集成电路设计实习 VLSI Design Labs

单元实验四

数字系统设计-后端

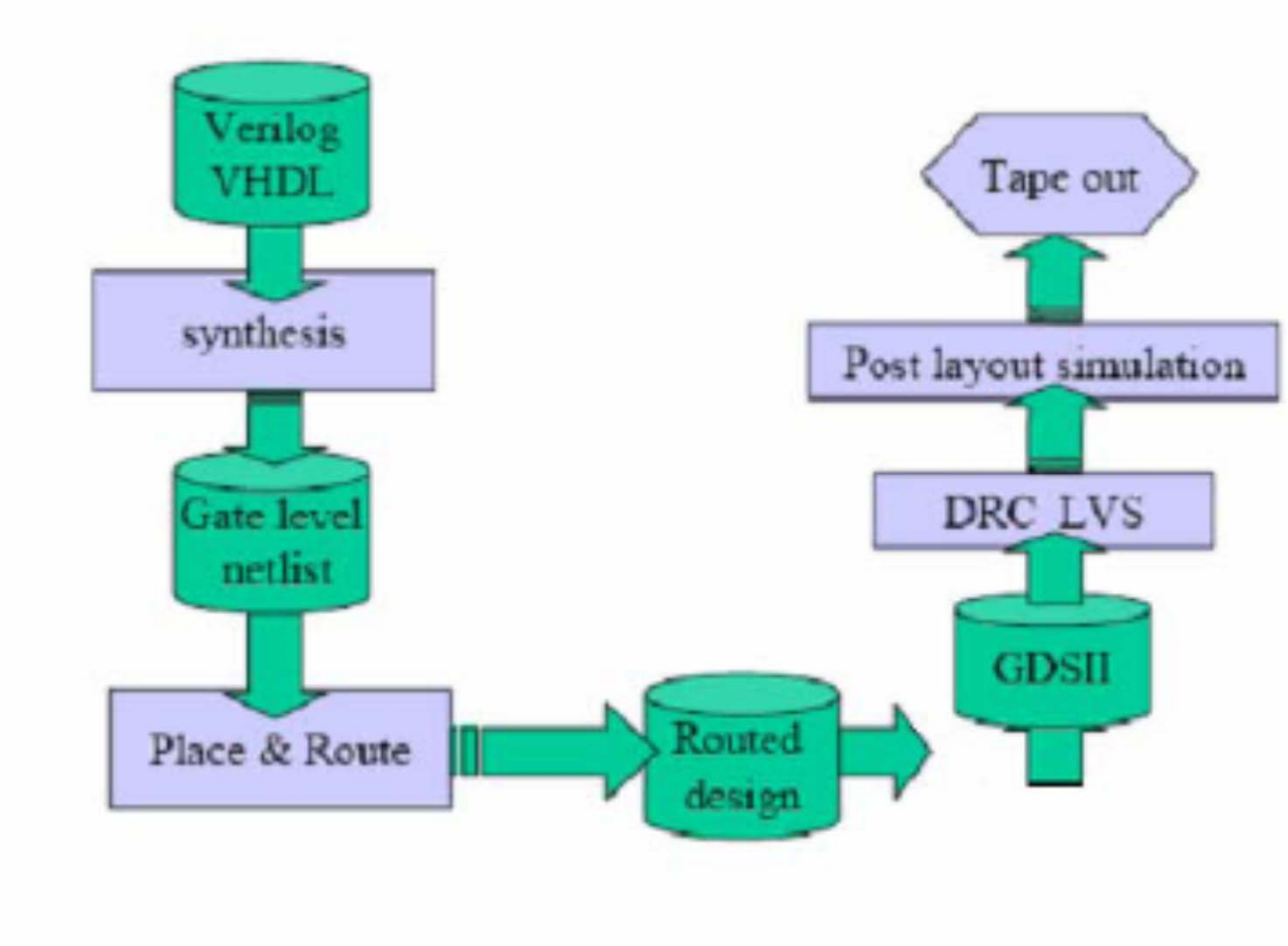
Institute of Microelectronics Peking University

实验目的及时间安排

- 掌握半定制的后端设计流程
- 学习Cadence自动布局布线工具SE
- 完成自动布局布线
- 设计时间：1次课

Cell-based ASIC 设计流程

- 基于标准单元的半定制设计流程



后端设计过程

- 后端设计完成基于标准单元的自动布局布线工作，利用cadence的SoC encounter工具
- 在前端设计（逻辑仿真和逻辑综合）完成后，得到综合器输出的映射到标准单元库的门级网表
- 将门级网表读入到自动布局布线工具中，进行版图规划floorplan(即对所有的标准单元进行布局place)和布线route(即按照网表中的连接关系对摆放好的标准单元的版图进行金属连线)，最后生成物理版图

布局布线前的准备

- 进行自动布局布线，需准备下列文件：
  - ✓ **Verilog Netlist (.v)**
  - ✓ **Timing Library File (.lib)**
  - ✓ **Library Exchange Format (.lef)**
  - ✓ **Timing Constraint File (.sdc)**
  - ✓ **IO Assigment File (.io)**

文件说明——netlist.v & *.lib

- **Verilog Netlist (.v)**
  - ✓ 综合后生成的门级网表
- **Timing Library files(.lib)**
  - ✓ Max Timing Libraries: for setup time check
  - ✓ Min Timing Libraries: for hold time check

文件说明——LEF & SDC

- **Library Exchange Format (.lef)**
  - ✓ LEF库描述标准单元库的工艺特性及单元(cell)数据
  - ✓ 描述用于宏单元互连的各布线层的详细信息
  - ✓ 布线器要用的via单元
  - ✓ core和pad的位置定义
- **Timing Constraint File (.sdc)**
  - ✓ SDC格式的设计约束文件
- **IO Assigment File (.io)**
  - ✓ 该文件用于指定I/O PAD的顺序

准备工作1——修改netlist.v文件

- 在网表netlist.v中插入I/O的PAD，输入采用PI(PAD,C)，输出采用PO2(PAD,I)，PI和PO2都是PAD的单元库
- 加入电源地PAD，给core供电 PVDD1(VDD), PVSS1(VSS)
- 具体做法是给netlist.v（打开看看）中的最下面的综合顶层模块adder包裹一圈PAD，即重新写一个adder_PAD模块，作为新的顶层模块，实例化调用adder模块：

```
module adder_PAD(Data_a_PAD, Data_b_PAD, Cin_PAD, Data_out_PAD, Cout_PAD);
  input [15:0] Data_a_PAD, Data_b_PAD;
  input Cin_PAD;
  output [15:0] Data_out_PAD;
  output Cout_PAD;

  wire [15:0] Data_a, Data_b;
  wire Cin, Cout;
  wire [15:0] Data_out;
  wire VDD, VSS
```

准备工作1——修改netlist.v文件

```
adder U_adder(Data_a, Data_b, Cin, Data_out, Cout);

PUDD1 U_PUDD1(.VDD(VDD));
PUSS1 U_PUSS1(.VSS(VSS));

PI U_PI1(.PAD(Cin_PAD), .C(Cin));
PI U_PI2(.PAD(Data_a_PAD[0]), .C(Data_a[0]));
PI U_PI3(.PAD(Data_a_PAD[1]), .C(Data_a[1]));
PI U_PI4(.PAD(Data_a_PAD[2]), .C(Data_a[2]));
```

- 依照上面的例子将adder_PAD模块补充完整
- 由于时间关系，本次实验不加PAD，但大实验时要求加PAD!

准备工作2——修改constraints.sdc文件

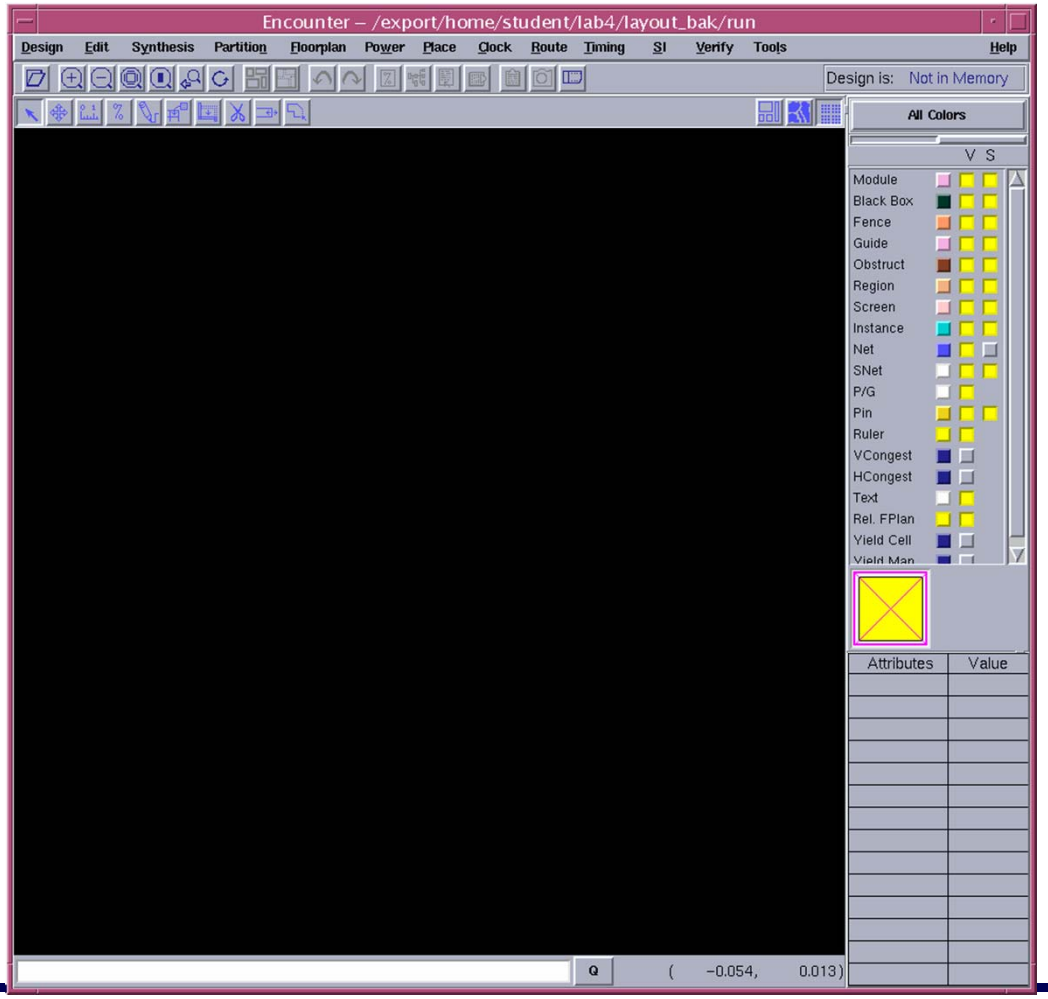
- 由于修改了netlist.v文件，顶层文件变成了adder_PAD，所以constraints.sdc文件也要做相应修改，如下：
- current_design 由adder改为adder_PAD
- 设置端口delay的命令中get_ports全部改为adder_PAD中的端口，如图：

```
set_input_delay -clock [get_clocks clk] -add_delay 5.0 [get_ports Cin_PAD]
set_input_delay -clock [get_clocks clk] -add_delay 5.0 [get_ports {Data_a_PAD[0]}]
set_input_delay -clock [get_clocks clk] -add_delay 5.0 [get_ports {Data_a_PAD[1]}]
set_input_delay -clock [get_clocks clk] -add_delay 5.0 [get_ports {Data_a_PAD[2]}]
set_input_delay -clock [get_clocks clk] -add_delay 5.0 [get_ports {Data_a_PAD[3]}]
set_input_delay -clock [get_clocks clk] -add_delay 5.0 [get_ports {Data_a_PAD[4]}]
set_input_delay -clock [get_clocks clk] -add_delay 5.0 [get_ports {Data_a_PAD[5]}]
set_input_delay -clock [get_clocks clk] -add_delay 5.0 [get_ports {Data_a_PAD[6]}]
set_input_delay -clock [get_clocks clk] -add_delay 5.0 [get_ports {Data_a_PAD[7]}]
```

- 本次实验没有要求加PAD，因此这里也不用修改！

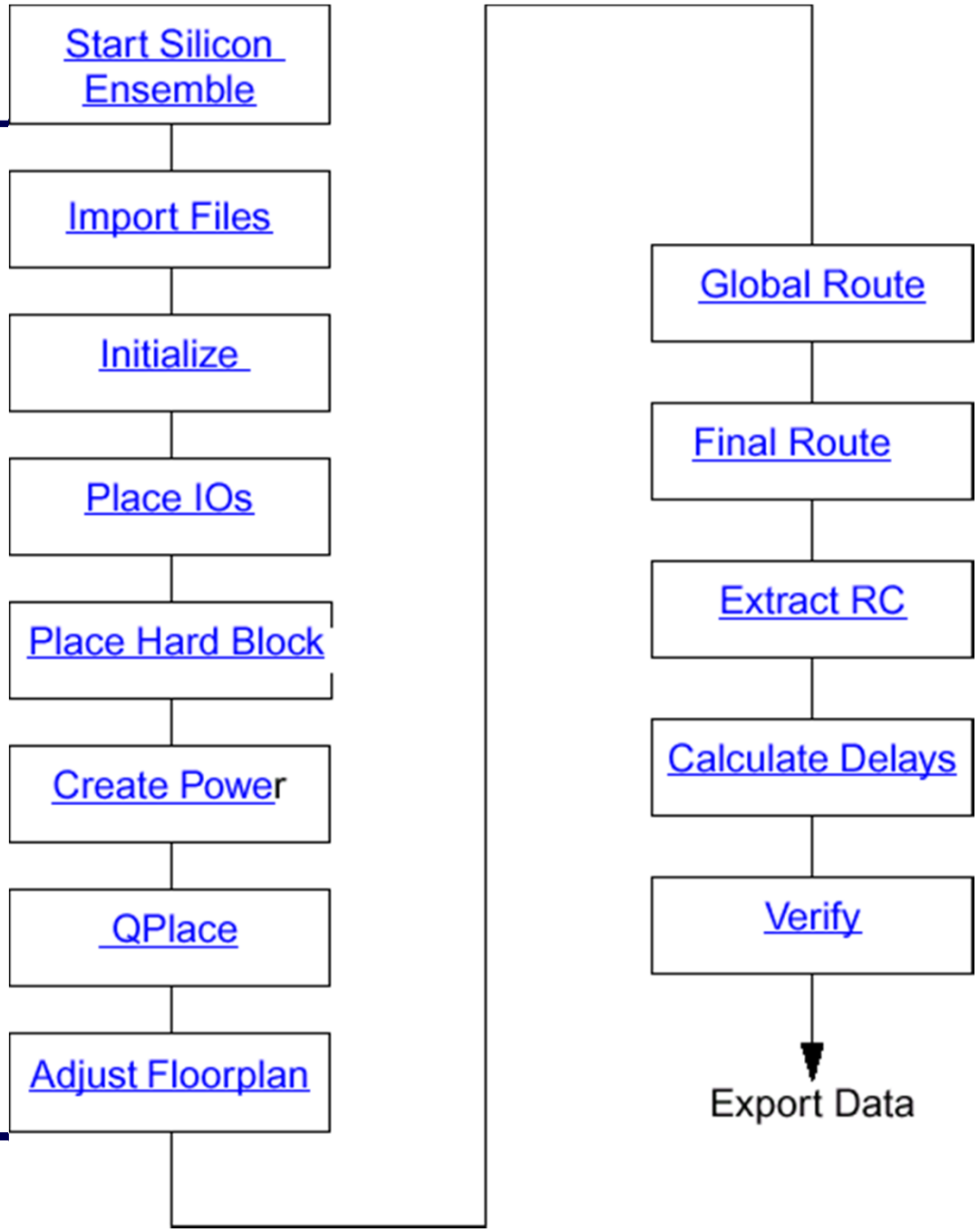
启动SE界面

- 在根目录下键入>soc52，进入相应目录下，键入> encounter 启动SoC encounter工具图形界面



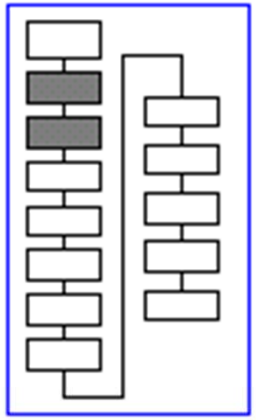
设计流程

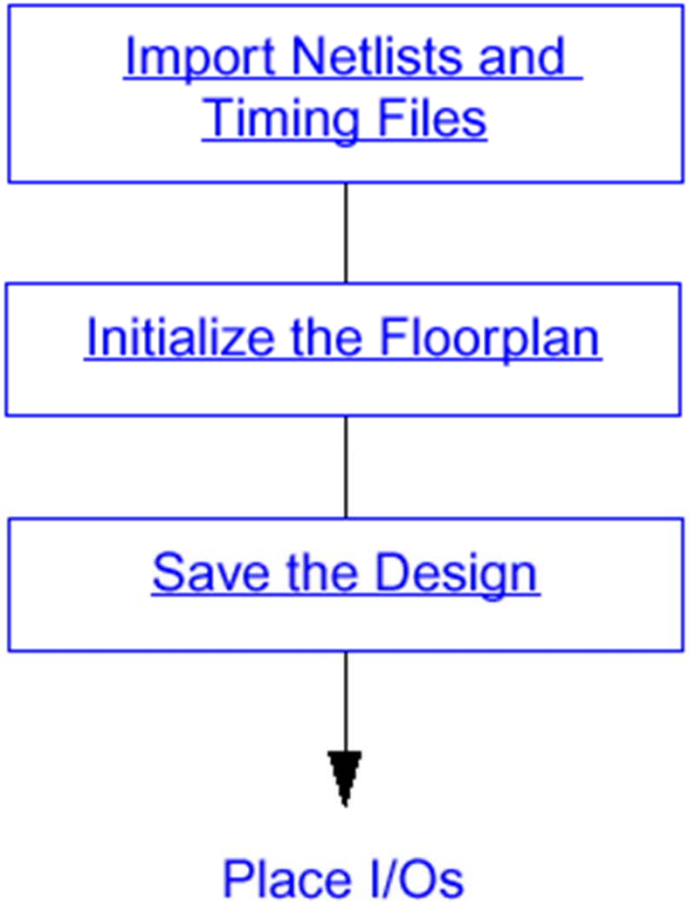
SE是一个基于时序的 floorplan, place、route工具



1、建立设计数据库

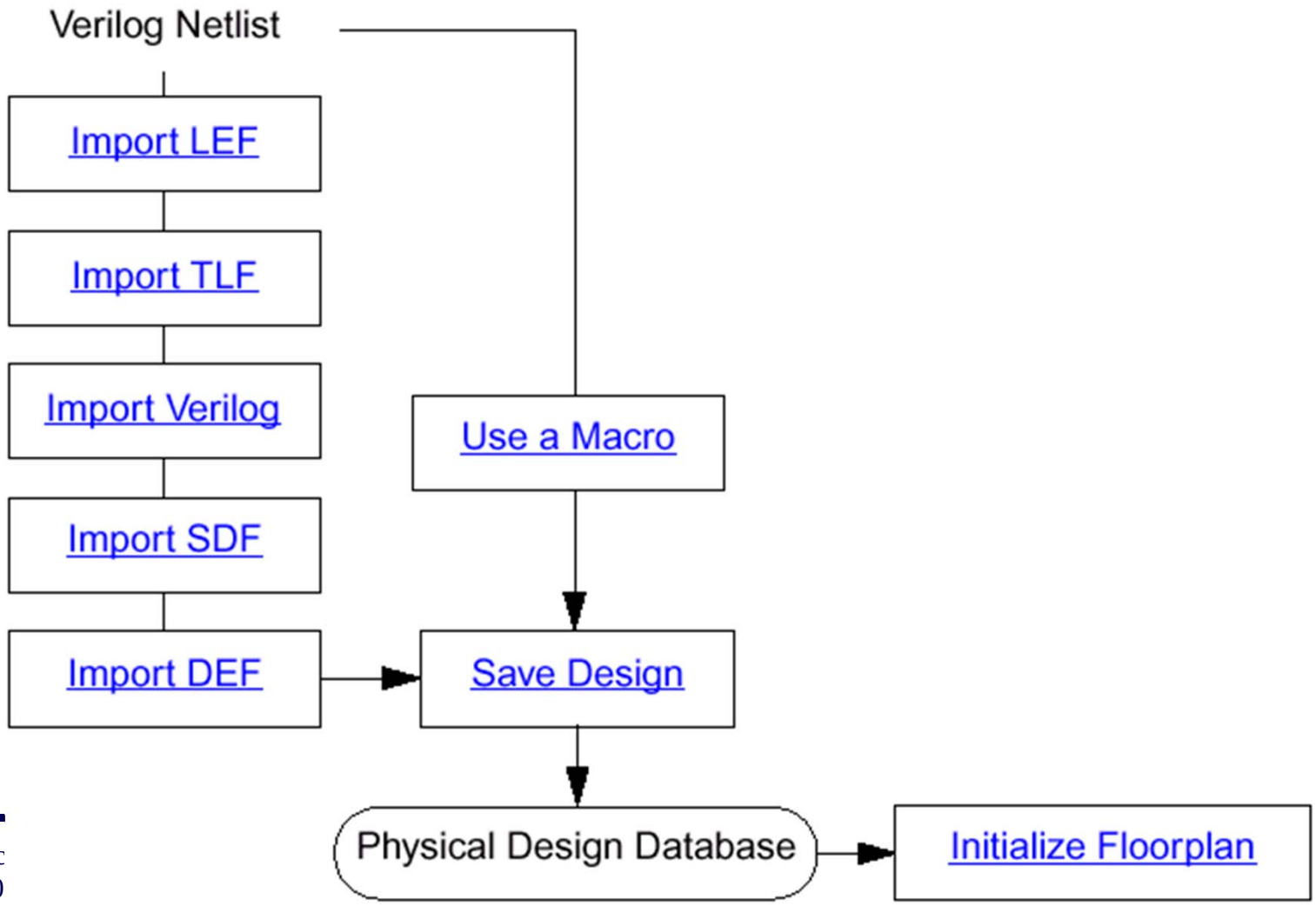
- 输入数据并对设计进行初始化





导入网表及时序文件

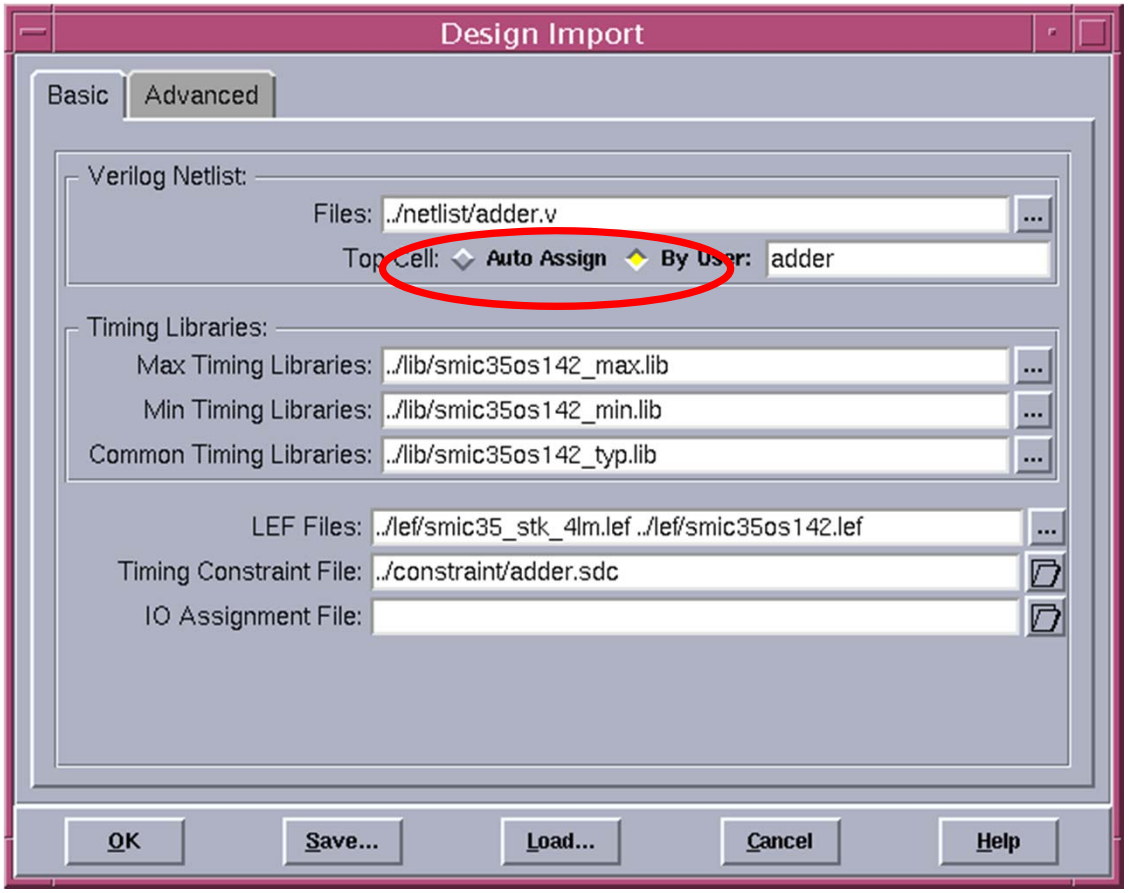
- 建立物理数据库：导入库，网表，以及时序信息



导入LEF文件

- 选择菜单： ***Design ->design import***

在**Basic**目录下的相应位置导入**netlist**，**lib**，**lef**以及**sdc**文件。各个文件的路径在前面已经给出。



注意：

1）顶层模块的实例化名称采用**Auto Assign**。

2）添加**lef**文件的时候先添加**s**mic35_stk_4lm.lef文件。后添加单元文件smic35os142.lef

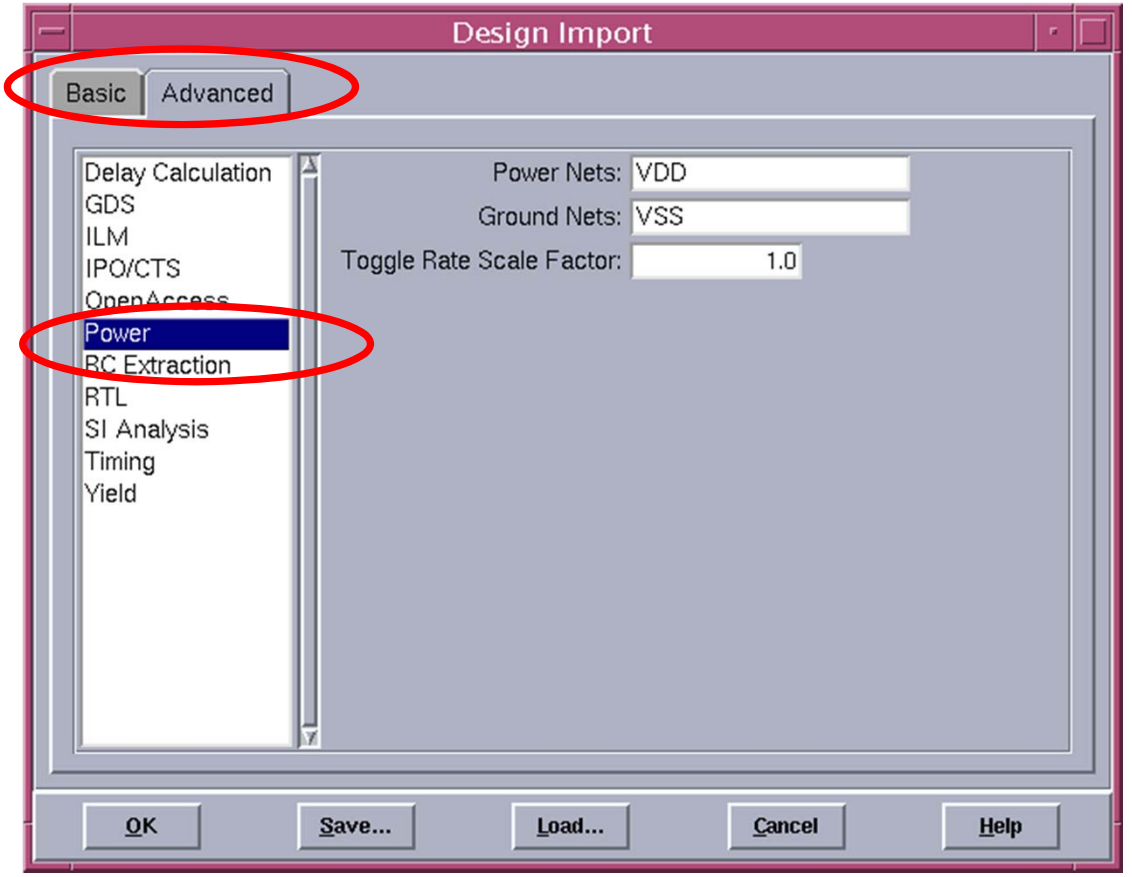
3）如果加了**PAD**，那么相应**IO**文件夹里面的**io**库也要加入在相应位置，做大实验时要注意！

导入LEF文件

- 选择菜单： ***Design ->design import***

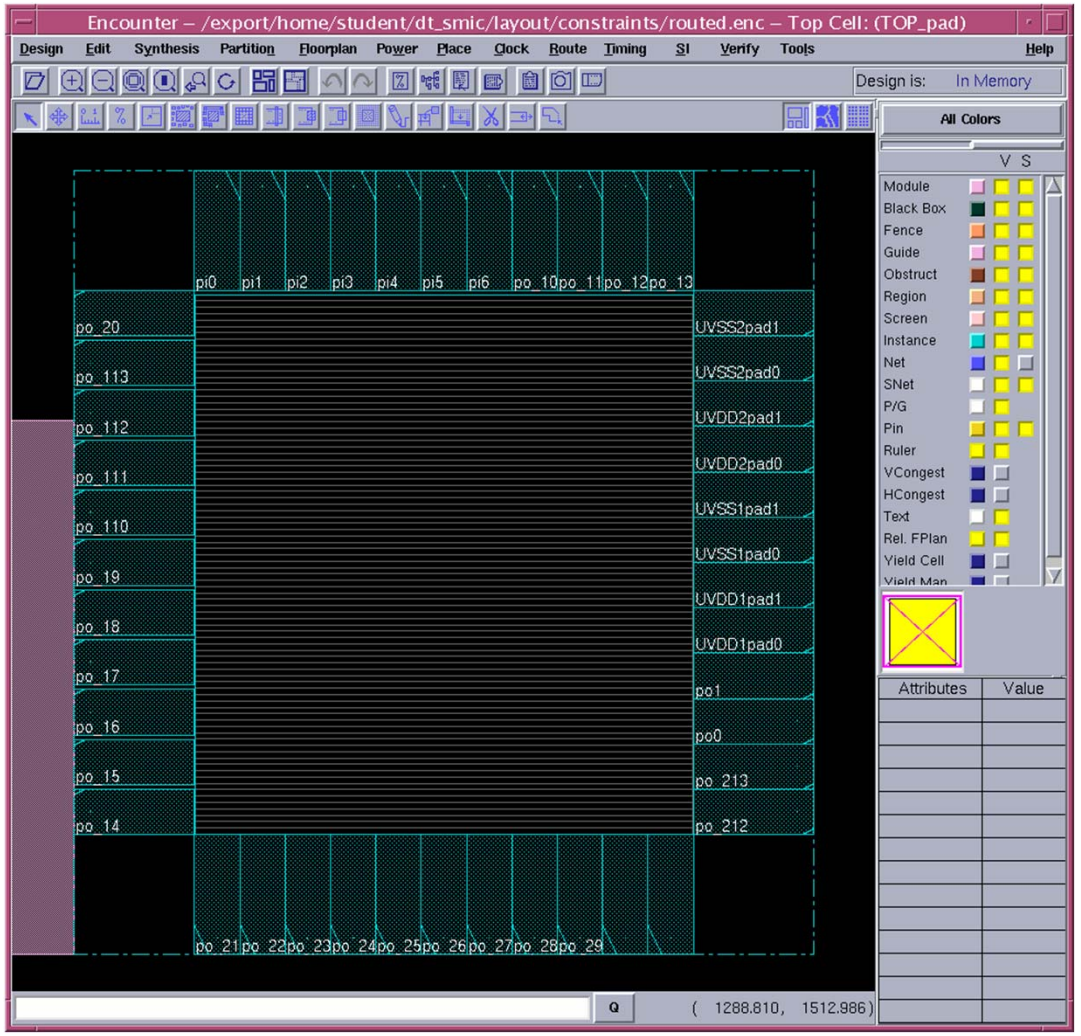
注意：

3）点击左上角的Advanced按键，在Power选项中设置PowerNet以及GroundNet的名称分别为VDD和VSS。



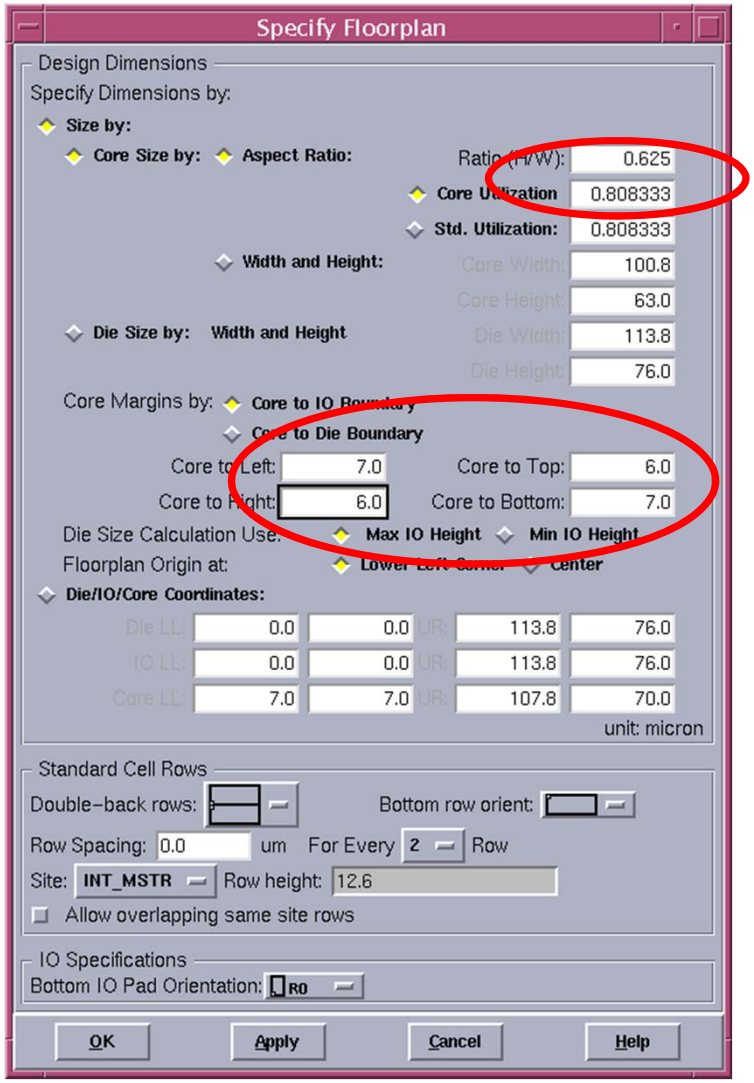
配置保存

- 点击上图中save按键，保存当前导入的配置。将当前配置存在constraint目录下。配置文件的后缀名为.conf。
- 保存是必要的，如果后面的步骤有错误，可以重新打开保存的数据，避免重复劳动。点击OK，出现下图结果
- 右图是加了PAD的（外围蓝绿色一圈），没加则没有



2、版图规划FloorPlan——设计初始化

- 选择菜单：
- ***Floorplan –> Specify Floorplan***
- 不放置任何单元，只控制芯片的大小、长宽比率，建立横向或纵向的**core和I/O row**

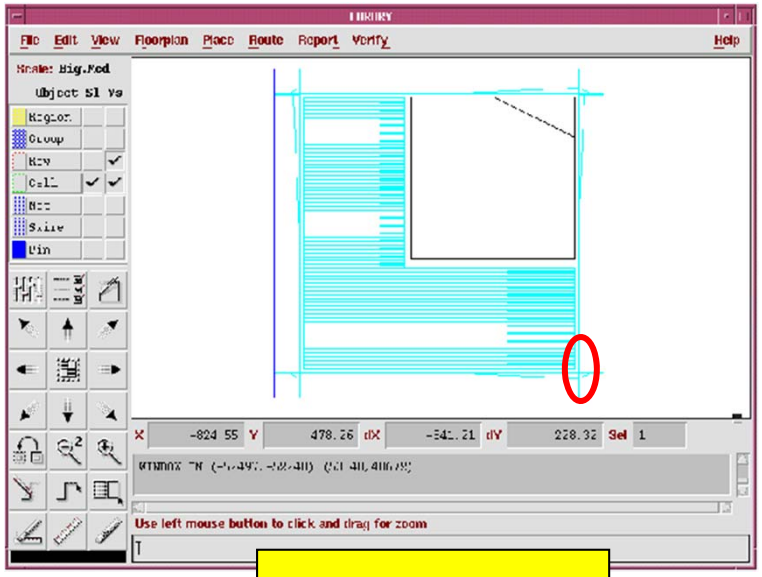


FP设置

1. **Design Dimensions：控制设计芯片的大小，可以指定：**
   - **aspect ratio: 设计 宽度/高度 比例**
   - **Core Utilization：芯片面积利用率**
   - **Core height：设置高度，由FP计算宽度**
   - **Core width：设置宽度，由FP计算高度**
   1. **Core to IO Boundary：设置core和IO row的距离**

- **Standard Cell Rows：控制core的面积**
  - **Double-back rows：取向设置**
  - **Bottom row orient：底部row取向设置**
  - **Row Spacing：row间距**
  - **Row height：row的高度**

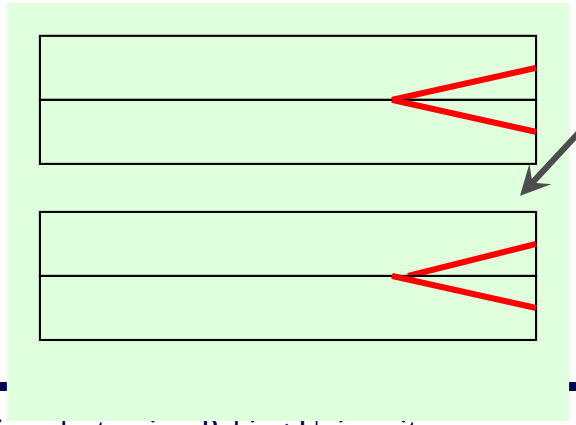
Flip Every Other Row



Row Spacing:

3 Tracks

Double-back Rows



For Every 1 Row

FP设置

- Die Size Constraint选Aspect Ratio，初始设置使宽长比为 1。
- Core Utilization 填入：0.8
- core to IO Boundary：填入10，10，10，10
- 点击Apply，Fp会计算得到真实的参数值，相应设置的参数会有小的变化。这是根据标准单元的高度以及用户的设计得到的最接近设定的布局方案。
- 点击OK载入数据。

Floorplan结果

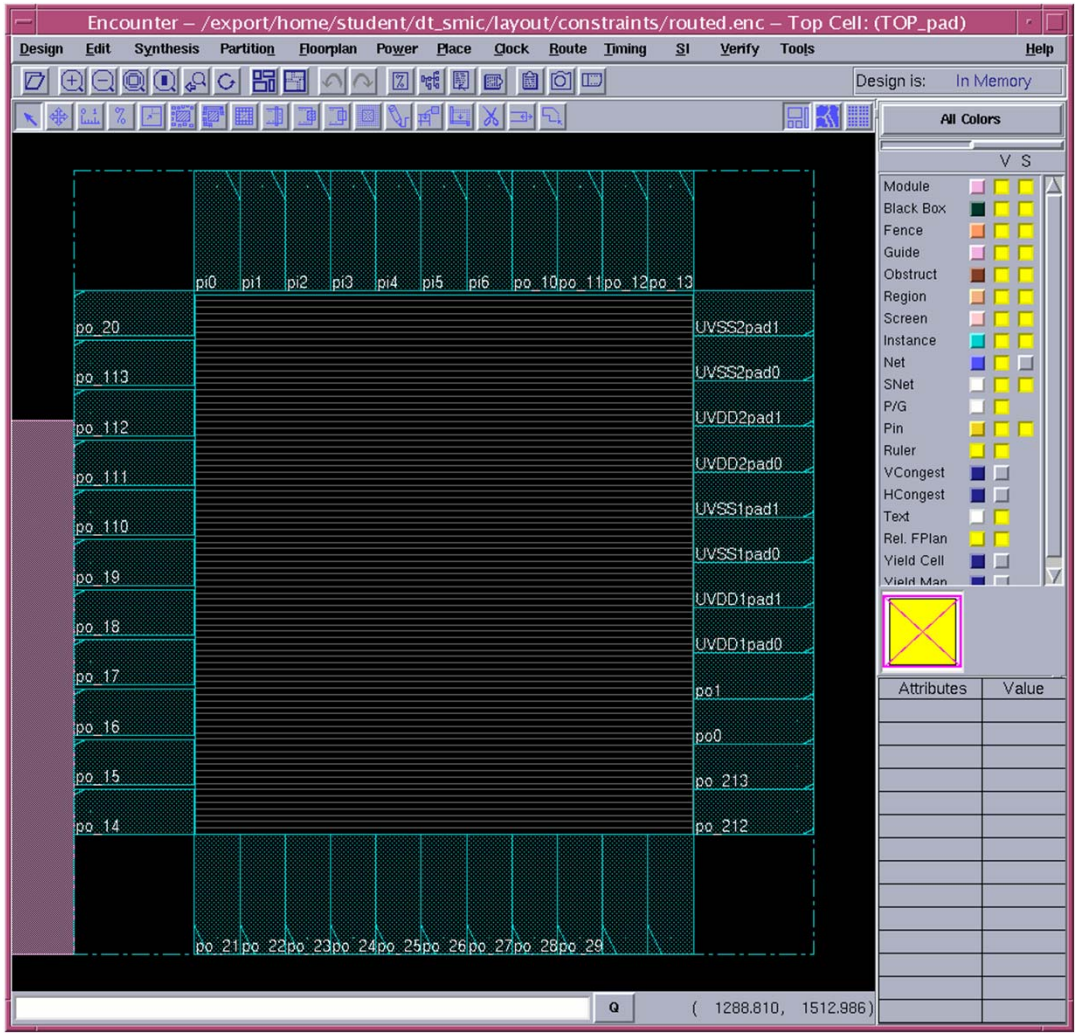
- 保存设计：选择菜单

  Design->save design

  保存在constraint目录下，floorPlan.enc

- 中间部分为芯片区域

- 之后每做完一步都需要保存相应的设计。这样方便发现问题后回溯。



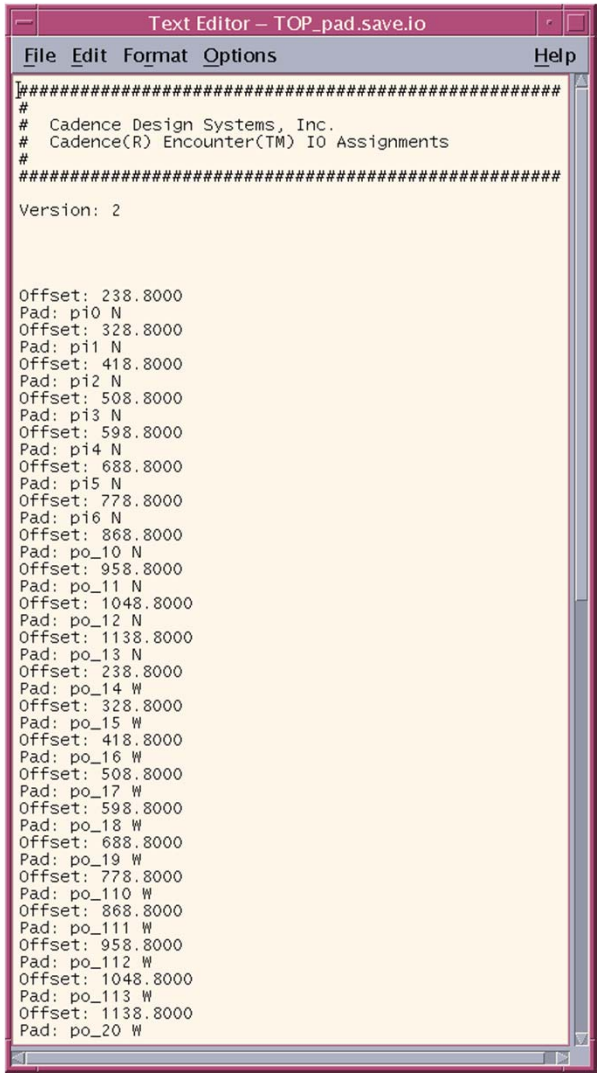
调整IO位置以及插入corner以及filler

- 保存设计：选择菜单

  Design->save->I/O File

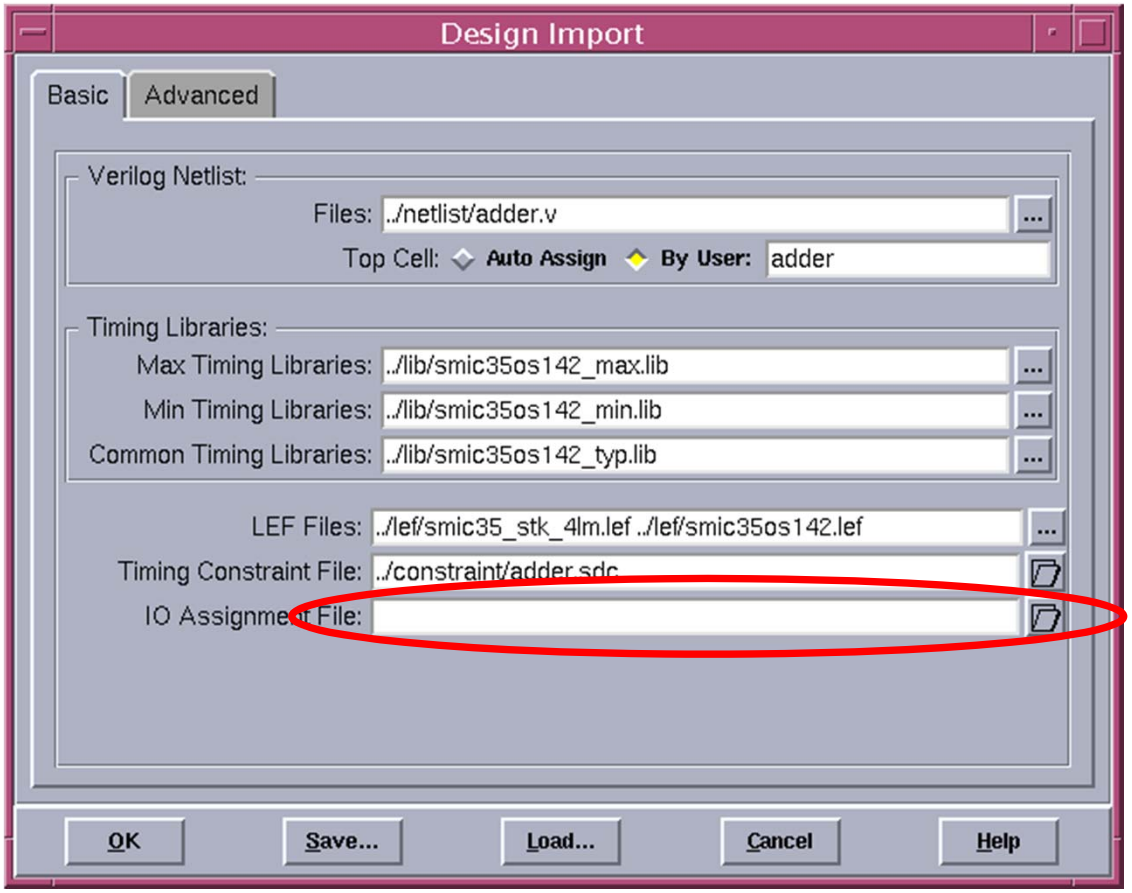
  得到一个原始的IO位置定义文件 adder_PAD.save.io

- 修改得到的adder_PAD.save.io文件来修改IO的摆放位置。
- 没加PAD不用做此步骤!



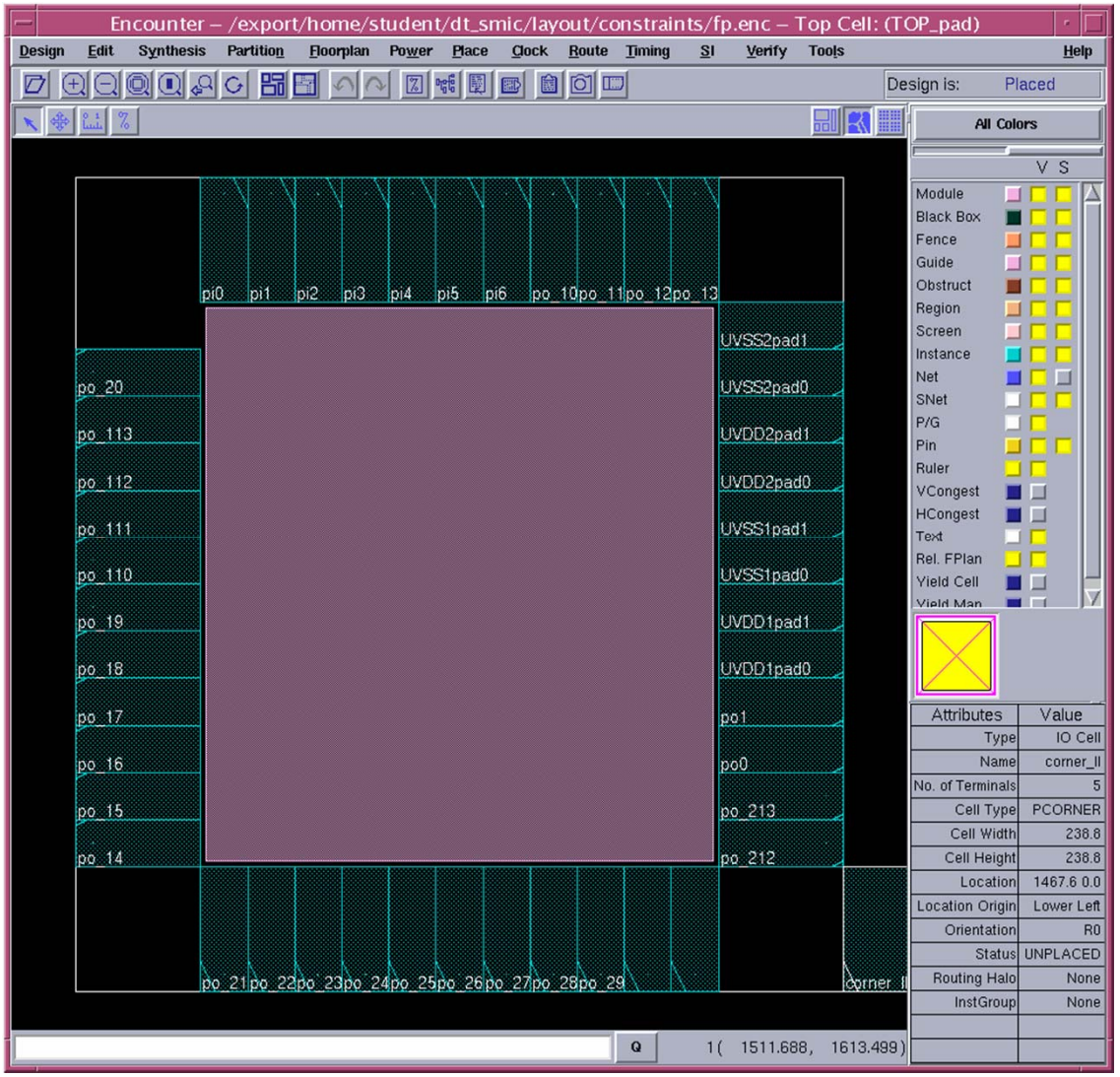
调整IO位置以及插入corner以及filler

- 控制台中输入freeDesign命令清空当前的输入。
- 重新导入设计，这次加上IO Assignment File。
- 这样就可以得到修改ioPAD位置后的版图
- 没加PAD不用做此步骤！



调整IO位置以及插入corner以及filler

- 得到新的core的布局



调整IO位置以及插入corner以及filler

- 插入corner
- 采用的corner的名称为PCORNER
- 在命令行中采用addInst 以及 placeInstance，命令来创建以及摆放corner

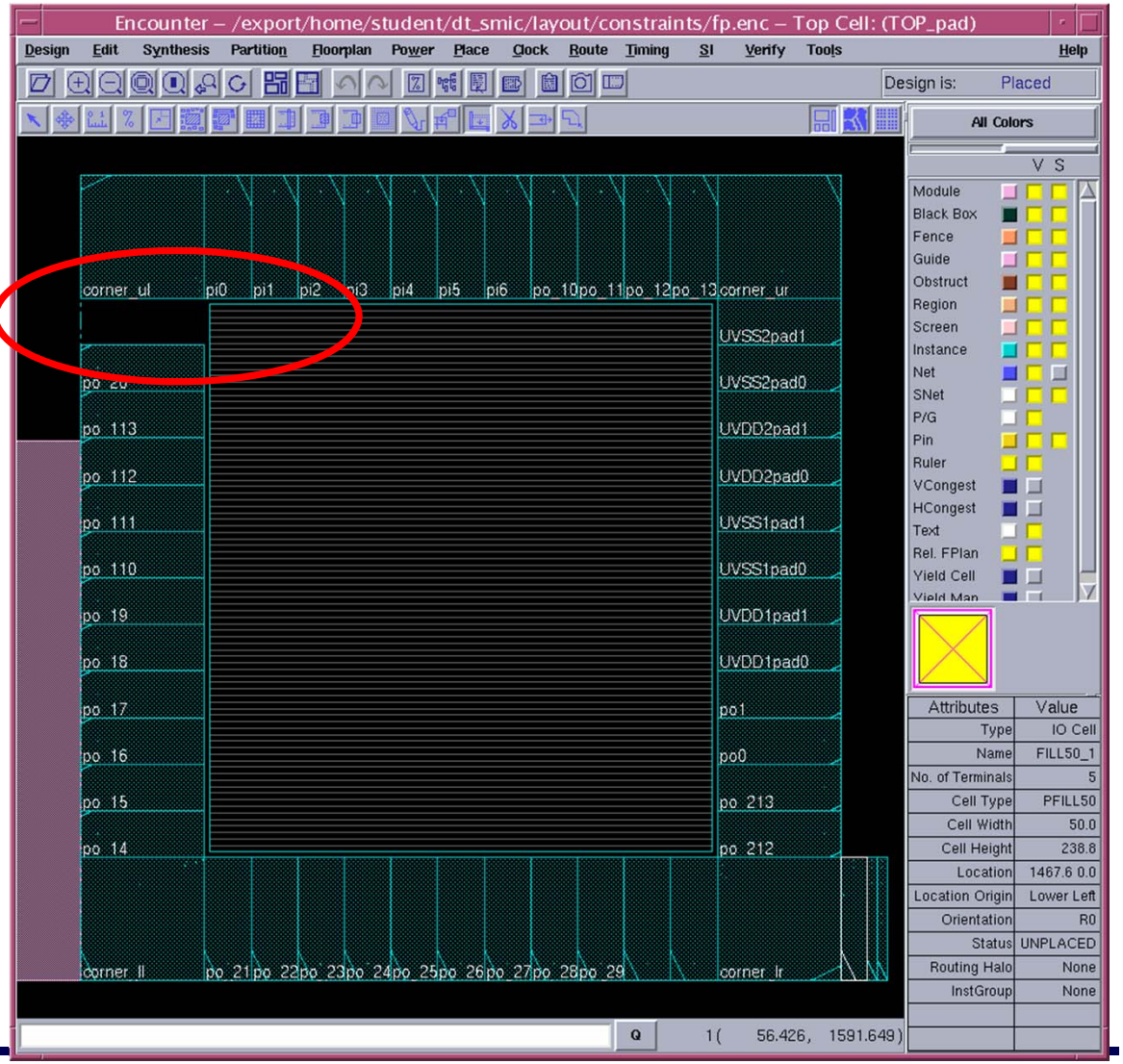
- addInst –cell <模块名称（PCORNER）> –inst <实例化名称>
- placeInstance <实例化名称> <x坐标> <y坐标> [旋转角度]

- addInst –cell PCORNER –inst corner_ll
- placeInstance corner_ll 0 0 R0

没加PAD不用插入corner和filler！

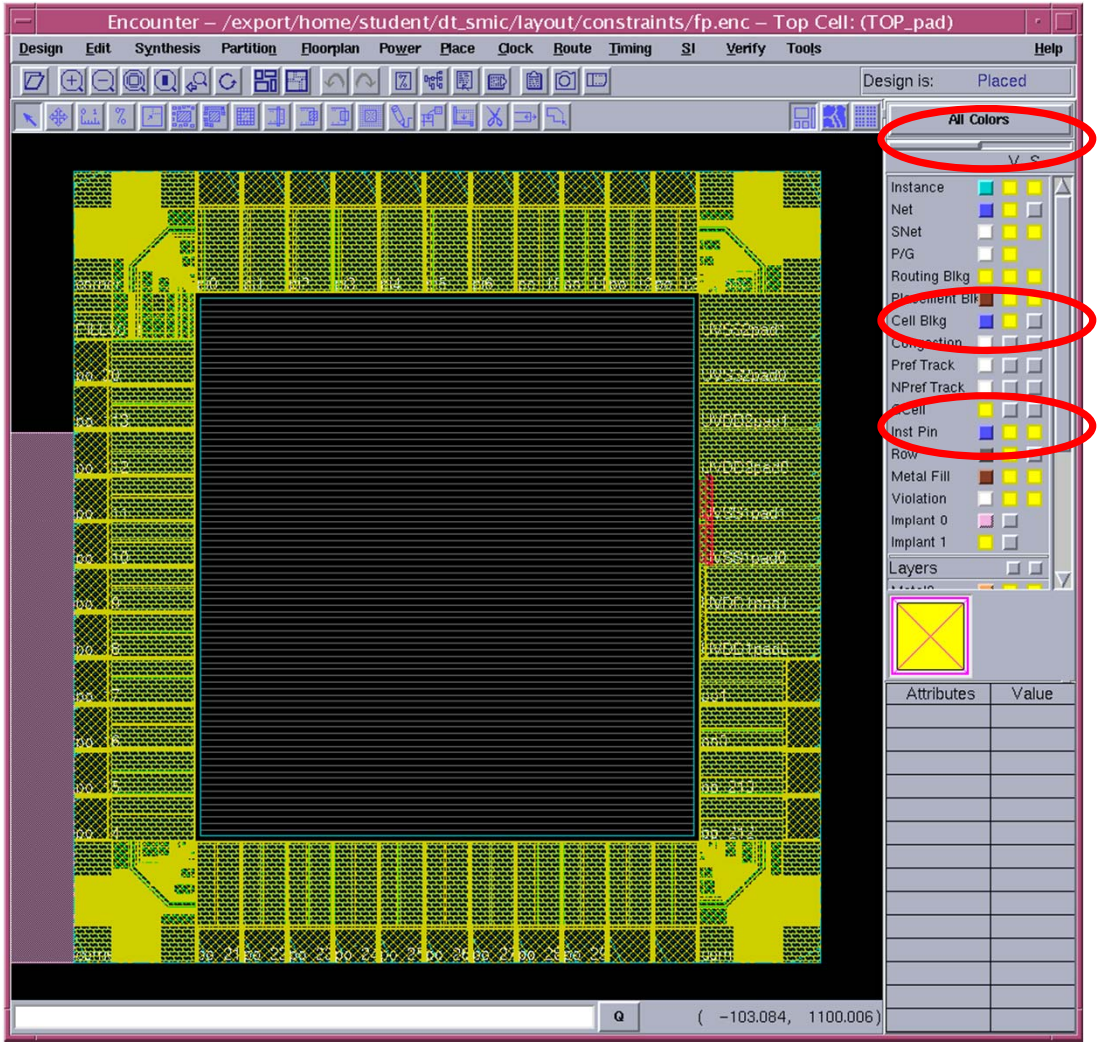
调整IO位置以及插入corner以及filler

- 得到插入CORNER的布局图,要调整CORNER的方向使得其中的金属连线位置连成一个环路。
- 点击右边侧栏的cell blkg 以及inst Pin可以看到pad的金属层
- 注意圈出的地方存在Pad间的空隙需要插入filler填补这些空隙



调整IO位置以及插入corner以及filler

- 插入FILLER
- 在I/O的PAD空隙处插入FILLER使得芯片周边一圈PAD都紧密相连形成一个供电回路。
- 插入的方法与插入CORNER的方法相同。采用模型可以参看lef库文件，其中有不同宽度的FILLER。根据设计自行选择:
  - PFILLER50 宽度50nm
  - PFILLER20 宽度20nm
  - ……



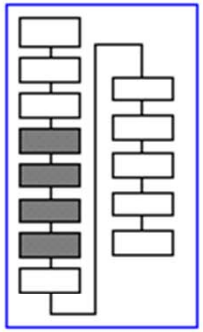
4、布局Place

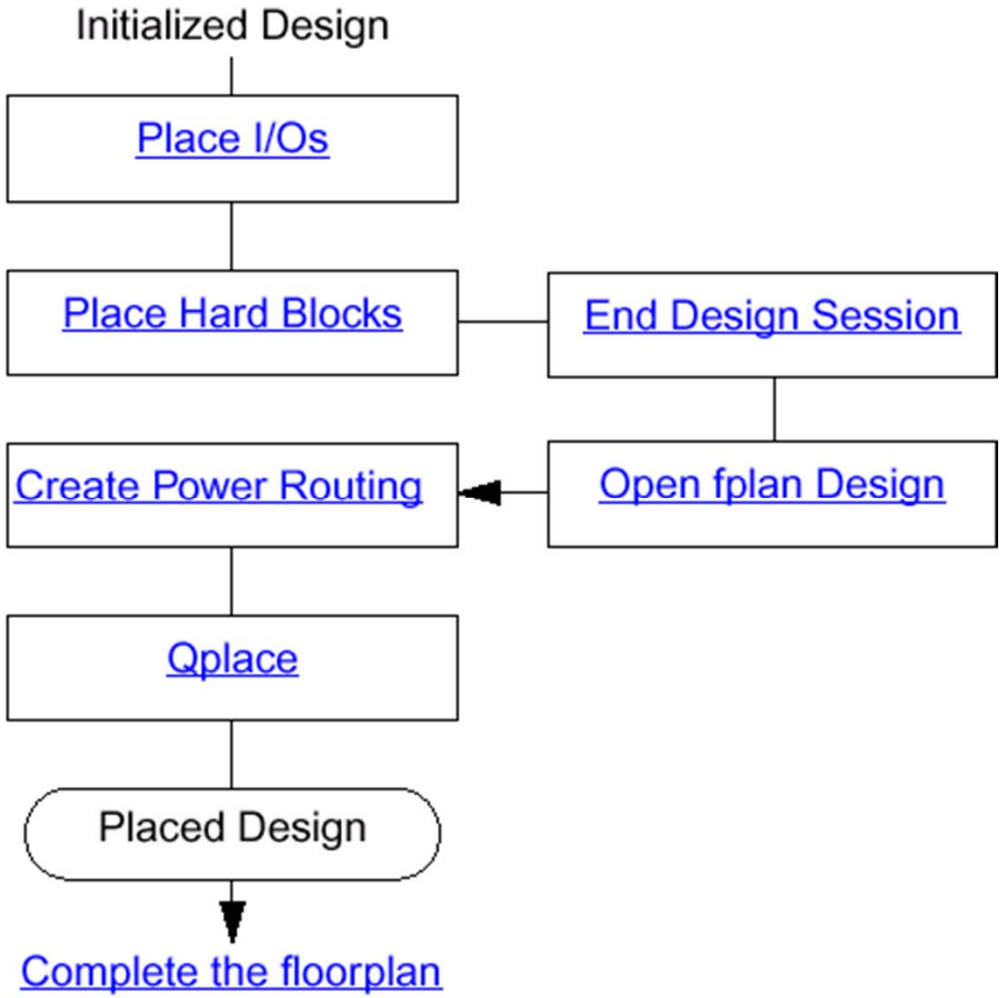
- create power routing & place instances

- 版图规划Floorplan确定了电源线的区域和标准单元布局的区域，之后进行单元的布局

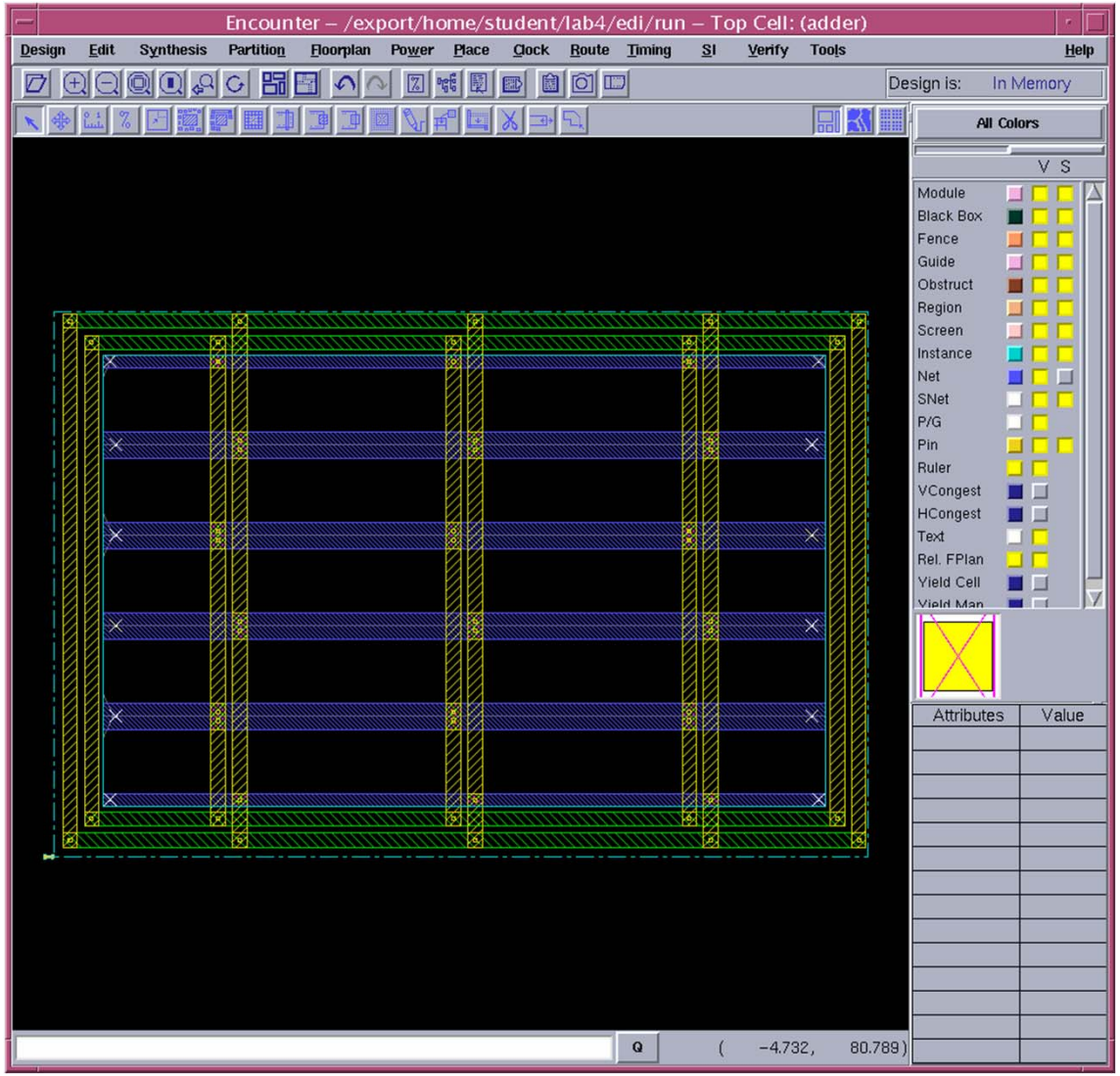
- 布局过程：

Add ring/strips —>
Connect ring—>
Place cells —>
Place filler



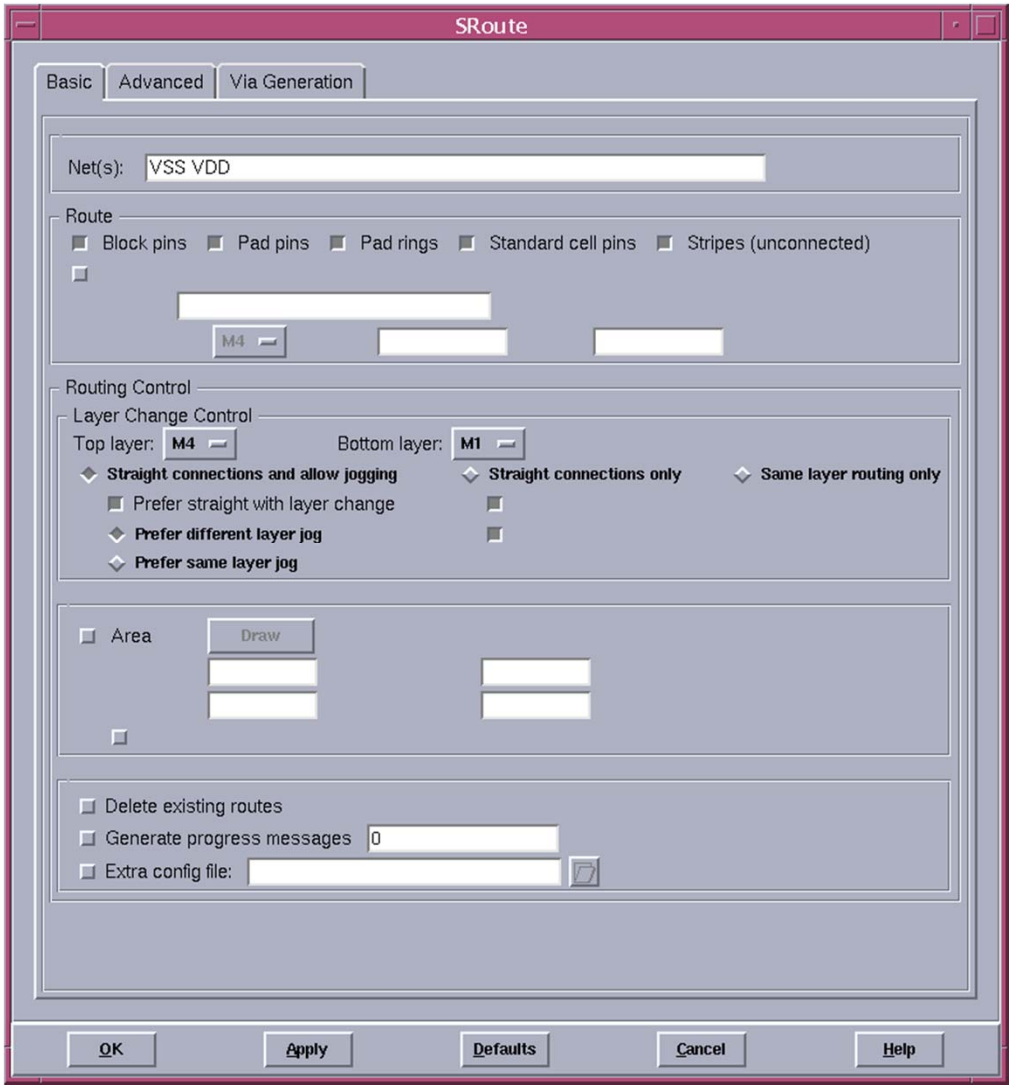


Power Planning



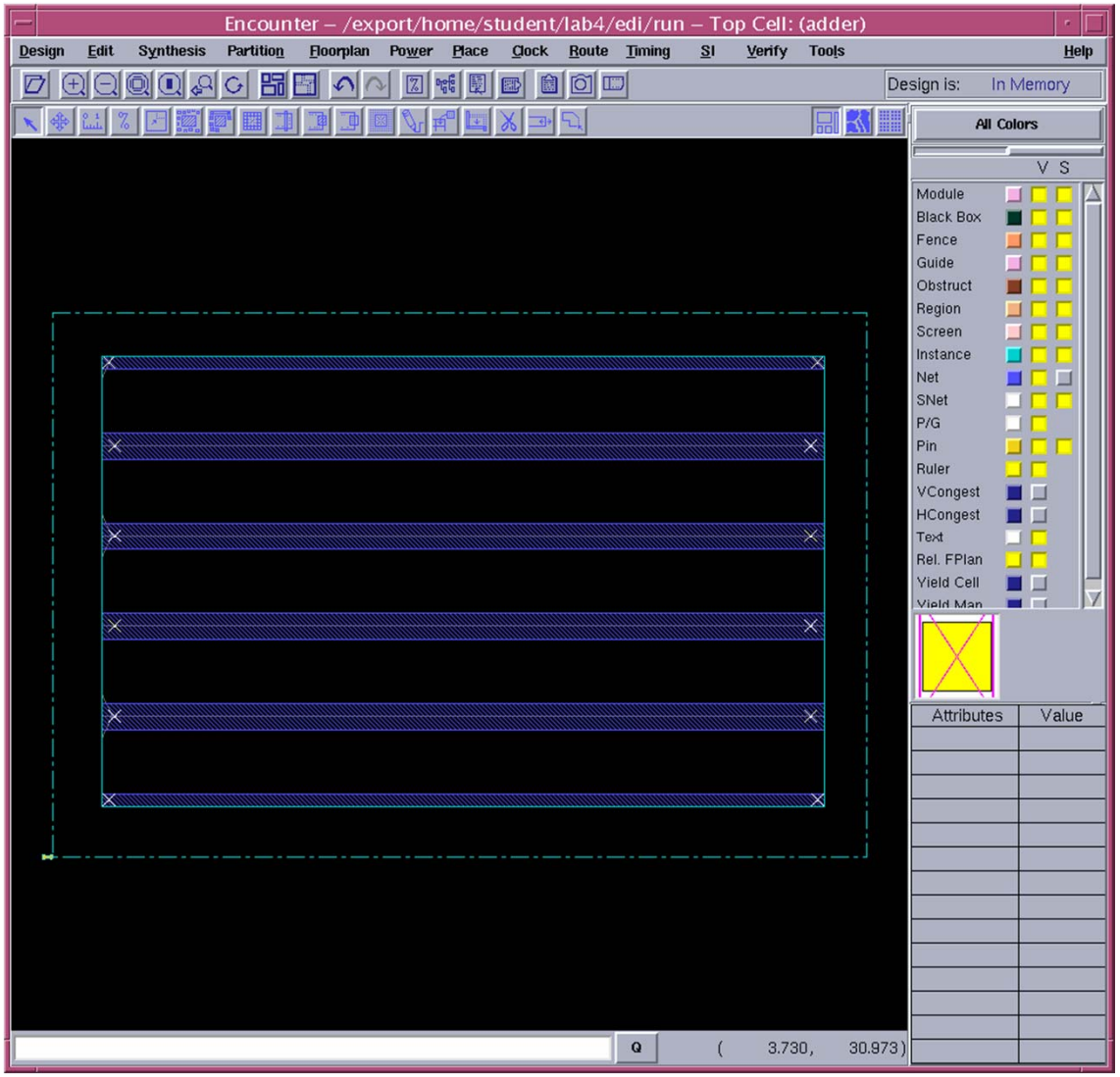
Plan Power

- 选择菜单
  Route ->
  Special Route
- 弹出sRoute菜单，全部采用默认值。
- 点击OK



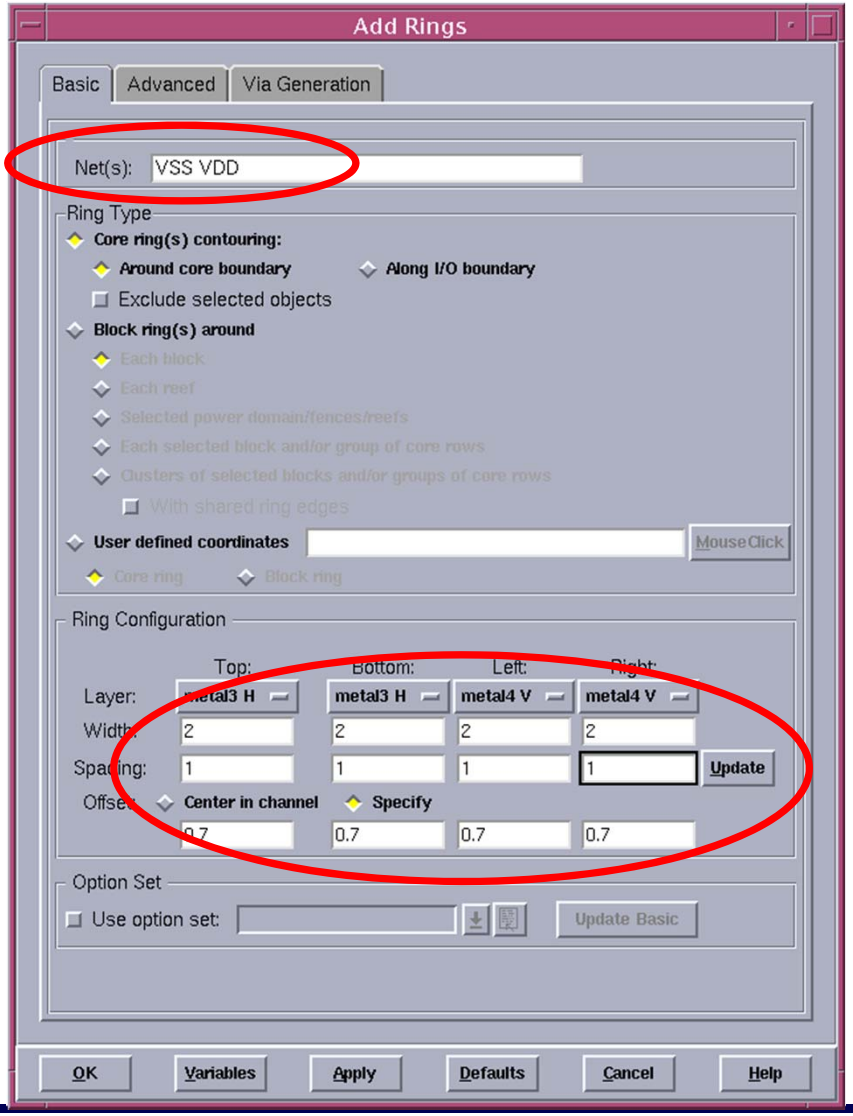
Plan Power

- sRoute后的结果



Plan Power —— Add Rings

- 选择菜单
- power -> Power Planning -> Add Rings



Plan Power —— Add Rings

- Net(s) — 指定特殊net的名称。缺省时为在数据库中出现的所有的电源线和地线。第一个出现的Net将出现在Ring的最内环（最靠近Core)
- Ring Type —设置ring的类型，实验采用around core boundary类型
- Ring configuration — 设置Ring的配置属性

  layer - 所用的金属层（采用“横奇纵偶”的原则，即top和bottom采用metal3、left和right采用metal4）

  width - 设置ring的宽度，这里设为 2；

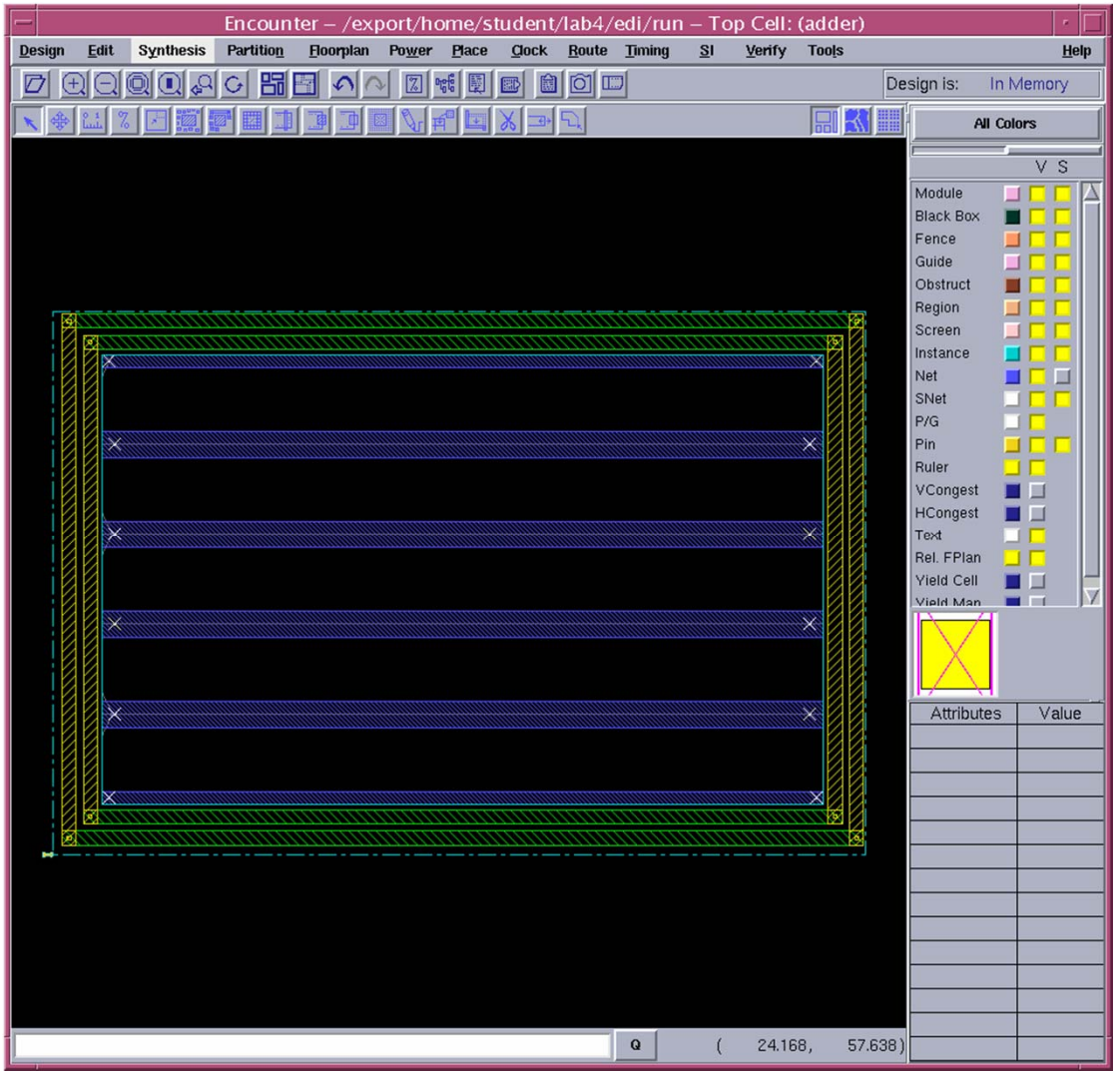
  spacing - 两个ring之间间距，这里设为1；

  offset – 最内圈的ring距core的距离，这里设为0.7。

点击OK，完成设置。

Plan Power —— Add Rings

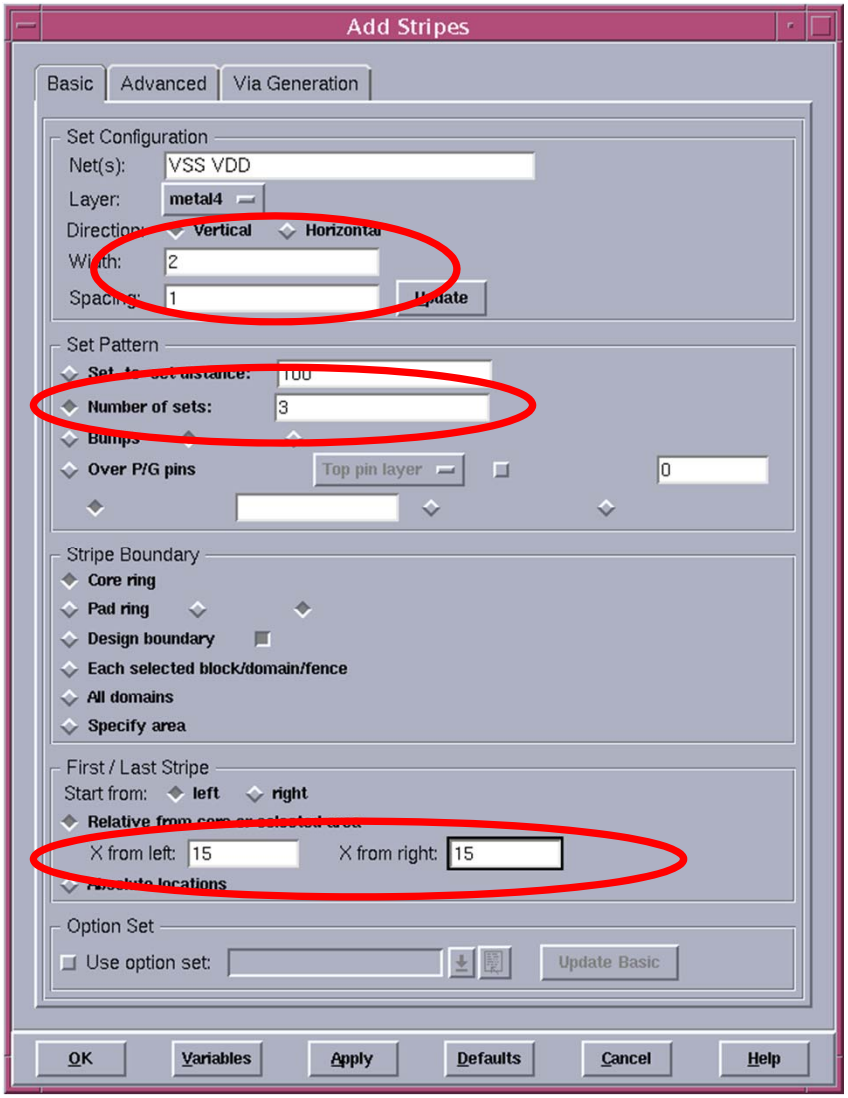
- **Add rings**后的结果



Plan Power —— Add Stripes

●选择菜单 Power -> Power Planning -> Add Stripes

●弹出add stripes菜单



Plan Power —— Add Stripes

- Set configuration – 设置stripe配置属性

Net(s) — 指定特殊net的名称。缺省时为在数据库中出现的所有的电源线和地线；

layer — 同样采用“横奇纵偶”，采用metal4；

Direction — 选择 vertical；　　width — stripe宽度，设为 2；

spacing — 每组stripes中的间距，设为 1；

- Set Pattern—设置stripes对的属性

Number of sets — 添加stripes的对数，这里设为3；

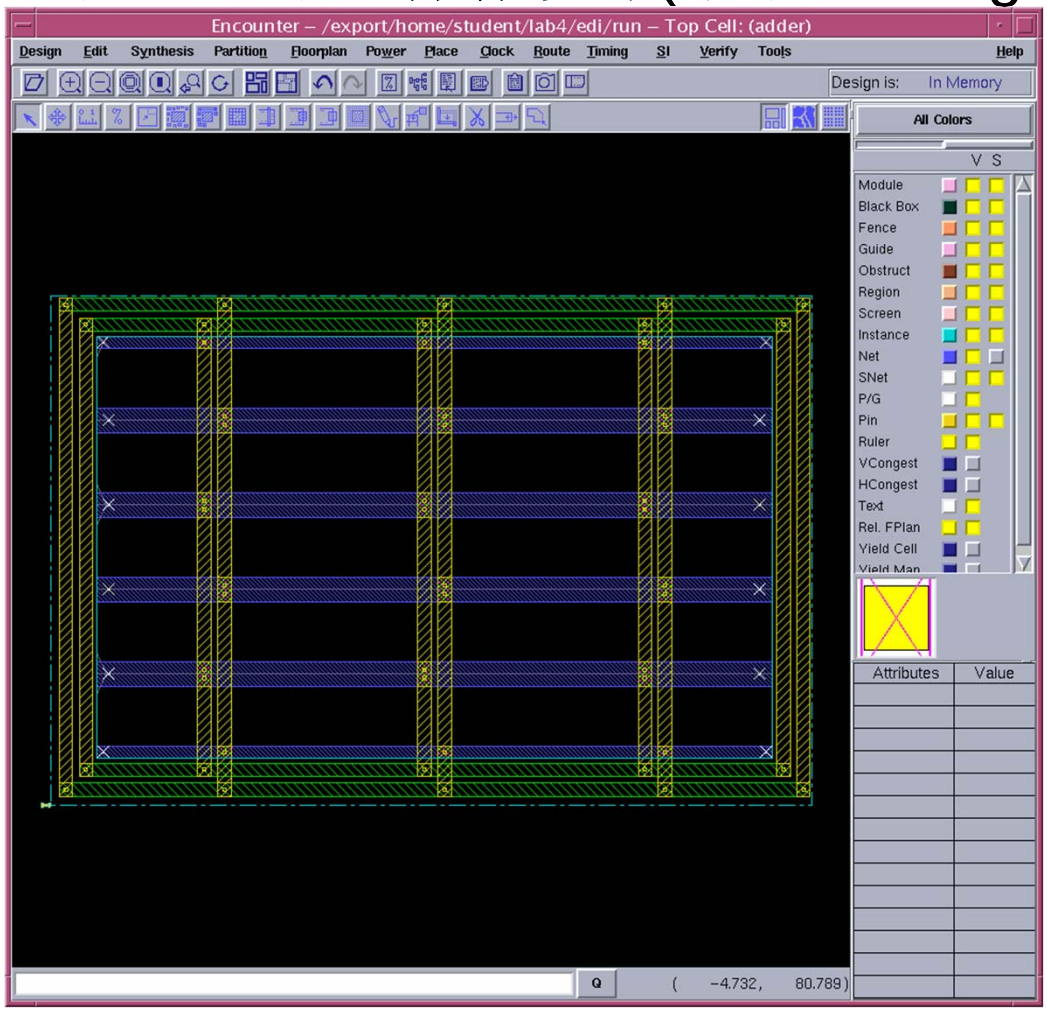
- First/Last Stripes — 设置stripes的排列属性

X from left：设为15　　X from right ：设为15

点击OK，完成设置

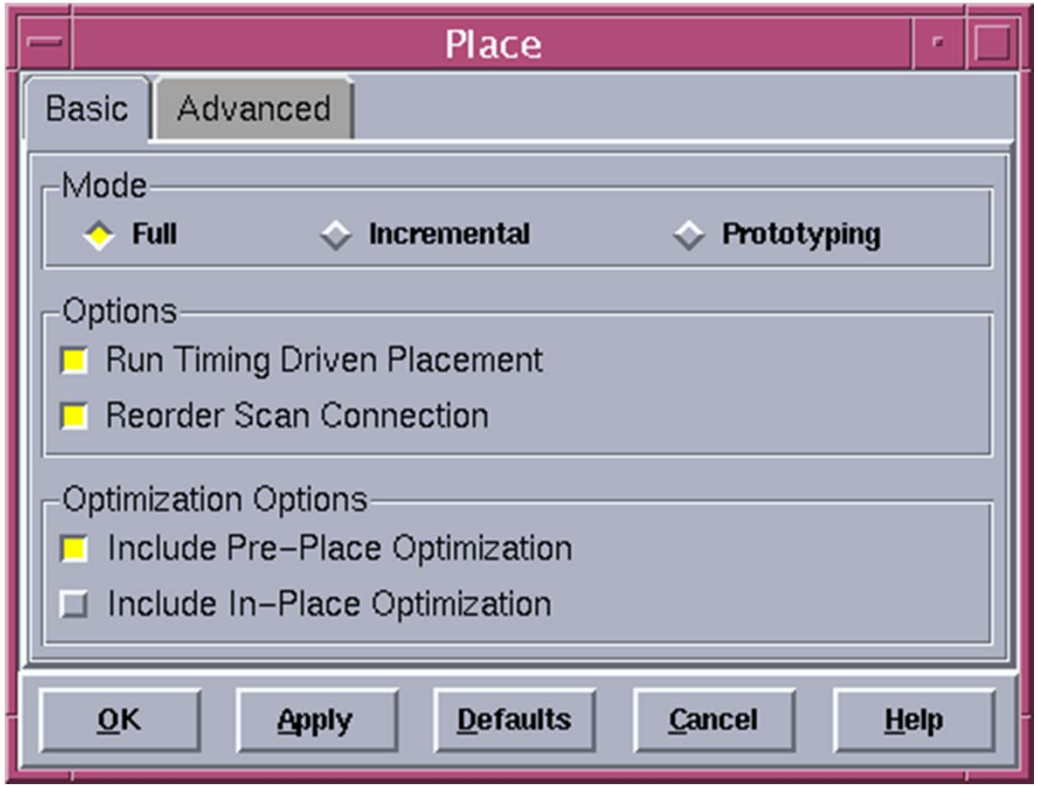
Plan Power —— Add Stripes

- Add Stripes 后的结果，保存设计(菜单：design->save design)



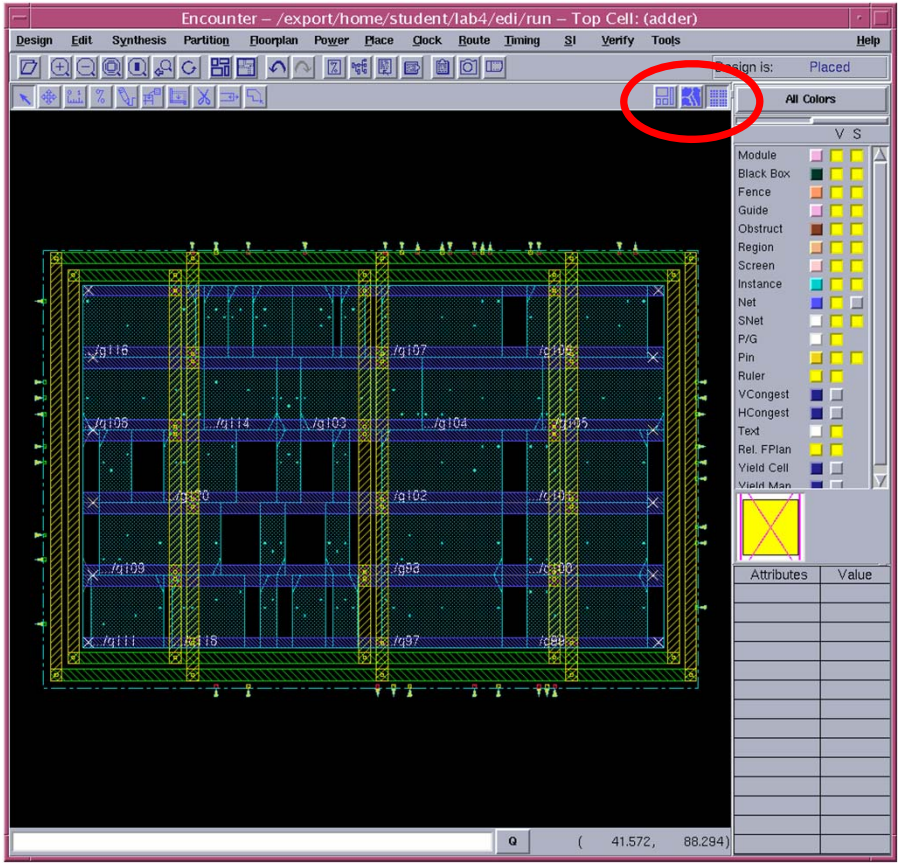
Place Cell

- 选择菜单：**Place -> Standard Cells And Blocks**
- 采用默认选项进行**place**，点击**OK**完成



Place Cell

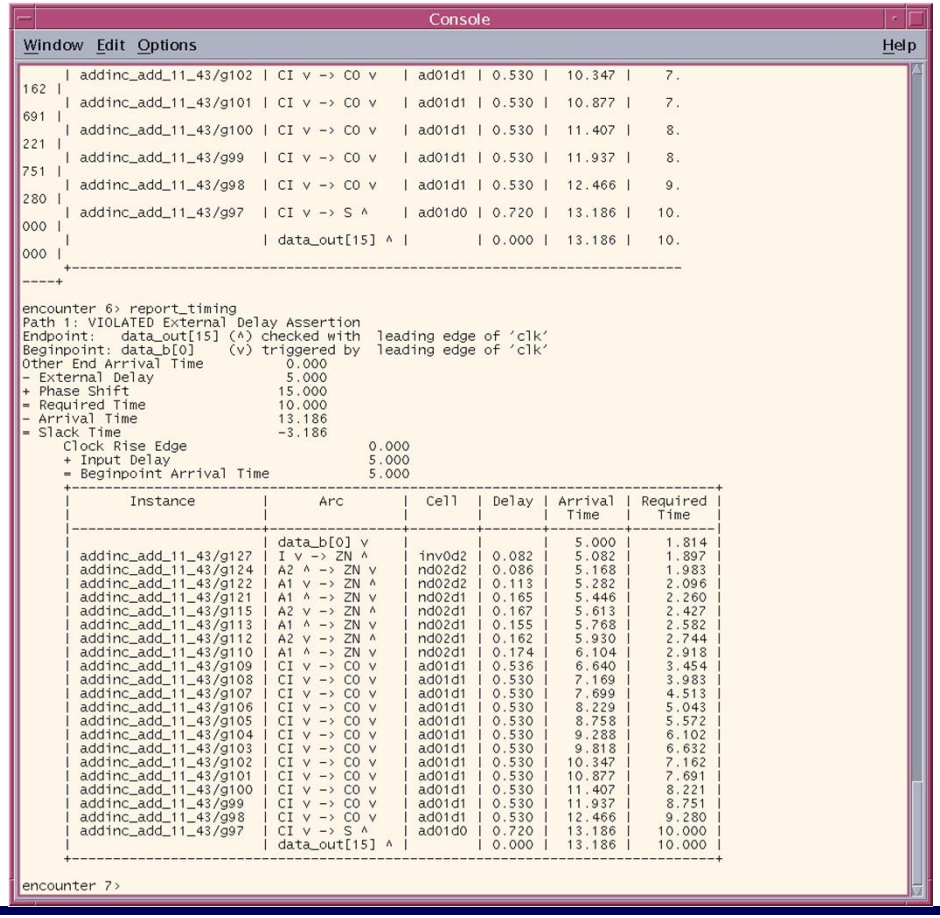
- 点击右上角的视图切换按钮，查看**place**后的结果。
- 保存设计（菜单：**design->save design**）



Place Cell

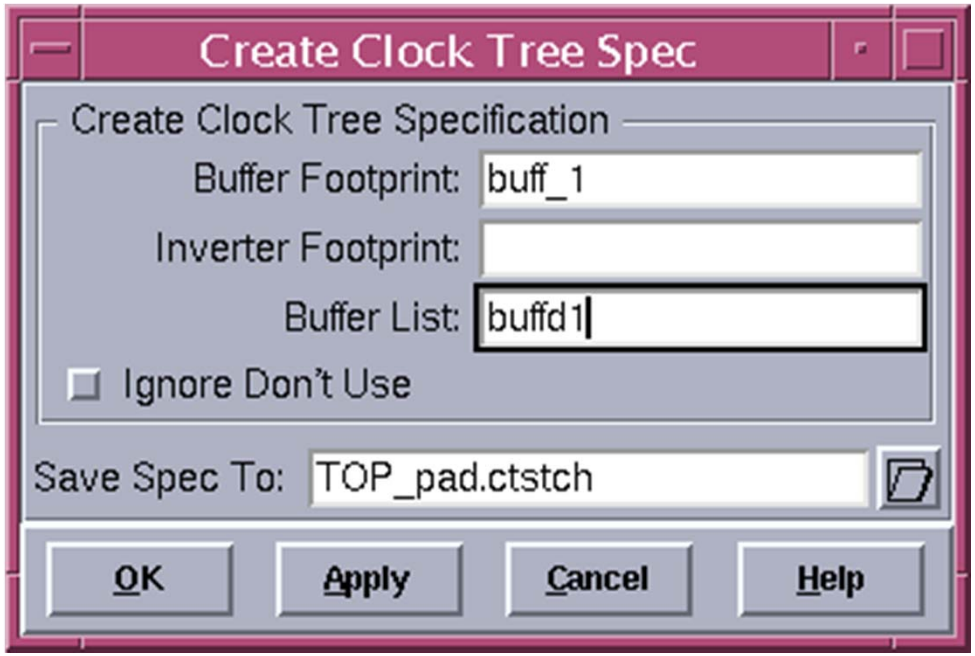
- 在控制台中键入命令: **report_timing**

encounter > report_timing > placeTiming.rpt查看时序报告。并记录slack的值



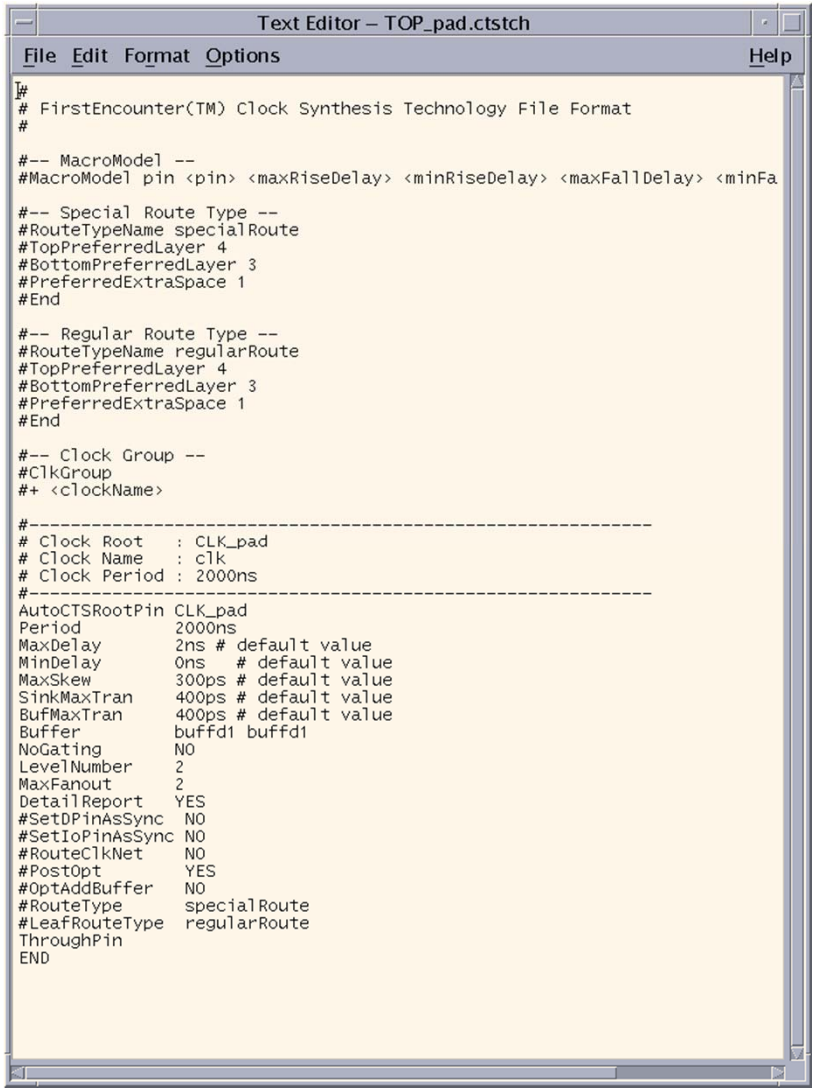
5、插入时钟树（设计中有时钟信号，本实验没有，不用做）

- 首先创建一个时钟树定义文件
- **Clock -> Create Clock Tree Spec**
  - **Biffer Footprint :buff_1**
  - **Buffer List :buffd1**



5、插入时钟树（设计中有时钟信号，本实验没有，不用做）

- 查看创建的**TOP_pad.ctstch**文件
- 其中调整各项参数
  - **MaxDelay/MinDelay**：最大/最小延迟
  - **MaxSkew**：时钟最大歪斜
- 也可以另外加入一些约束
  - **LevelNumber**： 时钟树级数
  - **MaxFanout**：最大分支数



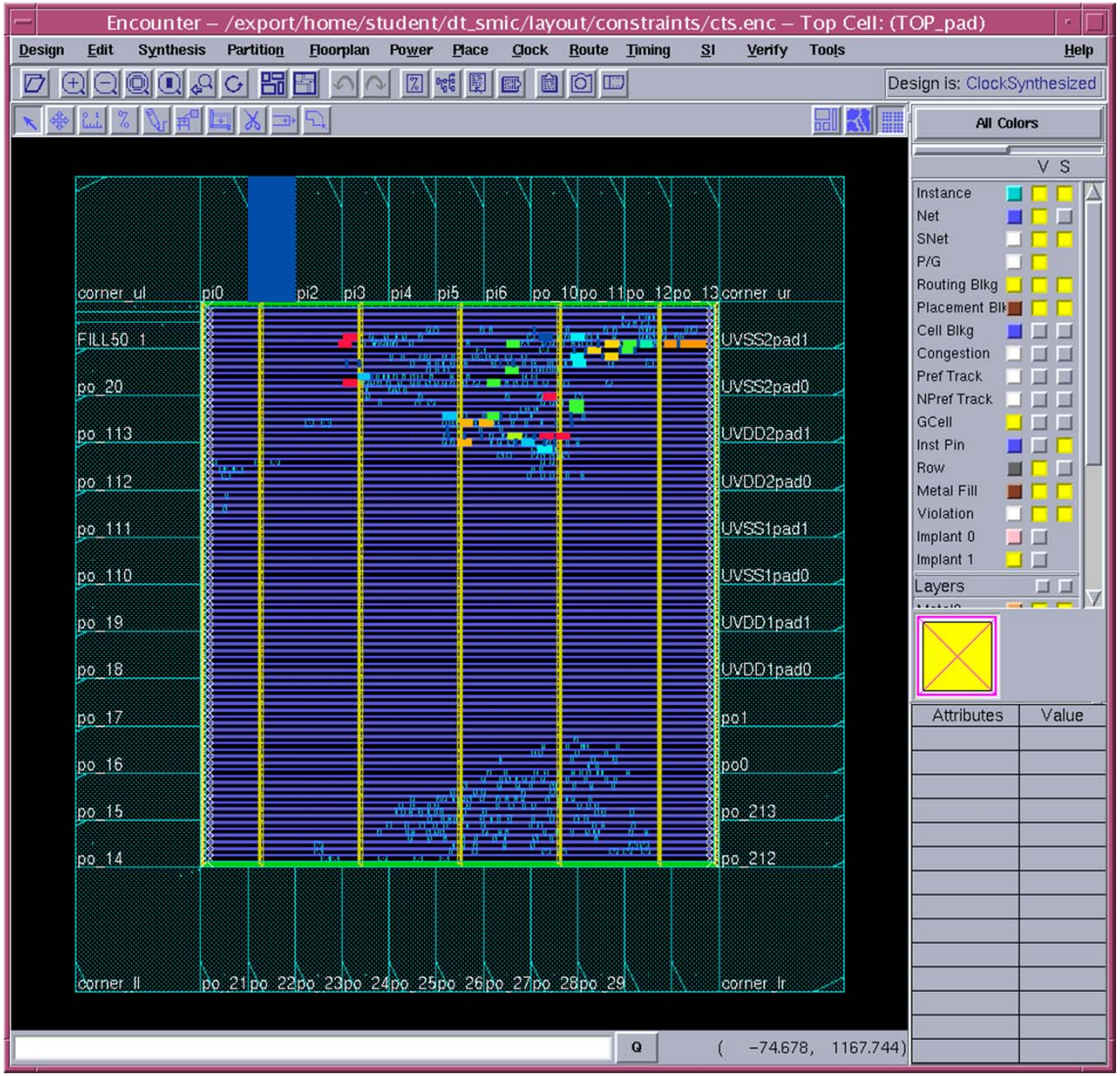
5、插入时钟树（设计中有时钟信号，本实验没有，不用做）

- 定义时钟树文件
  - **Clock -> Specify Clock Tree**

- 综合时钟树
  - **Clock -> Synthesize Clock Tree**

5、插入时钟树（设计中有时钟信号，本实验没有，不用做）

- 用时钟树查看工具查看综合得到的时钟树信息
  - **Clock -> Display**
- 可以看到插入的时钟树**Buff**以及与时钟相连的模块



5、插入时钟树（设计中有时钟信号，本实验没有，不用做）

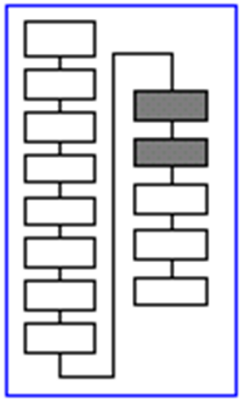
- 时钟树时序分析
- 采用脚本命令：

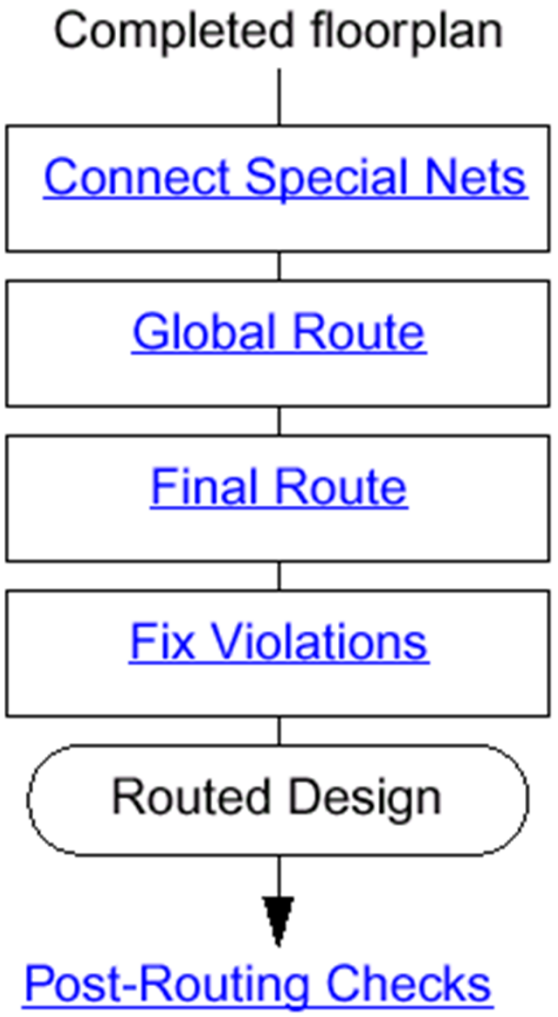
  set_propagated_clock [all_clocks]

  setAnalysisMode –skew –clockTree

  Report_clock_timing –type skew

6、布线Route —— Routing Flow



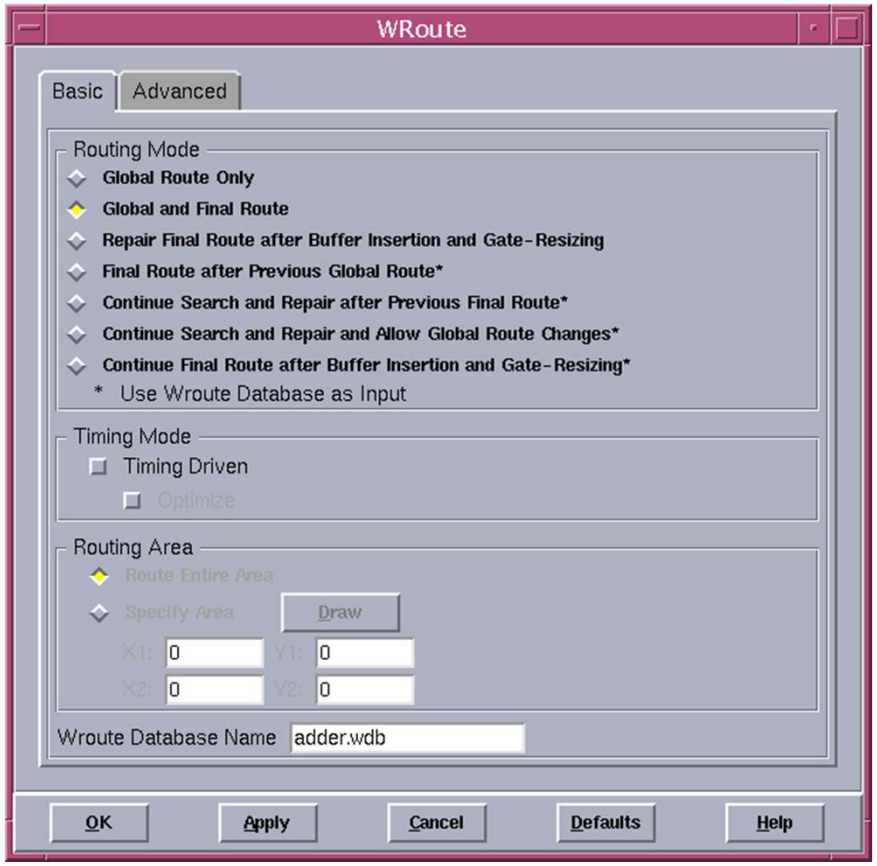


Routing — Wroute

选择菜单：**Route -> wRoute**

采用基于面积的算法完成信号线的**Globe**和**final**布线

参数如下图所示进行设置



Routing —— WRoute

●**Globe Route —** 进行全局布线并更新密度图。在全局布线时，布线器在提供的或要求的布线通道上，完成设计的DEF文件定义的常规连接线的互连规划。它只找出可能的通路，并不进行实际布线，并进行多次迭代来优化全局布线，降低连接线长度，并减少vias的使用。同时还更新密度图。**Global Route**只设置WRoute.Global环境变量。

●**Global and Final Route —** 进行全局及最终布线。在final布线时，布线器根据全局布线规划进行实际布线，连接引脚到相应的连接线上，同时排除布线violation并减小连接线长度及vias的数量。Final布线自动的进行查找并修复操作，除非用户指定不这样做。当超过了用户设定的布线时间限制，或者已经不能对布线做出改善时，Final布线会自动停止。final布线只能在 global布线之后才能进行。**Global及Final布线**设置WRoute.Global和WRoute.Final环境变量

Routing —— WRoute

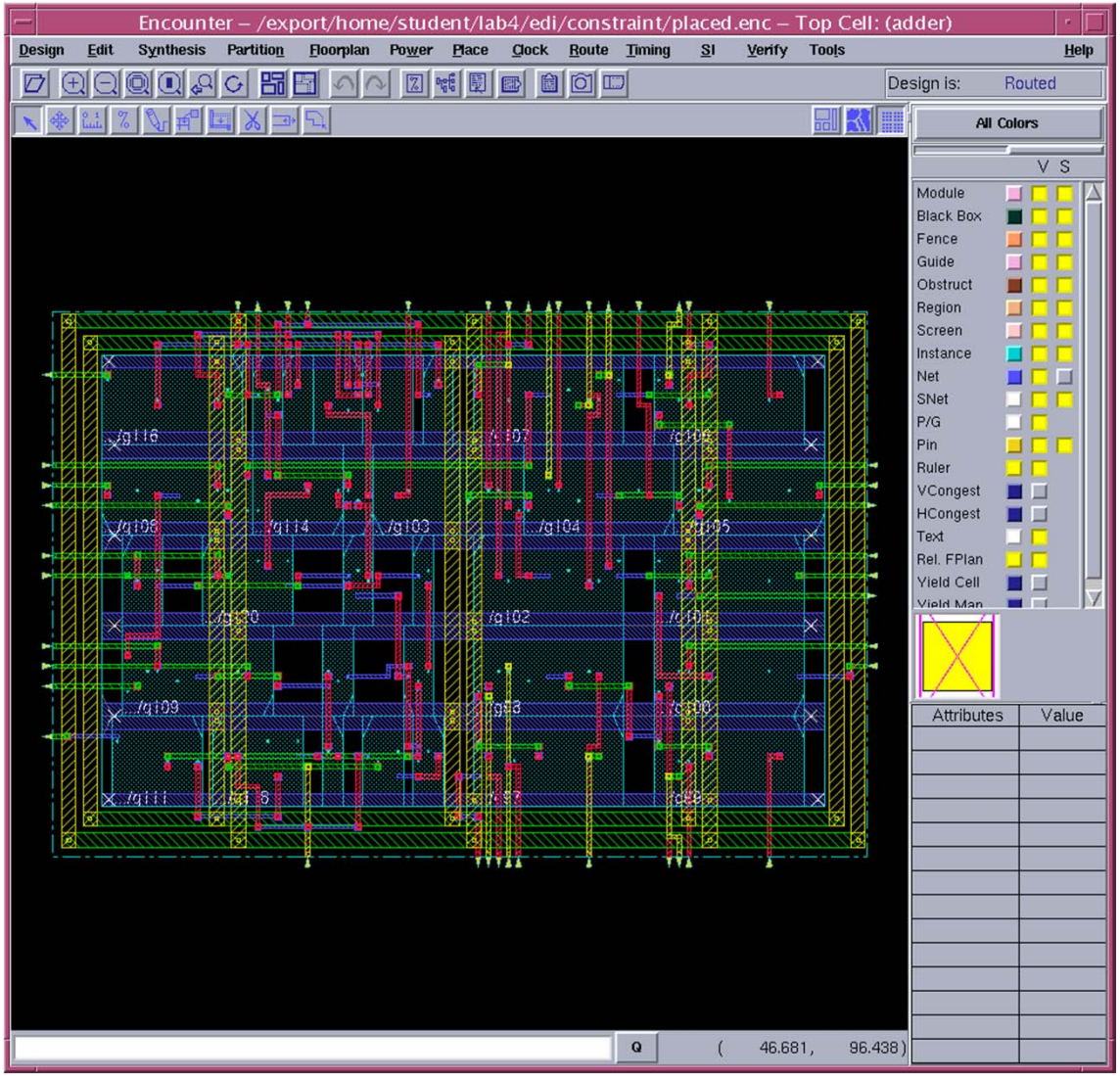
●**Timing Driven Routing —** 指示布线器使用数据库中的时序约束。布线器首先使用寄生参数小的层进行关键路径布线。如果选择了时序约束布线但没有设置时序约束，则会弹出一个窗口显示警告信息。布线可以继续进行，但在布线时不会考虑时序约束。设置了
WRoute.Timing.Driven环境变量

Routed Design

●点击 OK，开始自动布线

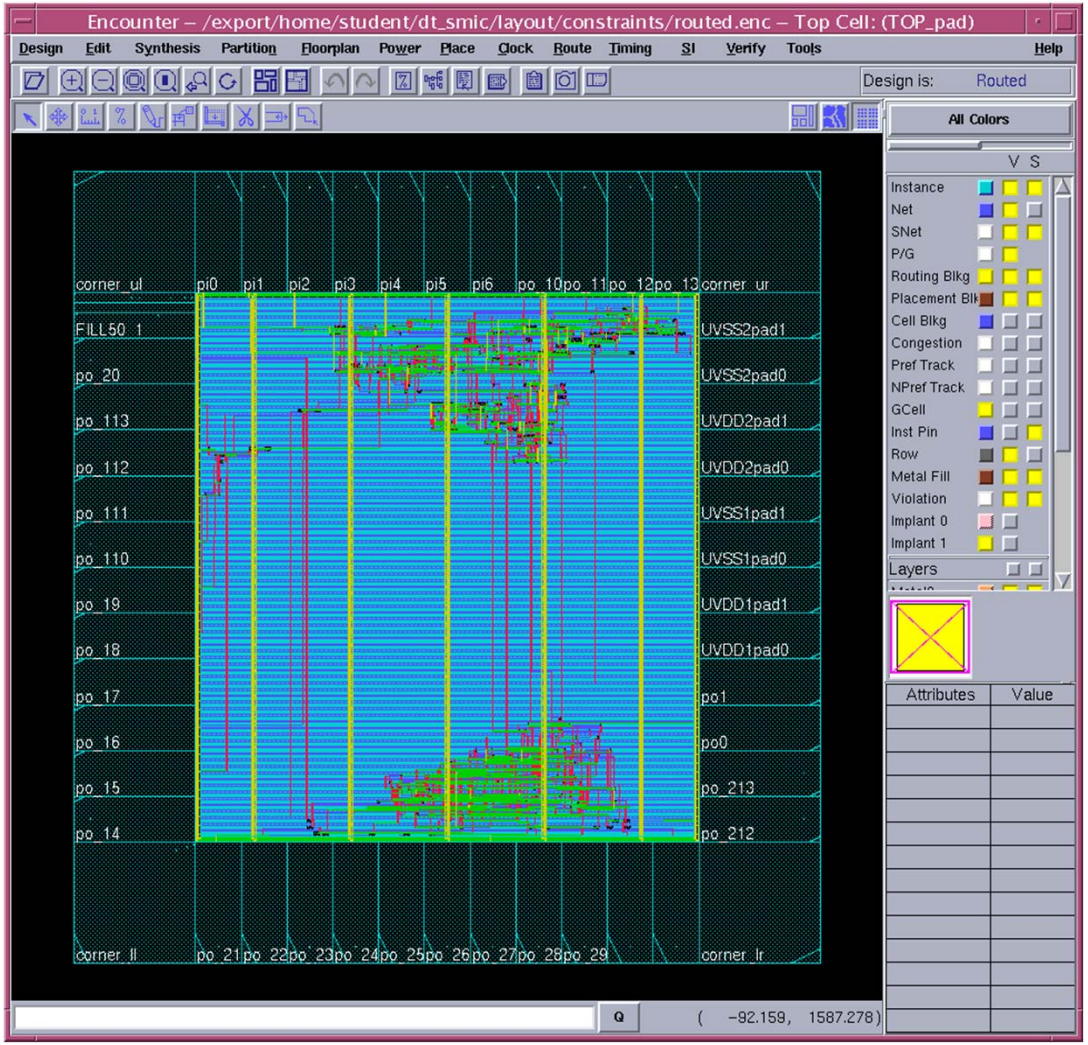
●布线结束后，如果没有error，保存文件为Routed.enc

●在控制台中键入命令：
report_timing,查看时序报告和没布线之前的时序进行对比。



7、插入CORE的Filler

- Place -> Filler -> Add
  - Cell Names : feedth



8、导出DRC以及LVS需要的文件

- saveNetlist：从版图导出电路网表
- write_sdf：导出时序信息sdf文件
- write_spef：提取RC延迟
- streamOut：导出GDS文件

数字后端实验总结

- 数字系统的后端设计流程
- 初步掌握Cadence自动布局布线工具SE
- 完成自动布局布线