УДК 004.3:681.518

Д.Е. ИВАНОВ
Институт прикладной математики и механики НАН Украины, г. Донецк

АНАЛИТИЧЕСКАЯ ОЦЕНКА МАСШТАБИРУЕМОСТИ ПАРАЛЛЕЛЬНЫХ МЕТОДОВ МОДЕЛИРОВАНИЯ ЦИФРОВЫХ УСТРОЙСТВ С НЕИСПРАВНОСТЯМИ

Построены аналитические оценки времени работы методов параллельного моделирования с неисправностями для цифровых устройств на логическом уровне представления. Это позволяет выполнять оценку параметров масштабируемости таких методов, а также помочь в изучении вопроса: как параллельное моделирование может быть ускорено с теоретической точки зрения.

Ключевые слова: цифровое устройство, моделирование с неисправностями, параллельные вычисления, масштабируемость.

D.E. IVANOV
Institute of Applied Mathematics and Mechanics, National Academy of Sciences of Ukraine, Donetsk

ANALYTIC EVALUATION OF SCALABILITY OF PARALLEL FAULT SIMULATION METHODS OF DIGITAL DEVICES

Abstract – The aim of the paper is to construct analytical estimations of the working time for parallel fault simulation methods of digital devices for logic level representation.

Review of approaches of parallel fault simulation methods was performed. Basing on this the fault list partitioning algorithm is chosen as the base for analysis. It shows good scalability with increasing number of processors. The modifications of the method for parallel multiprocessor computer systems with shared and distributed memory are considered. For these modifications the estimations of working time are built. The comments about the impact of data transmission among nodes of a computer system on the constructed evaluations are made.

This allows the estimation of the parameters of parallelization of such methods and also helps in the study of the question: how the parallel fault simulation methods can be accelerated from a theoretical point of view.

Keywords: digital device, fault simulation, parallel computing, scalability.

Введение

В процессе разработки цифровых устройств (ЦУ) важное место занимают методы моделирования с неисправностями. Выделяются два основных назначения таких методов. Во-первых, они служат для определения качества некоторой входной последовательности в соответствии с заданной метрикой. Во-вторых, они применяются в методах построения входных тестовых последовательностей различных классов, которые используют моделирование для оценки потенциальных решений [1]. Для современных проектируемых ЦУ в связи с большими списками анализируемых неисправностей процесс моделирования может продолжаться до нескольких десятков часов, что является наибольшим недостатком указанных методов. При этом согласно Международной Дорожной Карте [2] число транзисторов в SOC-системах потребительского уровня увеличится в 17 раз к 2024 году, а длина входных последовательностей для тестирования одиночных константных неисправностей увеличится с нынешних 30 тысяч входных векторов в 16 раз к указанному времени. Поэтому на сегодняшний день актуальной остаётся задача построения быстрых алгоритмов моделирования ЦУ с неисправностями.

В настоящее время наиболее перспективным является направление, связанной с разработкой параллельных версий алгоритмов моделирования ЦУ с неисправностями для параллельных ВС с различной аппаратной составляющей. Такие методы снижают стоимость процесса разработки современных ЦУ за счёт уменьшения времени работы соответствующих методов.

Поскольку основная цель методов параллельного моделирования – уменьшение времени работы, то все публикации по данному направлению содержат численные результаты с программной реализацией методов. Однако вне внимания почти всех авторов остаётся анализ масштабируемости предлагаемых методов. Такие оценки должны помочь сформулировать направления исследований по улучшению существующих, либо по разработке новых методов для современных многопроцессорных вычислительных систем (ВС).

Целью данной работы является построение аналитических оценок времени работы параллельных методов моделирования ЦУ, которые основаны на схеме с разбиением списка неисправностей.

Анализ методов параллельного моделирования СБИС с неисправностями

Разработка параллельных методов в диагностике ЦУ стала возможной с развитием параллельных ВС [3], причём первые работы относятся ещё к концу 80-х годов прошлого столетия [4]. Одной из наиболее значимых работ является [5], где предложена параллельная модификация алгоритма PROOFS, являющегося стандартом «де факто» для однопроцессорных систем. К настоящему времени выделились три основные схемы распараллеливания алгоритмов моделирования с неисправностями.

1) Разбиение схемы [6, 7]. В данном подходе схема разбивается на несколько подсхем, каждая из которых моделируется на отдельном узле вычислительной системы. Подход применяется как для

моделирования исправных схем, так и для моделирования с неисправностями. Основным преимуществом такого подхода является то, что резко уменьшаются требования к затратам памяти, в которой хранится описание схемы. Поэтому недостатком подхода является необходимость взаимодействия друг с другом отдельных задач, что требует написания протоколов такого взаимодействия, а также ведёт к сложности в реализации. Известны также работы в данном направлении отечественных авторов [8].

2) Разбиение теста [9] заключается в том, что входная последовательность T разбивается на ряд подпоследовательностей $T_1, T_2, ..., T_m$, с которыми и формируются подзадачи для вычислительных узлов. Процессы также должны взаимодействовать, чтобы информировать друг друга о том, какие неисправности обнаружены к текущему моменту времени. В случае если в заданной входной последовательности не известны такты времени, в которые начинаются подпоследовательности (что случается довольно часто), данный метод не может быть применён.

3) Разбиение списка неисправностей [3, 10, 11]. Здесь полный список неисправностей F разбивается на несколько подсписков $F_1, F_2, ..., F_n$, каждый из которых передаётся на отдельный процессор системы, где и выполняется его анализ путём моделирования на заданной входной последовательности. При такой схеме каждый узел вычислительной среды должен иметь свою копию описания схемы и теста. В настоящее время данный метод получил наибольшее распространение и считается, что он обладает хорошей масштабируемостью при росте числа процессоров. Именно данный подход будет центральным при рассмотрении в данной работе.

Самым существенным недостатком подхода с разбиением списка неисправностей является то, что активность неисправностей может быть неодинакова для подсписков. Это, в свою очередь, ведёт к разбалансировке нагрузки на узлы ВС. Поэтому разработан ряд реализаций основанных на динамическом разбиении списка неисправностей с целью уравновесить загрузку процессоров и уменьшить объём передаваемой информации [5, 12]. В [5] модель использует и статическое и динамическое разбиение списка неисправностей. Когда процессор завершает обработку списка неисправностей он посылает запрос другому процессору (выбранному случайно). Получивший запрос процессор разбивает свой список неисправностей и отдаёт его часть. Если же он уже закончил работу, то запрос передаётся далее случайному процессору. Процессоры завершают работу независимо. Для больших схем с относительно большой полнотой теста (s5378, s35932) из каталога ISCAS89 [13] ускорение составило 2.4–3.8 раза на 8-процессорной системе INTEL iPSC/.

Считается [14], что параллельное моделирование с разбиением списка неисправностей и параллелизмом тестовых наборов достаточно легко реализуемы и масштабируемыми. Также строятся подходы [15], которые объединяют два вышеназванных: сначала выполняется моделирование легкотестируемых неисправностей с разбиением списка, далее тяжелотестируемые неисправности моделируются на всех процессорах с разбиением теста.

В [4] для параллельной реализации CHIEFS ускорение составило от 2 до 6 раз на системе с 8-ю рабочими станциями Sun 3/280, соединёнными линиями связи 10 Мбит. В [11] ускорение составило 2–6 раз на гетерогенном кластере с 8-ю рабочими станциями Sun 3/10, Sun 3/60, кластере VAX200 и кластере VAXII/GPX. При этом производительность с динамической балансировкой неисправностей оказалась не существенно выше, чем статическое разбиение из-за накладных расходов на такую балансировку.

Существует и другое принципиальное ограничение для подходов со статической и динамической разбивкой неисправностей, связанное с тем, что наименьшее время работы клиента не может быть меньше времени моделирования исправного устройства. Такое моделирование необходимо выполнять для каждого из подсписков. Очевидно, что такое моделирование (более чем одном процессоре) является избыточным.

Иная ситуация при разработке метода для системы с общей памятью. В этом случае моделирование исправного устройства можно выполнить один раз. Но при этом остальные процессоры должны простаивать как минимум в первый такт времени для каждого входного набора. Это противоречие частично разрешено в [16]. Здесь параллельно с моделированием групп неисправностей выполняется моделирование поведения исправного устройства, но для следующего такта модельного времени. Таким образом, простой процессоров сокращается до времени моделирования поведения исправного ЦУ на первом входном наборе последовательности.

Ещё одно замечание, которое следует сделать относительно подхода с разбиением списка неисправностей, заключается в том, что большее ускорение получается для схем и последовательностей с более низким покрытием неисправностей [5, 12]. Это связано с тем, что тяжелотестируемые неисправности попадают на один или несколько процессоров.

Если рассматривать аппаратную составляющую методов, то вначале использовались специализированные параллельные ВС. Дальше развитие пошло в направлении параллельных кластеров общего назначения. Следующий этап характеризуется применением рабочих станций с многоядерными процессорами [17]. Первоначально они содержали 2 вычислительных ядра. Такие двухядерные процессоры стали основой целого ряда мощных многопроцессорных ВС. В настоящее время коммерчески доступными являются рабочие станции, в которых содержится 1–2 процессоров, каждый из которых содержит 6-8 вычислительных ядер. Алгоритм именно для таких рабочих станций предложен авторами в [16]. Между тем, практически отсутствуют исследования по эффективности параллельных методов моделирования для таких систем. Можно отметить публикацию [18], но здесь исследуется моделирование на системном уровне

представления. В отечественной литературе также данное направление представлено единичными публикациями [19]. Между тем очевидно, что такие системы обладают своими особенностями в сравнении с распределёнными ВС общего назначения, что должно учитываться при разработке новых методов.

В последнее время в связи с выходом графических ускорителей с большим числом графических процессоров (GPU) [20] оформилось ещё одно направление в параллельном моделировании, которое заключается в разработке алгоритмов моделирования для таких GPU. Алгоритмы данного типа предложены в [21–24]. Такие алгоритмы могут являться специальной версией для указанных GPU, либо использовать новые подходы и эвристики. Например, в [21] предлагается понятие макро-вентилей (macro-gates), моделирование которых на узлах производится в только случае их активации (изменения сигнала на его входах). Такой подход переносит понятие событийного моделирования на уровень макро-вентилей. В целом, данные алгоритмы по производительности существенно превосходят параллельные алгоритмы для остальных систем, в основном, за счёт очень большого числа параллельных процессоров.

Авторы в своё время предложили также ряд методов моделирования ЦУ с неисправностями как для вычислительного кластера с гомогенными узлами [25], так и для многоядерной ВС с общей памятью [16, 26]. Данные методы основаны на подходе с разбиением списка моделируемых неисправностей. Особенностями данных методов является то, что они комбинируют два уровня параллелизма. Верхний уровень назначает подсписок неисправностей на ядро/процессор ВС. На нижнем уровне неисправности моделируются параллельно по разрядам машинного слова. Таким образом, в отличие от известных методов сервер передаёт клиентам не одиночные неисправности для моделирования, а, как минимум, списки, размерность которых не ниже разрядности инструментальной ЭВМ.

В зависимости от конкретной задачи возможны два варианта реализации:

- исправное ЦУ моделируется в первом разряде машинного слова, остальные $n-1$ разрядов моделируют $n-1$ неисправных ЦУ, где n – число разрядов инструментальной ЭВМ;
- исправное ЦУ моделируется в одном машинном слове, в другом слове все n разрядов используются для моделирования неисправных ЦУ.

Именно данные модификации схемы с разбиением списка неисправностей выбраны для анализа в данной работе.

Несмотря на большое число подходов к построению параллельных версий методов моделирования ЦУ с неисправностями и разработанных на их основе методов, аналитические оценки параметров масштабируемости данных методов строятся в единичных случаях. Пожалуй, исключением является работа [10], в которой предложено пять модификаций синхронных и асинхронных методов моделирования, основанных на подходе с разбиением списка неисправностей, а также выполнены аналитические оценки масштабируемости.

В следующем разделе строятся аналитические оценки времени работы выбранных параллельных методов моделирования с неисправностями.

Оценки времени работы параллельных методов моделирования СБИС с неисправностями

Как уже было отмечено, формальная оценка ускорения работы параллельных методов моделирования ЦУ существенно затруднена и в работах проводится крайне редко. Основным предположением при построении таких оценок является то, что в начале процесса моделирования с неисправностями обнаруживается большее число неисправностей, тогда как в конце процесса моделирования новые неисправности детектируются реже. Обычно считают, что число обнаруженных неисправностей в единицу модельного времени уменьшается экспоненциально. Будем предполагать, что доля неисправностей, которая обнаруживается k-м набором последовательности S составляет $\alpha e^{-\lambda(k-1)}$ [10], где α – время моделирования первого входного набора, λ – индекс затухания. Часто также предполагается, что доля необнаруженных неисправностей после моделирования k-го набора составляет $\alpha_1 e^{-\lambda k}$.

Тогда можно определить долю неисправностей, которые обнаруживаются непосредственно набором с номером k:

$$\alpha_1 e^{-\lambda(k-1)} - \alpha_1 e^{-\lambda k} = \alpha_1 (1 - e^{-1}) e^{-\lambda(k-1)},$$

что также имеет вид $\alpha e^{-\lambda(k-1)}$.

Тогда часть неисправностей обнаруживаемых после моделирования $n-1$ вектора будет равна:

$$\sum_{k=1}^{n-1} \alpha e^{-\alpha(k-1)} = \alpha\left(\frac{1-e^{-\lambda(n-1)}}{1-e^{-\lambda}}\right) = \alpha r (1 - e^{-\lambda(n-1)}),$$

где $r = 1/(1-e^{-\lambda})$.

Тогда, если N_F – общее число моделируемых неисправностей, то число непроверенных неисправностей будет равно:

$$U(n) = N_F (1 - \alpha r (1 - e^{-\lambda(n-1)})).$$

Введём коэффициент стоимости моделирования одного логического элемента в единицах времени γ. Предположим, что δ - часть общего числа логических элементов $N_{эл}$, которые моделируются для одной неисправности. Тогда стоимость (время) моделирования неисправных ЦУ после n-го набора будет:

$$\gamma\delta N_{эл}U(n).$$

Предположим также, что β – часть логических элементов, необходимая для моделирования поведения исправного ЦУ. Для алгоритмов моделирования типа PROOFS [5] $\delta << \beta$ (моделируется только часть ЦУ, начиная от места неисправности, и только те узлы, которые имеют поведение отличное от исправного); для алгоритма типа [25] $\delta \approx \beta$ (отличием последнего – параллельное по разрядам моделирование 31-й неисправности). Тогда стоимость неисправного моделирования n векторов выражается как

$$\gamma(\beta N_{эл} + \delta N_{эл}U(n)).$$

Таким образом, общее время моделирования на однопроцессорной системе последовательности с L входными наборами составит:

$$T_1(L, N_F, N_{эл}) = \gamma(\sum_{t=1}^{L}(\beta N_{эл} + \delta N_{эл}U(n))) = \gamma\beta N_{эл}L + \gamma\delta N_{эл}N_F(L - \alpha r(L - r(1 - e^{-\lambda L}))).$$

Поскольку L достаточно велико, то можно положить $1 - e^{-\lambda L} \approx 1$. Тогда:

$$T_1(L, N_F, N_{эл}) = \gamma\beta N_{эл}L + \gamma\delta N_{эл}N_F(L - \alpha r(L - r)).$$

Пренебрегая r относительно L можно получить, что:

$$T_1(L, N_F, N_{эл}) = \gamma\beta N_{эл}L + \gamma\delta N_{эл}N_F L(1 - \alpha r)). \quad (1)$$

Таким образом, время моделирования на однопроцессорной ВС выражено суммой двух слагаемых, где первый компонент суммы показывает время моделирования исправного ЦУ, а второй – неисправных.

На основе формулы (1) построим оценку времени работы узлов ВС для параллельных версий метода моделирования ЦУ с неисправностями.

При параллельном моделировании на p процессорах с разбиением списка неисправностей каждый узел ВС должен выполнить моделирование поведения исправного ЦУ и N_F / p неисправных ЦУ. Тогда:

$$T_p(L, F, N_{эл}) = \gamma\beta N_{эл}L + \gamma\delta N_{эл}(N_F / p)L(1 - \alpha r)). \quad (2)$$

В таком виде оценка получена в [10]. Как и оценка (1) данная оценка построена для случая, когда выполняется раздельное моделирование исправного ЦУ и всех неисправных ЦУ. На основании данной оценки там же делается вывод о том, что подход с разбиением списка неисправностей не является масштабируемым (читай – достаточно масштабируемым) из-за первого слагаемого. Оно показывает, что моделирование исправного устройства необходимо выполнять на каждом из узлов параллельной ВС. Отметим, что данная оценка верна только для методов параллельного моделирования с неисправностями, которые на каждом узле применяют последовательное моделирование исправного и нескольких неисправных устройств.

Однако в предложенных авторами подходах применяется параллельное по разрядам машинного слова моделирование неисправностей [16, 25]. В этом случае оценка (2) является неверной, как и выводы из неё. Построим на основании (1) оценку метода параллельного моделирования ЦУ с неисправностями из [25], использующей указанный выше подход.

Однако сначала сделаем замечание о затратах, необходимых для передачи данных в параллельных методах моделирования. Оценка вида (2) достигается в идеальном случае, когда не учитывается скорость передачи данных в сети коммуникации, т.е. для паракомпьютера. Числовые данные в [26] показывает, что время передачи данных в таких методах может занимать существенное время: от 50% для небольших схем (s9234) до 8% для средних (s38471). Для точной оценки работы всего метода необходимо учитывать как время моделирования, так и время передачи данных. Пусть φ_1, φ_2 и φ_3 временные коэффициенты передачи информации об одном узле ЦУ, одной неисправности и одного входного набора теста соответственно. Учитывая, что на каждый узел ВС необходимо передать полное описание ЦУ и N_F / p неисправностей, получим:

$$T_p^{'}(L, N_F, N_{эл}) = N_{эл}(\gamma\beta L + \varphi_1) + (N_F / p)(\gamma\delta N_{эл}L(1 - \alpha r) + \varphi_2) + L\varphi_3. \quad (3)$$

Формула (3) оценивает время моделирования на одном узле параллельной ВС для метода моделирования ЦУ с неисправностями [25].

Поскольку передача теста занимает крайне небольшую часть времени обмена [26] то последним компонентом в (3) можно пренебречь. В [26] в таблице принято неудачное обозначение. Под заголовком «время передачи теста» следует понимать общее время передачи данных с момента начала обмена данными сервера с клиентами до окончания приём теста соответствующим клиентом. Тогда (3) можно упростить:

$$T_p^{'}(L, N_F, N_{эл}) = N_{эл}(\gamma\beta L + \varphi_1) + (N_F / p)(\gamma\delta N_{эл}L(1 - \alpha r) + \varphi_2). \quad (4)$$

С другой стороны, реализация метода была проведена на достаточно старом оборудовании, где

скорость связи между узлами кластера составляла 10 Мбит/сек. Более того, наибольшая схема из каталога ISCAS-89, для которой проводились эксперименты, является небольшой относительно сегодняшних практических дизайнов. В той же работе показано, что для паракомпьютера характеристики будут существенно лучше. Поэтому оценки в [26] являются очень консервативными. Следовательно, отброс фактора передачи данных для рассматриваемого идеального случая не должен приводить к существенной погрешности. Поэтому, с целью упрощения в дальнейшем, в данной работе мы его учитывать не будем, и базовой формулой будем считать (2), понимая, что при учёте времени пересылки данных необходимо использовать (3)-(4).

Формулы (2)–(4) показывают, что подход с разбиением списка неисправностей имеет ограничения по масштабированию при росте p из-за первого слагаемого. Такие же выводы были получены на основании машинных экспериментов в [26]. С другой стороны, при большом числе неисправностей N_F для больших ЦУ второй терм выражения становится определяющим, что в экспериментах выразилось в соответствующем росте коэффициента ускорения при росте размерности ЦУ – [26] табл.3, параметр H_8.

При адаптации метода для многоядерной ВС с общей памятью можно удалить процедуры пересылки описания ЦУ, теста и списка моделируемых неисправностей. Тогда (3)-(4) примут вид (2). В данной реализации моделирование исправного ЦУ по-прежнему выполняется в каждом узле ВС, поэтому замечания об ограничении масштабируемости справедливы и в данном случае.

Для метода [16] пересылка данных также отсутствует. Более того, при реализации наибольшее ускорение получено для конфигурации, когда один узел выполняет моделирование исправного ЦУ, а $p-1$ узел – моделирование неисправных ЦУ. Тогда (2) следует переписать, выделив узел, моделирующий исправное ЦУ:

$$\begin{gathered} T_p^1(L, N_F, N_{эл}) = \gamma\beta N_{эл} L; \\ T_p^i(L, N_F, N_{эл}) = \gamma\delta N_{эл}\left(N_F/(p-1)\right)L(1-\alpha r)), i = \overline{2, p}, \end{gathered} \tag{5}$$

где T_p^1 – время моделирования поведения исправного ЦУ в отдельном потоке, T_p^i – время моделирования группы неисправностей в i -м потоке.

Видно, что в T_p^i удалось избавиться от немасштибируемого слагаемого. И хотя для системы с p процессорами свойство масштабируемости будет ограничено подсистемой из $p-1$ процессора, эксперименты в [16] показали, что именно такая конфигурация является наиболее эффективной.

На основании оценок времени моделирования (2)–(5) можно получить характеристики параллелизации: ускорение работы, коэффициент загрузки процессоров и доля последовательного кода. Например, ускорение работы метода из [11] на системе с p ядрами равно:

$$S_p = \frac{\gamma\delta N_{эл}\left(N_F\right)L(1-\alpha r)) + \gamma\beta N_{эл} L}{\gamma\delta N_{эл}\left(N_F/(p-1)\right)L(1-\alpha r))}. \tag{6}$$

Здесь числитель показывает, что потоки моделирования исправного и групп неисправных ЦУ выполняются последовательно, поскольку имеется только одно вычислительное ядро. Знаменатель соответствует времени моделирования потока групп неисправностей из (5).

На основании полученных аналитических оценок можно также строить численные графики исследуемых зависимостей. На практике это возможно сделать для отдельных схем каталога. Обобщённый график построить затруднительно и он не будет носить презентативный характер ввиду сильного разброса численных значений параметров в (2)-(6). Поэтому для целей анализа, на наш взгляд, аналитические оценки в виде времени моделирования являются более предпочтительными.

Выводы

Выполнен обзор подходов построения параллельных методов моделирования ЦУ с неисправностями. На основании данного обзора для анализа выбрана схема с разбиением списка неисправностей. Для модификаций таких методов для аппаратных платформ с раздельной и общей памятью получены аналитические оценки времени моделирования в зависимости от числа вычислительных ядер. Это позволяет, в свою очередь, оценивать параметры масштабируемости разработанных методов.

Полученные аналитические зависимости времени работы алгоритмов позволяют строить графики его зависимости от числа процессоров, используемых для вычислений. На практике это реализовать сложно ввиду сильного разброса параметров, входящих в оценки. Преодоление этого противоречия является направлением дальнейших исследований.

Литература

1. Иванов Д.Е. Генетические алгоритмы построения входных идентифицирующих последовательностей цифровых устройств / Иванов Д.Е. – Донецк, 2012. – 240с.

2. ITRS 2010 technology roadmap (1.09.2013) http://www.itrs.net/Links/2010ITRS/Home2010.htm.

3. Mueller-Thuns R. B. VLSI Logic and Fault Simulation on General-Purpose Parallel Computers / R. B. Mueller-Thuns, D.G. Saab, R.F. Damiano, J.A. Abraham // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. – 1993. – № 12(3). – P. 446–460.

4. Duba P.A. Fault Simulation in a Distributed Environment / P.A. Duba, R.K. Roy, J.A. Abraham, W.A. Rogers // Proceedings of 25th Design Automation Conference. – 1988. – P. 686– 691.

5. Parker S. A parallel algorithm for fault simulation based on PROOFS / S. Parker, P. Banerjee, J. Patel // Proc. IEEE Int. Conf. Computer Design. – 1995. – P. 616–621.

6. Ghosh S. A distributed algorithm for fault simulation of combinatorial and asynchronous sequential digital designs, utilizing circuit partitioning, on loosely coupled parallel processors / S. Ghosh // Microelectronic Reliability.- 1995. – №35(6). – P.947-967.

7. Subbaraj P. Circuit Partitioning Problem using Graphical Processing Units / P. Subbaraj, P. Sivakumar, S. Nandhanam // Journal of Computer Science. – 2012. – №8(5). – P. 705–710.

8. Ладыженский, Ю.В. Программная система для исследования протоколов синхронизации при распределённом событийном логическом моделировании / Ю.В. Ладыженский, Ю.В. Попов // Наукові праці Донецького національного технічного університету, Серія "Обчислювальна техніка та автоматизація". – Донецьк: ДонНТУ. –2004. – № 74. – С. 201–209.

9. Ravikumar C.P. Distributed Fault Simulation Algorithms on Parallel Virtual Machine / C. P. Ravikumar, V. Jain, A. Dod // VLSI Design. – 2001. – Volume 12, Issue 1. – P. 81–99.

10. Krishnaswamy D. Asynchronous parallel algorithms for test set partitioned fault simulation / D. Krishnaswamy, P. Banerjee, E.M. Rudnick, J.H. Patel // ACM SIGSIM Simulation Digest. – 1997. – Volume 27, Issue 1. – P. 30–37.

11. Markas T. On distributed fault simulation / T. Markas, M. Royals, N. Kanopoulos // IEEE Computer. – 1990. – Vol. 7. – P.40–52.

12. Amin M.B. Data Parallel-Fault Simulation / M.B. Amin, B. Vinnakota // IEEE Trans. VLSI Systems. – 1999. – Vol. 7, № 2. – P.183–190.

13. Brgles F. Combinational profiles of sequential benchmark circuits / F. Brgles, D. Bryan, K. Kozminski // International symposium of circuits and systems, ISCAS-89. – 1989. – P. 1929–1934.

14. Han K. A Parallel Implementation of Fault Simulation on a Cluster of Workstations / K. Han, S.Y. Lee // Proc. IEEE International Symposium Parallel and Distributed Processing IPDPS. – 2008. – P. 1–8.

15. Rudnick E.M. Overcoming the serial logic simulation bottleneck in parallel fault simulation / E.M. Rudnick, J.H. Patel // Proc. 10th Int. Conf. VLSI Design. – 1997. – P. 495–501.

16. Иванов Д.Е. Параллельный алгоритм моделирования цифровых схем с неисправностями для многоядерных систем с общей памятью / Д.Е. Иванов // Электронное моделирование. – 2011. – Т. 33. – № 2. – С. 93–106.

17. Intel® Software Insight. Multi-core Capability / R. Wirt. – USA: Intel Corporation, 2005, July. – 11p.

18. Dömer R. Multi-core parallel simulation of system-level description languages / R. Dömer, W. Chen, X. Han // Proceedings of the 16th Asia and South Pacific Design Automation Conference, 2011. – P.311-316.

19. Hahanov V. Parallel Logic Simulation using Multi-Core Workstations / V. Hahanov, V. Obrizan, A. Gavryushenko, S. Mikhtonyuk // The Experience of Designing and Applications of CAD Systems in Microelectronics, 2007, CADSM '07, 9th International Conference. – 2007. – P. 256–257.

20. NVIDIA. CUDA Computer Unified Device Architecture, 2007.

21. Chatterjee D. High-performance gate-level simulation with GP-GPUs / D. Chatterjee, A. DeOrio, V. Bertacco // In Proceedings of DATE, 2009. – P. 1332–1337.

22. Gulati K. Towards acceleration of fault simulation using graphics processing units / K. Gulati, S. Khatri // Proceedings of the 45th annual Design Automation Conference, DAC '08. – 2008. – P. 822–827.

23. Perinkulam A. Logic simulation using graphics processors / A. Perinkulam, S. Kundu // In Proc. ITSW, 2007.

24. Kochte M.A. Efficient Fault Simulation on Many-Core Processors / M.A. Kochte, M. Schaal, H.-J. Wunderlich, C.G. Zoellin // Proceedings of the 47th Design Automation Conference ACM, New York, NY, USA. – 2010. – P. 380–385.

25. Иванов Д.Е. Распределённое параллельное моделирование цифровых схем с неисправностями / Д.Е. Иванов, Ю.А. Скобцов, А.И. Эль-Хатиб // Наукові праці Донецького національного технічного університету. Серія: "Обчислювальна техніка та автоматизація". – Донецьк : ДонНТУ. – 2006. – Вип. 107. – С. 128–134.

26. Иванов Д.Е. Методы параллельного моделирования СБИС с неисправностями / Д.Е. Иванов // «Радіоелектронні і комп'ютерні системи», 2012. – № 5. – С. 114–119.

27. Иванов Д.Е. Исследование характеристик масштабируемости метода моделирования ЦУ с неисправностями для распределённых ВС / Д.Е. Иванов // Проблемы информационных технологий. – 2013. – № 1 (13). – С. 69–77.

References

1. D.E. Ivanov, Geneticheskie algoritmy postroeniya vxodnyx identificiruyushhix posledovatel'nostej cifrovyx ustrojstv, Doneck: IAMM NASU, 2012.

2. ITRS 2010 technology roadmap (1.09.2013) http://www.itrs.net/Links/2010ITRS/Home2010.htm.

3. R. B. Mueller-Thuns, D.G. Saab, R.F. Damiano and J.A. Abraham, "VLSI Logic and Fault Simulation on General-Purpose Parallel Computers", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, №12(3), 1993, pp. 446-460.

4. P.A. Duba, R.K. Roy, J.A. Abraham and W.A. Rogers, "Fault Simulation in a Distributed Environment", Proceedings of 25th Design Automation Conference, 1988, pp. .686- 691.

5. S. Parker, P. Banerjee and J. Patel, "A parallel algorithm for fault simulation based on PROOFS", Proc. IEEE Int. Conf. Computer Design, 1995, pp. 616-621.

6. S. Ghosh, "A distributed algorithm for fault simulation of combinatorial and asynchronous sequential digital designs, utilizing circuit partitioning, on loosely coupled parallel processors". Microelectronic Reliability, 1995, №35(6), pp. 947-967.

7. P. Subbaraj, P. Sivakumar and S. Nandhanam, "Circuit Partitioning Problem using Graphical Processing Units", Journal of Computer Science, 2012, №8(5), pp. 705-710.

8. Yu.V. Ladyzhenskij and Yu.V. Popov, "Programmnaya sistema dlya issledovaniya protokolov sinxronizacii pri raspredelyonnom sobytijnom logicheskom modelirovanii", Naukovi pratsi Donetskoho natsionalnoho tekhnichnoho universytetu, Seriia "Obchysliuvalna tekhnika ta avtomatyzatsiia", Donetsk: DonNTU, 2004, №74, pp. 201-209.

9. C. P. Ravikumar, V. Jain and A. Dod, "Distributed Fault Simulation Algorithms on Parallel Virtual Machine", VLSI Design, 2001, Volume 12, Issue 1, pp. 81-99.

10. D. Krishnaswamy, P. Banerjee, E.M. Rudnick abd J.H. Patel, "Asynchronous parallel algorithms for test set partitioned fault simulation", ACM SIGSIM Simulation Digest, 1997, Volume 27, Issue 1, pp. 30-37.

11. T. Markas, M. Royals and N. Kanopoulos, "On distributed fault simulation", IEEE Computer, 1990, Vol.7, pp. 40-52.

12. M.B. Amin and B. Vinnakota, "Data Parallel-Fault Simulation", IEEE Trans. VLSI Systems, 1999, Vol.7, №2, pp. 183–190.

13. F. Brgles, D. Bryan and K. Kozminski, "Combinational profiles of sequential benchmark circuits", International symposium of circuits and systems, ISCAS-89, 1989, pp. 1929-1934.

14. K. Han and S.Y. Lee, "A Parallel Implementation of Fault Simulation on a Cluster of Workstations", Proc. IEEE International Symposium Parallel and Distributed Processing (IPDPS), 2008, pp. 1–8.

15. E.M. Rudnick and J.H. Patel, "Overcoming the serial logic simulation bottleneck in parallel fault simulation", Proc. 10th Int. Conf. VLSI Design, 1997, pp. 495–501.

16. D.E. Ivanov, "Parallel'nyj algoritm modelirovaniya cifrovyx sxem s neispravnostyami dlya mnogoyadernyx sistem s obshhej pamyat'yu", E'lektronnoe modelirovanie, 2011, Vol.33, №2.- pp. 93-106.

17. R. Wirt, Intel® Software Insight. Multi-core Capability, USA: Intel Corporation, 2005, July.

18. R. Dömer, W. Chen and X. Han, "Multi-core parallel simulation of system-level description languages", Proceedings of the 16th Asia and South Pacific Design Automation Conference, 2011, pp. 311-316.

19. V. Hahanov, V. Obrizan, A. Gavryushenko and S. Mikhtonyuk, "Parallel Logic Simulation using Multi-Core Workstations", The Experience of Designing and Applications of CAD Systems in Microelectronics (CADSM '07), 2007, pp. 256-257.

20. NVIDIA. CUDA Computer Unified Device Architecture, 2007.

21. D. Chatterjee, A. DeOrio and V. Bertacco, "High-performance gate-level simulation with GP-GPUs", In Proceedings of DATE, 2009, pp. 1332-1337.

22. K. Gulati and S. Khatri, "Towards acceleration of fault simulation using graphics processing units", Proceedings of the 45th annual Design Automation Conference (DAC'08), 2008, pp. 822-827.

23. Perinkulam and S. Kundu, "Logic simulation using graphics processors", In Proc. ITSW, 2007.

24. M.A. Kochte, M. Schaal, H.-J. Wunderlich and C.G. Zoellin, "Efficient Fault Simulation on Many-Core Processors", Proceedings of the 47th Design Automation Conference ACM, New York, NY, USA, 2010, pp. 380-385.

25. D.E. Ivanov, Yu.A. Skobcov and A.I. E'l'-Xatib, "Raspredelyonnoe parallel'noe modelirovanie cifrovyx sxem s neispravnostyami", Naukovi praci Donec'kogo nacional'nogo texnichnogo universitetu. Seriya: "Obchislyuval'na texnika ta avtomatizaciya, Vipusk 107, Donec'k:DonNTU, 2006, pp. 128-134.

26. D.E. Ivanov, "Metody parallel'nogo modelirovaniya SBIS s neispravnostyami", Radioelektronni i komp'yuterni sistemi, 2012, №5, pp. 114-119.

27. D.E. Ivanov, "Issledovanie xarakteristik masshtabiruemosti metoda modelirovaniya CU s neispravnostyami dlya raspredelyonnyx VS", Problemy informacionnyx texnologij, 2013, №1(13), pp. 69-77.

Рецензія/Peer review : 30.4.2014 р. Надрукована/Printed : 18.5.2014 р.
Рецензент: Скобєлєв В.Г., д.ф.-м.н., д.т.н., проф., п.н.с. відділу теорії керуючих систем ІПММ НАН України