Применение алгоритмов анализа изоморфизма графов для схемо-топологической верификации сложных микроэлектронных объектов

Г. А. Мамедов[1], С. Э. Миронов[2], А. И. Тихонова[3]

Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)

[1]geidar.mamedov1@gmail.com, [2]semironovspb@yandex.ru, [3]ai.tikhonova@mail.ru

Аннотация. **В работе приводятся результаты исследований авторов в области схемо-топологической верификации. Описываются методы и средства принятия решений о соответствии результатов топологического проектирования заданной электрической схеме. Верификация проводится путем построения графов исходной электрической схемы и схемы, извлеченной из топологии, и их анализа на изоморфизм.**

Ключевые слова: электронная схема; топология; граф; изоморфность

I. Введение

Микроэлектроника представляет собой раздел электроники, связанный с разработкой, изготовлением и использованием электронных устройств, обладающих минимально возможными габаритами и достаточно высокой надежностью [1]. В настоящее время наблюдается постоянный и все ускоряющийся рост уровня сложности реализуемых микроэлектронных схем. Число активных элементов в составе одной схемы постоянно увеличивается. Такие схемы обладают малыми габаритами и при этом способны обеспечивать довольно высокую производительность. Однако, вместе с этим происходит постоянное усложнение электронных схем, увеличение числа ячеек и их типов и расширение номенклатуры изготавливаемых изделий [2]. С постоянным ростом сложности возрастает вероятность возникновения ошибок при построении электронных схем. По данной причине, возникает необходимость в автоматизированной верификации результатов, полученных на разных стадиях проектирования электронных схем. В данной работе представлен способ проверки соответствия топологии электрической схеме путем поиска изоморфизма графов исходной электрической схемы и схемы, извлеченной из топологии.

Прежде чем переходить к верификации построенных схем, необходимо обозначить переход от системы внешних имён топологических слоев или их комбинаций к системе обозначений принятой в разработанной САПР. Описание топологии выполняется в конкретных использованных при проектировании технологических нормах на одном из используемых стандартных языков метрического описания реальной топологии. Потому перед работой с топологией ее описание следует перевести с традиционного языка на язык описания топологии в разработанной САПР. Такая трансляция предусматривает преобразование описания топологии, обеспечивающее переход:

- от реальных топологических слоев к виртуальным слоям конструктива;
- от реальных метрических единиц измерения к абстрактным виртуальным единицам;
- от операторов языка метрического описания реальной топологии к операторам языка символьного описания виртуальной топологии.

Первый тип преобразования можно проиллюстрировать с помощью рис. 1, на котором приведены примеры выделения комбинаций топологических примитивов в разных топологических слоях и замены их элементами слоев виртуального конструктива.

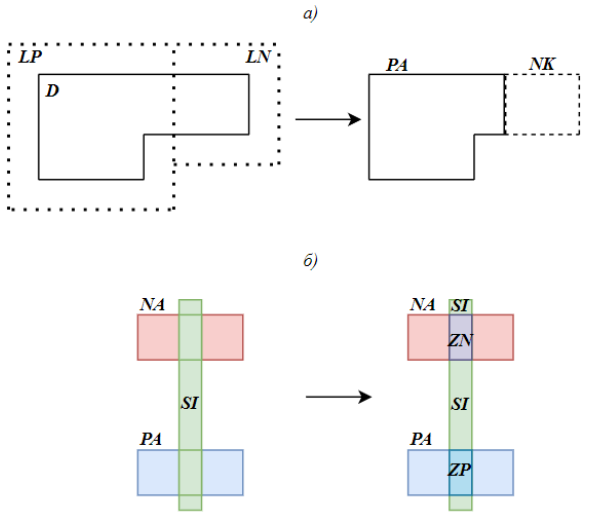


Рис. 1. Пример перехода от топологических слоев к конструктивным слоям: а) преобразование диффузионной области с учетом типа легирования в активную и контактную области; б) разбиение поликремния на межсоединения и затворы транзисторов с разным типом проводимости. Обозначения: D – диффузионная область; LP – легирование -типа; LN – легирование n-типа; PA – активная область p-типа; NA – активная область n-типа; NK – контактная область n-типа; SI – поликремний; ZN – затвор транзистора n-типа; ZP – затвор транзистора p-типа

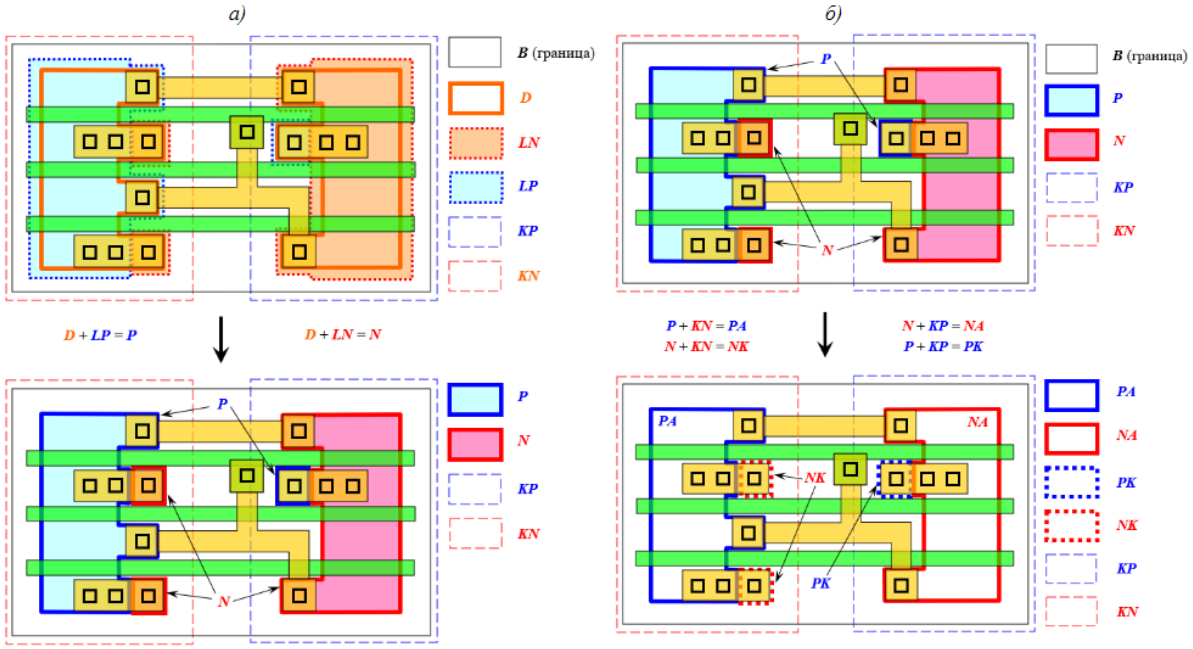


Рис. 2. Подробное описание перехода от реальной топологии к описанию виртуальной топологии: а)от описания топологии с одним типом диффузии (D) и двумя типами легирования (LN и LP) к описанию топологии с двумя типами диффузии (N, P) б) от описания топологии с двумя типами диффузии (N и P) к описанию виртуальной топологии с элементами в активных (NA и PA) и контактных (NK и PK) виртуальных слоях конструктива

На рис. 1 приведены примеры выделения комбинаций топологических примитивов в разных топологических слоях и замены их элементами слоев виртуального конструктива (примитивами символьного описания топологии). Рис. 1*а* демонстрирует преобразование диффузионной области топологического чертежа реальной топологии с учетом типа легирования в активную область транзистора и контактную область к карману в технологически инвариантном описании топологии (топологическом эскизе). Более детально этот процесс преобразования представлен на рис. 2 и 3, где помимо перехода от описания реальной топологии к описанию виртуальной топологии, от описания топологии с одним типом диффузии (*D*) и двумя типами легирования (*LN* и *LP*) к описанию топологии с двумя типами диффузии (*N* и *P*). Рис. 1*б* демонстрирует разбиение поликремниевой шины на поликремниевые межсоединения и затворы транзисторов с разным типом проводимости.

II. Переход к от схемы к графу

Перед построением по топологии или электрической схеме графа введем понятие того, что представляет собой вершина такого графа. На рис. 3 представлено преобразование транзистора в вершину графа.

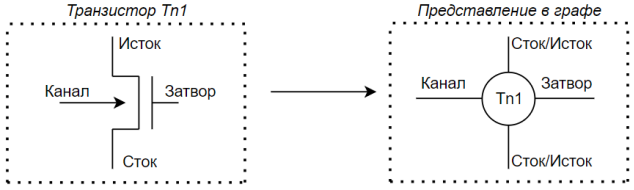


Рис. 3. Преобразование транзистора в узел графа

A. *Извлечение графа из электрической схемы*

Одним из форматов описания электрической схемы является net-list, который представляет собой текстовое описание входящих в состав схемы элементов со строго регламентированной последовательностью перечисления их выводов. В формате четко описаны соединения между компонентами, что важно для процесса разработки электронных схем. В net-list включена следующая информация:

- обозначение элемента, например, TpA;
- номера контактов;
- ключевые слова сигнала (GND, VCC);
- тип элемента (например, в случае транзистора: MbreakP (транзистор с каналом -типа), MbreakN (транзистор с каналом n-типа)).

На рис. 4 представлена схема LOGIC, а также приведено ее описание в формате net-list. Схема LOGIC состоит из 4 транзисторов -типа и 4 транзисторов n-типа. Входы схемы: A, B. Выход схемы – out. Описание в формате net-list схемы LOGIC происходит, например, следующим образом: транзистор -типа с названием TpA соединен с VCC (сток), входом A (затвор), узлом P1 (исток), канал транзистора подключен к VCC (подложка).

Осуществим перевод электрической схемы LOGIC в формат графа из формата Netlist. Формат NETlist крайне удобен для работы: в нем есть только информация о смежности отдельных компонент. Для построения графа достаточно считывать файл построчно, и каждый новый элемент добавлять в граф как вершину, при условии того, что его еще там нет.

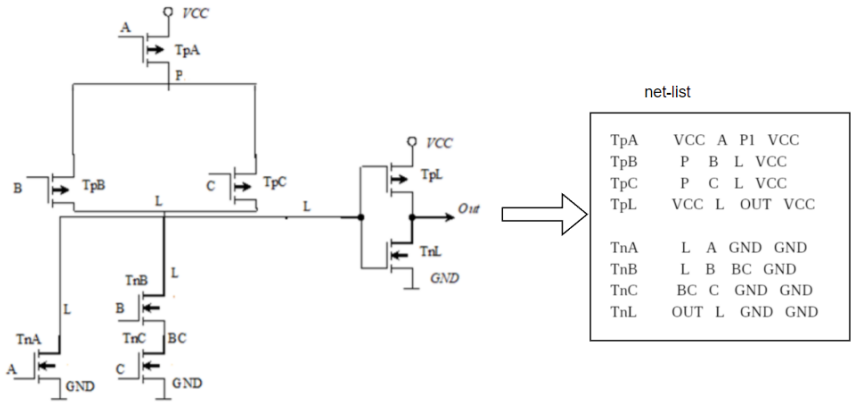


Рис. 4. Схема LOGIC и ее net-list

После добавления всех необходимых вершин, в разработанной системе верификации *Skim-Layout-Verification* идет построение ребер между вершинами в соответствии с исходным файлом. В итоге получается граф, представленный на рис. 5.

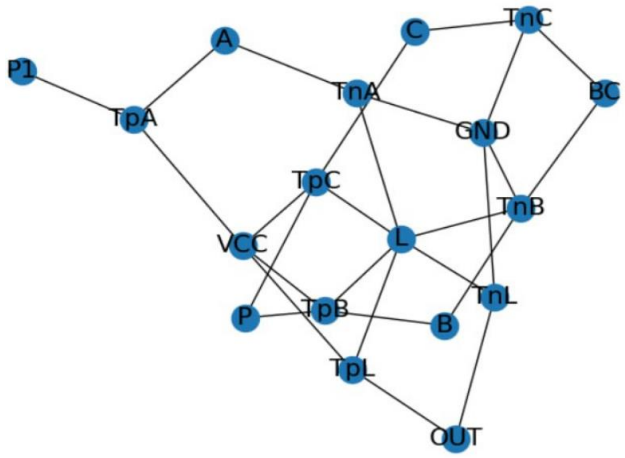


Рис. 5. Построение графа электрической схемы, описанной в формате net-list (окно системы «*Skim-Layout-Verification*»)

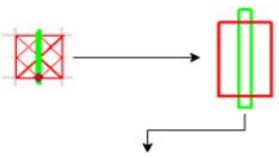
B. Извлечение графа из топологии

Описание топологии, состоящее из описания геометрических фигур в различных топологических слоях, выполнено в соответствии со стандартами языка CIF (Caltech Intermediate Form). CIF предоставляет ограниченный набор графических примитивов, которые полезны для описания двумерных форм на различных слоях микросхемы [3]. Формат допускает иерархическое описание, что делает представление лаконичным. Каждый оператор в CIF состоит из ключевого слова или буквы, за которыми следуют параметры и завершаются точкой с запятой. Параметры должны разделяться пробелами, но нет никаких ограничений на количество операторов в строке или на конкретные столбцы любого поля. Комментарии можно вставить в любое место, заключив их в круглые скобки. Существует всего несколько операторов CIF, и они относятся к одной из двух категорий: геометрия или управление. К геометрическим операциям относятся: LAYER для переключения слоев маски, BOX для рисования прямоугольника, WIRE для рисования пути, ROUNDFLASH для рисования окружности, POLYGON для рисования произвольной фигуры и CALL для рисования подпрограммы других геометрических операторов. Управляющие операторы: DS – для начала определения подпрограммы, DF – для завершения определения подпрограммы, DD – для удаления определения подпрограмм, от 0 до 9 – для включения дополнительной информации, заданной пользователем, и END – для завершения CIF-файла. Все эти ключевые слова обычно сокращаются до одной или двух букв, которые являются уникальными. В качестве примера на рис. 6 приведено описание прямоугольника в топологическом слое “SI” (поликремний).

```
L SI;
P 100 -140 380 -140 380 140 100 140;
```

Рис. 6. Описание прямоугольника в топологическом слое “SI” (поликремний)

Ряд элементов топологии представляет собой набор некоторых полигонов. Например, транзистор N-типа представляется как компоненты в двух слоях: SN (затвор) и NA (диффузия). На рис. 7 слева представлен вид транзистора в системе проектирования топологии TopDesign [4]: справа показан его перевод в графический формат CIF, снизу – его текстовый вид.



```
L SN;
P 70 -10 190 -10 190 970 70 970;
L NA;
P -110 110 370 110 370 850 -110 850;
```

Рис. 7. Пример описания транзистора в формат CIF

На рис. 8 продемонстрирован результат восстановления графа схемы LOGIC из топологии в формате vlf [5] (рис. 9) системы TopDesign.

Основным шагом в проверке изоморфности топологий является перевод топологии в формат графа. Для выполнения данного шага использовались ключевые данные технологий. В рамках работы рассматривается технология КМОП.

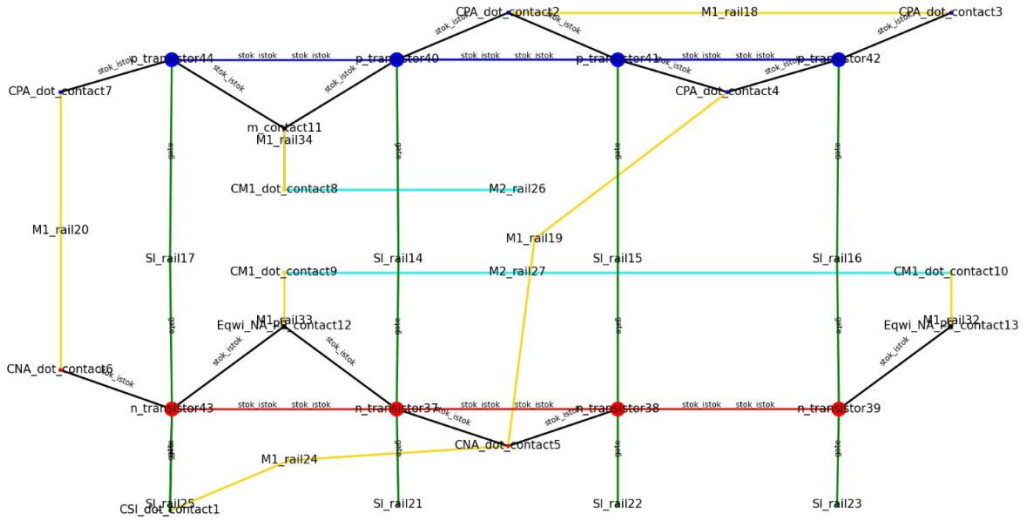


Рис. 8. Граф из топологии CIF (Обозначения: красный – n-транзисторы, желтый – шина M1, зеленый – шина Si, голубой – шина M2, синий – p-транзисторы, черный – разные виды контактных окон)

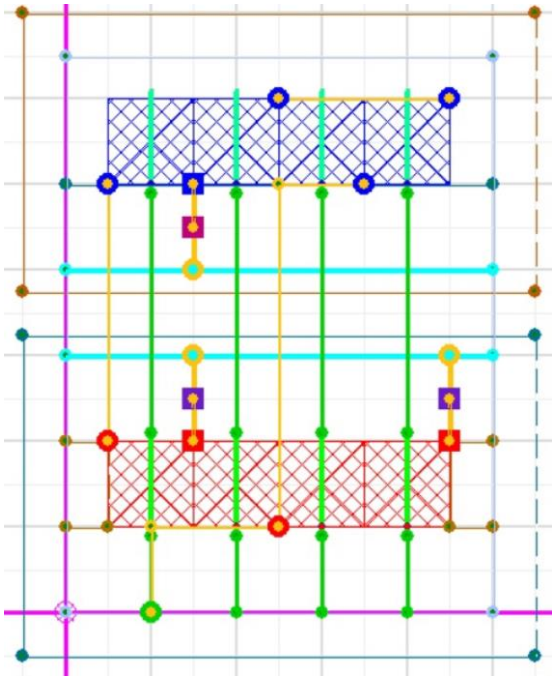


Рис. 9. Топология ячейки LOGIC в формате vlf

Например, для обработки контактного окна CM1 необходимо иметь заранее информацию о всех используемых слоях для контакта, в данном случае это список «CM1», «M1», «M2».

Также необходимо структурировать информацию о каждом из слоев. Хранить список координат в формате List недостаточно, так как в дальнейшем в программе понадобится вычисления различных свойств данных компонент, что ресурсозатратно. Было принято решение использовать библиотеку Shapely для работы с многоугольниками. Данный модуль предлагает множество удобных функций, таких как centroid – получение центроида многоугольника, а также contains – функция проверки вложенности одного многоугольника в другой, при этом Shapely производительней стандартного List.

Имея представление о технологии и всех ее элементах и том, как с ними работать, необходимо подготовить программу для обработки файла топологии. Сначала следует занести это представление в отдельный файл tehnol.txt, где данные сохранены в формате JSON словаря. Считав данное руководство, в программе создается отдельный DataClass для каждого из компонентов, где название класса – название компонента, а его поля – соответствующие ему слои и их координаты в формате Polygon. Это делается для жесткой структуризации данных, что делает дальнейшую работу с ними проще.

Затем происходит обработка файла самой топологии. Данные считываются линия за линией, и ориентируясь на руководство, сразу же переводятся в формат DataClass каждой из компонент, а затем добавляются в общий словарь полученных компонент Data. Тем самым мы получаем набор отдельных компонент, где каждая компонента представлена набором многоугольников.

Следующим шагом является построение графа. В качестве вершин используются ключи словаря Data, а для построения ребер используется следующий алгоритм. Из словаря выделяются все контактные компоненты, и идет проверка вложенности данной компоненты в другие. Если такие компоненты имеются, то между ними в графе строится ребро.

Тем самым из топологии шагами, описанными выше, получается граф, представленный на рис. 9, который описывает внутреннюю структуру топологии. Стоит отметить, что при переводе транзисторов из топологии в граф сохраняется, к какой именно части транзистора были произведены соединения: сток/исток, канал, затвор.

III. Изоморфизм графов

В библиотеке NetworkX представлен алгоритм VF2++ [6] для определения являются ли графы G_1 и G_2 изоморфными. Данный алгоритм использует упорядочение узлов: каждому узлу ставится определенная метка, после учета степени и редкости

метки каждого узла, они размещаются в оптимальном порядке (вначале размещены узлы, которые с большей вероятностью не совпадут). Использование такой сортировки узлов снижает вероятность того, что сначала алгоритм будет исследовать вершины с малой степенью, по которым не всегда можно определить, изоморфны ли графы.

Данный алгоритм удобен в использовании для полученных из предыдущих разделов графов, так как эти графы приведены к единому формату NetworkX. Также данный алгоритм позволяет выделить, что именно различается в двух графах – отсутствие контактного окна или же несоответствие количества транзисторов. При отсутствии изоморфизма программа выводит список вершин или ребер, которые присутствуют в одном графе и отсутствуют в другом.

В качестве примера сравним две построенные электрические схемы, преобразованные в граф: первая – LOGIC, вторая – LOGIC с удаленным ребром между одной из SI шин и транзисторов. Результат работы алгоритма представлен на рис. 10.

```
Missing values are edge(SI_rail16, p_transistor42)

Process finished with exit code 0
```

Рис. 10. Результат работы алгоритма VF2++

IV. Заключение

В ходе исследований реализована система схемо-топологической верификации «*Skim-Layout-Verification*».

В работе рассмотрены вопросы представления топологии в различных форматах. Описаны разработанные алгоритмы:

- перевода данных электрической схемы в формате net-list в формат графа;
- перевода данных топологии в формате CIF в формат графа;
- преобразования графа топологии в граф электрической схемы.

Данные алгоритмы позволяют анализировать изоморфность графов электрических схем и их топологических реализаций и, тем самым, находить ошибки, допущенные при проектировании.

Направлениями дальнейшего развития системы «*Skim-Layout-Verification*» может быть распространение на другие технологии, а также исследование вопросов повышения производительности на основе других алгоритмов анализа изоморфности.

Список литературы

[1] Свистова Т.В. Основы микроэлектроники: Учебное пособие. Воронеж: ФГБОУ ВО" Воронежский государственный технический университет. 2017.

[2] Миронов С.Э., Андреев Л.Е. Схемная верификация топологии ячеек БИС // Известия СПбГЭТУ ЛЭТИ. 2013. №. 2. С. 38-42.

[3] Sprou1I R., Lyon R., & Trinberger S. (1980). The Caltech intermediate form for LSI layout description.

[4] Зуев И.С., Миронов С.Э., Сафьянников Н.М. Проектирование специализированных кремниевых компиляторов в САПР параметризованных фрагментов КМОП БИС TopDesign. СПб: Изд-во СПбГЭТУ «ЛЭТИ», 2017. 226 с.

[5] Зуев И.С. САПР TopDesign виртуального символьного проектирования параметризованных фрагментов КМОП БИС.

[6] Jüttner A., Madarasi P. VF2++ — An improved subgraph isomorphism algorithm //Discrete Applied Mathematics. 2018. Т. 242. С. 69-81.