УДК 519.688

Трассировка битовых элементов памяти с автоматическим построением ограничений на границах ячеек

Н.В. Рыженко, С.А. Быков, А.А. Сорокин

АО «Интел АО», nikolai.v.ryzhenko@intel.com

***Аннотация* — В массивах памяти ячейки упакованы в плотные структуры, каждая ячейка имеет своим соседом ячейку такого же типа. В данной работе представлен алгоритм трассировки битовых ячеек памяти, который трассирует одновременно нескольких идентичных ячеек, составляющих блок. Границы между ячейками в данном блоке моделируют все возможные варианты границ в массиве памяти произвольного размера. Электрические цепи ячеек трассируются независимо, но накладывается ограничение: топологии всех ячеек должны быть идентичны друг другу. Данный подход позволяет одновременно построить трассировку битовой ячейки и удовлетворить требованиям на топологию на её границах. Ключевыми компонентами алгоритма являются: представление топологических правил с помощью булевых выражений и использование задачи булевой выполнимости для поиска решения с заданными ограничениями. Экспериментальные результаты показали применимость предложенного подхода для синтеза промышленных ячеек памяти.**

***Ключевые слова* — ячейка памяти, трассировка, булева выполнимость.**

I. Введение

На рис. 1 представлена упрощенная структура памяти. Память состоит из двух основных частей. Битовые массивы хранят цифровую информацию. Периферийная логика управляет записью и чтением. Массивы памяти состоят из идентичных, плотно упакованных электронных элементов, битовых ячеек. Ячейки формируют регулярные повторяющиеся ряды и колонки. Одна размерность, например, число колонок массива определяет длину слова (степень числа 2), а другая размерность, в данном случае, число рядов битового массива определяет размер памяти. Ячейки памяти и управляющую логику соединяют металлические проводники, вертикальные и горизонтальные шины чтения, записи и управления. Битовые массивы и управляющая логика занимают большую часть площади. Эти области может разделять узкий слой специальных нефункциональных ячеек, которые используются для сопряжения топологий этих двух функциональных блоков.

Разработка топологии битовых ячеек традиционно выполняется вручную. Это является экономически и техническим обоснованным решением. Типов битовых ячеек, которые используются в СБИС, сравнительно немного, а качество каждой ячейки чрезвычайно важно для плотности транзисторов и размеров блоков памяти, поскольку один тип ячейки повторяется сотни тысяч и миллионы раз.

Современные технологии нанометровой оптической литографии бросают новые вызовы традиционным методам проектирования. Длина световой волны оптической литографии уже много лет остаётся неизменной. Такие сложные технологии, как двойное экспонирование [1, 2], позволяют изготавливать транзисторы и объекты топологии существенно меньше длины 193 нм оптического лазера. При этом неуклонно растёт число и сложность технологических правил *(design rules)*. Ручное проектирования оптимальных, максимально плотных ячеек памяти становится всё более и более затруднительной задачей.

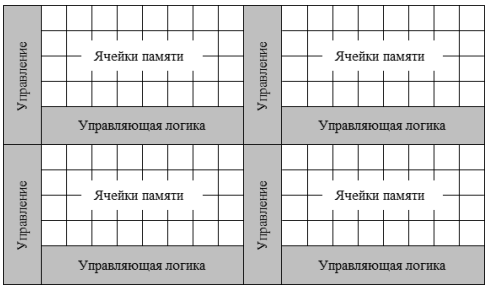


Рис. 1. Схематичная структура памяти

Целью работы является автоматизация процесса построения трассировки битовых ячеек памяти при автоматическом удовлетворении вспомогательных ограничений на границах ячейки. Для этого предлагается использовать задачу Булевой выполнимости для одновременной трассировки нескольких ячеек, поставленных в блок. Дополнительные специальные ограничения позволяют удовлетворить требованиям на топологию на границах ячейки.

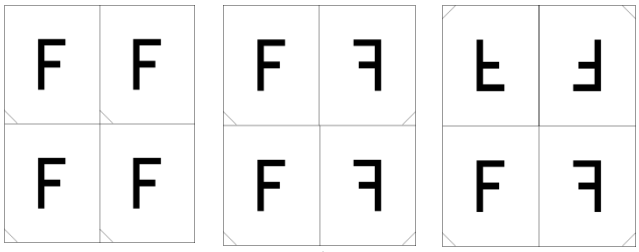
Содержание работы следующее. В главе 2 рассматриваются особенности проектирования битовых ячеек памяти. В главе 3 описаны SAT алгоритм трассировки стандартных ячеек и аспекты моделирования технологических правил для регулярных технологических процессов. В главе 4 представлен маршрут трассировки битовых ячеек. Экспериментальные результаты и заключение представлены в финальных главах 5 и 6.

II. Особенности проектирования битовых ячеек

Битовые ячейки принципиально отличаются от традиционных стандартных ячеек, использующихся для проектирования электронных устройств различного назначения. Библиотеки стандартных ячеек состоят из сотен и тысяч различных элементов. Каждый из этих элементов (при выполнении определённых правил размещения) должен быть легален любому другому элементу этой библиотеки. Никакое легальное взаимное расположение двух и более стандартных элементов библиотеки не должно нарушить ни одно из технологических правил. Данное основополагающее правило существенно ограничивает размещение транзисторов внутри стандартной ячейки и вид трассировочной топологии на её границах.

В случае битовой ячейки ситуация совершенно противоположная. Будущее окружение элемента памяти известно заранее – каждая битовая ячейка окружена со всех сторон точно такими же ячейками. Данный архитектурный подход используется специальными программами – компиляторами памяти – для синтеза блоков памяти произвольного размера. Ячейки ставятся встык, при этом автоматически формируется топология сигнальных шин, и выполняются все необходимые технологические правила. Знание окружения ячейки позволяет максимально плотно упаковывать транзисторы. Структуры памяти традиционно имеют максимальную плотность транзисторов среди прочих электронных компонент СБИС.

Здесь и далее в работе без ограничения общности полагается, что ячейки ставятся одна к другой без поворотов и отображений, как показано на рис. 2а (буква *F* используется для демонстрации типа размещения ячейки). В данном контексте это означает, что топология на правой границе ячейки должна быть легальна топологии на левой границе этой же ячейки. Аналогично, топология у верхней границы ячейки должны быть легальна топологии у нижней границы этой же ячейки.



а б в

Рис. 2. Примеры типового размещения битовых ячеек для различных архитектур памяти

На рис. 3 представлена схематичная частичная структура блока памяти 2x2, составленного из четырёх идентичных битовых ячеек. При разработке битовых ячеек учитывается их будущее размещение. Поэтому топология одной ячейки может попадать в ограничивающий прямоугольник топологии соседней ячейки, что недопустимо в обычных стандартных библиотеках.

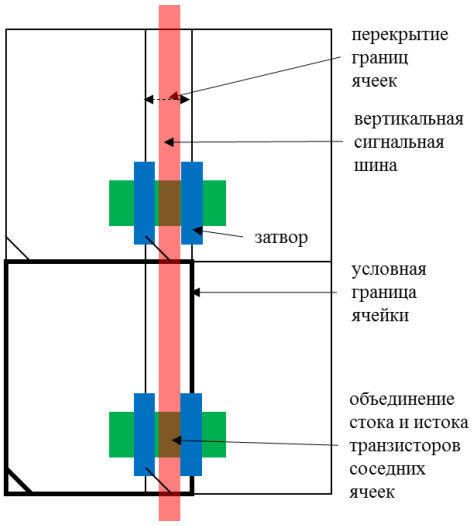


Рис. 3. Схематичная частичная структура блока 2x2 четырёх идентичных битовых ячеек

Также проектировщики стараются максимально объединить части топологии соседних ячеек, принадлежащие одной электрической цепи. Обычно это цепи земли и питания. В частности, могут объединяться эквипотенциальные стоки и истоки транзисторов. Объединение диффузионных областей транзисторов чрезвычайно важно для достижения максимально плотного их размещения. При этом электрическое соединение цепи получается автоматически без дополнительной трассировки по слоям металлизации.

Ещё одной отличительной особенностью битовых ячеек является отсутствие традиционных контактных площадок для входных и выходных сигналов и, соответственно, требований на минимальное количество пересечений с заданным слоем трассировки. Интерфейсы ячейки подключены к общим сигнальным шинам.

III. SAT трассировка

Исключительные технические сложности оптической литографии для нанометровых технологий приводят к тому, что топология становится всё более и более регулярной [3]. Регулярность умешает вычислительную сложность задачи оптической коррекции и позволяет более эффективно формулировать определённые технологические правила. Регулярная топология слоёв позволяет использовать для традиционных задач проектирования эффективные математические абстракции, такие как задача булевой выполнимости *(SAT)*. Достижения последних лет в области SAT решателей и регулярной топологии позволяют использовать SAT для оптимизированной и полной трассировки стандартных ячеек [4, 5].

Несомненным преимуществом любого трассировщика на основе SAT перед традиционными эвристическими подходами является то, что SAT поддерживает любые произвольные ограничения, которые могут быть выражены в виде Булевых выражений. В частности, данными ограничениями могут быть технологические правила. Эффективная модель технологических правил, представленных в виде булевых выражений для дискретных объектов топологии на двумерной решётке [6], позволяет транслировать произвольные правила в задачу трассировки без дополнительной настройки программы [5].

Другой особенностью любого трассировщика на основе задачи SAT является полнота. Он или находит решение (если оно существует), или же показывает, что решения при данных ограничениях не существует.

A. Алгоритм SAT трассировки стандартных ячеек

В работе [4] предложен способ трассировки стандартных ячеек с использованием задачи булевой выполнимости. На предварительном этапе все электрические цепи ячейки разбиваются на двухточечные соединения. Для каждого соединения создаётся список из нескольких возможных топологий. Специальная процедура определяет конфликты между каждой парой возможных топологий, используя как электрические правила, так и геометрические литографические правила для данной технологии. Связность трассировки, возможные реализации соединений, парные конфликты между ними и другие дополнительные ограничения переводятся в общую булеву формулу, представленную в конъюнктивной нормальной форме (КНФ). Для нахождения решения используется SAT решатель [7].

B. Моделирование технологических правил

В работе [5] алгоритм трассировки стандартных ячеек [4] расширен на более широкий класс решаемых задач за счёт использования модели представления технологических правил в виде булевых выражений для нелегальных вариантов топологии на двумерной решётке [6]. На рис. 4 представлено несколько нелегальных конфигураций проводника на регулярной трассировочной сетке S.

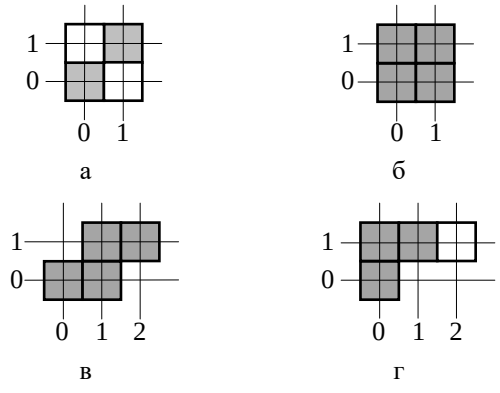


Рис. 4. Примеры запрещённых топологий

Выражение (1) определяет запрещённую топологию, в которой два проводника касаются друг друга в одной точке (рис. 4а). Выражения (2) и (3) определяют нелегальные топологии с рис. 4б и 4в. Выражение (4) определяет топологию, в которой горизонтальный отрезок проводника имеет недостаточную длину в точке пересечения с вертикальным проводником (рис. 4г).

$$S(0,0) \wedge S(1,1) \wedge \overline{S(0,1)} \wedge \overline{S(1,0)} \quad (1)$$

$$S(0,0) \wedge S(1,0) \wedge S(0,1) \wedge S(1,1) \quad (2)$$

$$S(0,0) \wedge S(1,0) \wedge S(1,1) \wedge S(2,1) \quad (3)$$

$$\mathrm{S}(0,0) \wedge \mathrm{S}(0,1) \wedge \mathrm{S}(1,1) \wedge \overline{S(2,1)} \quad (4)$$

В работах [5, 6] показано, что всё многообразие технологических правил можно свести к булевым выражениям для дискретных объектов топологии. Число правил и размер булевых функций определяются степенью дискретности технологического процесса. Поскольку трассировщик используется для стандартных ячеек, то отдельный класс геометрических правил описывает ограничения на топологию на границах ячейки.

Для перехода от выражений, таких как (1)-(4), в постановке задачи трассировки каждому дискретному отрезку проводника ставится в соответствие дизъюнкция булевых переменных $S(x,y) = \vee\{v\}$. Число переменных $\{\mathrm{v}\}$ равно числу электрических цепей $\{n\}$ трассируемой ячейки. Если после решения задачи SAT все переменные $\{v\}$ принимают значение 0, то значит, что данный отрезок проводника $S(x,y)$ отсутствует в трассировке ячейки. Если переменная v_i принимает значение *1* «истина», то значит, что данный отрезок проводника присутствует в трассировке ячейки и принадлежит электрической цепи n_i. Поскольку проводник не может принадлежать двум цепям одновременно, то в КНФ задачи SAT вводится дополнительное ограничение, и все остальные переменные $\{v_{j\neq i}\}$ после решения задачи SAT принимают значение *0* «ложь», если $v_i = 1$.

C. Оптимизация трассировки

Особенность SAT решателей такова, что они определяют первое допустимое решение. Булевы счётчики (специальный класс функций) позволяют оптимизировать трассировку, минимизируя число нежелательных топологий [5]. Нежелательные топологии представлены теми же булевыми выражениями, что и обычные технологические правила.

IV. Алгоритм трассировки битовых ячеек

В данной работе предлагается использовать подходы [4-6] для трассировки битовых ячеек памяти, введя определённые дополнительные ограничения, которые будут представлены ниже в этом разделе.

Маршрут проектирования битовой ячейки представлен на рис. 5.

1. Размещение транзисторов ячейки в заданный ограничивающий прямоугольник

2. Определение необходимого минимального числа ячеек для моделирования всех возможных границ

3. Размещение заданного числа ячеек в блок; модификация списка соединений ячеек согласно архитектуре сигнальных шин

4. Постановка задачи трассировки, введение дополнительных ограничений на идентичность топологии

5. Решение задачи булевой выполнимости, экстракция топологии отдельной ячейки

Рис. 5. Маршрут трассировки битовой ячейки

A. *Размещение транзисторов ячейки*

Для размещения транзисторов можно воспользоваться динамическим алгоритмом [8]. Также практический опыт показывает, что для компаний с долгой историей созданное много лет назад удачное расположение транзисторов мигрирует из технологии в технологию. Для построения удовлетворительного начального размещения достаточно сделать правильное масштабирование и поставить транзисторы в легальную позицию. Несложные программы позволяют создать достаточное количество перспективных вариантов ячейки за счёт инкрементальных модификаций исходного размещения.

B. *Определение минимального блока ячеек*

Для автоматического моделирования корректной топологии на границах ячейки предлагается трассировать одновременно несколько ячеек, составленных в небольшой блок, миниатюрный битовый массив. Блок должен быть такого размера, чтобы реализовать все возможные граничные условия ячейки в реальном битовом массиве.

В общем случае блок ячеек 3x3 моделирует окружение для всех четырех сторон и всех четырёх углов центральной ячейки. В частных случаях можно ограничиться блоком меньшего размера.

Для архитектуры, изображенной на рис. 3а, минимальный блок ячеек будет 2x2 (рис. 6). Поскольку ячейки одинаковые и отсутствуют повороты и отображения, то в блоке 2x2 моделируется окружение всех четырёх сторон S, N, W, E (отмечены толстыми линиями) и всех четырёх углов SW, SE, NW, NE (отмечены треугольниками). Для архитектуры, изображенной на рис. 3б, минимальный блок ячеек будет 3x2, т.к. для моделирования западной стороны и соответствующих углов требуется дополнительный столбец ячеек (рис. 7).

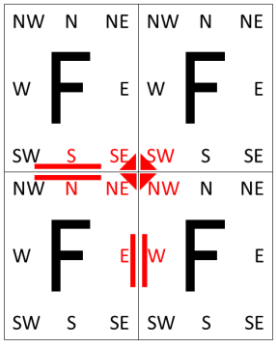


Рис. 6. Минимальный блок ячеек для архитектуры с рис. 3а

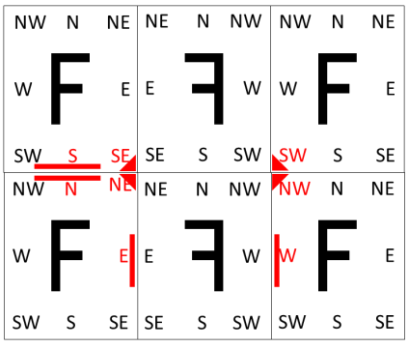


Рис. 7. Минимальный блок ячеек для архитектуры с рис. 3б

В общем случае – чем меньше блок, тем быстрее трассировка.

C. *Размещение ячеек в блок и формирование нового списка электрических соединений*

Особенностью данного этапа является переход от иерархии нескольких ячеек к одной плоской супер-ячейке. Пример на рис. 8 иллюстрирует исходную задачу. Контакт a подключается к вертикальной сигнальной шине, цепи b и c – внутренние цепи ячейки. В реальности и шин, и внутренних цепей гораздо больше. Показана регулярная трассировочная сетка для демонстрации работы алгоритма.

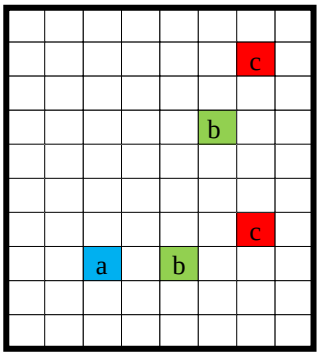


Рис. 8. Исходная битовая ячейка

Для перехода к супер-ячейке транзисторное размещение исходной ячейки мультиплицируется необходимое количество раз. Соответствующим образом изменяется список электрических соединений (рис. 9). Внутренние цепи b и c получают новую индексацию, число терминалов каждой цепи остаётся без изменений. Вертикальная сигнальная шина a трансформируется не в четыре, а в две цепи a_1 и a_2 по числу столбцов. Трассировочное пространство вокруг супер-ячейки дополнительно не ограничивается.

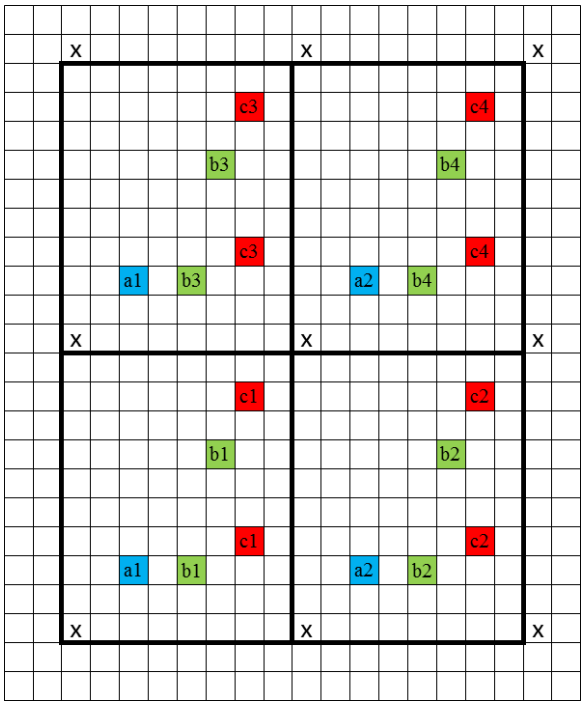


Рис. 9. Супер-ячейка блока 2x2

D. *Постановка задачи трассировки, введение дополнительных ограничений на идентичность топологии*

Без дополнительных ограничений трассировка супер-ячейки как обычной большой стандартной ячейки не решает задачу корректного моделирования граничных условий одиночной битовой ячейки. Поэтому в КНФ задачи SAT вводятся булевы выражения для достижения идентичности топологии в различных частях супер-ячейки.

На рис. 9 знаком × отмечены 9 квадратов трассировочной сетки. Каждому квадрату в соответствие ставится дополнительная переменная x_i. Переменные $\{x\}$ являются связанными. Если переменная x_i принимает значение 1, т.е. соответствующий квадрат сетки используется некой цепью для трассировки, то и остальные переменные $\{x_{j\neq i}\}$ также принимают значение 1, т.е. тоже используются этой или какой-то другой цепью для трассировки. Если же переменная x_i принимает значение 0, то и остальные $\{x_{j\neq i}\}$ также принимают значение 0, т.е. все квадраты X не содержат трассировки. Данное ограничение накладывается на все квадраты трассировочной сетки и добавляется в общую КНФ задачи.

E. *Трассировка супер-ячейки и экстракция топологии отдельной битовой ячейки*

Рис. 10 иллюстрирует возможную трассировку супер-ячейки. Трассировка построена таким образом, что ни одно их выражений (1)-(4) ни в одной из точек трассировочной сетки не принимает значений «истина». Таким образом, ни одна из запрещённых конфигураций топологии с рис. 4 не присутствует в данной трассировке.

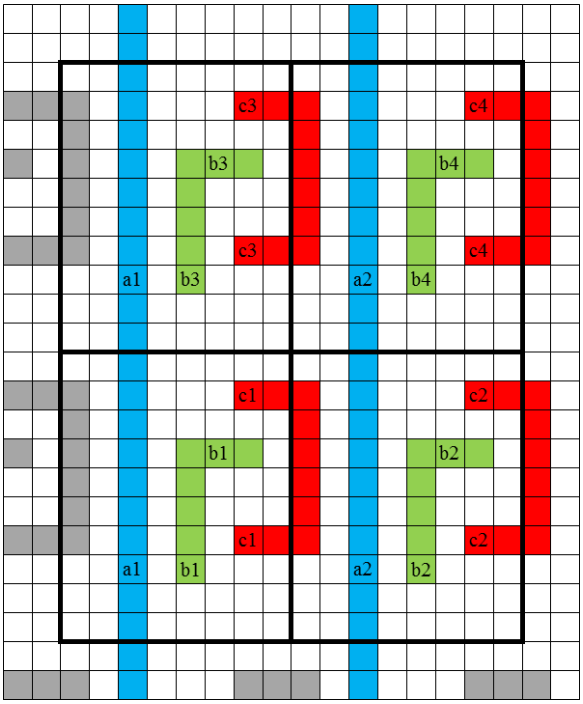


Рис. 10. Финальная трассировка супер-ячейки

Для экстракции топологии отдельной битовой ячейки достаточно взять трассировку цепей с индексом 1. При этом топология внутренних цепей (b_1 и c_1) сохранятся без изменений, а топология шин (цепь a_1 в нашем примере) обрезается по соответствующим границам ячейки (рис. 11).

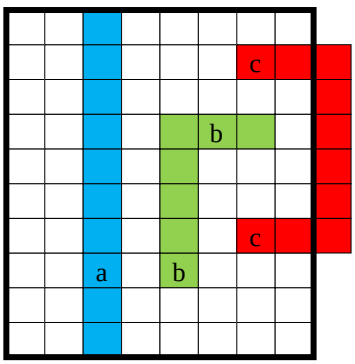


Рис. 11. Финальная трассировка битовой ячейки

Рис. 12 иллюстрирует битовый массив 4x3, составленный из 12 идентичных битовых ячеек. Топология данного массива корректна по построению. Сигнальные шины соединяются встык. Трассировка не нарушает технологических правил.

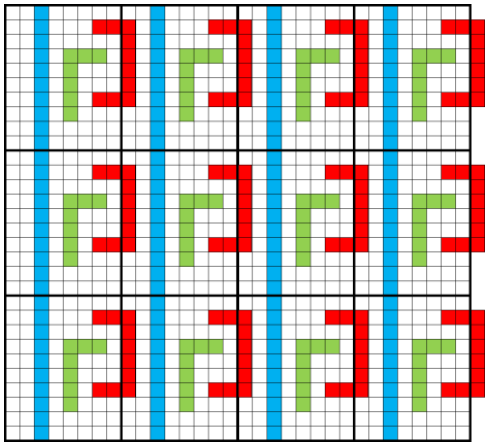


Рис. 12. Битовый массив 4x3

Мы видим, что топология одной ячейки может попадать в условную границу другой, если это необходимо и не нарушает правил проектирования. На практике для сопряжения битовых массивов и управляющей логики используют относительно узкие нефункциональные ячейки-разделители.

V. Экспериментальные результаты

Подход, описанный в данной работе, был реализован в качестве программного продукта на языке C++ и использовался в компании Intel для трассировки битовых элементов троичной ассоциативной памяти TCAM [9]. Данный тип памяти находит приложение в телекоммуникационных электронных устройствах. Специфика коммерческой организации не позволяет раскрыть в рамках данной работы информацию о схемотехнике, топологии и даже числе транзисторов и технологическом процессе. Но можно отметить, что программный комплекс позволил провести полноценные архитектурные исследования. Абсолютно легальная трассировка была построена для нескольких вариантов размещения транзисторов битовой ячейки в заданной площади. Из них был выбран наилучший. При ручном проектировании число рассматриваемых вариантов значительно меньше.

VI. Заключение

В данной работе представлен алгоритм трассировки битовых ячеек памяти, который трассирует одновременно нескольких идентичных ячеек, составляющих блок. Границы между ячейками в данном блоке моделируют все возможные варианты границ в массиве памяти произвольного размера. Особенностью подхода является переход от иерархии нескольких ячеек к одной плоской супер-ячейке. При этом соответствующим образом изменяется список электрических соединений. На дискретные объекты топологии накладываются дополнительные ограничения на идентичность. Данные ограничения формулируются в виде булевых выражений и добавляются в общую КНФ задачи SAT. Затем электрические цепи супер-ячейки трассируются независимо методами трассировки стандартных ячеек. Данный подход позволяет одновременно построить трассировку битовой ячейки и удовлетворить требованиям на топологию на её границах. Ключевыми компонентами алгоритма являются: представление топологических правил с помощью булевых выражений и использование задачи булевой выполнимости для поиска решения с заданными ограничениями.

Экспериментальные результаты показали применимость предложенного подхода для синтеза промышленных ячеек памяти.

Литература

[1] Y. Du, Q. Ma, H. Song, J. Shiely, G. Luk-Pat, A. Miloslavsky, and M. D. F. Wong. Spacer-is-dielectric-compliant detailed routing for selfaligned double patterning lithography. InProceedings of the 50th Annual Design Automation Conference, pages 93:1–93:6, 2013.

[2] Z. Xiao, Y. Du, H. Zhang, and M. Wong. A polynomial time exact algorithm for overlay-resistant self-aligned double patterning (sadp) layout decomposition. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 32(8):1228–1239, 2013.

[3] Kurt Ronse, Philippe Jansen, R. Gronheid, Eric Hendrickx, M. Maenhoudt, Vincent Wiaux, Anne-Marie Goethals, R. Jonckheere, and G. Vandenberghe. Lithography Options for the 32 nm Half Pitch Node and Beyond // IEEE Tran. on Circuits and Systems - I: Regular papers, Vol. 56, No. 8, August 2009.

[4] Ryzhenko N., Burns S. Standard cell routing via boolean satisfiability // Proceedings of the 49th Annual Design Automation Conference. – ACM, 2012. – С. 603-612.

[5] Рыженко Н.В., Сорокин А.А., Быков С.А., Талалай М.С. Минимизация числа нежелательных топологий при проектировании стандартных ячеек // Проблемы разработки перспективных микро- и наноэлектронных систем - 2014. Сборник трудов / под общ. ред. академика РАН А.Л. Стемпковского. М.: ИППМ РАН, 2014. Часть I. С. 121-126.

[6] Suto G. Rule agnostic routing by using design fabrics // Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE. – IEEE, 2012. – С. 471-475.

[7] URL: http://minisat.se (дата обращения: 1.03.2016).

[8] Сорокин А.А., Рыженко Н.В., Алгоритм размещения транзисторов в стандартных ячейках // Проблемы разработки перспективных микро- и наноэлектронных систем - 2014. Сборник трудов / под общ. ред. академика РАН А.Л. Стемпковского. М.: ИППМ РАН, 2014. Часть I. С. 133-136.

[9] URL: https://en.wikipedia.org/wiki/Content-addressable_memory (дата обращения: 1.03.2016).

Routing of Memory Bits Cells with Automated Construction of Boundary Layout Constraints

N.V. Ryzhenko, S.A. Bykov, A.A. Sorokin

AO “Intel A/O”, nikolai.v.ryzhenko@intel.com

Keywords — routing, memory, standard cells, design rules, Boolean satisfiability.

ABSTRACT

In this paper we present a solution for the automatic routing of memory bit array cells. Memory arrays are constructed of a specially designed memory bit cells. Memory cell design opposes to the standard cell methodology where each cell must be legal to any other cell from the library for all possible allowed mutual placements. Bit cells can share layout, for example adjacent cells can share diffusions. Usually it’s power and ground nets. Diffusion sharing of adjacent bit cells allows to achieve ultimate transistor density. And memory cells don’t have pins in traditional meaning. All nets are either power, ground, internal or connected to buses. Straight buses come across the whole memory array.

There are two technology trends in the industry. While the optical lithography wave length 193nm remains unchanged, feature size continues to scale down according to the Moor’s low. The complexity and the number of design rules constantly grow [1]-[2]. Another trend is that layout becomes more and more regular [3]. Limited number of patterns in regular layouts reduces the complexity of the optical proximity correction. Also regular layouts make easier formulation of particular design rules. Regular layout allows to use efficient data models [6]. While layouts of objects are complete, discrete and gridded, the data model codes all possible design rules with uncompromised accuracy. Regular layouts and recent dramatic improvements in the solvers make possible the use of Boolean Satisfiability (SAT) for standard cell routing [4]-[5]. Key feature of SAT routing is completeness. SAT solver either finds a legal DR-clean solution or reports that such solution doesn’t exist under constraints.

Design and development of memory bit cells is traditionally done manually because it is cost-effective. Any VLSI circuit utilizes very few types of memory, so very few types of memory bit cells are needed. At the same time there can be hundred thousand or millions of instances of identical bit cells in the design. That’s why every bit cell is designed to be super optimal in terms of transistor density, power and performance. Challenges of modern nanometer technologies make manual design of bit cells unpractical.

We propose to extend algorithms of standard cell routing [4]-[5] for memory bit cells. A proposal is to place bit cells in a small block. The variety of borders between adjacent cells in this block must model all possible borders between cells in a real-size memory array. One of cells in the block is marked as master, other cells are marked as children. We impose an additional constraint that if the master has a piece of layout at some point every child must have same piece of layout at the same point. If the master has an empty space at some point then every child must have same empty space at the same point. These extra constraints are translated into the common SAT formula and after the SAT routing master and children get identical layout and master cell is legal to all cells around. Another modification is that adjacent cells share busses as same nets. Basically the rest of the routing algorithm remains unchanged. A block of cells is routed as a big flat super-cell. Experimental results demonstrated applicability of the proposed approach for routing industrial types of bit cells [9]. The algorithm was used to route real bit cells.

REFERENCES

[1] Y. Du, Q. Ma, H. Song, J. Shiely, G. Luk-Pat, A. Miloslavsky, and M. D. F. Wong. Spacer-is-dielectric-compliant detailed routing for selfaligned double patterning lithography. InProceedings of the 50th Annual Design Automation Conference, pages 93:1–93:6, 2013.

[2] Z. Xiao, Y. Du, H. Zhang, and M. Wong. A polynomial time exact algorithm for overlay-resistant self-aligned double patterning (sadp) layout decomposition. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 32(8):1228–1239, 2013.

[3] Kurt Ronse, Philippe Jansen, R. Gronheid, Eric Hendrickx, M. Maenhoudt, Vincent Wiaux, Anne-Marie Goethals, R. Jonckheere, and G. Vandenberghe. Lithography Options for the 32 nm Half Pitch Node and Beyond // IEEE Tran. on Circuits and Systems - I: Regular papers, Vol. 56, No. 8, August 2009.

[4] Ryzhenko N., Burns S. Standard cell routing via boolean satisfiability // Proceedings of the 49th Annual Design Automation Conference. – ACM, 2012. – C. 603-612.

[5] Ryzhenko N.V., Sorokin A.A., Bykov S.A., Talalay M.S. Minimization of undesired layout patterns during standard cell synthesis // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2014. Proceedings / edited by A. Stempkovsky, Moscow, IPPM RAS, 2014. Part I. P. 121-126.

[6] Suto G. Rule agnostic routing by using design fabrics // Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE. – IEEE, 2012. – P. 471-475.

[7] URL: http://minisat.se (accessed: 1.03.2016).

[8] Sorokin A.A., Ryzhenko N.V. Transistor placement at standard cell level // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2014. Proceedings / edited by A. Stempkovsky, Moscow, IPPM RAS, 2014. Part I. P. 133-136.

[9] URL: https://en.wikipedia.org/wiki/Content-addressable_memory (accessed: 1.03.2016)