Tutorial for Synthesize, Place & Route Memory Block as Black Box

04/15/2014 By Junxiang Wu

The object of this tutorial is to show you how to synthesize, place & route a design has synthesizable memory component(s) without a memory complier. We will treat memory component(s) (or other synthesizable modules) as back box(es) in place & route, but leave that in high level HDL during simulation.

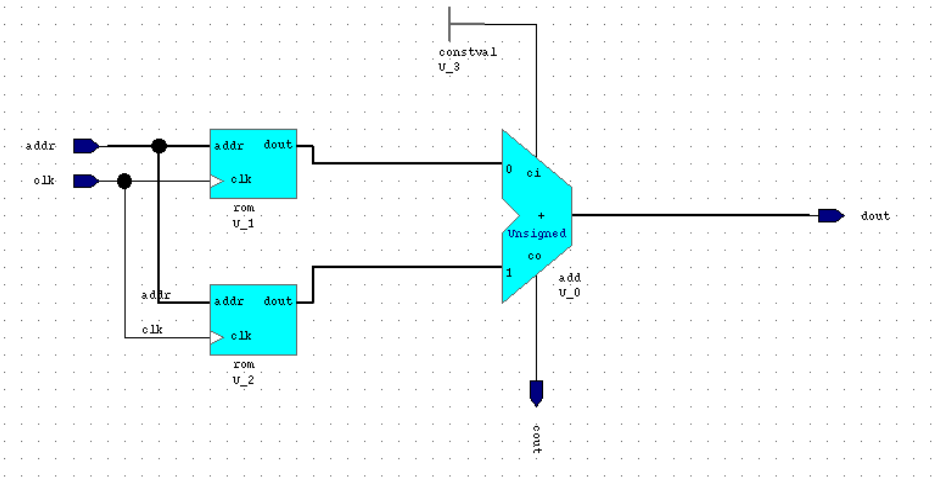
To achieve that, we will let memory components skip DC, and configure a black box for place & route them with the rest parts in encounter.

You should have read the tutorials relating to HDL Designer, Simulation after design, synthesizing, place & route, have a general idea about the design flow and some hands-on.

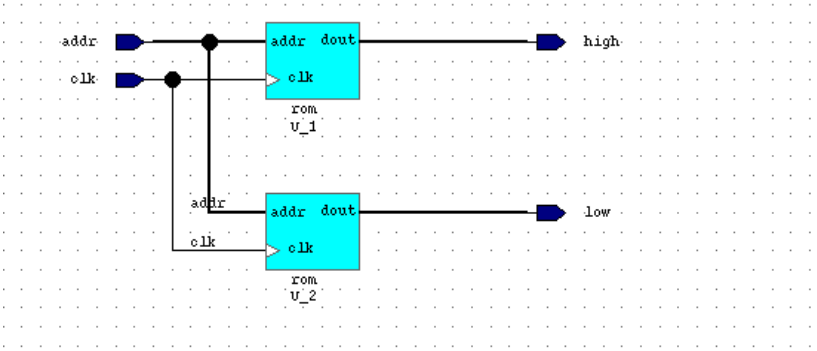
Next, we will go through the design flow by highlighting the deference with the general one.

Part1. Design & Synthesize

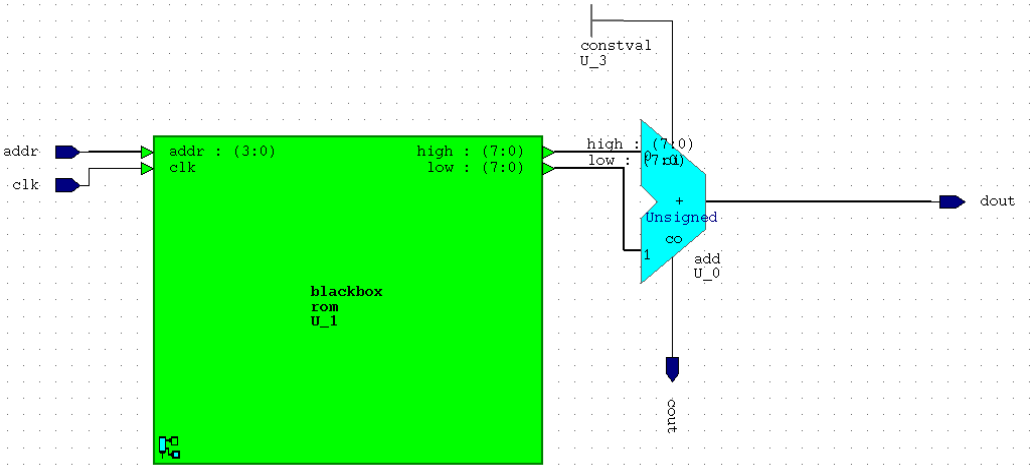
1. In **HDL Designer**: separate the module doesn't need to synthesize. .i.e. make it a separate module and then instantiate it in the top design. For example, here's two ROMs with a adder:



Separate the rom



Instantiate in the top



2. In **DC**, only synthesize the TOP design.

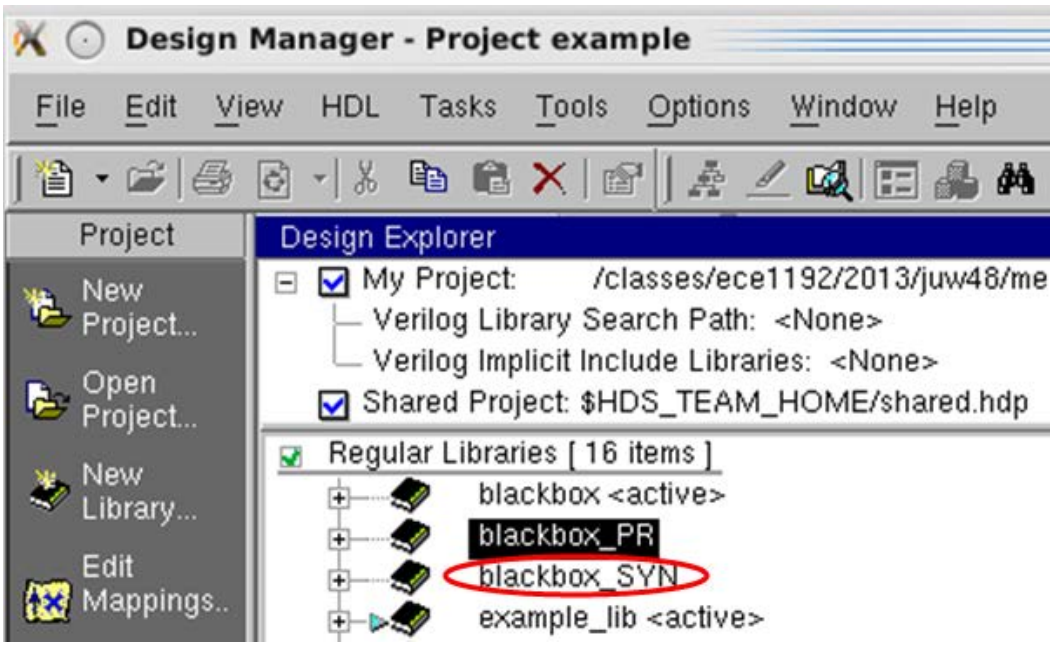
As for our example, we have rom_struct.vhd , adder_top_struct.vhd, copy both files to the working directory of DC, only synthesize adder_top.vhd.

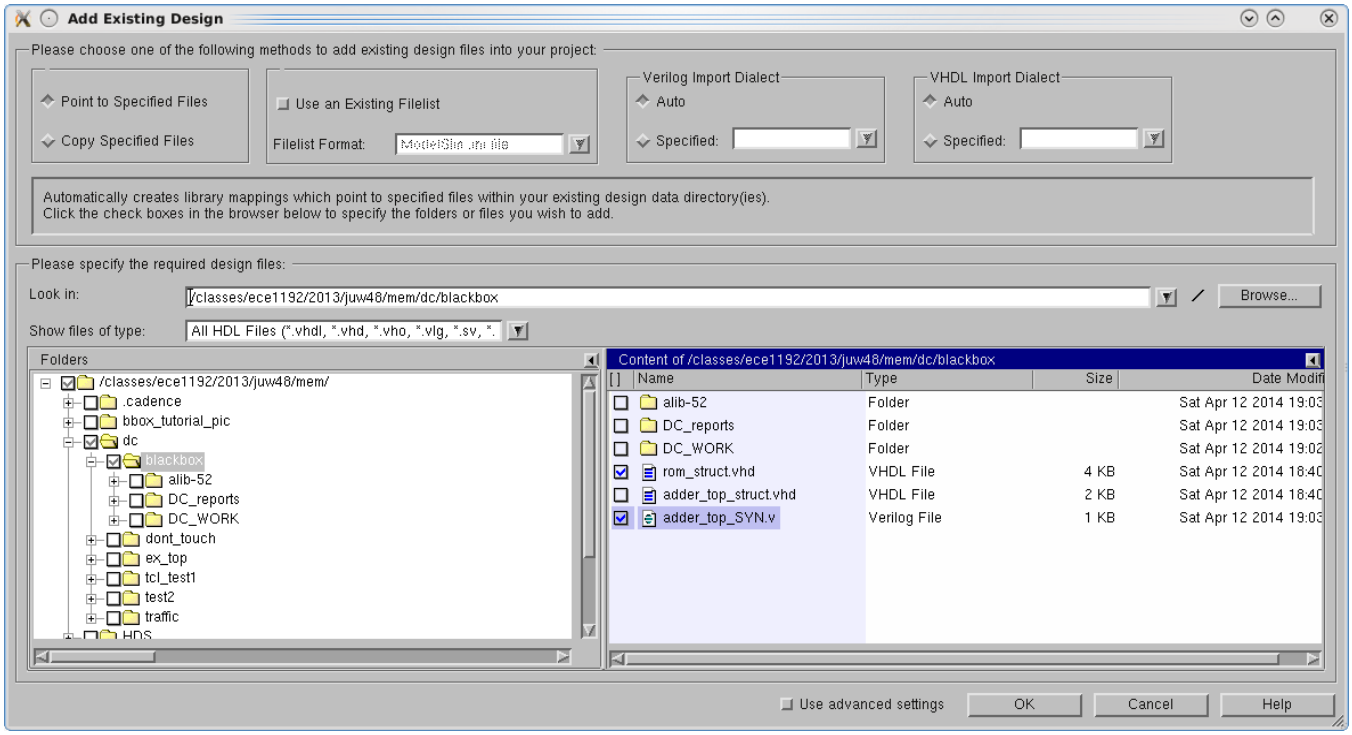
>dc_shell-t -f adder_top.tcl

Designer Complier will complain (warn) that black box funded. That is OK as long as it will successfully synthesize because we want it this way.

```
# HDL file names (.v or .vhd)                    #
set my_HDL_files [list adder_top_struct.vhd]

# Top-level Module / Entity name                 #
set my_toplevel adder_top

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

# Target frequency in MHz for optimization       #
set my_clk_freq_MHz 100

# Delay of input signals (Clock-to-Q, Package etc.)#
set my_input_delay_ns 0.1

# Reserved time for output signals (Holdtime etc.) #
set my_output_delay_ns 0.1
```

3. Now we have synthesized adder_top.v, add it back to our _SYN library in HDL designer with the original rom_struct.vhd.





Since we copy rom_struct.vhd to the same directory at the beginning, we can easily find it here.

Also, don't forget to add the standard cell library "gscl45nm.v", and provide the root with timing file adder_top_SYN.sdf. Then you can do after synthesize simulation.

Part2. Configure Black Box

Before we start place & route with encounter, we need configure and generate a LEF file for our black box.

LEF (Library Exchange Format) is ASCII file defines the elements of an IC process technology and associated library of cell models.

For us, we need the LEF file to let encounter know the size and the pins of black box. We use a script to generate LEF file by looking at the VHDL file. The usage of the script is:

>perl bbox.pl –option arg1 arg2 filename.vhd

It has two options

>*perl bbox.pl –s width height filename.vhd*; set specify the width and height directly (unit: um)

>*perl bbox.pl –m addr data filename.vhd*; if it's a memory block, provide with the address width and data width

>*perl bbox.pl filename.vhd*; in default, it will simply do a naïve calculation for the size based on number of ports.

So we can get the rom.lef by typing "*perl bbox.pl rom_struct.vhd*"

Note*: There is more than one way to do black box with encounter, even might without a LEF file. It is just one that works, not the best one. If you find another way to do it that better fits your design, feel free to do so.*

Part3. Place & Route

1. Before loading the design, **modify "globals"**.

In addition to the modifications needed for the .globals and .view files, there are two more areas we should change in the ".gobals"

1) add blackbox.lef file to init file list:

set init_lef_file {/classes/ece1192/2014/CLASS/lib/gscl45nm.lef **filepath**}

2) set init_import_mode as treat undefined cell as black box

set init_import_mode {-treatUndefinedCellAsBbox 1 –keepEmptyModule 0}

Then we can load our design

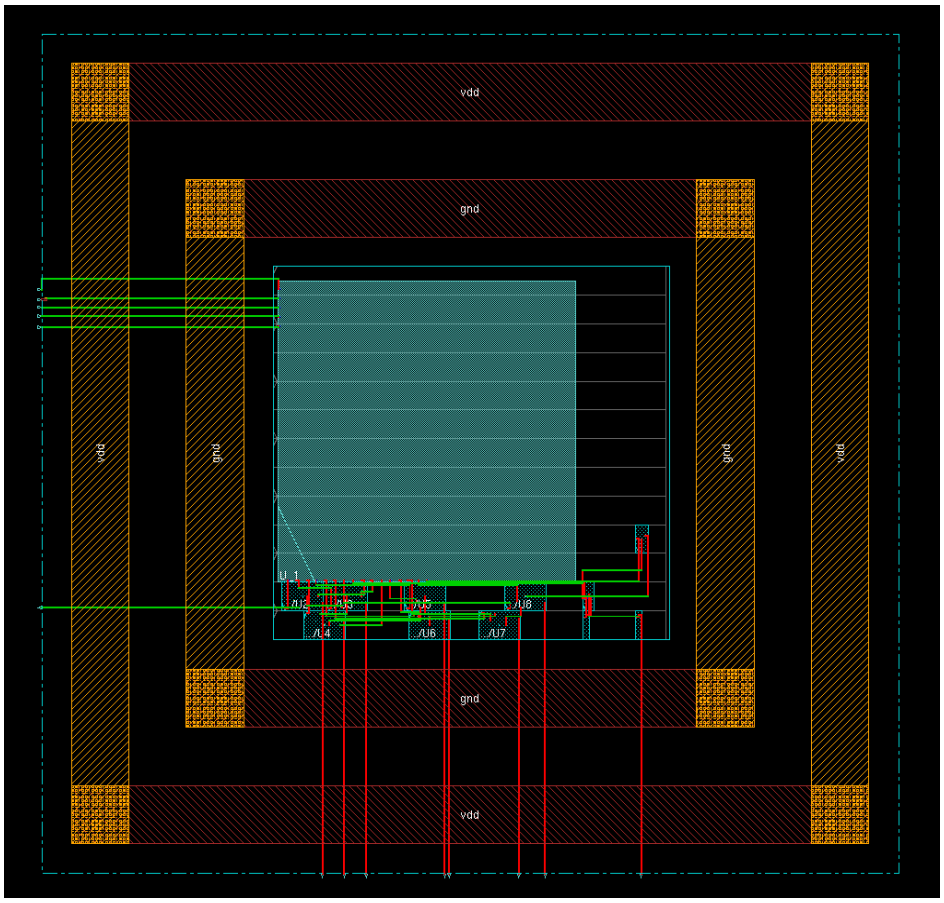
Design --> Import Design...

click "Load...", then find the modfiied configuration file "ProjectName.gobals" and click OK.

Here the *blackbox.globals*

```
##############################################################
#
# Version 1.1
#

set defHierChar {/}
set locv_inter_clock_use_worst_derate false
set cts_cell_list {CLKBUF1 CLKBUF2 CLKBUF3 INVX1 INVX2 INVX4 INVX8 BUFX2 BUFX4}
set init_oa_search_lib {}
set lsgOCPGainMult 1.000000
set conf_ioOri {R0}
set init_verilog {adder_top_SYN.v}
set init_pwr_net {vdd}
set init_mmmc_file {adder_top.view}
set init_assign_buffer {1}

#set init_lef_file {/classes/ece1192/2014/CLASS/lib/gscl45nm.lef }
set init_lef_file {/classes/ece1192/2014/CLASS/lib/gscl45nm.lef /classes/ece1192/2013/juw48/mem/virtuoso/blackbox/rom.lef}

set conf_in_tran_delay {50.0ps}
set conf_qxconf_file {NULL}

#set init_import_mode {-treatUndefinedCellAsBbox 0 -keepEmptyModule 1 }
set init_import_mode {-treatUndefinedCellAsBbox 1 -keepEmptyModule 0 }

set conf_qxlib_file {NULL}
set init_design_settop 0
set init_layout_view {layout}
set init_gnd_net {gnd}
set init_abstract_view {abstract}
set timing_case_analysis_for_icg_propagation false
```

2. After doing Floorplan, Power Panning, do **placement first,** before special route

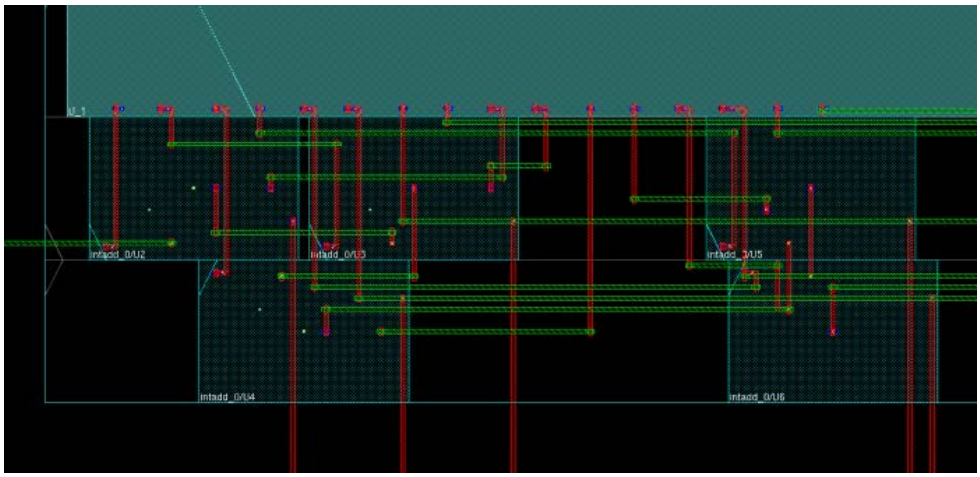


Click Place -> Standard Cells...

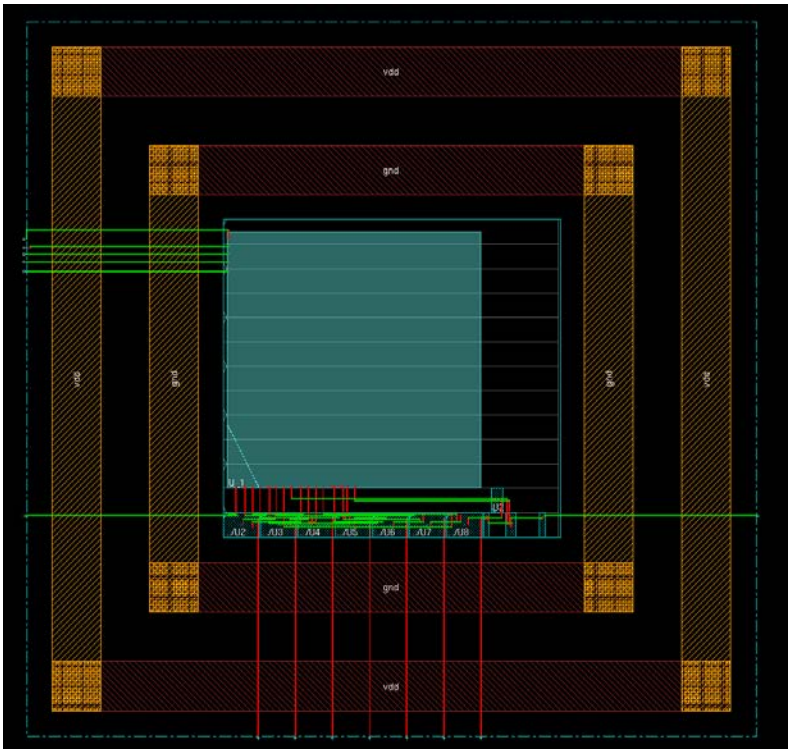
Click "Mode", (Placement Mode) deselect "Ignore Scan Connections" and "Reorder Scan Connection", and click OK.

Click OK, on the main placement window.

After it finished, click Place -> Check Placement..., and then OK



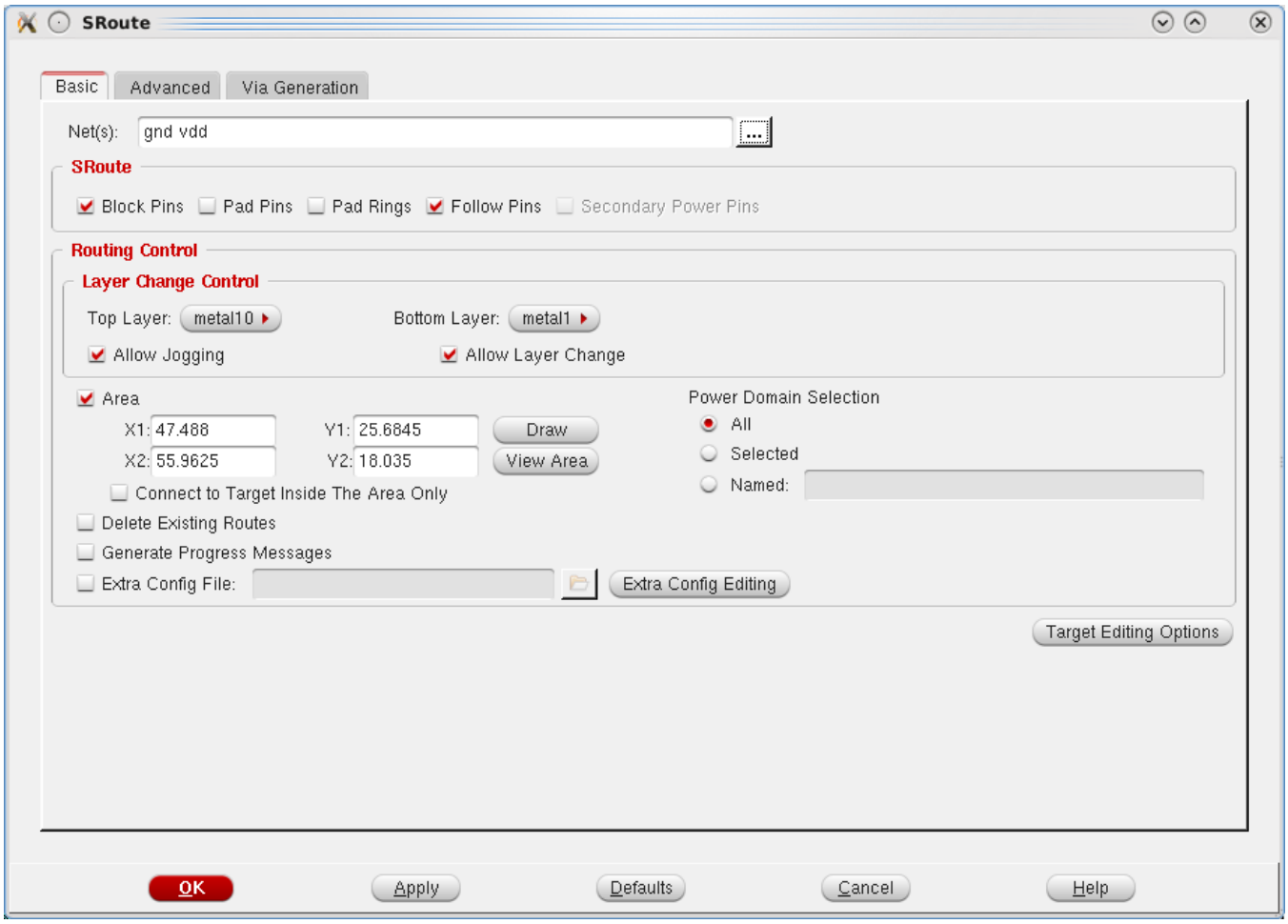
It should look like the picture above. Notice here the standard cell is tightly close to our Macro cell (black box), we can **do this step again** (place standard cells then check placement). Because if they're connected, it will have a short violation on the power rails.



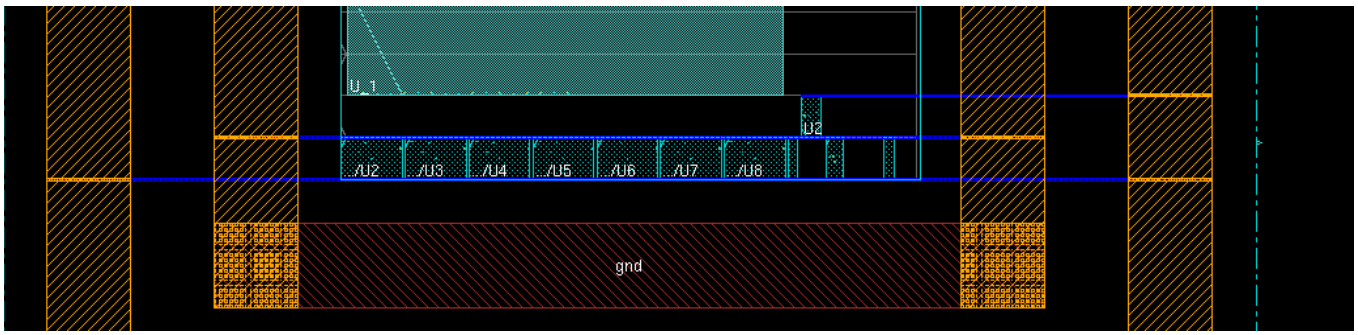
3. **Special Route**

Special route is to route the power rails of our standard cells. We don't want to route it to our Macro cell. So after set the options of special route, we Check "Area" -> Clik Draw-> Choose the area we want to be routed -> Click Apply.

You might need to repeat this until standard cells are well routed with vdd and gnd.

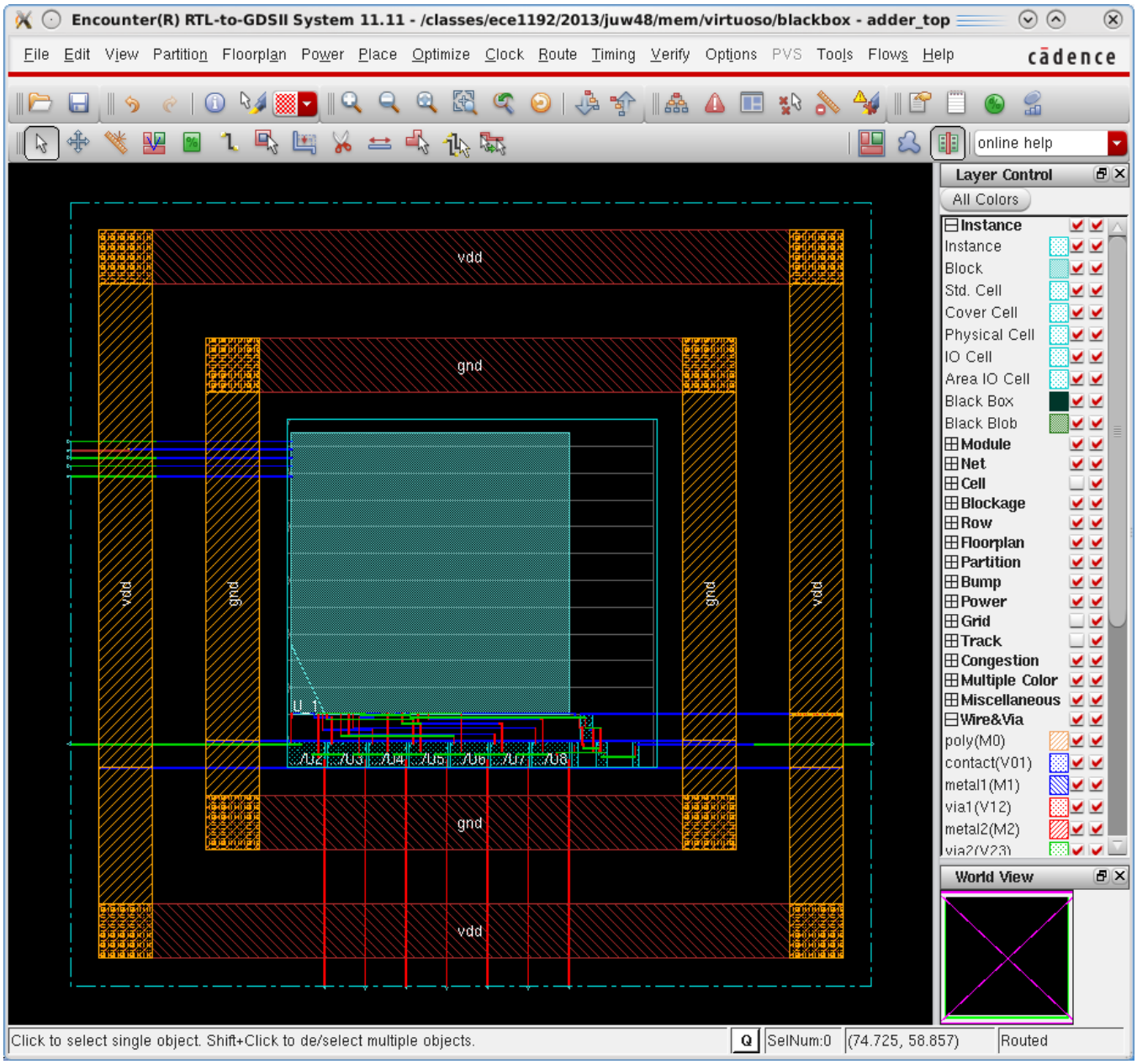


example.



4. Clock Tree Synthesis, do timing and power analysis as you need. Eventually, do **Actually Routing**: Nanoroute.

It result will look like this:



You can verify the design, and if it all goes well, export the netlist add_top_PR.v and write sdf file.

5. Filler Insertion

Notes: You may notice there is one more step before you can save and export the design: add filler cell.

That is because it may generate some new violations that had to eliminate.

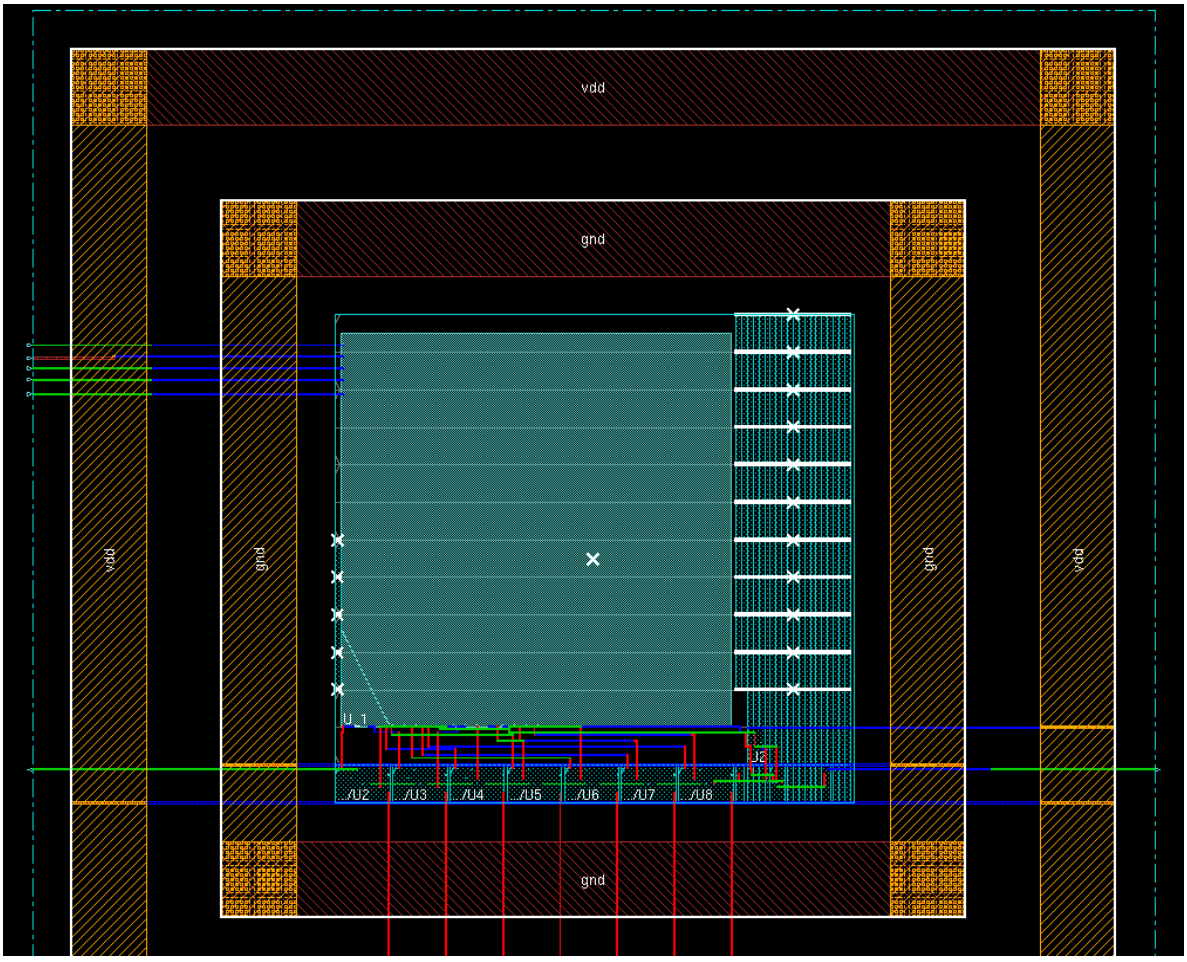
Click Place -> Physical Cell -> Add Filler...

Click "Select" and then select and add all the cells in the "Cells List" to "Selectable Cells List", then click OK.

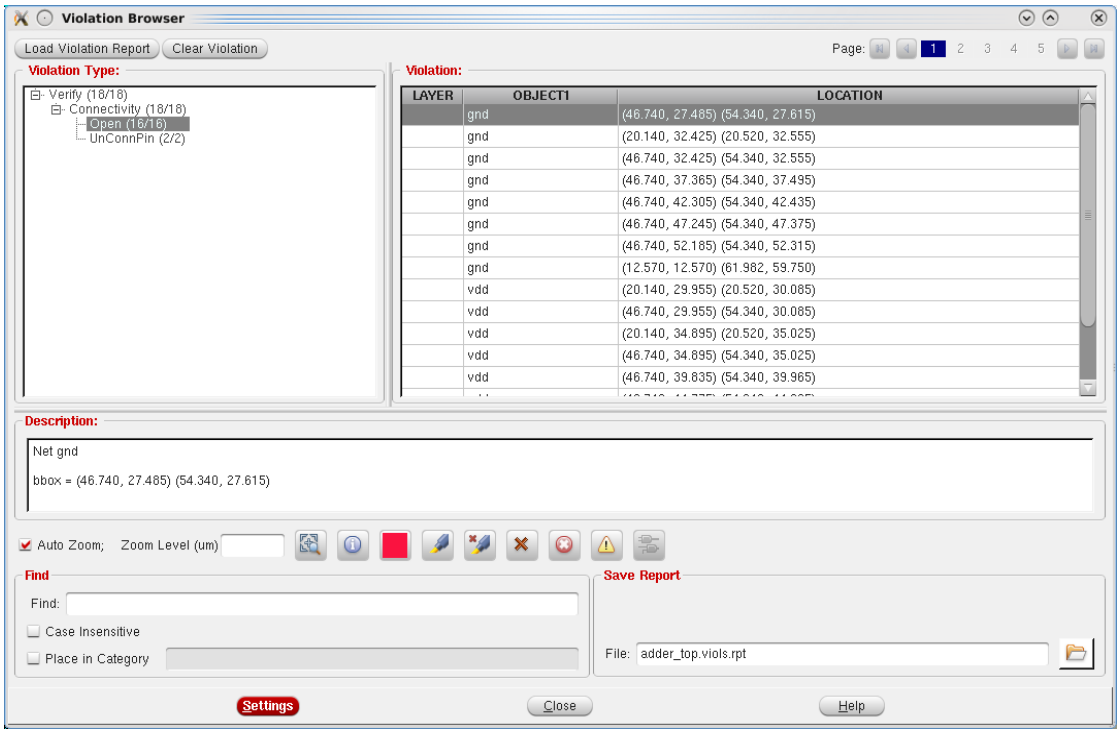
Once again, click Power -> Connect Global Nets...

Click Apply, then Close.

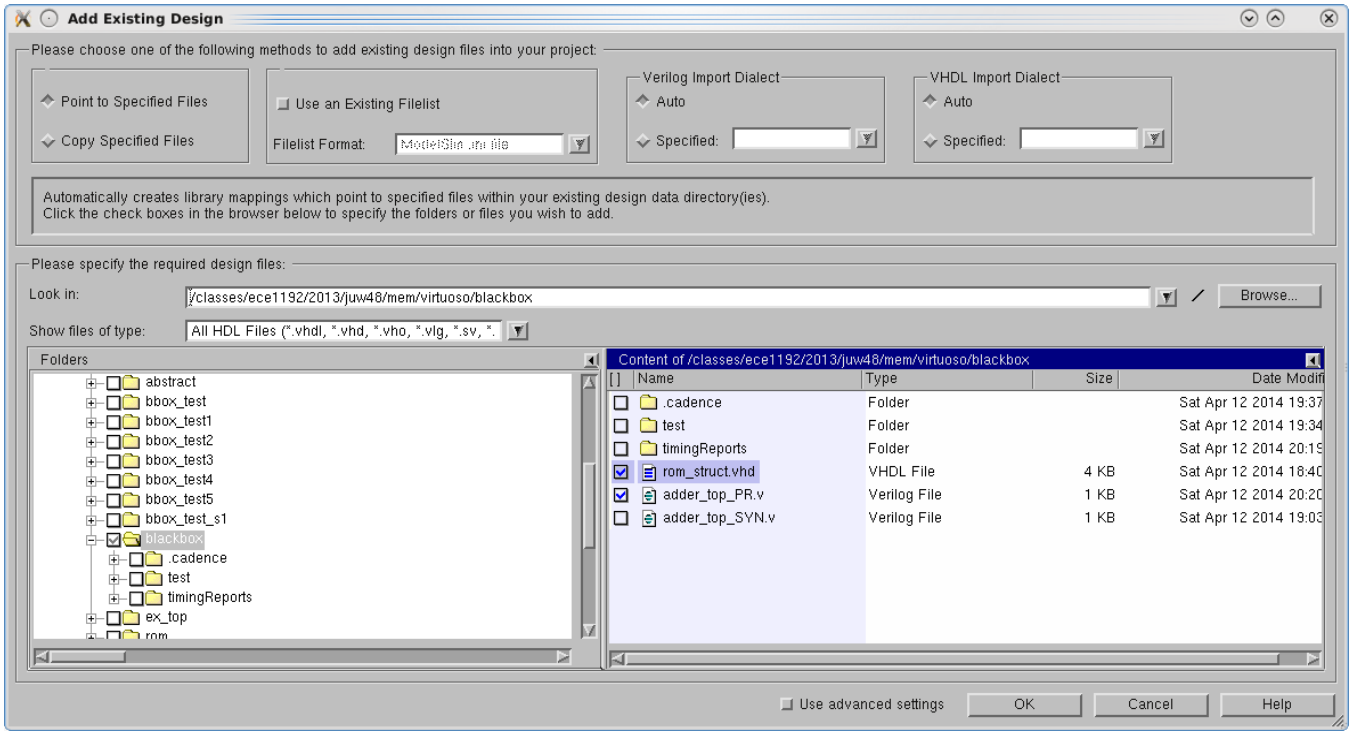
Well, you may find that new violations come up if you do geometry/connectivity verify. It may look like this.

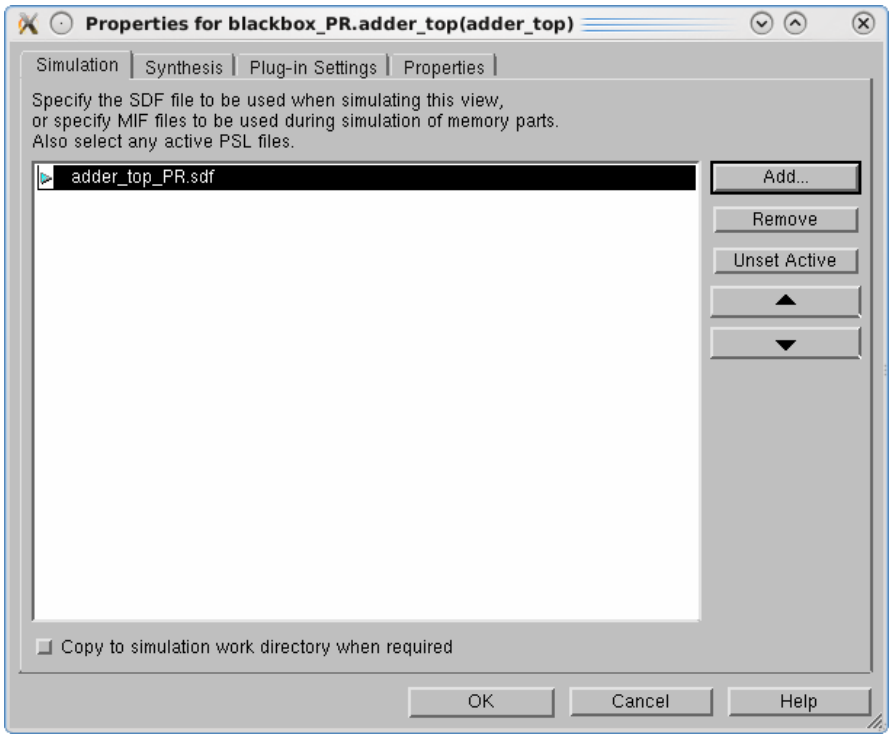


The geometry violation, or more specifically the spacing and short violations either at the edges or under the wires, you can eliminate them simply by deleting the related filler cells. However the connectivity violation is because the tool assumes that you should wire the power rails. Functionally, it's already acceptable. So you can just go ahead and use the netlist and sdf files generated for simulation.



6. Next step, you can do after Place & Route Simulation the same way we did after synthesize. Add both add_top_PR.v and rom_struct.vhd back to HDL. Compare with the results.




Properties for blackbox_PR.adder_top(adder_top)
Simulation
Synthesis
Plug-in Settings
Properties
Specify the SDF file to be used when simulating this view,
or specify MIF files to be used during simulation of memory parts.
Also select any active PSL files.
adder_top_PR.sdf
Add...
Remove
Unset Active
Copy to simulation work directory when required
OK
Cancel
Help