УДК 658.512.2.011

Исследование модели распределенного топологического проектирования СБИС средствами иерархической клиент-серверной архитектуры

В.М. Глушань[1], П.В. Лаврик[2], М.В. Рыбальченко[1]

[1]Инженерно-технологическая академия ЮФУ, gluval07@mail.ru

[2]ООО Лодосс, levarto@mail.ru

[1]Инженерно-технологическая академия ЮФУ, mic_v@mail.ru

***Аннотация* — Для построения перспективных подсистем конструкторского проектирования СБИС предлагается использовать иерархическую клиент-серверную архитектуру. Проведен концептуальный анализ такой архитектуры, в основу которого положено многоуровневое дихотомическое разбиение неориентированного графа с разными количествами вершин и локальных степеней. В результате проведенных исследований подтверждена гипотеза, что такая архитектура позволит строить подсистемы конструкторского проектирования СБИС с более высоким быстродействием по сравнению с сосредоточенными подсистемами.**

***Ключевые слова* — СБИС, иерархическая клиент-серверная архитектура, дихотомическое разбиение, неориентированный граф, локальная степень, трудоемкость.**

I. Введение

Прогрессивное развитие техники неразрывно связан с ее непрерывным усложнением как в количественном, так и в качественном отношении. С особой очевидностью количественный рост проявляется в электронно-вычислительной технике. Так, первая ЭВМ ЭНИАК, анонсированная в конце 40-х годов прошлого века, содержала порядка 17500 электронных ламп, 7200 диодов, 1500 реле и выполняла всего 5000 операций в секунду [1]. Размерности современных СБИС достигают $1{,}5 \cdot 10^9$ и более транзисторов на кристалле [2], они обладают быстродействием в миллионы операций в секунду. Как отмечено в [3], проектирование без использования средств автоматизированного проектирования СБИС регулярной структуры даже со значительно меньшим числом транзисторов – 10^5 – требовало (середина 70-хх годов прошлого века) затрат времени в 120 человеко-лет, что соизмеримо с проектированием авиалайнера. Если следовать закону Мура [4], который в одной из формулировок гласит, что удвоение транзисторов в чипе происходит каждые 2 года, и такая тенденция сохранится, то разработчики и производители СБИС будут еще долго вынуждены осуществлять погоню за быстродействием средств автоматизированного проектирования.

В области повышения быстродействия автоматизированных систем проектирования можно выделить три хронологических подхода: 70–80-е гг. прошлого столетия – расцвет многопроцессорных вычислительных систем,80–90-е гг. – интенсивные исследования по созданию и применению аппаратных ускорителей, 90-е гг. и настоящее время – сетевые информационные технологии. Среди последних развитие получили клиент-серверные технологии.

В работах [5-7] теоретически и экспериментально показано, что на основе распределенных клиент-серверных технологий можно строить подсистемы конструкторского проектирования электронных схем, в том числе СБИС. Основной вопрос, который решался в цитированных работах – повышение быстродействия подсистемы. Исследования показали, что клиент-серверная архитектура позволяет повысить быстродействие по сравнению с сосредоточенным проектированием (на одном компьютере), но лишь в несколько раз. Причем число клиентских компьютеров в зависимости от сложности проектируемой схемы имеет оптимальное значение, но растет оно очень медленно даже при очень быстром возрастании проектируемой схемы, а быстродействие увеличивается также только в разы. Такое повышение быстродействия проектирующих систем уже нельзя считать достаточным. Поэтому продолжаются поиски подходов построения подсистем на основе клиент-серверных архитектур, дающих более существенный выигрыш в быстродействии при конструкторском проектировании СБИС.

II. Возможности последовательно-параллельной и параллельно-параллельной архитектур

Использование этих архитектур для распределенного проектирования обеспечивает временной выигрыш за счет разделения функций проектирования между клиентами и сервером **[**6, 7**]**. Это разделение можно выполнять различными способами. В первом варианте подсистемы использовалась модель распределенной клиент-серверной архитектуры, в которой разбиение исходной схемы на части выполняется сервером, размещение элементов схемы и трассировка соединений между элементами схемы в каждой части выпол-

няется клиентами, а трассировка соединений между частями схемы (дотрассировка) выполняется тем же самым сервером. Поскольку в такой модели общее время проектирования определяется последовательным временем работы клиентской и серверной частей модели, то максимальный временной выигрыш по сравнению с сосредоточенной подсистемой можно получить, выполняя трассировку максимально возможного количества связей на клиентских компьютерах, уменьшая тем самым долю работы сервера при дотрассировке. Такую модель естественно считать последовательно-параллельной.

В цитированных работах показано, что реальный временной выигрыш оказывается лишь в разы больше, чем в сосредоточенной подсистеме проектирования. Казалось бы, что достигнутая величина выигрыша является предельной, и получить более высокие его значения невозможно. Как показал анализ, приведенный в [8], в последовательно-параллельной модели клиент-серверной архитектуры распределенной подсистемы имеются резервы для ее усовершенствования. Эти резервы могут быть реализованы, если последовательно-параллельную модель сделать полностью параллельной. Для этого необходимо, чтобы сервер не ожидал окончания работы клиентов, а одновременно, т.е. параллельно с ними, выполнял межблочную трассировку. Если эту идею реализовать, то последовательно-параллельная модель трансформируется в параллельно-параллельную модель клиент-серверной архитектуры распределенной подсистемы конструкторского проектирования СБИС.

Для реализации параллельно-параллельной модели необходимо задачу дотрассировки межблочных связей, выполняемую сервером, решать одновременно с решением задач размещения и трассировки отдельных частей схемы на клиентских компьютерах. Но как это сделать, если до окончания работы клиентов неизвестна топология соединений, полученных ими в различных частях схем? Однако здесь важно знать топологию отдельных трасс не внутри части схемы, а на ее границах. Это уже несколько упрощает задачу, так как можно всегда определиться, на какую сторону части схемы (если это БИС, то имеется в виду сторона кристалла) следует выводить цепь, которая имеет продолжение в других частях схемы. Это аналогично тому, как выводятся печатные проводники на разъем ТЭЗа (типового элемента замены) или как подводятся цепи к внешним выводам СБИС процессора, расположенным по его периметру.

Но это только первый шаг. Второй шаг состоит в том, что, зная взаимосвязи между частями схемы, которые определяются в процессе разбиения схемы на части, можно заранее, т.е. параллельно с работой клиентов, прокладывать нужное число магистралей в каналах, как это делается при канальной трассировке. И решение этой задачи можно возложить на сервер, который до этого простаивал.

Проведенные в [8] экспериментальные исследования позволяют сделать вывод о том, что параллельно-параллельная архитектура распределенной подсистемы действительно обладает большим быстродействием, но качество трассировки на данном этапе исследования оказалось хуже. Можно предположить, что это было следствием незначительного объема экспериментальных исследований.

III. Обоснование концепции иерархической клиент-серверной архитектуры

Частный вид этой архитектуры (дихотомической) приведен на рис. 1. Название этой архитектуры дихотомической следует из того, что основными архитектурно образующими элементами являются сервер, расположенный на предыдущем уровне иерархии, и два клиента, расположенных на следующем уровне иерархии, информационно связанных с указанным сервером.

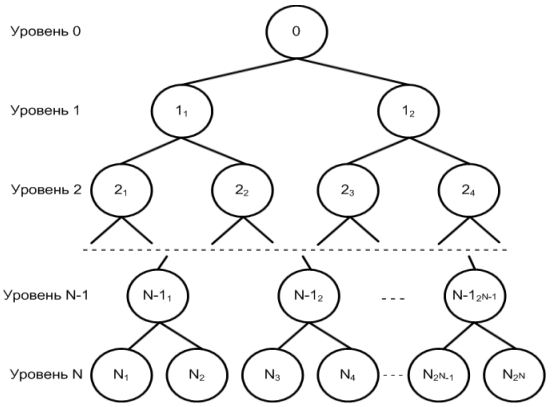

Рис. 1. Иерархическая клиент-серверная архитектура

В приведенном на рис.1 частном случае иерархической архитектуры на каждом уровне применяется 2^k компьютеров. В общем случае их число может быть любым. Но применение именно 2^k компьютеров объясняется двумя факторами. Во-первых, как показано в [6, 7], в большом диапазоне изменения сложности схем число компьютеров-клиентов ограничивается двумятремя. Во-вторых, концептуальный анализ дихотомической архитектуры выполнить проще, чем с другим числом компьютеров. Если этот анализ покажет эффективность иерархической архитектуры, то в последующих исследованиях можно провести анализ общего случая.

Процесс конструкторского проектирования начинается на нулевом уровне, где на верхнем уровне дихотомической иерархической архитектуры находится главный сервер C_0. Этот сервер вначале разбивает исходную схему на две подсхемы и пересылает каждую подсхему одному из двух компьютеров-клиентов первого уровня, каждый из которых одновременно является сервером для двух компьютеров второго уровня. Компьютеры второго уровня разбивают свою часть схемы также на две подсхемы и пересылают их на два компьютера третьего уровня. Этот процесс продолжается до тех пор, пока соответствующие части подсхем не поступят на компьютеры последнего N-го уровня. На последнем уровне находятся компьютеры, которые

являются только клиентами для предшествующего уровня.

После того, как на компьютеры *N*-го уровня поступят соответствующие части подсхем, в них начнется процесс решения задач размещения и трассировки. По окончании этого процесса информация о результатах размещения и трассировки с каждой пары компьютеров-клиентов *N*-го уровня поступает на соответствующий сервер предыдущего (*N*-1)-го уровня. Каждый из серверов (*N*-1)-го уровня решает задачу объединения двух частей схемы *N*-го уровня – выполняет трассировку между этими двумя частями схемы. В результате получается удвоенная часть схемы. Поскольку каждая пара компьютеров (*N*-1)-го уровня соединена с соответствующим компьютером (*N*-2)-го уровня, то каждый такой компьютер решает задачу объединения двух удвоенных частей схемы. В результате в каждом компьютере (*N*-2)-го уровня получается учетверенная часть схемы. Этот процесс продолжается до тех пор, пока на главный сервер C_0 нулевого уровня не поступит информация с двух компьютеров первого уровня, которую он объединит в окончательную полную схему.

IV. Методика анализа иерархической архитектуры

В качестве модели схемы мы используем неориентированный граф, вершины которого отображают элементы схемы, а ребра – соединения (цепи) между элементами. Цель анализа состоит в установлении зависимости трудоемкости процесса конструкторского проектирования схем различной сложности от числа иерархических уровней. Сложность схемы определяется числом вершин *M* и ребер *R* в моделирующем графе. В свою очередь число ребер зависит от локальных степеней вершин *L*. Поэтому исходными параметрами моделируемой схемы будут именно *M* и *L*. Если все вершины имеют одинаковую локальную степень, то число ребер в моделирующем графе, в соответствии с [9], определяется простой формулой $R = ML/2$.

Исходное число вершин и ребер моделирующего графа будем различным образом распределять по компьютерам дерева иерархической архитектуры. Наиболее просто и естественно это можно делать для вершин графа, распределяя их в равном количестве по всем компьютерам данного уровня. Сложнее это сделать для ребер даже в случае, когда локальная степень всех вершин одинакова, так как можно получить множество разбиений графа с одинаковым количеством вершин в подграфах, но с разным числом внешних ребер, т.е. между подграфами.

После анализа различных подходов по данному вопросу, изложенных, например, в [6], было принято решение использовать процентное значение числа внешних ребер графа, разбиваемого на два подграфа. Причем число вершин во всех подграфах одного уровня иерархии и число внешних ребер между подграфами каждой пары принимается одинаковым. Таким образом, в методику анализа положены следующие соотношения.

На нулевом уровне иерархии y_0 определяется общее число ребер исходного графа (они являются внутренними ребрами) по формуле $R_0 = ML/2$.

На каждом последующем уровне y_i, $i=1,\ldots,n$ определяется число внешних ребер между подграфами каждой пары и число внутренних ребер в каждом подграфе соответственно по формулам:

$$R(i)_{внеш} = \alpha R(i-1)_{вн},$$

$$R(1)_{вн} = \ldots = R(2^i)_{вн} = \frac{R(i-1)_{вн} - R(i)_{внеш}}{2},$$

где α – доля в % числа внешних ребер между каждой парой подграфов от внутренних ребер во всех подграфах i – го уровня.

Для демонстрации приведенной методики рассмотрим пример. Пусть $M = 1024$, $L = 6$, $\alpha = 0{,}2$. Тогда процесс распределения внутренних и внешних ребер по уровням иерархии по подграфам будет следующим.

Уровень 1:

$$R(1)_{внеш} = 0{,}2R(0)_{вн} = 0{,}2 \cdot 3072 = 614,$$

$$R(1)_{вн} = R(2)_{вн} = \frac{3072 - 614}{2} = 1229.$$

Уровень 2:

$$R(2)_{внеш} = 0{,}2 \cdot 1229 = 246,$$

$$R(1)_{вн} = \ldots = R(4)_{вн} = \frac{1229 - 246}{2} = 492.$$

Уровень 3:

$$R(3)_{внеш} = 0{,}2 \cdot 492 = 98,$$

$$R(1)_{вн} = \ldots = R(8)_{вн} = \frac{492 - 98}{2} = 197.$$

Уровень 4:

$$R(4)_{внеш} = 0{,}2 \cdot 197 = 39,$$

$$R(1)_{вн} = \ldots = R(16)_{вн} = \frac{197 - 39}{2} = 79.$$

Используя полученные результаты, можно рассчитать ориентировочную сложность процесса проектирования, используя 1, 2, 3, 4 и т.д. уровня для процесса иерархического проектирования. Уместно отметить, что при проектировании можно использовать алгоритмы различной временной сложности. Исходя из практических соображений, как отмечено в [10,11], целесообразно использовать полиномиальные алгоритмы как для размещения, так и для трассировки. Однако алгоритмы размещения существенно быстрее решают свою задачу по сравнению с алгоритмами трассировки. Наи-

более распространенными являются алгоритмы размещения с квадратичной временной сложностью [12, 13]. Наиболее сложную часть топологического проектирования представляют алгоритмы трассировки, временная сложность которых существенно выше, чем алгоритмов размещения. Наибольшее применение находит волновой алгоритм трассировки, сложность которого можно считать кубической [14].

Поскольку в статье предпринята попытка оценки возможностей распределенной иерархической клиент-серверной архитектуры для топологического проектирования СБИС, то на первом этапе исследования авторы сочли целесообразным использовать комбинацию алгоритмов размещения и трассировки разной сложности. При этом рассматривается два случая: в обоих случаях будем опираться на алгоритм размещения квадратичной сложности, а алгоритм трассировки в первом случае будем считать квадратичным, во втором – кубическим.

Отметим также характерную особенность иерархического проектирования. При переходе от некоторого уровня иерархии к следующему, более низкому, на каждом уровне, кроме последнего, выполняется разбиение схемы на части. При переходе от некоторого уровня иерархии к следующему, более высокому, на каждом уровне, кроме последнего, выполняется только трассировка. На последнем уровне надо решать задачи и размещения, и трассировки для каждой части схемы. Поскольку эти задачи решаются одновременно (параллельно) для всех частей и все они одинаковы, сложность проектирования на последнем уровне будет определяться временем размещения и трассировки только в одной части.

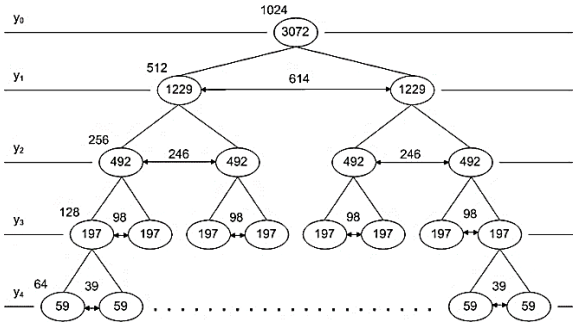

Рис. 2. Пример распределения вершин и ребер в подграфах на разных уровнях

Полученный результат размещения и трассировки с двух компьютеров последнего уровня будет передаваться на один предыдущий (материнский) компьютер. Очевидно, что в материнском компьютере нужно выполнить только трассировку внешних связей между частями схемы, пришедшими с последнего уровня. Такие же части схемы будут сформированы двумя другими компьютерами последнего уровня. Поэтому на предпоследнем (n-1)-м уровне будет сформировано четное число удвоенных частей схемы (два подграфа) с уже размещенными элементами. Для этих удвоенных частей схемы в следующем (n-2)-м уровне будет свой материнский компьютер, который будет выполнять только трассировку между пришедшими на него частями схемы. Этот процесс будет продолжаться аналогичным образом, пока не дойдет до нулевого уровня. Описанный процесс иллюстрирует рис. 2.

Таким образом, подсчет временной сложности T_n при использовании n уровней иерархии можно представить следующими конструктивными соотношениями:

$$T_n = O\left(\sum_{i=1}^{n} M^2(n-i) + M^2(n) + R^2(n)_{вн} + \sum_{i=0}^{n-1} R^2(n-i)_{внеш}\right) \quad (1)$$

– для квадратичного разбиения, размещения и трассировки, где $M(n)$ и $R(n)_{вн}$ – соответственно число вершин и число ребер в каждом подграфе на n-м уровне иерархии;

$$T_n = O\left(\sum_{i=1}^{n} M^2(n-i) + M^2(n) + R^3(n)_{вн} + \sum_{i=0}^{n-1} R^3(n-i)_{внеш}\right) \quad (2)$$

– для квадратичного разбиения и размещения, но кубической трассировки.

Рассмотрим пример использования соотношений (1) и (2) для расчета трудоемкости процесса проектирования иерархической архитектурой, опираясь на результаты расчета внутренних и внешних ребер подграфов на каждом уровне, приведенные выше для случая $M = 1024$, $L = 6$, $\alpha = 0{,}2$.

Для квадратичного разбиения, размещения и трассировки, используя соотношение (1), получим оценки трудоёмкости при использовании различного числа уровней иерархии:

$$\left.\begin{aligned} &T_4=128^2+256^2+512^2+1024^2+64^2+79^2+39^2+98^2+\\ &+246^2+614^2=1850000;\\ &T_3=256^2+512^2+1024^2+128^2+197^2+98^2+246^2+\\ &+614^2=1880000;\\ &T_2=512^2+1024^2+256^2+492^2+246^2+614^2=\\ &=2060000;\\ &T_1=1024^2+512^2+1229^2+614^2=3200000. \end{aligned}\right\} \quad (3)$$

Для квадратичного разбиения и размещения, но кубической трассировки, аналогично, но используя соотношение (2), получим следующие оценки:

$$\left.\begin{aligned} &T_4=128^2+256^2+512^2+1024^2+64^2+79^3+39^3+98^3+\\ &+246^3+614^3=250000000;\\ &T_3=256^2+512^2+1024^2+128^2+197^3+98^3+246^3+\\ &+614^3=257000000;\\ &T_2=512^2+1024^2+256^2+492^3+246^3+614^3=\\ &=367000000;\\ &T_1=1024^2+512^2+1229^3+614^3=2090000000. \end{aligned}\right\} \quad (4)$$

Результаты аналогичных расчётов по соотношениям (1) и (2), но для разных случаев значений L и α, включая результаты расчетов (3) и (4), приведены в таблице. Приведенные параметры модели схемы являются иллюстративными и выбраны, исходя, с одной стороны, из соображений относительной простоты вычислений, с другой стороны, путем вариации параметра α моделируются схемы с различной насыщенностью связями. Величина L = 6 соответствует типовому значению среднего коэффициента разветвления по выходу и среднего количества входов логического элемента СБИС в соответствии с [15].

Таблица 1

Результаты расчета трудоемкости для различных исходных данных

M =1024, L =6, $R_{внеш} = 0{,}2R_{вн}$, разбиение, размещение и трассировка квадратичные						
Уровни	1	2	3	4	–	–
T_i	$3{,}20\cdot10^6$	$2{,}06\cdot10^6$	$1{,}88\cdot10^6$	$1{,}85\cdot10^6$	–	–
M =1024, L =6, $R_{внеш} = 0{,}2R_{вн}$, разбиение и размещение квадратичные, трассировка кубическая						
Уровни	1	2	2	4	–	–
T_i	$0{,}21\cdot10^{10}$	$0{,}037\cdot10^{10}$	$0{,}026\cdot10^{10}$	$0{,}025\cdot10^{10}$	–	–
M =1024, L =6, $R_{внеш} = 0{,}3R_{вн}$, разбиение, размещение и трассировка квадратичные						
Уровни	1	2	3	4	–	–
T_i	$3{,}32\cdot10^6$	$2{,}47\cdot10^6$	$2{,}38\cdot10^6$	$2{,}37\cdot10^6$	–	–
M =1024, L =6, $R_{внеш} = 0{,}3R_{вн}$, разбиение и размещение квадратичные, трассировка кубическая						
Уровни	1	2	3	4	–	–
T_i	$0{,}2\cdot10^{10}$	$0{,}087\cdot10^{10}$	$0{,}082\cdot10^{10}$	$0{,}0819\cdot10^{10}$	–	–
M =1024, L =6, $R_{внеш} = 0{,}4R_{вн}$, разбиение, размещение и трассировка квадратичные						
Уровни	1	2	3	4	–	–
T_i	$3{,}67\cdot10^6$	$3{,}10\cdot10^6$	$3{,}058\cdot10^6$	$3{,}057\cdot10^6$	–	–
M =1024, L =6, $R_{внеш} = 0{,}4R_{вн}$, разбиение и размещение квадратичные, трассировка кубическая						
Уровни	1	2	3	4	–	–
T_i	$0{,}26\cdot10^{10}$	$0{,}193\cdot10^{10}$	$0{,}191\cdot10^{10}$	$0{,}1908\cdot10^{10}$	–	–
M =1024, L =12, $R_{внеш} = 0{,}2R_{вн}$, разбиение, размещение и трассировка квадратичные						
Уровни	1	2	3	4	5	6
T_i	$8{,}86\cdot10^6$	$4{,}1\cdot10^6$	$3{,}34\cdot10^6$	$3{,}22\cdot10^6$	$3{,}199\cdot10^6$	$3{,}196\cdot10^6$
M =1024, L =12, $R_{внеш} = 0{,}2R_{вн}$, разбиение и размещение квадратичные, трассировка кубическая						
Уровни	1	2	3	4	5	6
T_i	$1{,}67\cdot10^{10}$	$0{,}29\cdot10^{10}$	$0{,}204\cdot10^{10}$	$0{,}199\cdot10^{10}$	$0{,}198\cdot10^{10}$	$0{,}198\cdot10^{10}$

V. Заключение

Анализ приведенных графиков позволяет сделать следующие выводы:

- с увеличением числа уровней количество вершин дерева растет экспоненциально, трудоемкость быстро уменьшается независимо от количества элементов и насыщенности схемы связями, размерность задачи уменьшается;

- спад трудоемкости с увеличением числа уровней быстро стабилизируется и уже при трех, четырёх уровнях становится практически неизменным;

- при применении более сложных алгоритмов трассировки (предполагается и более высокое качество результатов их работы) увеличивается и трудоемкость, которая, однако, быстрее уменьшается по сравнению с менее сложными алгоритмами с ростом числа уровней;

- с увеличением доли внешних связей между частями схемы трудоемкость увеличивается, что естественно, а её спад по мере увеличения уровней происходит медленнее.

Отсюда можно сделать общий вывод – иерархическая клиент-серверная архитектура распределенной подсистемы конструкторского проектирования обладает потенциально большим быстродействием относительно сосредоточенной подсистемы. Поэтому более детальное и глубокое её исследование представляет как научный, так и практический интерес.

Поддержка

Работа выполнена при поддержке гранта РФФИ № 15-01-05669 «Разработка основ теории и принципов построения иерархических клиент-серверных архитектур для реализации быстродействующих подсистем конструкторского проектирования СБИС».

ЛИТЕРАТУРА

[1] URL:www.pochitat.com/pervyiy-kompyter-eniac-byil-zapyushhen-14-fevralya.html (accessed 15.03.16).
[2] Асмаков С.В., Пахомов С.О. Железо 2010. КомпьютерПресс рекомендует. – СПб.: Питер, 2010. – 416 с.
[3] Э.Е. Иванов, В.Н. Брюнин. Проблемы создания САПР СБИС. Электронная вычислительная техника. Сборник статей. Выпуск 2. Под ред. ВВ. Пржиялковского. М.: «Радио и связь». – 1988. С. 114-120.
[4] URL: www.tadviser.ru/index.php/ Статьи: Закон_Мура (accessed 15.03.16).
[5] В.М. Глушань, П.В. Лаврик Уточнение клиент-серверной модели распределенной САПР электронных схем. / Изв. Южного Федерального университета. Технические науки. – 2009. – Т. 12. С. 92-97.
[6] В.М. Глушань, П.В. Лаврик. Распределенные САПР. Архитектура и возможности. / Монография. – Старый Оскол: ТНТ, 2014. – 188 с.
[7] В.М. Глушань, П.В. Лаврик. Возможности распределенной подсистемы топологического проектирования, построенной на основе клиент-серверных технологий. / Проблемы разработки перспективных микро и нано-электронных систем. Сб. трудов// Под общей ред. Академика РАН А.Л. Стемпковского. М.: ИППМ РАН, 2014, ч. II. С. 11-14.
[8] В.М. Глушань, П.В. Лаврик. Параллельно-параллельная модель клиент-серверной архитектуры распределенной САПР/ Тр. Конгресса по интеллектуальным системам и информационным технологиям «IS-IT'12». Научное издание в 4-х томах. – М.: Физматлит, 2012. – Т. 1. С. 112-120.
[9] Оре О. Теория графов. – 2-е изд. – М.: Наука, Главная редакция физико-математической литературы, 1980. 336 с.
[10] Глушань В.М., Иванько Р.В. Анализ эффективности распределенных САПР. Изв. ТРТУ. Тематический выпуск «Интеллектуальные САПР». – Таганрог: Изд-во ТРТУ, 2006, № 8. С. 115-120.
[11] Глушань В.М., Лаврик П.В. Концептуальный анализ и построение распределенной подсистемы автоматизированного конструирования электронных схем. Интеллектуальные системы. Коллективная монография. Вып. 5 / Под ред. В.М. Курейчика. – М.: Физматлит, 2011. – 262 с.
[12] Абрайтис Л.Б. Автоматизация проектирования топологии цифровых интегральных микросхем. – М.: Радио и связь, 1985.– 200 с
[13] Проектирование СБИС: Пер. с япон./Ватанабэ М., Асада К., Кани К., Оцуки Т. – М.: Мир, 1988. – 304 с.
[14] http://fvn2009.narod.ru/ Olympiads/Rules_Olympiads/Rules23.htm
[15] Raj A.A., Latha T. VLSI Design. - New Delhi: PHI Learning Private Limited, 2008.

Research of the model of distributed topological VLSI design by means of the hierarchical client-server architecture.

V.M. Glushan[1], P.V. Lavrik[2], M.V. Rybal'chenko3

[1]Engineering and Technological Academy, Southern Federal University, gluval07@mail.ru

[2]Ltd. Lodoss, levarto@mail.ru

[3]Engineering and Technological Academy, Southern Federal University, mic_v@mail.ru

***Keywords* — VLSI, hierarchical client-server architecture, dichotomous partition, of labor input. СБИС, иерархическая клиент-серверная архитектура, дихотомическое разбиение, неориентированный граф, локальная степень, трудоемкость.**

In earlier studies the authors theoretically and experimentally showed that, based on the distribution-divided single-level client-server architectures, it is possible to build design subsystems for electronic circuits, including VLSI. The main question raised in those works - finding the ways to increase the performance of the subsystem. Studies have shown that one-tier client-server architecture allows increasing the speed at a certain optimal number of client computers involved in the design. However, the optimal number of them involved will not considerably increase with gain of complexity of the designed circuit. It is measured in single units even with a sharp increase in circuit complexity that increases the speed only in a few times. And this today is no longer sufficient.

To construct the advanced subsystems of VLSI design, it is proposed to use hierarchical client-server architecture. We conducted a conceptual analysis of this architecture, based on multilevel dichotomous partition of an undirected graph with a different number of vertices and local powers. The proposed method of analysis includes the use of the algorithms of placing and tracing of different complexity - quadratic and cubic. It is shown that with increasing number of levels of hierarchy complexity of the process decreases rapidly regardless of the number of local peaks and their degrees. The decrees of the complexity with increasing number of levels quickly stabilizes and becomes almost constant at three, four levels. The increase in the share of external edges between subgraphs increases the complexity, and its decrease with increasing levels is slower. When applying more sophisticated routing algorithms (it is assumed a higher quality of their performance) the complexity increases, which, however, decreases faster than less complicated algorithms with increasing number of levels. As the number of client computers with increas-

ing number of levels grows exponentially, the number of computers effectively used in the design can be significantly bigger.

The studies confirmed the hypothesis that the multilayered architecture allows building VLSI design subsystems with a higher speed compared to single-layer distributed subsystems.

REFERENCES

[1] URL: www.pochitat.com/pervyiy-kompyter-eniac-byil-zapyushhen-14-fevralya.html (15.03.2016).

[2] Asmakov S.V., Pakhomov S.O. 2010 Iron Computerpress recommends. Saint Petersburg, Peter, 2010. 416 p. (in Russian).

[3] E.E. Ivanov, V.N. Bryunin. Problems of creation of VLSI CAD. Electronic computing machinery. Digest of articles. Issue 2. Ed. BB. Przyjalkowski. Moscow, "Radio and communication." 1988, pp 114-120. (in Russian).

[4] URL: www.tadviser.ru/index.php/ Articles: Zakon_Mura (03.15.2016).

[5] V.M. Glushan, P.V. Lavrik Clarification client-server distributed CAD models of electronic circuits. / Math. Southern Federal University. Technical science. 2009. T. 12. P. 92-97. (in Russian).

[6] V.M. Glushan, P.V. Lavrik. Distributed CAD. Architecture and opportunities. /Monograph. Stary Oskol: TNT, 2014. 188 p. (in Russian).

[7] V.M. Glushan, P.V. Lavrik. Features a distributed subsystem of topological design, built on a client-server based technology. / Problems of advanced micro and nano-electronic systems. Coll. // works, ed. RAN Academician A.L. Stempkovsky. Moscow, IPPM RAN, 2014 hours. II. P. 11-14. (in Russian).

[8] V.M. Glushan, P.V. Lavrik. Parallel-parallel model client-server architecture of distributed CAD / Tr. Congress on Intelligent Systems and Information Technology «IS-IT'12». Scientific edition in 4 volumes. Moscow, FIZMATLIT, 2012. V. 1. P. 112-120. (in Russian).

[9] Ore O. Graph Theory. 2nd ed. Moscow, Science, Home edition of Physical and mathematical literature, 1980. 336 p. (in Russian).

[10] Glushan V.M., Ivanko R.V. Analysis of the effectiveness of distributed CAD. Math. TSURE. Special Issue "Intelligent CAD". Taganrog: TSURE 2006, number 8. pp. 115-120. (in Russian).

[11] Glushan V.M., Lavrik P.V. Conceptual analysis and construction of distributed sub-aided design of electronic circuits. Intelligent systems. The collective monograph. Vol. 5 / Ed. VM Kureichik. Moscow, FIZMATLIT, 2011. 262 p. (in Russian).

[12] Abraytis L.B. Computer-aided design of digital integrated circuits topology. Moscow, Radio and Communications. 1985. 200p. (in Russian).

[13] VLSI Design: Trans. from Japanese / M. Watanabe, K. Asada, Kani K., T. Otsuki. Moscow Mir, 1988. 304 p. (in Russian).

[14] https://mail.rambler.ru/m/redirect?url= http%3A//fvn2009.narod.ru/Olympiads/Rules_Olympiads/Rules23.htm&hash=453f4f66a5187c90d133b698e7447b88.

[15] Raj A.A., Latha T. VLSI Design. New Delhi: PHI Learning Private Limited, 2008.