第五章 版图编辑与版图验证

5.1 版图设计基础

5.1.1 版图设计方法简介

1. 版图设计方法

目前的版图设计方法有三种：

(1) 人工设计。人工设计和绘制版图有利于充分利用芯片面积，并能满足多种电路性能要求。但是效率低，周期长，容易出错，特别是不能设计规模稍大一些的电路版图。 因此，该方法多用于随机格式的、 产量较大的MSI和LSI或单元库的建立。

(2) 计算机辅助设计。在计算机辅助设计系统数据库中，预先存入版图的基本图形，形成图形库。设计者通过一定的操作命令可以调用、 修改、 变换和装配库中的图形，从而形成设计者所需要的版图。

在整个设计过程中，设计者可以通过显示器显示来观察任意层次版图的局部和全貌；可以通过键盘、 数字化仪或光笔进行设计操作；可以通过绘图仪得到所要绘制的版图图形。利用计算机辅助设计，可以降低设计费用， 缩短设计周期。

(3) 自动化设计。 在版图自动设计系统的数据库中，预先设计好各种结构单元的电路图、电路性能参数及版图，并有相应的设计软件。 在版图设计时，只要将设计的网表文件(Netlist)输入到自动设计系统中，再输入版图的设计规则和电路的性能要求， 自动设计软件就可以进行自动布局设计与自动布线设计， 并根据设计要求进行设计优化，最终输出版图。

2. 版图设计过程

版图设计的输入是电路的元件说明和网表文件， 其输出是设计好的版图。 在通常情况下，整个版图设计可分为划分(Partition)、 布图规划(Floorplanning)、 布局(Placement)、布线((Routing)和压缩(Compaction)。

(1) 划分。由于一个芯片包含上千万个晶体管， 加之受计算机存储空间和计算能力的限制，通常我们把整个电路划分成若干个模块，将处理问题的规模缩小。 划分时要考虑的因素包括模块的大小、 模块的数目和模块之间的连线数等。

(2) 布图规划和布局。布图规划是根据模块包含的器件数估计其面积，再根据该模块和其它模块的连接关系以及上一层模块或芯片的形状估计该模块的形状和相对位置。

布局的任务是确定模块在芯片上的精确位置，其目标是在保证布通的前提下使芯片面积尽可能小。

(3) 布线。布线阶段的首要目标是百分之百地完成模块间的互连，其次是在完成布线的前提下进一步优化布线结果，如提高电性能， 减小通孔数等。

(4) 压缩。压缩是布线完成后的优化处理过程，它试图进一步减小芯片的面积。目前常用的有一维压缩和二维压缩，较为成熟的是一维压缩技术。在压缩过程中必须保证版图几何图形间不违反设计规则。

整个布图过程可以用图来表示，布图过程往往是一个反复迭代求解过程。必须注意布图中各个步骤算法间目标函数的一致性， 前面阶段的算法要尽可能考虑到对后续阶段的影响。

3. 版图自动设计中的基本问题

VLSI版图是一组有规则的由若干层平面几何图形元素组成的集合。通常，这些图形元素只限于曼哈顿图形，即只由垂直边和水平边构成的图形， 且在同一层内不允许重叠。

(1) 图的定义及数据结构。需要了解的基本术语有：图、完全图和子图、 通路和回路、连接图和树、有向图、 二分图、平面图。有关的数据结构有：邻接矩阵、 关联矩阵、边-节点表（数组）和链表结构。

(2) 算法及算法复杂性。由于我们面对的处理对象是上千万个，甚至是上亿个图形，即使是二次方数量级的算法也是无法实现的。 以下算法和问题都是值得关注的。

① 需要考虑算法问题及算法复杂性、最优化问题、可行解问题、 NP-困难问题。

② 一些图论中问题的复杂性，如判别平面性、 最小生成树、最短路（从一点到所有点）、所有节点间的最短路，平面化、着色、 最长路、 斯坦纳树、旅行商问题等一些NP问题。

③ 几种求解NP-困难问题的方法。

· 限制问题的范围： 只对某一类问题求解。 例如， 在求图上的最小树时只求最小生成树，即限制问题数量的交叉点只能是原有的顶点。 求最小生成树可在一个多项式时间内求解，但不一定能获得最小树。

· 限制问题的规模： 例如， 旅行商问题的分区优化。

· 分支定界法。

· 启发式算法。

(3) 基本算法。 现在常用的图论算法有：DFS、 BFS、最短路径、最小生成树、斯坦纳树算法、匹配算法、网络流问题；计算几何算法有：扫描线算法； 基于运筹学的算法有：构形图和局部搜索、线性规划、整数规划、 动态规划、 非线性规划、模拟退火法。

(4) 版图数据的基本操作有：点查找、邻接查找、区域搜索、定向区域遍历、模块插入、模块删除、推移、压缩、建立通道。基本数据结构有：链表结构、基于BIN的结构、邻接指针、角勾链、 四叉树和二叉排序树等。

4. 复杂数字系统中的版图设计问题

版图设计主要包括模块划分、 模块规划、 布局、布线、版图集成等环节，是一个组合排列规划和合理拼接图形的工作， 是在一个规则形状的平面区域内不重叠地布置多个部件，在各部件之间依据电路图的信号连接要求进行连线。 虽然版图设计方法日益完善，针对版图设计的各个环节，各式各样的设计软件应运而生，使版图设计自动化，但是，无论何种方法，都是针对所有模块版图的共性而开发的， 对于某一个或某一类模块的特性，软件常显得力不从心。为了利用模块的特性进行版图设计，以追求指标较高的版图，对版图特别是单元库的手工设计的探索和研究是很有必要的。

复杂数字系统版图设计的主要技术指标如下：

(1) 延时性能。只有延时满足模块功能时序的版图才是合格的。 延时缩短有利于模块速度的提高。

(2) 功耗指标。功耗的大小是衡量版图性能的重要指标。只有功耗较小的版图才具有实用性。

(3) 面积大小。面积大小将直接影响芯片的成本。必须最大限度提高面积的利用率。

(4) 调整能力。版图设计必须有利于版图的最终调整。调整是版图符合模块功能要求的必经之路。

5.1.2 版图设计规则

1. 设计规则的内容与作用

(1) 设计规则是集成电路设计与制造的桥梁。如何向电路设计及版图设计工程师精确说明工艺线的加工能力，就是设计规则描述的内容。

(2) 这些规则是以掩模版各层几何图形的宽度、间距及重叠量等最小允许值的形式出现的。

(3) 设计规则本身并不代表光刻、化学腐蚀、对准容差的极限尺寸， 它所代表的是容差的要求。

2. 设计规则的描述

(1) 自由格式。一般的MOS集成电路的研制和生产，基本上采用这类规则，其中每个被规定的尺寸之间没有必然的比例关系。显然，在这种方法所规定的规则中，对于一个设计级别， 就要有一整套数字，因而显得烦琐，但由于各尺寸相对独立， 因此，可把尺寸定得合理。

(2) 规整格式。其基本思想是由Mead和Conway提出的，在这类规则中，把绝大多数尺寸规定为某一特征尺寸“λ”的某个倍数。

3. 简单的λ规则

(1) 最小宽度及间距的λ规则如表5-1所示，宽度及间距的含义如图5-1所示。

① Diff：两个扩散区之间的间距不仅取决于工艺上几何图形的分辨率，还取决于所形成的器件的物理参数。如果两个扩散区靠得太近，在工作时可能会连通， 产生不希望出现的电流。

② Poly-Si： 多晶硅取决于工艺上几何图形的分辨率。

③ Al：铝生长在最不平坦的二氧化硅上，因此，铝的宽度和间距都要大些，以免短路或断铝。

④ Diff-Poly：无关多晶硅与扩散区不能相互重叠，否则将产生寄生电容或寄生晶体管。

表5-1 最小宽度及间距的 λ 规则表

类型	最小宽度	最小间距
Diff	3λ	3λ
Poly-Si	2λ	2λ
A1	3λ	3λ
Diff-Poly		λ

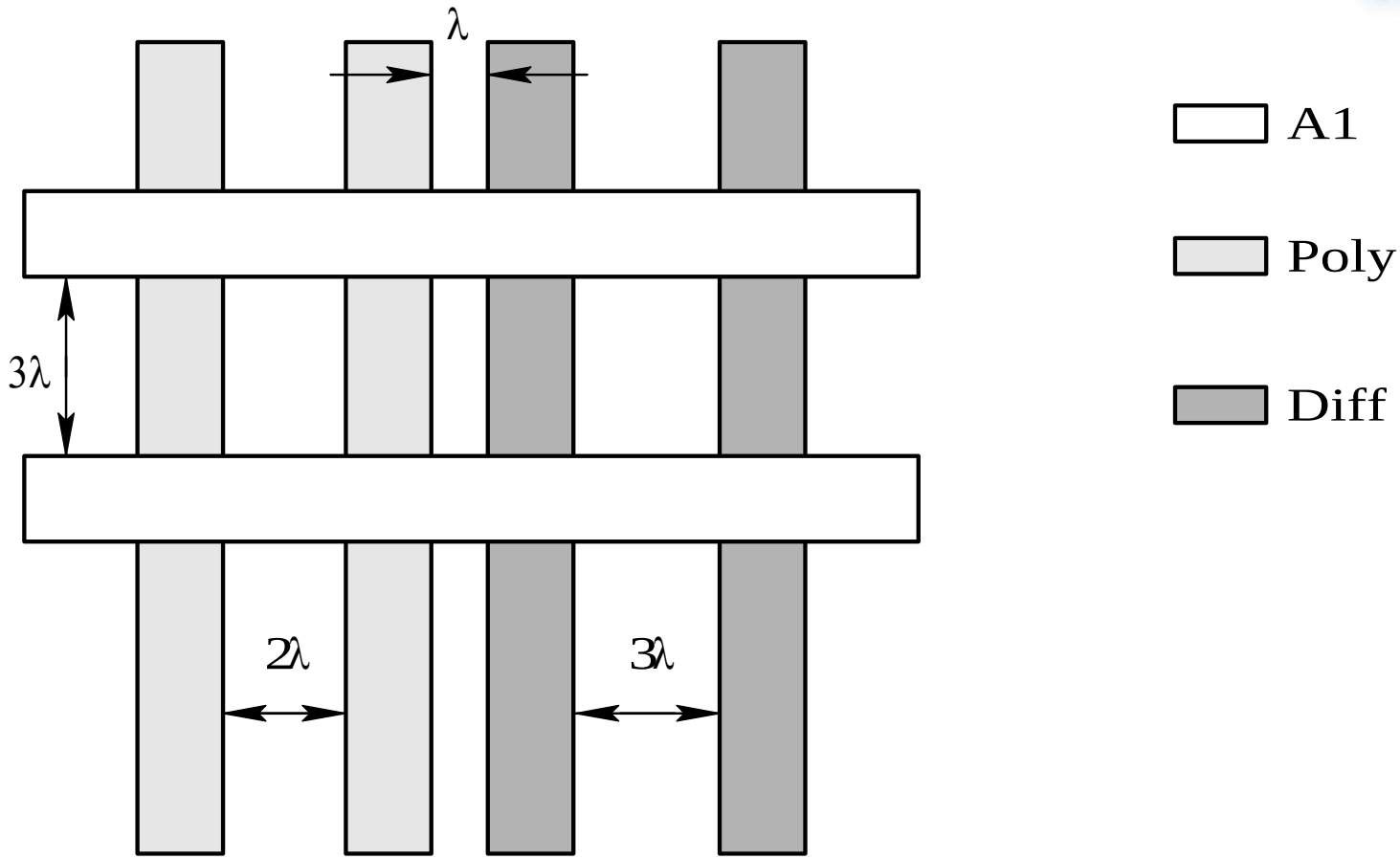


图 5-1 宽度及间距的含义图

(2) 接触孔的大小规则如图5-2所示。

① 孔的大小： 2λ ×2λ ；

② Diff、 Poly的包孔：1λ ；

③ 孔间距： 1λ ；

(3) 晶体管的λ规则如图5-3所示。

① 多晶硅与扩散区的最小间距： λ；

② 栅出头： 2λ， 否则会出现S、 D短路的现象；

③ 扩散区出头： 2λ，以保证S或D有一定的面积。

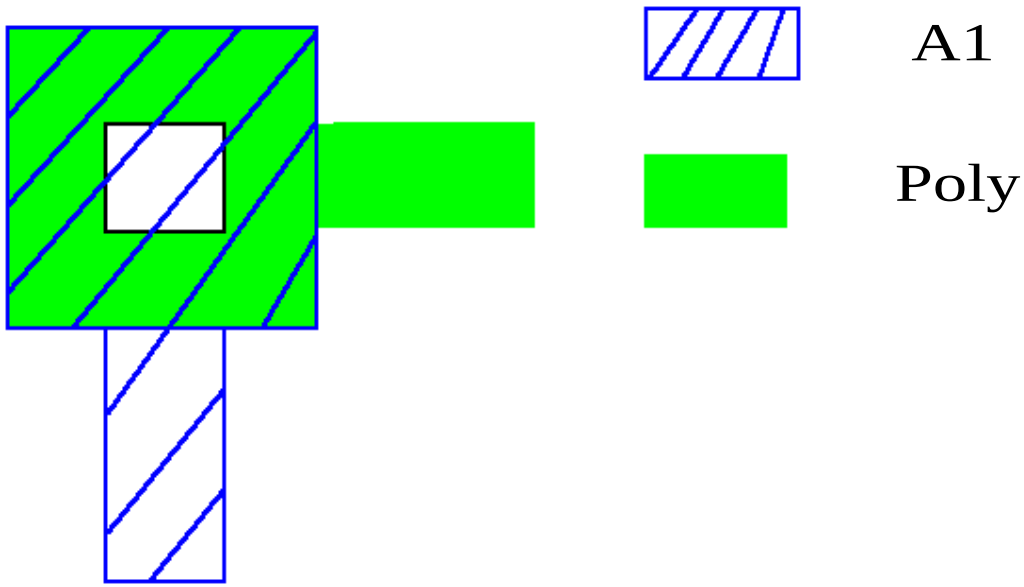


图 5-2 接触孔的大小规则示意图

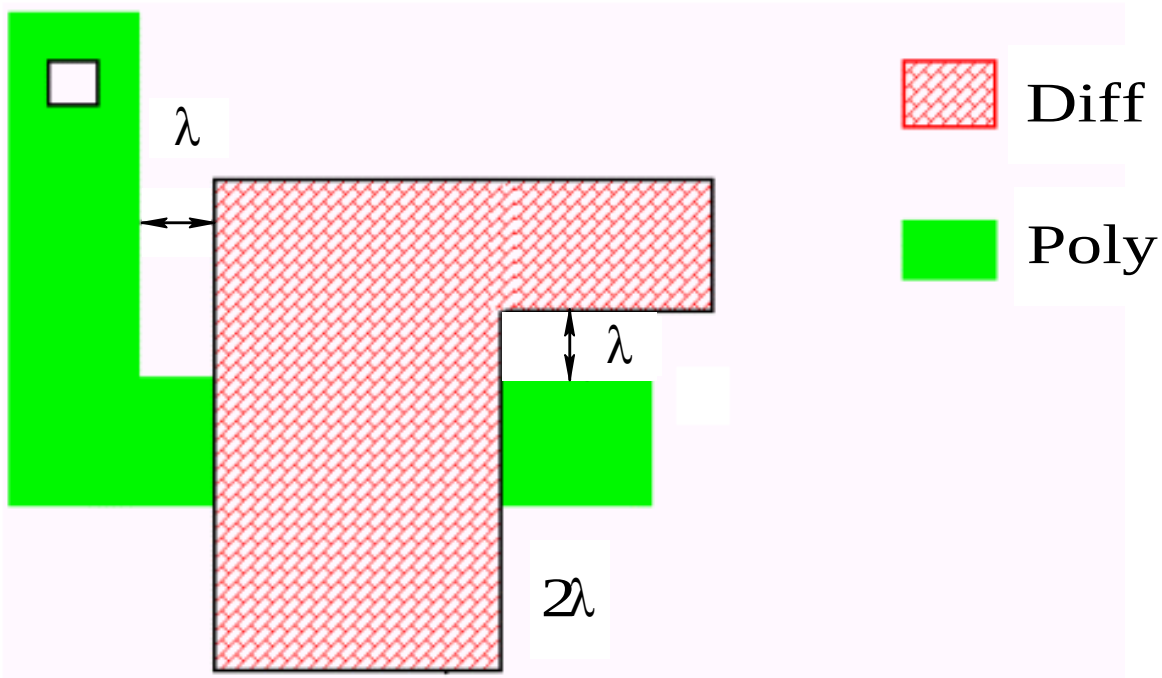


图 5-3 晶体管的 λ 规则示意图

(4) P阱规则如图 5-4所示。

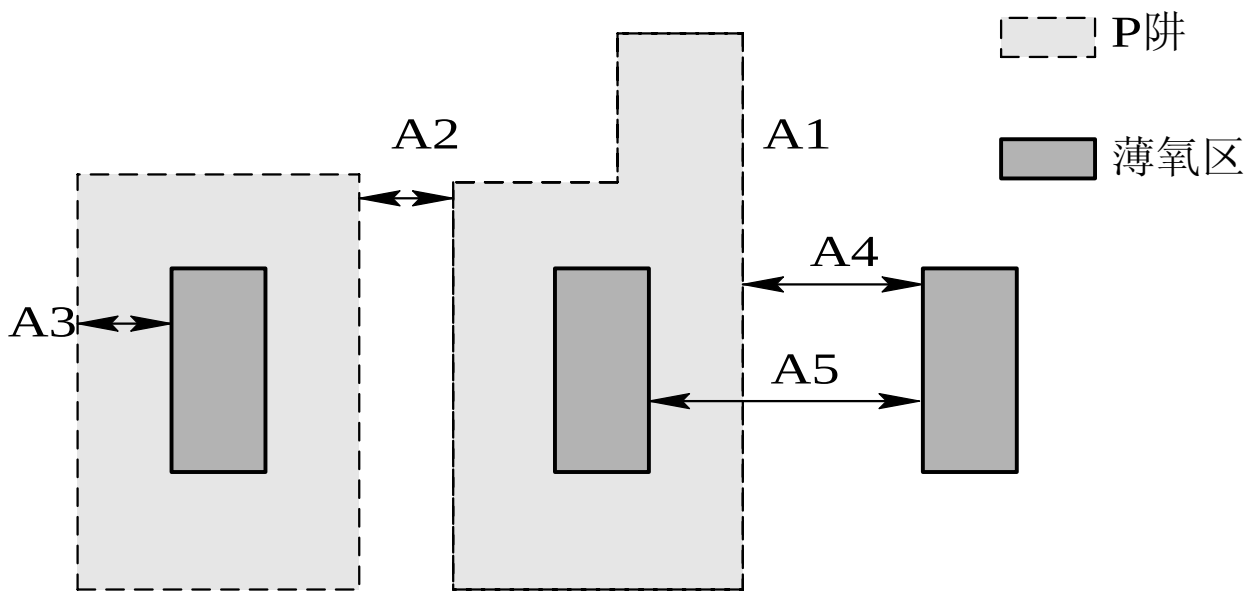


图 5-4　P阱规则示意图

① A1=4λ： 最小P阱宽度；

② A2=2λ/6λ：P阱间距。当两个P阱同电位时，A2=2λ；当两个P阱异电位时，A2=6λ；

③ A3=3λ： P阱边沿与内部薄氧化区(有源区)的间距；

④ A4=5λ： P阱边沿与外部薄氧化区(有源区)的间距；

⑤ A5=8λ： P管薄氧化区与N管薄氧化区的间距。

4. 深亚微米CMOS设计规则

我们知道，当前数字系统芯片的制作工艺主要是CMOS工艺，CMOS中的门电路是由非门、与门等组合构成的。下面，我们以最常见的CMOS双阱工艺的非门(反相器)来介绍在深亚微米电路中的设计规则描述(仅介绍N阱和有源区，其余略)。反相器的CMOS工艺剖面图与布局图如图5-5所示。

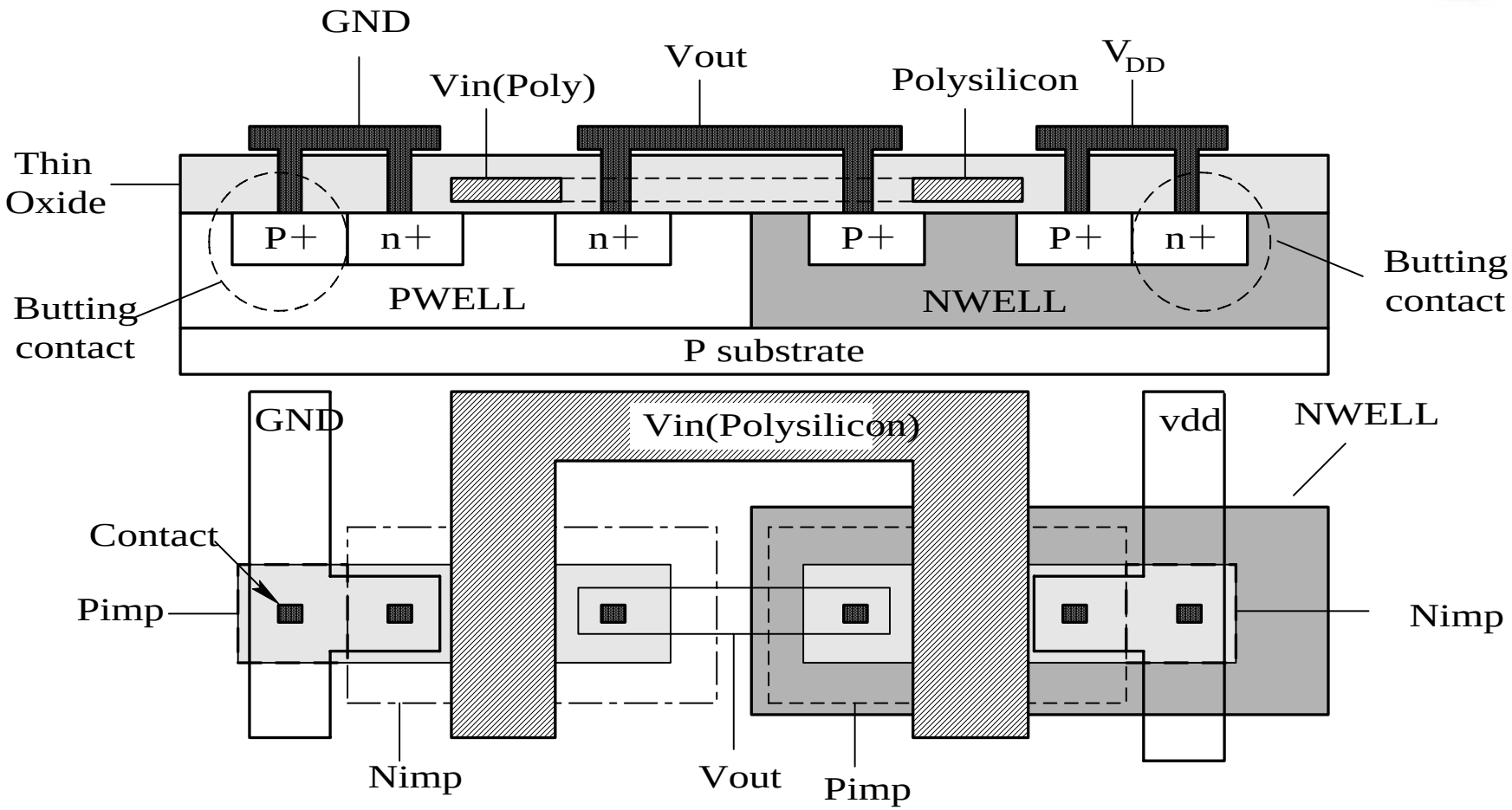


图 5-5 反相器的CMOS工艺剖面图与布局图

(1) 基本定义如图5-6所示。当我们确定最终芯片设计工艺流程后，应先将工艺厂家提供的设计(版图)规则理解并熟记，然后在设计中务必遵守这些规则。

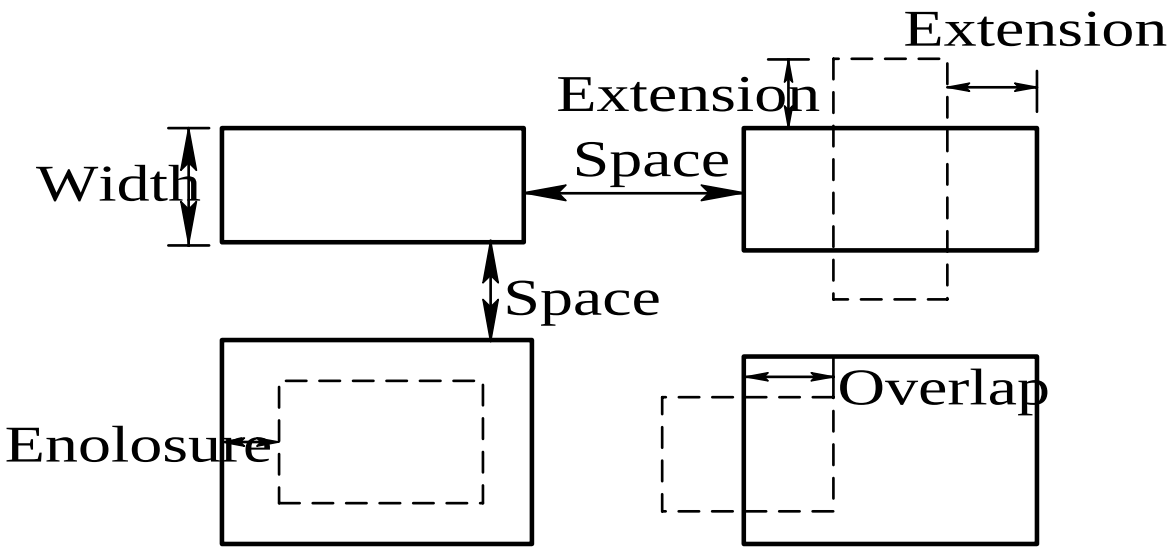


图 5-6 基本定义

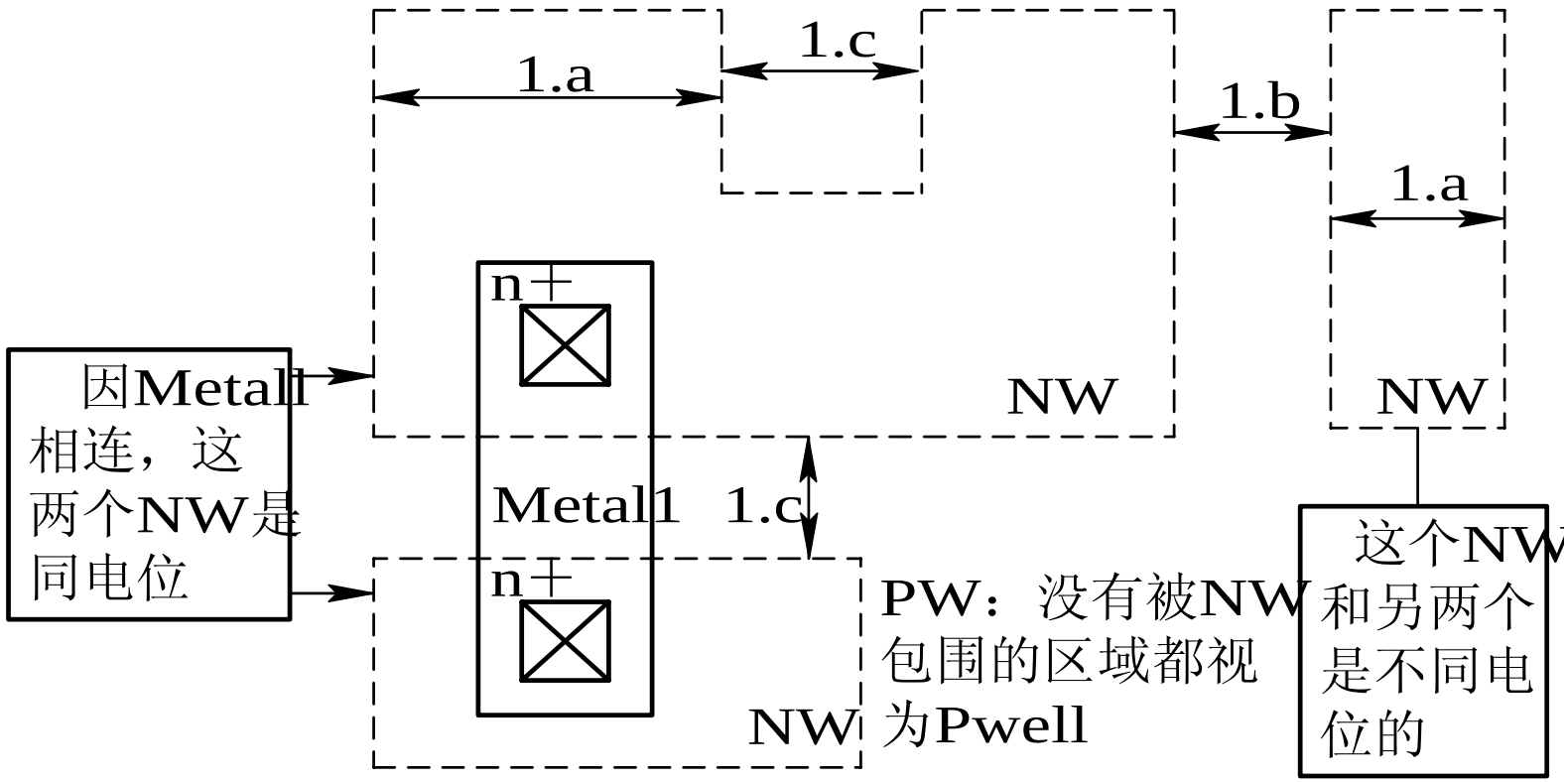


图 5-7 NW(N阱)设计规则

(2) NW(N阱)设计规则如图5-7所示。图5-7中规则编号1.a、1.b、1.c的意义如表5-2所示。

表 5-2 NW(N阱)设计规则

Rule No.	Rule Description	T-0.6 SPTM
1.a	minimum Width NW(最小N阱宽度)	3.0 μm
1.b	minimum Space NW-to-NW with different potentials (不同电势下,最小N阱间距)	4.8 μm
1.c	minimum Space NW-to-NW with the same potentials (相同电势下,最小N阱间距)	1.5 μm

(3) 有源区设计规则如图5-8所示。图5-8中规则编号2.a、2.b、 2.c、 2.d、 2.e、 2.f、 2.g、 2.h、 2.i、2.j的意义分别如表5-3所示。

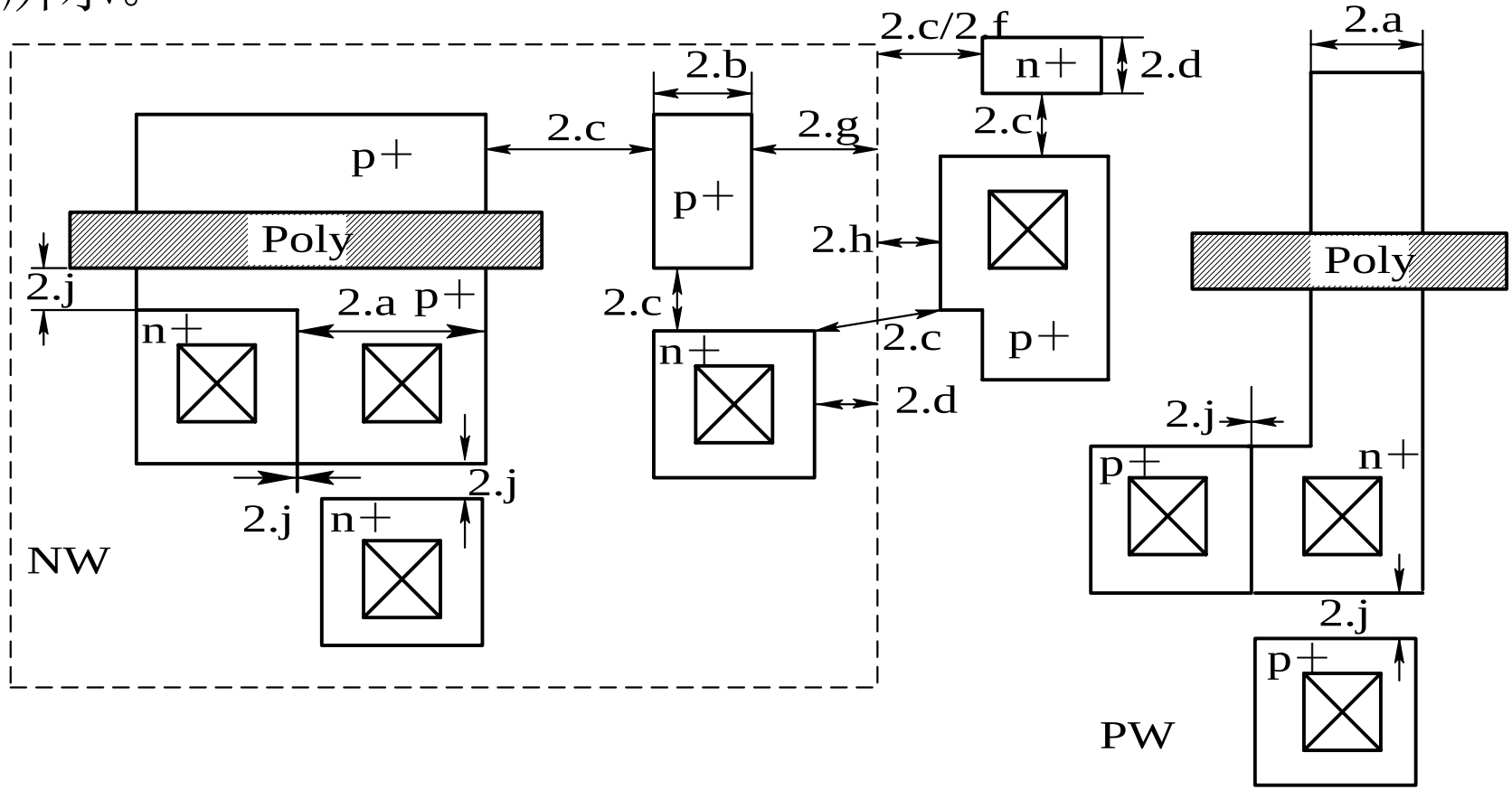


图5-8 有源区设计规则示意图

表5-3 OC(薄氧化层)设计规则

Rule No.	Rule Description	T-0.6 SPTM
2.a	minimum Width OD(Thin Oxide)for Active Devices(有源区最小宽度)	0.75 μm
2.b	minimum Width OD for interconnection(连线最小宽度)	0.6 μm
2.c	minimum Space OD-to-OD(有源区最小间距)	1.2 μm
2.d	minimum Enclosure NW[OD(n+)](N 阱内 n+区与 N 阱边沿最小宽度)	0.4 μm
2.e	minimum Space NW(cold)-to-OD(n+)(N 阱(cold)与 P 阱区边沿最小间距)	1.8 μm
2.f	minimum Space NW(hot)-to-OD(n+)(最小宽度)(N 阱(hot)与 P 阱区边沿最小间距)	4.0 μm
2.g	minimum Enclosure NW[OD(p+)](N 阱内 p+区与 N 阱边沿最小间距)	1.8 μm
2.h	minimum Space NW-to-OD(p+)(N 阱到 p+区最小间距)	0.4 μm
2.i	minimum Space PO-to-OD (on active region)(多晶硅到有源区最小间距)	0.75 μm
2.j	minimum Space OD(p+)-to-OD(n+)(p+区到 n+区最小间距)	0.0 or 1.2 μm

5.2 Tanner Research Tools组成与功能

5.2.1 安装并熟悉L-edit Pro 9.0/8.xx版

(1) 访问http: //www.tanner.com或到FTP网站查找并下载L edit Pro 9.0/8.xx的最新教学版本， 安装到本地硬盘。

(2) 安装完毕后，可启动L-edit Pro，以熟悉其基本用法。例如，首先启动L-edit 8.3版，其界面是标准的Windows风格，这一点有别于Cadence等UNIX系统的EDA工具。点击“File”菜单下的“Open”命令，则弹出“打开”对话框。分别选择“Samples”文件夹下“Spr”子文件夹中“Examples”子文件夹的light.tdb、 lighslb.tdb文件，参见图5-9。

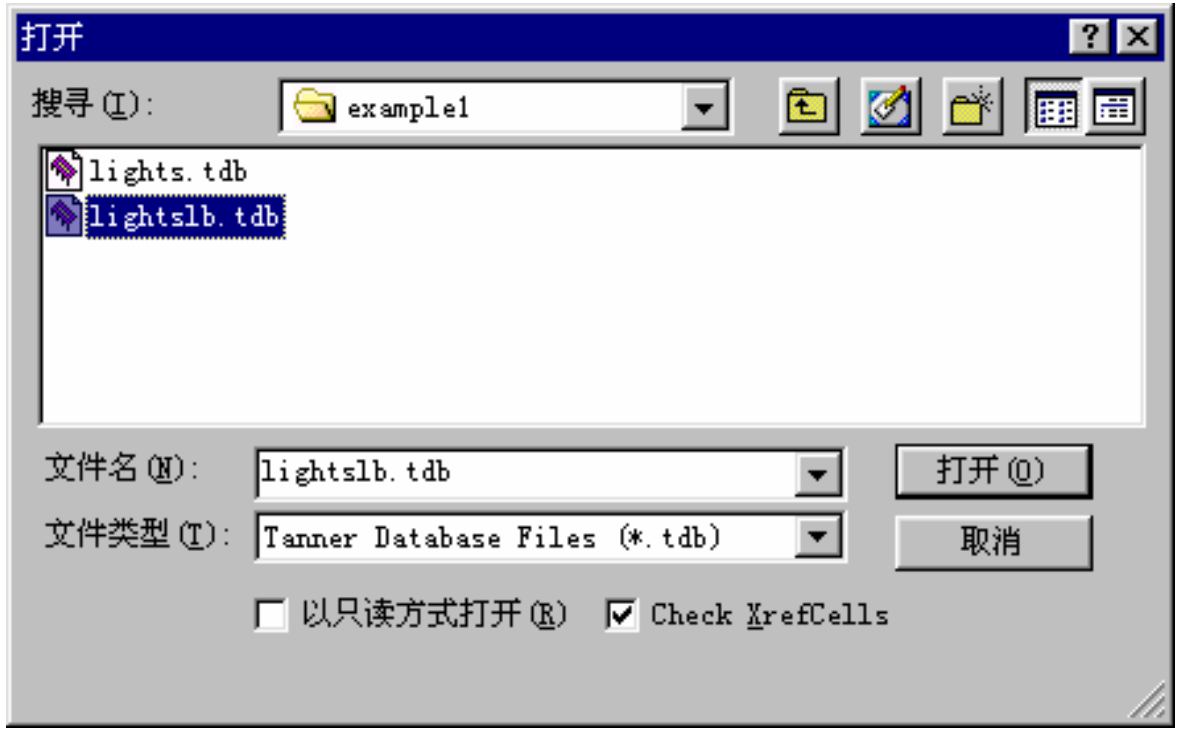

打开
搜寻(I):
example1
lights.tdb
lightslb.tdb
文件名(N):
lightslb.tdb
打开(O)
文件类型(T):
Tanner Database Files (*.tdb)
取消
以只读方式打开(R)
Check XrefCells

图5-9 “打开”对话框

(3) 在lighslb.tdb文件被打开后， 在如图5-10所示的下拉钮中可拖出“Top-Down-all cell”等4种库类型供我们选用。

(4) 例如，要选中一个“Inv”器件进行相关的操作，则出现如图5-10右下方所示的“虚线图框”。

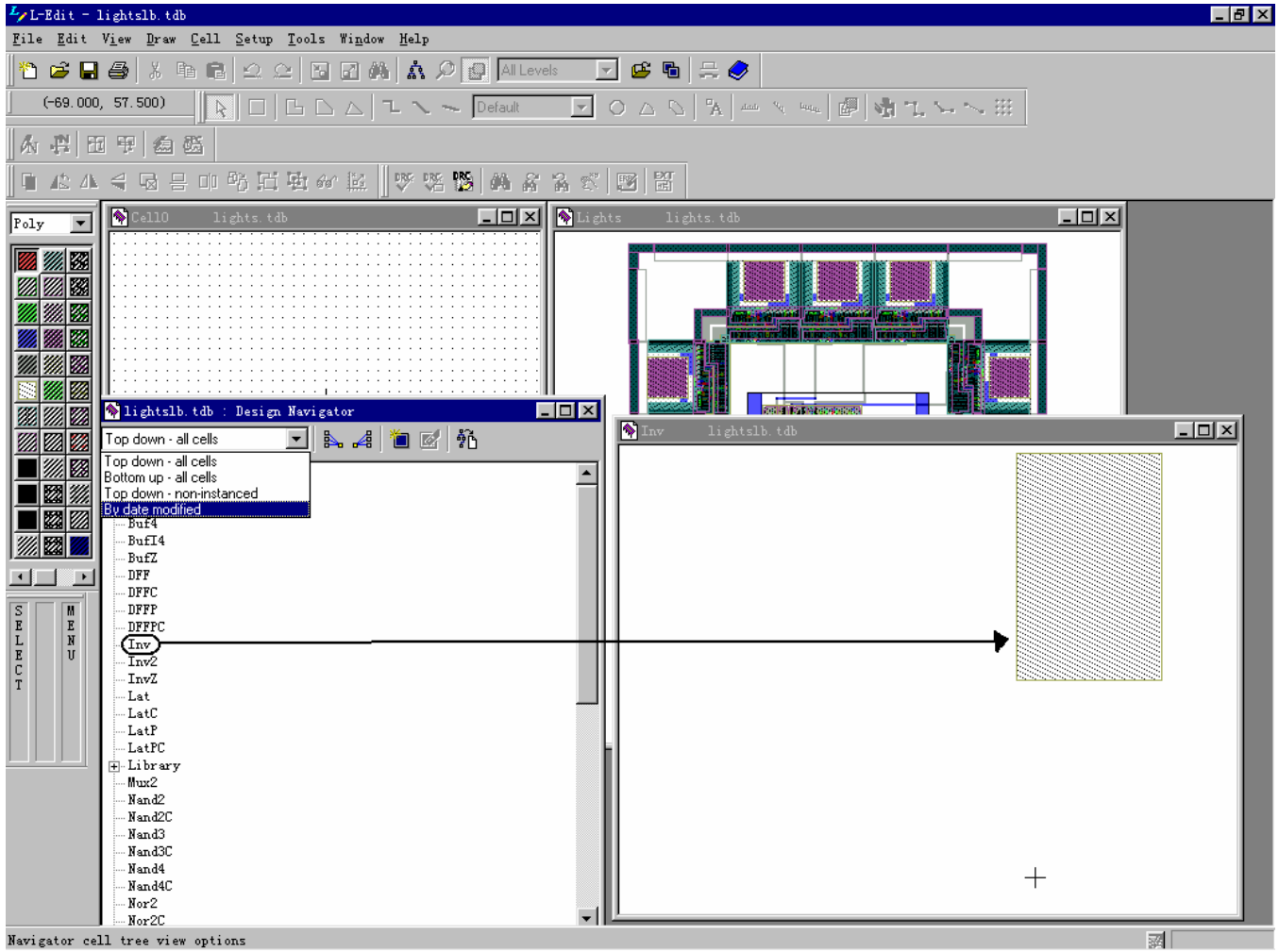


图 5-10 打开“Example”中“Inv”版图

(5) 为便于操作，可先将其它与“Inv”无关的窗口关闭，再将鼠标光标移至“层单元颜色区”。例如，指向红色的“Poly”层， 右击弹出下拉菜单， 其中第一行命令为“√Show Poly”，它表示当前激活并选中的是显示“多晶硅层”。 如果选择下拉菜单中的“Hide All”，则将所有层都隐藏；反之，选“Show All”，则所有的层都显示，详见图5-11所示。一般应用时，先隐藏全部再逐一显示各层。

(6) 选取“Samples”文件夹下的其它子文件夹中的文件，再分别演示和熟悉诸如DRC等操作步骤。

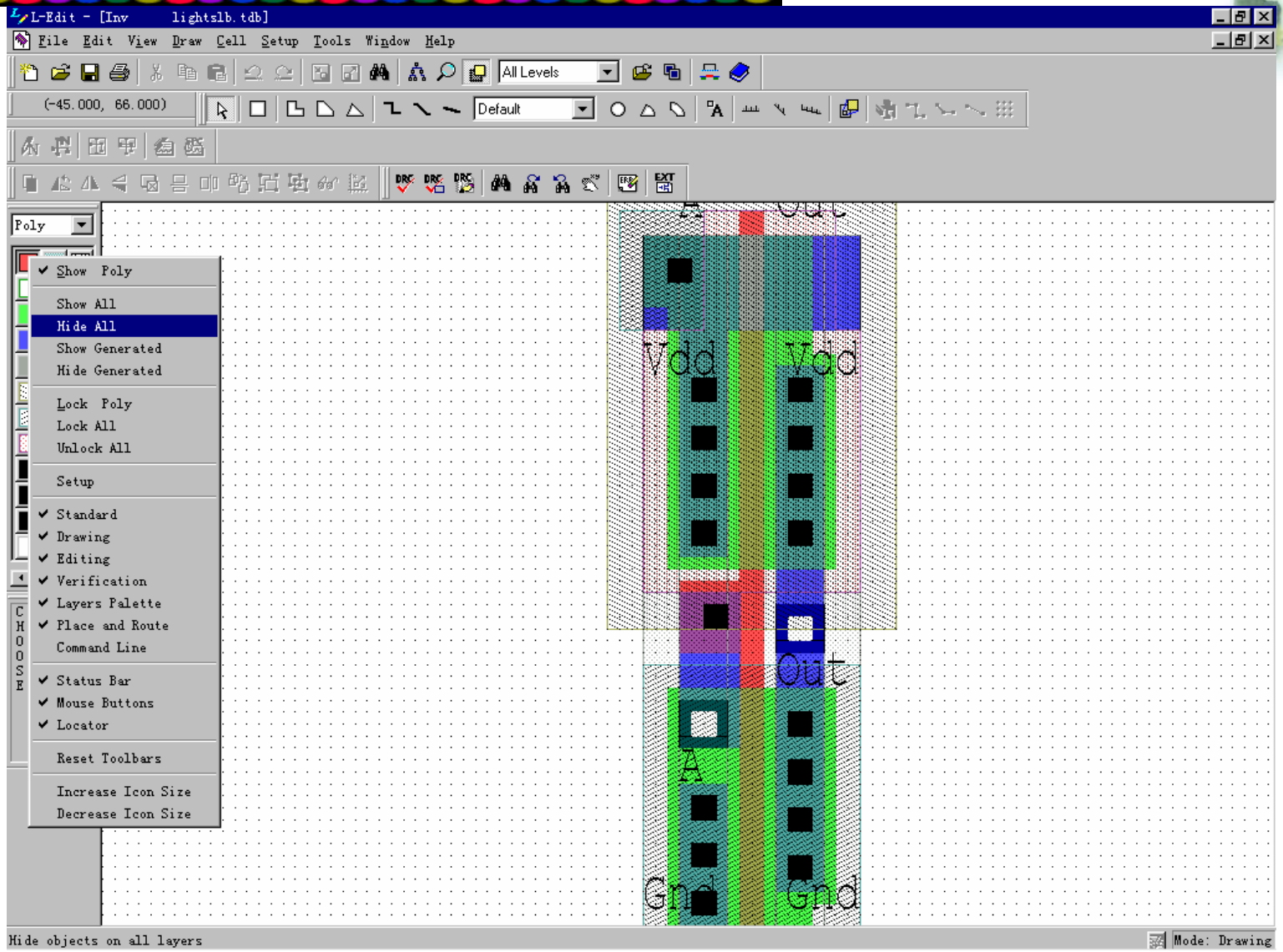


图 5-11 “层”打开和隐藏命令示意图

5.2.2 安装DOS版L-edit 5.0

L-edit Pro 9.0/8.xx功能繁多， 初学者如果没有特殊的要求，可以选用L-edit 5.0，其大小只有几百KB。 其主要功能和基本用法和基于Windows的最新版类似，对于小规模版图，尤其是模拟集成电路版图设计，已能较好地完成设计任务。 其设计好的文件已经能被L-edit Pro 8.xx以上版本读出(L-edit Pro 7.xx以下版本不支持低版本)。

(1) 先下载L-edit 5.0(该版本是DOS版，可在Windows的DOS模式下运行)，将其安装到本地硬盘。

(2) 在桌面上建立一个快捷方式，修改属性，在运行命令项后， 加上“/V”的命令。

(3) 下面以DOS版L-edit为例介绍一些CMOS版图知识，这些知识也适用于L-edit Pro 9.0/8.xx。

5.2.3 版图编辑实践

(1) 学会L-edit各种命令的用法。打开如图5-12所示的CMOSLIB.TDB(L-edit 5.0中提供)， 观察其中的单元。

(2) 从CMOSLIB.TDB中独立提出两个单元的电路。该文件共有6个单元： Order、 HysDif、TcAmp、WRAmp、 FlwInt、WRMult。其中Order是必做的单元，另外几个可选做。

(3) 将其中的一个父单元Library转换成CIF 和 GDS格式，将有关内容填入到表5-4中空出的部分。

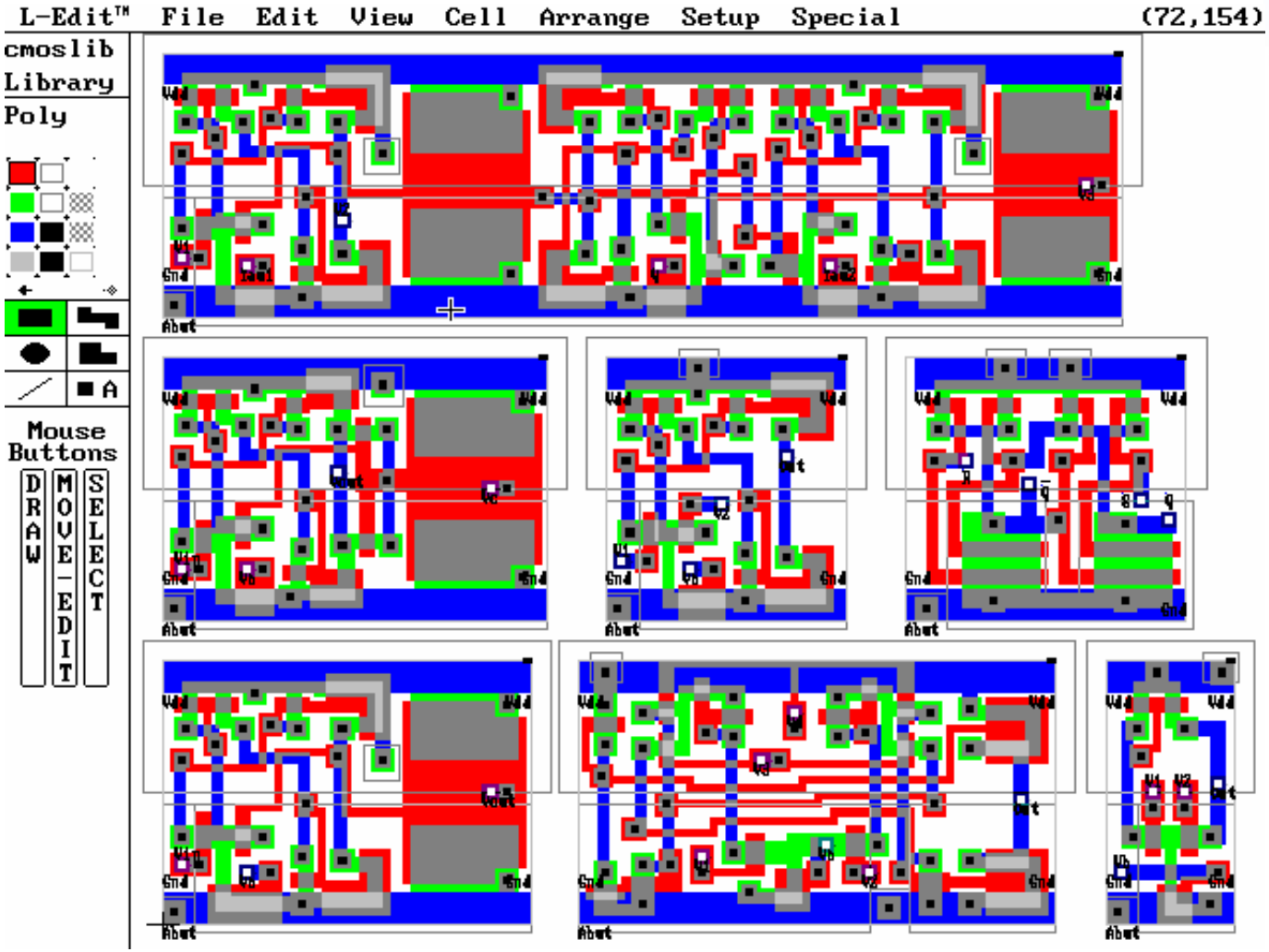


图5-12 L-edit 5.0下CMOSLIB.TDB示例

表5-4 Occupied Disk Volume

File Name	Disk Volume (byte)	Time of Creation	Route of the File
CMOSLIB. TDB			
CMOSLIB. CIF			
CMOSLIB. GDS			

5.2.4 读CMOSLIB.TDB的方法

表5-5 CMOSLIB.TDB版图层次表

层次	层次名称(英)	层次名称(中)	颜色
1	Poly	多晶硅	填满红色
2	Active	有源区	填满绿色
3	Metal 1	下铝，即 I-Al	填满蓝色
4	Metal 2	上铝，即 II-Al	填满灰色
5	N Select	N 选择扩散区，做 N 管用	草绿色
6	N Well	N 阱，做 P 管用	灰色
7	Poly Contact	多晶硅接触孔	填满黑色
8	Active Contact	有源区接触孔	填满黑色
9	Via	通孔，连接上、下铝用	全透明
12	Icon/Outline	拼接界的轮廓线	灰色

(1) 关闭所有层次，只打开多晶硅和有源区两层，相交区域就是管子，有几个相交区域就有几个管子，在纸上大概位置画出MOS管子符号， 并编上管号。

(2) 被多晶硅分开的每一块有源区，就是源区或漏区，它是绿色的。每一块绿色的区域具有同样的电位，在电路图上是同一个节点。与它的边相邻的多晶硅，都是管子，且必有源(或漏)与其相连，把它们画出来。

(3) 开Poly Contact。 它们都是在Poly 上的孔，即栅或Poly连线上的孔。

(4) 开Metal 1。 将栅或Poly连线上的孔连起来， Metal 1和Poly都是连线，画上。

(5) 再开Active Contact。把连接源区、漏区和多晶硅的线都画出来。

(6) 再开Metal 2、Via 和Icon/Outline。有Via的地方是输入或输出端口，写下端口的名字。

(7) 标出各管子的L/W，V_{DD}, GND， 一张电路图的草图就成了。 经核对无误后，稍加整理， 就完成了该电路图。

5.2.5 L-edit模块介绍

我们知道，L-edit 5.0是Tanner Research公司为开发ASIC而专门设计的版图编辑器，它是该公司软件系列Tanner Tools中的一个模块。 其它模块还有：

(1) 版图自动生成工具—L -edit/SPR。

(2) 验证工具—L -edit/DRC， Extract， LVS。

(3) 标准单元版图库— COMS3Lib，SCMOSLib，AnaCMOSLib。

(4) 门级时序仿真器—GateSim。

(5) 工艺映射库—Technology Mapping Library。

(6) 原理图网表转换工具—NetTran。

(7) 横截面观察工具—Cross Section Viewer（在给定工艺条件下，在版图上划出一条线(如一刀切下去似的)， 这样可以方便地观察到这条线下面的纵向结构图）。

5.2.6 L-edit主菜单使用导引

点击“L-edit”命令后，将给出L-edit的版本说明和有关的统计信息。 Edit命令功能最强大。View 包含所有的视窗命令。Cell中的命令全是有关单元的， 而单元设计的方法既便于修改,又能节省存储空间，所以很受欢迎。Arrange可对选择的图形进行旋转和镜面对称的操作。Setup供设置环境和Option 之用。Special包含了产生逻辑操作结果层次的命令，以及清除它们的命令， 执行DRC的命令。

(1) Edit命令功能强大。 如CIF (或GDS)格式的读入和转换就由它来完成: 读入(Open)版图文件时, 如它是CIF (或GDS)格式, 则选.CIF (或.GDS) 作后缀, 这时系统就将CIF (或GDS)格式转换成内部的TDB格式, 并显示出来。若需将TDB格式转成CIF (或GDS)格式， 则用Save as 命令, 并选.CIF (或.GDS)作后缀，这时系统就将内部的TDB格式转换成CIF (或GDS)格式，并保存于原文件目录中。CIF 和GDS格式的相互转换是通过TDB格式过渡来实现的。至于其它命令，使用方法很简单，这里不再赘述。

(2) View是视窗命令， 可用于放大、 缩小窗口，用鼠标随心所欲打开窗口以及上下左右移动窗口，且每个子命令都有相应的热键，十分方便。 另外， 是否要显示坐标、原点、格点、单元外轮廓线等, 都可由用户自己来选择。

(3) Cell是单元命令。设计新单元用New，修改原有单元用Open。 在修改原有单元时，如需返回到原单元，用Revert。在修改单元后用Close as另起名字保存，原单元依然存在。Delete用于删除单元。Rename用于对单元重命名。Copy在单元后进行修改， 可获得新单元。 Instance为调用单元提供方便。

(4) 假设名为Total的文件中包含了Totala、Totalb、 Totalc三个总图，它们各自有四个单元—(a1 ， a2 ， a3，a4) 、 (b1 ， b2 ， b3 ， b4)和（c1，c2，c3，c4 ），在输出CIF (或GDS)格式时，需用Fabricate来指定三个总图中的一个输出CIF (或GDS)格式。显然，Totala、 Totalb、 Totalc三个总图是父单元， 而(a1 ， a2 ， a3，a4)、 (b1 ， b2 ， b3 ， b4)和（c1，c2，c3，c4）都是子单元。父单元和子单元在库中都是平等的。 最好在命名时有所区别，才会调用方便，不混乱。

(5) 不得递归调用， 即子单元不能调用含其本身的父单元。

(6) Setup应用于设置环境和Option。设置比例因子用Technology。 设置格点用Grid。设置层逻辑操作用Derived Layer。设置具有等宽的线条用Wire (其中，还有圆角、方角、45度角三种选择)。设置颜色用调色板Palette。设置或取消层次用Layer。设置检查用的设计规则用DRC。做格式转换时，用CIF (或GDS)来规定层次表。

(7) Special将Setup已设置好的命令付诸实施。如执行层逻辑操作，用Derived Layer命令，执行设计规则检查操作，用DRC命令(可规定范围)。

5.2.7 DRC文件实例

```
;  example of DRC file  ; 1
drc Extra RULES((bkgnd=geomBKgnd())
    (L1=geomor("L1"))
    (L1=geomor("L1"))
    (L2=geomor("L2"))
    (L15=geomor("L15"))                    ; 5
    (L10=geomor("L10"))
    (L3=geomor("L3"))
    (L4=geomor("L4"))
    (L5=geomor("L5"))
    (L6=geomor("L6"))                      ; 10
    (L7=geomor("L7"))
    (L8=geomor("L8"))
    (L9=geomor("L9"))
    (L31=geomor("L31"))
```

```
(L32=geomor("L32")) ; 15
(mosC=geomor("mosC")
        L6a=geominside("L6 L5")
        L6b=geomandNot("L6 L6a")
        L7a=geominside("L7 L6a")
        L5b=geomOr("L5 L6")                    ; 20
        L5b=geomStraddle("L5 L3")
        L5c=geomAndNot("L5 L5b")
        L5d=geominside("L5 L3")
        Ivif((switch "drc?") then
        drc(L8 sep < 6)                        ; 25
        drc(L8 width < 9.5)
        drc(L3 L5c enc <19.8)
        drc(L3 L2 enc <29.5)
```

```
        drc(L3 L1 enc <19.5)
        drc(L3 L6 enc < 17.5)                    ; 30
        drc(L5 L6a enc < 3.5)
        drc(L5b L7 enc < 1.5)
        drc(L6a L7a enc < 3.5)
        drc(L5 L6b sep < 9.5)
        drc(L7 width < 8)                        ; 35
        drc(L5 width < 7.5 parallel)
        drc(L6 width < 7.5)
        drc(L5 sep < 9.5 parallel)
        drc(L8 L7 enc < 3.5)
        drc(L3 sep < 9.5)                        ; 40
        drc(L10 L7 enc <3.5)
    )
)                                                ; 44
```

(1) 第1行以分号开始，分号之后直至本行末尾，都是注解。第1行仅是标题，以下的分号仅用以指示行号。在Cadence系统中，drc文件、 Extracting 文件以及LVS 文件都是需要用户在UNIX下自己编写的。所用的语言是Skill， 它是Cadence系统为用户开发的专用语言。

(2) 第2行指出程序处理的数据范围，以能包含所有坐标数据的正交矩形为处理对象。

(3) 第3~16行对版图的原始层次进行“或”处理，目的是消除拼接处的冗余线。 有引号的层次是版图的原始层次，无引号的层次是程序推导出来的层次， 简称导出层次。

(4) 第16行中的“mosC”是层次的名字。层次的名字经常用L1、L2、 L3来表示。

(5) 第3~16行，首尾都有一对括号，它们是可以省略的。括号必须成对使用。如果你仔细观察，会发现第2行多了一个左括号，第24行也多了一个左括号，而第43、第44行则多了一个右括号。原来，第24行和第43行多余的括号配成了对；而第2行和第44行多余的括号也配成了对。前一对括号执行DRC操作， 后一对括号表示整个DRC操作全部完成了。

(6) 第24行是启动DRC操作的开关。 如果前面加了分号，DRC操作就被封锁了。

(7) 第17~23行是为以下的DRC操作做准备的。 如L6a = L5内部的L6上的图形，即发射区，L6b = L6的图形除去L6a外的图形，即磷扩散层上除去发射区(L6a)外的图形， 包括集电区、磷桥、岛上高电位接触孔所在的区域。 L6a和L6b遵循的设计规则是不同的，所以，有必要将它们区别开来。

这一段包含逻辑操作和依位置关系来提取图形的操作，定义如下：

geomOr(Li Lj) =Li“或”Lj上的图形，与 Li、 Lj的次序无关。

geomAnd(Li Lj) =Li“与”Lj上的图形，与 Li、 Lj的次序无关。

geomAndNot(Li Lj) =Li“减去”(Li“与”Lj)，与 Li、 Lj的次序有关。

geomInside(Li Lj) =提取在Lj “内部的”且在Li 上的图形， 与 Li、 Lj的次序有关。

geomStraddle(Li Lj) =提取与Lj “骑跨的”且在Li 上的图形，与 Li、Lj的次序有关。

(8) 第25~41行是具体的DRC操作，意义如下:

第25行，铝线间隔小于6时出错，L8是铝线。

第26行，铝线宽度小于9.5时出错。

第27行，隔离岛将基区包围，且套刻的距离小于19.8时出错。L3为隔离岛，L5为基区。

第28行，隔离岛将深磷包围，且套刻的距离小于29.5时出错。L2为深磷。

第29行，隔离岛将N+埋层包围，且套刻的距离小于19.5时出错。L1为N+埋层。

第30行，隔离岛将磷扩散区包围，且套刻的距离小于17.5时出错。L6为磷扩散区。

第31行，基区将发射区包围，且套刻的距离小于3.5时出错。L6a为发射区。

第32行，基区或发射区将接触孔包围，且套刻的距离小于1.5时出错。L7为接触孔。

第33行，发射区将发射区接触孔包围，且套刻的距离小于3.5时出错。L6a为发射区，L7a为发射区接触孔。

第34行，基区和集电区间隔小于9.5时出错。L6b包括：集电区、磷桥、岛上高电位接触孔所在的区域。

第35行，接触孔宽度小于8时出错。

第36行， 基区宽度小于7.5时出错。 只检查平行对边之间的宽度。

第37行， 磷扩散区宽度小于7.5时出错。

第38行， 基区图形间隔小于9.5时出错， 只检查平行对边之间的间隔。

第39行， 铝线将接触孔包围，且套刻的距离小于3.5时出错。

第40行， 隔离岛图形间隔小于9.5时出错。 换言之，隔离槽宽度小于9.5时出错。

第41行，测试铝线将接触孔包围，且套刻的距离小于3.5时出错。 L10为测试铝线。

5.3 版图设计流程和方法研究

5.3.1 IC Craftsman版图软件介绍

IC Craftsman与其它处理版图的系列软件的不同之处在于， 允许手工和自动布线单独、交互、混合使用，并且能够在走线规定、 Pin的线宽和间距和延时的设定等方面随时体现设计者的设计思想。其价值在于以下方面：设计者只需进行布局和规划， ICC则可以帮助设计者做数字化工作；在手动布线时自动做DRC、 LVS； 节约DRC、 LVS的执行时间等。

ICC的尺寸工艺： Wire的存储是以形状存储（左上、右下），而不是网格式的存储（网格尺寸，横纵坐标形式）。这种工艺的优点在于可探出前方走线的宽度，更重要的是节省了大量内存。

ICC在布线时自动考虑了线宽与其它部件的间距。如在已布好的几条线中再布上若干条线，当后布的线走动时，其余的线自动按工艺尺寸闪开(像磁铁同极相斥)； 布线时，走线的起点、 终点都已确定（有条细线相连），不必怕连错端点；有时走线遇到拐弯，由Critic Route自动优化走线。

ICC最多可以布255层线。ICC有背景设置，在布某层线时， 可将其它层线屏蔽掉，以便于观察，防止花眼。在块(Block)的布局布线中，可以自动调整沟道(Channel)的宽度，使器件利用率达到最优。

ICC的尺寸工艺：Wire的存储是以形状存储（左上、右下），而不是网格式的存储（网格尺寸，横纵坐标形式）。这种工艺的优点在于可探出前方走线的宽度，更重要的是节省了大量内存。

ICC在布线时自动考虑了线宽与其它部件的间距。如在已布好的几条线中再布上若干条线，当后布的线走动时，其余的线自动按工艺尺寸闪开(像磁铁同极相斥)；布线时，走线的起点、终点都已确定（有条细线相连），不必怕连错端点；有时走线遇到拐弯，由Critic Route自动优化走线。

ICC最多可以布255层线。ICC有背景设置，在布某层线时，可将其它层线屏蔽掉，以便于观察，防止花眼。在块(Block)的布局布线中，可以自动调整沟道(Channel)的宽度，使器件利用率达到最优。

5.3.2 布局设计

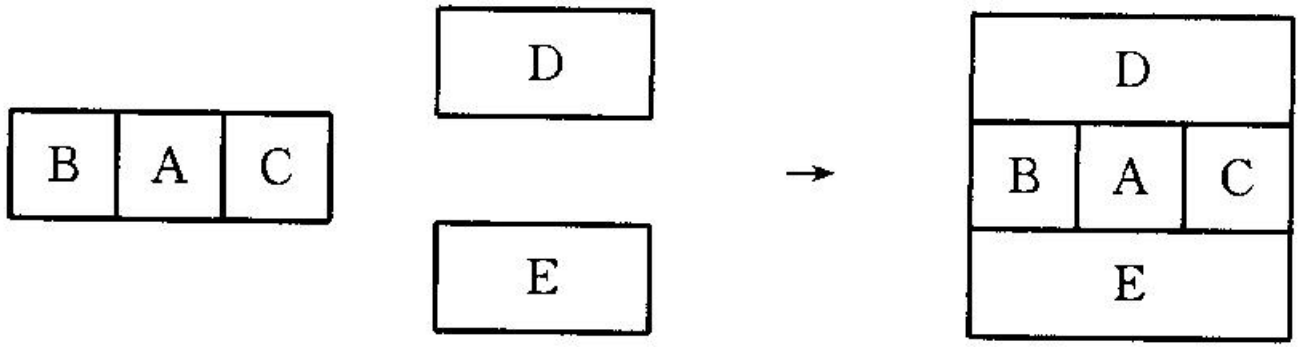
(1) 成对拼接法： 即将版图的构成部分进行两两放置，然后将这些成对单元再依次两两进行放置，依照此法继续进行下去， 直至版图形成。

(2) 集中展开法：是从所有部件中选取一个接线较多又比较大的部件，先行放置，然后以此为中心，相关部件在其四周排列， 直到所有部件布置完毕。

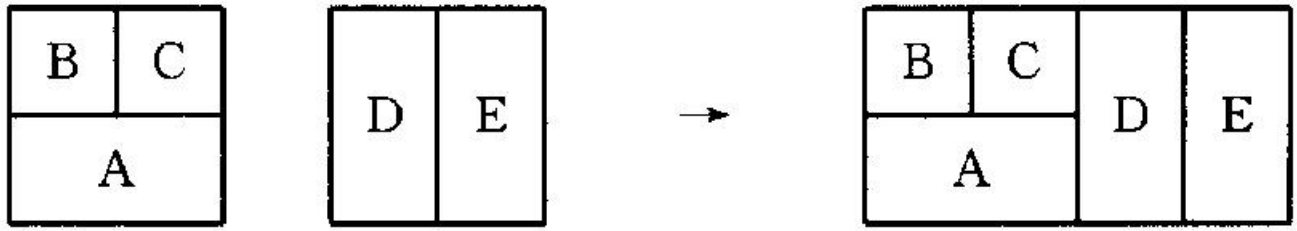
(3) 群体拼接法：群体拼接法与前两种方法不同，它是依据版图部件的相关关系，先将联系紧密、走线较多的部件分部分放置，构成部件组合单元，再对这些部件组合单元进行布置， 以形成模块版图。

成对拼接法主要用于部件版图形状大小相近的模块， 其图示如下：

集成展开法主要用于部件版图尺寸种类较少的模块， 其图示如下：



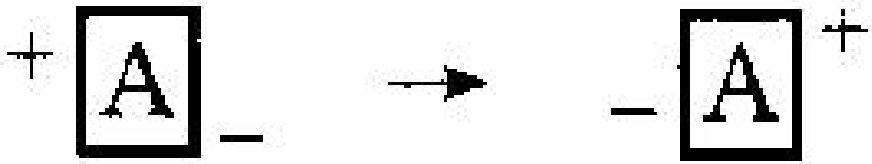
群体拼接法主要用于部件版图尺寸种类较多的模块， 其图示如下：



布局优化是对初始布局加以改善，以得到较好的布局，其实质就是将模块部件的位置进行交换，形成一个新的布局，然后与初始布局进行比较，选择一个较好的布局。 本文给出两种优化布局的方法：对称交换法和位置颠倒法。对称交换法就是将处于对称位置上的部件进行交换，以得到新的布局；位置颠倒法就是将模块中某些部件在原位置进行上下颠倒或左右颠倒或旋转位置而得到新的布局。另外，若对初始布局不满意，还可以对部件重新进行排列组合， 得到新布局。

对称交换法的图示如下：

位置颠倒法的图示如下：



5.3.3布线设计

布线设计就是把位置确定的各个模块部件按照工艺规则和电路要求用互连线连接起来，在尽可能小的区域内实现模块部件的物理连接。

布线设计与布局设计相似，也分三个层次：最小成分内的布线，子模块内的布线，模块集成布线。布线应遵循两个原则：信号流经总线应比较短， 布线分布应比较均匀。布线在版图设计中占有相当重要的位置，布线的优劣将直接影响到模块版图面积的大小，线的长度是影响延时的一个重要因素，因此，版图设计的一切环节都应该以布线为核心，兼顾布局。

布线设计不仅重要，而且方法也相当灵活， 布线的方向和层次也不惟一，要视具体情况而定。下面给出三种布线方法。

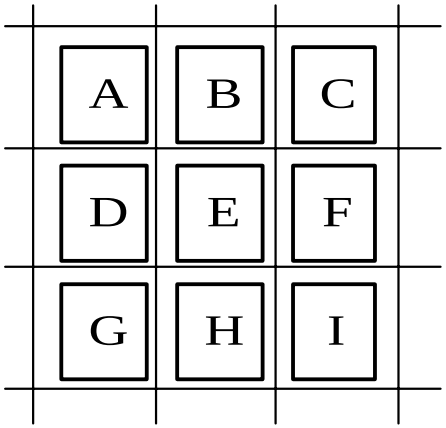
(1) 网络布线法：网络布线法就是在优化后的模块布局中依据横向和纵向两个方向，开出均匀的布线通道进行布线。这种方法一般适用于器件版图比较单一、规则且重复性极强的模块版图。

(2) 干线支线法：干线支线法就是在布线中先确定一些比较长的线作为干线， 然后以这些干线为基础向四面展开连其支线。 这种方法比较适合于最小成分的布线。

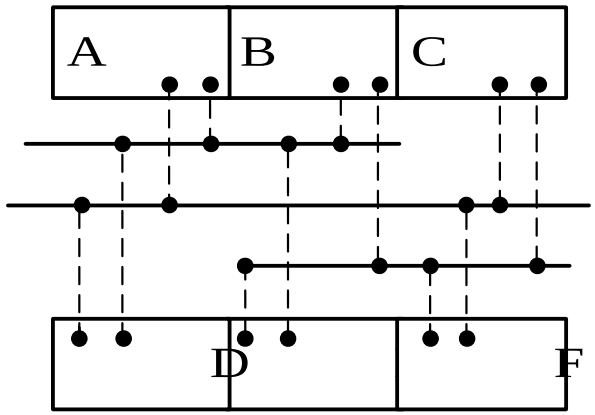
(3) 迂回布线法：迂回布线法就是将线伸到有空的地方弯曲通过，或将需要的不同类的两条线伸到有空隙的地方，进行打孔相连。这种方法适用于解决一些需要直接相连而不能走通的布线。

三种布线法的示意如图5-13所示。

网络布线法



干线支线法



迂回布线法

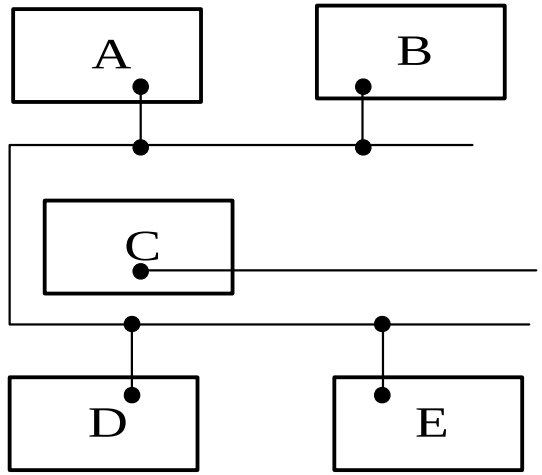


图 5-13 三种布线法的示意图

5.3.4 制作器件版图

1. 源漏共用

由CMOS工艺可知，基本管子的构成包括栅、源、漏三个部分， 这三个部分也就是信号的输入和输出端， 由此可将相邻管子的上级输出端和下级输入端结合起来， 构成源漏共用。也可将相邻管子电源或地端连在一起，形成管子电源端或地端的共用。采用这种共用方法可以大大减少器件的尺寸，同进也改善了器件的性能。

例如， 由两级MUX元件所构成的如图5-14(a)所示的器件版图，采用的就是上级MUX输出端漏极与下级MUX输入端源极的共用。

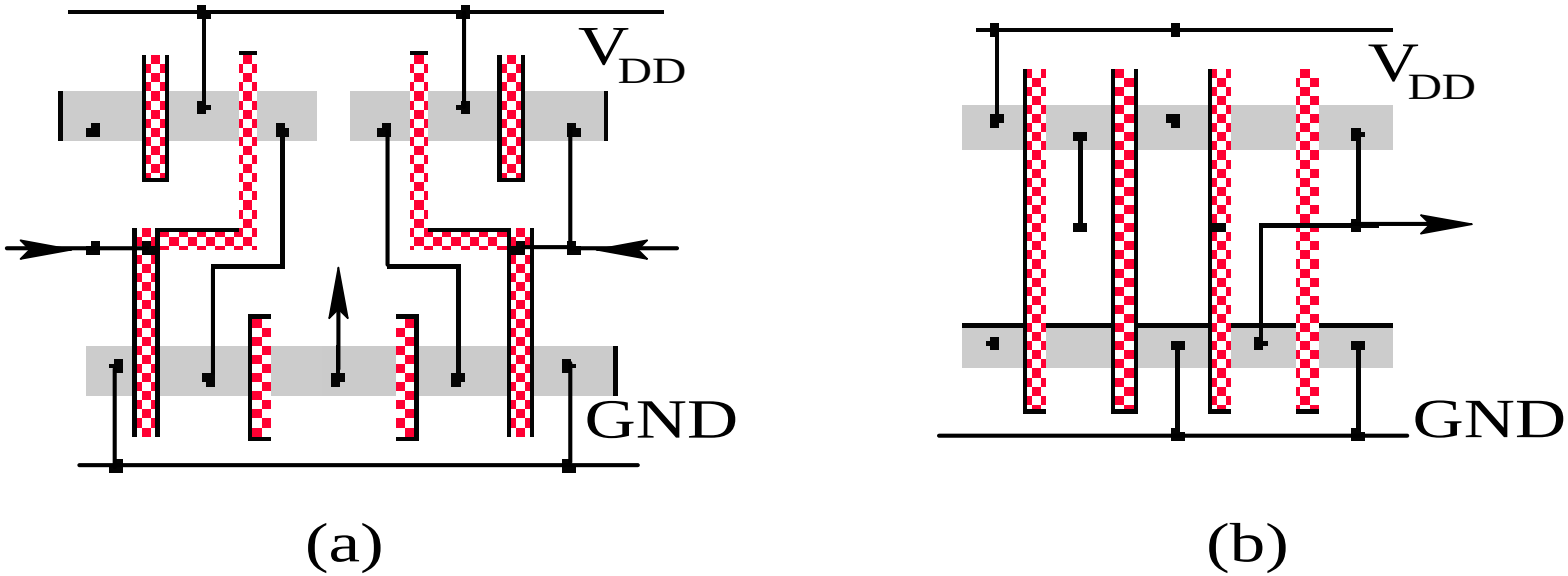


图5-14 源漏共用示意图

2. 版图与走线相结合

对于同一种器件， 版图的连接形式可能不同，究竟采用哪种连接形式，这要由版图上的走线来确定(包括版图器件的输入输出线)。 由于器件版图的输入输出端往往要打孔连接，而孔在走线中的覆盖尺寸比走线本身的宽度要大得多，因此，尽量使孔在器件版图中均匀分布，充分利用间隙，是减小器件版图面积的一个重要因素。

在由四输入与门构成的4-16译码器中，ND2的版图形式可有图5-15所示的两种形式。为了与其相邻器件NR2合理结合，应采用第二种形式。

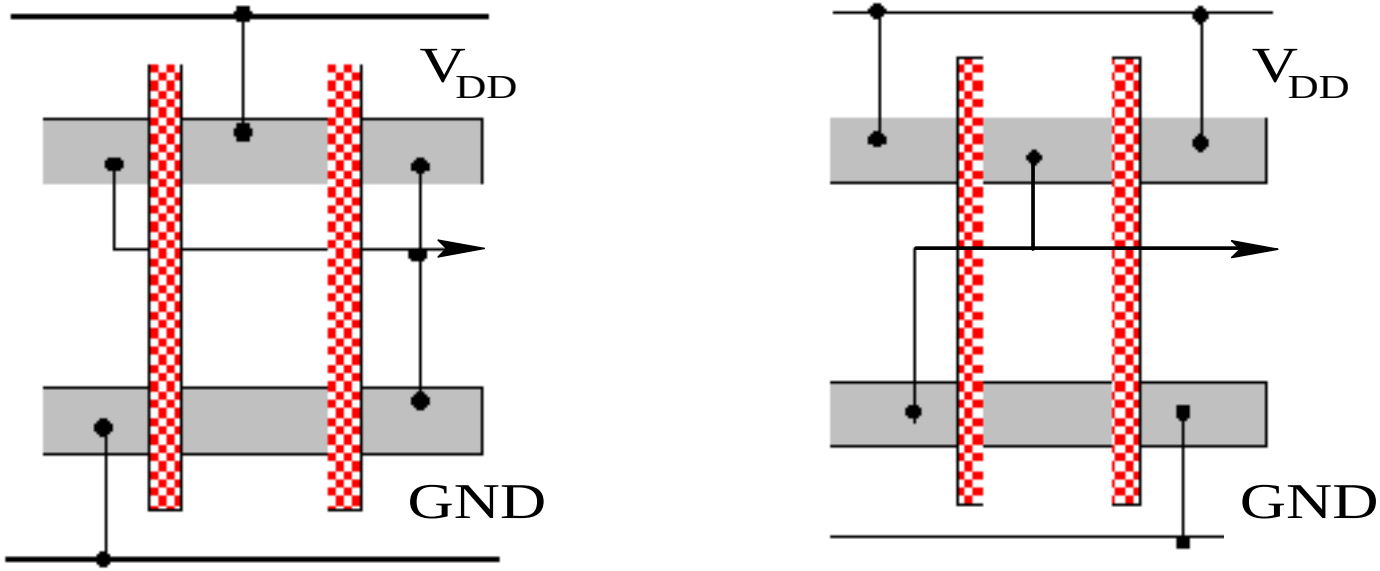


图5-15 走线变化示意图

3. 变换管子形式

在版图器件相拼接时，为连接方便，可变换管子的结构形式，以此来达到减小面积的目的。变换管子的形式要考虑两个方面的因素， 即器件版图的宽与高。一般来说，在一个方向上压缩会导致在在另一个方向上的扩展，因此必须综合权衡。下面从两个方面阐述：

(1) 变换管子的结构如图5-16所示。当管子宽度较大时，可改为右边的结构，这样虽在宽度上有一点扩展，但在高度上得到了较大压缩， 因此， 整个器件版图的面积将会减小。

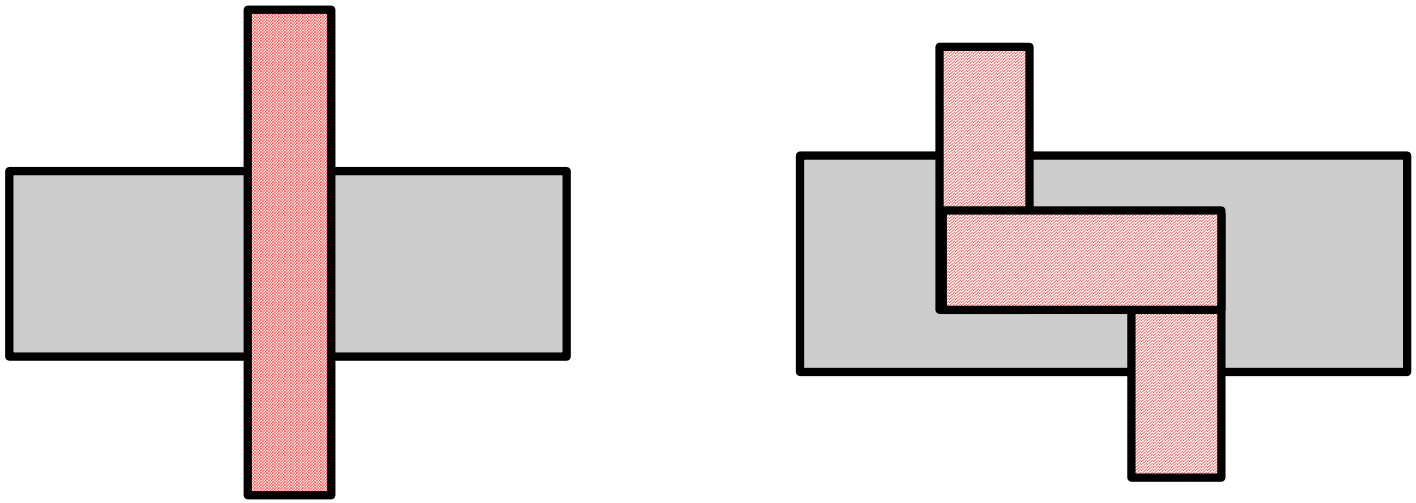


图5-16 变换管子示意图

(2) 对于宽长比较大的IV器件，可将图5-17中左图器件版图改为右图。由于IV的P管一般比N管要大，刚好利用IV组合间隙进行打孔，且所打的孔避免了和P管与N管间横贯的铝线(Metal1)发生矛盾。这样做的目的是为了降低器件版图高度。此方法适用于器件所在位置宽度要求较低的情形。

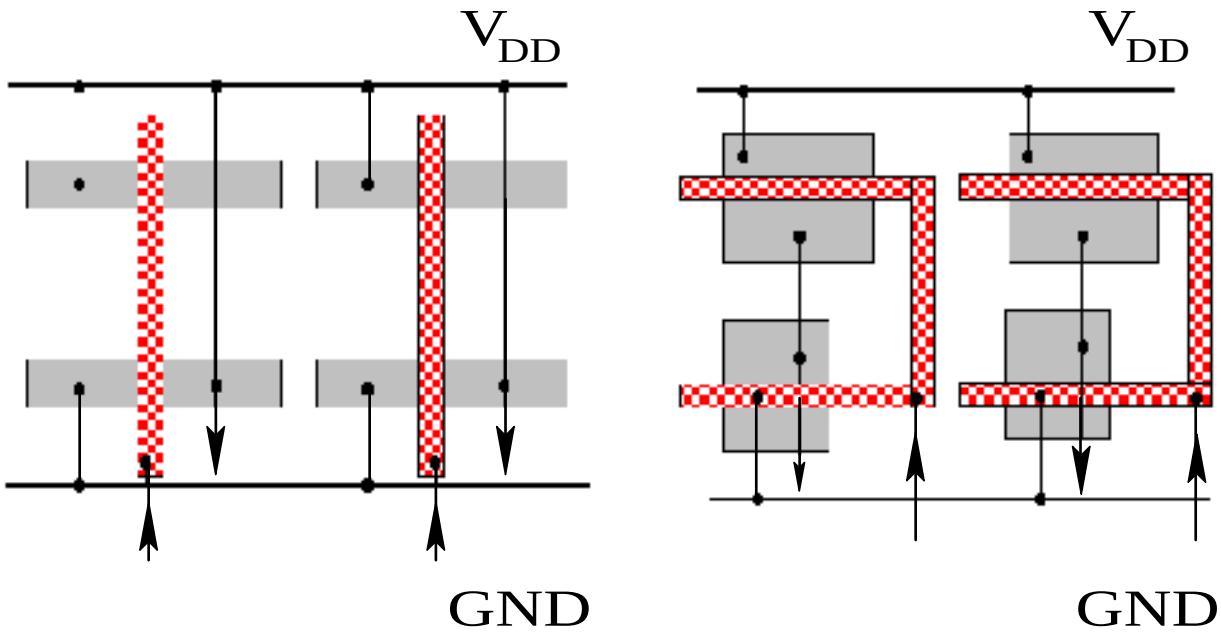
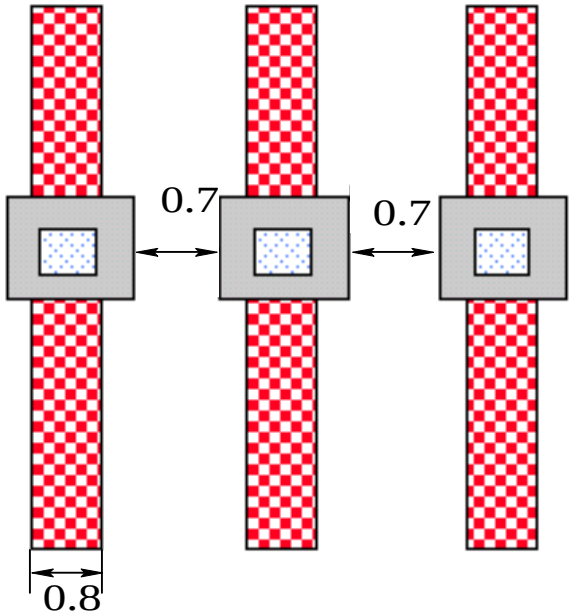


图5-17 变换宽长比示意图

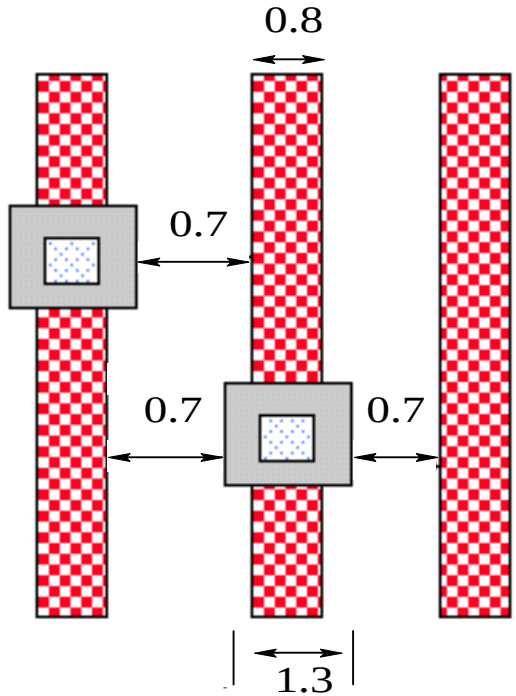
5.3.5 形成最小成分版图

器件版图完成以后，就可以依据布局布线方案，进行器件版图的连接，形成最小成分版图。最小成分版图的优劣可以反映模块版图的优劣，因此，最小成分版图和器件版图一样，也要精心制作。由器件相连形成最小成分， 应特别注意器件的尺寸与走线的关系，以及接头处所产生的弯曲问题。下面以LG公司的工艺规则为参照进行讨论。

在LG设计规则中，Metal2的宽度最小为0.8 μm，线间距最小为0.7 μm，打孔覆盖宽度为1.3 μm。当有成批条线从器件上经过时，每条线都要与器件打孔相连。若这些孔彼此能够错开， 则单线实际占据宽度为1.75 μm；若有孔无法错开，则该线实际占据宽度为2 μm。因此，当一条线上有众多孔时，让走线紧靠覆盖区边沿比让孔居线中要好；当有数条线错位相接时，用斜线相接比用弯曲直线相接走通的可能性要大，这是因为前者接线占据高度较小，如图5-18所示。图5-18中，表示孔覆盖。



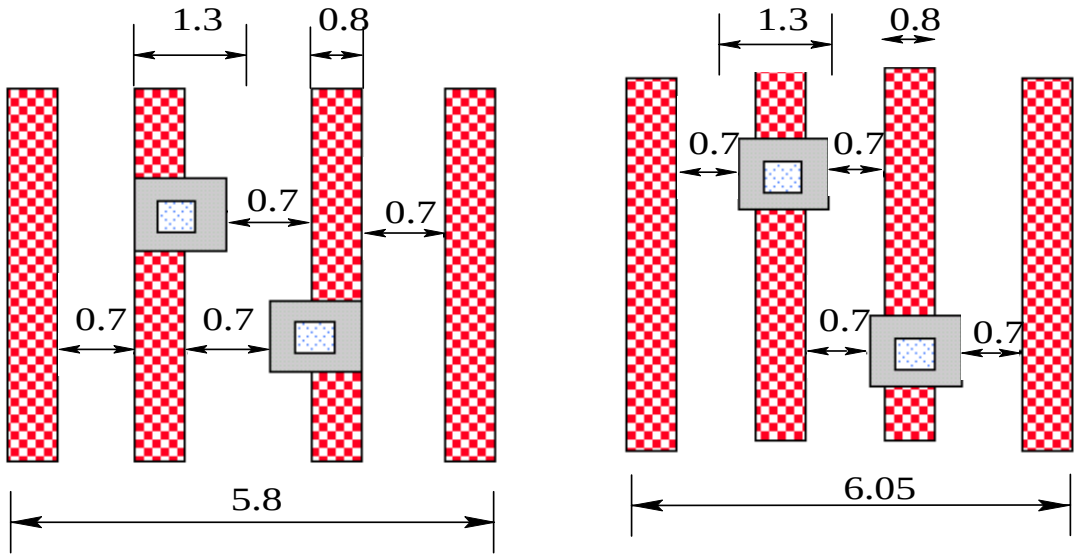
占据宽度为1.3μm＋0.7μm＝2μm



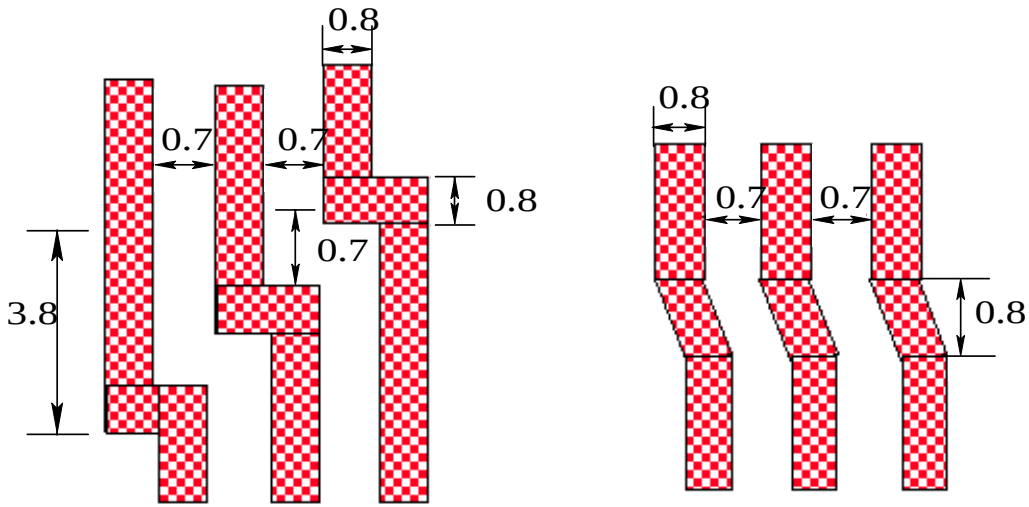
占据宽度为0.8μm＋0.25μm＋0.7 mm＝1.75μm

图5-18 Meta12占据宽度示意图

由图5-19可以得到n条走线占据宽度的估算结果为1.75 n或2 n。同样，由图5-20也可以得到基本器件版图宽度的估算公式为1.8x+1.2+0.4y(x表示N管数目，y表示宽度小于1.2 μm的N管数目)。



走线打孔比较图



连线比较图

图5-19 连线比较示意图

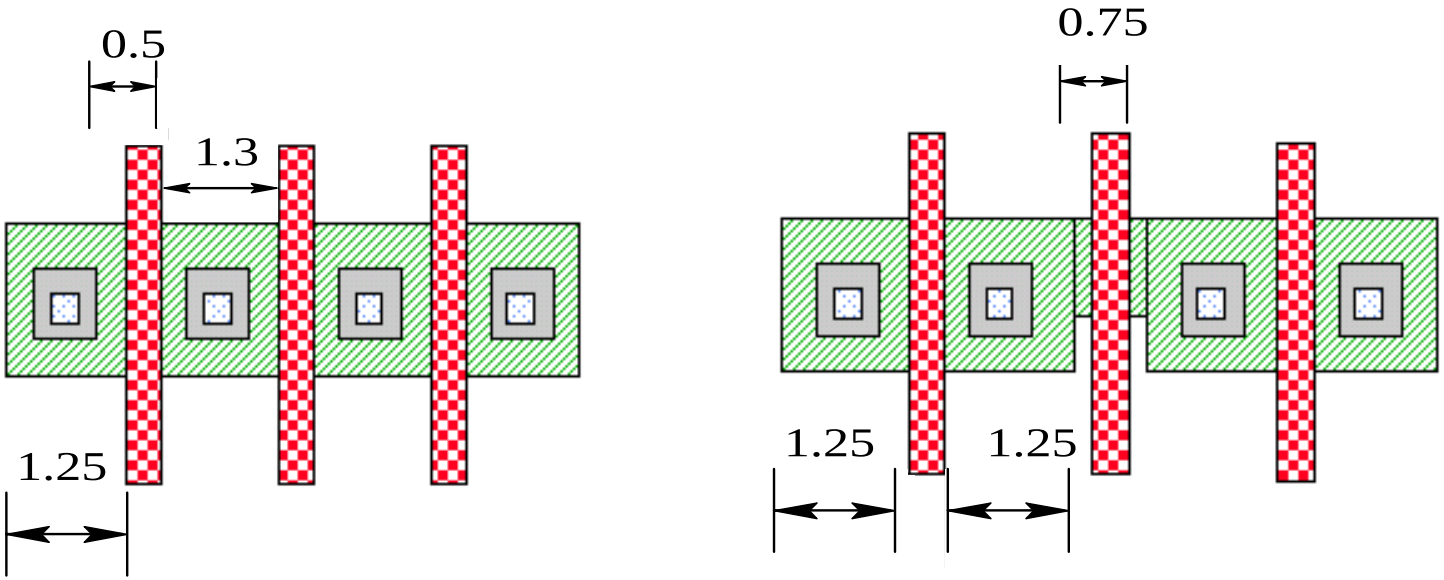


图5-20 基本器件版图宽度示意图

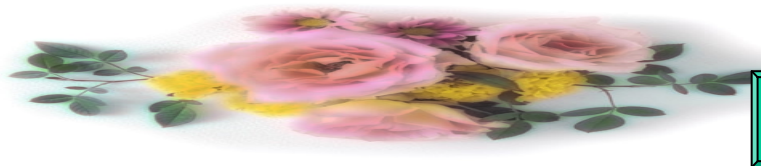
5.3.6 版图集成

版图集成就是由最小成分版图连接形成子模块，再由子模块连接形成模块版图的过程。版图集成是版图形成的最后一个环节， 这期间最重要的就是版图的连接。有效的连接方法是版图能够迅速完成和最终版图正确的保证。 版图连接除可以利用ICC布线软件进行以外，手工连接也非常重要。这里给出一个手工连接的方法—指数倍增连接， 也就是先将两个最小成分进行连接，连接时对于同等位置的连线实行翻倍拷贝连接，然后再将这种连接组合体两两进行连接， 直至连接完毕。

5.3.7 版图检测与调整

版图形成后，再对版图进行正确性验证，以保证版图的正确无误。版图检测一般包括以下四个环节：设计规则检查(DRC)，电学规则检查(ERC)， 版图参数提取(PRE)，版图与电路图一致性检查(LVS)。通过版图检测对版图进行修正和调整， 包括制作器件版图直至形成模块版图的整个过程。

版图检测无误后，要进行Hspice 逻辑功能验证，若不合格， 需对版图管子宽长比进行调整， 直至版图符合逻辑。



5.4 版图生成、验证

5.4.1 DataPath设计

DataPath设计的前期工作包括DataPath库准备、 Schematic准备及装载工艺条件三项。DataPath库准备：在启动Cadence集成环境后，在CIW窗口选择Set Library Path，将/user5/PubLib设置到搜索路径列表中，随即在Library Browser中可看到DataPath库。Schematic准备：运行SmartPath的Schematic与 普通的Schematic略有不同。要将所有DataPath的单元统一用DataPath库中的专用单元来表示。装载工艺文件：在CIW菜单选择*Design Manage→Technology File→Compile Technology...* 后，出现Compile Technology File表格。 在Library Name一栏填写当前库的库名；在Technology File一栏填写工艺文件所在的全路径及文件名。

1. 处理Schematic

1) 打开设计的器件Cell的Schematic

(1) 设置当前库所在的路径。 在 CIW菜单选择Design Manage→Set Search Path后，再在出现菜单的Path项内填入你的Library的路径。

(2) 在CIW菜单选择Design Manage→Library Browser。

(3) 在Library Browser中用鼠标左键单击你的库。在库名的后面会显示库中所有器件Cell。

(4) 用鼠标左键单击器件Cell 。 在Cell的后面会显示CellView。

(5) 用鼠标右键点中 Schematic， 在弹出的菜单中选择Edit， 然后松开鼠标。

2) 运行SmartPath

在Schematic窗口选择Tools→DataPath/Schematic,

DataPath菜单将出现在主菜单条的最右边。

3) 初始化DataPath生成器

(1) 选择DataPath→*Initial Generators*。

(2) 接受默认的设置， 单击OK按钮。

4) 在Schematic中加入DataPath

(1) 选择DataPath→Add Generated Component...。

(2) 在Category中选择Buffer。

(3) 在窗口中选择Inv。

(4) 填写位宽。

(5) 选择驱动。

(6) 单击OK按钮。

5) 运行PR Flatten

(1) 在CIW窗口选择 Translators→PR Flatten。

(2) 填写库名、 Cell名、 Cellview名时，用Browser按钮进行相关操作。

(3) 单击Browser按钮。

(4) 在Library Browser窗口单击当前Cell的Scheamtic。

(5) 在PR Flatten表格的底部选择Generate Physical Hierarchy，将生成该Cell的autoLayout View。

(6) 如果出现一个是否覆盖已有的autoLayout View的询问，则单击OK按钮。

6) 关闭Schematic编辑窗口

在Schematic编辑窗口选择Windows→ Close。

2. 对设计模块中的DataPath进行平面规划与布局

1) 以Edit方式打开autoLayout View

选择Edit菜单中autoLayoutView命令， 使其处于激活状态。

2) 运行SmartPath与Cell3

(1) 选择Tools→Floorplan→DataPath/Cell3 Ensemble，Preview Cell3与DataPath的菜单将出现在主菜单条上。

(2) 生成布局布线格式。选择Floorplan→Reinitialize，保持默认的设置，单击OK按钮。

3) 建立DataPath Function

选择DataPath→*Create DataPath Function*, 保持默认设置, 单击OK按钮之后, 在autoLayout View中属于DataPath Function的Cell全部排列在Block的下部。

4) 建立DataPath Region

(1) 选择 DataPath→Create DataPath Region， 填写表格:

RegionName :	用默认的值
Functions To Assign:	All unassigned
Bit Order, :	Bit 0 At Bottom
Row Flipping :	flip every other row, flip bit 0不选中
Row Per Bit:	1
Region Origin:	Offset from Boundary
	Offset: 10 10
Auto-resize Boundary:	选中

(2) 单击OK按钮退出。

3. 设置线的边约束Net Side Constrain

(1) 建立线边约束文件Net Side Constrains File。选择DataPath→Set Placement Constrain→Set Net Side Constrain, 再选择Write and Edit, 生成包含当前线边约束的文件，单击Apply。

(2) 在出现的VI程序的编辑窗口做适当修改后退出。

(3) 若有修改, 并想使之有效, 则单击Read, 然后单击OK按钮退出。

4. 对DataPath进行布局

(1) 选择DataPath→*Placement*。

(2) 保持默认的设置，单击OK按钮。如果成功，在CIW窗口将提示Placement Completed Successfully。

5. 为非DataPath Cells增加Row

如果该设计模块中全部是DataPath单元，则此步骤及对非DataPath单元布局的操作步骤可以跳过。

(1) 选择DataPath→Add Rows。要估算为非DataPath Cells增加了多少Row，先将这些未被分配的Cell单元加到DataPath Region中。

(2) 选中Include Unassigned Cells。

(3) 估算加一个Row后的利用率。 如:

Add Row Above DataPath: 1

单击Compute Utilization检查Row的利用率Row Utilization。若太大，则增加Row，再算，直到利用率低于80%。

(4) 如果要加大增加的Row的利用率，还可以少加Row， 将DataPath的Row向右扩展一些，扩展的量根据单元的多少而定。如：

Add Row Above DataPath: 1

Extend Rows To Right: 25, 默认的单位为Micron

单击Compute Utilization， 调整直到Row Utilization 比Row 80%略大。

(5) 单击OK按钮。

(6) 保存中间结果， 另存为autoDPplaced。

6. 对非DataPath单元布局

1) 确定已经以Edit方式打开autoLayout View

如果不是以Edit方式打开，则选择Edit菜单中autoLayout View命令， 使其处于激活状态。

2) 建立Cell3环境

(1) 选择 Place&Route→Engine→Environment。

(2) 将Memory Size改为48。

(3) 确定Send Design Parameters设为Send Cell Library(LEF) with Design。

(4) 在Technology From Library中填入当前库的库名。

(5) 单击OK按钮退出。

3) 对非DataPath单元布局

(1) 选择　Place&Route→Place。

(2) 对Log选项，选择Overwrite。

(3) 对Retrieve Design to View Name选项，填写autoPlaced。

(4) 对Pre　placed Instance Fixed选项，选择Yes。

(5) 单击OK按钮退出。

4) 退出非DataPath单元布局

布局完成后, Cell3将生成一个autoPlaced View，同时将会出现一个布局的Log文件窗口。浏览该Log文件窗口中的内容，有错的地方都标有“infos”。浏览完后关闭此窗口(选择File→Close)。

5) 关闭autoLayout View

选择Window→Close。

7. 整体布线

1) 打开完成布局的设计

以Edit方式打开autoPlaced。 如果有错， 选择Analyze→Show Markers。

2) 设置I/O边位置(Side)约束

通常，要将数据Pin放在上下部位， 将控制Pin放在左右部位。

(1) 搜索将要放在上边的数据Pin。选择Edit→Search。在Search窗口选择按钮Pin及In Cell View。在By子窗口选择Terminal Name，填入要放在左边的Pin名或通配符。点一下背景窗口，确定在编辑窗口还没有做任何选择。在OSW窗口，选择NS，再选择Pin。在Search窗口单击Select按钮。至此，要放在左边的Pin应已选中。

(2) 选择Edit→*Properties*。单击Common按钮。在Side Constrain位置选择Top，单击OK按钮。

(3) 搜索将要放在下边的数据Pin。重复操作以上两步。

(4) 将控制Pin分放在左右。操作步骤和数据Pin的操作方法一致。

3) 优化I/O Pin

(1) 优化Soft Pins。在Edit窗口，选择Floorplan→Soft Pins→Optimize/Float。单击Float。按下Switch Pin Layer 按钮，在Top/Bottom格填写met2，在Left/Right格填写met3,单击Optimize按钮。

(2) 检查并将Pin调整到Cell3的布线格上, Zoom In到设计的边界上，检查Soft Pin，注意Pin是放在设计的边界上的。在OSW窗口，先选择NS，再选择Design。在Edit窗口，点中设计的边界，选择Place&Route→Block→Adjust Pins。选中Snap Pins To Routing Grid, 单击OK按钮。

(3) 检查是否有重叠的Pin。选择Analyze→Check→Soft Pin Overlap。查看CIW窗口中的提示信息，如果没有重叠的Pin，将会出现提示信息End of Overlap。

4) 布Power Stripe

(1) 设置Special Net。选择Place&Route→Special Route→Set Nets Special。在Net Name List格中填写 vdd! gnd!，单击OK按钮。

(2) 布第一个Power Stripe。选择Place&Route→Special Route→Stripe。在随即出现的表格中填写或设置如下参数：

Retrieve Design to View Name:	autoStripe
Net Name List:	vdd!
Layer Name:	met2
Stripe Width:	4.5
Stripe Per Net:	1
Direction:	Vertical
Start:	4.6 (单击Point，再点击设计靠左边界的位置)
Step:	1
Stop:	同Start 的设置
Area:	All
Core Area:	All

(3) 以Edit 方式打开autoStripe。

(4) 布另一个Power Stripe。选择Place& Route→Special Route→Stripe。在随即出现的表格中填写或设置如下参数：

Retrieve Design to View Name: autoStripe

Net Name List: gnd!

Layer Name: met2

Stripe Width: 4.5

Stripe Per Net: 1

Direction: Vertical

Start: (单击Point，再点击设计靠右边界的位置)

Step: 1

Stop: 同Start 的设置

Area All

Core Area: All

单击OK按钮。

5) 当Stripe布完后，将出现Log File窗口，检查是否有Infos后关闭

(1) 将Pin移动到Power Stripe上。搜索vdd! Pin，并移动到vdd! Power Stripe上。选择Edit→Search。选择Pin，In Cellview，By Terminal Name。 填入vdd!。在OSW窗口选择Pin。点击Select，Zoom In到选中的vdd! Pin的位置。将vdd! Pin移动到vdd! Power Stripe的下端，单击Q键。在Edit Properties窗口, 将Pin Type 设为m2pin, 将Pin Layer Width 设为4.5，单击OK按钮。

(2) 搜索gnd! Pin，并移动到gnd! Power Stripe上。操作同上面vdd! Pin的操作一样。

6) 生成Power Rail

(1) 连接Power Pin 与Power Stripe。选择Place&Route→Special Route→Follow Pins。

在随即出现的表格中填写或设置如下参数:

Retrieve Design to View Name:	autoFollow
Net Name List:	vdd! gnd!
Layer Name:	met1
Direction:	Horizontal
Fill:	Unselected
Area:	All
Core Area:	All

单击OK按钮退出。

(2) 如果没有错误，在检查完后，关闭生成的Log File文件。

(3) 关闭autoStripe Edit窗口。

7) 整体布信号线 Globle Route

(1) 以Edit 方式打开autoFollow View。

(2) 选择Place&Route→Globle Route，然后单击OK按钮。

(3) 完成后将会出现Log File窗口，检查是否有错，最后关闭Log File文件窗口。

8) 布信号线的最终布线Final Route

(1) 选择Place&Route→Final Route，然后单击OK按钮。

(2) 完成后将会出现Log File窗口，检查是否有错，最后关闭Log File文件窗口。Cell3将生成autoRouted View，这就是最终的布局布线结果。

5.4.2 版图输入流程

本小节介绍Cadence的Layout Edit版图输入工具的使用方法， 包括版图输入的准备、LGS工艺规则简介、版图的设计原则及具体版图绘制中的应用方法、操作步骤。

在使用Layout Edit工具编辑版图前，必须有和该Cell相对应的与Schematic、Layout相关的工艺文件(Technology File)。 可根据工艺文件(Technology File)中的设计规则对版图进行编辑。Cadence另外提供有DIVA在线式验证及Dracula处理，使得在版图编辑时可暂不考虑设计规则，且不同工艺之间的版图转换也很方便。

Cadence的Layout Editor 工具有两种使用模式: 一种为专家模式; 一种为新手模式。

1. 输入的准备

(1) 启动LayoutPlus。启动Cadence后端仿真工具的命令是Layout Plus&。

(2) 敲入*icfb*&命令，出现CIW窗口。

(3) 在CIW中选中Open→Library，出现Open→Library窗口。

(4) 输入库名及库路径。在Open→Library窗口中输入库名及库路径。 注意，在Mode选择Edit时，若目标库不存在，系统会提示创建一个新库； 若目标库存在， 则将其打开。

2. 确定和修改Technology File

(1) 选择CIW窗口中的Technology File→Compile Technology，出现Compile Technology File窗口。

(2) 库名及装载Technology File。在Compile Technology File窗口中，填入库名和需要装载的Technology File。注意，在*Action*需选择类似*Load/NIS+user/Sysadm/ywh/work/techfile*的命令。

3. 创建或打开一个Cell

(1) 若目标库未打开，则要先打开目标库；若目标库已打开， 则在CIW中选择*Design Manager →Library Browser*， 此时会出现Library Browser窗口。

(2) 从Library Browser窗口中找到目标库， 用鼠标中键点击所选库名，在弹出的菜单中选择Create Cell项后松开鼠标，此时会出现Create Cell窗口。在Create Cell窗口中填写好Cell名。注意owner Access的read、edit、 delete的选择，然后单击OK按钮。

4. 创建CellView　Layout

用鼠标右键点击选中创建好的Cell。在弹出的菜单中选择Create CellView, 会出现Create CellView窗口，在View Name中填入Layout。

注意：owner Access的 read、edit、 delete的选择，然后单击OK按钮。在Library Browser窗口中用左键单击Cell名，则可见CellView的Layout存在。若目标库、目标Cell及CellView 的Layout早已存在， 则可跳过创建的步骤。

5. 版图的编辑

1) 关于图形的绘制

首先必须选中LSW中的一个定义层。可利用Layout窗口条进行所选层的绘制。

(1) 显示的设置。在Layout窗口中选择 *Design→Option→Display*, 将出现Display Options 窗口。在isplay Options窗口中, 可以进行显示设置。

例如，*Grid Controls* 通过Grid的设置，可以控制鼠标在Layout窗口中的定位精度。

X Snap Spacing .25

Y Snap Spacing .25

Snap Modes

Create Snap Mode L90XFirst

Edit Snap Mode anyAngle

Display Levels 通过Displaylevel的设置可以控制显示内容的多少

From 0

To 0

(2) 某工艺的设计规则。版图编辑的规则是工艺加工技术和器件物理对版图设计的约束，是芯片面积和成品率的折衷。设计规则融合了理论计算和经验数值。绘制版图必须首先看懂设计规则，版图编辑对照设计规则进行。其中well、diff、 poly、cont、metal、pimpat、via等层次的定义由设计规则确定（并在LSW中表达）。我们给定的某工艺是双阱工艺，管子的结构描述如下。

PMOS管的工艺结构：在N型氧化硅衬底上做P型阱（低浓度P型扩散区），再在P阱上做N型阱（低浓度N型扩散区），然后在N阱上注入高浓度P型扩散层有源区(Pdiff)， 并加入多晶硅(Poly)， 用金属及接触孔进行连线。

(3) 单元版图的设计步骤：电源/地线→信号线→晶体管→连接孔和衬底的接触→连线。

(4) 版图的绘制原则： 减小连线电容、 电阻， 减小面积。结合绘制原则， 具体绘制注意事项如下：

① 减小连线接到输出端的漏端的个数。

② 改变栅区形状， 增大晶体管的宽度， 减小面积。

③ 优化电路结构， 减小节点个数。

④ 布局优化：连接相同节点的器件尽可能靠近，连接相同节点的功能模块端口了尽可能靠近。这要求单元版图的绘制高度和Pin及电源/地位置相同。

⑤ 减少连线拐弯。

⑥ 减少总线绕圈。

⑦ 突出总线的设计。

2) 几类图形的绘制

(1) 多边形(well、np、pp、poly、metal)的绘制。在LSW中点well/dg、polygin，再在Edit窗口中根据设计规则绘图。

(2) Path的绘制。①同层相连。在LSW中用layer/dg .Path进行绘制。②不同层相连。 在命令窗口点中cont/via/via2，从起点层开始绘制，中间进行层间的切换，最后进行目标层的绘制。

(3) Pin的绘制。①单层Pin。在LSW中用layer/ pin.Rectangle polygin label 进行绘制。②双层Pin。在命令窗口点中cont/via/via2，从起点层开始绘制，中间进行层间的切换，至目标层的绘制。

(4) 选用Label命令加label。

5.4.3 MUX2的版图编辑步骤

1. MUX21 1的分析

(1) MUX21 1的组成分析。MUX21 1(二选一选通器)由INV(反相器)及NAND2(与非门)组成， 如图5-21所示。

(2) INV的组成分析。由CMOS构成的反相器如图5-22所示。反相器可由PMOS管、NMOS管构成，即INV的版图可由PMOS、NMOS的版图拼成（也可整体绘制）。

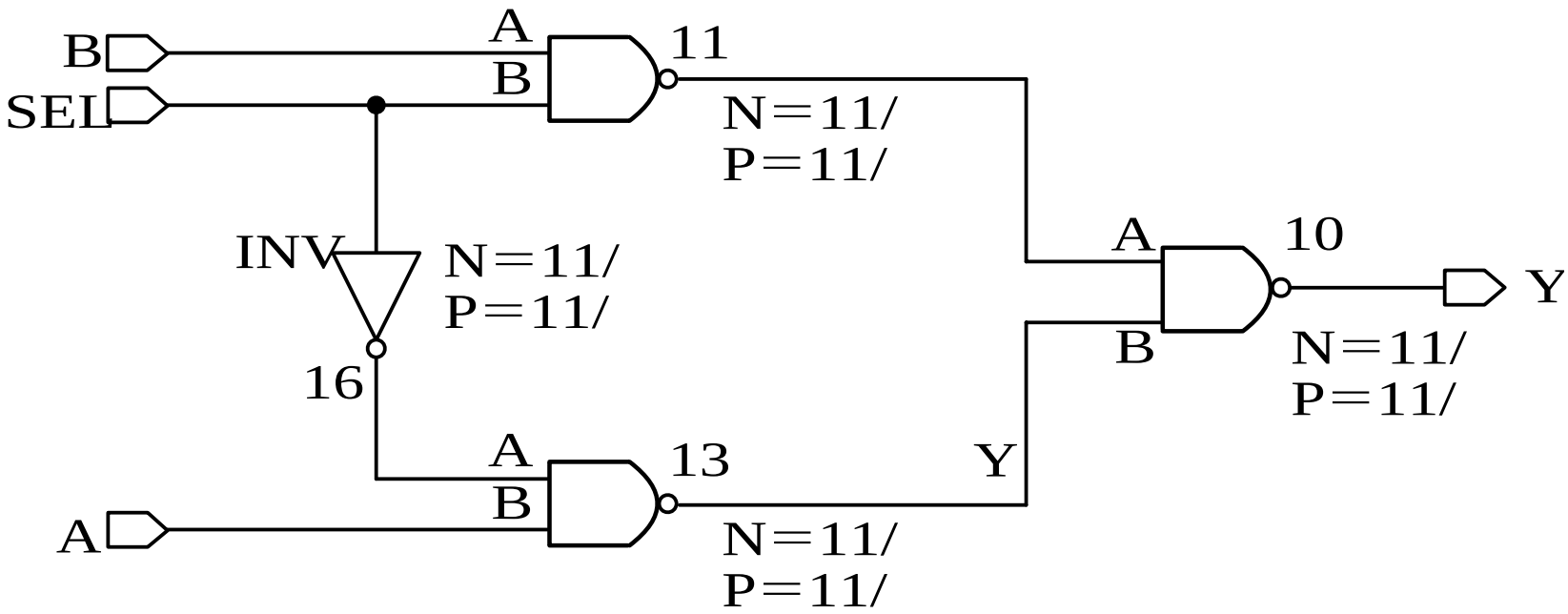


图 5-21 MUX21-1电路图

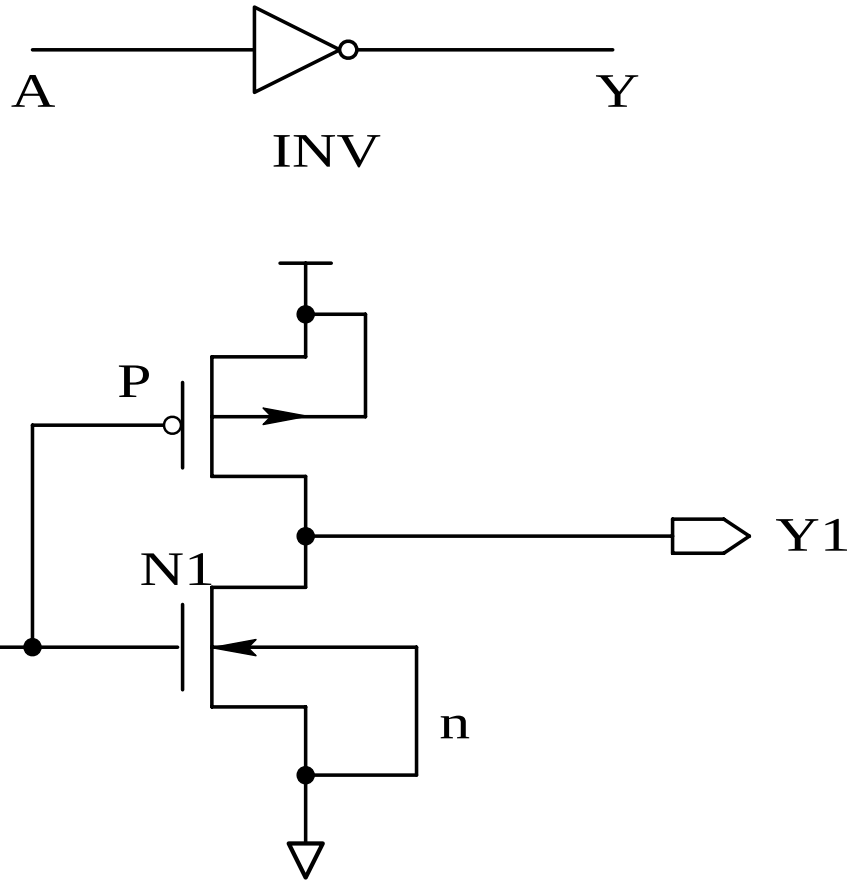


图5-22 CMOS工艺的反相器

(3) NAND2的组成分析。二输入与非门如图5-23所示。NAND2 由两个PMOS管和两个NMOS管组成，因此，NADA2可由PMOS管、NMOS管组合而成，即NAND2的版图可由PMOS、NMOS的版图拼成（也可整体绘制）。

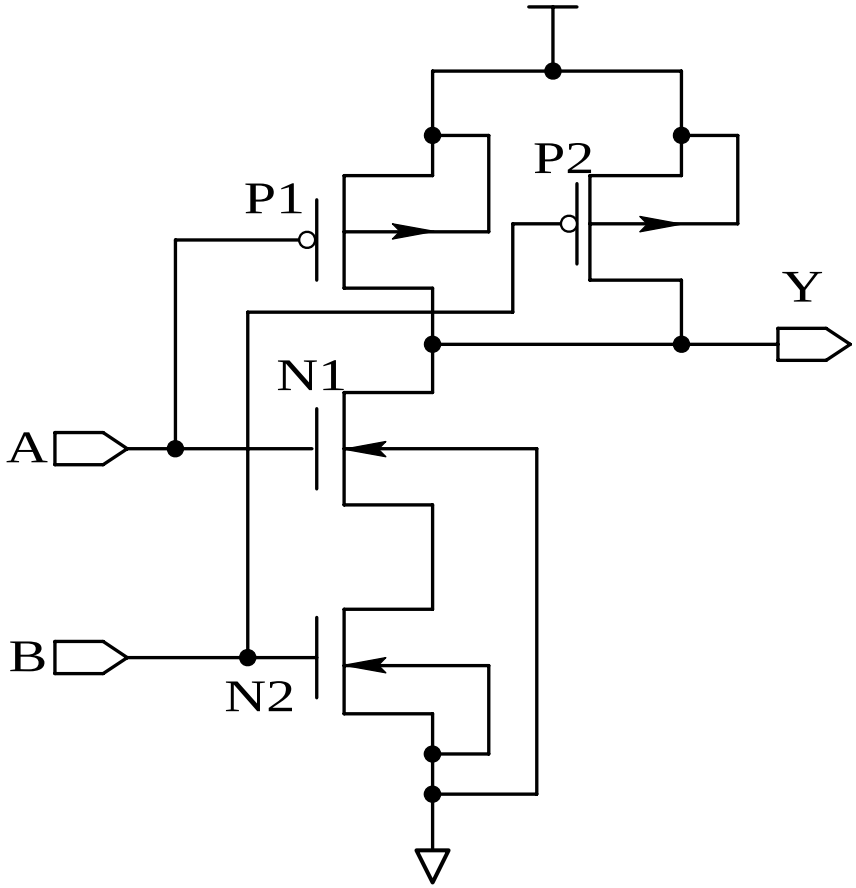


图 5-23 二输入与非门示意图

2. MUX21-1的绘制方法

分步画图：

（1） 绘制PMOS的Layout。

（2） 绘制NMOS的Layout。

（3） 绘制INV的Layout。

（4） 绘制NAND2的Layout。

（5） 绘制MUX21-1的Layout。

3. 绘制Layout的准备

在个人的目录中启动 Layoutplus&。建立目标库(Cell)：PMOS、NMOS、INV、 NAND2、MUX21-1。建立CellView：PMOS 、 NMOS 、 INV 、 NAND2 、 MUX21-1 的 symbol 、schematic描述（其内容可从厂家提供的基本库或其它基本库中拷贝）。建立CellView的Layout描述，以进行Edit编辑。在Library Browser中用鼠标点中Cell（PMOS、NMOS、INV、NAND2、 MUX21-1）的Layout，按下鼠标中键选择Edit 进入编辑环境。 利用LSW工具窗口条在出现的Editing Layout窗口中按设计规则的要求进行版图的编辑。

4. 绘制PMOS的Layout

PMOS管的图形如5-24所示。 绘制PMOS管的步骤如下：

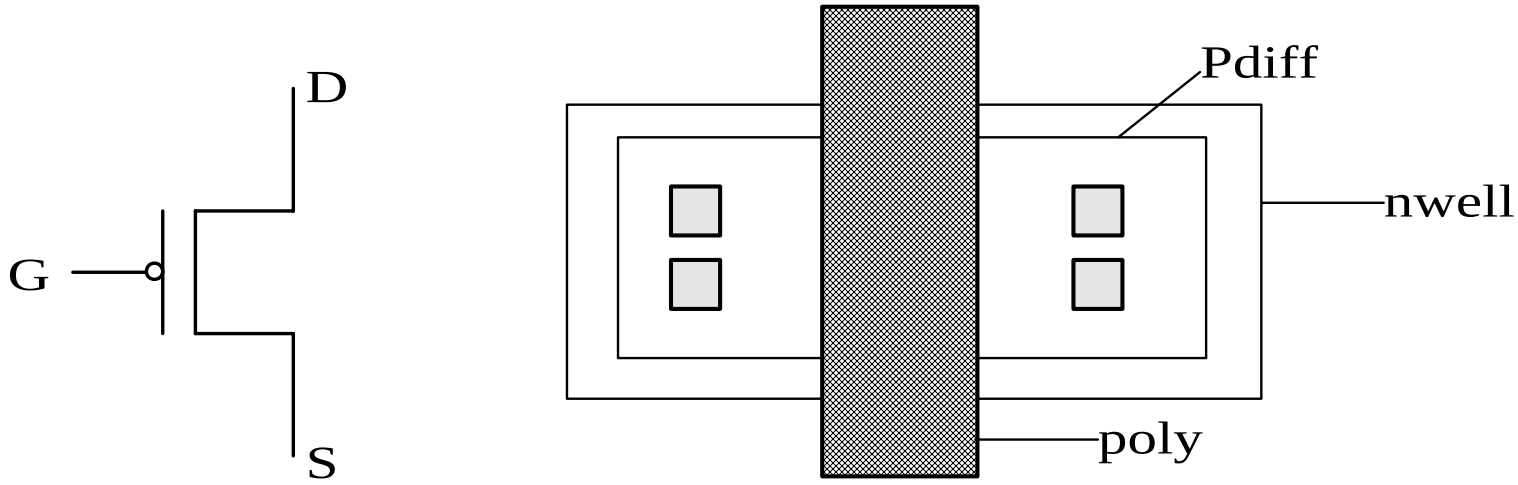


图 5-24 PMOS管的图形

(1) 在LSW中选定nwell，点Rectangle（矩形），在Edit的窗口之中画出nwell区域。

(2) 在LSW中选定pdiff，点Rectangle（矩形），在Edit的窗口之中的nwell中画出pdiff区域。

(3) 在LSW中选定poly， 点Rectangle（矩形），画出poly区域。

(4) 在LSW中选定cont，点Rectangle（矩形），画出一个cont， 再使用copy命令画出其余的cont。

(5) 绘制完毕之后，用Save命令将相关结果保存于目标库中PMOS的Layout。

5. 绘制NMOS的Layout

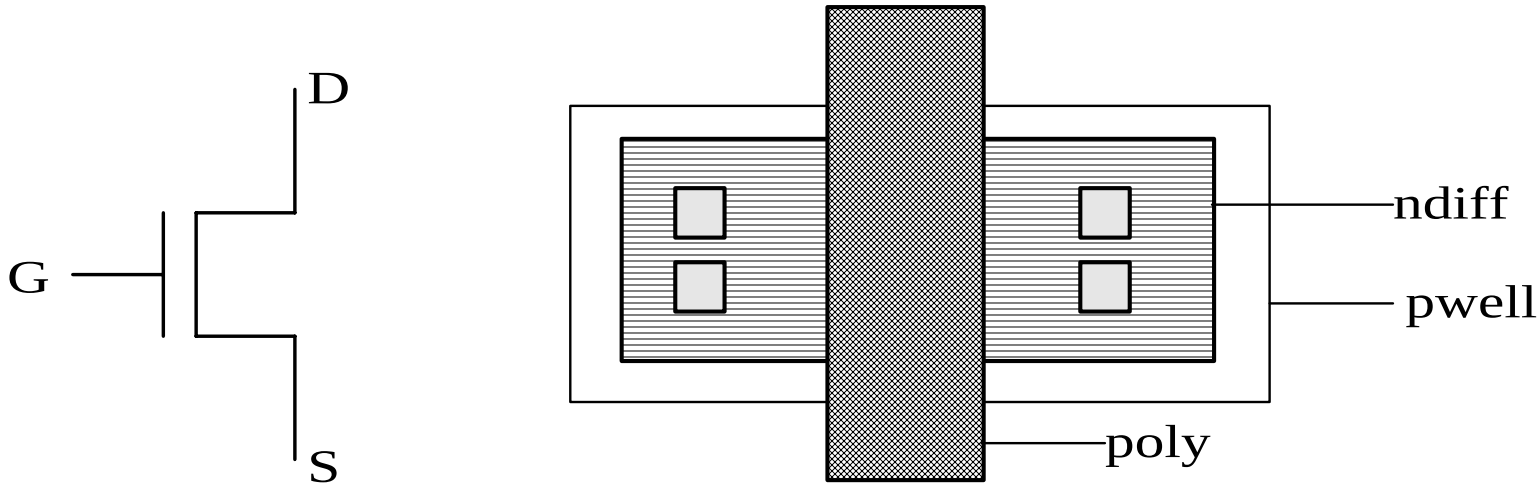


图 5-25 NMOS管的图形

(1) 在LSW中选定pwell，点Rectangle（矩形），在Edit的窗口之中画出pwell区域。

(2) 在LSW中选定ndiff，点Rectangle（矩形），在Edit的窗口之中的pwell中画出pdiff区域。

(3) 在LSW中选定poly， 点Rectangle（矩形），画出poly区域。

(4) 在LSW中选定cont，点Rectangle（矩形），画出一个cont，再使用copy命令画出其余的cont。

(5) 绘制完毕之后， 用Save命令将相关结果保存于目标库中NMOS的Layout。

6. 绘制INV的Layout

绘制INV的Layout的方法有二。

1) 方法一

从目标库中拷贝一个NMOS及一个PMOS的Layout，对照如图5-22所示的原理图进行绘制。具体步骤如下：

(1) 点击copy，选cell-NMOS-layout并拖曳光标至Edit窗口内。

(2) 点击copy，选cell-PMOS-layout并拖曳光标至Edit窗口内。

(3) 在LSW中选定Metal，点Rectangle（矩形），画出VDD-（电源）、GND（地）。

(4) 在LSW中选定Metal，点line（线），连接vdd-pidff、poly-poly（或将NMOS的poly和PMOS的poly改画为一条）、pdiff-ndIff、ndiff-gnd。

(5) 在LSW中选定Pin，点Rectangle（矩形），画出输入、输出的Pin（输入的Pin用Metal与poly相连，输出的Pin用Metal与ndiff-pdiff相连）。

(6) 点Label为Layout加注（vdd，gnd，input，ouput）。拖曳光标至Edit窗口内，双击鼠标左键则可，之后再点击cancel。

(7) 画完之后，用Save命令将相关结果保存于目标库中INV的Layout。绘制好的反相器版图如图5-26所示。

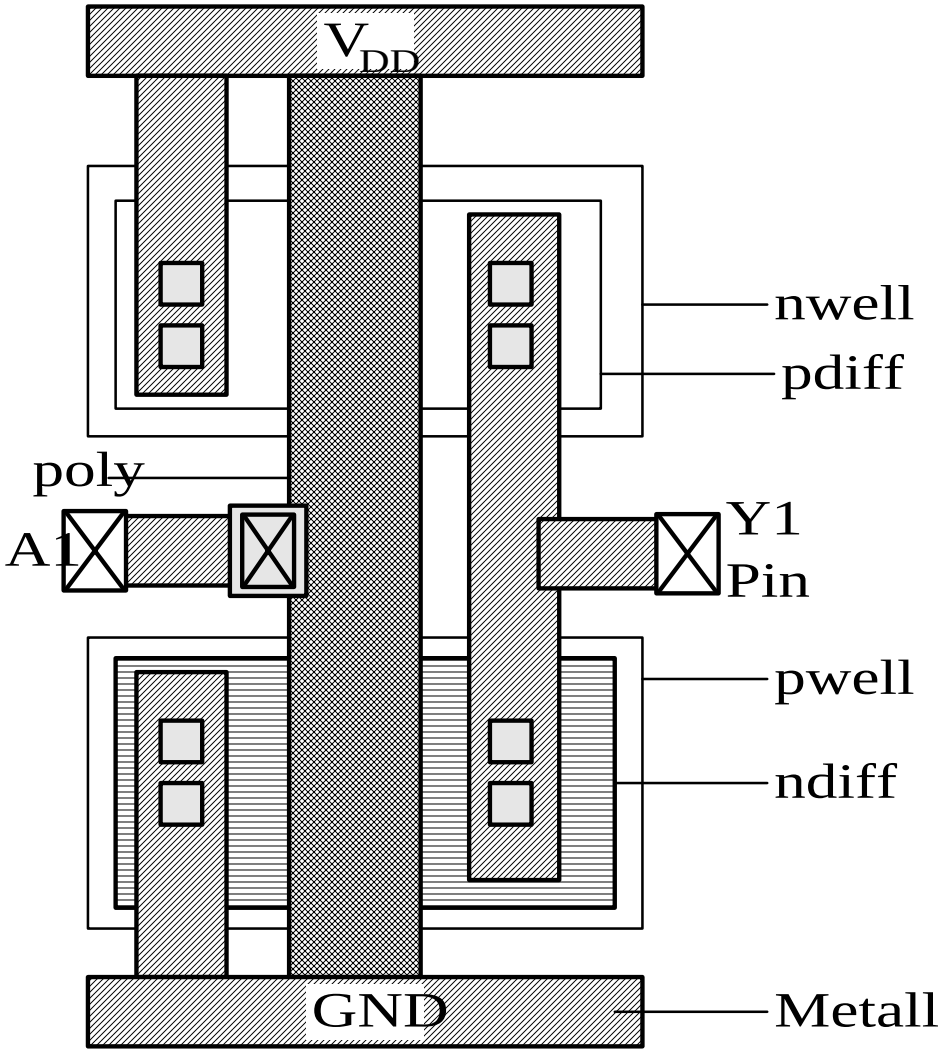


图5-26 绘制好的反相器版图

2) 方法二

从整体上进行绘制，具体步骤如下：

(1) 在LSW中选定nwell，点Rectangle（矩形），在Edit的窗口之中画出nwell区域。

(2) 在LSW中选定pdiff，点Rectangle（矩形），在Edit的窗口之中的nwell中画出pdiff区域。

(3) 在LSW中选定pwell，点Rectangle（矩形），在Edit的窗口之中画出pwell区域。

(4) 在LSW中选定ndiff，点Rectangle（矩形），在Edit的窗口之中的pwell中画出pdiff区域。

(5) 在LSW中选定poly，点Rectangle（矩形）， 画出poly区域。

(6) 在LSW中选定cont，点Rectangle（矩形），画出一个cont， 再使用copy命令画出其余的cont。

(7) 在LSW中选定Metal，点line（线），连接vdd pidff、pdiff ndIff、ndiff gnd。

(8) 在LSW中选定Pin，点Rectangle（矩形），画出输入、输出的Pin（输入的Pin用Metal与poly相连，输出的Pin用Metal与ndiff-pdiff相连）

(9) 点Label为Layout加注（vdd，gnd，input，ouput）。拖曳光标至Edit窗口内，双击鼠标左键则可，之后再点cancel。

(10) 画完之后，用Save命令将相关结果保存于目标库中INV的Layout。 整体绘制的反相器版图如图5-27所示。

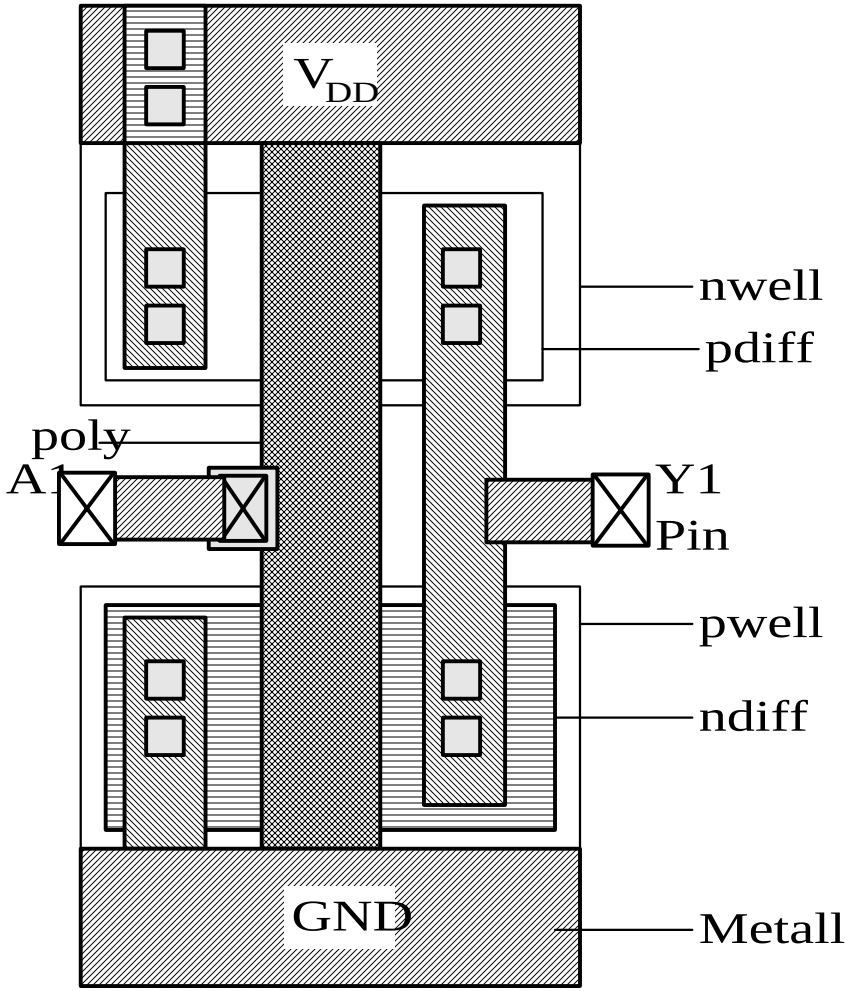


图5-27 整体绘制的反相器版图

7. 绘制NAND2的Layout

1) 方法一

从目标库中拷贝两个NMOS及两个PMOS的Layout，对照NAND2的Schematic进行连线(Metal)，绘制NAND2。绘制NAND2的步骤如下：

(1) 点copy，选cell-NMOS-layout并拖曳光标至Edit窗口内，再copy一次。

(2) 点copy，选cell-PMOS-layout并拖曳光标至Edit窗口内，再copy一次。

(3) 对照NAND2的Schematic，安排MOS管的位置，以便连线。

(4) 在LSW中选定Metal，点Rectangle（矩形），画出V_{DD}（电源）、GND（地）。

(5) 在LSW中选定Metal，点line（线），连接vdd-pidff、poly-poly、pdiff-ndIff、ndiff-gnd（可调出NAND2的Schematic图进行参考）。

(6) 在LSW中选定Pin，点Rectangle（矩形），画出NAND2输入、 输出的Pin。

(7) 点Label为Layout加注（vdd，gnd，input，ouput）。拖曳光标至Edit窗口内，双击鼠标左键则可，之后再点cancel。

(8) 绘制完毕之后，用Save命令将相关结果保存于目标库中的NAND2。 绘制好的NAND2版图如图5-28所示。

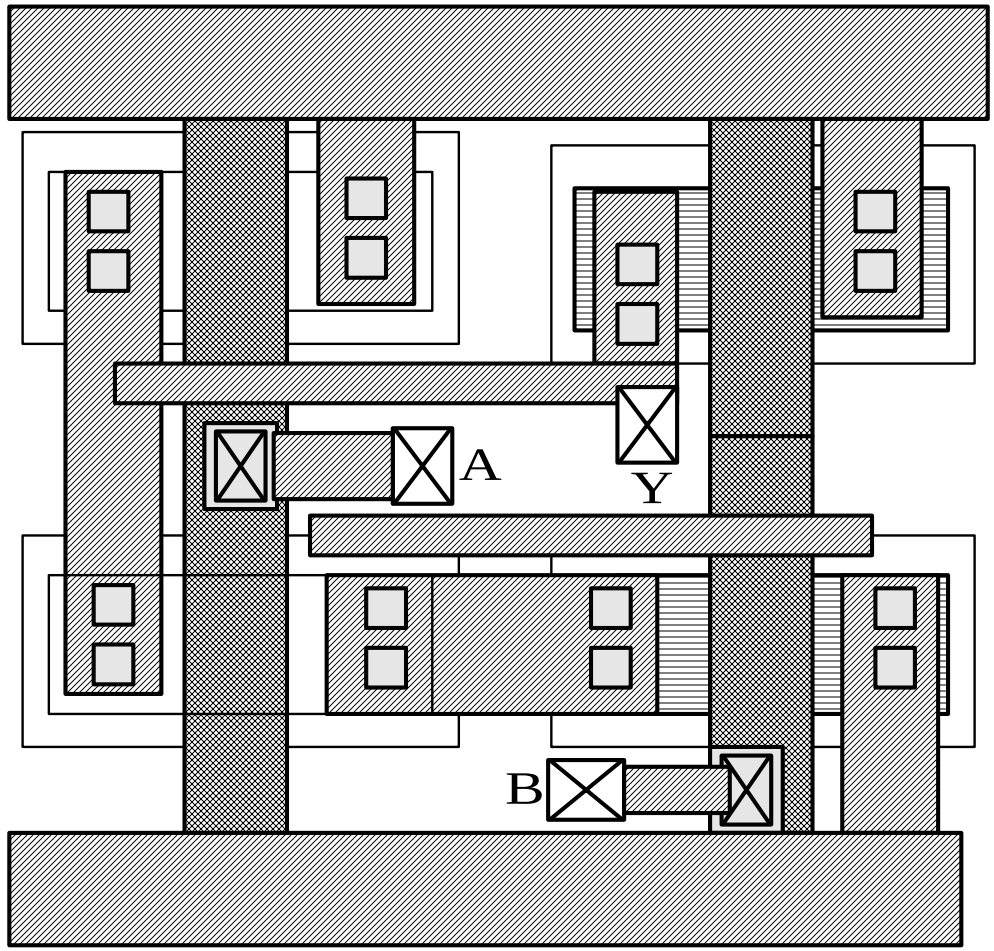


图 5-28 绘制好的NAND2版图

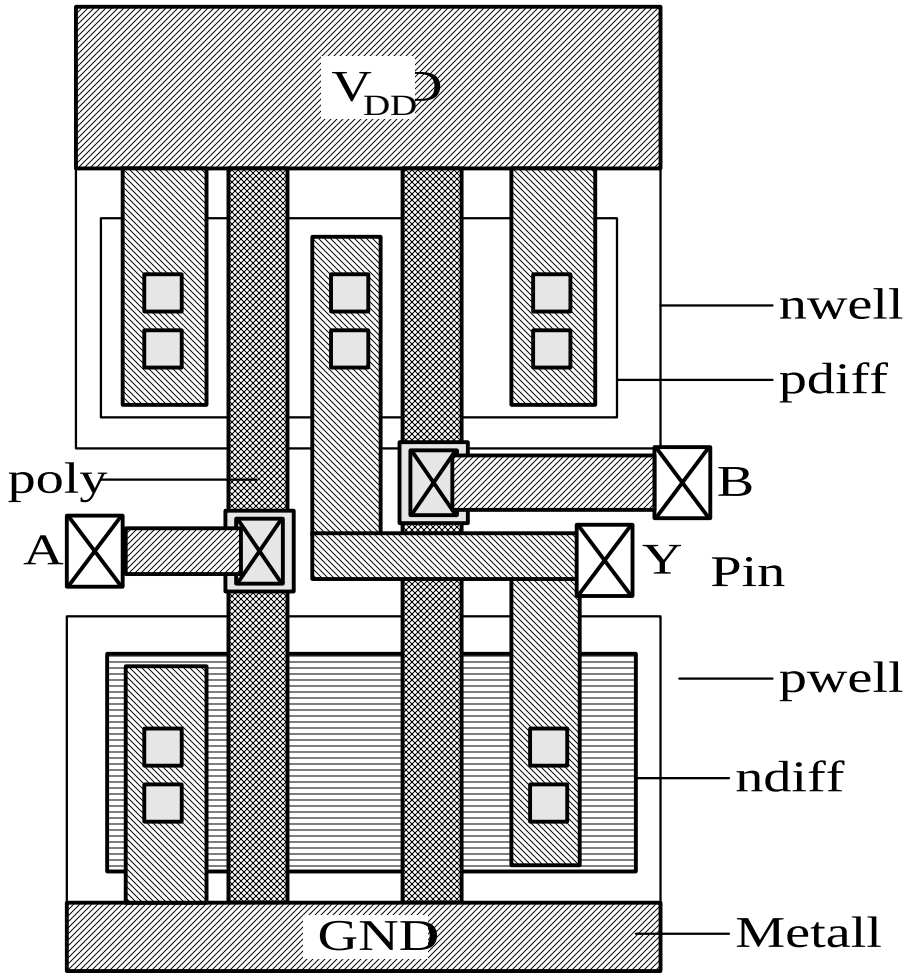


图 5-29 整体方法绘制的NAND2

2) 方法二

采用整体方法绘制的NAND2如图5-29所示。 其步骤如下：

(1) 在LSW中选定nwell，点Rectangle（矩形），在Edit的窗口之中画出nwell区域（两个PMOS公用）。

(2) 在LSW中选定pdiff，点Rectangle（矩形），在Edit的窗口之中的nwell中画出pdiff区域（两个PMOS公用）。

(3) 在LSW中选定pwell，点Rectangle（矩形），在Edit的窗口之中画出pwell区域（两个NMOS公用）。

(4) 在LSW中选定ndiff，点Rectangle（矩形）， 在Edit的窗口之中的pwell中画出pdiff区域（两个NMOS公用）。

(5) 在LSW中选定poly， 点Rectangle（矩形），画出poly区域。

(6) 在LSW中选定cont，点Rectangle（矩形），画出一个cont， 再使用copy命令画出其余的cont。

(7) 在LSW中选定Metal，点line（线），连接vdd-pidff、pdiff-ndIff、 ndiff-gnd。

(8) 在LSW中选定Pin，点Rectangle（矩形），画出输入、输出的Pin（输入的Pin用Metal与poly相连，输出的Pin用Metal与ndiff　pdiff相连）。

点Label为Layout加注（vdd、gnd、input、ouput分次填入）。拖曳光标至Edit窗口内，双击鼠标左键则可，之后再点cancel。

(9) 绘制完毕后， 用Save命令将相关结果保存于目标库中NAND2的Layout。

8. 绘制MUX21-1的Layout

综上所述, 可知较为复杂的库单元的版图可由简单库单元的版图拼成。常用的基本库单元的版图已存放在厂家的基本库及其它基本库之中，可直接调用和拷贝。 MUX21-1可用INV、NAND2的Layout拼成。拼接的步骤如下：

(1) 点copy，选cell-INV-layout并拖曳至Edit窗口内。

(2) 点copy，选cell-NAND2-layout并拖曳至Edit窗口内。

(3) 对照MUX21-1的电路图进行连线（方法参照INV、NAND2的画图步骤）。

(4) 点Label为Layout加注（vdd，gnd，input，ouput）。拖曳光标至Edit窗口内，双击鼠标左键则可，之后再点cancel。

(5) 绘制完毕之后， 用Save命令将相关结果保存于目标库中NAND2的Layout。

5.4.4 Diva流程

1. 设计规则检查（Diva/DRC）

(1) 输入验证模块的Layout View。打开Cell的autoRouted View，选择Floorplan中的Replace CellView，将所有元件的Abstract View替换为 Layout View， 选择Tools下的Layout和Verify中的DRC。

(2) 验证结果的获取和说明。选择Verify中的Marker中的find， 在option中输入zoom to error，选择Verify中的Marker中的explain， 即可获取验证结果的说明。

(3) 对于DRC中的错误可手工修改或重新布局布线。

2. Diva/Extract

(1) 输入验证模块的Layout View。选择Verify中的extract，在switch中输入do-extract(iLPE) (提取寄生电容)，在switch中中输入do-pre(iPRE)(提取寄生电阻)。switch选项可选择多个，用空格隔开。

(2) Diva/Extract的结果是产生了Extracted View文件。 选择Show Run Info，其中option有：log(显示执行信息)，netlist(网表文件)， Layout netlist (显示网表)。

3. 电学规则检查（Diva/ERC）流程

(1) 输入电学规则检查的是extracted View。选择Verify中的extract命令，在Form中分别填入Library name、Cell name、View name(extracted View)。

(2) 电学规则检查的结果。选择Show Run Info(选项同DRC)，运行目录下的netlist为电学规则检查的Hspice网表。

4. 逻辑图与版图对比（Diva/LVS）流程

(1) 输入逻辑图与版图对比的Schematic View及extracted View，要求所有器件的Schematic均为版图设计中使用的Schematic。选择Verify中的LVS，在Form中填入Library name、Cell name、 View name(Schematic and extracted View)。

(2) 逻辑图与版图对比的结果。选择Show Run Info(选项同DRC)，运行目录下的netlist为逻辑图与版图对比的Hspice网表文件。

5.4.5 Dracula流程

1. 设计规则检查(DRC)

(1) 编写Dracule命令文件。 路径：

/user5/Sch/Sch/lic/comfile/drc05.cds(Cadence格式)

/user5/Sch/Sch/lic/comfile/drc05.gds(GDSII格式)

(2) 生成Dracule运行文件。 在Dracule命令文件中加入以下命令：

① 格式1(Cadence格式)

SYSTEM = Cadence

LIBRARY = 设计库名称

INDISK = 输入文件所在路径

PRIMARY = 单元名

OUTLIB = 输出设计库名

OUTDISK = 输出文件名

② 格式2(GDS 格式)

SYSTEM = GDS2

INDISK = 输入文件所在路径及名称

PRIMARY = 单元名

OUTLIB = 输出设计库名

OUTDISK = 输出文件名

分别键入PDRACULA、 /get Dracula命令文件名键入/fin后， 产生jxrun.com程序。

(3) 执行jxrun.com程序。

(4) 打开单元的Layout View。

(5) 打开Tools菜单下的verification命令中的inquery， 进行相应的操作。

2. 电学规则检查(ERC):

(1) 编写Dracula命令文件。 路径:

/user5/Sch/Sch/lic/comfile/erc05.cds(Cadence格式)

(2) 生成Dracula运行文件。在Dracula命令文件中加入以下命令:

SYSTEM = Cadence

LIBRARY = 设计库名称

INDISK = 输入文件所在路径

PRIMARY = 单元名

OUTLIB = 输出设计库名

OUTDISK = 输出文件名

分别键入PDRACULA、 /get Dracula命令文件名， 键入/fin后，产生jxrun.com程序。

(3) 执行jxrun.com程序。

(4) 打开单元的Layout View。

(5) 打开Tools菜单下的verification命令中的inquery，进行相应的操作。

3. 版图与逻辑图对比(LVS)

(1) 编写Dracula命令文件。 路径：

/user5/Sch/Sch/lic/comfile/lvs05.cds(Cadence格式)

(2) 生成Dracula运行文件。

① 准备电路网表。 用CDL Out将Schematic生成Spice网表文件。

② 将Spice网表文件编译成Dracula可接受的格式。

LOGLVS

: cir spice netlist

: con 器件名

: x

结果是产生LVSLOGIC.DAT文件。

③ 在Dracula命令文件中加入以下命令：

SYSTEM = Cadence

LIBRARY = 设计库名称

INDISK = 输入文件所在路径

PRIMARY = 单元名

SCHEMATIC = LVSLOGIC.DAT

OUTLIB = 输出设计库名

OUTDISK = 输出文件名

LVSCHK=［option］

分别键入PDRACULA、 /get Dracula命令文件名， 键入/fin后，产生jxrun.com程序。

(3) 执行jxrun.com程序。

(4) 打开单元的Layout View。

(5) 打开Tools菜单下的Verification命令中的inquery， 进行相应的操作。

4. 版图参数提取(LPE)

(1) 编写Dracula命令文件。 路径：

/user5/Sch/Sch/lic/comfile/lpe05.cds(Cadence格式)

(2) 生成Dracula运行文件。 在Dracula命令文件中加入以下命令：

SYSTEM = Cadence

LIBRARY = 设计库名称

INDISK = 输入文件所在路径

PRIMARY = 单元名

OUTLIB = 输出设计库名

OUTDISK = 输出文件名

分别键入PDRACULA、 /get Dracula命令文件名， 键入/fin后，产生jxrun.com程序。

(3) 执行jxrun.com程序。

5. 寄生电阻提取

(1) 编写Dracule命令文件。 路径：

/user5/Sch/Sch/lic/comfile/pre05.cds(Cadence格式)

(2) 生成Dracule运行文件。 在Dracule命令文件中加入以下命令：

SYSTEM = Cadence

LIBRARY = 设计库名称

INDISK = 输入文件所在路径

SCHEMATIC = LVSLOGIC.DAT

PRIMARY = 单元名

OUTLIB = 输出设计库名

OUTDISK = 输出文件名

分别键入PDRACULA/get Dracula命令文件名/fin后， 产生jxrun.com程序。

(3) 执行jxrun.com程序。

5.4.6 参数提取反标

版图反标(Back annotate)与后仿真是版图验证的重要工作，是版图设计质量的保证。

(1) 若采用Diva进行LVS后，可采用如下方法进行版图反标及Hspice仿真。

① 打开单元的Schematic View。

② 选择LVS Form中的Backannoted选项。

③ 选择Form中的Add parastic，将参数反标到Schematic View中。

④ 选择Schematic View菜单Tools中的Analog artist。

⑤ 选择Analog artist菜单中的Simulation， 即进入Hspice仿真环境。

(2) 若采用Dracula进行PRE后，生成SPICE.DAT，在该文件中加入Hspice仿真激励， 即可进行Hspice仿真。

5.4.7 门级时序分析

门级(Gate Level)分析的方法和步骤如下:

1. 准备过程

1) Setup env

在.cshrc里加上：

Set Path = ($Path $PEARL/bin)

Setenv PEARL /Usr / Valid / tools / Pearl (或/cds/cds9502/tools/pearl)

Copy $ PEARL/etc/pearl-init

Home下为： ~/.pearl

2) 准备技术文件

TECHNOLOGY std　　//定义此段tec

hfile的名字

POWER_NODE_NAME VDD 5　　//定义电源

GND_NODE_NAME VSS　　//定义地线

LOGIC_THRESHOLD 2.5　　//定义开启电压

RISE_TIME 20_80 .5NS

WIRE_CAPACITANCE_ESTIMATE .05P 0　//定义线电容

SPICE_PROGRAM SPICE2 pearl_spice2 vcmos.spice-models

SYNOPSYS_UNITS 1N 1P 1K　//依次定义时间、电容、电阻的单位

END_TECHNOLOGY

3) 准备Timing Model for Standard Cell

系统准备部分， 主要完成Model标准单元参数的定义。

```
# units are pF, ns, kohms  //系统准备部分(定义Model标准单元的参数)
UNITS C=P T=N R=K
MODEL DREG                    //定义DREG此标准单元的参数

OUTPUT Q DRVR=CMOS
INPUT D CAP=.08               //电容
INPUT CLK CAP=.08             //电容
DELAY R CLK -> Q 1.0 1.0 1.0 1.0          //R.触发方式
SETHLD D _> CLK ^ .5 .5 0 0               //Setup hold的延时
END_MODEL
```

```
MODEL INVERTER    //以下标准单元定义均同上述OUTPUT
OUT DRVR=CMOS
INPUT IN CAP=.08
DELAY I IN -> OUT 1.0 1.0 1.0 1.0
END-MODEL

MODEL BUFFER
OUTPUT OUT DRVR=CMOS
INPUT IN CAP=.08
DELAY N IN -> OUT 1.0 1.0 1.0 1.0
END_MODEL
```

```
MODEL MUX2
OUTPUT OUT DRVR=CMOS
INPUT I0 CAP=.08
INPUT I1 CAP=.08
INPUT S CAP=.08
DELAY E S -> OUT 1.0 1.0 1.0 1.0
DELAY N I0 -> OUT 1.0 1.0 1.0 1.0
DELAY N I1 -> OUT 1.0 1.0 1.0 1.0
END_MODEL
```

```
MODEL ADDER
OUTPUT SUM DRVR=CMOS
OUTPUT COUT DRVR=CMOS
INPUT A CAP=.08
INPUT B CAP=.08
INPUT CIN CAP=.08
DELAY E A -> SUM 1.0 1.0 1.0 1.0
DELAY E B -> SUM 1.0 1.0 1.0 1.0
DELAY E CIN -> SUM 1.0 1.0 1.0 1.0
DELAY E A -> COUT 1.0 1.0 1.0 1.0
DELAY E B -> COUT 1.0 1.0 1.0 1.0
DELAY E CIN -> COUT 1.0 1.0 1.0 1.0
END_MODEL
```

```
MODEL LATCH

OUTPUT Q DRVR=CMOS

INPUT D CAP=.08

INPUT EN CAP=.08

DELAY NL D -> Q 1.0 1.0 1.0 1.0

DELAY GL EN -> Q 1.0 1.0 1.0 1.0

SETHLD D -> EN v .5 .5 0 0

END_MODEL
```

```
MODEL NAND2
OUTPUT OUT DRVR=CMOS
INPUT IN1 CAP=.08
INPUT IN2 CAP=.08
DELAY I IN1 -> OUT 1.0 1.0 1.0 1.0
DELAY I IN2 -> OUT 1.0 1.0 1.0 1.0
END_MODEL

DEFDRIVER CMOS RH=20.0 RL=10.0
```

2. 分析过程

1) Invoke engine　　//启动pearl时序分析工具

Pearl

2) 将原始文件读进来　　//用include命令

(1) 原始文件说明。原始文件包括前面定义的技术文件及Timing Models文件，还有对模块的Verilog文件等。文件内容如下：

Diff-engine-gate.cmd（包含上述原始文件的文件名）

```
# exerpts from the default .pearl init file
Alias dn DescribeNode
Alias sp ShowPossibility                    //定义命令
ReadTechnology std-cell.tech                //读技术文件
ReadTimingModels diff-engine-cells.m        //读Timi
ng Models文件
ReadVerilog diff    engine.v                //读Verilog文件（对
器件的Verilog描述）
TopLevelCell DIFF    -ENGINE                //定义顶级文件
EstimateStrayCapacitances                   //加线电容
```

(2) 操作命令

cmd> include diff-engine-gate.cmd

3) 分析

(1) 常用命令说明。

```
# sample shorthands for .pearl initialization file
alias sp ShowPossibility          //显示某一path
alias dn DescribeNode             //显示某一节点
alias dd DescribeDevice           //显示某一器件
alias fpf FindPathsFrom           //查找Longest Path
alias tv TimingVerify             //验证 setup hold
alias sd ShowNodeDelays  //显示某一节点在CLK^变化时，新值什么时候会到
aliad sdp ShowDelayPath  //进一步看sd的结 果， 用sdp
alias soe ShowNodeOutputEqns
```

alias seb ShowEqnsBetweenNodes //显示两节点间delay(只能是某一器件的in/out)

alias sie ShowNodeInputEqns

alias sh ShowNodeHierarchyNames

alias spice SpiceDelayPath

alias log Logfile

alias cd SetDirectory //进入目录

alias q quit //退出分析环境

(2) 分析过程。举例：一宏单元模块DIFF -ENGINE的Verilog描述：

```
module DIFF-ENGINE (LDA, LDB, -LDC, CLK, BUS, SUM);

input LDA;

input LDB;

input-LDC;

output ［7：0］ SUM;

input ［7：0］ BUS;

input CLK;

wire ［7：0］ COUT;

DIFF_ENGINE-BITDEB_0(LDA, LDB,_LDC, CLK, BUS［0］, VSS,
COUT［0］, SUM［0］, SUM［7］);
```

DIFF_ENGINE_BIT DEB_1(LDA, LDB, _LDC, CLK, BUS［1］, COUT［0］, COUT［1］, SUM［1］, SUM［7］);

DIFF_ENGINE_BIT DEB_2(LDA, LDB,_LDC,CLK, BUS［2］, COUT［1］, COUT［2］, SUM［2］, SUM［7］);

DIFF_ENGINE-BIT DEB-3(LDA, LDB,_LDC, CLK, BUS［3］, COUT［2］, COUT［3］, SUM［3］, SUM［7］);

DIFF_ENGINE_BIT DEB_4(LDA, LDB,_LDC, CLK, BUS［4］, COUT［3］, COUT［4］, SUM［4］, SUM［7］);

```
DIFF_ENGINE_BIT DEB_5(LDA, LDB,_LDC, CLK, BUS［5］,
COUT［4］, COUT［5］, SUM［5］, SUM［7］);
DIFF _ ENGINE _ BIT DEB _ 6(LDA, LDB, _ LDC, CLK, BUS［6］,
COUT［5］, COUT［6］, SUM［6］, SUM［7］);
DIFF _ ENGINE _ BIT DEB _ 7(LDA, LDB, _ LDC, CLK, BUS［7］,
COUT［6］, COUT［7］, SUM［7］, SUM［7］);
endmodule

module DIFF_ENGINE_BIT(LDA, LDB,_LDC, CLK, BUS, CIN,
COUT, SUM, SUM_SIGN);
output COUT;
input LDA;
```

```
output SUM;
input CIN;
input SUM_SIGN;
input LDB;
input BUS;
Input_LDC;
input CLK;

DREG DREGC(C, NEW_C, CLK);
DREG DREGB(B, BUS, LDB);
DREG DREGA(A, BUS, LDA);
ADDER ADDER(NEW_SUM, COUT, C, AB, CIN);
BUFFER BUF(SUM, C);
MUX2 MUX1(AB, A, B, SUM_SIGN);
MUX2 MUX2(NEW_C, BUS, NEW_SUM,_LDC);
endmodule
```

DIFF_ENGINE的分析步骤及输入的命令如下：

~/Pearl>pearl

Cmd>include diff　pate.Cmd

Cmd>fpf.CLK^

Cmd>

用fef clk^可找出最长的path。先找出最长的path，再找到其最大的延时在什么地方。用seb查看为什么delay大：seb node［edge］ node ［edge］。用dn查看负载为什么大：dn node。用sd查看在clk^有效时节点的delay： sd node。用sdp查看上述节点的delay(min, max)path， 格式如下：

sdp node edge man\ min

(3)设定时钟周期， 命令格式如下：

Clock Clk 0 25

CycleTime 50

(4) 分析用命令： tv。

用sp可查看有无Setup hold violation， 命令格式如下：

Violation : Means violate setup hold constraint

Slack : Means have slack tim

算法是：

① CycleTime Setup delay；

② delay hold。

分析最小可达到的时钟频率， 其命令格式如下：

Find Min Cycle Time

输出SDF格式文件，其命令格式如下：

write SDF Path Constraint

5.4.8 晶体管级时序分析

晶体管级(Transistor Level)的分析步骤如下。

1. 准备文件

1) 门级以下网表文件。可以是verilog文件， 也可以是Spice网表文件。 举例：

```
module DREG (Q,  D,  CLOCK);
    input D;
    output Q;
    input CLOCK;
```

```
  INVERTER INV3(Q,  SB);
  INVERTER INV2(SB,  S);
  INVERTER CLK_BUF2(C,_C);
  INVERTER INV1(MB,  M);
  INVERTER CLK_BUF1(_C,  CLOCK);
  TR_INV TR_INV4(S,  SB, _C,  C);
  TR_INV TR_INV3(S,  M,  C, _C);
  TR_INV TR_INV2(M,  MB,  C, _C);
  TR_INV TR_INV1(M,  D, _C,  C);
endmodule

module INVERTER (OUT,  IN);
   output OUT;
   input IN;
```

```
    pmos #(3200, 200) PU(OUT, VDD, IN);
    nmos #(1600, 200) PD(VSS, OUT, IN);
endmodule

module TR_INV (OUT, IN, S, _S);
  input _S;
  output OUT;
  input IN;
  input S;
```

```
        pmos #(400, 200) SPU(OUT, NODE4, _S);
        pmos #(400, 200) IPU(NODE4, VDD, IN);
        nmos #(400, 200) SPD(NODE3, OUT, S);
        nmos #(400, 200) IPD(VSS, NODE3, IN);
endmodule

module ADDER (SUM, COUT, A, B, CIN);
        input CIN;
        output SUM;
        input A;
        output COUT;
        input B;
```

```
INVERTER INV1(SUM,  _S);
INVERTER INV2(COUT,  _C);
nmos #(400,  200) PD42(NODE6,  NODE4,  B);
pmos #(400,  200) PU32(NODE5,  VDD,  A);
nmos #(400,  200) PD32(VSS,  NODE7,  B);
nmos #(400,  200) PD34(VSS,  NODE7,  CIN);
nmos #(400,  200) PD41(NODE4, _S,  CIN);
nmos #(400,  200) PD13(VSS,  NODE8,  A);
pmos #(400,  200) PU41(NODE11,  VDD,  A);
pmos #(400,  200) PU34(_S,  NODE5,  _C);
pmos #(400,  200) PU13(_C,  NODE9,  CIN);
nmos #(400,  200) PD22(VSS,  NODE12,  A);
nmos #(400,  200) PD33(VSS,  NODE7,  A);
```

```
    pmos #(400, 200) PU33(NODE5, VDD, CIN);
    nmos #(400, 200) PD12(VSS, NODE8, B);
    pmos #(400, 200) PU22(_C, NODE10, B);
    pmos #(400, 200) PU12(NODE9, VDD, A);
    pmos #(400, 200) PU42(NODE3, NODE11, B);
    nmos #(400, 200) PD43(VSS, NODE6, A);
    pmos #(400, 200) PU31(NODE5, VDD, B);
    nmos #(400, 200) PD31(NODE7,_S,_C);
    pmos #(400, 200) PU43(_S, NODE3, CIN);
    nmos #(400, 200) PD11(NODE8,_C, CIN);
    pmos #(400, 200) PU21(NODE10, VDD, A);
    pmos #(400, 200) PU11(NODE9, VDD, B);
    nmos #(400, 200) PD21(NODE12,_C, B);
endmodule
```

```
module BUFFER (OUT,  IN);
   output OUT;
   input IN;

   INVERTER INV1(OUT,  NODE3);
   INVERTER INV2(NODE3,  IN);
endmodule

module MUX2 (OUT,  I0,  I1,  S);
   output OUT;
   input I0;
   input I1;
   input S;
```

```
    INVERTER SBUF1(SEL0,  S);

    INVERTER SBUF2(SEL1,  SEL0);

    TR_INV TR_INV1(O,  I0,  SEL0,  SEL1);

    TR_INV TR_INV0(O,  I1,  SEL1,  SEL0);

    INVERTER OUT_BUF(OUT,  O);

endmodule
```

2) 准备技术文件

cmd> lnclude VCMOS.tech

```
# 2 micron cmos technology
TECHNOLOGY VCMOS
POWER_NODE_NAME VDD 5.0
GND_NODE_NAME VSS
LOGIC_THRESHOLD 2.5
RISE_TIME 20-80 .5NS
FET_MINIMUM_NORMAL_SIZE 1
FET_OVERHANG_ESTIMATE 5.4U
SPICE_PROGRAM SPICE2 pearl-spice2 vcmos.spice-models
# only estimate diffusion capacitance
WIRE_CAPACITANCE_ESTIMATE 0 0
```

```
LAYER GATE 8.1E-4 0
LAYER NDIFF 1.13E-4 2.84E-10
LAYER PDIFF 2.21E-4 3.21E-10

FET TN NFET
GATE_LAYER GATE
DIFF_LAYER NDIFF
RSIM_STRENGTH_THRESHOLDS 2.0 8.0
# FindTimingResistances TN 4/2 .1P 20 .2 .5 1 2 5 7 10
TIMING_RESISTANCE RISE 0.0 31.0K
TIMING_RESISTANCE RISE 0.08 31.0K
TIMING_RESISTANCE RISE 0.12 34.9K
TIMING_RESISTANCE RISE 0.23 41.2K
```

TIMING_RESISTANCE RISE 0.46 53.4K

TIMING_RESISTANCE RISE 1.12 87.2K

TIMING_RESISTANCE RISE 1.57 108K

TIMING_RESISTANCE RISE 2.24 138K

TIMING_RESISTANCE FALL 0.0 11.8K

TIMING_RESISTANCE FALL 0.20 12.7K

TIMING_RESISTANCE FALL 0.31 15.5K

TIMING_RESISTANCE FALL 0.61 17.9K

TIMING_RESISTANCE FALL 1.20 20.9K

TIMING_RESISTANCE FALL 2.95 23.4K

TIMING_RESISTANCE FALL 4.13 22.6K

TIMING_RESISTANCE FALL 5.88 19.6K

END_FET

FET TP PFET

GATE_LAYER GATE

DIFF_LAYER PDIFF

RSIM_STRENGTH_THRESHOLDS 2.0 8.0

FindTimingResistances TP 8/2 .1P 20 .2 .5 1 2 5 7 10

TIMING_RESISTANCE RISE 0.0 30.4K

TIMING_RESISTANCE RISE 0.16 32.6K

TIMING_RESISTANCE RISE 0.24 38.5K

TIMING_RESISTANCE RISE 0.47 43.9K

TIMING_RESISTANCE RISE 0.93 50.6K

TIMING_RESISTANCE RISE 2.29 58.0K

TIMING_RESISTANCE RISE 3.20 57.8K

TIMING_RESISTANCE RISE 4.56 53.6K

TIMING_RESISTANCE FALL 0.0 73.6K

TIMING_RESISTANCE FALL 0.06 71.1K

TIMING_RESISTANCE FALL 0.10 78.3K

TIMING_RESISTANCE FALL 0.19 89.4K

TIMING_RESISTANCE FALL 0.38 113K

TIMING_RESISTANCE FALL 0.95 177K

TIMING_RESISTANCE FALL 1.32 218K

TIMING_RESISTANCE FALL 1.88 277K

END_FET

```
# Lsim netlist terminal types
# These are not used by the diff-engine example
LSIM_TERMINAL IN INPUT
LSIM_TERMINAL OUT OUTPUT
LSIM_TERMINAL INOUT BIDIRECT
LSIM_TERMINAL VDD POWER
LSIM_TERMINAL GND GROUND

SYNOPSYS_UNITS 1N 1P 1K

END_TECHNOLOGY
```

3) 准备Cell Declaration文件

在Cell Declaration文件中， 加上晶体管级时序分析所必需的一些声明。

2. 分析过程

晶体管级的分析过程同门级时序的分析是一样的， 这里不再赘述。

3. 分析

(1) 所有在门级能使用的分析方法， 都适用于晶体管时序分析。

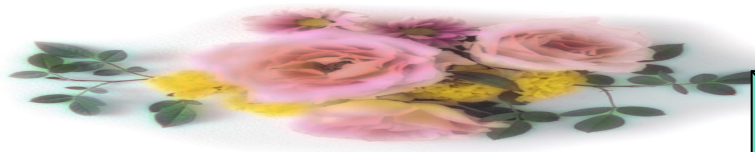
(2) 可以对某一通路作Spice分析。

① 准备工作。在Technology文件中定义所使用的Spice仿真器的名字及路径：

SPICE_OROGRAM HSPICE Pearl　hspice

② 启动以下命令：

Spice Delay Path



复习思考题

1. 对给出的pwell或nwell单元的版图， 写出最常见的设计规则。

2. 知道版图的正向和逆向设计流程。

3. 理解布局布线的各种方法。

4. 熟悉DataPath的操作流程。

5. 理解读CMOS电路图的要领。

6. 对常见的版图应能绘出逻辑电路图。

7. 能读懂简单的DRC程序。