СИСТЕМА РАСПРЕДЕЛЕННОГО ЛОГИЧЕСКОГО МОДЕЛИРОВАНИЯ ЦИФРОВЫХ УСТРОЙСТВ С ИСПОЛЬЗОВАНИЕМ КОНСЕРВАТИВНОГО ПРОТОКОЛА СИНХРОНИЗАЦИИ

Ладыженский Ю.В., Попов Ю.В.

Кафедра ПМиИ, ДонНТУ
ly@cs.dgtu.donetsk.ua

Abstract

Ladyzhensky Y.V., Popoff Y.V. System for distributed logical simulation of digital devices with conservative synchronization protocol. *Analysis of synchronization protocols for logic simulations is fulfilled. Algorithms for experimental research of parallel simulations are developed. A system architecture is proposed. Software was implemented and tested in a local network.*

Введение

Моделирование поведения во времени логических схем является важным этапом создания цифровых устройств. Высокая размерность и сложность реальных логических схем приводит к большим затратам времени и памяти при моделировании. Ускорение моделирования и ресурсы памяти можно получить при использовании нескольких процессоров, работающих параллельно. Для увеличения скорости моделирования следует использовать внутренний параллелизм реальных схем [1,2].

1. Последовательное событийное моделирование

Функционирование всех дискретных логических схем представляется наборами состояний и событий. Текущие значения сигналов на узлах схемы представляют собой состояние схемы, а изменения сигналов во времени – события. Выполнение моделирования означает обработку событий, которые появляются в определенном порядке во времени и изменение состояний, соответствующее этим событиям. Для изменения состояний в будущем должны быть запланированы новые события.

Моделирование выполняется в дискретном виртуальном времени VT. Один тик VT соответствует одному такту работы схемы. Моделирование начинается в момент виртуального времени 0 и заканчивается в конечный момент VT или при отсутствии событий в схеме. Каждому событию соответствует значение VT его наступления.

Существует очередь (список) событий EVL. Перед началом моделирования она заполняется событиями на входах схемы. События из EVL выбираются в порядке увеличения VT их наступления. Во время моделирования могут возникнуть события, запланированные на момент VT в будущем. Такие события вносятся в EVL и ожидают своего момента наступления. Когда в EVL более нет событий, запланированных на текущий момент VT, выбирается событие с наименьшим VT наступления. При этом VT увеличивается до значения VT наступления этого события.

2. Параллельное и распределенное событийное моделирование логических схем

В процессе моделирования несколько событий могут быть запланированы на одно и то же время. Эта ситуация возникает каждый раз, когда два и более событий могут появиться в одно и то же время и не являются взаимоисключающими или действительно появляются в физической логической схеме. Эти события можно моделировать параллельно.

Параллельную обработку событий можно выполнять на одном или на нескольких моделирующих процессорах (МодПр). Мы будем рассматривать распределенное моделирование, когда параллельная обработка событий производится на нескольких МодПр, соединенных между собой в сеть. При этом исходная схема разрезается на части и каждому МодПр назначается своя часть схемы и своя локальная очередь событий LEVL.

На рис. 1а приведен пример комбинационной схемы. Сигналы подаются на входы X1 и X2; !X1 и !!X1 – внутренние узлы схемы; Res – выход схемы. На рис. 1б, 1в изображена эта же схема после разрезания на две части. Первому МодПр назначаются элементы D1, D2 и вход X1 исходной схемы; второму МодПр – элемент D3 и вход X2 исходной схемы. При этом на схеме, назначенной первому МодПр, появляется выход i1@Proc2, сигналы с которого подаются на вход i1 схемы на втором МодПр.

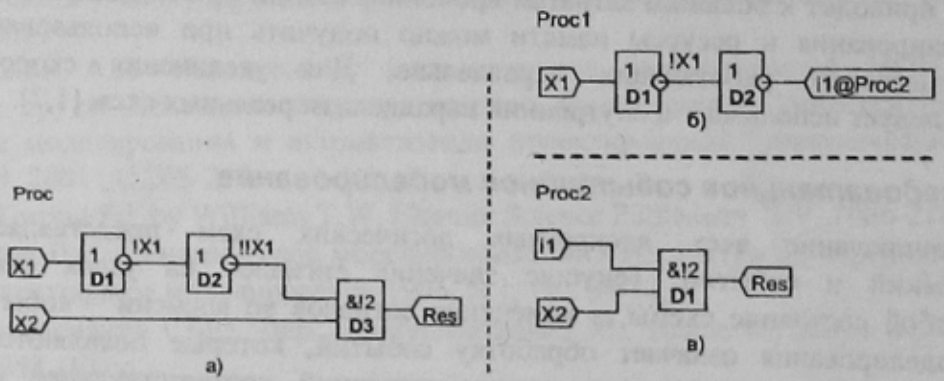

Рисунок 1 – Пример комбинационной схемы. а) схема для моделирования на одном МодПр Proc; б), в) схемы при моделировании на двух МодПр Proc1 и Proc2

При моделировании на нескольких МодПр возникают следующие типы событий:

- входные события подаются на глобальные входы схемы (X1 и X2 на рис. 1);
- внутренние события действуют только на часть схемы, назначенную МодПр, на котором они возникли. Например, если задержка на элементе D1 (рис. 1) равна одному такту, то изменение сигнала X1 на входе элемента D1 в момент VT=2 вызовет появление внутреннего события на входе элемента D2, которое будет запланировано на момент VT=3;
- внешние события возникают на одной из частей схемы и действуют на другую часть схемы (назначенную удаленному МодПр). На рис. 1б,в такие события возникают на выходе i1@Proc2 процессора Proc1 и подаются на вход i1 процессора Proc2. При этом по отношению к процессору Proc1 эти события называются экспортируемыми, а по отношению к процессору Proc2 – импортируемыми. Появление таких событий

вызывает отправку процессором Proc1 процессору Proc2 сообщения об этом событии;
- результирующие события возникают на глобальных выходах схемы. На рис. 1 - это выход Res.

При моделировании используются глобальные часы, которые ведут отсчет глобального виртуального времени GVT. Каждый процессор использует часы, которые показывают локальное виртуальное время LVT. При этом:
- значение LVT на каждом МодПр может представлять собой копию GVT. В таком случае имеет место синхронное моделирование. Если на процессоре нет событий, назначенных на текущий момент LVT, он будет ожидать его увеличения другими МодПр;
- значение LVT может быть различным на каждом МодПр. В этом случае имеет место асинхронное моделирование. В асинхронном моделировании, если на процессоре заканчиваются события, назначенные на текущий момент LVT, оно увеличивается и процессор обрабатывает следующие события. Потенциальное уменьшение ожиданий событий от удаленных процессоров может привести к увеличению скорости моделирования по сравнению с синхронным. В дальнейшем мы будем рассматривать только асинхронное моделирование.

Для того, что бы моделирование было правильным, события должны обрабатываться только в порядке увеличения VT их наступления. Поскольку при асинхронном моделировании значения LVT на каждом из МодПр различны, возможна ситуация, когда на одном из МодПр возникнет экспортируемое событие, назначенное на момент VT, меньшее, чем значение LVT на процессоре – приемнике. Следовательно, у процессора – приемника появляется событие «в прошлом». При этом порядок обработки событий нарушается. Есть две стратегии увеличения LVT, которые позволяют избежать появления таких ошибок:

- МодПр не увеличивает значение LVT до тех пор, пока он не будет уверен в том, что события в прошлом не произойдут. Эта стратегия моделирования применяется в консервативных протоколах синхронизации моделирования [3,4];

- МодПр увеличивает значение LVT каждый раз, когда в его LEVL более нет событий, запланированных на текущий момент LVT. В таком случае МодПр запоминает все изменения, которые происходят в схеме, и при появлении события «в прошлом» производит откат изменений и уменьшение LVT. В результате отката модель возвращается в прошлое состояние, соответствующее моменту наступления этого события. Такая стратегия моделирования применяется в оптимистическом протоколе синхронизации моделирования.

3. Консервативный протокол синхронизации процессов параллельного логического моделирования

Архитектура взаимодействия процессов моделирования при использовании консервативного протокола показана на рис. 2.

Консервативный логический процесс (ЛП) моделирования включает следующие основные части:
- структурно-функциональная модель схемы (СФМС) содержит описание структуры и функций моделируемой схемы (Rk – часть моделируемой схемы на процессоре k), текущие значения сигналов (Sk –переменные состояния) в ее узлах;

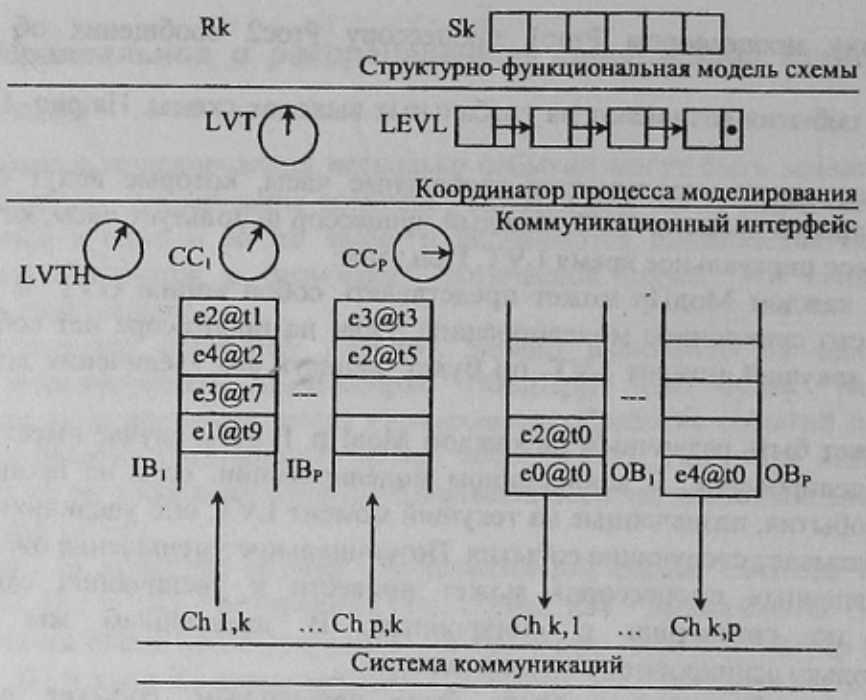

Рисунок 2 – Архитектура k-го процесса моделирования

- координатор процесса моделирования (КПМ) выбирает события из LEVL в соответствии с моделируемой частью схемы и текущими значениями сигналов в узлах схемы, вносит изменения в Sk, продвигает LVT;
- коммуникационный интерфейс (КИ) на входной стороне содержит входной буфер IBi и канальные часы CCi для каждого канала связи Ch i,k (рис. 2). IBi временно хранит входящие сообщения в порядке FIFO. CCi содержит копию времени первого сообщения в буфере IBi. Первоначально CCi=0. Граница увеличения локального виртуального времени $LVTH = \min_i(CCi)$ – время, до которого можно увеличить LVT путем обработки внутренних и экспортированных событий. На выходной стороне КИ имеются выходные буферы OBj для каждого канала связи Ch k,j;
- система коммуникаций используется для передачи сообщений по направленным каналам связи Ch i,j от i-го процессора к j-му.

Если события отправляются в порядке возрастания VT их появления, то внешнее событие с отметкой времени меньше, чем LVTH прийти не может. КИ в таком случае разрешает КПМ провести моделирование событий с VT наступления меньше LVTH. При обработке этих событий КПМ может сгенерировать экспортируемые события для удаленных ЛП. Эти сообщения помещаются в выходные FIFO буферы OBi и отправляются другим ЛП по системе коммуникаций.

После того, как все события с VT наступления меньше LVTH обработаны, не могут произойти внутренние события и не могут быть импортированы внешние события со временем наступления меньше LVTH. В таком случае КПМ останавливает обработку событий и ожидает получения сообщения о новом «безопасном» событии в будущем, которое увеличит LVTH.

4. Сравнение параллельного и последовательного логического моделирования

Основные различия параллельного и последовательного моделирования состоят в следующем:

Свойство	**Последовательное моделирование**	**Распределенное моделирование**
Порядок выборки событий из очереди	События выбираются по одному в порядке наступления.	Одновременно может быть выбрано несколько событий. Общий порядок обработки событий должен соответствовать порядку обработки при последовательном моделировании.
Синхронизация процессоров	Отсутствуют затраты времени на синхронизацию.	Требуется синхронизация для обеспечения правильного порядка обработки событий.
Глобальное виртуальное время	Глобальное виртуальное время равно локальному виртуальному времени.	Требуется вычисление GVT. Централизованное вычисление GVT ведет к перегрузкам системы связи, распределенное вычисление GVT ведет к дополнительным затратам времени у МодПр.
Переменные состояния	Можно использовать относительно много переменных состояния.	Неконсервативные алгоритмы распределенного моделирования работают лучше с небольшим числом переменных.
Память	Простое управление памятью, затраты памяти меньше.	Требуются дополнительное управление и ресурсы памяти.
Система коммуникаций	Отсутствует.	Необходима синхронизация процессоров. Затраты времени на коммуникацию могут снизить скорость моделирования.
Реализация	Структуры и обработка данных проще.	Сложнее реализовать и отладить. Простые структуры данных, но сложнее обработка данных.
Быстродействие	Высокое быстродействие при моделировании небольших схем; низкое быстродействие при моделировании больших схем.	Зависит от протокола синхронизации процессов моделирования, от стратегии управления памятью, алгоритма вычисления GVT, от быстродействия системы коммуникации.

5. Использование маркера в консервативном протоколе

Использование блокировки моделирования до прихода безопасного события ведет к появлению тупиков и переполнению памяти. Если один ЛП ожидает события от уже заблокированного ЛП, то возникает тупик. ЛП, попавшие в тупик, оставляют

события в своих входных буферах необработанными. Это ведет к переполнению памяти. Известно несколько методов предотвращения, определения и восстановления системы из тупика [3]. Один из методов состоит в использовании маркера.

Маркер – это сообщение, которое циркулирует между всеми ЛП. Путь следования маркера определяется до начала моделирования и каждому ЛП сообщается о том, кому он должен передать маркер.

Работа маркера состоит в следующем. Каждый ЛП условно окрашивается в красный или белый цвет. В белый цвет окрашиваются ЛП перед началом моделирования и после получения маркера. В красный цвет окрашиваются ЛП, которые получили или отправили сообщения о событии.

Ожидается, что маркер обойдет все ЛП за конечное время. Маркер несет в себе информацию о том, сколько белых ЛП, он посетил. Если количество белых ЛП, которые посетил маркер, равняется количеству ЛП в системе, фиксируется возникновение тупика.

Маркер также несет в себе информацию о времени и месте наступления следующего события в системе. На очередном процессоре сравнивается информация о времени наступления следующего события в системе из маркера со временем наступления следующего события на данном процессоре. Если время наступления события на данном процессоре меньше, чем время, записанное в маркере, информация в маркере обновляется.

6. Разработка системы параллельного логического моделирования

Система моделирования дискретных логических схем, написана на языке программирования Delphi 5 с использованием компонентов DirectX для Delphi [5,6]. Система моделирования состоит из программных МодПр, которые выполняются на компьютерах, соединенных в сеть. Каждому МодПр присваивается имя и назначается своя часть схемы для моделирования. Всему сеансу моделирования назначается имя сессии.

Программная система имеет следующие основные особенности:

1) ввод и редактирование логических схем. Для этих целей система содержит графический редактор схем. СФМС задает функции элементов с помощью таблиц истинности, других схем и в виде динамически загружаемых библиотек;
2) моделирование схемы выполняется в 6-значном алфавите (истина, ложь, возрастающий фронт, убывающий фронт, состояние высокого импеданса и неопределенное состояние), что обеспечивает высокую адекватность моделей;
3) система позволяет проводить асинхронное логическое моделирование на нескольких компьютерах. При моделировании используется консервативный протокол синхронизации с маркером. Во время моделирования на каждом МодПр ведется журнал моделирования;
4) сервер отчетов ведет журналы моделирования и собирает их с целью последующего анализа. Сервер отчетов позволяет:
   - выводить диаграммы сигналов в любом узле схемы;
   - устанавливать причинно-следственные связи между записями при просмотре;
   - выводить статистическую информацию о ходе моделирования: количество отправленных/принятых сообщений (по каждому типу и в сумме); количество обработанных событий, среднее время ожидания событий в очереди, среднее

время задержки маркера на процессоре, время обработки схемы, время ожидания. Характеристики выводятся отдельно по каждому процессору и в сумме (в среднем) по всем процессорам.

7. Структура программного моделирующего процессора

Структурная схема МодПр приведена на рис. 3:

- координатор потока исходящих сообщений (КПИС) форматирует сообщения и отправляет их в сеть через интерфейс DirectPlay другим процессорам;
- координатор потока входящих сообщений (КПВС) распознает входные сообщения, передает их на обработку соответствующим модулям МодПр;
- коммуникационный интерфейс (КИ) контролирует порядок передачи экспортируемых событий, отслеживает LVT на каждом удаленном процессоре анализируя импортируемые события, определяет LVTH;
- структурно-функциональная модель схемы (СФМС) хранит структуру, функции и текущие значения сигналов в узлах моделируемой схемы.
- координатор процесса моделирования (КПМ) выбирает события из LEVL и подает на СФМС новые значения в узлах схемы, на которые действуют эти события. Продвигает LVT;
- локальный список событий LEVL хранит события, запланированные на данном МодПр. Все импортированные и внутренние события записываются в LEVL перед тем, как они будут выбраны КПМ для обработки;

Рисунок 3 – Структурная схема моделирующего процессора

8. Функционирование моделирующего процессора

КПМ при переходе в режим моделирования выполняет следующие действия:

1) сбрасывает LVT в 0 очищает LEVL;
2) подготавливает к моделированию СФМС: значения во всех узлах схемы устанавливаются в "не определено";
3) события, которые запланированы на входах схемы на данном процессоре, записываетс в LEVL;
4) создает новый файл с журналом моделирования.

В процессе моделирования:

1) из КИ на КПМ направляется информация об LVTH;
2) на основе значений LVTH и LVT КПМ определяет время, до которого события могут быть безопасно обработаны, и начинает последовательно выбирать из LEVL события до данного момента времени;
3) если LVTH>LVT, и следующее событие LEVL запланировано на момент $LVT < t \le LVTH$, то LVT увеличивается до момента t. Теперь известно, что новых событий, запланированных ранее момента LVT более не возникнет, и КПМ позволяет КИ отправить другим МодПр сообщения о событиях, запланированных до момента LVT из своих выходных буферов;
4) новые значения в узлах схемы, определенные по выбранным событиям, подаются на СФМС. КПМ содержит список элементов, на входах которых изменились значения сигналов;
5) если стало известно, что на входах какого-либо элемента значения далее не изменятся, КПМ разрешает обновление значений на выходах этого элемента;
6) вследствие обновлений на выходах элементов возникают новые события:
   - внутренние события, если сигнал с выхода данного элемента подается на вход другого, который моделируется на этом же МодПр. События на этих входах записываются в LEVL;
   - экспортируемые события, если сигнал с выхода локальной части схемы подается на вход части схемы, которая моделируется на удаленном МодПр. События с этих входов подаются на КИ;
   - результирующие события, сигналы на глобальных выходах схемы, которые не действуют на входы других элементов и просто запоминаются в жарнале.

КПВС управляет следующими типами сообщений:

- управляющие сообщения передаются на КПМ;
- для сообщения - маркера выполняется обработка его информации, маркер заполняется новой информацией и передается на КПИС;
- информация из сообщений о событиях передается в КИ.

По приходу сообщения-маркера КПВС выбирает из LEVL информацию о VT наступления следующего события на данном МодПр. Если время наступления следующего события на данном процессоре меньше, чем время наступления следующего события, записанное в маркере, информация в маркере обновляется. При определении тупика:

- если в системе более нет ни одного события, то моделирование прекращается;
- если в системе есть хотя бы одно необработанное событие, то в маркере записано время и имя МодПр, на котором произойдет следующее событие. В таком случае этому МодПр отправляется сообщение "увеличить LVT";
- если сообщение "увеличить LVT" было отправлено локально, т.е. следующее событие в системе должно наступить на процессоре, на котором сейчас находится

маркер, то маркер задерживается на этом процессоре до тех пор, пока на данном процессоре есть события, которые могут быть безопасно обработаны. Затем информация в маркере обновляется и маркер отправляется следующему МодПр;

Коммуникационный интерфейс выполняет следующие функции:

- по приходу импортируемого события обновляет значения канальных часов для соответствующего входного буфера и отправляет это событие в LEVL;
- по приходу экспортируемого события помещает его в выходной буфер;
- по команде КПМ предает события из выходных буферов на КПИС;
- по значениям канальных часов всех входных буферов определяет LVTH и передает его на КПМ.

На сервер отчетов поступают данные о процессе моделирования:

а) из LEVL передается информация о внесенных и выбранных из него событиях;
б) из КПВС передается информация о сообщениях, принятых от других МодПр;
в) из КПИС передается информация о сообщениях, отправленных другим МодПр;
г) из КПМ передается информация о продвижении LVT;
д) информация об ошибках, которые могли возникнуть при моделировании.

Заключение

Ускорение логического моделирования сверхбольших интегральных схем может быть достигнуто путем распараллеливания вычислений. Проведенный сравнительный анализ различных протоколов синхронизации процессов вычислений показал их существенное влияние на качество и эффективность моделирования.

Для экспериментального исследования параллельного моделирования разработаны алгоритмы, предложена архитектура и выполнена реализация программной системы. Программная система обеспечивает распределенное асинхронное моделирование дискретных логических схем. Проведена апробация программной системы для дискретных схем на компьютерах, соединенных локальной сетью, подтвердившая ее работоспособность.

Перспективным направлением дальнейших исследований является оценка влияния размерности моделируемых схем, способов управления обменами сообщений и временем моделирования на скорость, адекватность моделирования и загрузку вычислительных ресурсов.

Перечень источников

1. Ладыженский Ю.В. Исследование цифровых систем с программируемой логикой методами моделирования. В кн.: Наукоемкие технологии образования: Труды IX Международной научно-методической конференции. Таганрог, ТРТУ, 1999, С. 59.

2. Bashkov E.A., Ladyzhensky Y.V. Study by research in improving of EDA tools teaching in a technical university. 2nd Global Congress of Engineering Education. Wismar, Germany. Congress Proceedings, UICEE 2000, p. 462-464.

3. Ferscha Alois. Parallel and Distributed Simulation of Discrete Event Systems. In Hardbound of Parallel and Distributed Computing. McGraw-Hill, 1995.

4. Chandy K. M., Misra J. Asynchronous Distributed Simulation via a Sequence of Parallel Computations. Communications of the ACM, 24(11): 198-206, November, 1981.

5. Hori H. Компоненты DirectX для Delphi 3, 4, 5 от 24.10.1999. http://www.ingjapan.ne.jp/hori/.

6. DirectX 7.0 Programmer's reference. Microsoft Corporation, 1995-1999.

Поступила в редакційну колегію 10. 11.2001 р.