3.2 Layoutdaten: Layer und Polygone

Wie in Abb. 3.1 dargestellt, sind die Layoutdaten das Ergebnis des Layoutentwurfs. Diese Daten werden aber nicht nur dazu genutzt, ein Entwurfsergebnis abzuspeichern und dieses anschließend für die Fertigung weiterzuverarbeiten; vielmehr arbeitet ein Layouter während des gesamten Entwurfsprozesses kontinuierlich mit den Layoutdaten. Daher werden wir im Folgenden die Struktur dieser Daten und die wichtigsten grafischen Operationen, die der Layouter auf diese Daten anwenden kann, genauer betrachten.

3.2.1 Struktur der Layoutdaten

In Kap. 1 (Abschn. 1.3) haben wir bereits einige wichtige Aspekte der Layoutdaten angesprochen. Dabei haben wir gesehen, dass die IC-Layoutdaten ausschließlich aus Grafikdaten bestehen. Diese Grafikdaten enthalten sämtliche Informationen für die Erstellung der Belichtungsmasken. Sie sind ausschließlich zweidimensional. PCB-Layoutdaten bestehen ebenfalls aus (zweidimensionalen) Grafikdaten, die noch ergänzt werden durch Daten mit den Positionen und Durchmessern für das Bohren der Via- und Montagelöcher und mit den Koordinaten zur Positionierung der Bauteile.

Prinzipiell kann eine zweidimensionale grafische Darstellung als eine Bitmap aus Pixeln oder als Vektorgrafik gehandhabt werden. In einer zweidimensionalen Vektorgrafik werden die Bilder aus grafischen Primitiven (Linienstücken) zusammengesetzt.

Layoutdaten werden ausschließlich in einer vektoriellen Datenstruktur gespeichert und verarbeitet. Dies hat mehrere Gründe: (i) Vektorgrafiken eignen sich ideal für die Darstellung von Layouts; (ii) sie benötigen viel weniger Speicherplatz als Bitmaps; (iii) sie lassen sich aufgrund der darin enthaltenen Information wesentlich schneller und einfacher verarbeiten; und (iv) sie können ohne kritischen Genauigkeitsverlust umgestaltet werden. Der einzige Nachteil der vektoriellen Datenstruktur besteht darin, dass sie für eine Darstellung auf Bildschirmen in eine Bitmap umzurechnen ist. Hierfür stehen in modernen Entwurfsumgebungen aber sehr effiziente Algorithmen und eine leistungsfähige Hardware zur Verfügung, so dass dies heute kein Problem mehr darstellt.

Layer Ein einzelnes Grafikelement der Layoutdaten, z. B. die laterale Struktur eines Dotiergebietes oder eines Leiterbahnsegments, bezeichnen wir als *Shape*. Jedes Shape gehört eindeutig einem *Layer* an. Die Layer-Zugehörigkeit ist eine elementare Eigenschaft jedes Shapes. Sie ermöglicht die Zuordnung zu Belichtungsmasken. Darüber hinaus bildet sie die Voraussetzung für die Anwendung grafischer Verknüpfungsoperationen, wie in Abschn. 3.2.3 gezeigt wird.

Ein wichtiger Hinweis muss an dieser Stelle gegeben werden. Wenn wir bei Layoutdaten von „Layern“ sprechen, dann ist damit zunächst nur ein Attribut der Datenstruktur gemeint. Zwar wird es oft so sein, dass ein Layer eine Entsprechung im

Jens Lienig, Jürgen Scheible
Grundlagen des Layoutentwurfs elektronischer Schaltungen
ISBN 978-3-031-15767-7, Springer Vieweg 2023

Herstellungsprozess hat, wie z. B. der Layer „Nwell“ für eine Dotierschicht oder der Layer „Metal1“ für eine Metallisierungsschicht stehen. Dies ist aber nicht für jeden Layer der Fall. Layoutdaten enthalten auch Layer, die keine Entsprechung auf einem Wafer haben, wie wir noch sehen werden (Abschn. 3.3.4). Das Umgekehrte kommt ebenfalls vor. Beispielsweise wird die Gate-Oxidschicht nicht als Layer in den Layoutdaten abgebildet.

Um diesen Unterschied bei der Verwendung des Begriffs „Layer“ kenntlich machen zu können, werden wir die Layer in den Layoutdaten als *gezeichnete Layer* bezeichnen und diejenigen unter ihnen, die im Herstellungsprozess auftreten, als *physikalische Layer*. Wir werden diese erweiterte Terminologie aber nur verwenden, wenn im Falle des Weglassens die Gefahr von Missverständnissen besteht. Ansonsten sprechen wir der Einfachheit halber nur von „Layern“. In diesem Kapitel sind fast immer „gezeichnete Layer“ gemeint.

Shapes Die Shapes der Layoutdaten bestehen ausschließlich aus Polygonen. Ein Polygon ist ein zweidimensionales, zusammenhängendes Grafikelement, das von geraden Kantenstücken begrenzt wird. Dies ermöglicht eine sehr effiziente Abbildung in der vektoriellen Datenstruktur. Polygone lassen sich einfach als eine Liste aufeinander folgender Eckkoordinaten abspeichern. Der dadurch entstehende geschlossene Linienzug bestimmt das Polygon, wobei definiert sein muss, ob sich das Polygon links oder rechts des Linienzugs befindet. Das kann abhängig vom Entwurfswerkzeug variieren. In manchen Entwurfswerkzeugen, die wir fortan auch als *Tools* bezeichnen, wird die erste Koordinate der Liste nochmals an deren Ende angehängt, um das Listenende zu markieren.

Abb. 3.6a zeigt ein Beispiel für ein allgemeines Polygon mit sieben Ecken. Seine Koordinaten werden mit C_i bezeichnet und bestehen aus zwei Zahlenwerten (x_i, y_i). Die Definition eines *Arbeitsrasters* (oft als *Manufacturing grid*, *Working grid* oder kurz als *Grid* bezeichnet) ermöglicht die Verwendung von Integer-Werten, also ganzen Zahlen. Das spart Speicherplatz und die Genauigkeit der Darstellung ist klar definiert.

Donuts Polygone mit Löchern (auch *Donuts* genannt) erfordern in der Datenstruktur „doppelte“ Kantenzüge, die abschnittsweise in beiden Richtungen „durchlaufen“ werden. Die aufeinanderliegenden Kantenstücke bilden keinen realen Rand des Polygons. Ein Beispiel ist in Abb. 3.6b dargestellt. Hier liegt das Kantenstück (C_8–C_9) auf dem Kantenstück (C_4–C_5). Die Koordinaten C_4 und C_9 bilden in diesem Beispiel keine realen Ecken.

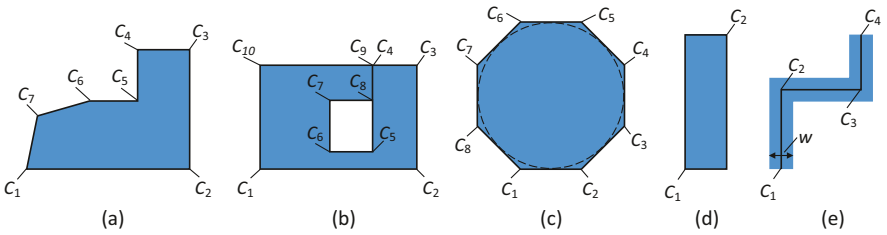

Abb. 3.6 Verschiedene Typen und Sonderformen von Shapes in den Layoutdaten, **a** allgemeines Polygon, **b** Polygon mit Loch („Donut“), **c** stückweise linearisierter Kreis, **d** Rechteck, **e** Pfad

Rundungen Grafikelemente mit kurvenförmigen Begrenzungen (manche Toolhersteller sprechen hier von „*Conics*“) lassen sich durch die vektorielle Datenstruktur nicht exakt abbilden. Kurvenförmige Begrenzungen werden stets durch gerade Kantenstücke angenähert. Die Art der Approximation variiert von Tool zu Tool; z. B. gibt es die Möglichkeit, die Anzahl der zu verwendenden Kantenstücke bezogen auf einen Kreis als Parameter zu definieren. Abb. 3.6c zeigt einen Kreis, der durch ein Polygon mit acht Ecken angenähert wird.

Neben den allgemeinen Polygonen gibt es noch zwei besondere Typen von Shapes: das *Rechteck* und den *Pfad*. Sie stellen Spezialfälle von Polygonen dar, die aufgrund ihrer besonderen Eigenschaften eine noch effizientere Darstellung in der Datenstruktur erlauben. Da Rechtecke und Pfade die mit Abstand am häufigsten vorkommenden Grafikelemente in einem typischen Layout sind, ist deren effiziente Darstellung wichtig.

Rechtecke Ein Rechteck ist ein Polygon mit vier Seiten und vier rechten Winkeln. Liegen die Seiten parallel zu den Achsen des verwendeten kartesischen Koordinatensystems (was fast immer der Fall ist), kann ein Rechteck durch die Koordinaten zweier gegenüberliegender Ecken dargestellt werden (Abb. 3.6d). Dadurch lässt sich die Datenmenge etwa halbieren. Diese Datenstruktur entspricht auch der Art, wie ein Rechteck im Grafikeditor erzeugt wird, nämlich durch Digitalisierung zweier diagonal gegenüberliegender Punkte.

Pfade Pfade sind Linienzüge, denen eine bestimmte Dicke w zugeordnet ist. Sie werden sehr häufig zur Konstruktion von Leiterbahnen genutzt, um dem elektrischen Strom einen bestimmten, konstanten Querschnitt bereitzustellen. Im Editor werden sie konstruiert, indem man die Mittellinie der Leiterbahn digitalisiert und die gewünschte Pfadweite als Parameter einstellt. Die digitalisierten Punkte (im Beispiel in Abb. 3.6e die Koordinaten C_1 bis C_4) werden zusammen mit der Pfadweite w in der Datenstruktur abgelegt. Damit lässt sich die Datenmenge gegenüber allgemeinen Polygonen ebenfalls etwa halbieren. Zudem wird dadurch eine Nachbearbeitung von Pfaden wesentlich vereinfacht.

Neben der in Abb. 3.6e dargestellten Pfadform gibt es noch weitere Sonderformen, mit denen man das Aussehen an spezielle Anforderungen der Technologie anpassen kann. Hierzu gehört insbesondere die Aufweitung der Dicke bei diagonalen Pfadsegmenten. (Manche Toolhersteller bezeichnen dies als *Padded paths*.) Dadurch lässt sich erreichen, dass die Ecken des durch den Pfad gebildeten Polygons auf das Arbeitsraster zu liegen kommen. Damit werden Rundungsfehler, die ansonsten bei der Erstellung der Maskenstrukturen entstehen können, von vornherein vermieden. Weitere Sonderformen bei Pfaden erlauben eine automatische Verlängerung (sogenannte *Extension*) am Beginn und Ende der Pfade. Auf diese Sonderformen, die auch toolabhängig sind, gehen wir nicht weiter ein. Wir empfehlen, hierzu das zugehörige Werkzeughandbuch zu Rate zu ziehen.

Kantenstücke Moderne Tools erlauben nicht nur den Zugriff auf Shapes, sondern auch auf deren Bestandteile. Insbesondere stellen die einzelnen Kantensegmente (C_i, C_{i+1}) adressierbare Datenelemente dar. In einem Layouteditor können also auch einzelne Kantenstücke bzw. Pfadsegmente selektiert und bearbeitet werden. Das ermöglicht für die Layoutarbeit besonders hilfreiche Grafikoperationen, die wir in Abschn. 3.2.3 ansprechen.

Hierarchische Organisation der Layoutdaten

Die Layoutdaten sind hierarchisch aufgebaut. Die hierarchische Struktur ist analog der Hierarchie der zugehörigen Strukturbeschreibung. Jedem Funktionsblock der Strukturbeschreibung – und somit jedem Einzelschaltplan – entspricht eine abgeschlossene Teilmenge des Gesamtlayouts, die man als *Layoutblock* oder kurz als *Block* bezeichnet. Jedem Bauelement der Strukturbeschreibung – und somit jedem Bauelementsymbol in einem Schaltplan – entspricht ein Bauelement im Layout. Unter einem *Bauelement* verstehen wir in diesem Kontext die bekannten Grundbauelemente (Transistoren, Widerstände etc.), die man im Layoutkontext auch als *Devices* bezeichnet. Blöcke und Bauelemente fasst man im Layoutkontext unter dem Begriff *Zellen* zusammen. Zellen sind also die Layoutrepräsentationen der Funktionseinheiten einer Strukturbeschreibung.

Der hierarchische Aufbau eines Layouts (die *Layouthierarchie*) ist in Abb. 3.7 als Baum veranschaulicht. Ein Layoutblock (LB) kann Bauelemente (BE) und weitere Layoutblöcke enthalten. Die Bauelementzellen bilden die unterste Hierarchieebene eines Layouts. Sie enthalten die Shapes der physikalischen Layer, aus denen im FEOL[1] der IC-Fertigung die Bauelemente entstehen (in Abb. 3.7 als „Bauelement-Shapes“ bezeichnet). Für die in analogen Schaltungen verwendeten Grundbauelemente stehen meistens Programme zur Verfügung, die das Layout des Bauelements automatisch erzeugen und als *Bauelementgeneratoren* oder *Device-Generatoren* bezeichnet werden. Dabei lässt sich die schaltungsspezifische Dimensionierung

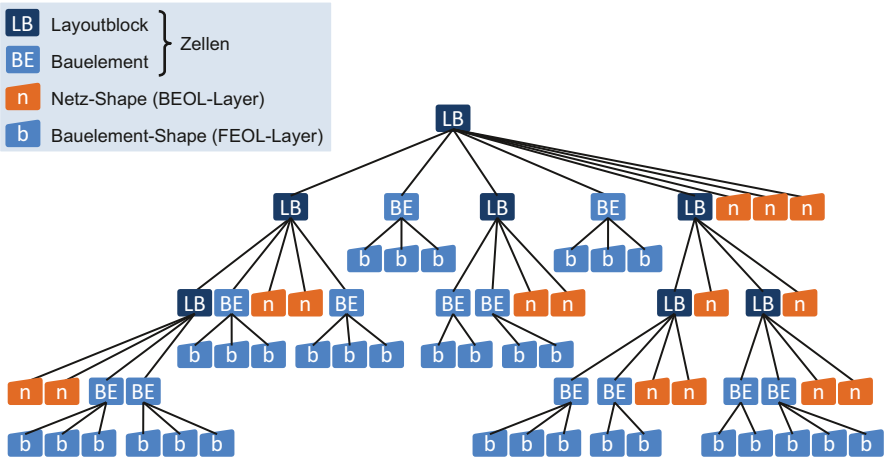

Abb. 3.7 Aus der hierarchischen Strukturbeschreibung abgeleitete Layouthierarchie. Layoutblöcke (LB), die eine Teilmenge des Gesamtlayouts darstellen, enthalten andere Blöcke (LB), Bauelemente (BE) und Netz-Shapes (n). Bauelemente bestehen aus Bauelement-Shapes (b)

[1] „FEOL“ steht für „Front-end-of-line“ und bezeichnet die erste Phase einer IC-Fertigung, in der die Bauelemente im Silizium entstehen (vgl. Kap. 1, Abschn. 1.1.3) Sie wird in Kap. 2 (Abschn. 2.10.2) detailliert vorgestellt.

über Parameter steuern. Derartige Generatoren nennt man daher auch *parametrisierte Zellen* oder *PCells* (von „parameterized cells“). Wir gehen auf PCells zur Erzeugung von Bauelementen in Kap. 6 (Abschn. 6.4) näher ein.

Des Weiteren enthält ein Layoutblock die Shapes, welche die im BEOL[2] der IC-Fertigung entstehenden Metallisierungsstrukturen definieren, also die Strukturen der physikalischen Routing-Layer (in Abb. 3.7 als „Netz-Shapes“ bezeichnet). Betrachten wir die Inhalte eines Layoutblockes, so sehen wir, dass die grafischen Strukturen der Zellen (Bauelement-Shapes) grundsätzlich in tieferen Hierarchieebenen liegen, während die grafischen Strukturen der die Zellen verbindenden Netze (Netz-Shapes) Teil des jeweiligen Layoutblockes sind. Routing-Strukturen sind innerhalb eines Blocks also immer sogenannte „flache“ Daten.

Der hierarchische Aufbau eines Gesamtlayouts findet sich nicht nur auf der Arbeitsebene des Layouters, sondern setzt sich auch in der Organisation der Entwurfsdaten im Speicher der Entwurfsumgebung fort. Die genaue Art dieser Datenorganisation hängt vom verwendeten Tool ab.

Obwohl es bei der Maskenerstellung letztlich nur auf die Shapes der hierfür relevanten physikalischen Layer ankommt, arbeitet der Layouter mit dieser hierarchisch strukturierten Layoutdarstellung. Dies hat gute Gründe. Durch die Möglichkeit, verschiedene Hierarchieebenen zu nutzen, lässt sich die zu betrachtende Komplexität massiv verringern, was den Layoutentwurf entscheidend erleichtert. Je nach Bedarf lassen sich hierarchisch tiefer liegende Details ausblenden oder man verzichtet – umgekehrt – bewusst auf die Betrachtung hierarchisch höher liegender Blöcke, um sich auf den Layoutkontext eines einzelnen Blocks zu konzentrieren. Zudem unterstützt die Layouthierarchie das Denken in Funktionseinheiten und vermittelt dem Layouter stets ein klares Bild der Schaltungstopologie, was für die vorteilhafte Anordnung und Verdrahtung der Zellen unverzichtbar ist.

Trotz dieser Erleichterung muss ein Layouter die prozesstechnische Bedeutung jedes Layers und die Auswirkung ihrer Kombination gut kennen, da es im Einzelfall durchaus notwendig ist, auch auf der Polygonebene zu arbeiten (man nennt dies „Polygon pushing“). Hierzu muss man Layouts „lesen“ können, was wir im folgenden Abschnitt üben.

3.2.2 Lesen eines Layouts

Der obere Teil von Abb. 3.8 zeigt einen Ausschnitt aus einem typischen Layout, wie es in einem Layouteditor erscheint. Wir werden dieses Beispiel nun Schritt für Schritt analysieren und dabei lernen, ein Layout zu „lesen“.

[2] „BEOL“ steht für „Back-end-of-line“ und bezeichnet die zweite Phase einer IC-Fertigung, in der die elektrischen Verbindungen entstehen (vgl. Kap. 1, Abschn. 1.1.3). Sie wird in Kap. 2 (Abschn. 2.8) beschrieben.

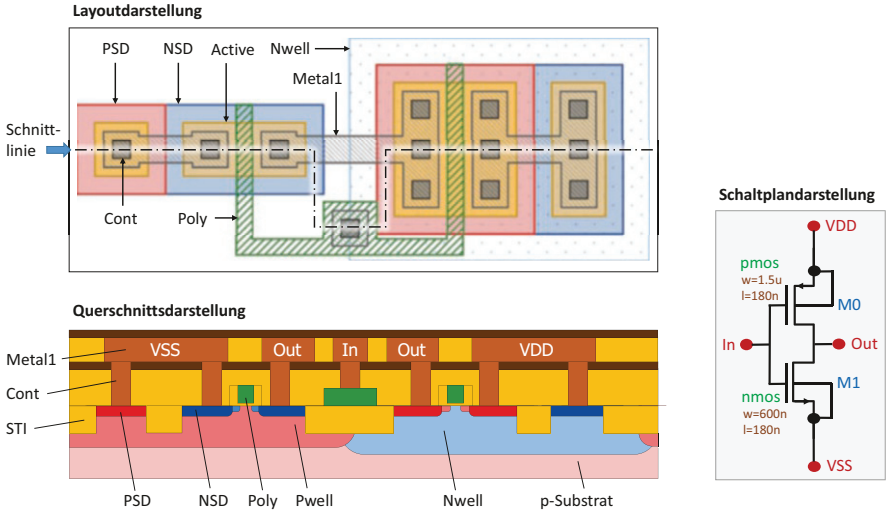

Abb. 3.8 Layout eines einfachen CMOS-Inverters (oben), wie es in einem typischen Layouteditor angezeigt wird, sowie die entsprechende Schnittdarstellung (unten) und das Schaltbild (rechts)

Die in Abb. 3.8 dargestellte Szenerie basiert auf einem CMOS-Standardprozess, wie wir ihn in Kap. 2 (Abschn. 2.10) besprochen haben. Die Querschnittsdarstellung im unteren Teil der Abb. 3.8 (wir verwenden hier die gleichen Farben wie in Kap. 2, Abschn. 2.10) zeigt die Strukturen, die aus dem darüber dargestellten Layout in der Fertigung erzeugt werden. Der Querschnitt bezieht sich auf die im Layout gestrichelt eingezeichnete Schnittlinie.

Aus der Querschnittsdarstellung, welche einen Blick „von der Seite“ zeigt, können wir erkennen, dass es sich offenbar um eine Anordnung aus einem NMOS- und einem PMOS-FET handelt. (Man vergleiche den Querschnitt der Abb. 3.8 mit Abb. 2.35 aus Kap. 2.) Im Querschnitt ist auch zu erkennen, dass die an der Siliziumoberfläche befindlichen Anschlusspunkte dieser Transistoren über Kontakte und die erste Metallebene teilweise miteinander verbunden sind. Wir können also bereits von einer „Schaltung“ sprechen.

Allerdings hat ein Layouter diese Querschnittsdarstellung nicht zur Verfügung. Er bzw. sie sieht lediglich die in Abb. 3.8 oben gezeigte Layoutdarstellung, die immer nur eine Draufsicht „von oben“ bietet. Das heißt, er bzw. sie *arbeitet* zweidimensional. *Denken* sollte man aber dreidimensional! Das „Lesen“ eines Layouts bedeutet, die Bauelemente und deren elektrische Verbindungen auf dem Chip hieraus zu erkennen und sich ihren räumlichen Aufbau in allen drei Dimensionen vorzustellen. Wie liest man nun am besten ein Layout?

Als erstes muss man die Bauelemente erkennen. Hierzu achtet man zunächst auf die Layer des FEOL. Die beste Methode, sich hier zurecht zu finden, besteht darin, sich zuerst auf die „aktiven“ Gebiete und das Polysilizium zu konzentrieren. Überall dort, wo sich Shapes dieser beiden Layer kreuzförmig überlappen, befindet sich der Kanal eines Feldeffekttransistors. Die Erkennung dieses Musters ist sehr

einfach, da man nur auf zwei Layer achten muss. Zudem hat man damit sehr oft bereits die meisten Instanzen eines Layouts identifiziert, da FETs die mit Abstand häufigsten Grundbauelemente sind. Dies gilt für Digitalschaltungen und auch für die meisten Analogschaltungen.

Als *aktive Gebiete* bezeichnet man die Stellen auf dem Chip ohne Feldoxid. In allen Prozessen gibt es hierfür einen Layer, wobei die Bezeichnungen von Hersteller zu Hersteller sehr unterschiedlich sind. In unserem Layoutbeispiel in Abb. 3.8 heißt dieser Layer „Active“ und ist ockerfarben dargestellt. Die Maske „STI“ wird aus diesem Layer durch Negation erzeugt, d. h. die Shapes in „Active“ definieren gerade die Regionen, die von STI frei bleiben (s. Querschnittsbild). „Active“ und „Poly“ (grün schraffiert) überlappen sich in unserem Beispiel an zwei Stellen. Wir haben also zwei FETs.

In Abb. 3.9 sind die Layouts und Querschnitte dieser beiden Transistoren getrennt dargestellt. In Kap. 2 (Abschn. 2.10.2) hatten wir beobachtet, wie diese Transistoren im FEOL des CMOS-Standardprozesses entstehen. Im Vergleich zur dortigen Abb. 2.35e sehen wir sie hier mit ihren in Metall herausgeführten Source-, Drain- und Bulk-Anschlusspins (gekennzeichnet durch „S“, „D“ und „B“). Das Anschlusspin des Bulks (auch Backgate genannt) gehört zu einem vollständigen Transistorlayout dazu, da hierüber das Potenzial der jeweiligen Wanne definiert wird. In den Layouts der FETs (Abb. 3.9 oben) sind auch die Abmessungen eingezeichnet, welche die Kanalweite w und Kanallänge l der beiden Transistoren bestimmen. Wie man Bauelemente im Layout dimensioniert, erläutern wir ausführlich in Kap. 6 (Abschn. 6.3).

Die Unterscheidung von NMOS- und PMOS-FETs erkennen wir am Layer „Nwell“ (hellblau getupft, s. Abb. 3.9, oben rechts), der in Prozessen mit p-Substrat die Bulk-Gebiete der PMOS-FETs definiert. Die Transistoren außerhalb von

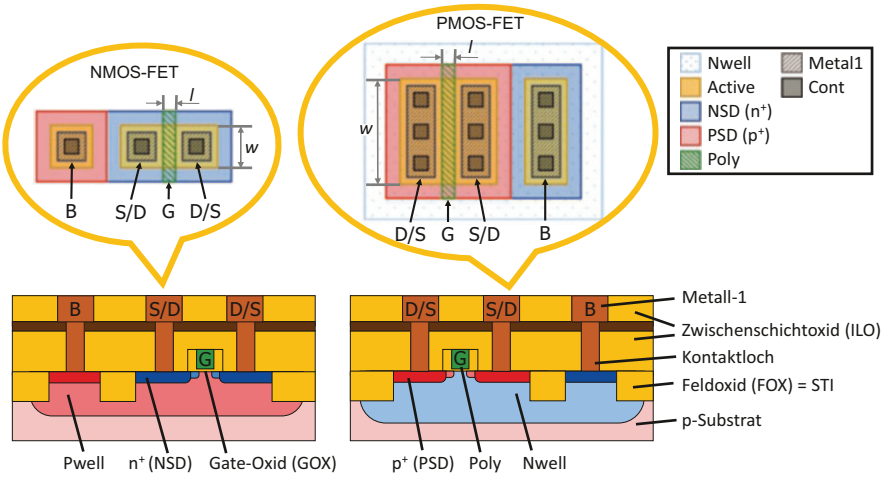

Abb. 3.9 Das Layout und die entsprechenden Schnittbilder der beiden Transistoren aus Abb. 3.8 mit markierten Kontakten (D/S: drain/source, B: bulk) und Gates (G). Transistoren lassen sich in jeder Layout-Struktur identifizieren, indem man sich auf die Kreuzungen der aktiven Flächen (Schicht „Active“, hier in ocker dargestellt) und des Polysiliziums (Schicht „Poly“, hier grün schattiert) konzentriert

Nwell-Gebieten sind folglich NMOS-FETs. Deren Bulks sind entweder das p-dotierte Substrat des Wafers (bei Single-Well-Prozessen) oder (bei Twin-Well-Prozessen) die Gebiete mit Pwell. In unserem Layoutbeispiel könnte beides der Fall sein, da die Gebiete der Pwell-Dotierung (wie wir in Kap. 2, Abschn. 2.10.2, gesehen haben) aus dem Layer „Nwell“ durch Negation abgeleitet werden können und daher im Layout nicht als eigener Layer auftauchen.

Da wir nun wissen, dass es sich um NMOS-FETs und PMOS-FETs handelt, sind auch die Dotierungsarten (n oder p) der blauen und roten Schichten, die die Source- und Drain-Bereiche bilden, klar. Üblicherweise erkennt man n- und p-Dotierung auch an den Namen der Layer. In unserem Layoutbeispiel sind dies die Layer „NSD“ und „PSD“, welche die n^+- bzw. p^+-dotierten Gebiete definieren.

Recht einfach zu erkennen sind die Layer des BEOL. Es sind zum einen die Durchkontakte, welche in modernen Prozessen normalerweise aus identischen kleinen Quadraten bestehen. Zum anderen sind es Metallstrukturen, welche die Durchkontakte immer überdecken müssen und die über die Bauelemente hinweg führenden Leiterbahnen bilden. In unserem Layoutbeispiel haben wir Kontaktlöcher im Layer „Cont“ (dunkelgrau) zur Kontaktierung der Source-, Drain- und Bulk-Gebiete. Die Leiterbahnstrukturen sind durch den Layer „Met1“ (hellgrau schraffiert) gegeben. Die durch Poly elektrisch verbundenen Gates besitzen einen gemeinsamen Anschluss in Metall, der durch denselben Layer „Cont“ kontaktiert wird.

Die beiden Transistoren sind zu einem logischen Inverter verbunden. Der Schaltplan für das Beispiel ist rechts in Abb. 3.8 dargestellt.

3.2.3 Grafik-Operationen

Moderne Layouteditoren bieten eine Fülle von Editierbefehlen und Grafikoperation an. Wir betrachten hier nur die Operationen, welche sich auf die Manipulation und Selektion von Shapes beziehen. Komfortfunktionen, wie z. B. zur Anordnung von mehreren Elementen (Verteilen, Ausrichten, Kompaktieren etc.) behandeln wir hier nicht, da sie bekannt und intuitiv anwendbar sind.

Interaktive Bearbeitung von Shapes
Selbstverständlich bieten Layouteditoren alle gängigen Grafikbefehle, die wir auch aus anderen Zeichenprogrammen kennen. Hierzu gehören das Erzeugen („Add“), Löschen („Delete“), Verschieben („Move“), Kopieren („Copy“), Einsetzen („Paste“), Spiegeln („Flip“) und Drehen („Rotate“) von Shapes. Layouteditoren bieten hier verschiedene Bedienkonzepte an, die das Arbeiten erleichtern. Neben der Eingabe mit der Maus sind zusätzlich auch numerische Eingaben möglich.

Zu diesen gängigen Standardfunktionen kommen weitere Befehle, die spezielle im Layoutentwurf nützliche Manipulation einzelner Shapes ermöglichen:

- Verschieben einer Teilmenge von Kanten oder Ecken („Stretch“),
- Ändern von Polygonen durch Ausschneiden, Abschneiden und Anfügen von Rechtecken oder komplexeren Polygonen (z. B. „Notch“),

- Zusammenführen von sich überlappenden Shapes zu einem Shape („Merge“),
- Teilen von Shapes entlang einer (beliebigen) Schnittlinie („Split“).

Logische Verknüpfungen von Layern
Die aus der Logik bekannten booleschen Operationen lassen sich auch auf Shapes unterschiedlicher Layer anwenden. Sie sind sehr wichtige und leistungsfähige Operatoren, die den grafischen Inhalt dieser Layer „logisch verknüpfen“. Sie werden zwar gelegentlich bei Layoutentwurfsarbeit eingesetzt, aber ihr Haupteinsatzgebiet ist der *Design Rule Check* (*DRC*) zur Identifizierung bestimmter Layoutkonstellationen für die Prüfung auf geometrische Entwurfsregeln (Abschn. 3.4 und Kap. 5, Abschn. 5.4.5) und im *Layout-Postprozess* (Abschn. 3.3) bei der Erzeugung von Maskendaten. Wir demonstrieren in Abb. 3.10 die folgenden, allgemein gebräuchlichen logischen Verknüpfungsoperationen:

- OR: erzeugt die geometrische Vereinigungsmenge zweier Layer,
- UND: erzeugt die geometrische Schnittmenge zweier Layer,
- XOR: erzeugt die Vereinigungsmenge abzüglich der Schnittmenge zweier Layer,
- ANDNOT: erzeugt die geometrische Differenz zweier Layer (aus dem ersten Layer wird alles „ausgestanzt“, was sich im zweiten Layer befindet).

Der obere Teil der Abbildung zeigt ein einfaches Beispiellayout, bestehend aus vier rechteckigen Shapes, von denen zwei dem Layer „rot“ und die anderen beiden dem Layer „blau“ angehören. Die Ergebnisse der Operationen werden in eine neue Ebene „x“ geschrieben, die unten in Abb. 3.10 grau dargestellt ist.

Selektierbefehle
Mit speziellen Selektierbefehlen lassen sich aus einem Layer diejenigen Shapes herausfiltern, die ein Auswahlkriterium erfüllen. In Abb. 3.11 demonstrieren wir einige wichtige Auswahlkriterien, die auf bestimmten Relationen zwischen den Shapes der beteiligten Layer basieren.

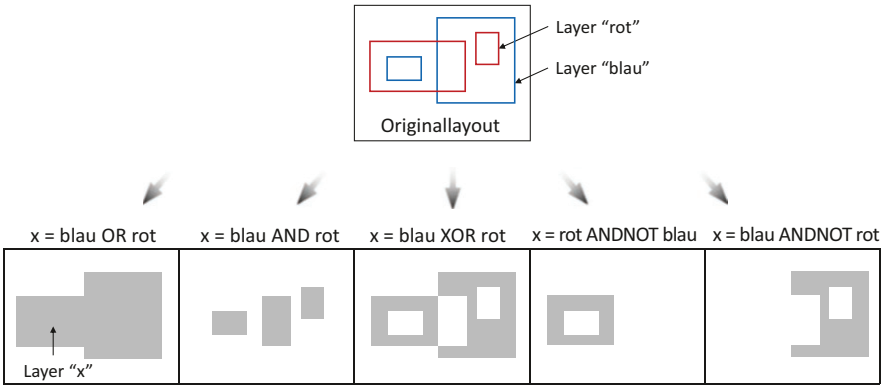

Abb. 3.10 Layoutbeispiel mit vier Shapes in zwei Layern (oben). Ergebnis aus der Anwendung der logischen Verknüpfungsoperatoren OR, AND, XOR, ANDNOT auf das Layoutbeispiel (unten, von links nach rechts)

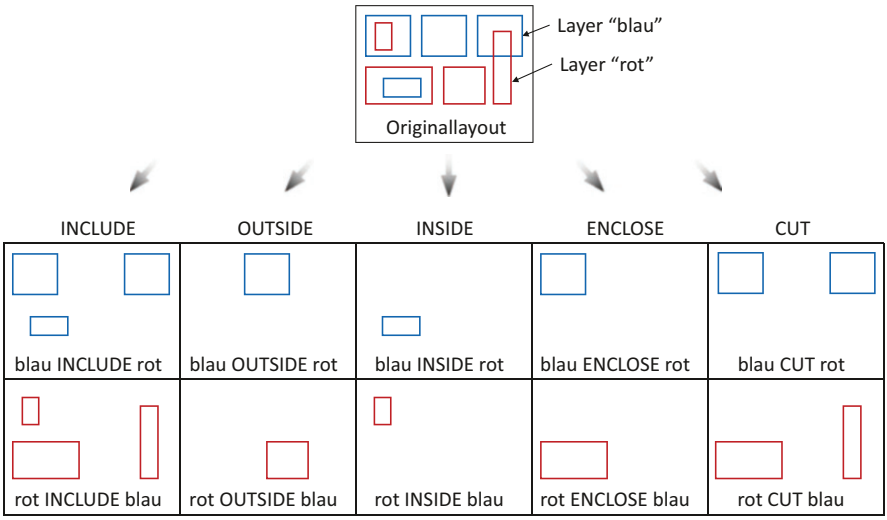

Abb. 3.11 Selektierbefehle zum Filtern von Shapes eines Layers basierend auf geometrischen Relationen zwischen diesen Shapes und den Shapes eines anderen Layers

- INCLUDE: wählt Shapes eines Layers aus, die sich in irgendeiner Weise mit Shapes eines zweiten Layers überschneiden,
- OUTSIDE: wählt Shapes eines Layers aus, die sich nicht mit Shapes eines zweiten Layers überschneiden,
- INSIDE: wählt Shapes eines Layers aus, die vollständig von Shapes eines zweiten Layers bedeckt werden,
- ENCLOSE: wählt Shapes eines Layers aus, die Shapes eines zweiten Layers vollständig bedecken,
- CUT: wählt Shapes eines Layers aus, die einen Teil ihrer Fläche mit Shapes eines zweiten Layers teilen, aber nicht ihre ganze Fläche.

Bei den Selektierbefehlen entstehen im Gegensatz zu den Verknüpfungsoperationen keine neuen Geometrien. Bei Bedarf können die Ergebnisse in einem neuen Layer abgespeichert werden. Die Selektierbefehle sind insbesondere für den DRC von großer Bedeutung, da man hiermit interessierende Teilmengen von Layern selektieren kann (Abschn. 3.4.3, Beispiel 3).

Sizing

Beim Sizing verändert man ein Polygon, indem alle Kanten um einen bestimmten Wert senkrecht zur Kantenausrichtung verschoben werden. Bei einer Verschiebung um positive Werte findet eine Vergrößerung des Polygons statt. In diesem Fall spricht man vom *Oversizing*. Bei negativen Werten handelt sich um ein sogenanntes *Undersizing*, wodurch sich das Polygon verkleinert.

Die Sizing-Operation haben wir bereits in Kap. 2 (Abschn. 2.4.2) zur Umsetzung von Technologievorhalten kennengelernt. (Dabei werden im Fertigungsprozess auf-

tretende Kantenverschiebungen durch Sizing der Layoutstrukturen mit umgekehrtem Vorzeichen kompensiert.) Sizing ist auch im DRC sehr hilfreich, um Layouts auf Einhaltung komplexerer Entwurfsregeln zu überprüfen (Abschn. 3.4.3, Beispiele 2 und 4).

Das Sizing hat einige besondere Eigenschaften, die zu überraschenden Ergebnissen führen können, wenn man diese Eigenschaften nicht kennt. Man kann diese Eigenschaften auch bewusst nutzen, um bestimmte Effekte zu erzielen. Beispielsweise lassen sich mithilfe des Sizings Layouts „bereinigen“. Diese Effekte werden wir uns nun anschauen.

Ungleichmäßiges Wachstum beim Sizing

Beim Sizing um einen Wert s werden die Ecken eines Polygons um eine Strecke $v{\cdot}s$ verschoben, wobei grundsätzlich $v > s$ ist, d. h. die Ecken entfernen sich von ihren ursprünglichen Positionen immer weiter als der Verschiebungswert s. Für rechte Winkel ist z. B. $v = \sqrt{2}$. Bei spitzen Winkeln (Winkel < 90°) wird der Wert v größer als $\sqrt{2}$ und kann theoretisch über alle Grenzen wachsen (Abb. 3.12, links). Dieser Effekt ist ein prinzipieller Fehler des Sizings, da man i. Allg. ein „gleichmäßiges“ Wachstum in alle Richtungen wünscht.

Hierfür müssten beim Oversizing idealerweise an den Ecken Kreisbögen entstehen, welche aber mit der vektoriellen Datenstruktur nicht darstellbar sind. Abhängig vom verwendeten Tool gibt es daher die Möglichkeit, das Oversizing so zu konfigurieren, dass die Ecken durch zusätzliche Kanten „abgeschrägt“ werden, um sich einem Kreisbogen anzunähern, wie in Abb. 3.6c dargestellt. Abb. 3.12 (rechts) zeigt hierzu zwei Beispiele.

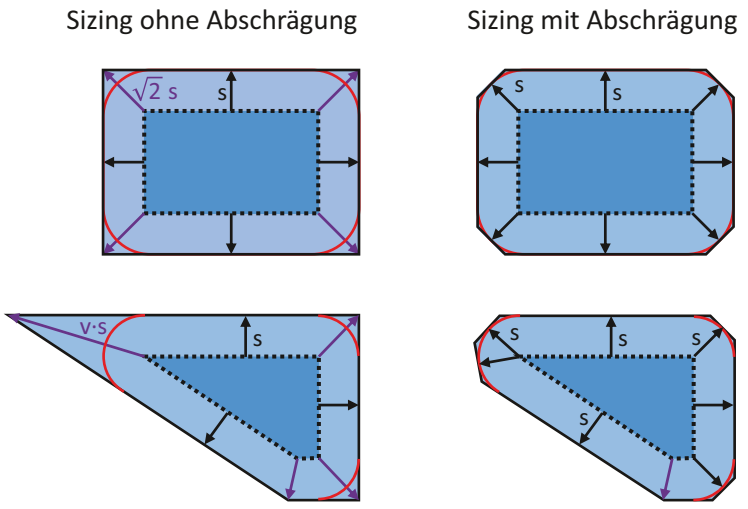

Abb. 3.12 Beispiele zum Oversizing ohne (links) und mit abgeschrägten Ecken (rechts)

Irreversibilität und Rundungsfehler
Führt man zwei Sizing-Operationen, die den gleichen Betrag aber umgekehrtes Vorzeichen haben, direkt hintereinander aus, so führt dies nicht immer zu den ursprünglichen Strukturen. Hierfür gibt es folgende Ursachen:

- Beim Undersizing verschwinden kleine Polygone (auch schmale Stege), Abb. 3.13a,
- Beim Oversizing verschwinden kleine Löcher (auch schmale Schlitze) in Polygonen, Abb. 3.13b,
- Rundungsfehler verursachen Formveränderungen, Abb. 3.13c.

Ursache der Formveränderungen ist, dass im Falle schräger Kanten die Ecken nach dem Sizing rechnerisch nicht auf dem Arbeitsraster zu liegen kommen. Durch die Verwendung von Integer-Werten bei den Koordinaten entstehen Rundungsfehler. Diese können oft vernachlässigt werden. Im Allgemeinen verändern sich dadurch allerdings die Winkel gegenüber den Achsen des Koordinatensystems (Abb. 3.13c), was in manchen Fällen zu unerwünschten Ergebnissen führt. Beispielsweise können durch diesen Effekt Entwurfsregeln verletzt werden, die bei mathematisch exaktem Sizing nicht aufgetreten wären.

Bereinigung von Layouts
Nutzt man Sequenzen aus Sizing-Operationen und logischen Verknüpfungen zur Erzeugung bestimmter Layoutstrukturen (z. B. im Postprozess oder im DRC), so kann es insbesondere durch Rundungsfehler zur Erzeugung von unerwünschten Shapes kommen, die i. Allg. sehr kleine Abmessungen haben. Die in Abb. 3.13a, b beschriebenen Effekte können dazu genutzt werden, um derartige Artefakte zu eliminieren, d. h. Layouts lassen sich auf diese Weise „bereinigen“.

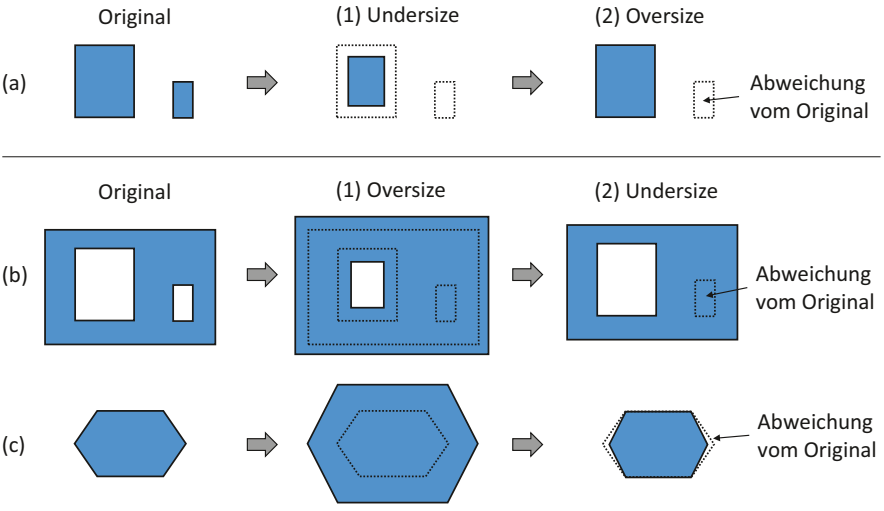

Abb. 3.13 Effekte bei zweimaligem Sizing mit gleichem Betrag und umgekehrtem Vorzeichen. Es können Polygone entstehen, die von der ursprünglichen Form abweichen