in or select the configuration file name (eg: map.conf) and select
  `Open`
- In the Design Import window select
  `OK`





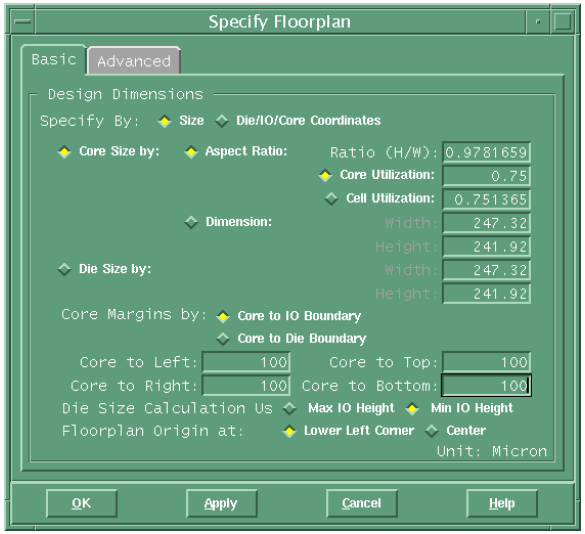
Floor Plan Your Design

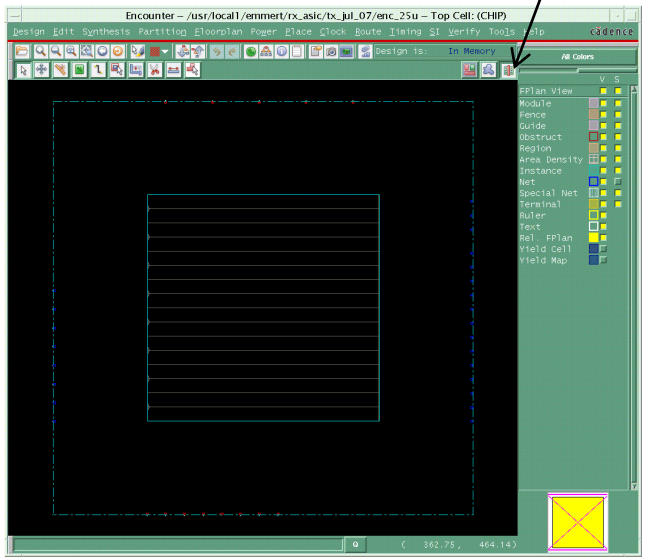
- From the Floorplan tab

  `Floorplan → Specify Floorplan`

- There are many options for defining the floorplan
- Example below shows
  - Size
  - Core Utilization of 75%
  - Core space for Power Rings 100.0 from Core to IO boundary

Note: Select the layout tab button to remove the pink box on the left.

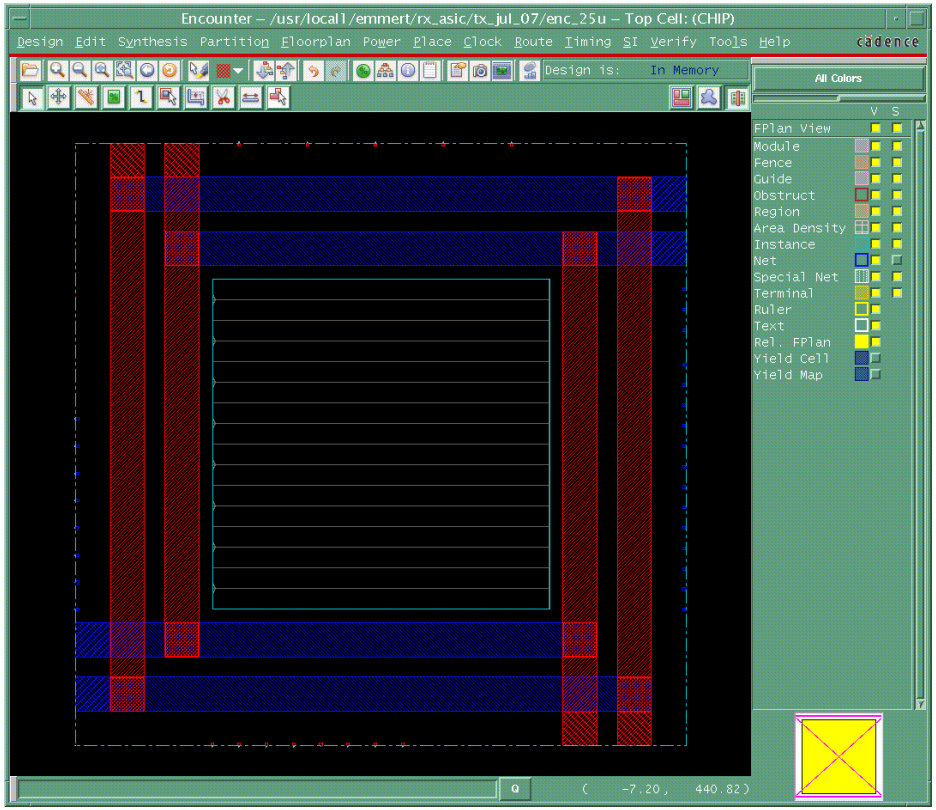


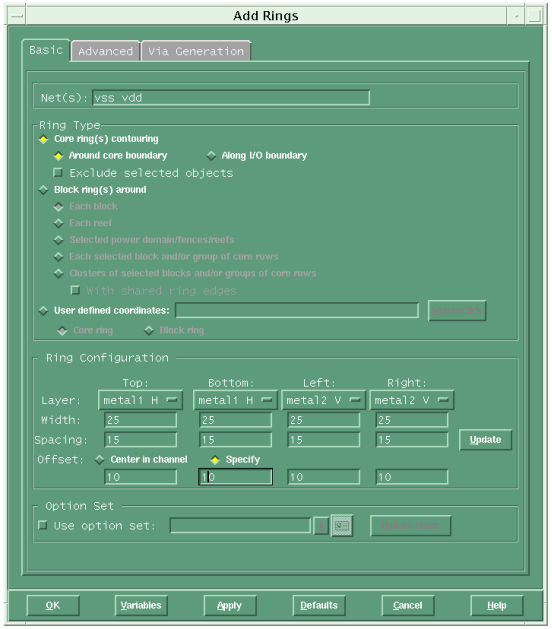


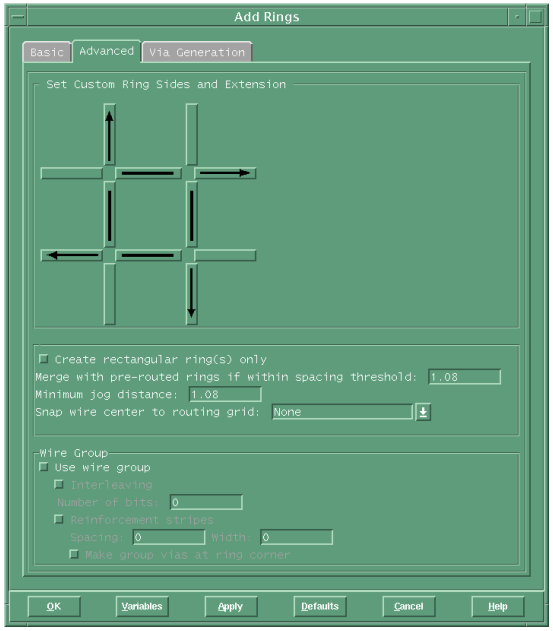
Power Rails

- In this step we add Power Rails
- From the Power Tab

  `Power → Power Planning → Add Rings`



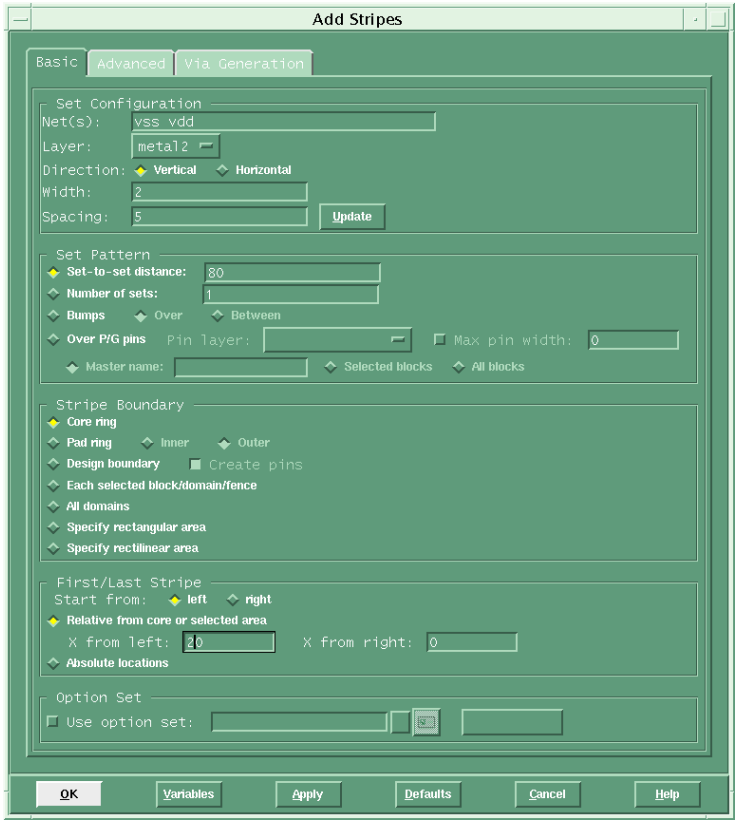


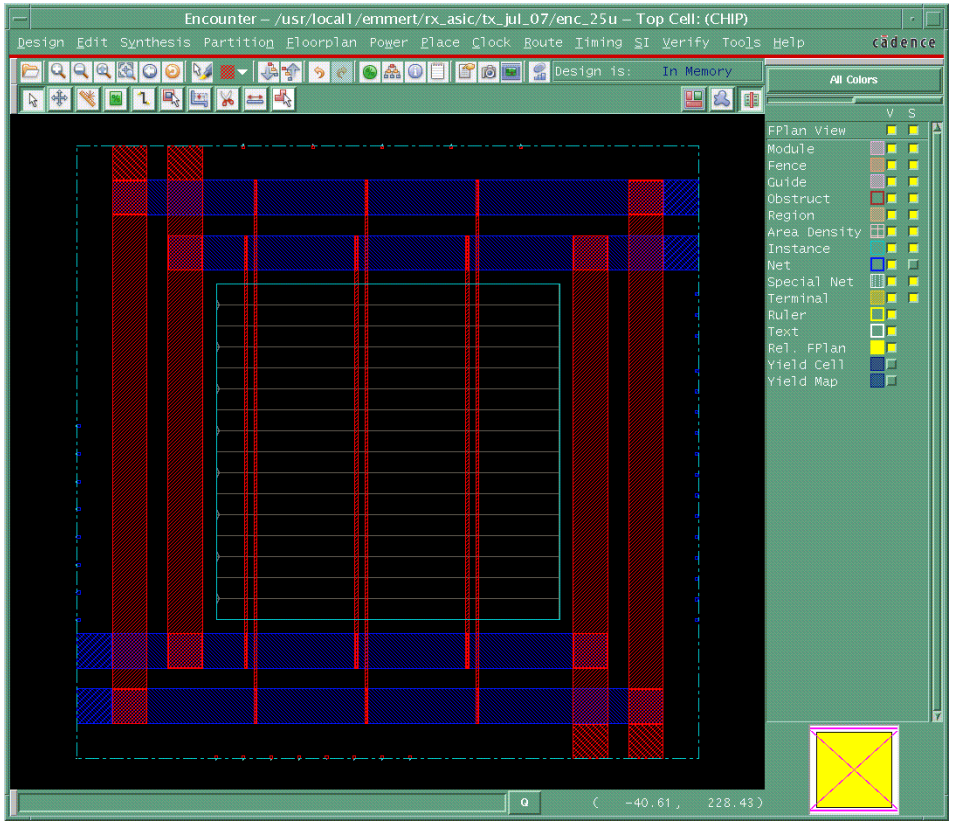


Power Stripes

- In this step we add Power Stripes
- From the Power Tab

  `Power → Power Planning → Add Stripes`

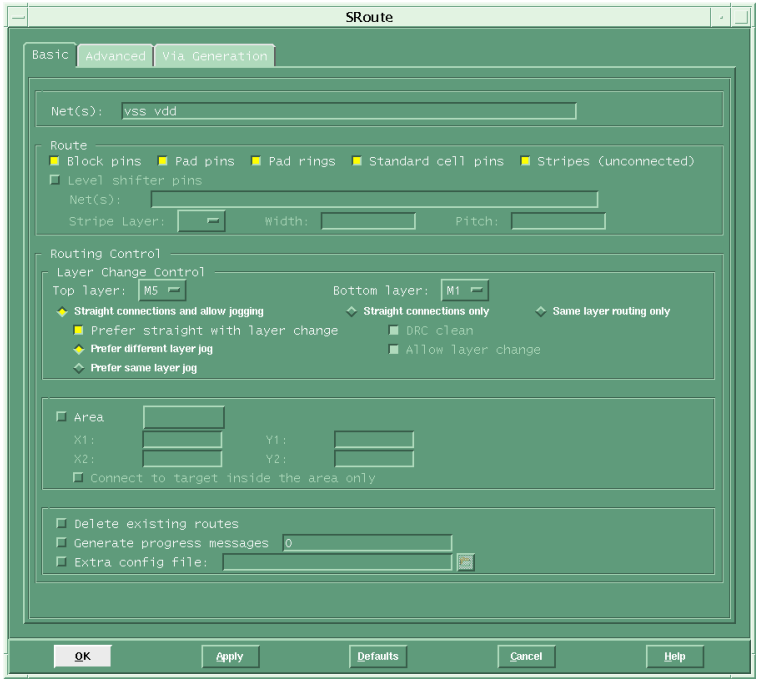


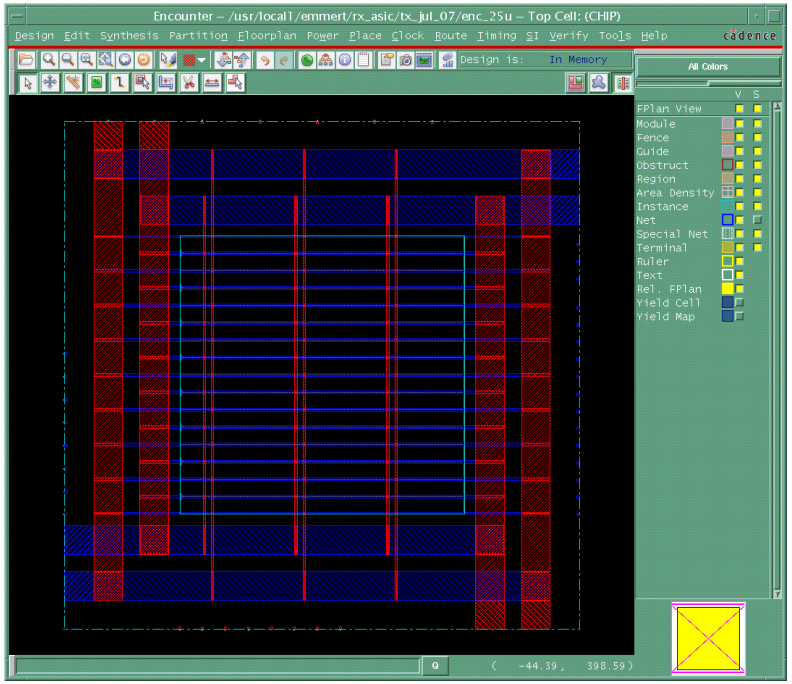


Routing Power Stripes

- Route vdd and vss

  `Route → Special Route(SRoute) → OK`





Placement

- Next we perform placement
- If required, insert well contacts first

  `Place → Fill → Add Well Tap`

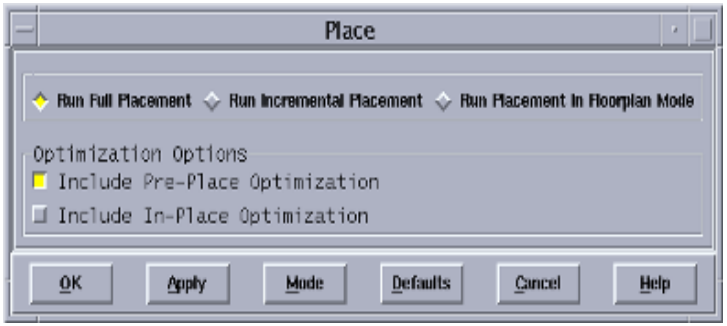
  - Hit the Select button to view available well tap cells in the library
  - Set the space between taps to the distance required for the target technology
- If well taps are included within the cells or for SOI designs, continue on to placement (next page)

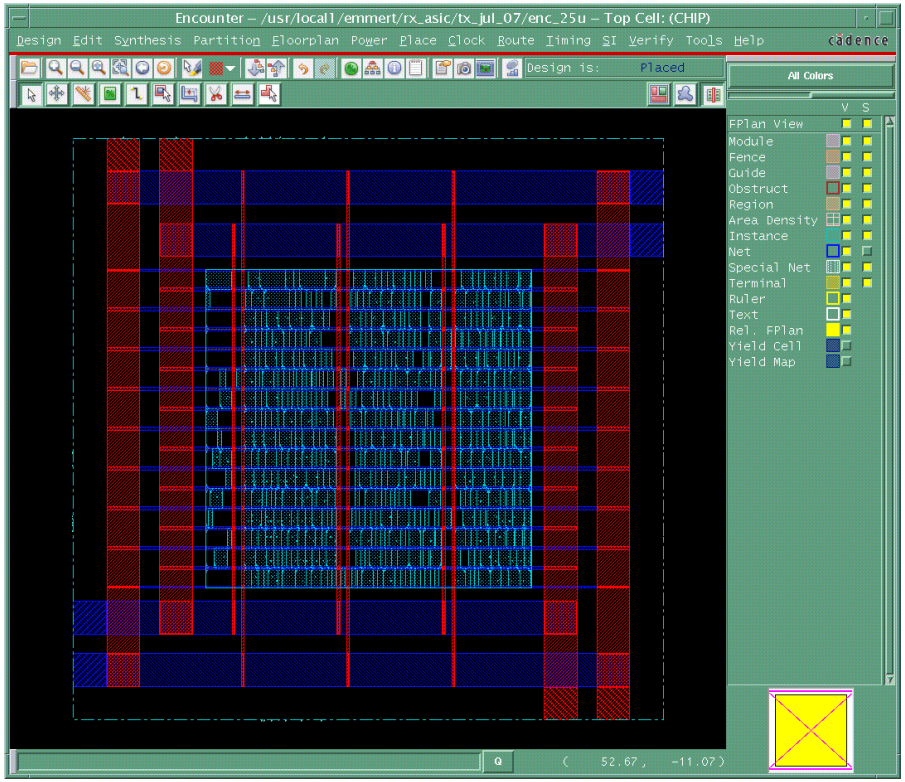
Placement

- Next, place the cells

  `Place → Standard Cells and Blocks`

  `OK`

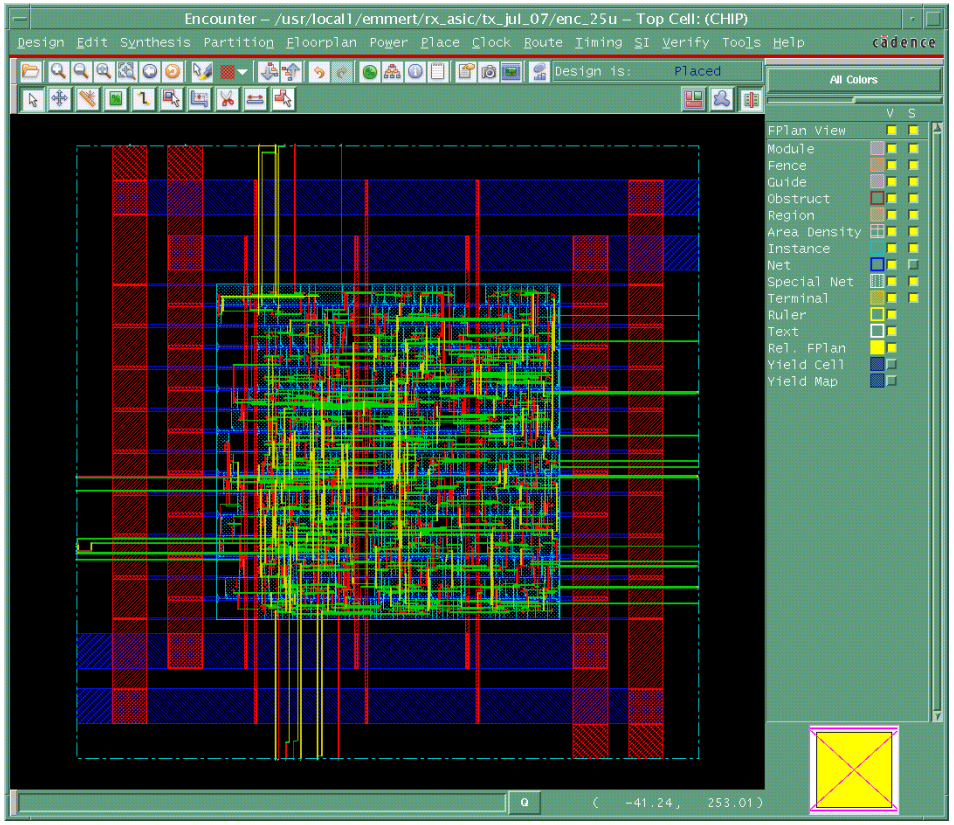




Clock Tree Generation

- For sequential circuits, add clock trees
- The map.ctstch file describes the requirements for the clock tree

```
Clock → Design Clock
OK
```

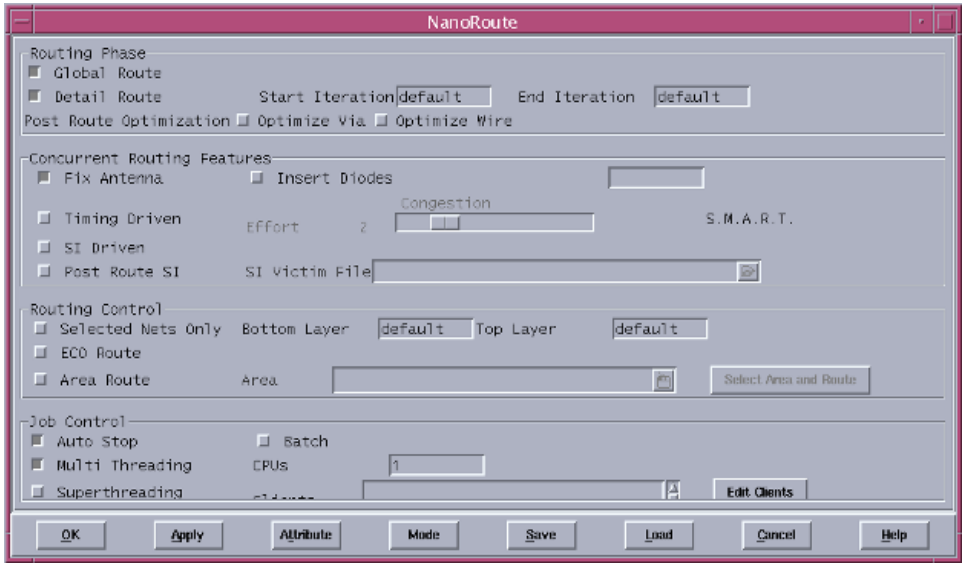


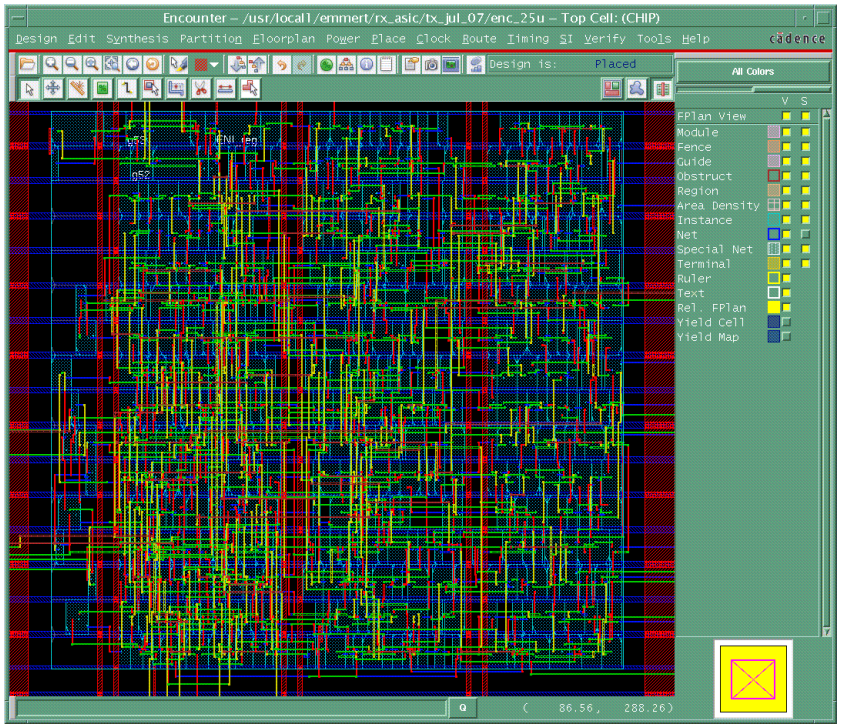
Routing

- Now use nanoroute to route the design

  `Route → NanoRoute → Route`

  `OK`





Fill

- Now you will add fill cells and metal fill

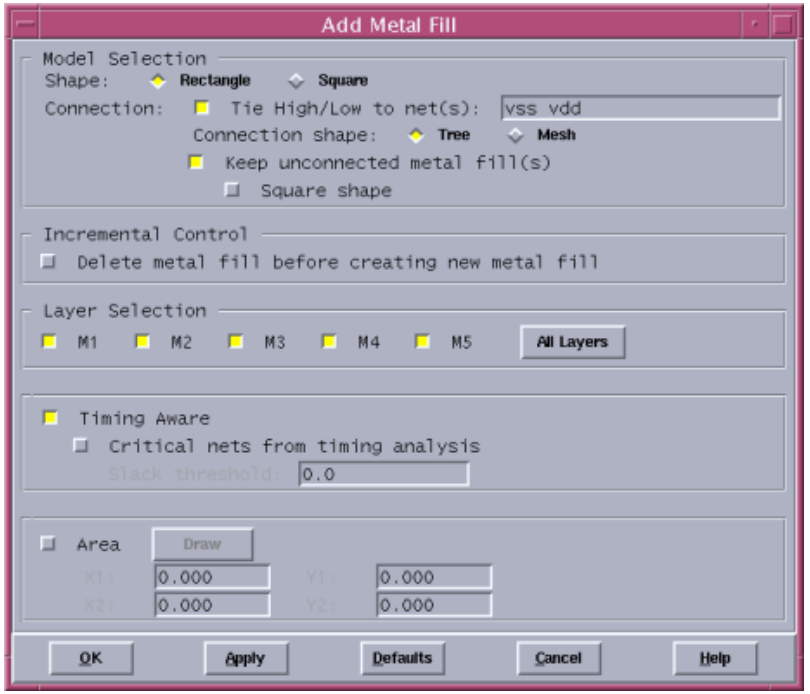
`Place → Filler → Add Filler`

`Select` to select fill cells

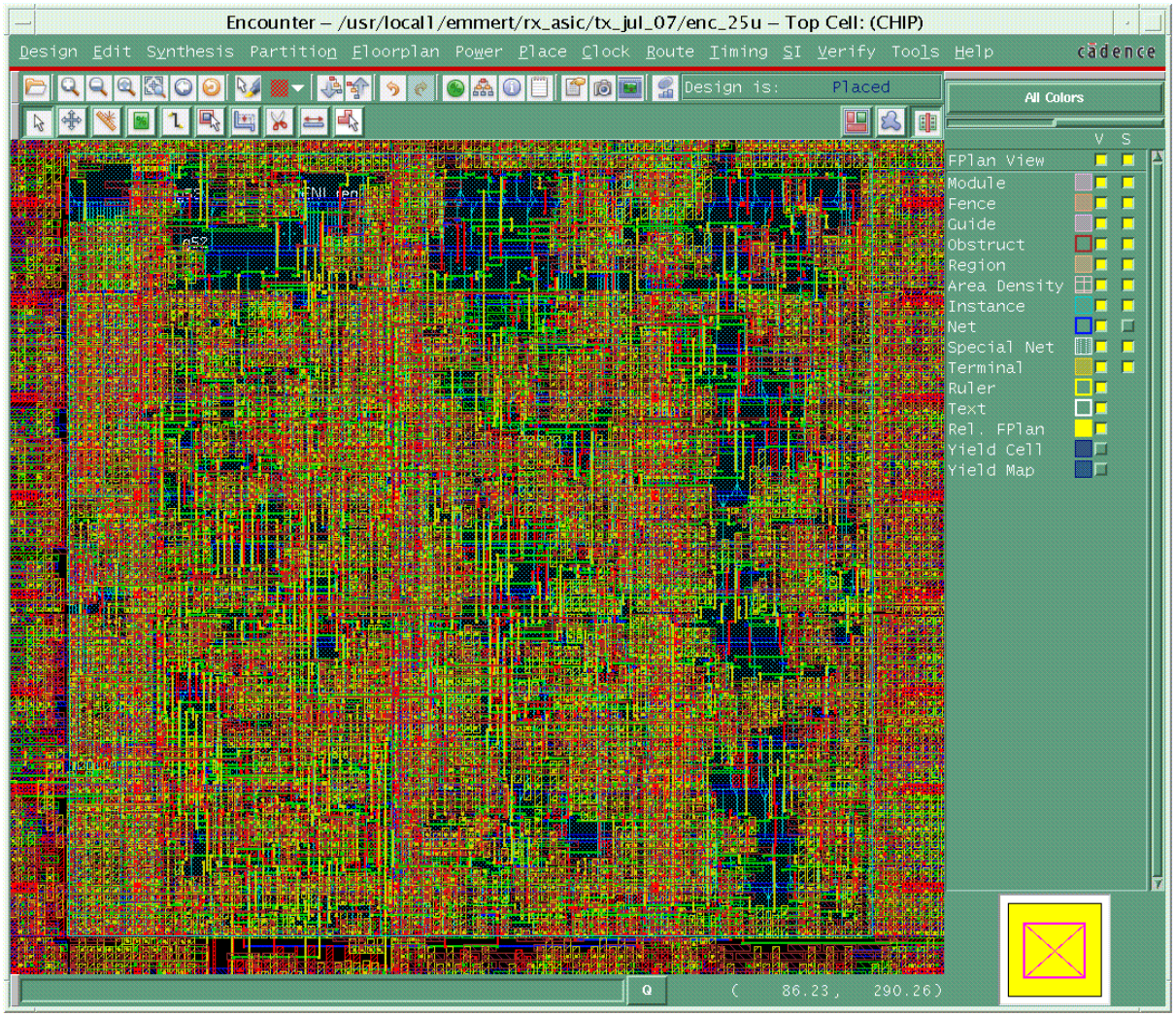
`OK`

`Route → Metal Fill → Add`

`OK`



Placed, Routed, and Filled

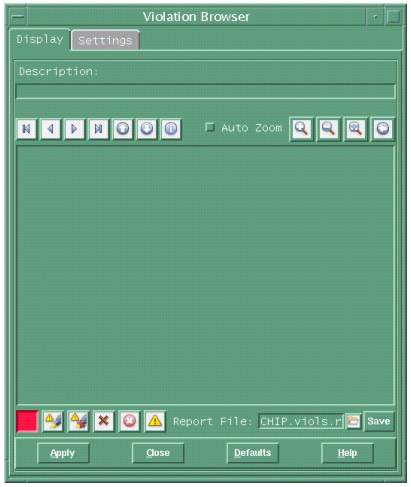


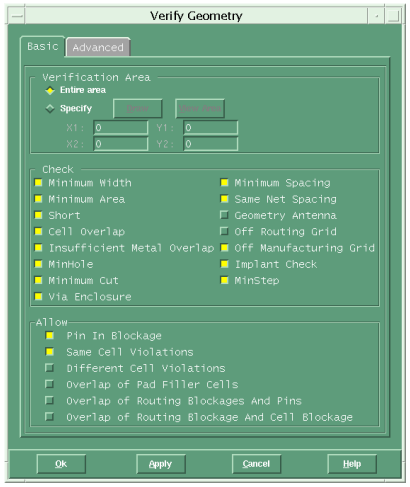
Dr. J M Emmert

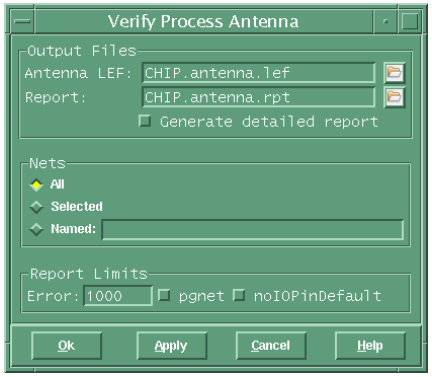
Verification

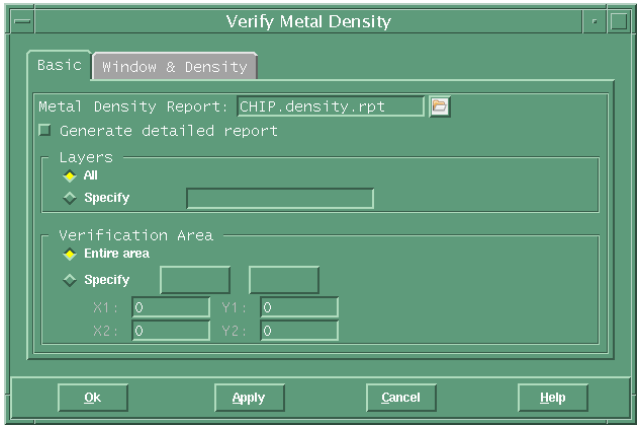
- Verify the design and view the errors

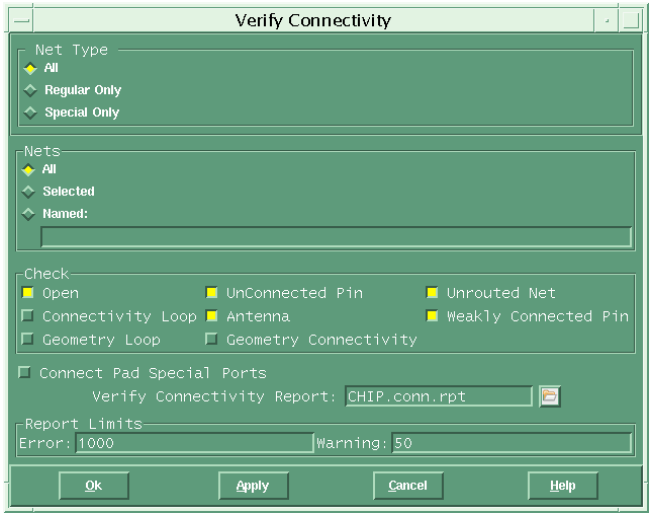
```
Verify →
        Verify Geometry
        Verify Metal Density
        Verify Connectivity
        Violation Browser
```







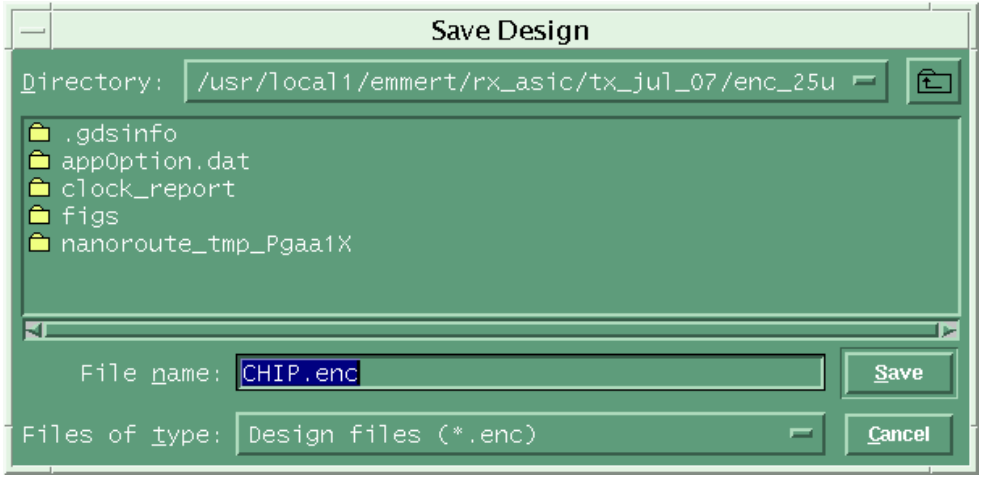


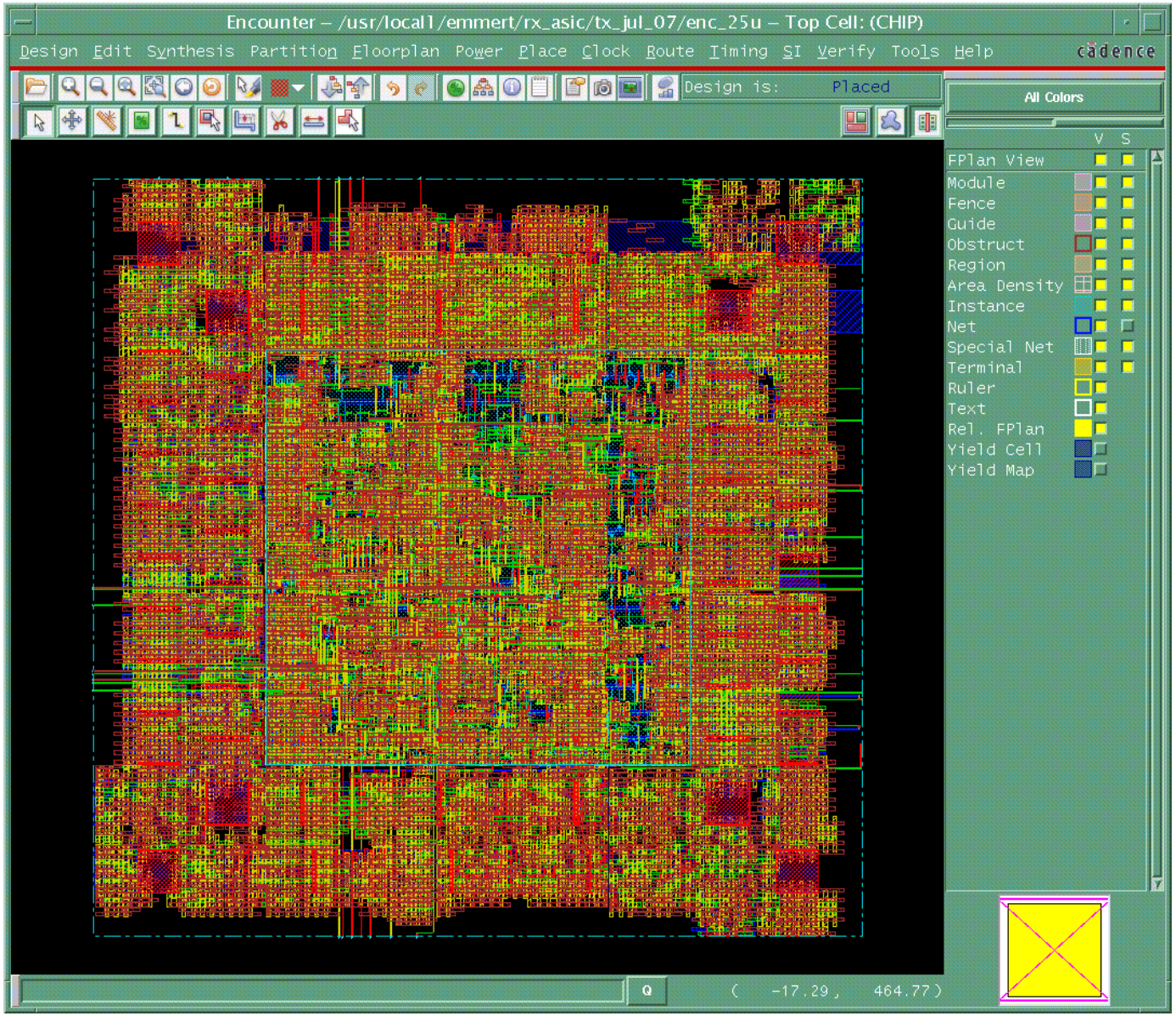


Save

- Save the design and write out the design exchange format (def)

```
Design → Save Design As → SoCE
Save
Design → Save → DEF
OK
```




Encounter – /usr/local1/emmert/rx_asic/tx_jul_07/enc_25u – Top Cell: (CHIP)
Design Edit Synthesis Partition Floorplan Power Place Clock Route Timing SI Verify Tools Help
cadence
Design is: Placed
All Colors
V S
FPlan View
Module
Fence
Guide
Obstruct
Region
Area Density
Instance
Net
Special Net
Terminal
Ruler
Text
Rel. FPlan
Yield Cell
Yield Map
Q
(-17.29 , 464.77)

Dr. J M Emmert

Notes

- For instructions on tool functionality and capabilities, after sourcing the environment file, type

  `cdnshelp <CR>`
- For help while in encounter just type

  `help` *`command_name`* `<CR>`
  - Wild cards (*) can also be used
- Make sure you save at regular intervals during the design process
- All commands can be run on the command line without starting the GUI by starting encounter with the nowin option

  `encounter –nowin <CR>`
  - At the command line you can source prewritten command files or enter the command