Tutorial 2流程：

打开DC， HDL替换.v文件，复制一份testbench file（主文件和testbench）在simu/run\_s下

1. 建立文件夹(workdic)： >perl ./PAD\_Flow.pl –op setup
2. 生成 pre place and route netlist(./SYNTH/run\_f/ **module\_final.v**):

**改：read.tcl, setup.tcl**

(/SYNTH/run\_s) >source synth.tcl

1. 在网标文件moudel\_final.v的基础上生成./PR/run\_f/**module.spef**以及./PR/run\_f/**module\_routed.v**:

**改：PAD\_Flow.pl, routed\_auto.tcl**

(workdic) >source routed\_auto.tcl

1. 生成./SIMULATION/run\_s/**module.vcd** 以及./SIMULATION/run\_s/**module.saif:**
2. 在testfixture.v中加入：

Initial begin

$dumpfile(“module.vcd”);

$dumpvars; //which variable should be dumped

End

1. **在testfixture.v中最上面一行添加：//`timescale 1ns / 1ps**
2. (./SIMULATION/run\_s) >vlog ../../PR/run\_f/module\_routed.v
3. 编译库文件

(./SIMULATION/run\_s)

>vlog .../Nangate OpenCellLibrary\_PDKv1\_2\_v2008\_10\_typical\_conditional.v（路径需要正确即可）

1. (./SIMULATION/run\_s) >vlog testfixture.v
2. (./SIMULATION/run\_s) >vsim –c –novopt module\_test （或者vsim -voptargs=+acc Lmodeule\_test）

VSIM#>run 10000ns

VSIM#>exit

1. (./SIMULATION/run\_s) >vcd2saif –input module.vcd –instance module\_test/uut –output module.saif
2. 在**module.saif的基础上进行最后的功耗分析：**

**改：final\_power\_auto.tcl**

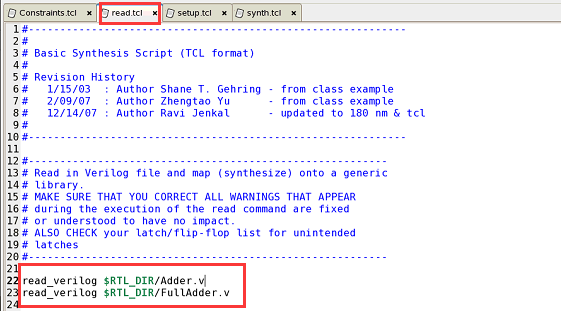
(workdir) >source final\_power\_auto.tcl

1. .rpt文件在./SYNTH/run\_f中

（dv打不开时先lmli2再dv）

Verilog工程包含多个.v文件时的改动：

第一步 (/SYNTH/run\_s) >source synth.tcl， 修改read.tcl读主程序+子程序，其他不变:



得到布局布线前的功耗改动：

1. 主程序.v文件里面要定义一个input clock，单独放在那里即可。testbench里面不需要给特定的clock输入。
2. 第一步不变，得到modulename\_final.v

不运行(workdic) >source routed\_auto.tcl

把modulename\_final.v复制到PR/run\_f文件夹下当做modulename\_routed.v（第四步用到）

第三步编译modulename\_final.v得到saif文件

执行第四步：(workdic) >source final\_power\_auto.tcl

1. 修改Synthesis步骤中SYNTH/run\_f中的CompileAnalyze.tcl，添加了report\_area命令，因为源flow中是在routed\_auto中执行的。