УДК 004.272.43:004.42

Параллельный алгоритм трансформации слоя топологии субмикронных СБИС с произвольной геометрией для технологии двойного шаблона

В.А. Шахнов, Л.А. Зинченко, В.А. Верстов

Московский государственный технический университет им. Н.Э. Баумана,

v.verstov@gmail.com

***Аннотация* — В работе предложены подход к трансформации топологии СБИС с неманхеттеновской топологией; параллельный алгоритм трансформации слоя топологии субмикронных СБИС для технологии двойного шаблона, ориентированный на применение высокопроизводительных вычислительных систем.**

***Ключевые слова* — СБИС, технология двойного шаблона, трансформация топологии, параллельный алгоритм.**

I. Введение

В настоящий момент литография является одним из основных технологических процессов в микроэлектронике. Постоянное уменьшение проектных норм приводит к все большему проявлению эффекта взаимной дифракции и, как следствие этого, значительным искажениям при воспроизводстве топологий субмикронных сверхбольших интегральных схем (СБИС).

В документах International Technology Roadmap for Semiconductors (ITRS) технология двойного шаблона выделяется как одно из наиболее перспективных направлений развития литографии [1].

Технология двойного шаблона заключается в разделении критических слоев топологии СБИС на два шаблона для последующего поочередного воспроизведения. Этот подход позволяет:

1) существенно улучшить разрешение проекционной литографии;

2) перейти к меньшим проектным нормам, используя существующее оборудование;

3) повысить процент выхода годных изделий при производстве.

Следует отметить, что в настоящее время для повышения радиационной стойкости используются принципиально новые топологии полевых транзисторов (О-транзисторы, А-транзисторы и т. п.) с произвольной, неманхеттеновской геометрией, что требует разработки специальных алгоритмов трансформации топологии этого класса СБИС для технологии двойного шаблона.

Различные подходы к хранению и представлению данных, описывающих топологию субмикронных СБИС и систем на кристалле (system-on-chip) рассмотрены в работе [2].

В работе [3] были предложены последовательные алгоритмы, позволяющие выполнить трансформацию топологии СБИС для технологии двойного шаблона. Для миграции топологии СБИС было предложено использовать граф ограничений и граф противоречий. Предложенные подходы были реализованы в программе TPL Converter [4]. Однако в настоящее время при проектировании субмикронных СБИС необходимо обрабатывать файлы описания топологии СБИС объемом в десятки гигабайт, что требует использования соответствующих вычислительных ресурсов.

В [5, 6] были предложены параллельные алгоритмы трансформации топологии СБИС для технологии двойного шаблона для манхэттеновской геометрии. В [5] также проанализированы метрики для оценки качества декомпозиции топологии. Предложенные алгоритмы [5] были реализованы в программе Parallel DPLayout Migrator.

В данной работе рассматриваются подходы для трансформации топологии СБИС с произвольной геометрией.

II. Структура данных для параллельной обработки топологии СБИС

В работе обсуждается специальная структура данных для представления топологии СБИС, в которой исходный топологический слой разделен на горизонтальные полосы, которые можно обрабатывать на отдельных процессорах (рис. 1):

$$\alpha =< \alpha_1, \alpha_2, \dots, \alpha_N >, \qquad (1)$$

где α_i – элемент кортежа, включающий в себя кортеж более низкого уровня иерархии, содержащий несколько произвольных отрезков $\{O_m, \dots, O_{m+p}\}$, входящих в i-ю горизонтальную полосу; N – число вычислительных узлов.

Для учета взаимного влияния близко расположенных полигонов разделение на горизонтальные полосы выполняется таким образом, чтобы обеспечить перекрытие разделяемых участков топологии СБИС (рис. 1).

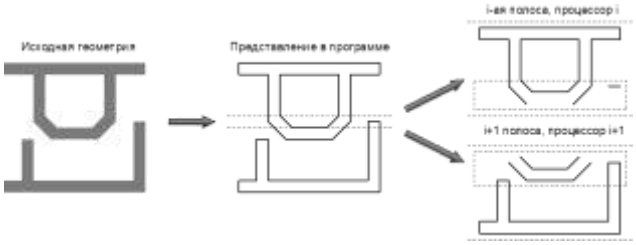


Рис. 1. Распределение задач между процессорами

На рис. 2 приведен алгоритм построения графа противоречий для исходного топологического слоя с учетом его разделения на *N* горизонтальных полос.

Рис. 2. Распределение задач между процессорами

Входными данными для алгоритма является множество полигонов *P*. Каждому полигону из *P* соответствует определенная вершина в графе противоречий *G*.

На первом шаге алгоритма формируется множество отрезков *O*, которые образуют полигоны из множества *P*. На следующем шаге алгоритма происходит разделение отрезков из множества *O* между кортежами α_i. В каждый кортеж α_i добавляются отрезки $\{O_m,\ldots,O_{m+p}\}$, которые попадают в *i*-ую горизонтальную полосу (рис. 1). Отрезок считается попавшим в *i*-ую полосу, если он ее пересекает или один из его концов лежит внутри полосы.

Граф противоречий *G* является общим объектом для всех потоков, которые выполняются на различных вычислительных узлах. В свою очередь кортеж α_i является объектом только *i*-ого потока, который выполняется на *i*-ом вычислительном узле.

На следующем шаге алгоритма выполняется поиск противоречий в кортежах α_i и добавление соответствующих ребер в *G* в параллельном режиме на *N* вычислительных узлах. Результатом работы алгоритма является сформированный граф противоречий *G*.

III. Алгоритм построения графа противоречий для участка топологии на основе алгоритма заметающей прямой

На предварительном этапе алгоритма построения графа противоречий для участка топологии (кортеж α_i) на основе алгоритма заметающей прямой заполняется очередь с приоритетом точек-событий *E*. В очередь *E* добавляются все точки – концы отрезков, которые попали в кортеж α_i.

Приоритет точек определяется согласно координатам по оси абсцисс: чем меньше координата, тем выше приоритет. Если координаты по оси абсцисс равны, то больший приоритет имеет точка с меньшей координатой по оси ординат. Если координаты точек совпадают, то приоритет выше у точки начала отрезка.

Результатом предварительного этапа являются множество отрезков *L* и очередь точек *E*.

На рис. 3, 4 и 5 представлен алгоритм построения графа противоречий на основе алгоритма заметающей прямой. Входными данными для алгоритма являются очередь с приоритетом точек-событий *E*, упорядоченное множество отрезков *S* – множество отрезков, которое пересекает выметающая прямая в каждый момент времени.

Упорядочивание отрезков во множестве *S* производится по возрастанию координат по оси ординат точек начала отрезков. Если координаты точек начала по оси ординат равны у двух отрезков, то упорядочивание происходит по координате по оси ординат точек концов отрезков.

На первом шаге алгоритма производится проверка, что очередь *E* не пустая. Если в *E* не осталось точек-событий, алгоритм завершается. Иначе из *E* извлекается точка с наибольшим приоритетом – *e*.

Если *e* – точка начала отрезка, то в *S* добавляется отрезок s_1, началом которого является *e*. Если в *S* существуют отрезки выше или ниже s_1, то для таких пар отрезков производится проверка их взаимного положения согласно алгоритму, представленному на рис. 5.

Если *e* – точка начала отрезка и в *S* существуют отрезки выше или ниже s_1, то для таких пар отрезков производится проверка их взаимного положения согласно алгоритму, представленному на рис. 6. Далее s_1 удаляется из *S*.

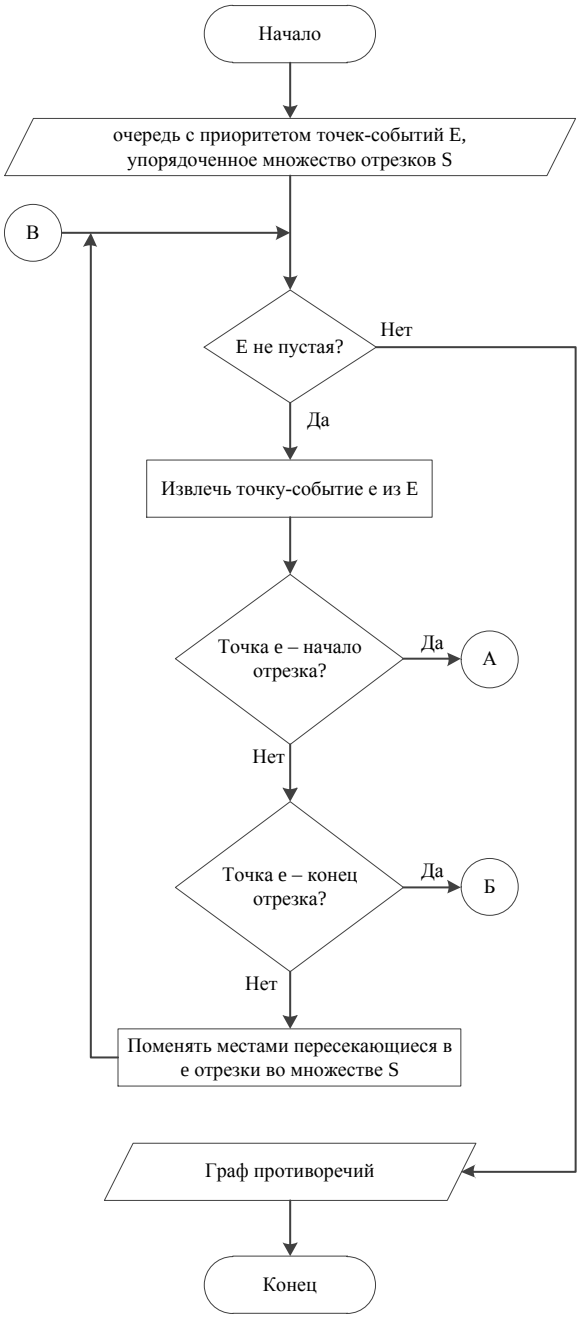


Рис. 3. Алгоритм построения графа противоречий

Если *e* – точка пересечения отрезков, то пересекающиеся в *e* отрезки меняются местами в *S*. Далее повторяются все действия, описанные выше, до тех пор, пока в *E* существует хотя бы одна точка-событие.

В результате работы алгоритма, представленного на рис. 3, 4 и 5, формируется граф противоречий для критического исходного слоя топологии СБИС.

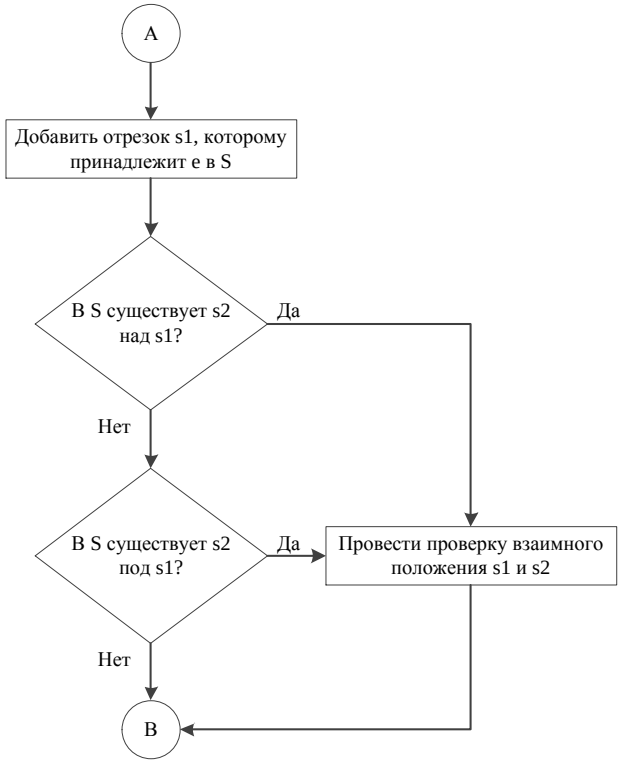


Рис. 4. Обработка точки-события – начала отрезка

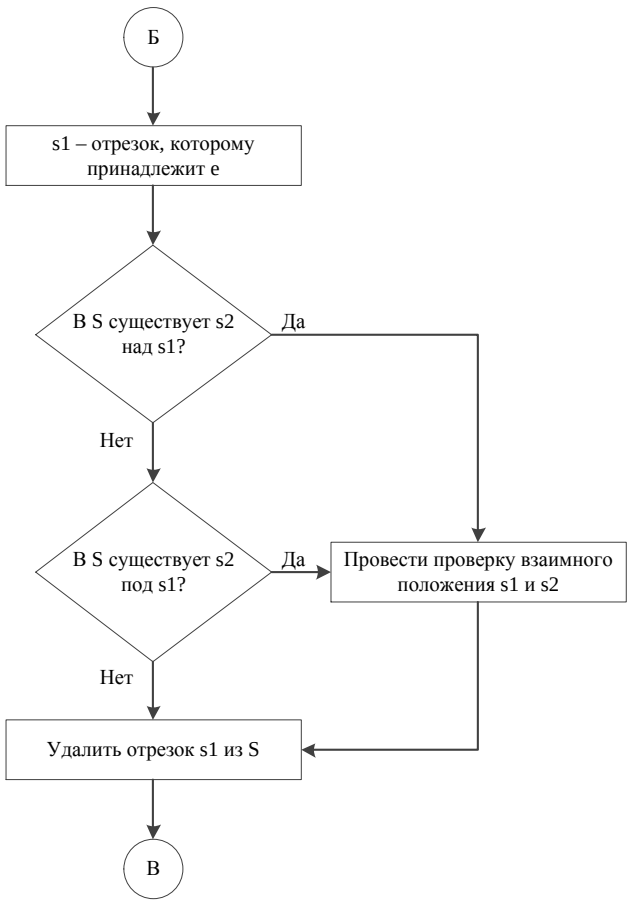


Рис. 5. Обработка точки-события – конца отрезка

На рис. 6 представлен алгоритм проверки взаимного расположения двух отрезков - s_1 и s_2, который используется для коррекции графа противоречий, полученного по алгоритму (рис. 3). Входными параметрами алгоритма являются параметр технологии двойного шаблона *d*, граф противоречий G_1, полученный по алгоритму (рис. 3), граф перекрывающих друг друга полигонов G_2, очередь с приоритетом точек-событий *E*, отрезки s_1 и s_2.

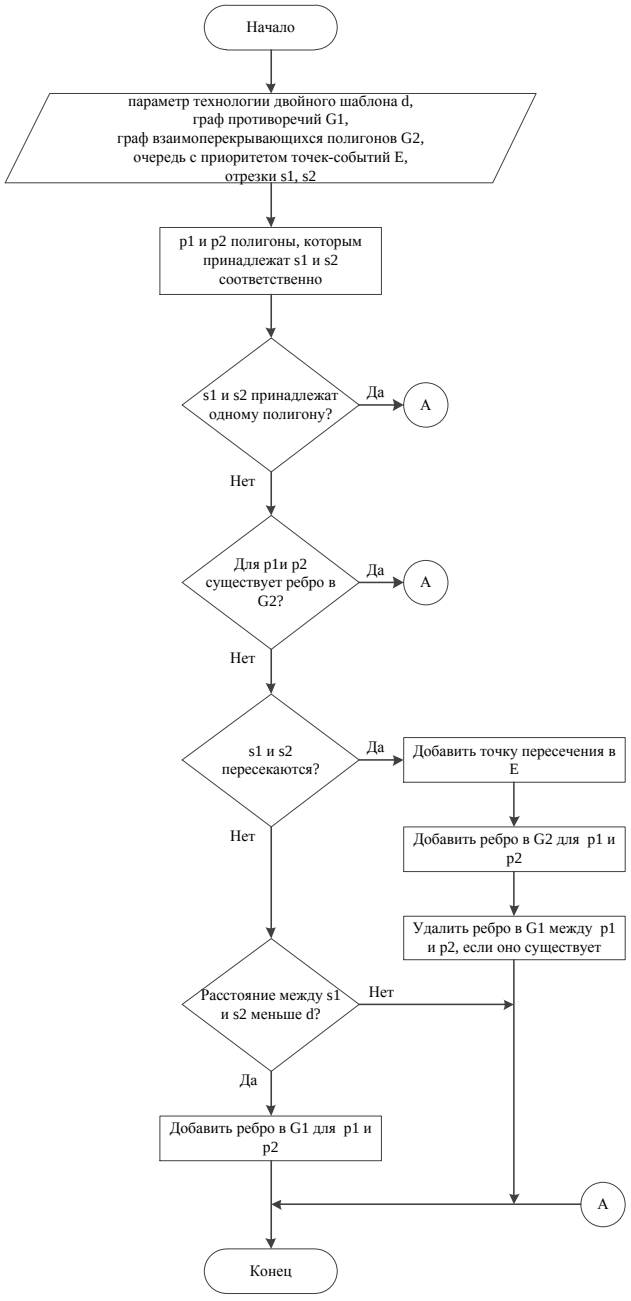


Рис. 6. Алгоритм проверки взаимного расположения двух отрезков

Вершины графа противоречий G_1 соответствуют полигонам, составляющим топологию СБИС. Ребра между вершинами добавляются в случае нахождения конфликта между соответствующими полигонами. Вершины графа G_2 соответствуют полигонам, составляющим топологию СБИС. Ребра между вершинами добавляются в случае, если полигоны накладываются друг на друга.

На первом шаге алгоритма определяются полигоны p_1 и p_2, которым принадлежат отрезки s_1 и s_2, соответственно. Если s_1 и s_2 принадлежат одному и тому же полигону или если для p_1 и p_2 существует ребро в G_2, работа алгоритма завершается.

Если s_1 и s_2 пересекаются, то выполняются следующие шаги алгоритма. Точка их пересечения добавляется в *E*; в G_2 добавляется ребро для p_1 и p_2; если для p_1 и p_2 существует ребро в графе противоречий G_1, оно удаляется.

Если расстояние между s_1 и s_2 меньше заданного параметра технологии двойного шаблона, то для p_1 и p_2 добавляется ребро в G_1.

IV. Декомпозиция топологии СБИС для технологии двойного шаблона на основе графа противоречий

Для декомпозиции топологии СБИС для технологии двойного шаблона предлагается использовать разработанный модифицированный параллельный алгоритм проверки графа на двудольность [7]. Если соответствующий граф двуцветный, то критический слой можно декомпозировать на два новых. В том случае, если граф не двудольный, собирается соответствующая статистика для предоставления рекомендаций к технологии мультишаблона.

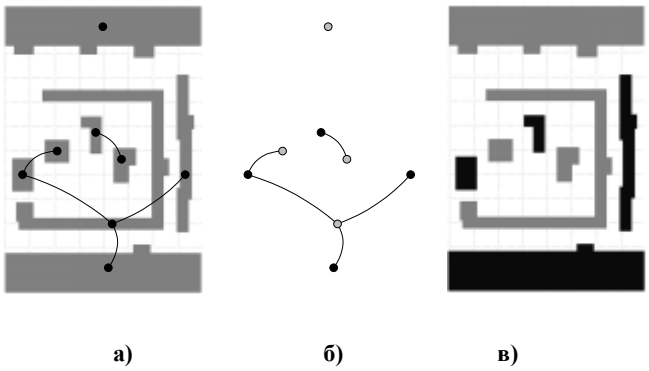
Для проверки графа на двудольность в [7] предлагается использовать параллельный поиск в ширину. Для проверки достаточно в каждой компоненте связности выбрать любую вершину и помечать оставшиеся вершины во время обхода графа поочерёдно в разные цвета. Если при этом не возникнет конфликта, вершины каждого цвета образуют отдельное множество, которое не имеет пересечений с множеством вершин другого цвета.

На основании предложенных алгоритмов реализовано программное обеспечение Parallel DPLayout Migrator на языке C++ с использованием фреймворка OpenMP.

На рис. 7 представлена иллюстрация работы алгоритма на примере топологии логического элемента 4И. На рис. 7а показаны слой металлизации элемента 4И и граф противоречий до проверки на двудольность. На рис. 7б показан граф противоречий после его окраски. На рис. 7в показаны два новых слоя, полученных в результате декомпозиции критического слоя.

В таблице 1 приведены результаты экспериментальных исследований корректности разработанных подходов. В качестве основной метрики выбрано относительное минимальное расстояние [5] между элементами топологии, которое

рассчитывается как отношение между минимальным расстоянием между полигонами в исходном критическом слое до декомпозиции к минимальному расстоянию между полигонами в слоях декомпозиции.



а) б) в)

Рис. 7. Трансформация слоя металлизации элемента 4И для технологии двойного шаблона: а – топологический слой и граф противоречий; б – граф противоречий после окраски; в – два новых полученных в результате декомпозиции слоя (первый слой дан серым, второй слой черным)

Следует отметить, что для всех тестовых топологий удалось увеличить минимальное расстояние между полигонами в критическом слое в среднем на 35%. Учитывая, что одним из наиболее простых правил DFM является увеличение минимального расстояния между полигонами [8], можно сделать вывод о потенциальном улучшении воспроизводимости критического топологического слоя. Однако эта метрика сильно варьируется для различных топологий: от минимума 10% для ячейки памяти до 80% для сумматора.

Таблица 1

Результаты экспериментального исследования

Топология	Относительное минимальное расстояния, слой 1	Относительное минимальное расстояния, слой 2
Мультиплексор	1,15	3,68
4И	1,65	2,18
4ИЛИ	1,27	6,35
Искл. ИЛИ	1,15	1,18
Ячейка памяти	1,10	1,24
Сумматор	1,80	2,05
Всего	**1,35**	**2,78**

Было проведено экспериментальное исследование временной сложности разработанных алгоритмов. Экспериментальное исследование проводилось на рабочей станции следующей конфигурации: четырехъядерный процессор Intel Core i5, 4 Гб оперативной памяти.

В качестве метрики, описывающей размерность задачи, было выбрано количество полигонов в исходном критическом слое. Для проведения исследования были подготовлены тестовые файлы в формате GDSII размером 10 Кб (18 полигонов), 100 Кб (180 полигонов), 1 Мб (1 800 полигонов), 10 Мб (18 000 полигонов), 100 Мб (180 000 полигонов), 1 Гб (1 800 000 полигонов). Результаты исследования приведены на рис. 8.

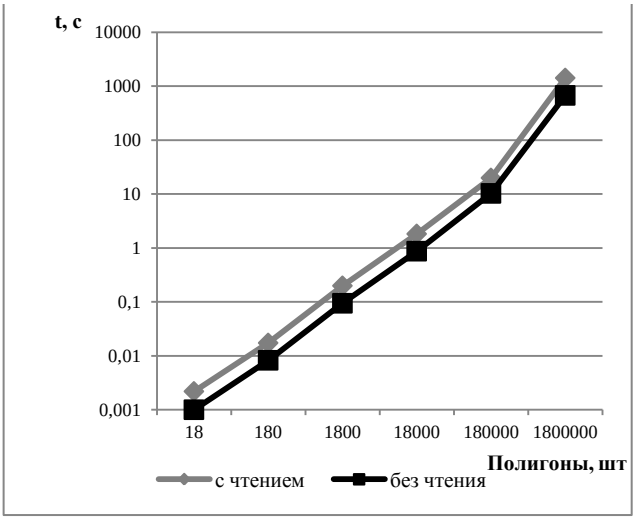


Рис. 8. Зависимость времени работы Parallel DPLayout Migrator от размерности задачи

На основании полученной зависимости временных затрат на трансформацию топологии для технологии двойного шаблона от размерности задачи можно сделать вывод, что алгоритм имеет временную сложность *nlog(n).*

Также необходимо отметить, что с помощью разработанного программного обеспечения Parallel DPLayout Migrator была произведена попытка провести декомпозицию реальной топологии (размер файла 1,5 Гб). Декомпозиция оказалась невозможна из-за неразрешимых противоречий между полигонами критического слоя. Время работы программы составило 40 минут.

Было проведено экспериментальное исследование времени работы программного обеспечения Parallel DPLayout Migrator в зависимости от количества, используемых вычислительных потоков. В качестве тестовой топологии использовался файл описания топологии в формате GDSII 100 Мб (180 000 полигонов). Результаты исследования приведены на рис. 9. При увеличении числа потоков от 1 до 8 было получено двукратное сокращение времени работы программы.

V. Заключение

Предложенный подход способен повысить качество воспроизведения топологий субмикронных СБИС за счет увеличения минимального расстояния между полигонами. Увеличение минимального расстояние подтверждается результатами экспериментального исследования (табл. 1).

Декомпозиция критических слоев топологии СБИС выполняется при помощи разработанного алгоритма построения графа противоречий на основе модифицированного алгоритма сканирующей прямой и разработанного алгоритма декомпозиции топологического слоя на основе модифицированного алгоритма проверки графа противоречий на двудольность.

Следует отметить, что разработанные алгоритмы достаточно хорошо масштабируются для решения задач с большой размерностью, что подтверждается экспериментальными данными (рис. 8).

Помимо этого предложенный подход способен снизить временные затраты на проектирование топологий субмикронных СБИС за счет снижения времени, которое затрачивается на трансформацию топологии для технологии двойного шаблона. Снижение временных затрат достигается путем эффективного использования вычислительных ресурсов высокопроизводительных систем, что подтверждается результатами экспериментального исследования (рис. 9).

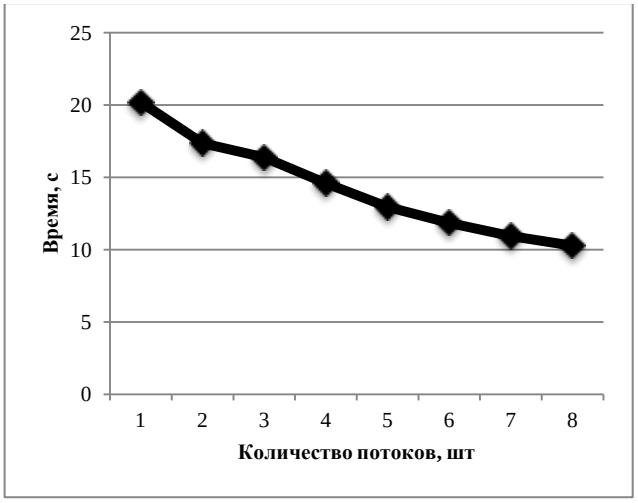


Рис. 9. Зависимость времени работы Parallel DPLayout Migrator от количества потоков

Разработанные параллельные алгоритмы позволяют достичь двукратного сокращения времени декомпозиции исходного критического слоя (180 000 полигонов, размер файла более 100 Мб) тестовой топологии на два новых уже при использовании 8 потоков на 4 процессорах.

Дальнейшее развитие предложенных подходов, прежде всего, заключается в получении максимально равномерного распределения полигонов по слоям после декомпозиции и в повышении эффективности использования вычислительных ресурсов высокопроизводительных вычислительных систем.

Поддержка

Работа выполнена при частичной финансовой поддержке гранта Президента РФ по государственной поддержке ведущих научных школ (грант НШ-2903.2014.9) и гранта РФФИ, проект 14-07-31074.

Литература

[1] URL: http://www.itrs.net/ (дата обращения: 28.01.2014).
[2] Shakhnov V., Makarchuk V., Zinchenko L., Verstov V. Heterogeneous knowledge representation for VLSI systems and MEMS design // Proc. IEEE 4th International Conference CogInfoCom'13. Hungary. P. 189-194.
[3] Зинченко Л.А., Аверьянихин А.Е. Программа трансформации топологии субмикронных сверхбольших интегральных схем для технологии двойного фотошаблона // Программные продукты и системы. 2011. № 1. С. 7-10.
[4] Шахнов В.А., Зинченко Л.А. и др. Алгоритмы трансформации топологии субмикронных сверхбольших интегральных схем // Вестник МГТУ им. Н.Э. Баумана. 2011. № 1. С. 76-87.
[5] Шахнов В.А., Зинченко Л.А., Верстов В.А.. Трансформация топологии субмикронных СБИС для технологии двойного шаблона // Микроэлектроника. 2013. Т. 42. № 6. С. 427-439.
[6] Yao et al. WIPAL: Window-based parallel layout decomposition in double patterning lithography // Proc. IEEE ICSICT. 2012. P. 1-4.
[7] Verstov V., Shakhnov V., Zinchenko L. Parallel Algorithm of SOI Layout Decomposition for Double Patterning Lithography on High-Performance Computer Platforms // 5th IFIP WG 5.5/SOCOLNET Doctoral Conference on Computing, Electrical and Industrial Systems. DoCEIS, 2014. Proceedings. Costa de Caparica, Portugal. April 7-9, 2014. P. 543-550.
[8] De Dood P. Impact of DFM and RET on Standard-Cell Design Methodology // Proc. of EDP Workshop. 2003.