УДК 519.688

Использование задачи булевой выполнимости для трассировки стандартных ячеек промышленной библиотеки элементов

Н.В. Рыженко

ЗАО «Интел А/О», nikolai.v.ryzhenko@intel.com

***Аннотация* — В данной работе представлен алгоритм трассировки стандартных ячеек, использующий постановку задачи булевой выполнимости *(Boolean Satisfiability, SAT)*. На предварительном этапе все электрические цепи ячейки разбиваются на двухточечные соединения. Для каждого соединения создаётся список из нескольких возможных топологий. Затем специальная процедура определяет конфликты между каждой парой возможных топологий, используя как электрические правила, так и геометрические литографические правила для данной технологии. Для возможных топологий и списка конфликтов выполняется постановка задачи булевой выполнимости. Решение данной задачи определяет полную трассировку стандартной ячейки. В работе также представлен подход по выбору перспективных возможных топологий, позволяющий значительно уменьшить размер поставленной задачи. Другой подход, направленный на улучшение качества трассировки, использует последовательные булевы ограничения. Предложенный алгоритм реализован в качестве программного комплекса. Комплекс используется для трассировки стандартных ячеек промышленной библиотеки элементов, демонстрируя приемлемое время работы и 89% покрытие библиотеки, включая сканирующие триггеры, сумматоры и мультиплексоры.**

***Ключевые слова* — трассировка, стандартные ячейки, библиотека элементов, булева выполнимость.**

I. Введение

Проектирование топологии стандартных ячеек требует значительного ручного труда специальных инженеров *(mask designers)*. В компании Intel, ручное и слабо автоматизированное проектирование стандартных ячеек составляет до 50% времени от всего цикла разработки библиотеки элементов. Большую часть данного времени составляет трассировка размещённых транзисторов.

Известные промышленные алгоритмы трассировки транзисторов на уровне стандартной ячейки сводятся к традиционным итерационным подходам. Канальная трассировка используется в [1]-[2]. Волновой алгоритм реализован в [3]. Последовательный алгоритм применяется в [4] для трассировки экспериментальной модели топологии с использованием «диагональных» слоёв металлизации. Последовательные итерационные алгоритмы многократно трассируют одни и те же соединения. Инкрементальная трассировка используется как для обеспечения полной связности электрической схемы элемента, так и для достижения заданных электрических параметров стандартной ячейки. Как результат, естественной проблемой последовательных итерационных подходов является трассировка, выходящая за границы минимального прямоугольника на границах портов, которая приводит к увеличению ёмкости входных портов и времени распространения сигнала от входа ячейки к выходу.

Сортировка цепей по критичности, использование участков топологии, добавленных в ячейку вручную, шаблонная трассировка и прочие методы снижают негативный эффект от последовательной трассировки цепей, но не устраняют фундаментальных причин появления нежелательных длинных цепей трассировки. Переход к нанометровым технологиям производства интегральных схем делает использование последовательных подходов ещё менее целесообразным. Поочерёдное построение и перестроение цепей не позволяет эффективно учитывать всё многообразие современных правил, включающих сразу до нескольких объектов, лежащих на смежных слоях.

Важно отметить, что полупроводниковая индустрия движется в сторону регулярности топологии [12]. Обусловлено это исключительными техническими проблемами нанометровой оптической литографии. Регулярность топологии значительно сужает область поиска допустимого решения и позволяет использовать точные методы для ограниченного класса элементов. К таковым относятся стандартные ячейки с фиксированными масками диффузии и поликремния и с «программируемыми» переходами между слоями металлизации [5]-[6]. На топологии экстремальной регулярности используются даже полностью переборные методы [7].

В данной работе именно регулярность и дискретность топологии позволяют использовать постановку задачи булевой выполнимости *(Boolean Satisfiability, общепринятое международное сокращение - SAT)* для построения полной, корректной и оптимизированной трассировки транзисторов на уровне стандартных ячеек. SAT, как универсальный инструмент, широко используется для решения задач, поставленных для ограниченного числа объектов, для которых определены произвольные ограничения, выраженные с помощью элементарных булевых выражений [4], [8]-[11].

В данной работе постановка задачи SAT для трассировки стандартных ячеек – совместна для всех электрических соединений. Это фундаментальное отличие от последовательных алгоритмов: определив допустимое решение SAT, мы автоматически определяем полную трассировку всей стандартной ячейки; дополнительная перетрассировка отдельных сегментов цепей не требуется по определению. Также, SAT позволяет задавать с помощью булевых выражений произвольные правила и ограничения между проводниками и межслойными переходами. Третье, и немаловажное, преимущество заключается в том, что SAT относится к классу точных алгоритмов. Он или находит некое решение задачи, или же определяет с абсолютной точностью, что допустимого решения не существует.

Основной фактор SAT, лимитирующий область его применения, – это время, необходимое специальным программам *(SAT solvers)* для поиска решения. Так, в работе [8] SAT используется для построения грубой глобальной трассировки ячеек минимального размера. В работах [9] и [10] SAT применяется для трассировки регулярных структур. В работе [4] задача SAT используется для окончательной трассировки четырёх и менее незаконченных соединений. В работе [11] постановка задачи SAT используется для пере-трассировки небольшого числа цепей в ограниченных проблемных регионах.

Естественная эволюция полупроводниковой индустрии в направлении регулярной и дискретной топологии [12], существенный прогресс в разработке программных комплексов для решения задач SAT [13] и эффективная процедура отбора недостижимых топологий соединений, описанная в данной работе, позволяют использовать SAT для трассировки промышленных библиотек. Использование унарного счётчика, предложенное в данной работе, обеспечивает максимально возможное качество трассировки среди списка возможных топологических реализаций.

Содержание работы следующее. Задача трассировки формулируется в главе 2. Этапы маршрута трассировки представлены в главе 3. В главе 4 мы детально описываем этапы, предшествующие постановке задачи SAT. В главе 5 представлена постановка задачи SAT для детальной трассировки стандартной ячейки. В главе 6 описан подход, использующий унарный счётчик для улучшения качества трассировки. Экспериментальные результаты представлены в главе 7.

II. Задача трассировки

Каждая электрическая цепь в стандартной ячейке имеет терминалы двух типов: поликремниевый затвор и контакт диффузии стока/истока транзистора. Мы различаем цепи земли и питания, порты и сигнальные цепи. Порт может быть входным или выходным. Порты используются для трассировки ячеек друг с другом на блочном уровне. Задача трассировки заключается в том, чтобы соединить все терминалы электрических цепей проводниками без электрических замыканий и с соблюдением технологических правил; терминалы земли и питания должны быть соединены с соответствующими сетями проводников, выполненными на блочном уровне; каждый порт должен иметь контактную площадку на соответствующем слое металлизации, который используется для трассировки стандартных ячеек между собой на блочном уровне.

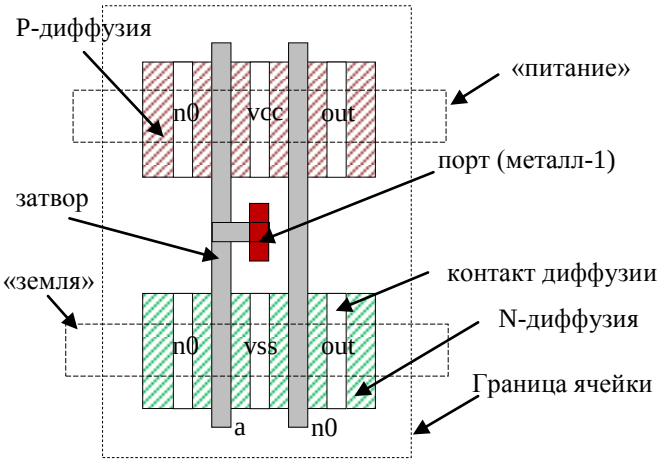

Рис. 1. Базовые элементы стандартной ячейки

Без ограничения общности предлагаемого подхода SAT, мы используем следующие технологические особенности проектирования стандартных ячеек. 1) Металл-1 – верхний слой металлизации стандартной ячейки. Использование горизонтального металла-2 для трассировки внутренних цепей стандартной ячейки допустимо, но нежелательно. 2) Каждый порт должен иметь, по крайней мере, одну контактную площадку на металле-1. 3) Цепи земли и питания выполнены в виде полос металла-2, проходящих через всю ячейку от края до края. Желательно прямое подключение стоков и истоков к данным полосам. Также допускаются соединения между терминалами земли и питания внутри ячейки. 4) Затворы транзисторов, примыкающие в вертикальном направлении, соединяются по слою поликремния до этапа трассировки (цепи «a» и «n0» на рис. 1).

III. Краткое описание маршрута трассировки

Маршрут трассировки отдельной стандартной ячейки представлен на рис. 2.

На этапе 1 трассировщик создаёт виртуальные соединения для каждой пары терминалов. В общем случае, для N терминалов цепи создаётся $(N^2 - N)/2$ соединений (рис. 3). Для решения задачи трассировки, все терминалы цепи должны быть соединены ацикличным набором из $(N - 1)$ соединений, и должен существовать непрерывный путь из соединений для каждой пары терминалов. Часть соединений может быть удалена из рассмотрения из-за дополнительных ограничений, например, на длину, но на данном предварительном этапе мы не решаем задачу окончательно выбора указанных $(N - 1)$ соединений.

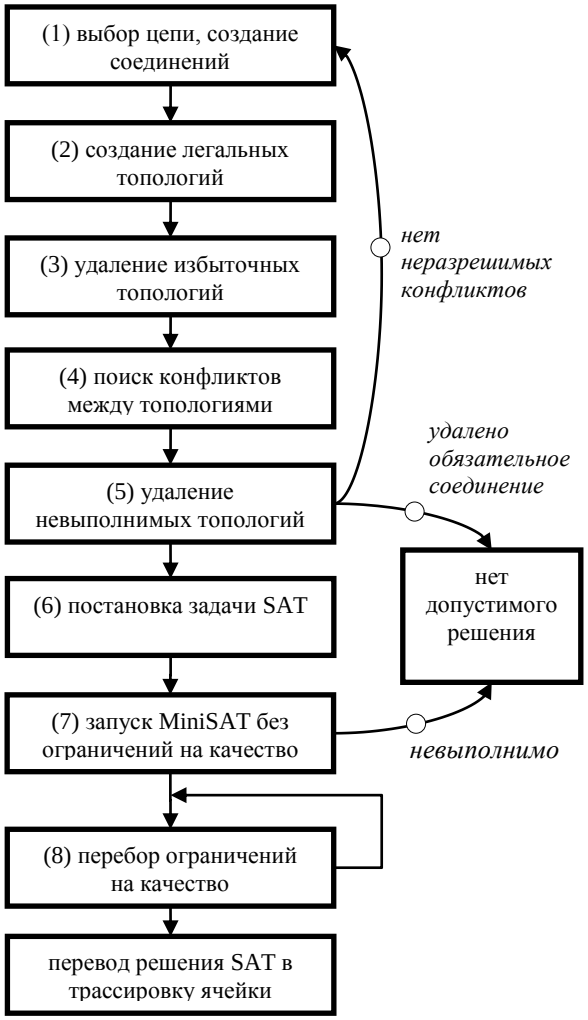

Рис. 2. Маршрут трассировки

На этапе 2 для каждого соединения создается несколько возможных топологических реализаций (далее сокр. – топологий). Топология обеспечивает электрическое соединение между *соответствующей парой терминалов* и состоит из отрезков проводников и межслойных переходов. Начальное число создаваемых топологий на одно соединение ограничено 1-3 тысячами. Избыточные топологии удаляются из рассмотрения на этапе 3.

На этапе 4 специальная процедура определяет конфликты между каждой парой топологий всех соединений. Процедура учитывает топологические правила, электрические правила и дополнительные ограничения на архитектуру стандартных ячеек.

Топологии, которые не могут быть реализованы в стандартной ячейке из-за конфликтов с другими топологиями, удаляются из рассмотрения на этапе 5.

На этапе 6 мы ставим задачу SAT. Постановка включает в себя условия полноты трассировки, конфликты между топологиями и дополнительные ограничения на качество трассировки.

На этапе 7 мы запускаем программу *(SAT solver)* для поставленной задачи в первый раз. Ограничения на качество трассировки на данном этапе выключены. После работы программы мы или имеем некое допустимое решение задачи, или же находим, что легальное решение не существует.

На этапе 8 программа выполняется несколько раз для списка ограничений на качество трассировки. Если решение не существует для данного набора ограничений, мы ослабляем ограничения и запускаем программу вновь. По окончанию этапа 8 мы или имеем начальное решение (после этапа 7), или же другое оптимизированное решение.

IV. Этапы до постановки задачи SAT

A. *Создание соединений*

Рис. 3 иллюстрирует задачу трассировки для цепи «n0», состоящей из 4 терминалов. В цепи – 2 поликремниевых затвора и 2 контакта, подключенных к стокам/истокам транзисторов. Всего 6 возможных соединений среди этих терминалов. Для получения полной трассировки 3 из этих 6 соединений должны быть реализованы в виде топологий; эти 3 соединения не должны образовывать цикл.

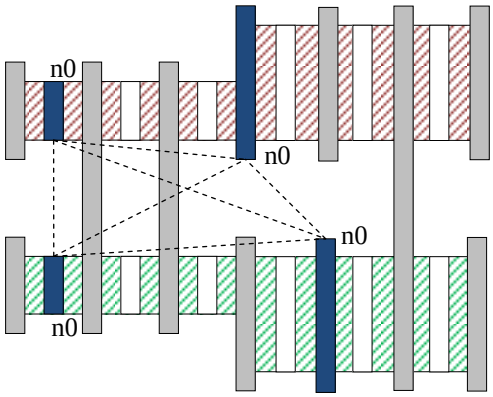

Рис. 3. Возможные соединения цепи «n0»

B. *Создание возможных топологий*

На этапе 2 для каждого соединения создаётся несколько различных возможных топологий. Мы используем волновой алгоритм для поиска топологий внутри ограничивающего прямоугольника, построенного вокруг соответствующей пары терминалов. Топологии строятся независимо друг от друга. Число проводников на топологию варьируется и ограничено небольшим числом (6-8 на соединение). Также варьируется точка подключения к терминалу.

Волновой алгоритм учитывает существующие препятствия (затворы транзисторов, контакты диффузии, полосы земли и питания), поэтому построенные топологии изначально не имеют электрических замыканий.

Топологии для портов, состоящих из одного терминала, создаются отдельно (рис. 4).

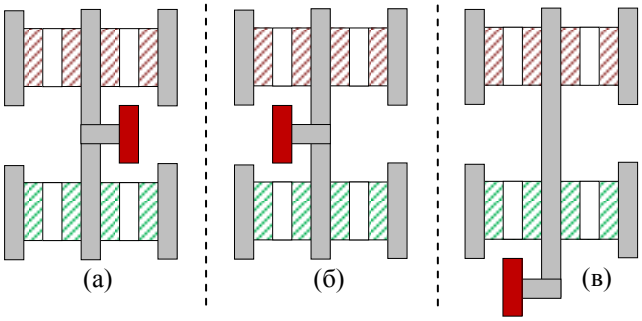

Рис. 4. Возможные топологии для терминала-порта

C. Удаление избыточных топологий

На этапе 3, мы удаляем из рассмотрения избыточные топологии. Мы удаляем топологию, соединяющую терминалы A и B, если данная топология может быть заменена парой топологий, соединяющих терминалы A и B через терминал C. Геометрически, топологии AC и CB после объединения должны в точности реализовывать топологию AB.

D. Поиск конфликтов между топологиями

На этапе 4, мы определяем конфликты между каждой парой топологий. Две топологии имеют *конфликт*, если данные топологии не могут быть одновременно реализованы в стандартной ячейке. Парные конфликты используются на этапе 5 для удаления из рассмотрения невыполнимых топологий. Электрические замыкания составляют большинство конфликтов. Другие конфликты определяются технологическими правилами, электрическими правилами и дополнительными ограничениями на архитектуру стандартной ячейки. Топологии одного соединения не могут иметь конфликт по определению, поскольку любое соединение может иметь только одну топологию, реализованную в стандартной ячейке. Топологии разных соединений, принадлежащих одной электрической цепи, не могут иметь электрического замыкания, но могут иметь конфликты других типов. Сложные правила могут одновременно вовлекать 3 и более топологий. Такие конфликты добавляются в формулу SAT отдельно, что будет показано в главе 5.

Для определения парных конфликтов, в худшем случае, требуется $R * W^2 * (T^2 - T)/2$ проверок, где T– число топологий, R – число правил, W – среднее число проводников и межслойных переходов на одну топологию. Для уменьшения вычислительной сложности задачи, мы определяем список уникальных проводников и переходов. Чем выше регулярность и дискретность технологического процесса [12], тем меньше данный список. Одним вызовом процедуры проверки для проводников W_1 и W_2 мы определяем $P * Q$ конфликтов, где P – число топологий, содержащих W_1, и Q – число топологий, содержащих W_2. Мы используем специальную структуру хранения данных в виде прямоугольной сетки для уменьшения числа избыточных вычислений. Проводники из различных регионов сетки проверяются на возможное нарушение правил в том и только в том случае, если эти регионы достаточно близки, чтобы иметь возможность нарушить данное технологическое правило между проводниками из данных регионов.

Дополнительные ограничения возникают на границах стандартных ячеек. Корректная топология потенциально может нарушить правила с другой топологией из соседней стандартной ячейки. Пары топологий внутри одной стандартной ячейки, ведущие к запрещённым ситуациям на границе ячейки, также отмечаются как конфликтующие.

Высокое электрическое сопротивление слоя поликремния накладывает дополнительные электрические ограничения. Так, например, в нашем маршруте проектирования стандартных ячеек запрещена ситуация, когда электрический сигнал распространяется по участку провода поликремния. Топологии p и q на рис. 5 создают такую ситуацию, и должны быть помечены как конфликтные топологии.

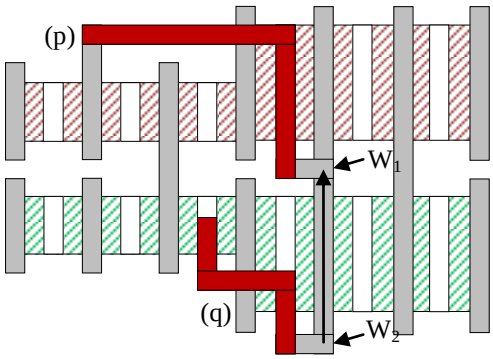

Рис. 5. Пример запрещённой электрической ситуации

Для поиска данной запрещённой ситуации не надо перебирать все возможные пары топологий, а можно воспользоваться упомянутой выше структурой данных. Для каждого провода поликремния мы определяем пару несовпадающих контактов W_1 и W_2 и помечаем соответствующие P и Q топологии как конфликтующие. Таким образом, мы сразу определяем $P * Q$ конфликтов. Все геометрические правила проверяются аналогично. Топологии могут иметь сколь угодно сложные и протяжённые формы, но правила проверяются для элементарных групп проводников, отстоящих друг от друга на относительно небольшом расстоянии.

E. Удаление недостижимых топологий

На этапе 5 мы удаляем из рассмотрения топологии, которые не могут быть реализованы в стандартной ячейке из-за конфликтов с другими топологиями. Так, мы удаляем из рассмотрения топологию, если она имеет конфликты со всеми топологиями некого обязательного соединения. В другом случае мы удаляем топологию, если она имеет конфликты со всеми топологиями, исходящими из одного терминала.

Мы удаляем из рассмотрения соединение, если оно теряет все свои возможные топологии. Если удаляются соединения, соединяющие некий терминал с другими терминалами, то последнее оставшееся соединение объявляется обязательным. Также, если две топологии одновременно блокируют некое обязательное соединение, то данные топологии помечаются как конфликтующие. Данные процедуры повторяются несколько раз, пока мы не определим все дополнительные конфликты между проводниками и все обязательные соединения, и пока мы не удалим все недостижимые топологии.

V. Постановка задачи SAT

Для произвольной булевой формулы, состоящей из ограниченного числа переменных, задача SAT состоит в том, чтобы определить: существует ли такой набор значений переменных, делающих эту формулу верной. Для решения задачи мы используем программу MiniSAT [13], которая, как и большинство аналогичных программ, требует запись булевой формулы в конъюктивной нормальной форме (КНФ). Мы используем общепринятые формулы перевода булевых выражений в КНФ [14].

Соединения и топологии. Булевый литерал r_i создаётся для каждой топологии. Когда r_i принимает значение 1, это значит, что топология реализуется в стандартной ячейке. Другой литерал создаётся на каждое соединение (1). t_k^n может принимать значение 1 (соединение есть) или 0 (соединения нет). Соединение может иметь только одну реализованную топологию, так что формула (2) устанавливается в *всегда-0*. В формуле (2) $\{r_i\}$ все R топологий соединения t_k^n.

$$\begin{cases} t_k^n = \bigvee_{i=1}^{R} r_i & (1) \\ \bigvee_{i=1;\ j=i+1}^{i\le R;\ j\le R} (r_i \wedge r_j) = 0 & (2) \end{cases}$$

Связность. Для N терминалов мы строим матрицу M размером $N \times N$. Изначально, все элементы M установлены в ноль. Если два терминала i и j имеют соединение t_{ij} с ненулевым числом топологий, то $M(i,j) = M(j,i) = t_{ij}$. Диагональные элементы записываются в следующем виде $M(i,i) = \bigvee_{k=1}^{T_i} t_k$, где $\{t_k\}$ все T_i соединений терминала i. После инициализации элементы матрицы M трансформируется в булевы формулы алгоритмом Флойда-Уоршала [15][16] с временной сложностью $O(N^3)$. По окончанию работы алгоритма, каждый элемент матрицы становится объединением всех возможных путей между соответствующей парой терминалов. Элементы матрицы устанавливаются в *всегда-1*. Это значит, что, по крайней мере, один путь между каждой парой терминалов должен быть реализован в решении задачи SAT. Данная постановка задачи не ограничивает число реализованных соединений. Однако их должно быть в точности $N-1$, чтобы получить ациклическое дерево для N терминалов. Используя унарный счётчик [17], мы накладываем ограничения на число реализованных соединений. Унарный счётчик описан в главе 6.

Конфликты. Парные конфликты между топологиями представляются той же формулой (2), что и для топологий одного соединения.

Сложные правила могут вовлекать объекты из трёх и более топологий. Если v – такой объект, мы создаём дополнительный литерал: $v = \bigvee_{i=1}^{R} r_i$, где $\{r_i\}$ все R топологий, содержащих v. Затем мы создаём соответствующие выражения и устанавливаем их в всегда-0: $\bigwedge v_i = 0$, чтобы избежать подобные нарушения.

Порты должны иметь как минимум один провод металл-1. В формулу SAT добавляется дополнительное ограничение $\bigvee_{k=1}^{R_n} r_k^{n/m1} = 1$, где $\{r_k^{n/m1}\}$ все R_n топологий портовой цепи n с одним и более проводом металла-1.

VI. Контроль качества трассировки

Использование унарного счётчика [17] позволяет контролировать и ограничивать число нежелательных топологий. Полный унарный счётчик для 4 булевых литералов представлен в виде логической схемы на рис. 6. Выходы c_1, c_2, c_3 и c_4 принимают значение 1, когда число значений ‘1’ среди $\{x_1, x_2, x_3, x_4\} \ge 1, \ge 2, \ge 3$ и $= 4$. Используя инвертированные значения входов, мы получаем инвертированный унарный счётчик. В таком случае выходы c_1, c_2, c_3 и c_4 принимают значение 1, когда число значений ‘1’ среди $\{x_1, x_2, x_3, x_4\}$ ≤ 3, ≤ 2, ≤ 1 и $= 0$. Логика, представленная на рис. 6, может быть расширена на произвольное число входов и выходов. Также мы можем построить частичный унарный счётчик, если нам нужен ограниченный набор выходов (например, только c_2). В таком случае ненужные узлы удаляются (объекты с пунктирными контурами на рис. 6), новые узлы не добавляются.

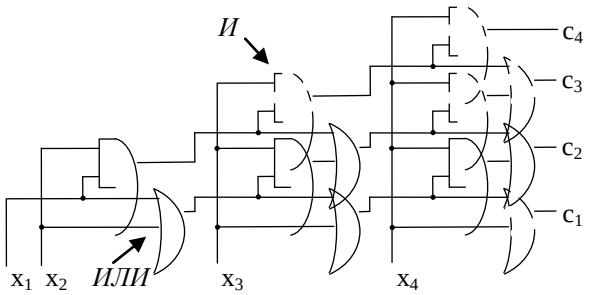

Рис. 6. Унарный счётчик для 4 литералов

Использование унарных счётчиков позволяет минимизировать число проводов на слое металл-2, число переходов между слоями, общую длину проводников и другие важные параметры трассировки.

Так, для всех переходов $\{v_k\}$ между слоями металл-1 и металл-2 создаётся отдельный литерал: $v_k = \bigvee_{i=1}^{R_k} r_i$, где $\{r_i\}$ все R_k топологий, содержащих межслойный переход v_k. Для литералов $\{v_k\}$ строится частичный инвертированный унарный счётчик. Программа MiniSAT имеет возможность поиска решения SAT с дополнительными булевыми допущениями, и эта программная возможность используется на этапе 8. Поиск решения SAT выполняется поочерёдно с допущениями, что число межслойных переходов $= 0$, ≤ 2 , ≤ 4 и ≤ 6 . Соответствующие Булевы формулы («выходы» счётчика) добавляются в общую КНФ и поочерёдно устанавливаются в *всегда-1*.

Остальные параметры качества трассировки оптимизируются в аналогичной манере. Формируются группы нежелательных топологий, строится соответствующий унарный счётчик, и программа MiniSAT поочерёдно использует в виде внешнего допущения *всегда-1* каждый «выход» счётчика, начиная с наиболее жёсткого ограничения.

VII. Экспериментальные результаты

Для проверки эффективности предложенного маршрута трассировки, мы перетрассировали стандартные ячейки из промышленной библиотеки компании Intel, созданной инженерами вручную по проектируемой нанометровой технологии. Библиотека содержит 188 различных семейств элементов. Мы выбрали 112 однорядных элементов и 157 двухрядных элементов, всего 269 различных элементов из всех 188 семейств. 78 стандартных ячеек (29%) имеют провода на металле-2. Мы экстрагировали размещение транзисторов и трассировали каждый элемент. 89% элементов (239 из 269) были успешно трассированы. Провода на металле-2 были успешно удалены в 59% случаев (табл. 1). После трассировки была проведена проверка на соответствие технологическим правилам, ячейки были признаны годными для последующих этапов проектирования стандартных ячеек.

Таблица 1

Результаты трассировки 269 элементов

тип	элементы	успешн.	усп. %	удал. м2
комбинац.	120	115	96%	6 / 11
э-ты задержки	11	11	100%	0 / 0
защёлка	14	14	100%	1 / 1
триггер	14	13	93%	9 / 9
скан. триггер	29	21	72%	16 / 24
сумматор	16	9	56%	5 / 13
мультиплексор	25	21	84%	7 / 13
ген-тор синх.	40	35	88%	2 / 7
Всего	**269**	**239**	**89%**	**46 / 78**

Временные характеристики представлены в табл. 2. В качестве примера, триггер имеет следующие характерные числа: портов – 3, сигнальных цепей – 11, всего возможных соединений – 87, из них удалено из рассмотрения – 25. Для 62 соединений всего создано 29762 корректных топологий, из них удалено на основе информации о парных конфликтах – 21107 (70%). Программа MisiSAT рапортовала число созданных термов: ~270 тысяч, дизъюнкций: ~9 миллионов. Общее время трассировки – 340 секунд.

Таблица 2

Время трассировки

	< 1 мин.	< 5 мин.	< 1 час	> 1 час
число элементов	169	20	37	14
% от успешн.	~70%	~8%	~15%	~6%

Зависимость числа успешно трассированных элементов от числа транзисторов представлена в табл. 3. Наибольший успешно трассированный элемент в данном эксперименте был комбинационный элемент, состоящий из 63 транзисторов. Трассировка для 16 элементов была принудительно оборвана из-за превышения лимита на используемую память. Отметим, что в 7 из 16 случаев первоначальное решение было уже получено. Программа превысила память, оптимизируя начальную трассировку.

Таблица 3

Статистика успешных трассировок

транз-ры	2-10	11-20	21-30	31-40	41-50	51-70	всего
эл-ты	99	81	27	31	23	8	269
успешн.	94	78	26	22	15	4	239
%	95%	96%	96%	71%	65%	50%	89%

Литература

[1] Sanjay Rekhi, J. Donald Trotter, Daniel H. Linder. Automatic layout synthesis of leaf cells // DAC 1995.

[2] Gupta A., J. P. Hayes. CLIP: Integer-Programming-Based Optimal Layout Synthesis of 2D CMOS Cells // ACM Tr. on DA El. Sys. July 2000. V. 5. № 3.

[3] Mohan G. et al. CELLERITY: a fully automatic layout synthesis system for standard cell libraries // DAC 1997.

[4] Lin Y.-W., Marek-Sadowska M., Maly W. Transistor-level layout of high-density regular circuits // ISPD 2009.

[5] Li M.-C. et al. Standard cell like via-configurable logic block for structured ASICs // Proc. of IEEE CSA Symp. on VLSI. 2008.

[6] M. Pons, F. Moll, A. Rubio, J. Abella, X. Vera, and A. González. VCTA: a via-configurable transistor array regular fabric // Proc. of VLSISoC. 2010.

[7] N. Ryzhenko, S. Burns. Physical Synthesis onto a Layout Fabric with Regular Diffusion and Polysilicon Geometries // DAC 2011.

[8] T. Iizuka, M. Ikeda, K. Asada. High Speed Layout Synthesis for Minimum-Width CMOS Logic Cells via Boolean Satisfiability // ASP-DAC. 2004.

[9] Nam G.-J. Satisfiability-Based Layout Revisited: Detailed Routing of Complex FPGAs via Search-Based Boolean SAT // FPGA. 1999.

[10] B. Taylor, L. Pileggi. Exact Combinatorial Optimization Methods for Physical Design of Regular Logic Bricks // DAC 2007.

[11] F. Yang, Y. Cai, Q. Zhou, J. Hu. SAT Based Multi-Net Rip-up-and-Reroute for Manufacturing Hotspot Removal // DATE 2010.

[12] T. Jhaveri et al. Co-Optimization of Circuits, Layout and Lithography for Predictive Technology Scaling Beyond Gratings // IEEE Trans. of ICs and S. Apr. 2010. Vol. 29. №4. pp. 509-527.

[13] http://minisat.se.

[14] Tseitin G.S. On the Complexity of Derivation in Propositional Calculus // In Studies in CMML. 1968. Part 2.

[15] Warshall S. A theorem on Boolean matrices // Journal of the ACM. Jan. 1962.

[16] Floyd W. R. Algorithm 97: Shortest Path // Journal of the ACM. June 1962.

[17] Claude E. Shannon, A symbolic analysis of relay and switching circuits. Thesis (M.S.), MIT, Dept. of Electrical Engineering. 1940.