РОССИЙСКАЯ ФЕДЕРАЦИЯ

(19) **RU** (11) **2 427 889** (13) **C2**

(51) МПК
G06F 9/30 (2006.01)
G06F 15/16 (2006.01)

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ,
ПАТЕНТАМ И ТОВАРНЫМ ЗНАКАМ

(12) ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ПАТЕНТУ

(21)(22) Заявка: **2009131712/08, 24.01.2008**

(24) Дата начала отсчета срока действия патента:
24.01.2008

Приоритет(ы):
(30) Конвенционный приоритет:
24.01.2007 US 11/626,443

(43) Дата публикации заявки: **27.02.2011** Бюл. № 6

(45) Опубликовано: **27.08.2011** Бюл. № **24**

(56) Список документов, цитированных в отчете о поиске: **US 5586278 A, 17.12.1996. US 5627985 A, 06.05.1997. US 5812839 A, 22.09.1998. SU 1300471 A1, 30.03.1987.**

(85) Дата начала рассмотрения заявки PCT на национальной фазе: **24.08.2009**

(86) Заявка PCT:
US 2008/051966 (24.01.2008)

(87) Публикация заявки PCT:
WO 2008/092045 (31.07.2008)

Адрес для переписки:
129090, Москва, ул. Б.Спасская, 25, стр.3, ООО "Юридическая фирма Городисский и Партнеры", пат.пов. Ю.Д.Кузнецову, рег.№ 595

(72) Автор(ы):
МАКИЛВЕЙН Майкл Скотт (US),
ДИФФЕНДЕРФЕР Джеймс Норрис (US),
САРТОРИУС Томас Эндрю (US),
СМИТ Родни Уэйн (US)

(73) Патентообладатель(и):
КВЭЛКОММ ИНКОРПОРЕЙТЕД (US)

(54) ОЧИСТКА СЕГМЕНТИРОВАННОГО КОНВЕЙЕРА ДЛЯ НЕВЕРНО ПРЕДСКАЗАННЫХ ПЕРЕХОДОВ

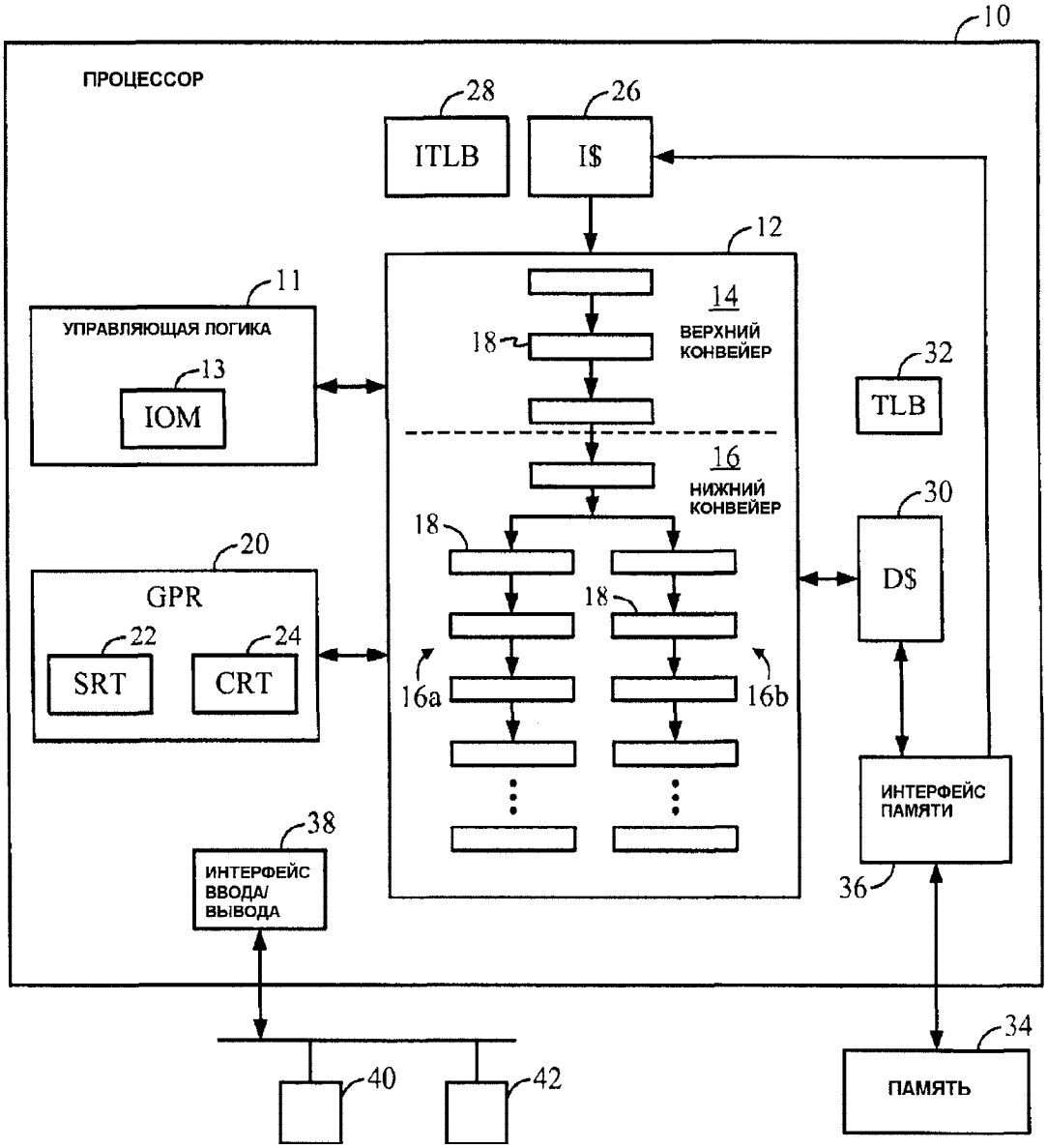
(57) Реферат:

Изобретение относится к предсказанию перехода в конвейерных процессорах и, в частности, к системе и способу, предназначенным для независимой очистки двух сегментов одного и того же конвейера в разные моменты времени. Техническим результатом является повышение производительности процессора. Конвейер процессора сегментируют на верхнюю часть, до команд, идущих не в программном порядке, и одну или более нижних частей после верхней части. Верхний конвейер очищают после обнаружения того, что команда перехода была неверно предсказана, минимизируя задержку при выборке команд из целевого адреса верного перехода. Нижние конвейеры могут продолжать выполнение до тех пор, пока не подтвердится команда неверно предсказанного перехода, причем в этот момент времени все незафиксированные команды очищают из нижних конвейеров. Могут быть использованы

RU 2427889 C2

RU 2427889 C2

существующие механизмы исключения очистки конвейера с помощью добавления идентификатора неверно предсказанного перехода, при этом уменьшаются сложность и стоимость аппаратного обеспечения очистки нижних конвейеров. 2 н. и 14 з.п. ф-лы, 2 ил.



Фиг. 1

RU 2427889 C2

RU 2427889 C2

RUSSIAN FEDERATION

FEDERAL SERVICE
FOR INTELLECTUAL PROPERTY,
PATENTS AND TRADEMARKS

(19) **RU** (11) **2 427 889** (13) **C2**

(51) Int. Cl.
G06F 9/30 (2006.01)
G06F 15/16 (2006.01)

(12) ABSTRACT OF INVENTION

(21)(22) Application: **2009131712/08, 24.01.2008**

(24) Effective date for property rights:
24.01.2008

Priority:
(30) Priority:
24.01.2007 US 11/626,443

(43) Application published: **27.02.2011** Bull. 6

(45) Date of publication: **27.08.2011 Bull. 24**

(85) Commencement of national phase: **24.08.2009**

(86) PCT application:
US 2008/051966 (24.01.2008)

(87) PCT publication:
WO 2008/092045 (31.07.2008)

Mail address:
129090, Moskva, ul. B.Spasskaja, 25, str.3, OOO "Juridicheskaja firma Gorodisskij i Partnery", pat.pov. Ju.D.Kuznetsovu, reg.№ 595

(72) Inventor(s):
MAKILVEJN Majkl Skott (US),
DIFFENDERFER Dzhejms Norris (US),
SARTORIUS Tomas Ehndrju (US),
SMIT Rodni Uehjn (US)

(73) Proprietor(s):
KVEhLKOMM INKORPOREJTED (US)

RU 2427889 C2

(54) CLEANING OF SEGMENTED CONVEYOR FOR WRONGLY PREDICTED TRANSITIONS

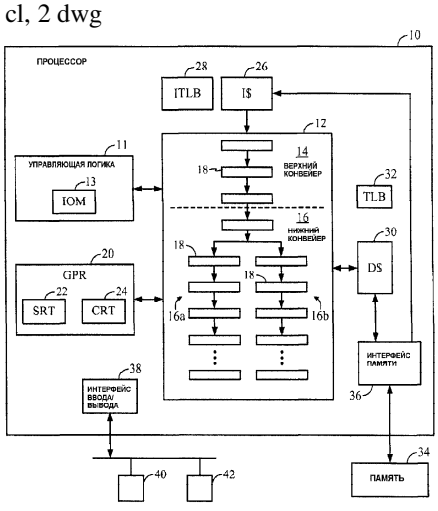
(57) Abstract:

FIELD: information technologies.

SUBSTANCE: processor conveyor is segmented into an upper part, prior to commands that follow not in a program order, and one or more lower parts downstream the upper part. The upper conveyor is cleaned after detection of the fact that the transition command was wrongly predicted, minimising delay in selection of commands from a target address of the right transition. Lower conveyors may continue execution until the command of the wrongly predicted transition is confirmed, besides, at this moment of time all non-fixed commands are cleaned from the lower conveyors. Existing mechanisms of conveyor cleaning elimination may be used with the help of adding the identifier of the wrongly predicted transition, at the same time complexity and cost of hardware for lower conveyors cleaning are less.

EFFECT: increased efficiency of processor.

16 cl, 2 dwg



Фиг. 1

Уровень техники

Настоящее изобретение в целом относится к предсказанию перехода в конвейерных процессорах и, в частности, к системе и способу, предназначенным для независимой очистки двух сегментов одного и того же конвейера в разные моменты времени, чтобы максимизировать производительность и эффективно корректировать команды неверно предсказанного перехода.

Большинство современных процессоров используют конвейерную архитектуру, в которой последовательные команды, причем каждая из которых имеет множество этапов выполнения, перекрываются при выполнении. Для максимальной производительности команды должны проходить непрерывно через конвейер. Однако команды часто останавливаются в конвейере из-за множества причин, таких как зависимости данных между командами, задержки, связанные с доступами к памяти, невозможность назначать достаточные ресурсы конвейера командам и тому подобное. Минимизация остановок конвейера и их эффективное разрешение являются важными факторами при достижении улучшенной производительности процессора.

Большинство практических программ включают в себя команды условного перехода, фактическое поведение ветвления которых не известно до тех пор, пока команду не оценят глубоко в конвейере. Большинство современных процессоров используют различные виды предсказания перехода, причем поведение ветвления команд условного перехода предсказывают на ранней стадии в конвейере, и процессор выбирает и спекулятивно выполняет команды, спекулятивно назначая им ресурсы конвейера на основании предсказания перехода. Когда определяют фактическое поведение перехода, если переход был неверно предсказан, спекулятивно выбранные команды должны быть очищены из конвейера, спекулятивно назначенные ресурсы должны быть отменены и возвращены в их состояние до предсказания перехода, и новые команды должны быть выбраны из целевого адреса верного перехода.

С одной стороны, идеально конвейер должен бы быть очищен немедленно после обнаружения неверного предсказания таким образом, что верные команды могли быть выбраны и запущены в конвейер с минимизацией задержки, вызванной неверным предсказанием перехода. С другой стороны, команды, выбранные из ошибочного маршрута перехода, могут находиться на разных этапах спекулятивного выполнения, и им могут быть спекулятивно назначены различные ресурсы процессора. Немедленное "ослабление" этих назначений, чтобы восстановить ресурсы в их состояние до предсказания перехода, является трудным и могут произойти многочисленные циклы процессора, и/или потребоваться многочисленные дублированные копии спекулятивно назначенных ресурсов. Потери, претерпеваемые при немедленной очистке конвейера, дополнительно обостряются в процессорах, которые поддерживают выполнение команд не по порядку, таких как суперскалярные процессоры, вследствие дополнительной проблемы отслеживания относительного времени жизни команды. Это отслеживание времени жизни команды необходимо, чтобы гарантировать, что очищаются только те команды, которые были выбраны после команды неверно предсказанного перехода (в программном порядке), и что выполняют все команды до команды перехода (в программной последовательности), даже если они могут быть после команды перехода в конвейере.

US 5,586,278 описывает способ и устройство для восстановления состояния, следующего за неверно предсказанным переходом в процессоре, работающем в не программном порядке.

US 5,812,839 описывает систему с двойным предсказанием перехода, имеющую два

этапа обработки восстановления перехода, которая задействуется, только когда неверно предсказанный переход является самой старой командой в блоке, работающем не в программном порядке.

US 5,627,985 описывает файлы спекулятивных и фиксированных ресурсов в процессоре, работающем в не программном порядке.

Сущность изобретения

После определения неверного предсказания перехода процессор может облегчить задачу очистки своего конвейера от ошибочно выбираемых и спекулятивно выполняемых команд с помощью продолжения обычного выполнения в течение (идеально) только нескольких циклов и использования существующего аппаратного обеспечения, которое обрабатывает очистки конвейера в случае исключений. Этот процесс может существенно уменьшить затраты и сложность очистки конвейера в ответ на неверное предсказание перехода. Однако дополнительная внесенная задержка противоречит цели немедленной очистки конвейера, чтобы быстро выбирать команды из целевого адреса верного перехода.

Согласно изобретению предоставляется способ по п.1.

Согласно изобретению предоставляется процессор по п.10.

В соответствии с одним или более вариантами осуществления верхнюю часть конвейера, до команд, идущих не в программном порядке, очищают немедленно после обнаружения, что команда перехода была неверно предсказана, предусматривая немедленную очистку команд из целевого адреса верного перехода. Каждая нижняя часть конвейера может продолжать выполнение до тех пор, пока команда неверно предсказанного перехода не подтвердится, причем в этот момент времени все незавершенные команды очищаются из нижнего конвейера. Могут быть использованы существующие механизмы исключения очистки конвейера с помощью добавления идентификатора неверно предсказанного перехода, уменьшающие сложность и стоимость аппаратного обеспечения очистки нижнего конвейера.

Один вариант осуществления относится к способу выполнения команд в конвейерном процессоре, который позволяет выполнение не по порядку. После обнаружения неверного предсказания команды перехода очищают верхний конвейер, т.е. верхнюю часть конвейера до точки, в которой команды могут быть не в программном порядке. После фиксации команды неверно предсказанного перехода для выполнения все незафиксированные команды очищаются из нижнего конвейера, т.е. части конвейера, за которой команды могут быть не в программном порядке.

Другой вариант осуществления относится к процессору. Процессор включает в себя управляющую логику и сегментированный конвейер выполнения команд, содержащий верхний конвейер до точки, в которой команды могут быть не в программном порядке, и один или более нижних конвейеров за точкой, в которой команды могут быть не в программном порядке. Управляющая логика действует для очистки всех команд из верхнего конвейера после обнаружения неверно предсказанного перехода и дополнительно действует для очистки всех незафиксированных команд из нижних конвейеров после фиксации выполнения команды неверно предсказанного перехода.

Краткий перечень чертежей

На фиг.1 показана функциональная блок-схема одного варианта осуществления суперскалярного конвейерного процессора, имеющего очистку сегментированного конвейера для неверных предсказаний перехода.

На фиг.2 показана схема последовательности этапов способа выполнения команд в процессоре по фиг.1.

Подробное описание изобретения

Фиг.1 изображает функциональную блок-схему процессора 10, имеющего очистку сегментированного конвейера для неверно предсказанного перехода. Процессор 10 выполняет команды в конвейере 12 выполнения команд в соответствии с управляющей логикой 11, которая включает в себя механизм 13 упорядочивания команд (IOM). Конвейер 12 логически разделяют, как объяснено более подробно ниже, на верхнюю часть 14 и нижнюю часть 16. Конвейер 12 может быть суперскалярной конструкцией с множеством параллельных нижних конвейеров 16a, 16b. Конвейеры 14, 16 включают в себя различные регистры или защелки 18, организованные в ступенях конвейера, а также логические и вычислительные схемы, такие как арифметико-логическое устройство (ALU) (не изображено). Файл 20 регистра общего назначения (GPR) обеспечивает регистры, содержащие вершину иерархии памяти. Как обсуждается в данном документе, GPR 20 может включать в себя таблицу 22 спекулятивного переименования (SRT) и таблицу 24 зафиксированного переименования (CRT), чтобы спекулятивно назначать ресурсы GPR командам.

Верхний конвейер 14 выбирает команды из кэша 26 команд (I-кэша или I$) посредством обращения к памяти и полномочий доступа к памяти, управляемых с помощью буфера 28 предыстории трансляции команд (ITLB). К данным осуществляют доступ из кэша 30 данных (D-кэша или D$) посредством обращения к памяти и полномочий доступа к памяти, управляемых с помощью главного буфера 32 предыстории трансляции (TLB). В разных вариантах осуществления ITLB 28 может содержать копию части TLB 32. В качестве альтернативы ITLB 28 и TLB 32 могут быть интегрированы. Подобным образом в разных вариантах осуществления процессор 10, I-кэш 26 и D-кэш 30 могут быть объединены.

Неудачи при поиске в I-кэше 26 и/или D-кэше 30 являются причиной доступа к главной (вне кристалла) памяти 34 под управлением интерфейса 36 памяти (другие кэши, не изображенные, могут быть помещены между процессором 10 и главной памятью 34). Процессор 10 может включать в себя интерфейс 38 ввода/вывода (I/O), управляющий доступом к различным периферийным устройствам 40, 42. Специалисты в данной области техники поймут, что возможны многочисленные варианты процессора 10. Например, процессор 10 может включать в себя кэш второго уровня (L2) либо для каждого, либо для обоих кэшей I и D 26, 30. Кроме того, один или более из функциональных блоков, изображенных в процессоре 10, могут быть не включены в конкретный вариант осуществления.

Логическое разделение конвейера 12 на верхний конвейер 14 и один или более нижних конвейеров 16 определяют с помощью достоверности, с которой команды могут быть немедленно очищены из верхнего конвейера 14, безотносительно к порядку команд. В частности, верхний конвейер 14 определяют как часть конвейера 12 до точки, в которой команды могут идти не в программном порядке. Подобным образом нижний конвейер 16 определяют как часть конвейера 12 после или следующим за верхним конвейером 14. В процессорах, которые поддерживают выполнение не по порядку, команды в нижнем конвейере 16 могут быть отправлены в ступени конвейера в порядке, отличном от порядка программы. В большинстве вариантов осуществления верхний конвейер 14 будет содержать все ступени выборки и декодирования, а нижний конвейер 16 включает в себя ступень выдачи команд (в случае суперскалярного конвейера) и один или более конвейеров выполнения.

Это логическое разделение конвейера 12 позволяет верхнему конвейеру 14 быть

очищенным немедленно после обнаружения неверного предсказания перехода, таким образом минимизируя время, требуемое для того, чтобы выбирать команды из целевого адреса верного перехода. Это дополнительно позволяет нижнему конвейеру 16 продолжать выполнение, таким образом используют преимущество существующих механизмов исключения очистки, чтобы эффективно очищать нижний конвейер 16 от всех команд, спекулятивно выбранных после неверно предсказанного перехода.

Очистка верхнего конвейера 14 является прямой. Поскольку верхний конвейер 14 определяют как расположенный до точки, в которой команды могут идти не в программном порядке, и определение неверно предсказанного перехода происходит в ступени выполнения в нижнем конвейере 16, известно, что все команды в верхнем конвейере 14 в момент времени определения неверного предсказания являются более новыми, чем команда неверно предсказанного перехода. То есть все они были бы очищены в расчете на предсказание перехода, и могут быть все безопасно очищены, таким образом, как в цикле после обнаружения неверно предсказанного перехода. Это позволяет процессору 10 как можно раньше начать команды очистки из целевого адреса верного перехода, при этом минимизируют продолжительность остановки конвейера.

Очистка нижнего конвейера 16 является более проблематичной. Поскольку команды могут выполняться не в программном порядке, нельзя допустить, что все команды после неверно предсказанного перехода в конвейере являются более новыми, чем команда неверно предсказанного перехода, и являются безопасными для очистки. Например, заявитель предлагает рассмотреть следующие команды: LD, ADD, BR, где LD предоставляет операнд для ADD, но BR не зависит от любой из двух команд. Операция LD требует доступа к памяти, который может остановить конвейер 16, в частности, в случае неудачи при поиске в D-кэше 30. Конечно, ADD должен ждать завершения LD. Однако независимая команда BR может быть направлена в нижний конвейер 16 для выполнения раньше LD и ADD. Если оценка условия BR обнаруживает, что она была неверно предсказана, процессор не может просто очистить нижний конвейер 16 после команды BR. Это очистило бы LD и ADD, обе из которых предшествуют BR в программном порядке и должны быть выполнены.

Большинство суперскалярных процессоров 10 включают в себя администратора порядка команд (IOM) в качестве части управляющей логики 11 конвейера. IOM отслеживает порядок выполнения команд через конвейер, то есть какая команда является более старой или более новой, чем данная команда, с помощью поддержания представлений команд в правильном программном порядке таким образом, как в кольцевом буфере, FIFO или тому подобном. Посредством атрибутов или флагов, связанных с представлениями команд, IOM дополнительно отслеживает зависимости команд и является инструментальным средством при обработке исключения. В соответствии с одним или более вариантами осуществления IOM 13 в процессоре 10 включает в себя дополнительные схемы и логику, которые облегчают эффективную очистку нижнего конвейера после обнаружения команды неверно предсказанного перехода. Краткий обзор роли IOM 13 при обработке исключения разъяснит очистку неверно предсказанного перехода нижнего конвейера 16 в соответствии с вариантами осуществления, раскрытыми в настоящей заявке и приведенными в формуле изобретения.

Исключение происходит каждый раз, когда ступень конвейера не может завершить свое выполнение этапа команды. Например, команда сохранения, записывающая

данные в память, может вызвать исключение, если просмотр TLB 32 указывает, что страница памяти доступна только по чтению. Другие типы исключений широко известны в данной области техники. После столкновения с исключением процессор 10 должен выполнить все предыдущие, все более старые команды в конвейере 12; очистить команду, вызывающую исключение и все более новые команды из конвейера 12; а затем выбрать и выполнить код обработки прерывания. IOM 13 помогает в этом процессе с помощью отслеживания, какие команды являются “подтвержденными”, а какие “зафиксированными”.

Команду подтверждают, когда определяют, что никакие опасности конвейера не будут препятствовать ее выполнению, то есть команда не будет останавливаться. Например, команда, выполняющая арифметическую или логическую операцию, может быть подтверждена, когда известно, что оба операнда должны быть сгенерированы из предыдущих команд, выбраны из памяти или являются доступными другим способом. Команда фиксируется, когда эта команда и все более старые команды являются подтвержденными. Известно, что зафиксированная команда может завершить выполнение, так как никакие опасности конвейера не препятствуют либо ей (сама команда является подтвержденной), либо любой команде раньше нее (все более старые команды являются подтвержденными). Все зафиксированные команды должны быть выполнены.

Традиционным правилом во время обработки исключения является то, что все незафиксированные команды очищают из конвейера 12, когда команда, вызывающая исключение, является “последней незафиксированной командой”. То есть все команды до команды, вызывающей исключение, зафиксированы для выполнения, но команда, вызывающая исключение, и все команды, более новые, чем она, не зафиксированы. Незафиксированные команды (включая команду, вызывающую исключение) очищают и продолжают выполнение зафиксированных команд. Новые команды выбирают из адреса обработчика прерывания.

В отличие от команды, вызывающей исключение, команда неверно предсказанного перехода должна быть выполнена, а не очищена. Только тогда, когда предсказание перехода было ошибочным, сама команда перехода должна быть выполнена и должна направить прохождение программы в целевой адрес соответствующего перехода. Таким образом, все незафиксированные команды очищаются из всех нижних конвейеров 16a, 16b, когда команда не предсказанного перехода является “самой новой зафиксированной командой”. То есть команда неверно предсказанного перехода и все команды до того, как она зафиксировалась для выполнения, но все команды, следующие после перехода (команды, выбранные из целевого адреса ошибочно предсказанного перехода), остались незафиксированными. Незафиксированные команды очищаются, а зафиксированные команды (включая команду неверно предсказанного перехода) продолжают выполнение.

В одном или более вариантах осуществления каждый элемент в IOM 13 включает в себя флаг неверно предсказанного перехода (MPB) или поле битов, который инициализируется в неутвержденное состояние, когда создается элемент IOM 13. Команда перехода обычно будет подтверждаться немедленно после оценки ее перехода, так как она не имеет зависимостей с другими данными. Ступень конвейера подтверждения следует за ступенью оценки условия перехода в каждом конвейере 16a, 16b, в котором выполняются команды перехода, и устанавливает подтвержденный флаг в IOM 13. Ступень конвейера подтверждения дополнительно сравнивает оценку перехода с ее предсказанным значением, а также устанавливает

флаг MPB в IOM 13, если переход был неверно предсказан. После столкновения с флагом MPB при фиксации команд для выполнения IOM 13 и связанная управляющая логика 11 может осуществить очистку нижнего конвейера 16 в соответствии с правилом “самые новые завершенные команды”, описанном выше.

Задержка от обнаружения неверного предсказания до подтверждения и фиксации команды неверно предсказанного перехода в IOM 13 обычно должна занимать только несколько циклов. Между тем немедленная очистка верхнего конвейера 14 означает, что выборка команды из целевого адреса верного перехода уже началась. В некоторых случаях, таких как пример LD, ADD, BR, описанный выше, может быть существенная задержка между обнаружением неверного предсказания перехода и подтверждением команды перехода, что дает возможность очистки нижнего конвейера 16. В качестве меры предосторожности против возможности того, что очистка нижнего конвейера 16 занимает больше циклов, чем, например, глубина верхнего конвейера 14, блокировка конвейера может быть помещена на границе или до границы между верхним конвейером 14 и нижним конвейером 16. Блокировка, которую удаляют, когда завершают очистку нижнего конвейера 16, предупреждает ошибочную очистку любых команд, выбранных из целевого адреса верного перехода. Условные блокировки конвейера широко известны в данной области техники и могут быть без труда осуществлены специалистами в данной области техники без дополнительного объяснения в настоящем описании.

Другой сложностью при очистке нижнего конвейера 16 является восстановление верного состояния спекулятивно назначенных ресурсов процессора после очистки. Например, переименование регистра является известным способом администрирования GPR 20, в котором логические идентификаторы (r0, r1, r2, …) GPR динамически отображаются в большой набор физических регистров посредством отображения в таблице переименования. Системы переименования регистра избегают многие из опасностей данных, присущих выполнению команды не по порядку. Во время операции назначают новый физический регистр, и новое “переименование” логического в физический записывают в таблице переименования для каждой команды, которая записывает регистр GPR 20. Команды, которые считывают GPR 20, транслируют свой логический идентификатор GPR в номер физического регистра посредством просмотра таблицы переименования. Номер физического регистра остается связанным с командой считывания регистра по всему его пребыванию через конвейер 16.

В системе переименования регистра GPR 20 записывает “не портить” до значений, записанных в том же самом логическом идентификаторе GPR; запись направляют в новый неиспользованный физический регистр. Команды, которые следуют после команды записи в программном порядке, направляют в новый физический регистр и получают записанное значение. Команды, предшествующие команде записи в программном порядке, были отображены с помощью таблицы переименования в другой физический регистр (до операции переименования) и будут продолжаться, чтобы осуществлять доступ к этому физическому регистру. Таким образом, команды, которые записывают данный идентификатор GPR, могут быть выполнены раньше команд, которые считывают предыдущее значение из этого идентификатора GPR (опасность записи после считывания или WaR), или записывают предыдущий результат в него (опасность записи после записи или WaW). Таким образом, можно избежать опасности WaR и Waw данных.

Переименование логических идентификаторов GPR в физические регистры является

спекулятивным назначением ресурсов процессора. Поскольку команды могут быть выполнены не по порядку, команда может записывать логический идентификатор GPR (и ей может быть назначен новый физический регистр) до выполнения другой команды, которая предшествует команде записи в программном порядке. Если другая команда вызывает исключение или является командой неверно предсказанного перехода, команда записи может быть очищена из конвейера 16, а ее физическому регистру отменено назначение. С другой стороны, переименования регистров, которые предшествуют команде записи, должны быть сохранены.

Чтобы иметь возможность очищать нижний конвейер 16 в любой момент времени (например, немедленно после обнаружения неверно предсказанного перехода) и восстанавливать, отдельная копия таблицы переименования должна создаваться каждый раз, когда логический идентификатор GPR переименовывают в физический регистр, т.е. каждый раз, когда команду записи регистра запускают в конвейер 16, и копия должна быть поддержана до тех пор, пока команда записи регистра не зафиксируется для выполнения. Эти таблицы регистров затем могут быть выборочно отброшены, на основании которых команды очищают из конвейера 16. Этот подход является дорогим как в поддержании многочисленных копий таблиц переименования, так и в логическом отслеживании, какие таблицы переименования должны быть отброшены при очистке конвейера 16.

В одном или более вариантах осуществления поддерживают только две копии таблицы переименования, таблицу 22 спекулятивного переименования (SRT) и таблицу 24 зафиксированного переименования (CRT). SRT 22 обновляют каждый раз, когда логический идентификатор GPR переименовывают в новый физический регистр. Отображения переименования регистра в CRT 24 обновляют, только когда соответственные команды записи регистра фиксируются для выполнения. Когда нижний конвейер 16 очищают вследствие неверно предсказанного перехода, поскольку команда неверно предсказанного перехода является самой новой зафиксированной командой, известно, что все команды записи регистра, которые зафиксировались раньше перехода (в программном порядке), записали свои преобразования отображения регистра в CRT 24. Кроме того, известно, что любое переименование регистра, выполненное с помощью команд записи регистра, которые следуют после команды неверно предсказанного перехода (в программном порядке), записано только в SRT 22, и не записано в CRT 24. Эти переименования регистров должны быть отброшены, а связанным физическим регистрам отменено назначение как часть очистки нижнего конвейера 16.

В одном или более вариантах осуществления, когда нижний конвейер 16 очищают вследствие неверно предсказанного перехода, когда команда неверно предсказанного перехода является самой новой зафиксированной командой, CRT 24 копируют в SRT 22. Это помещает SRT 22 в верное состояние, начиная с команды неверно предсказанного перехода, и отменяет назначение физических регистров, которые были спекулятивно назначены очищенным командам записи регистра. С помощью поддержания только двух таблиц переименования и отсрочки очистки нижнего конвейера 16 до тех пор, пока не зафиксируется команда неверно предсказанного перехода, значительно упрощается задача отмены назначения спекулятивно назначенных ресурсов процессора.

Фиг.2 изображает способ выполнения команд в конвейерном процессоре 10, который позволяет выполнение не по порядку, в соответствии с одним или более вариантами осуществления. Соответственный способ начинается, когда процессор 10

обнаруживает неверно предсказанный переход (MPB) (этап 100). Обычно это будет возникать на ступени выполнения (EXE) в нижнем конвейере 16a, 16b. Немедленно после обнаружения MPB процессор 10 очищает верхний конвейер 14 (этап 102), например, в следующем цикле, и начинает выборку команд из целевого адреса верного перехода (этап 114). Затем процессор 10 помещает блокировку в верхнем конвейере 14 (этап 116) до тех пор, пока нижний конвейер 16 не будет очищен. Когда блокировку освобождают, продолжается обычное выполнение процессора 10 (этап 118).

Параллельно процессор 10 продолжает выполнение в нижнем конвейере 16 до тех пор, пока не подтвердится команда MPB (этап 120). Это подтверждение обычно будет возникать на ступени конвейера немедленно после обнаружения неверного предсказания перехода. Когда команда MPB подтверждается (этап 120), подтвержденный флаг и флаг MPB устанавливают в IOM 13 (этап 122). Обычная обработка продолжатся до тех пор, пока не зафиксируется команда MPB. Когда команда MPB является самой новой зафиксированной командой (этап 124), IOM 13 инициирует очистку незафиксированных команд из нижнего конвейера 16 (этап 126). Затем процессор 10 копирует CRT 24 в SRT 22 и продолжает обычное выполнение (этап 118).

Таким образом, варианты осуществления выполняют как немедленную очистку верхнего конвейера 14, при этом минимизируют задержку при выборке команд из целевого адреса верхнего перехода, так и эффективную очистку нижнего конвейера 16, при этом исключают сложные вычисления и дублированное аппаратное обеспечение, чтобы правильно очищать только команды, выбранные из ошибочного целевого адреса перехода, и отменять назначение ресурсов процессора, спекулятивно назначенное им. Несмотря на то что варианты осуществления описаны в настоящей заявке со ссылкой на IOM 13, который осуществляет очистку исключения, настоящее изобретение не ограничено этими вариантами осуществления. В частности, механизм отслеживания команд, описанный в настоящей заявке, чтобы очищать все незавершенные команды из нижнего конвейера 16, когда команда неверно предсказанного перехода становится самой новой зафиксированной командой, не должен также выполнять очистку исключения. Кроме того, несмотря на то что спекулятивное назначение ресурсов процессора 10 описано в данном документе относительно схемы переименования регистра, в данной области техники известны многочисленные другие виды спекулятивного назначения ресурсов, такие как буферы переименования и тому подобные.

Конечно, настоящее изобретение может быть выполнено другими способами, отличными от способов, конкретно изложенных в данном документе, не выходя за существенные характеристики изобретения. Настоящие варианты осуществления должны рассматриваться во всех аспектах как иллюстративные, а не ограничивающие, и подразумевается, что все изменения осуществляются в пределах объема прилагаемой формулы изобретения.

Формула изобретения

1. Способ выполнения команд в конвейерном процессоре, который позволяет выполнение не по порядку, содержащий этапы, на которых после обнаружения неверного предсказания команды перехода очищают (102) верхний конвейер до точки, в которой команды могут быть не в программном порядке, и

после фиксации команды неверно предсказанного перехода для выполнения очищают (126) все незафиксированные команды из нижнего конвейера за точкой, в

которой команды могут быть не в программном порядке.

2. Способ по п.1, дополнительно содержащий этап, на котором указывают команду неверно предсказанного перехода как таковую в механизме упорядочивания команд после ее подтверждения.

3. Способ по п.2, в котором этап, на котором очищают (126) все незафиксированные команды из нижнего конвейера, содержит этап, на котором очищают команды в ответ на указатель команды неверно предсказанного перехода в механизме упорядочивания команд, когда команда неверно предсказанного перехода является самой новой зафиксированной командой.

4. Способ по п.2, в котором процессор включает в себя два или более конвейеров и в котором каждый конвейер, осуществляющий выполнение команды перехода, включает в себя ступень подтверждения, действующую для установки указателя команды неверно предсказанного перехода в механизме упорядочивания команд.

5. Способ по п.1, дополнительно содержащий этап, на котором после очистки верхнего конвейера выбирают команды из целевого адреса команды верного перехода.

6. Способ по п.5, дополнительно содержащий этап, на котором после очистки (102) верхнего конвейера останавливают верхний конвейер в точке или до точки, в которой команды могут идти не по порядку, до тех пор, пока незафиксированные команды не будут очищены из нижнего конвейера.

7. Способ по п.6, в котором спекулятивно назначенные ресурсы включают в себя ресурсы переименования регистров.

8. Способ по п.1, дополнительно содержащий этап, на котором после очистки (126) незафиксированных команд из нижнего конвейера отменяют назначение ресурсов, спекулятивно назначенных незафиксированным командам.

9. Способ по п.8, в котором отмена назначения ресурсов, спекулятивно назначенных незафиксированным командам, включает в себя этап, на котором копируют содержимое таблицы зафиксированного переименования регистра в таблицу спекулятивного переименования регистра.

10. Процессор (10), содержащий

управляющую логику (11) и

сегментированный конвейер (12) выполнения команд, содержащий верхний конвейер (14) до точки, в которой команды могут быть не в программном порядке, и один или более нижних конвейеров (16) после точки, в которой команды могут быть не в программном порядке,

причем управляющая логика (11) действует для очистки всех команд из верхнего конвейера (14) после обнаружения неверно предсказанного перехода, отличающийся тем, что управляющая логика дополнительно действует для очистки всех незафиксированных команд из нижних конвейеров (16) после фиксации команды неверно предсказанного перехода для выполнения.

11. Процессор (10) по п.10, в котором управляющая логика (11) дополнительно действует для выбора команд из целевого адреса верного перехода в цикле после очистки верхнего конвейера.

12. Процессор (10) по п.11, в котором управляющая логика (11) дополнительно действует для размещения блокировки в конце верхнего конвейера (14) после очистки верхнего конвейера и удаления блокировки после очистки всех незафиксированных команд из нижних конвейеров (16).

13. Процессор (10) по п.10, в котором управляющая логика (11) включает в себя механизм упорядочивания команд, действующий для отслеживания подтвержденного

и зафиксированного статуса команд и дополнительно действующий для очистки всех незафиксированных команд из нижних конвейеров (16), когда команда неверно предсказанного перехода является самой новой подтвержденной командой.

14. Процессор (10) по п.13, в котором каждый элемент в механизме упорядочивания команд включает в себя указатель того, является ли связанная команда командой неверно предсказанного перехода.

15. Процессор (10) по п.14, в котором каждый нижний конвейер (16), который выполняет команды перехода, включает в себя состояние подтверждения, действующее для установки указателя неверно предсказанного перехода в механизме упорядочивания команд, когда определяют, что команда перехода неверно предсказана.

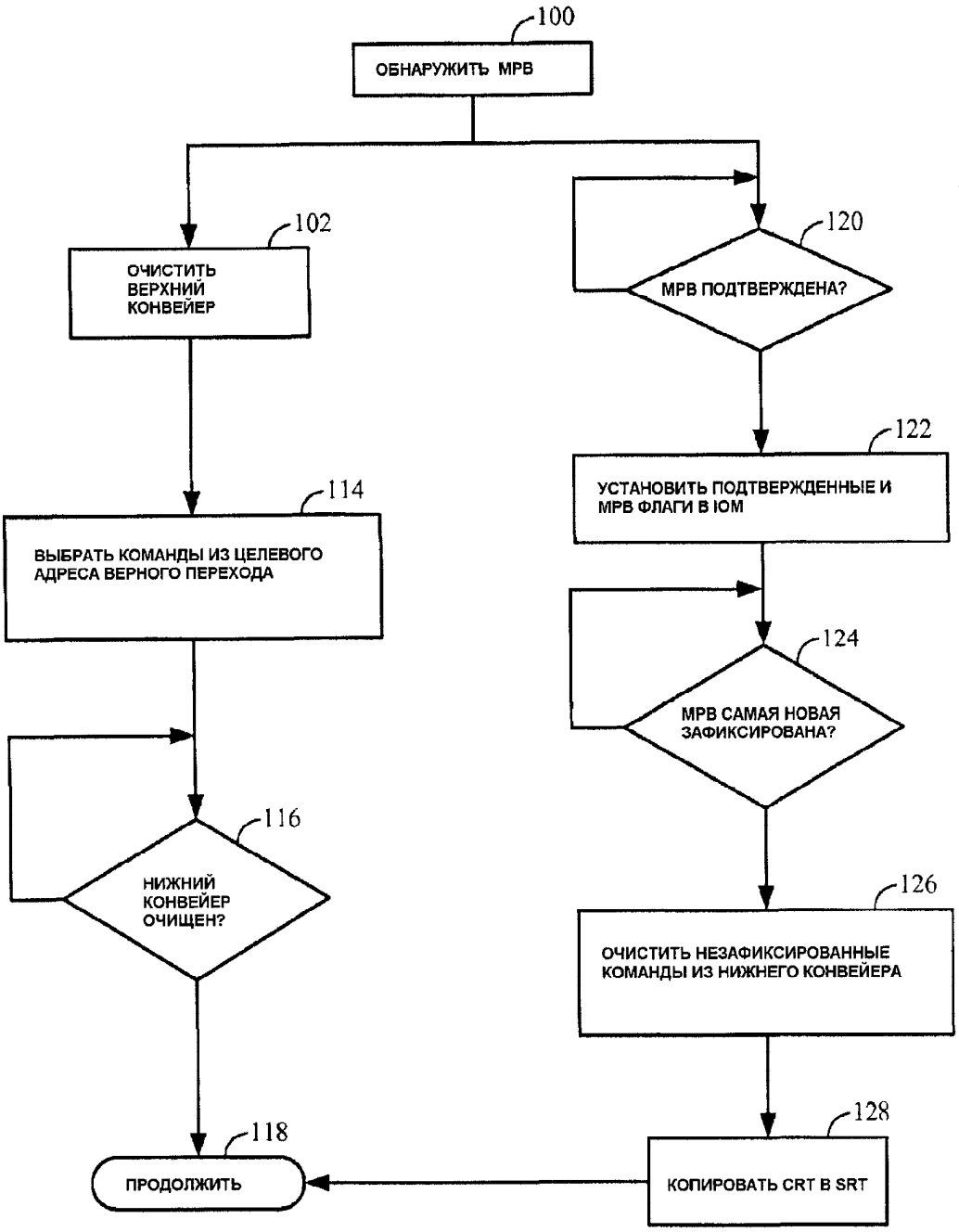
16. Процессор (10) по п.10, дополнительно содержащий

файл (20) регистра общего назначения (GPR), действующий для динамического связывания логических идентификаторов GRP с физическими регистрами, содержащий множество физических регистров,

таблицу (22) спекулятивного переименования (SRT), содержащую отображения всех текущих логических идентификаторов GPR в физические регистры, и

таблицу зафиксированного переименования (CRT), содержащую отображения логических идентификаторов GPR в физические регистры только для команд, которые зафиксированы,

причем управляющая логика (11) действует для копирования содержимого CRT в SRT после очистки всех незафиксированных команд из нижних конвейеров.


100
ОБНАРУЖИТЬ MPB
102
ОЧИСТИТЬ ВЕРХНИЙ КОНВЕЙЕР
114
ВЫБРАТЬ КОМАНДЫ ИЗ ЦЕЛЕВОГО АДРЕСА ВЕРНОГО ПЕРЕХОДА
116
НИЖНИЙ КОНВЕЙЕР ОЧИЩЕН?
118
ПРОДОЛЖИТЬ
120
MPB ПОДТВЕРЖДЕНА?
122
УСТАНОВИТЬ ПОДТВЕРЖДЕННЫЕ И MPB ФЛАГИ В IOM
124
MPB САМАЯ НОВАЯ ЗАФИКСИРОВАНА?
126
ОЧИСТИТЬ НЕЗАФИКСИРОВАННЫЕ КОМАНДЫ ИЗ НИЖНЕГО КОНВЕЙЕРА
128
КОПИРОВАТЬ CRT В SRT

Фиг. 2