РОССИЙСКАЯ ФЕДЕРАЦИЯ

(19) **RU** (11) **2 308 754** (13) **C2**

(51) МПК
G06F 9/46 (2006.01)

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ,
ПАТЕНТАМ И ТОВАРНЫМ ЗНАКАМ

(12) ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ПАТЕНТУ

(21), (22) Заявка: **2004123622/09, 11.12.2002**

(24) Дата начала отсчета срока действия патента:
11.12.2002

(30) Конвенционный приоритет:
31.12.2001 US 10/039,579

(43) Дата публикации заявки: **27.03.2005**

(45) Опубликовано: **20.10.2007 Бюл. № 29**

(56) Список документов, цитированных в отчете о поиске: **CRAIG ZILLES. Time-shifted modules: Exploiting code modularity for fine grain parallelization. Технический отчет, Университет Висконси, октябрь 2001, [http: www.cs.wisc.edu/techreports/2001/TR1430.pdf], с.1, 4, 8-10. US 5530597 A, 25.07.1996. RU 2042193 C1, 20.08.1995. ГРИГОРЬЕВ В.Л. Микропроцессор i486. Архитектура и программирование. - М.: Гранал, 1993, книга 1, с.214.**

(85) Дата перевода заявки PCT на национальную фазу: **02.08.2004**

(86) Заявка PCT:
US 02/39786 (11.12.2002)

(87) Публикация PCT:
WO 03/058447 (17.07.2003)

Адрес для переписки:
129010, Москва, ул. Б. Спасская, 25, стр.3, ООО "Юридическая фирма Городисский и Партнеры", пат.пов. Ю.Д.Кузнецову, рег.№ 595

(72) Автор(ы):
МАРР Дебора (US),
РОДЖЕРС Скотт (US),
ХИЛЛ Дэвид (US),
КАУШИК Шивнандан (US),
КРОССЛЭНД Джеймс (US),
КАУФЭЙТИ Дэвид (US)

(73) Патентообладатель(и):
ИНТЕЛ КОРПОРЕЙШН (US)

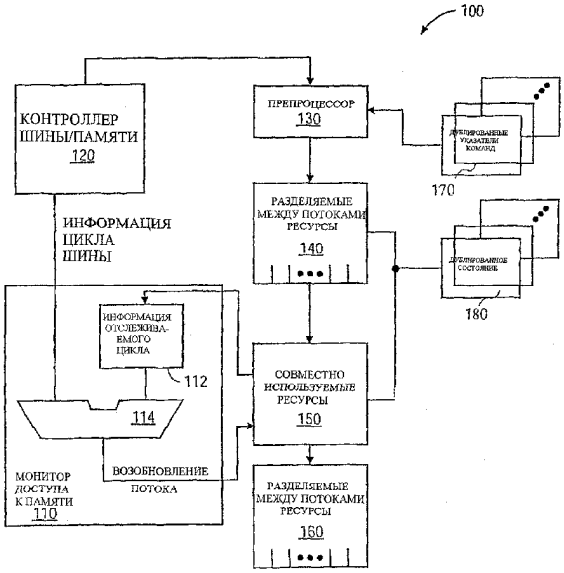
(54) СПОСОБ И УСТРОЙСТВО ДЛЯ ПРИОСТАНОВКИ ИСПОЛНЕНИЯ ПОТОКА ДО МОМЕНТА ОСУЩЕСТВЛЕНИЯ ОПРЕДЕЛЕННОГО ДОСТУПА К ПАМЯТИ

(57) Реферат:

Изобретение относится к средствам для приостановки исполнения потока до тех пор, пока не произойдет определенный доступ к памяти. Техническим результатом является повышение производительности процессора. В одном варианте осуществления процессор содержит множество исполняющих устройств, способных исполнять множество потоков. Первый поток включает в себя команду, которая определяет отслеживаемый адрес. Логические средства приостановки приостанавливают выполнение первого потока, а монитор вызывает возобновление первого потока в качестве реакции на доступ по заданному отслеживаемому адресу. 5 н. и 40 з.п. ф-лы, 13 ил.

RU 2308754 C2

RU 2308754 C2



ФИГ.1

RU 2308754 C2

RU 2308754 C2

RUSSIAN FEDERATION

FEDERAL SERVICE
FOR INTELLECTUAL PROPERTY,
PATENTS AND TRADEMARKS

(19) **RU** (11) **2 308 754** (13) **C2**

(51) Int. Cl.
G06F 9/46 (2006.01)

(12) ABSTRACT OF INVENTION

(21), (22) Application: **2004123622/09, 11.12.2002**

(24) Effective date for property rights: **11.12.2002**

(30) Priority:
31.12.2001 US 10/039,579

(43) Application published: **27.03.2005**

(45) Date of publication: **20.10.2007 Bull. 29**

(85) Commencement of national phase: **02.08.2004**

(86) PCT application:
US 02/39786 (11.12.2002)

(87) PCT publication:
WO 03/058447 (17.07.2003)

Mail address:
129010, Moskva, ul. B. Spasskaja, 25, str.3, OOO "Juridicheskaja firma Gorodisskij i Partnery", pat.pov. Ju.D.Kuznetsovu, reg.№ 595

(72) Inventor(s):
MARR Debora (US),
RODZhERS Skott (US),
KhILL Dehvid (US),
KAUShIK Shivnandan (US),
KROSSLEhND Dzhejms (US),
KAUFEhJTI Dehvid (US)

(73) Proprietor(s):
INTEL KORPOREJShN (US)

(54) METHOD AND DEVICE FOR PAUSING EXECUTION OF A STREAM UNTIL A CERTAIN MEMORY ACCESS IS PERFORMED

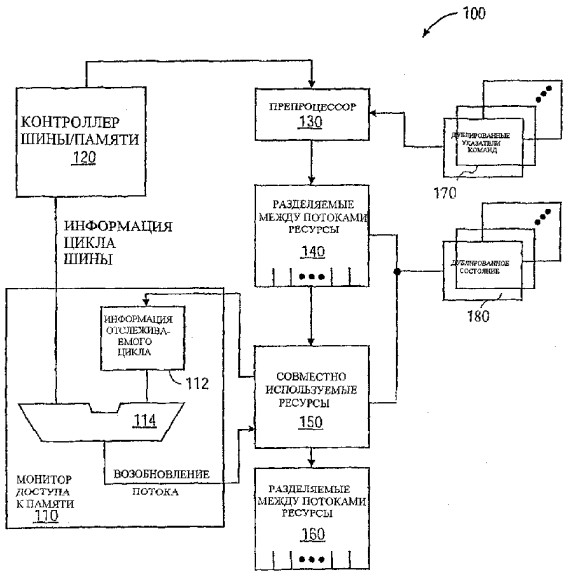
(57) Abstract:

FIELD: engineering of means for pausing execution of a stream until certain memory access occurs.

SUBSTANCE: in one variant of realization, processor contains a set of executive devices, capable of executing a set of streams. First stream includes a command, which determines the address being tracked. Logical pausing means pause execution of first stream, and monitor causes renewal of first flow as reaction to access of given address being tracked.

EFFECT: increased processor productiveness.

5 cl, 14 dwg



ФИГ.1

RU 2308754 C2

RU 2308754 C2

РОДСТВЕННЫЕ ЗАЯВКИ

Данная заявка соответствует заявке с серийным номером 10/039777, озаглавленной "Suspending Execution of a Thread in a Multi-threaded Processor" ("Приостановка выполнения потока в многопоточном процессоре"); заявке с серийным номером 10/039656, озаглавленной "Coherency Techniques for Suspending Execution of a Thread Until a Specified Memory Access Occurs" ("Способы когерентности для приостановки выполнения потока до тех пор, пока не будет осуществлен определенный доступ к памяти"); заявке с серийным номером 10/039650, озаглавленной "Instruction Sequences for Suspending Execution of a Thread Until a Specified Memory Access Occurs" ("Последовательности команд для приостановки выполнения потока до тех пор, пока не будет осуществлен определенный доступ к памяти"), все из которых поданы в тот же день, что и настоящая заявка.

Область техники, к которой относится изобретение

Данное изобретение относится к области процессоров. Более конкретно, настоящее изобретение относится к многопоточным процессорам и способам для временной приостановки обработки одного потока в многопоточном процессоре.

Предшествующий уровень техники

Многопоточный процессор способен обработать одновременно множество различных последовательностей команд. Главным побуждающим мотивом выполнения множества потоков команд в одном процессоре является увеличение эффективности использования процессора. Высокопараллельные архитектуры развились за эти годы, но часто оказывалось трудно извлечь достаточную параллельность из одного потока команд при работе с множеством исполняющих устройств. Процессоры параллельной обработки множества процессов стремятся выполнять множество потоков команд одновременно в различных исполняющих ресурсах выполнения в попытке лучше использовать эти ресурсы. Многопоточность может быть особенно выгодна для программ, которые сталкиваются с длительными задержками или которые часто ждут наступления какого-либо события. Когда один поток ожидает завершения задачи, имеющей в себе длительную задержку, или ожидает наступления определенного события, в это время может быть обработан другой поток.

Было предложено много различных способов управления переключением процессора между потоками. Например, некоторые процессоры обнаруживают особенно долгое ожидание события, такое как неудачное обращение к кэшу L2 (кэшу второго уровня), и в соответствии с этим переключают потоки. Хотя обнаружение подобных событий долгого ожидания может быть эффективным в некоторых обстоятельствах, такой способ обнаружения событий вряд ли обнаружит все точки, в которых может быть эффективным переключать потоки. В частности, переключение между потоками, основанное на событиях, может оказаться не в состоянии обнаружить точки в программе, где задержки предусмотрены программистом.

Фактически, программист зачастую находится в лучшем положении в плане определения того, когда было бы эффективно переключить потоки, чтобы избежать неэкономичных циклов ожидания семафора или других ресурсопотребляющих способов задержки. Таким образом, позволяя программам управлять переключениями между потоками, можно дать возможность программам работать более эффективно. В данном отношении могут быть выгодны явные команды программы, которые воздействуют на выбор потока. Например, команда "Pause" ("Пауза"), описанная в патентной заявке США № 09/489,130, поданной 21.01.2000. Команда Pause позволяет потоку выполнения быть временно приостановленным либо пока не будет достигнут подсчет, либо пока команда не пройдет конвейер процессора. Можно предложить различные способы для предоставления программистам возможности использовать ресурсы многопоточного процессора более эффективно.

Перечень фигур чертежей

Настоящее изобретение проиллюстрировано в качестве примера, но не ограничения, на фигурах сопроводительных чертежей.

Фиг.1 - иллюстрация одного варианта осуществления многопоточного процессора, имеющего средство мониторинга (монитор), предназначенного для мониторинга доступа к памяти.

Фиг.2 - блок-схема последовательности операций, поясняющая работу многопоточного процессора по Фиг.1 согласно одному варианту осуществления.

Фиг.3 - более подробная иллюстрация одного варианта осуществления многопоточного процессора.

Фиг.4 - иллюстрация разделения, совместного использования и дублирования ресурсов согласно одному варианту осуществления.

Фиг.5 - блок-схема последовательности операций, иллюстрирующая приостановку и возобновление выполнения потока согласно одному варианту осуществления.

Фиг.6a - блок-схема последовательности операций, иллюстрирующая активацию и работу логического средства мониторинга согласно одному варианту осуществления.

Фиг.6b - блок-схема последовательности операций, иллюстрирующая расширение возможностей наблюдения за операциями записи согласно одному варианту осуществления.

Фиг.7 - блок-схема последовательности операций, иллюстрирующая операции монитора согласно одному варианту осуществления.

Фиг.8 - иллюстрация системы согласно одному варианту осуществления.

Фиг.9a-9c - иллюстрации различных вариантов осуществления программных последовательностей, использующих раскрытые методики и команды процессора.

Фиг.10 - иллюстрация альтернативного варианта осуществления, который позволяет отслеживаемому адресу оставаться кэшированным.

Фиг.11 - иллюстрация различных конструкционных представлений или форматов для моделирования, эмуляции и изготовления конструкции с использованием раскрытых методик.

Детальное описание

В нижеследующем описании раскрыты способы для приостановки исполнения потока до тех пор, пока не произойдет определенный доступ к памяти. В нижеследующем описании многочисленные конкретные подробности, такие как варианты реализации логических средств, коды операций, средства задания операндов, варианты реализации разделения/совместного использования/дублирования ресурсов, типы и взаимосвязи компонентов системы и варианты выбора в отношении разделения/объединения логических средств, сформулированы так, чтобы дать наиболее полное понимание изобретения. Тем не менее специалисты в данной области техники оценят тот факт, что данное изобретение может быть осуществлено на практике и без таких конкретных подробностей. В других примерах управляющие структуры, цепи уровня логического вентиля и полные программные последовательности команд не показаны подробно, чтобы не затенять изобретения. Специалисты в данной области техники, используя приложенные описания, будут в состоянии реализовать соответствующие функциональные возможности без чрезмерного экспериментирования.

Раскрытые способы могут позволить программисту реализовать механизм ожидания в одном потоке, при этом разрешая другим потокам использовать вычислительные ресурсы. Монитор может быть сконфигурирован так, чтобы поток мог быть приостановлен до тех пор, пока не произойдет определенный доступ к памяти, такой как запись в заданную ячейку памяти. Таким образом, поток может быть возобновлен по происшествии заданного события без выполнения процедуры, чрезмерно потребляющей ресурсы процессора, такой как цикл ожидания семафора. В некоторых вариантах осуществления разделы, заранее предназначенные для приостановленного потока, могут быть переданы другим потокам, в то время как поток приостановлен. Эти и/или другие раскрытые способы могут улучшить совокупную производительность процессора.

Фиг.1 иллюстрирует один вариант осуществления многопоточного процессора 100, имеющего монитор 110 доступа к памяти для мониторинга доступа к памяти. "Процессор" в некоторых вариантах исполнения может быть сформирован как единая интегральная

микросхема. В других вариантах осуществления несколько интегральных микросхем могут совместно формировать процессор, а еще в одних вариантах осуществления совместно формировать процессор могут аппаратные средства и программное обеспечение (например, процедуры двоичной трансляции). В варианте осуществления по Фиг.1 контроллер 120 шины/памяти передает команды для выполнения на препроцессор 130. Препроцессор 130 управляет извлечением команд из различных потоков согласно указателям 170 команд. Логическое средство указателя команды дублируется для поддержки множества потоков.

Препроцессор 130 подает команды в разделяемые между потоками ресурсы 140 для дальнейшей обработки. Разделяемые между потоками ресурсы 140 включают в себя логически отделенные разделы, выделенные для конкретных потоков, когда несколько потоков активны в процессоре 100. В одном варианте осуществления каждый отдельный раздел содержит только команды из потока, для которого выделен этот раздел. Разделяемые между потоками ресурсы 140 могут включать в себя, например, очереди команд. В однопотоковом режиме разделяемые между потоками ресурсы 140 могут быть объединены с образованием единого большого раздела, выделяемого для одного потока.

Процессор 100 также включает в себя дублированное состояние 180. Дублированное состояние 180 включает в себя переменные состояния, достаточные для поддержания контекста для логического процессора. С помощью дублированного состояния 180 множество потоков может выполняться без конкуренции в отношении доступа к хранилищу переменных состояния. Кроме того, логические средства выделения регистров могут быть продублированы для каждого потока. Логические средства, связанные с дублированным состоянием, работают с соответствующими разделами ресурсов, чтобы подготавливать входящие команды к исполнению.

Разделяемые между потоками ресурсы 140 передают команды к совместно используемым ресурсам 150. Совместно используемые ресурсы 150 оперируют командами безотносительно их источника. Например, планировщик и исполняющие устройства могут быть совместно используемыми ресурсами, не осведомленными о потоках. Разделяемые ресурсы 140 могут подавать команды из множества потоков к совместно используемым ресурсам 150 посредством чередования между потоками справедливым способом, что обеспечивает постоянный прогресс выполнения в каждом активном потоке. Так, совместно используемые ресурсы могут выполнять передаваемые им команды в соответствующем состоянии, не заботясь о смешении потоков.

За совместно используемыми ресурсами 150 может следовать другой набор разделяемых между потоками ресурсов 160. Разделяемые между потоками ресурсы 160 могут включать в себя сбрасываемые ресурсы типа буфера переупорядочения и т.п. Соответственно, разделяемые между потоками ресурсы 160 могут гарантировать, что выполнение команд каждого потока будет завершено должным образом и что соответствующее состояние для этого потока будет соответствующим образом обновлено.

Как было ранее упомянуто, может быть желательно дать программистам возможность организации цикла ожидания семафора без требования постоянного опроса ячейки памяти или даже исполнения команд. Так, процессор 100 по Фиг.1 включает в себя монитор 110 доступа к памяти. Монитор 110 доступа к памяти выполнен с возможностью программирования с помощью информации относительно цикла доступа к памяти, в отношении которого монитор 110 может быть активирован для слежения. Соответственно, монитор 110 включает в себя информационный регистр 112 отслеживаемого цикла, который сравнивается с информацией цикла шины, принятой от контроллера 120 шины/памяти, посредством логических средств 114 сравнения. Если имеет место соответствие, то генерируется сигнал к возобновлению приостановленного потока. Информация о доступе к памяти может быть получена от внутренних и/или внешних шин процессора.

Информационный регистр 112 отслеживаемого цикла может содержать подробности, определяющие тип цикла и/или адрес, который должен запустить возобновление потока. В одном варианте осуществления информационный регистр 112 отслеживаемого цикла

хранит физический адрес, и монитор отслеживает каждый цикл шины, который указывает на фактическую или потенциальную операцию записи по этому физическому адресу. Такой цикл может быть в форме явного цикла записи и/или может быть операцией чтения для монопольного использования или циклом лишения действительного статуса, осуществляемым другим абонентом шины, пытающимся получить исключительное монопольное использование кэшируемой строки так, чтобы он мог записывать в эту строку без внешней транзакции шины. В любом случае, монитор может быть запрограммирован на запуск от различных транзакций в различных вариантах осуществления.

Функционирование по варианту осуществления по Фиг.1 может быть дополнительно пояснено с ссылкой на блок-схему последовательности операций по Фиг.2. В одном варианте осуществления набор команд процессора 100 включает в себя код операции (команду) MONITOR, которая задает информацию об отслеживаемой транзакции. На этапе 200 код операции MONITOR принимают как часть последовательности команд первого потока (T1). Как обозначено этапом 210, в качестве реализации на код операции MONITOR, процессор 100 активирует монитор 110 в целью мониторинга доступов к в отношении заданного доступа к памяти. Запускающий доступ к памяти может быть задан неявным или явным операндом. Поэтому, выполняя код операции MONITOR, можно задать отслеживаемый адрес, поскольку отслеживаемый адрес может быть сохранен заранее в регистре или в другом месте как неявный операнд. Как обозначено этапом 215, монитор проверяет, обнаружен ли заданный цикл. Если нет, то монитор продолжает мониторинг доступов к памяти. Если же запускающий цикл обнаружен, то устанавливают индикатор отслеживаемого события, ожидающего очереди на обработку, как обозначено этапом 220.

Исполнение кода операции MONITOR запускает активацию монитора 110. Монитор 110 может начать работать параллельно с другими операциями в процессоре. В одном варианте осуществления сама по себе команда MONITOR только конфигурирует монитор 110 с помощью надлежащей информации о цикле памяти и активирует монитор 110 без демаскирования отслеживаемых событий. Другими словами, в этом варианте осуществления после исполнения кода операции MONITOR отслеживаемые события могут накапливаться, но не могут быть распознаны, если они явно не демаскированы.

Так, на этапе 225 запуск ожидания памяти обозначают как отдельное событие. В некоторых вариантах осуществления код операции ожидания памяти (MWAIT) может использоваться для запуска распознавания отслеживаемых событий и приостановки потока T1. Использование двух отдельных команд для конфигурирования и запуска приостановки потока может обеспечить программисту дополнительную гибкость в программировании и возможность достижения большей эффективности в программировании. Однако в альтернативном варианте осуществления запускают ожидание памяти из первого кода операции, который также конфигурирует монитор 110. В том и другом случае одна или более команд активируют монитор и запускают распознавание отслеживаемых событий.

В вариантах осуществления, где используются отдельные коды операций для активации монитора 110 и для запуска распознавания отслеживаемых событий, может быть выгодно выполнить проверку, чтобы убедиться в том, что монитор был активирован перед приостановкой потока, как показано на этапе 230. Дополнительно, посредством проверки того, ожидает ли уже отслеживаемое событие очереди на обработку (не показано), приостановки T1 можно избежать, и работа может продолжаться на этапе 250. Предполагая, что монитор 110 активирован и нет событий монитора, уже ожидающих очереди на обработку, T1 может быть приостановлен как показано этапом 235.

С приостановленным T1 процессор входит в зависящее от реализации состояние, которое позволяет другим потокам более полно использовать ресурсы процессора. В некоторых вариантах осуществления процессор может освобождать некоторые или все разделы разделяемых ресурсов 140 и 160, которые были выделены для T1. В других вариантах осуществления различные изменения кода операции MONITOR или связанных с ним установок могут указывать, какие ресурсы освободить, если таковые имеются в наличии. Например, когда программист предполагает короткое ожидание, поток может быть приостановлен, но продолжает поддерживать свои разделы ресурсов. Производительность

по-прежнему повышена, потому что совместно используемые ресурсы могут использоваться исключительно другими потоками в течение периода приостановки потока. Когда предполагается более долгое ожидание, освобождение всех разделов, связанных с приостановленным потоком, позволяет другим потокам иметь дополнительные ресурсы, что потенциально увеличивает производительность других потоков. Дополнительная производительность, однако, достигается ценой непроизводительных издержек, связанных с удалением и добавлением разделов, когда потоки соответственно приостанавливаются и возобновляются.

Т1 остается в приостановленном состоянии до тех пор, пока не будет отслеживаемого события, ожидающего очереди на обработку. Как было ранее сказано, монитор 110 функционирует независимо с целью обнаружения отслеживаемых событий и сигнализации о них (этапы 215-220). Если на этапе 240 процессор определяет, что отслеживаемое событие ожидает очереди на обработку, то Т1 возобновляется, как обозначено этапом 250. Не требуется никакой активной обработки команд в Т1, чтобы отслеживаемое событие пробудило Т1. Напротив, Т1 остается приостановленным, и активированный монитор 110 сигнализирует о событии процессору. Процессор обрабатывает событие, распознает, что событие указывает на то, что Т1 должен быть возобновлен, и выполняет соответствующие действия по возобновлению Т1.

Так, варианты осуществления по Фиг.1 и 2 предоставляют способы, позволяющие потоку, приостановленному программой, быть возобновленным при происшествии заданного доступа к памяти. В одном варианте осуществления, другие события также обуславливают возобновление Т1. Например, прерывание может обусловить возобновление Т1. Такой вариант реализации выгодным образом позволяет монитору быть не совсем совершенным в том плане, что он может пропускать (не обнаруживать) некоторые доступы к памяти или другие условия, которые должны вызвать возобновление потока. В результате, время от времени Т1 может быть пробужден без необходимости. Однако такой вариант реализации уменьшает вероятность того, что Т1 станет постоянно остановленным (замороженным) из-за пропущенного события, что упрощает конструкцию и тестирование аппаратных средств. Ненужные пробуждения Т1 могут быть лишь незначительным неудобством, поскольку цикл может быть создан так, чтобы иметь двойную проверку для Т1 в отношении того, что условие, которого он ожидал, действительно произошло, и если нет, то приостановить себя еще раз.

В некоторых вариантах осуществления разделяемые между потоками ресурсы, сдублированные ресурсы и совместно используемые ресурсы могут размещаться по-разному. В некоторых вариантах осуществления могут отсутствовать разделяемые ресурсы на обоих концах совместно используемых ресурсов. В некоторых вариантах осуществления разделяемые ресурсы могут не быть строго разделены, а напротив, могут позволить некоторым командам пересекать разделы или могут позволить разделам изменяться в размере в зависимости от потока, выполняющегося в этом разделе, или от общего количества выполняющихся потоков. Дополнительно, различные смеси ресурсов могут быть обозначены как совместно используемые, дублированные и разбитые на разделы ресурсы.

Фиг.3 иллюстрирует дополнительные детали одного варианта осуществления многопоточного процессора. Вариант осуществления по Фиг.3 включает в себя, среди прочего, связанные с когерентностью логические средства 350, один вариант реализации монитора 310 и один конкретный вариант реализации логических средств 377 приостановки и возобновления потока. В варианте осуществления по Фиг.3 шинный интерфейс 300 включает в себя контроллер 340 шины, логические средства 345 обнаружения событий, монитор 310 и связанные с когерентностью логические средства 350.

Шинный интерфейс 300 передает команды препроцессору 365, который выполняет генерацию микрооперанда (и ОР), генерируя микрооперанды из макрокоманд. Ресурсы 370 исполнения принимают микрооперанды от препроцессора 365, а логические средства 380 постпроцессора сбрасывают различные микрооперанды после того, как они были выполнены. В одном варианте осуществления неупорядоченное исполнение

поддерживается препроцессором, постпроцессором и ресурсами исполнения.

Различные подробности операций более подробно рассмотрены в соответствии с Фиг.5-9. Кратко, код операции MONITOR может быть введен в процессор через шинный интерфейс 300 и быть подготовлен к исполнению препроцессором 365. В одном варианте осуществления генерируется специальный микрооперанд MONITOR для выполнения ресурсами 370 исполнения. Микрооперанд MONITOR может быть обработан исполняющими устройствами аналогично с операциями сохранения с отслеживаемым адресом, транслируемым логическими средствами 375 трансляции в физический адрес, который передается монитору 310. Монитор 310 осуществляет связь с логическими средствами 377 приостановки и возобновления потоков с целью того, чтобы вызвать возобновление потоков. Логические средства приостановки и возобновления потока могут выполнять разбиение веществ на разделы и воссоединение ресурсов по мере изменения числа активных потоков.

Например, Фиг.4 поясняет разбиение на разделы, дублирование и совместное использование ресурсов согласно одному варианту осуществления. Разбитые на разделы ресурсы могут быть разбиты на разделы и воссоединены (соединены вместе для повторного использования другими потоками) в соответствии с убыванием и прибавлением активных потоков в машине. В варианте осуществления по Фиг.4 дублированные ресурсы включают в себя логическое средство указателя команды в части извлечения команды из состава конвейера, логические средства переименования регистра в части переименования из состава конвейера, переменные состояния (не показаны, но к ним производится обращение на различных стадиях в конвейере) и контроллер прерываний (не показан, обычно асинхронный по отношению к конвейеру). Совместно используемые ресурсы в варианте осуществления по Фиг.4 включают в себя планировщики на стадии планирования конвейера, совокупность (пул) регистров в частях чтения и записи регистров из состава конвейера, ресурсы исполнения в части исполнения из состава конвейера. Дополнительно, кэш с отслеживанием и кэш L1 (первого уровня) данных могут быть совместно используемыми ресурсами, заполняемыми согласно доступам к памяти безотносительно контекста потока. В других вариантах осуществления рассмотрение контекста потока может использоваться в принятии решений о кэшировании. Разбитые на разделы ресурсы в варианте осуществления по Фиг.4 включают в себя две очереди в стадиях организации очереди конвейера, буфер переупорядочения на стадии сброса конвейера и буфер хранения. Мультиплексирующие логические средства выбора потока переключаются между различными дублированными и разбитыми на разделы ресурсами, чтобы обеспечить приемлемый доступ к обоим потокам.

Для целей примера предполагается, что разбиение на разделы, совместное использование, разделение и дублирование, показанные на Фиг.4, используются вместе с вариантом осуществления по Фиг.3 в дальнейшем описании функционирования варианта осуществления процессора по Фиг.3. В частности, дополнительные детали работы варианта осуществления по Фиг.3 будут рассмотрены далее в соответствии с блок-схемой последовательности операций по Фиг.5. Предполагается, что процессор работает в многопоточном режиме с по меньшей мере двумя активными потоками.

На этапе 500 препроцессор 365 принимает код операции MONITOR во время исполнения первого потока (T1). Специальный микрооперанд монитора генерируется препроцессором 365 в одном варианте осуществления. Микрооперанд монитора передают ресурсам 370 исполнения. Микрооперанд монитора имеет ассоциированный с ним адрес, который указывает адрес, который подлежит мониторингу (отслеживаемый адрес). Ассоциированный адрес может быть в форме явного или неявного операнда (то есть, ассоциированный адрес должен быть взят из заранее определенного регистра или другой ячейки памяти). Ассоциированный адрес "служит индикатором" отслеживаемого адреса в том плане, что он обладает достаточной информацией, чтобы определить отслеживаемый адрес (возможно, в сочетании с другими регистрами или информацией). Например, ассоциированный адрес может быть линейным адресом, который имеет соответствующий физический адрес, который является надлежащим отслеживаемым адресом.

Альтернативно, отслеживаемый адрес может быть дан в формате виртуального адреса, или может быть обозначен как относительный адрес, или задан другим известным или удобным способом задания адреса. Если используются операнды в виде виртуальных адресов, то может быть желательно позволить общим нарушениям защиты быть распознанными как события принудительного останова.

Отслеживаемый адрес может указывать на любой подходящий блок памяти для мониторинга. Например, в одном варианте осуществления отслеживаемый адрес может указывать на строку кэша. Однако в альтернативных вариантах осуществления отслеживаемый адрес может указывать на часть строки кэша, на часть или блок памяти определенного/выбранного размера, который может иметь различные отношения к размерам строк кэша разных процессоров, или на отдельный адрес адреса. Отслеживаемый адрес, таким образом, может указывать на блок, который содержит данные, заданные операндом (и большее количество данных), или может указывать конкретно на адрес для требующегося блока данных.

В варианте осуществления по Фиг.3 отслеживаемый адрес подается на логические средства 375 трансляции адресов и передается на монитор 310, где его сохраняют в регистре 335 отслеживаемого адреса. В ответ на код операции MONITOR, ресурсы 370 исполнения далее запускают и активируют монитор 310, как обозначено этапом 510 и более детально показано на Фиг.6. Как будет более подробно обсуждено ниже в отношении Фиг.6, может быть выгодно временно заблокировать любые операции сохранения, происходящие после кода операции MONITOR, чтобы гарантировать, что операции сохранения обработаны и, следовательно, обнаружены прежде, чем произойдет любая приостановка потока. Таким образом, некоторые операции могут иметь место как результат активации монитора 310 прежде, чем любые последующие команды могут быть исполнены в этом варианте осуществления. Тем не менее, этап 510 показан, как происходящий параллельно этапу 505, потому что монитор 310 продолжает работать параллельно с другими операциями, пока не произойдет событие принудительного останова после его активации кодом операции MONITOR в этом варианте осуществления.

На этапе 505 код операции ожидания памяти (MWAIT) принят в потоке 1 и передан на исполнение. Исполнение кода операции MWAIT демаскирует отслеживаемые события в варианте осуществления по Фиг.5. В качестве реакции на код операции MWAIT выполняется тест, как это обозначено этапом 515, для определения того, ожидает ли очереди на обработку отслеживаемое событие. Если нет никаких отслеживаемых событий, ожидающих очереди на обработку, то на этапе 520 выполняется тест, чтобы удостовериться в том, что монитор активен. Например, если бы MWAIT выполнялся без предварительного выполнения MONITOR, то монитор 310 не был бы активен. Если либо монитор неактивен, либо отслеживаемое событие ожидает очереди на обработку, то исполнение потока 1 продолжается на этапе 580.

Если монитор 310 активен и нет никаких событий монитора, ожидающих очереди на обработку, то исполнение потока 1 приостанавливается, как обозначено этапом 525. Логические средства 377 приостановки/возобновления потока включают в себя логические средства 382 очистки конвейера, которые очищают конвейер процессора, чтобы убрать все команды, как обозначено этапом 530. После очистки конвейера логические средства 385 разбиения/воссоединения разделов предписывают освободить все разбитые на разделы ресурсы, связанные исключительно с потоком 1, для использования другими потоками, как обозначено этапом 535. Эти освобожденные ресурсы воссоединяют для формирования набора больших ресурсов для использования их остающимися активными потоками. Например, что касается двух иллюстрируемых потоков по Фиг.4, удаляют все команды, относящиеся к потоку 1, из обеих очередей. Каждую пару очередей затем объединяют, чтобы создать большую очередь для второго потока. Точно так же, большее количество регистров из пула регистров делают доступным для второго потока, большее количество элементов из буфера сохранения освобождают для второго потока и большее количество элементов в буфере переупорядочения делают доступными для второго потока. В сущности, эти структуры возвращаются к назначаемым для одного потока структурам с

удвоенным размером. Конечно, разные пропорции могут следовать из разных вариантов осуществления, использующих разные количества потоков.

На этапах 540, 545, и 550 проверяют различные события, чтобы определить, должен ли поток 1 быть возобновлен. Следует отметить, что эти тесты не выполняются командами, исполняющимися как часть потока 1. Наоборот, эти операции выполняются процессором параллельно с обработкой других потоков. Как будет обсуждено более детально в соответствии с Фиг. 6, монитор сам проверяет, произошло ли отслеживаемое событие записи и указывает на результат проверки установкой индикатора события, ожидающего очереди на обработку. Индикатор события, ожидающего очереди на обработку, передается через сигнал EVENT (СОБЫТИЕ) логическим средствам 377 приостановки/возобновления (например, микрокоду). Микрокод может распознать отслеживаемое событие на соответствующей границе команды в одном варианте осуществления (этап 540), так как это событие было демаскировано кодом операции MWAIT на этапе 505. Логические средства 345 обнаружения событий могут обнаруживать другие события, такие как прерывания, которые обозначены как события принудительного останова (этап 545). Кроме того, в необязательном порядке может использоваться таймер для периодического выхода из состояния ожидания памяти, чтобы гарантировать, что процессор не войдет в замороженное состояние из-за некоторых специфических последовательностей событий (этап 550). Если ни одно из этих событий не сигнализирует о выходе из состояния ожидания памяти, то поток 1 остается приостановленным.

Если поток 1 возобновляется, то логические средства 377 приостановки/возобновления потока снова активируются при обнаружении соответствующего события. Опять же, конвейер очищают как обозначено этапом 560, чтобы удалить команды из конвейера так, чтобы ресурсы могли быть еще раз разбиты на разделы с целью адаптации к потоку 1, который должен быть вскоре возобновлен. На этапе 570 соответствующие ресурсы заново разбивают на разделы, и поток 1 возобновляют на этапе 580.

Фиг.6a иллюстрирует более подробно активацию и работу монитора 310. На этапе 600 выполняемое препроцессором извлечение для потока 1 останавливают, чтобы предотвратить ввод в машину дальнейших команд потока 1. На этапе 605 ассоциированный адресный операнд преобразовывают из линейного адреса в физический адрес посредством логических средств 375 трансляции адресов. На этапе 610 возможность наблюдения за операциями записи по отслеживаемому адресу увеличивается. В общем, задачей этой операции является заставить кэширующих абонентов сделать операции записи, которые воздействуют на информацию, хранящуюся по отслеживаемому адресу, видимыми для самого монитора 310. Более подробно один конкретный вариант осуществления обсужден в соответствии с Фиг.6b. На этапе 615 физический адрес для мониторинга сохраняют, хотя следует отметить, что этот адрес может быть сохранен в этой последовательности как раньше, так и позже.

Затем, как обозначено этапом 620, монитор активируют. Монитор осуществляет мониторинг циклов шины в отношении записей по физическому адресу, который является отслеживаемым адресом, хранящимся в регистре 335 отслеживаемого адреса. Более подробно операции мониторинга обсуждены ниже в соответствии с Фиг.7. После активации монитора выполняют операцию временного блокирования сохранения, как обозначено этапом 625. Операция временного блокирования сохранения гарантирует, что все операции сохранения в машине обработаны к моменту завершения исполнения кода операции MONITOR. Со всеми операциями сохранения, предшествующими удалению MONITOR из машины, вероятность того, что вход в состояние ожидания памяти будет совершен ошибочно, снижается. Операция временного блокирования сохранения, однако является мерой предосторожности, и может быть длительной операцией.

Эта временное блокирование сохранения является необязательным, потому что механизм MONITOR/MWAIT этого варианта осуществления был спроектирован как механизм со множеством выходов. Другими словами, различные события типа некоторых прерываний, системных или встроенных таймеров и т.д. могут также вызывать выход из состояния ожидания памяти. Таким образом, не гарантируется в этом варианте

осуществления, что единственная причина, по которой поток будет возобновлен, состоит в изменении значения отслеживаемых данных. Соответственно (смотри также Фиг.9a-c ниже), в этом варианте реализации программное обеспечение должно дважды проверить, изменилось ли конкретное значение, сохраненное в памяти. В одном варианте осуществления некоторые события, включая ввод прерываний INTR, NMI и SMI; прерывания машинной проверки; нарушения являются событиями принудительного останова, а другие, включая события выключения питания, таковыми не являются. В одном варианте осуществления проверка штекера A20M - также событие принудительного останова.

Как обозначено этапом 630, монитор продолжает проверять, указывают ли происходящие циклы шины на операцию записи по отслеживаемому адресу или представляются ли происходящие циклы указывающими на операцию записи по отслеживаемому адресу. Если такой цикл шины обнаружен, устанавливают индикатор отслеживаемого события, ожидающего очереди на обработку, как обозначено этапом 635. После исполнения кода операции MWAIT (этап 505, Фиг. 5) этот индикатор события, ожидающего очереди на обработку, обрабатывается как событие и вызывает возобновление потока на этапах 560-580 по Фиг.5. Дополнительно, события, которые изменяют трансляцию адресов, могут вызвать возобновление потока 1. Например, события, которые вызывают очистку буфера предыстории трансляции, могут запускать возобновление потока 1, так как трансляция, выполненная для генерирования отслеживаемого адреса из адреса, преобразованного из линейного в физический, может больше не быть действительной. Например, в процессоре, совместимом с архитектурой Intelx86, записи в управляющие регистры CRO, CR3 и CR4, так же как и в некоторые регистры, зависящие от конкретной машины, могут вызывать выход из состояния ожидания памяти.

Как было отмечено выше, Фиг.6b иллюстрирует более подробно расширение возможности наблюдения за операциями записи по отслеживаемому адресу (этап 610 по Фиг.6a). В одном варианте осуществления процессор вычищает строку кэша, связанную с отслеживаемым адресом, из всех внутренних кэшей процессора, как обозначено этапом 650. В результате этой очистки любая последующая операция записи по отслеживаемому адресу достигает шинного интерфейса 300, предоставляя возможность обнаружения монитору 310, который включен в шинный интерфейс 300. В одном варианте осуществления микрооперанд MONITOR моделируют после и имеет ту же самую модель неисправности, что и команда CLFLUSH очистки строки кэша, которая является командой из системы команд x86. Микрооперанд MONITOR выполняет трансляцию адреса из линейного в физический и очищает внутренние кэши, по большому счету, так же, как CLFLUSH; однако, шинный интерфейс распознает разницу между MONITOR и CLFLUSH и обрабатывает микрооперанд MONITOR соответственно.

Затем, как обозначено этапом 655, логические средства 350, связанные с когерентностью, в шинном интерфейсе 300 активируют логические средства 355 генерации считывания строки, чтобы сгенерировать транзакцию считывания строки на процессорной шине. Транзакция считывания строки к отслеживаемому адресу гарантирует, что никакие другие кэши в процессоре на шине не хранят данные по отслеживаемому адресу ни в совместно используемом, ни в исключительном состоянии (согласно известному протоколу MESI). В других протоколах могут использоваться другие состояния; однако, транзакция разработана так, чтобы уменьшить вероятность того, что другой абонент может осуществить запись по отслеживаемому адресу без транзакции, наблюдаемой монитором 310. Другими словами, в отношении операций записи или указывающих на операцию записи транзакций впоследствии осуществляется широковещание так, что они могут быть обнаружены монитором. Как только операция считывания строки выполнена, монитор 310 начинает мониторинг транзакций на шине.

Поскольку на шине происходят дополнительные транзакции, логические средства, связанные с когерентностью, продолжают сохранять возможность наблюдения отслеживаемого адреса, пытаясь предотвратить попытки абонентов шины получить строку

кэша, связанную с отслеживаемым адресом, в монопольное использование. Согласно одному протоколу шины, это может быть выполнено логическими средствами 360 генерации сигналов обращения, формирующими сигнал HIT# в течение стадии слежения в отношении любой операции чтения по отслеживаемому адреса, как обозначено этапом 660. Формирование HIT# предотвращает другие кэши от перемещения за пределы состояния совместного использования в протоколе MESI в эксклюзивное состояние, а затем, возможно, в модифицированное состояние. В результате, как обозначено этапом 665, никакие абоненты в выбранном домене когерентности (часть памяти, которая поддерживается когерентной) не могут иметь данные в изменяемом или исключительном состоянии (или их эквивалентах). Процессор эффективно проявляется как имеющий строку кэша отслеживаемого адреса, кэшированную несмотря на то, что она была удалена из внутренних кэшей в этом варианте осуществления.

Обращаясь теперь к Фиг.7, далее более подробно даны действия, связанные с этапом 620 по Фиг.6а. В частности, Фиг.7 поясняет более подробно операции монитора 310. На этапе 700 монитор 310 принимает запрос и информацию об адресе от контроллера 340 шины для транзакции шины. Как обозначено этапом 710, монитор 310 исследует тип цикла шины и задействованный адрес (адреса). В частности, логические средства 320 сравнения циклов определяют, является ли цикл шины заданным циклом. В одном варианте осуществления схема 330 сравнения адресов сравнивает адрес транзакции шины с отслеживаемым адресом, хранящимся в регистре 335 отслеживаемого адреса, и логические средства 325 обнаружения записи декодируют информацию типа цикла от контроллера 340 шины, чтобы обнаружить, произошла ли операция записи. Если операция записи по отслеживаемому адресу произошла, то устанавливают индикатор отслеживаемого события, ожидающего очереди на обработку, как обозначено этапом 720. Сигнал ЗАПИСЬ ОБНАРУЖЕНА (WRITE DETECTED) передается логическим средствам 377 приостановки/возобновления потока для сигнализации о событии (и далее будет обслужен в предположении, что они были активированы при выполнении MWAIT). Наконец, монитор 310 приостанавливают, как обозначено этапом 730. Останов монитора экономит энергию, но это не критично, пока ложные отслеживаемые события замаскированы или, в противном случае, не генерируются. В этот момент индикатор отслеживаемого события может быть также переустановлен. Как правило, обслуживание отслеживаемого события также маскирует распознавание дальнейших отслеживаемых событий, пока MWAIT снова не будет выполнен.

В случае чтения отслеживаемого адреса активируют логические средства 350, связанные с когерентностью. Как обозначено этапом 740, формируют сигнал (типа HIT#), чтобы предотвратить получение монопольного использования другим абонентом, что позволило бы в будущем выполнять операции записи без широковещания когерентности. Монитор 310 остается активным и затем возвращается на этап 700 незатронутым чтением по отслеживаемому адресу. Дополнительно, если транзакция не является ни чтением, ни записью по отслеживаемому адресу, то монитор остается активным и возвращается на этапе 700.

В некоторых вариантах осуществления команда MONITOR ограничена таким образом, что только некоторые типы доступов могут быть отслежены. Эти доступы могут быть доступами, выбранными как показательные для эффективных способов программирования, или могут быть выбраны по другим причинам. Например, в одном варианте осуществления доступ к памяти должен быть кэшируемым хранилищем в памяти с обратной записью, которое естественным образом выровнено. Выровненный естественным образом элемент - это N-битный элемент, который начинается с адреса, делящегося на N. В результате использования выровненных естественным образом элементов нужно осуществить доступ к одной строке кэша (в отличие от двух строк кэша, как было бы необходимо в случае, когда данные разделены между двумя строками кэша), чтобы осуществить запись по отслеживаемому адресу. В результате, используя выровненные естественным образом адреса памяти, можно упростить процесс наблюдения шины.

Фиг.8 поясняет один вариант осуществления системы, который использует раскрытые

здесь способы многопоточного ожидания памяти. В варианте осуществления по Фиг.8 набор N многопоточных процессоров, процессоры от 805-1 до 805-N, подключены к шине 802. В других вариантах осуществления может использоваться один процессор или комбинация многопоточных процессоров и однопоточных процессоров. Также могут использоваться и другие известные или иным образом доступные компоновки систем. Например, процессор может быть подсоединен способом «от точки к точке», и части, такие как интерфейс памяти, могут быть интегрированы в каждый процессор.

В варианте осуществления по Фиг.8 интерфейс памяти 815, подключенный к шине, подключен к памяти 830 и интерфейсу 820 среды. Память 830 содержит операционную систему 835, поддерживающую работу множества процессоров, а также команды для первого потока 840 и команды для второго потока 845. Команды 830 включают в себя цикл ожидания согласно раскрытым способам, различные варианты которых показаны на Фиг.9a-9c.

Чтобы исполнить эти различные функции, соответствующее программное обеспечение может быть предоставлено на любом из ряда машиночитаемых носителей. Интерфейс 820 среды представляет интерфейс для такого программного обеспечения. Интерфейс 820 среды может быть интерфейсом к носителю данных (например, накопителю на магнитных дисках, оптическому дисководу, накопителю на магнитной ленте, энергозависимую память, энергонезависимую память и т.п.) или к среде передачи (например, сетевой интерфейс или другой цифровой или аналоговый интерфейс). Интерфейс 820 среды может считывать программные процедуры со среды (например, с носителя 792 данных или среды 795 передачи). Машиночитаемые носители - это любые среды, которые могут хранить по меньшей мере временно информацию для чтения через машинный интерфейс. Это может включать в себя передачи сигналов (через проводную, оптическую или эфирную среду) и/или материальные носители 792 данных, такие как различные типы запоминающих и дисковых устройств.

Фиг. 9a иллюстрирует цикл ожидания согласно одному варианту осуществления. На этапе 905 команду MONITOR выполняют с адресом 1 в качестве ее операнда, т.е. отслеживаемого адреса. Команду MWAIT выполняют на этапе 910 в пределах одного и того же потока. Как было ранее рассмотрено, команда MWAIT вызывает приостановку потока, предполагая, что другие условия должным образом выполнены. Когда событие принудительного останова происходит на этапе 915, процедура переходит на этап 920, чтобы определить, изменилось ли сохраненное значение по отслеживаемому адресу. Если значение по отслеживаемому адресу изменилось, то исполнение потока продолжается, как обозначено этапом 922. Если значение не изменялось, то произошло событие ложного пробуждения. Событие пробуждения ложно в смысле, что из MWAIT был осуществлен выход, и при этом не было записи по отслеживаемому адресу. Если значение не изменилось, то цикл возвращается к этапу 905, где монитор конфигурируют еще раз. Эта программная реализация цикла позволяет разработать монитор так, чтобы допускать ложные пробуждения.

Фиг. 9b иллюстрирует альтернативный цикл ожидания. В варианте осуществления по Фиг.9b добавлена одна дополнительная проверка, чтобы еще сильнее уменьшить вероятность того, что команда MWAIT будет не в состоянии отловить операцию записи по отслеживаемому адресу памяти. Снова поток начинается по Фиг.9b с выполнения команды MONITOR с адресом 1 в качестве ее операнда, как обозначено этапом 925. Дополнительно, на этапе 930 программная процедура считывает значение памяти по отслеживаемому адресу. На этапе 935 программа выполняет двойную проверку, чтобы гарантировать, что значение памяти не изменилось по отношению к значению, указывающему, что поток должен быть переведен в состояние ожидания. Если значение изменилось, то выполнение потока продолжают, как обозначено этапом 952. Если значение не изменилось, то выполняют команду MWAIT, как обозначено этапом 940. Как ранее было рассмотрено, поток приостановлен, пока событие принудительного останова не произойдет на этапе 945. Однако опять же, так как ложные события принудительного останова позволены, проверку в отношении того, изменилось ли значение, снова выполняют на этапе 950. Если

значение не изменилось, то цикл возвращается, чтобы еще раз активировать монитор для проверки адреса 1, возвращаясь на этап 925. Если значение изменилось, то исполнение потока продолжается на этапе 952. В некоторых вариантах осуществления может оказаться не нужно выполнять команду MONITOR снова после события ложного пробуждения прежде, чем команда MWAIT будет выполнена, чтобы снова приостановить поток.

Фиг.9c иллюстрирует другой пример программной последовательности, использующей команды MONITOR и MWAIT. В примере по Фиг.9c цикл не переходит в режим ожидания, пока в пределах потока не возникнут два отдельных задания, которые не имеют никакой работы. Постоянное значение CV1 сохраняют в рабочей ячейке WL1, когда имеется работа, которая должна быть выполнена первой процедурой. Точно так же второе постоянное значение CV2 сохраняют в WL2, когда имеется работа, которая должна быть выполнена второй процедурой. Чтобы использовать один отслеживаемый адрес, WL1 и WL2 выбраны так, чтобы быть ячейками памяти в одной строке кэша. Альтернативно, одна рабочая ячейка может также использоваться для хранения индикаторов состояния памяти для множества заданий. Например, каждый из одного или большего количества битов в одном байте или другом блоке может представлять отличающееся от других задание задания.

Как обозначено этапом 955, монитор конфигурируют для мониторинга WL1. На этапе 960 проверяют, хранит ли WL1 постоянное значение, указывающее, что имеется работа, которую требуется выполнить. Если это так, то работу, связанную с WL1, выполняют, как обозначено этапом 965. Если нет, то на этапе 970 проверяют, хранит ли WL2 значение CV2, указывающее на то, что имеется работа, которая должна быть выполнена относительно WL2. Если это так, то выполняют работу, связанную с WL2, как обозначено этапом 975. Если нет, то цикл может перейти к определению того, следует ли вызвать обработчик управления питанием на этапе 980. Например, если прошло выбранное количество времени, то логический процессор может быть переведен в состояние уменьшенного расхода энергии (например, в одно из набора состояний "C", определенных в соответствии со спецификацией Усовершенствованного конфигурирования и Интерфейса питания (ACPI), Версия 1.Ob (или более поздняя), изданная 08.02.99, доступная по адресу www.acpi.info, что касается подачи настоящей заявки). Если это так, то обработчик управления питанием вызывают на этапе 985. В одном из случаев 965, 975 и 985, где имелась работа, которая должна быть выполнена, поток выполняет эту работу и затем возвращается, чтобы сделать снова те же самые определения после установки монитора на этапе 955. В альтернативном варианте осуществления возврат цикла из блоков 965, 975 и 985 может быть осуществлен к блоку 960 пока монитор остается активным.

Если не нашлось никакой работы на этапах 965, 975 и 985, то выполняют команду MWAIT, как обозначено этапом 990. Из состояния приостановки потока, обусловленного MWAIT, в конечном счете, выходят, когда происходит событие принудительного останова, как обозначено этапом 995. В этот момент цикл возвращается на этап 955, чтобы установить монитор и после того определить, указывает ли WL1 или WL2 на то, что имеется работа, которую нужно выполнить. Если никакой работы не нужно выполнять (например, в случае события ложного пробуждения), цикл возвратится к MWAIT на этапе 990 и снова приостановит поток, пока не произойдет событие принудительного останова.

Фиг.10 иллюстрирует один альтернативный вариант осуществления процессора, который позволяет отслеживаемому значению оставаться кэшированным в кэше L1. Процессор по Фиг.10 включает в себя исполняющие устройства 1005, кэш 1010 L1 и буферы 1020 объединения записей между кэшем L1 и включительным кэшем 1030 L2. Буферы 1020 объединения записей включают в себя порт 1044 слежения, который гарантирует когерентность внутренних кэшей с другой памятью через операции, полученные посредством шинного интерфейса 1040 от шины 1045. Так как воздействующие на когерентность транзакции достигают буферов 1020 объединения записей через порт 1044 слежения, монитор может быть расположен на уровне кэша L1, и все равно принимать достаточно информации, чтобы определить, когда событие записи в память имеет место на шине 1045. Так, строка памяти, соответствующая отслеживаемому адресу, может

поддерживаться в кэше L1. Монитор в состоянии обнаружить, как записи в кэш L1 от исполняющих устройств, так и записи от шины 1045 через порт 1044 слежения.

В другом альтернативном варианте осуществления поддерживается команда MONITOR с двумя операндами. Один операнд указывает адрес памяти, как было ранее обсуждено. Второй операнд - маска, которая указывает, которое из множества событий, которые иначе не прервали бы состояние ожидания памяти, должно вызвать прерывание этого конкретного ожидания памяти. Например, один бит маски может указывать, что маскированным прерываниям следует разрешить прервать состояние ожидания памяти несмотря на тот факт, что прерывания замаскированы (например, разрешение события пробуждения даже когда бит IF EFLAGS установлен на маскировку прерываний). Возможно тогда одна из команд, выполненных после прерывания состояния ожидания памяти, демаскирует это прерывание, и таким образом оно будет обслужено. Другим событиям, которые иначе не прервали бы состояние ожидания памяти, можно разрешить прервать состояние ожидания памяти, или наоборот события, которые обычно прерывают состояние ожидания памяти можно заблокировать. Как было обсуждено в случае первого операнда, второй операнд может быть явным или неявным.

Фиг. 11 иллюстрирует различные конструкционные представления или форматы для моделирования, эмуляции и изготовления конструкции, используя раскрытые способы. Данные, представляющие конструкцию, могут представлять конструкцию рядом способов. Сначала, как полезно при моделировании, аппаратные средства могут быть представлены с использованием языка описания аппаратных средств (HDL) или другого языка функционального описания, который по существу дает компьютеризированную модель того, как сконструированные аппаратные средства по идее должны работать. Модель 1110 аппаратных средств может быть сохранена в носителе 1100 данных, таком как память компьютера так, чтобы эта модель могла быть воспроизведена с использованием программного обеспечения моделирования 1120, которое применяет определенный испытательный набор 1130 программ к модели 1110 аппаратных средств, чтобы определить, действительно ли она функционирует так как предназначено. В некоторых вариантах осуществления программное обеспечение моделирования не записывается, не фиксируется и не содержится на носителе.

Дополнительно, модель уровня схем с логическими и/или транзисторными вентилями может быть выполнена на некоторых стадиях процесса конструирования. Эта модель может аналогичным образом воспроизводиться иногда с использованием специализированных имитаторов оборудования, которые формируют модель, используя программируемые логические средства. Этот тип моделирования, предпринимаемый на следующем этапе, может быть способом эмуляции. В любом случае переконфигурируемые аппаратные средства могут представлять собой другой вариант осуществления, который может включать в себя машиночитаемый носитель, хранящую модель, использующую раскрытые способы.

Кроме того, большинство конструкций на некоторой стадии достигают уровня данных, представляющих физическое расположение различных устройств в аппаратной модели. В случае, где используются обычные способы изготовления полупроводников, данные, представляющие модель аппаратных средств, могут быть данными, определяющими присутствие или отсутствие различных признаков на различных уровнях маски для масок, используемых для производства интегральных микросхем. Опять же, эти данные, представляющие интегральную микросхему, воплощают способы, раскрытые в том плане, что схемы и логические средства в этих данных могут быть смоделированы и изготовлены для выполнения этих способов.

В любом представлении конструкции данные могут храниться в любой форме на машиночитаемом носителе. Носителем может быть оптическая или электрическая волна 1160, модулируемая или иным образом сгенерированная, чтобы передать такую информацию, память 1150 или магнитный или оптический носитель 1140, такой как диск. Набор битов, описывающих конструкцию или определенную часть конструкции, - это изделие, которое может быть продано само по себе или может использоваться другими для

дальнейшего исполнения или изготовления.

Таким образом, раскрыты способы для приостановки исполнения потока до тех пор, пока не произойдет определенный доступ к памяти. Хотя некоторые иллюстративные варианты осуществления описаны и показаны на сопроводительных чертежах, следует понимать, что такие варианты осуществления даны просто для пояснения и не ограничивают объем изобретения, и что это изобретение не ограничивается конкретными показанными и описанными конструкциями и вариантами размещения, поскольку различные другие модификации могут реализовываться специалистами в данной области техники после изучения этого раскрытия.

Формула изобретения

1. Процессор, содержащий множество исполняющих устройств для исполнения множества потоков, включая первый поток, имеющий первую команду, имеющую ассоциированный с ней адресный операнд, указывающий отслеживаемый адрес; средства приостановки для приостановки исполнения упомянутого первого потока; средство мониторинга для мониторинга, в качестве реакции на первую команду, доступов к памяти по отслеживаемому адресу, при этом средство мониторинга выполнено с возможностью вызвать возобновление первого потока в качестве реакции на доступ к памяти по отслеживаемому адресу.

2. Процессор по п.1, в котором средство мониторинга вызывает возобновление в качестве реакции на доступ к памяти, только если доступ к памяти указывает на фактическую или потенциальную операцию записи по отслеживаемому адресу.

3. Процессор по п.1 или 2, в котором средство мониторинга вызывает возобновление первого потока в качестве реакции на результат сравнения адреса, по которому осуществляется доступ к памяти, и отслеживаемого адреса, если первый поток приостановлен и отслеживаемые события немаскированы.

4. Процессор по п.1, дополнительно содержащий логические средства обнаружения событий, предназначенные для того, чтобы вызвать возобновление первого потока в качестве реакции на событие, отличное от упомянутого доступа к памяти.

5. Процессор по п.4, в котором упомянутое событие представляет собой прерывание или нарушение.

6. Процессор по п.1, в котором упомянутый ассоциированный адресный операнд есть неявный операнд.

7. Процессор по п.6, в котором упомянутый ассоциированный адресный операнд хранится в заранее определенном регистре.

8. Процессор по п.1, в котором упомянутые средства приостановки приостанавливают исполнение первого потока в качестве реакции на вторую команду, которая демаскирует события, о которых просигнализировало средство мониторинга.

9. Процессор по п.8, в котором упомянутая вторая команда только запускает монитор, если была выполнена первая команда.

10. Процессор по п.1, в котором средства приостановки приостанавливают исполнение упомянутого первого потока в качестве реакции на первую команду.

11. Процессор по п.1, дополнительно содержащий средства когерентности для улучшения возможности наблюдения за операциями сохранения по упомянутому отслеживаемому адресу.

12. Процессор по п.1, дополнительно содержащий средства когерентности для гарантирования того, что никакой кэш в пределах домена когерентности не хранит информацию по упомянутому отслеживаемому адресу в модифицированном или исключительном состоянии.

13. Процессор по п.11 или 12, в котором упомянутые средства когерентности вычищают строку кэша, связанную с упомянутым отслеживаемым адресом, из всех внутренних кэшей и генерируют транзакцию шины, соответствующую считыванию строки, в отношении строки кэша, связанной с упомянутым отслеживаемым адресом, к другим процессорам, подключенным к данному процессору.

14. Процессор по п.13, в котором средства когерентности вызывают генерирование процессором цикла шины для предотвращения исполнения транзакции записи по упомянутому отслеживаемому адресу любыми другими абонентами шины без широковещания о транзакции записи.

15. Процессор по п.13, дополнительно содержащий средства для формирования сигнала обращения в качестве реакции на считывание информации другим абонентом шины по упомянутому отслеживаемому адресу.

16. Процессор по п.15, в котором упомянутые средства для формирования сигнала обращения формируют сигнал обращения во время фазы слежения в отношении транзакции шины, которая включает в себя отслеживаемый адрес.

17. Процессор по п.16, дополнительно содержащий логические средства трансляции адресов для трансляции упомянутого ассоциированного адресного операнда в отслеживаемый адрес, который представляет собой физический адрес.

18. Процессор по п.17, в котором отслеживаемый адрес выбран из набора, состоящего из физического адреса, виртуального адреса, относительного адреса и линейного адреса.

19. Процессор по п.1, дополнительно содержащий множество разделяемых ресурсов, подлежащих разбиению на разделы таким образом, чтобы выделить часть каждого разделяемого ресурса каждому активному потоку из упомянутого множества потоков, когда несколько потоков активны, при этом упомянутые средства приостановки освобождают любой из упомянутого множества разделов, выделенных для первого потока, в качестве реакции на приостановку выполнения упомянутого первого потока.

20. Процессор по п.19, в котором средство мониторинга вызывает повторное разбиение на разделы упомянутого множества разделяемых ресурсов, чтобы приспособиться к исполнению первого потока, в качестве реакции на доступ к памяти по отслеживаемому адресу.

21. Процессор по п.20, в котором множество разделяемых ресурсов содержит очередь команд; буфер переупорядочения; пул регистров; множество буферов сохранения.

22. Процессор по п.21, дополнительно содержащий множество дублированных ресурсов, причем упомянутое множество дублированных ресурсов дублируется для каждого из множества потоков и упомянутое множество дублированных ресурсов содержит множество переменных состояния процессора; указатель команды; логические средства переименования регистров.

23. Процессор по п.22, дополнительно содержащий множество совместно используемых ресурсов, которое доступно для использования любым из упомянутого множества потоков, причем упомянутое множество совместно используемых ресурсов включает в себя множество исполняющих средств; кэш; планировщик.

24. Система для обработки машинных команд, содержащая процессор по любому из пп.1-23; память для хранения команд для упомянутого множества потоков.

25. Система по п.24, дополнительно содержащая многопоточную операционную систему; интерфейс мультимедиа.

26. Процессор, содержащий препроцессор для приема первой команды и второй команды, причем первая команда имеет операнд, указывающий отслеживаемый адрес; множество ресурсов исполнения для выполнения первой команды и второй команды и для входа в зависящее от реализации состояние в качестве реакции на вторую команду, если первая команда была выполнена; монитор, предназначенный для того, чтобы вызвать выход из зависящего от реализации состояния в качестве реакции на доступ к памяти по отслеживаемому адресу.

27. Процессор по п.26, в котором множество ресурсов исполнения входит в зависящее от реализации состояние в качестве реакции на вторую команду, если первая команда была выполнена и не произошло событий принудительного останова после выполнения первой команды, но перед выполнением второй команды.

28. Процессор по п.26 или 27, в котором упомянутый операнд представляет собой неявный операнд из заранее определенного регистра, указывающий линейный адрес, при этом процессор дополнительно содержит логические средства трансляции адресов для

трансляции упомянутого линейного адреса с целью получения отслеживаемого адреса, который представляет собой физический адрес.

29. Процессор по п.26, дополнительно содержащий логические средства когерентности, предназначенные для гарантирования того, что никакой кэш в другом процессоре, подключенном к процессору, не хранит информацию по упомянутому отслеживаемому адресу в модифицированном или исключительном состоянии.

30. Процессор по п.29, дополнительно содержащий средства для формирования сигнала обращения в качестве реакции на слежение за отслеживаемым адресом со стороны другого процессора.

31. Процессор по п.26, в котором первая команда и вторая команда представляют собой команды из первого потока множества потоков, и упомянутое зависящее от реализации состояние включает в себя состояние без разбиения на разделы, в котором разбитые на разделы ресурсы для первого потока освобождены.

32. Процессор, содержащий множество разделяемых между потоками ресурсов для приема команд; множество совместно используемых ресурсов для выполнения команды совместно с упомянутым множеством разделяемых между потоками ресурсов; логические средства приостановки потока для приостановки первого потока в качестве реакции на команду в первом потоке, причем упомянутые логические средства приостановки потока освобождают разделы упомянутого множества разделяемых между потоками ресурсов, связанные с первым потоком, в дополнение к приостановке первого потока; монитор, предназначенный для того, чтобы вызвать выполнение процессором повторного разбиения множества разделяемых между потоками ресурсов на разделы и возобновления выполнения первого потока в качестве реакции на доступ к адресу памяти, указанному первым потоком.

33. Процессор по п.32, в котором доступ к адресу памяти задан первой командой, выполняемой в упомянутом первом потоке, и монитор демаскирован для сигнализации об отслеживаемых событиях, чтобы вызвать возобновление потока командой, в качестве реакции на которую логические средства приостановки потока приостанавливают первый поток.

34. Процессор по п.32 или 33, в котором множество разделяемых ресурсов включает в себя очередь команд; буфер переупорядочения; пул регистров; множество буферов сохранения, причем упомянутый процессор дополнительно содержит множество дублированных ресурсов, причем упомянутое множество дублированных ресурсов дублируется для каждого из упомянутого множества потоков, при этом упомянутое множество дублированных ресурсов содержит множество переменных состояния процессора; указатель команды; логические средства переименования регистров.

35. Способ обработки машинных команд, содержащий этапы, на которых принимают первый код операции в первом потоке исполнения, причем упомянутый первый код операции имеет ассоциированный с ним адресный операнд, указывающий отслеживаемый адрес; выполняют мониторинг доступов к памяти по отслеживаемому адресу в качестве реакции на первый код операции; приостанавливают упомянутый первый поток; обнаруживают доступ к памяти по отслеживаемому адресу; возобновляют упомянутый первый поток в качестве реакции на обнаружение доступа к памяти по отслеживаемому адресу.

36. Способ по п.35, в котором этап приостановки первого потока включает в себя этапы, на которых принимают вторую команду в первом потоке; приостанавливают первый поток в качестве реакции на вторую команду.

37. Способ по п.35 или 36, в котором доступ к памяти представляет собой доступ для записи.

38. Способ по п.37, дополнительно содержащий этапы, на которых транслируют упомянутый ассоциированный адресный операнд в отслеживаемый физический адрес, причем этап обнаружения доступа к памяти по отслеживаемому адресу включает в себя этап, на котором обнаруживают доступ для записи по отслеживаемому физическому адресу.

39. Способ по п.38, дополнительно содержащий этап, на котором предотвращают получение другими абонентами монопольного использования информации, хранящейся по отслеживаемому адресу.

40. Способ по п.35, в котором этап обнаружения включает в себя этапы, на которых принимают информацию цикла из внешней транзакции шины; обнаруживают операции записи по отслеживаемому адресу.

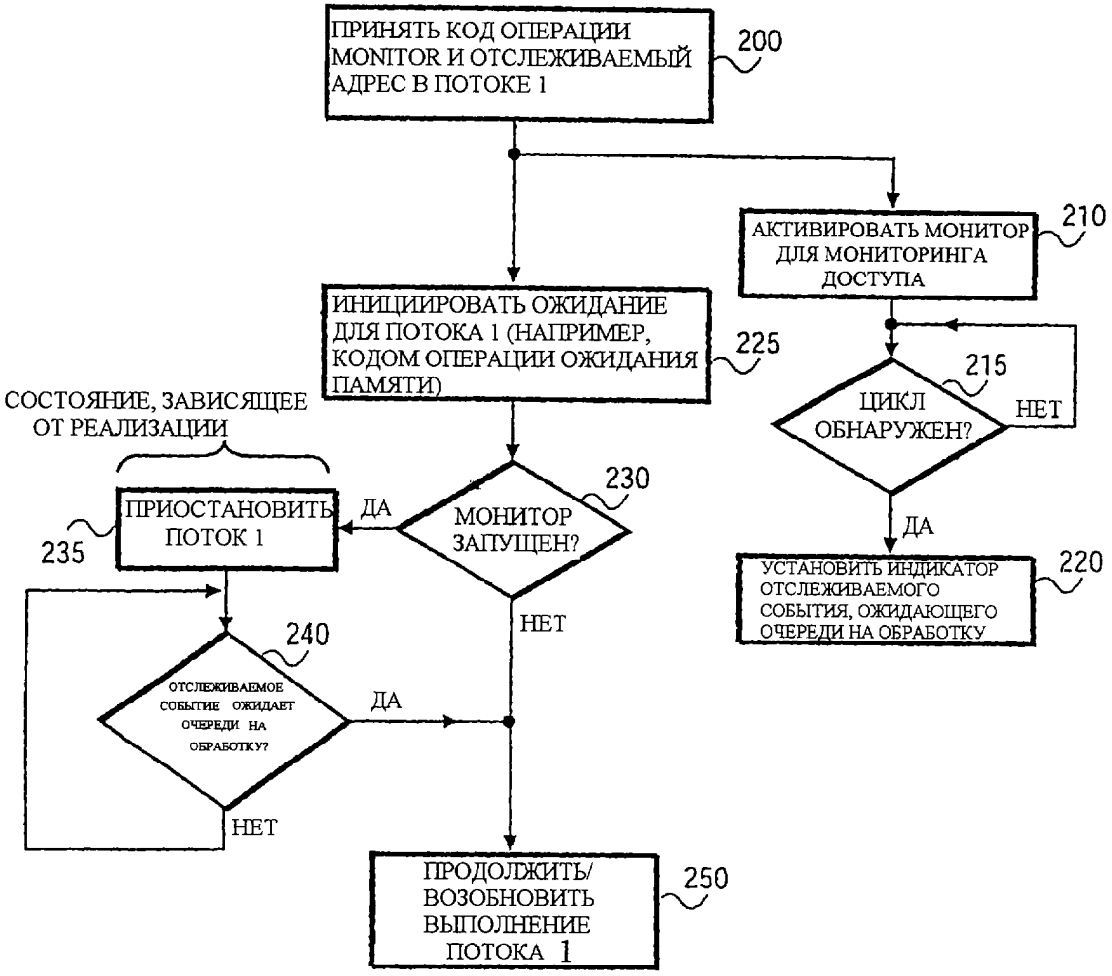
41. Способ по п.35, дополнительно содержащий этап, на котором возобновляют упомянутый первый поток в качестве реакции на событие, отличающееся от доступа к памяти по отслеживаемому адресу.

42. Способ по п.41, в котором упомянутое событие представляет собой прерывание.

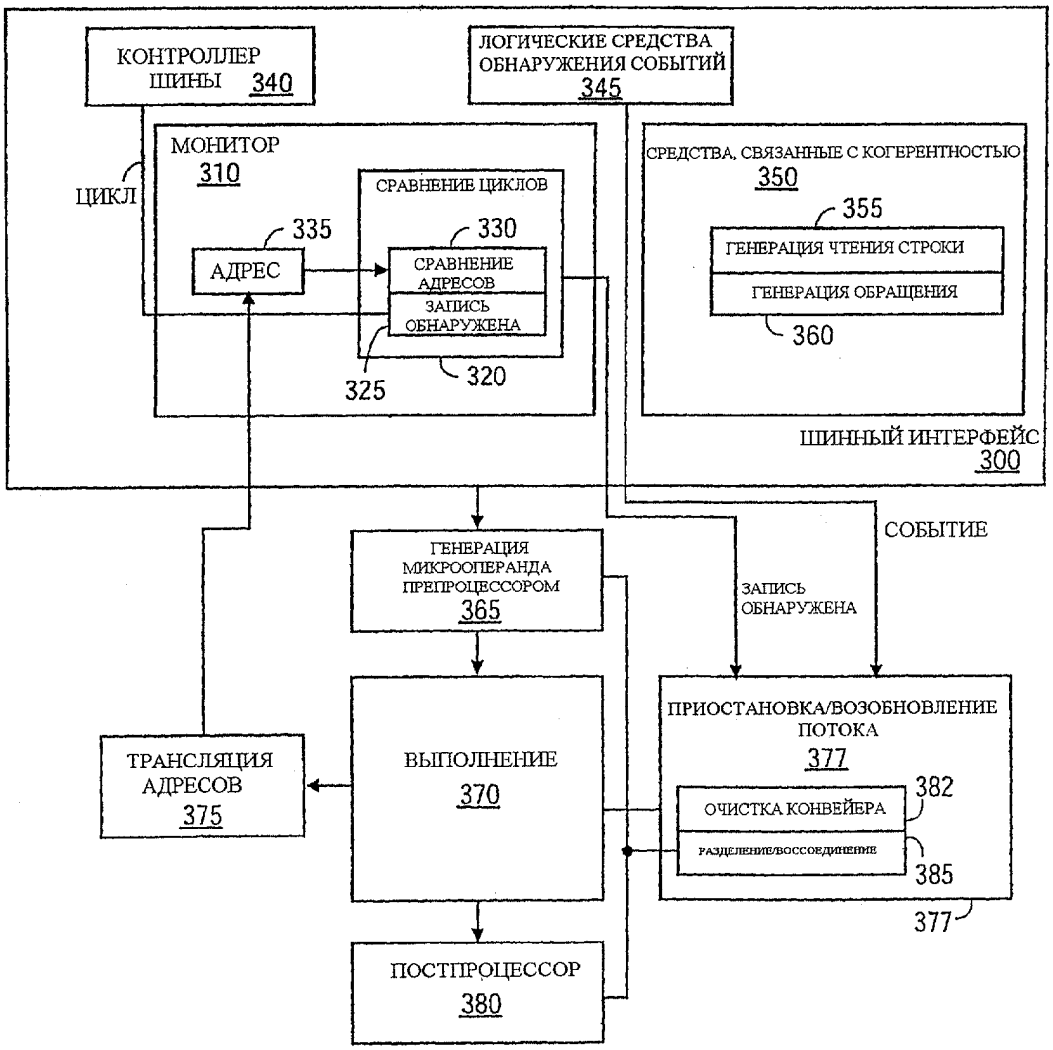
43. Способ по п.42, в котором упомянутое прерывание представляет собой маскированное прерывание, что указано вторым операндом, которое тем не менее должно рассматриваться как событие принудительного останова.

44. Способ по п.35, дополнительно содержащий этап, на котором транслируют линейный адрес, ассоциированный с первым кодом операции, в физический адрес.

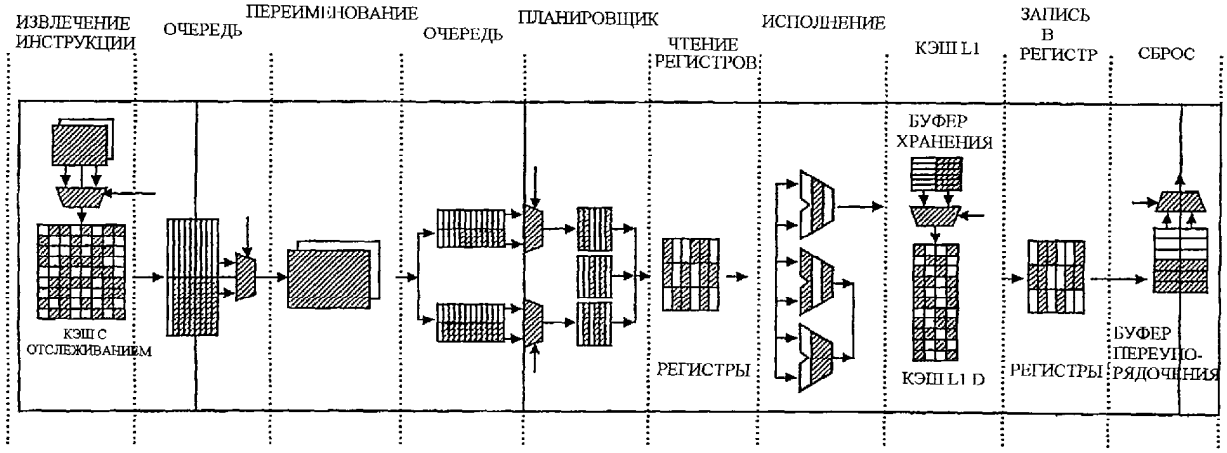
45. Способ по п.44, дополнительно содержащий этап, на котором сигнализируют об обращении, если другой абонент шины считывает по упомянутому физическому адресу.


ПРИНЯТЬ КОД ОПЕРАЦИИ MONITOR И ОТСЛЕЖИВАЕМЫЙ АДРЕС В ПОТОКЕ 1
200
АКТИВИРОВАТЬ МОНИТОР ДЛЯ МОНИТОРИНГА ДОСТУПА
210
ИНИЦИИРОВАТЬ ОЖИДАНИЕ ДЛЯ ПОТОКА 1 (НАПРИМЕР, КОДОМ ОПЕРАЦИИ ОЖИДАНИЯ ПАМЯТИ)
225
ЦИКЛ ОБНАРУЖЕН?
215
НЕТ
ДА
СОСТОЯНИЕ, ЗАВИСЯЩЕЕ ОТ РЕАЛИЗАЦИИ
ПРИОСТАНОВИТЬ ПОТОК 1
235
ДА
МОНИТОР ЗАПУЩЕН?
230
УСТАНОВИТЬ ИНДИКАТОР ОТСЛЕЖИВАЕМОГО СОБЫТИЯ, ОЖИДАЮЩЕГО ОЧЕРЕДИ НА ОБРАБОТКУ
220
НЕТ
240
ОТСЛЕЖИВАЕМОЕ СОБЫТИЕ ОЖИДАЕТ ОЧЕРЕДИ НА ОБРАБОТКУ?
ДА
НЕТ
ПРОДОЛЖИТЬ/ ВОЗОБНОВИТЬ ВЫПОЛНЕНИЕ ПОТОКА 1
250

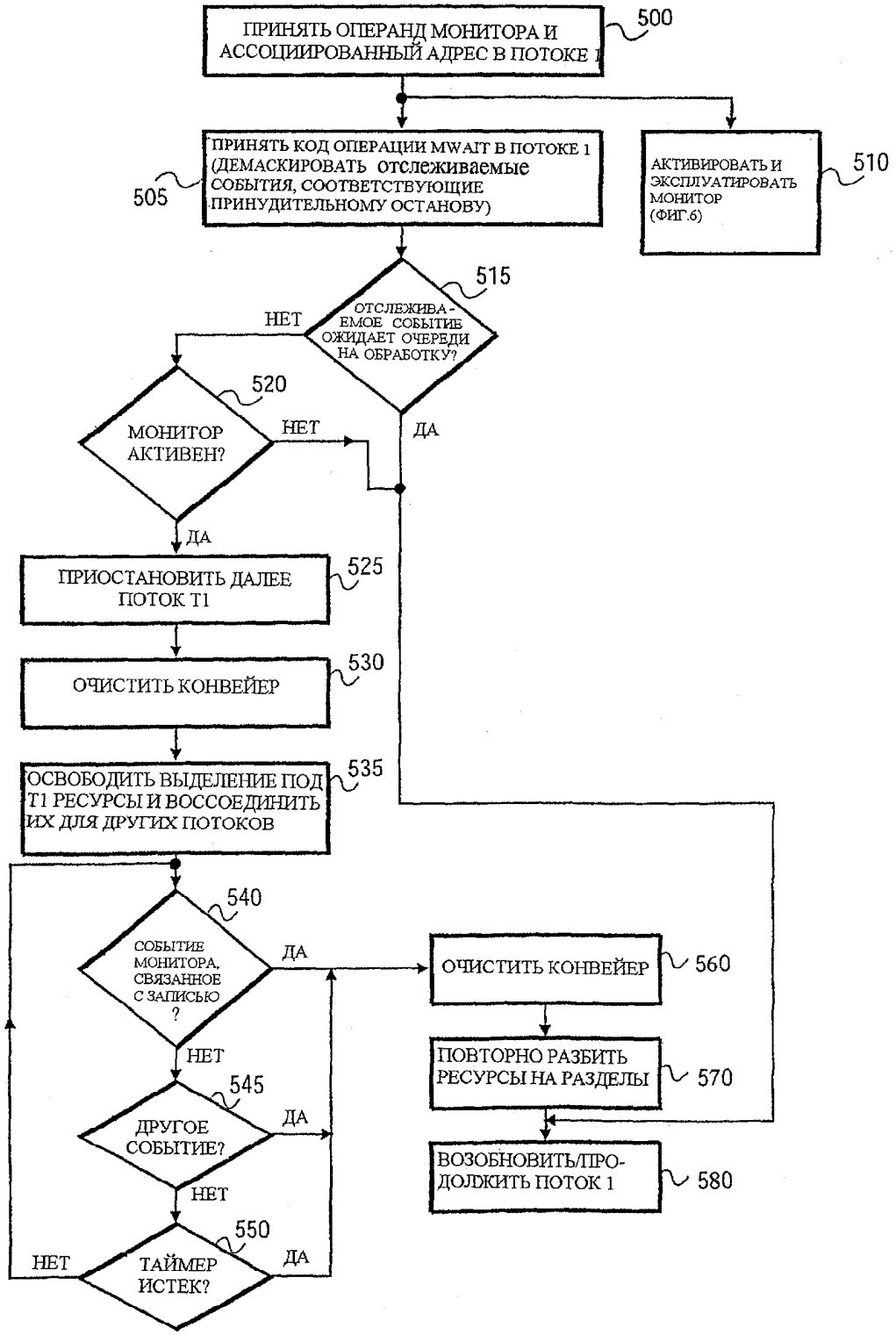
ФИГ.2


КОНТРОЛЛЕР ШИНЫ 340
ЛОГИЧЕСКИЕ СРЕДСТВА ОБНАРУЖЕНИЯ СОБЫТИЙ 345
МОНИТОР 310
ЦИКЛ
СРАВНЕНИЕ ЦИКЛОВ
335
330
АДРЕС
СРАВНЕНИЕ АДРЕСОВ
ЗАПИСЬ ОБНАРУЖЕНА
325
320
СРЕДСТВА, СВЯЗАННЫЕ С КОГЕРЕНТНОСТЬЮ 350
355
ГЕНЕРАЦИЯ ЧТЕНИЯ СТРОКИ
ГЕНЕРАЦИЯ ОБРАЩЕНИЯ
360
ШИННЫЙ ИНТЕРФЕЙС 300
СОБЫТИЕ
ГЕНЕРАЦИЯ МИКРООПЕРАНДА ПРЕПРОЦЕССОРОМ 365
ЗАПИСЬ ОБНАРУЖЕНА
ПРИОСТАНОВКА/ВОЗОБНОВЛЕНИЕ ПОТОКА 377
ТРАНСЛЯЦИЯ АДРЕСОВ 375
ВЫПОЛНЕНИЕ 370
ОЧИСТКА КОНВЕЙЕРА
382
РАЗДЕЛЕНИЕ/ВОССОЕДИНЕНИЕ
385
377
ПОСТПРОЦЕССОР 380

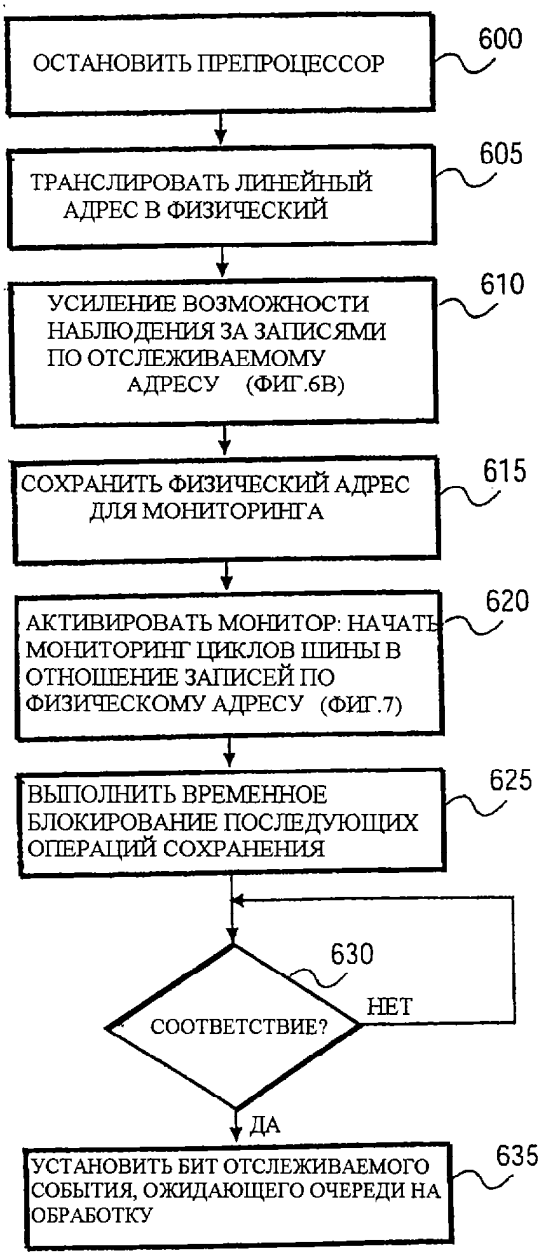
ФИГ.3


ИЗВЛЕЧЕНИЕ ИНСТРУКЦИИ
ОЧЕРЕДЬ
ПЕРЕИМЕНОВАНИЕ
ОЧЕРЕДЬ
ПЛАНИРОВЩИК
ЧТЕНИЕ РЕГИСТРОВ
ИСПОЛНЕНИЕ
КЭШ L1
ЗАПИСЬ В РЕГИСТР
СБРОС
КЭШ С ОТСЛЕЖИВАНИЕМ
РЕГИСТРЫ
БУФЕР ХРАНЕНИЯ
КЭШ L1 D
РЕГИСТРЫ
БУФЕР ПЕРЕУПОРЯДОЧЕНИЯ

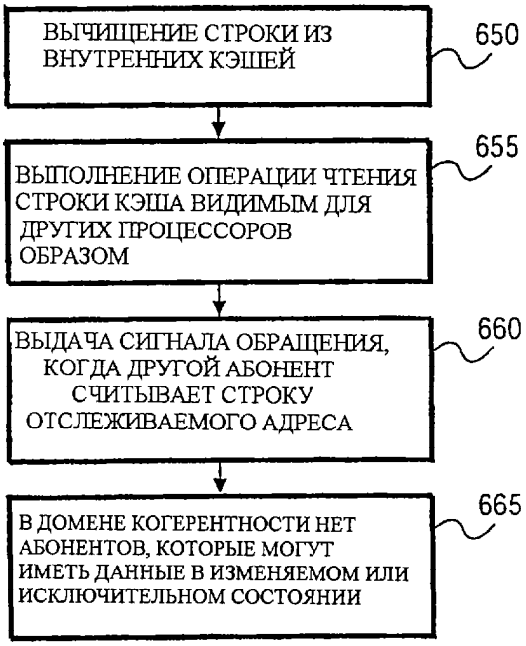
ФИГ.4


ПРИНЯТЬ ОПЕРАНД МОНИТОРА И АССОЦИИРОВАННЫЙ АДРЕС В ПОТОКЕ 1
500
ПРИНЯТЬ КОД ОПЕРАЦИИ MWAIT В ПОТОКЕ 1 (ДЕМАСКИРОВАТЬ отслеживаемые СОБЫТИЯ, СООТВЕТСТВУЮЩИЕ ПРИНУДИТЕЛЬНОМУ ОСТАНОВУ)
505
АКТИВИРОВАТЬ И ЭКСПЛУАТИРОВАТЬ МОНИТОР (ФИГ.6)
510
ОТСЛЕЖИВАЕМОЕ СОБЫТИЕ ОЖИДАЕТ ОЧЕРЕДИ НА ОБРАБОТКУ?
515
НЕТ
ДА
МОНИТОР АКТИВЕН?
520
НЕТ
ДА
ПРИОСТАНОВИТЬ ДАЛЕЕ ПОТОК T1
525
ОЧИСТИТЬ КОНВЕЙЕР
530
ОСВОБОДИТЬ ВЫДЕЛЕНИЕ ПОД T1 РЕСУРСЫ И ВОССОЕДИНИТЬ ИХ ДЛЯ ДРУГИХ ПОТОКОВ
535
СОБЫТИЕ МОНИТОРА, СВЯЗАННОЕ С ЗАПИСЬЮ ?
540
ДА
НЕТ
ОЧИСТИТЬ КОНВЕЙЕР
560
ПОВТОРНО РАЗБИТЬ РЕСУРСЫ НА РАЗДЕЛЫ
570
ДРУГОЕ СОБЫТИЕ?
545
ДА
НЕТ
ВОЗОБНОВИТЬ/ПРОДОЛЖИТЬ ПОТОК 1
580
ТАЙМЕР ИСТЕК?
550
НЕТ
ДА

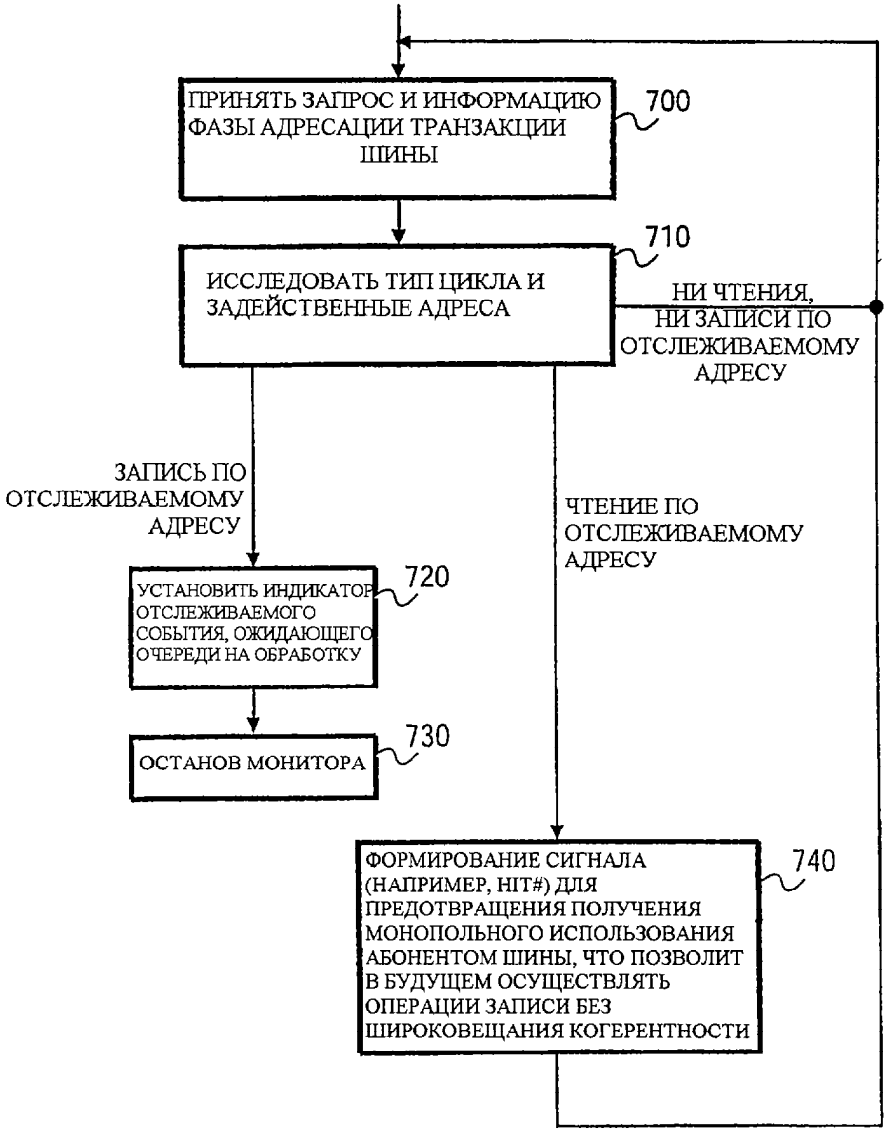
ФИГ.5


ОСТАНОВИТЬ ПРЕПРОЦЕССОР
600
ТРАНСЛИРОВАТЬ ЛИНЕЙНЫЙ АДРЕС В ФИЗИЧЕСКИЙ
605
УСИЛЕНИЕ ВОЗМОЖНОСТИ НАБЛЮДЕНИЯ ЗА ЗАПИСЯМИ ПО ОТСЛЕЖИВАЕМОМУ АДРЕСУ (ФИГ.6В)
610
СОХРАНИТЬ ФИЗИЧЕСКИЙ АДРЕС ДЛЯ МОНИТОРИНГА
615
АКТИВИРОВАТЬ МОНИТОР: НАЧАТЬ МОНИТОРИНГ ЦИКЛОВ ШИНЫ В ОТНОШЕНИЕ ЗАПИСЕЙ ПО ФИЗИЧЕСКОМУ АДРЕСУ (ФИГ.7)
620
ВЫПОЛНИТЬ ВРЕМЕННОЕ БЛОКИРОВАНИЕ ПОСЛЕДУЮЩИХ ОПЕРАЦИЙ СОХРАНЕНИЯ
625
630
СООТВЕТСТВИЕ?
НЕТ
ДА
УСТАНОВИТЬ БИТ ОТСЛЕЖИВАЕМОГО СОБЫТИЯ, ОЖИДАЮЩЕГО ОЧЕРЕДИ НА ОБРАБОТКУ
635

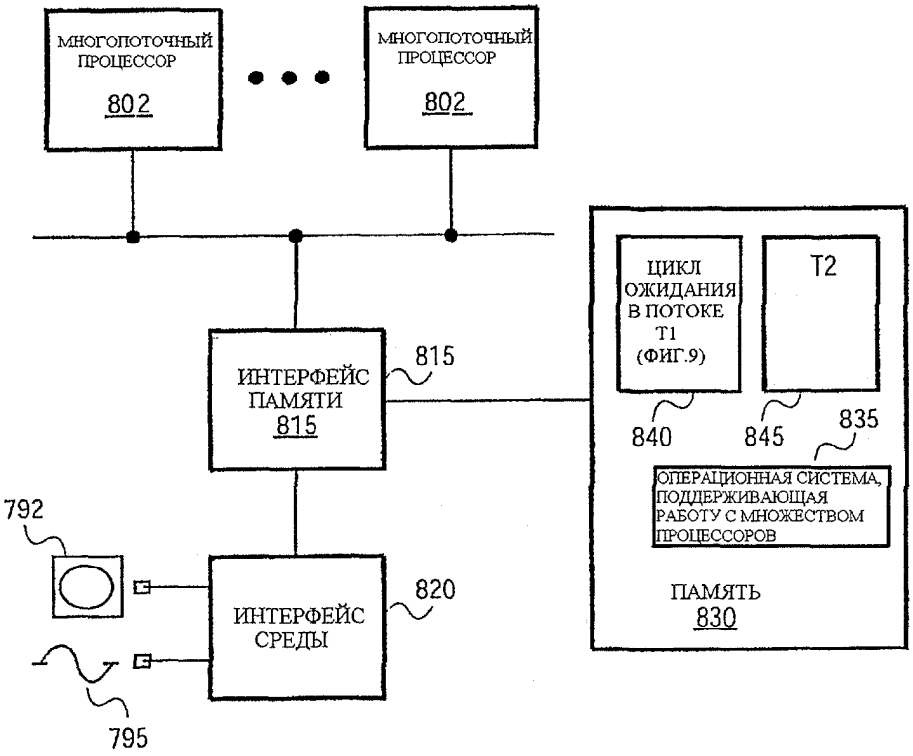
ФИГ.6А


ВЫЧИЩЕНИЕ СТРОКИ ИЗ ВНУТРЕННИХ КЭШЕЙ
650
ВЫПОЛНЕНИЕ ОПЕРАЦИИ ЧТЕНИЯ СТРОКИ КЭША ВИДИМЫМ ДЛЯ ДРУГИХ ПРОЦЕССОРОВ ОБРАЗОМ
655
ВЫДАЧА СИГНАЛА ОБРАЩЕНИЯ, КОГДА ДРУГОЙ АБОНЕНТ СЧИТЫВАЕТ СТРОКУ ОТСЛЕЖИВАЕМОГО АДРЕСА
660
В ДОМЕНЕ КОГЕРЕНТНОСТИ НЕТ АБОНЕНТОВ, КОТОРЫЕ МОГУТ ИМЕТЬ ДАННЫЕ В ИЗМЕНЯЕМОМ ИЛИ ИСКЛЮЧИТЕЛЬНОМ СОСТОЯНИИ
665

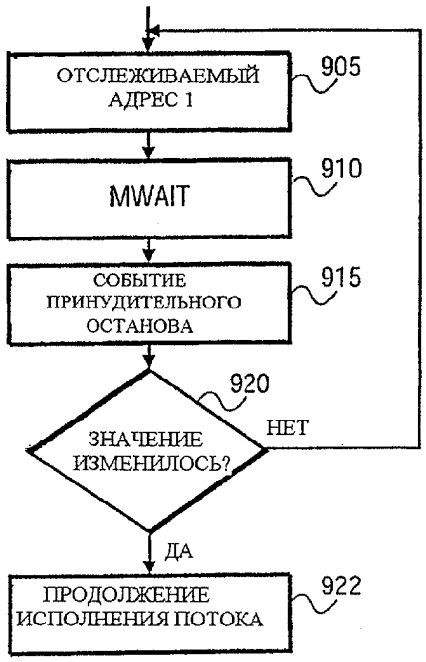
ФИГ.6В


ПРИНЯТЬ ЗАПРОС И ИНФОРМАЦИЮ ФАЗЫ АДРЕСАЦИИ ТРАНЗАКЦИИ ШИНЫ
700
ИССЛЕДОВАТЬ ТИП ЦИКЛА И ЗАДЕЙСТВЕННЫЕ АДРЕСА
710
НИ ЧТЕНИЯ, НИ ЗАПИСИ ПО ОТСЛЕЖИВАЕМОМУ АДРЕСУ
ЗАПИСЬ ПО ОТСЛЕЖИВАЕМОМУ АДРЕСУ
ЧТЕНИЕ ПО ОТСЛЕЖИВАЕМОМУ АДРЕСУ
УСТАНОВИТЬ ИНДИКАТОР ОТСЛЕЖИВАЕМОГО СОБЫТИЯ, ОЖИДАЮЩЕГО ОЧЕРЕДИ НА ОБРАБОТКУ
720
ОСТАНОВ МОНИТОРА
730
ФОРМИРОВАНИЕ СИГНАЛА (НАПРИМЕР, HIT#) ДЛЯ ПРЕДОТВРАЩЕНИЯ ПОЛУЧЕНИЯ МОНОПОЛЬНОГО ИСПОЛЬЗОВАНИЯ АБОНЕНТОМ ШИНЫ, ЧТО ПОЗВОЛИТ В БУДУЩЕМ ОСУЩЕСТВЛЯТЬ ОПЕРАЦИИ ЗАПИСИ БЕЗ ШИРОКОВЕЩАНИЯ КОГЕРЕНТНОСТИ
740

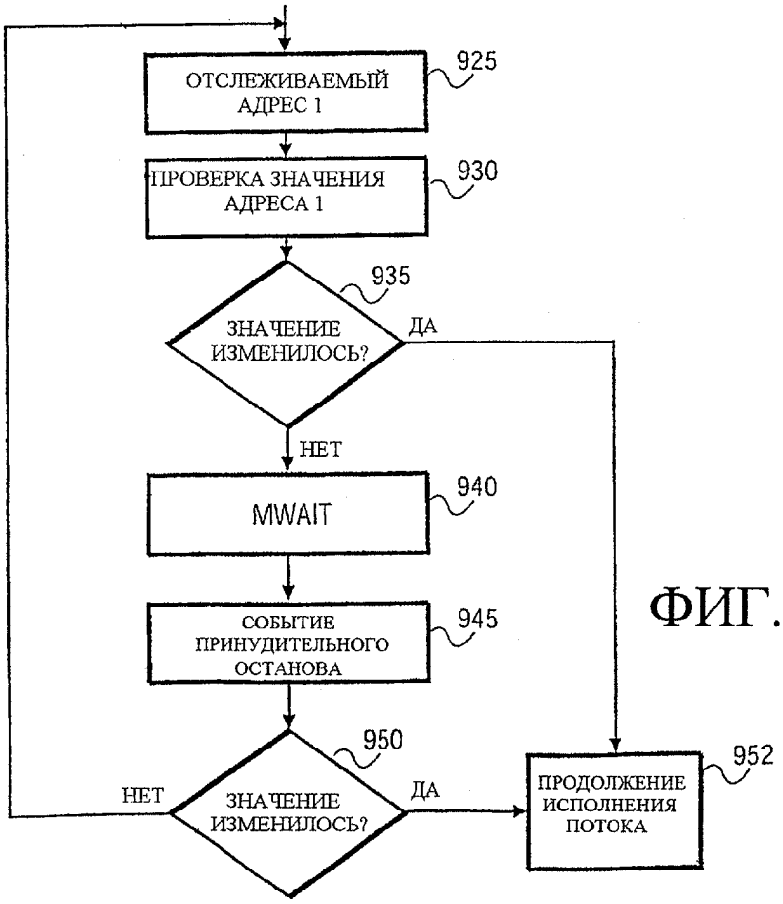
ФИГ.7


МНОГОПОТОЧНЫЙ ПРОЦЕССОР
802
МНОГОПОТОЧНЫЙ ПРОЦЕССОР
802
ИНТЕРФЕЙС ПАМЯТИ
815
815
ЦИКЛ ОЖИДАНИЯ В ПОТОКЕ T1 (ФИГ.9)
T2
840
845
835
ОПЕРАЦИОННАЯ СИСТЕМА, ПОДДЕРЖИВАЮЩАЯ РАБОТУ С МНОЖЕСТВОМ ПРОЦЕССОРОВ
ПАМЯТЬ
830
792
ИНТЕРФЕЙС СРЕДЫ
820
795

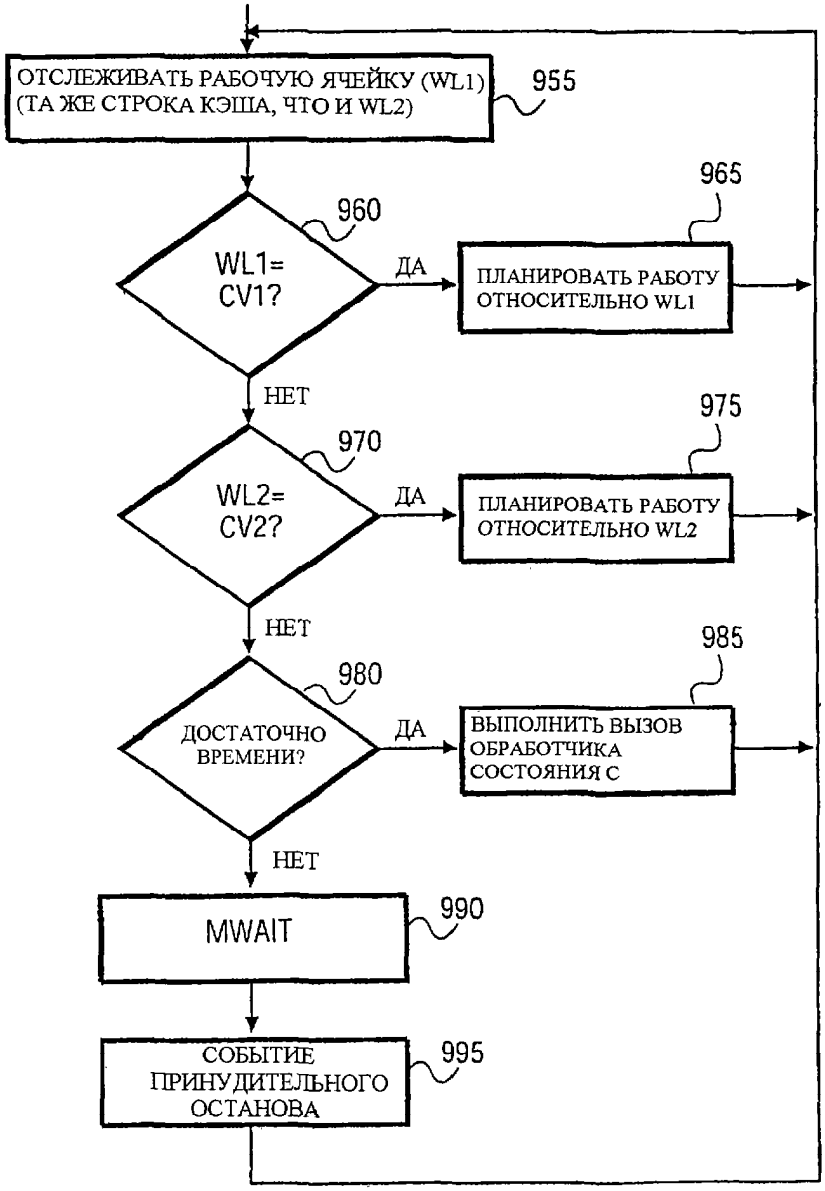
ФИГ.8


ОТСЛЕЖИВАЕМЫЙ АДРЕС 1
905
MWAIT
910
СОБЫТИЕ ПРИНУДИТЕЛЬНОГО ОСТАНОВА
915
ЗНАЧЕНИЕ ИЗМЕНИЛОСЬ?
920
НЕТ
ДА
ПРОДОЛЖЕНИЕ ИСПОЛНЕНИЯ ПОТОКА
922

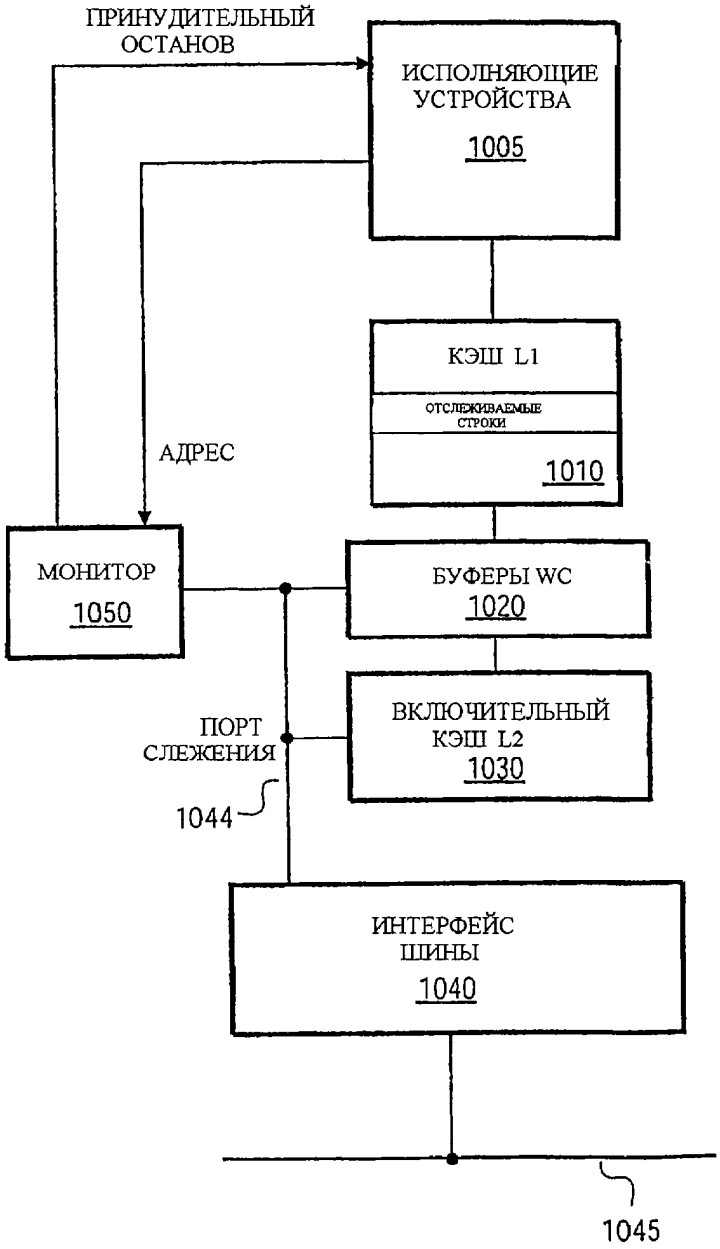
ФИГ.9А


ОТСЛЕЖИВАЕМЫЙ АДРЕС 1
925
ПРОВЕРКА ЗНАЧЕНИЯ АДРЕСА 1
930
ЗНАЧЕНИЕ ИЗМЕНИЛОСЬ?
935
ДА
НЕТ
MWAIT
940
СОБЫТИЕ ПРИНУДИТЕЛЬНОГО ОСТАНОВА
945
ЗНАЧЕНИЕ ИЗМЕНИЛОСЬ?
950
НЕТ
ДА
ПРОДОЛЖЕНИЕ ИСПОЛНЕНИЯ ПОТОКА
952

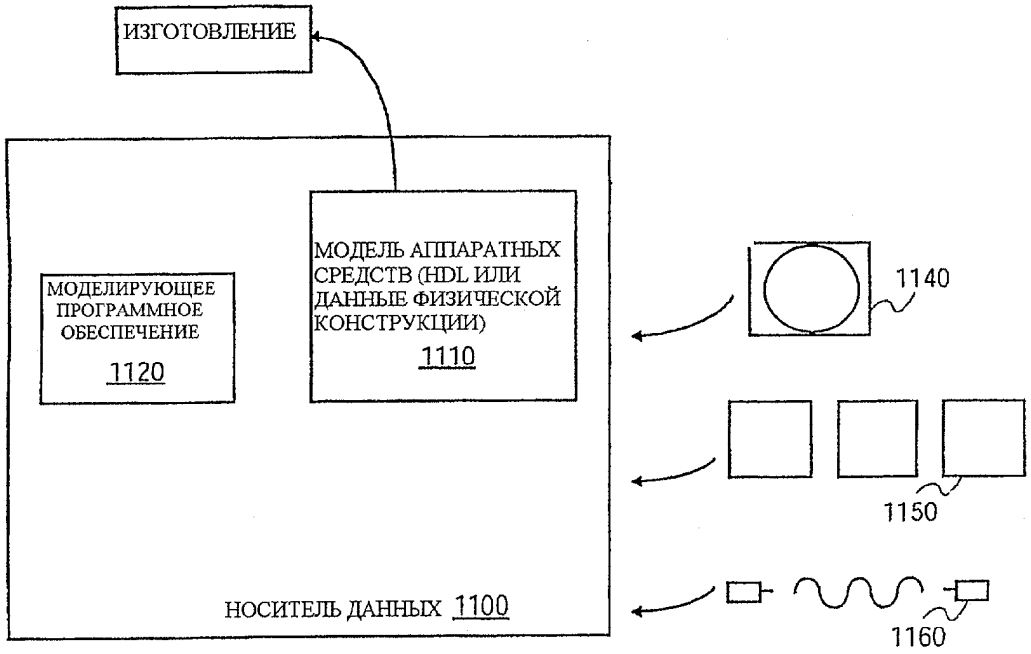
ФИГ.9В


ОТСЛЕЖИВАТЬ РАБОЧУЮ ЯЧЕЙКУ (WL1)
(ТА ЖЕ СТРОКА КЭША, ЧТО И WL2)
955
960
WL1=
CV1?
ДА
965
ПЛАНИРОВАТЬ РАБОТУ
ОТНОСИТЕЛЬНО WL1
НЕТ
970
WL2=
CV2?
ДА
975
ПЛАНИРОВАТЬ РАБОТУ
ОТНОСИТЕЛЬНО WL2
НЕТ
980
ДОСТАТОЧНО
ВРЕМЕНИ?
ДА
985
ВЫПОЛНИТЬ ВЫЗОВ
ОБРАБОТЧИКА
СОСТОЯНИЯ C
НЕТ
MWAIT
990
СОБЫТИЕ
ПРИНУДИТЕЛЬНОГО
ОСТАНОВА
995

ФИГ.9С


ПРИНУДИТЕЛЬНЫЙ ОСТАНОВ
ИСПОЛНЯЮЩИЕ УСТРОЙСТВА
1005
КЭШ L1
ОТСЛЕЖИВАЕМЫЕ СТРОКИ
1010
АДРЕС
МОНИТОР
1050
БУФЕРЫ WC
1020
ВКЛЮЧИТЕЛЬНЫЙ КЭШ L2
1030
ПОРТ СЛЕЖЕНИЯ
1044
ИНТЕРФЕЙС ШИНЫ
1040
1045

ФИГ.10


ИЗГОТОВЛЕНИЕ
МОДЕЛЬ АППАРАТНЫХ СРЕДСТВ (HDL ИЛИ ДАННЫЕ ФИЗИЧЕСКОЙ КОНСТРУКЦИИ)
1110
МОДЕЛИРУЮЩЕЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ
1120
НОСИТЕЛЬ ДАННЫХ 1100
1140
1150
1160

ФИГ.11