ФЕДЕРАЛЬНОЕ БЮДЖЕТНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

МИКРОПРОЦЕССОРНЫЕ СИСТЕМЫ

ЛЕКЦИЯ №6

«Архитектура микропроцессора» (продолжение)

Лектор:
доцент каф. ЭАФУ ФТИ
Горюнов А.Г.

Томск 2012 г.

План лекции

6.1 Обзор микроархитектур современный десктопных процессоров (продолжение);

6.2 Внешние интерфейсы процессоров;

6.3 Эволюция и ближайшие перспективы развития процессорных архитектур;

Контрольная точка.

6.1 Обзор микроархитектур современный десктопных процессоров (продолжение)

Важной особенностью современных процессоров является также **предварительное преобразование машинных инструкций** в **промежуточные операции (микрооперации)**, более удобные для обработки и исполнения.

Иногда такие операции называют **RISC-подобными**.

Преобразование инструкций является необходимым для архитектуры x86 с ее крайне нерегулярной и запутанной системой команд.

Однако и для достаточно регулярной RISC-архитектуры PPC970 такое преобразование также производится – оно необходимо для разбиения некоторых сложных инструкций на простые операции.

В рассматриваемых нами процессорах эти промежуточные микрооперации обозначаются по-разному:

- **uOP** (Intel P-III, P-4, P8),
- **MOP** (AMD K8),
- **IOP** (IBM PPC970).

Для унификации в дальнейшем во всех случаях будем называть эти микрооперации **«МОПами»** (либо просто «операциями»).

Итак, современный процессор состоит из различных блоков, или подсистем, работающих параллельно и независимо.

1. Блоки могут являться конвейеризованными устройствами, работающими на тактовой частоте процессора (бывают также исключения, когда блок работает на половинной либо на удвоенной частоте).

2. В процессоре имеется большое количество очередей или буферов, которые необходимы в первую очередь для сглаживания задержек, возникающих при работе устройств.

3. Конкретная операция может находиться в какой-либо очереди продолжительное время, ожидая готовности данных либо ресурсов для своего дальнейшего продвижения.

Однако во многих случаях возможно и «гладкое» продвижения операции без ожиданий в очередях. Когда говорят о длине конвейера процессора, подразумевают как раз такой режим прохождения операции.

4. Таким образом, **длина конвейера** – это минимальное время прохождения операции (в тактах) при условии, что нет никаких внешних причин для задержек.

Поскольку исходная программа подразумевает последовательную модель исполнения, машинные инструкции должны считываться также последовательно. В этом случае вводится понятие, как **«отставка» инструкции.**

«Отставка» инструкции подразумевает, что данная инструкция выполнена вместе со всеми инструкциями, которые ей предшествовали в коде программы.

Таким образом, операции (инструкции) уходят в отставку строго последовательно, именно в том порядке, который задаётся программой.

Понятия отставки является ключевым в отображении внеочередного исполнения инструкций внутри процессора в чисто последовательное исполнение в «модели процессора», как она представляется пользователю.

Может оказаться, что какая-то операция уже выполнилась в своем функциональном устройстве – но при этом не должна была выполняться, так как было неправильно предсказано направление условного перехода либо были считаны данные с неправильного адреса.

Такое выполнение называется **спекулятивным**.

Когда будет правильно исполнена предшествующая ей операция перехода или чтения данных, такая неверная спекулятивная ветвь будет отменена, и при необходимости инструкция будет выполнена вновь (либо выполнится другая ветвь).

Процедура отставки инструкции гарантирует, что она была выполнена «правильно».

Любые прерывания процессора (по вводу-выводу либо аварийному событию) могут происходить только в момент отставки инструкции, к которой это прерывание относится.

Та часть процессора, в которую операции поступают строго последовательно, в соответствии с кодом программы, и из которой они выходят после выполнения тоже последовательно, называется **подсистемой внеочередного исполнения**.

Внутри этой подсистемы обеспечивается **асинхронная обработка операций**, с учетом зависимостей между ними, готовности операндов и наличия требуемых ресурсов (устройств и очередей).

Рассмотрим общую структуру процессора и взаимодействие его элементов при прохождении машинной инструкции (и операций, в которые она была преобразована).

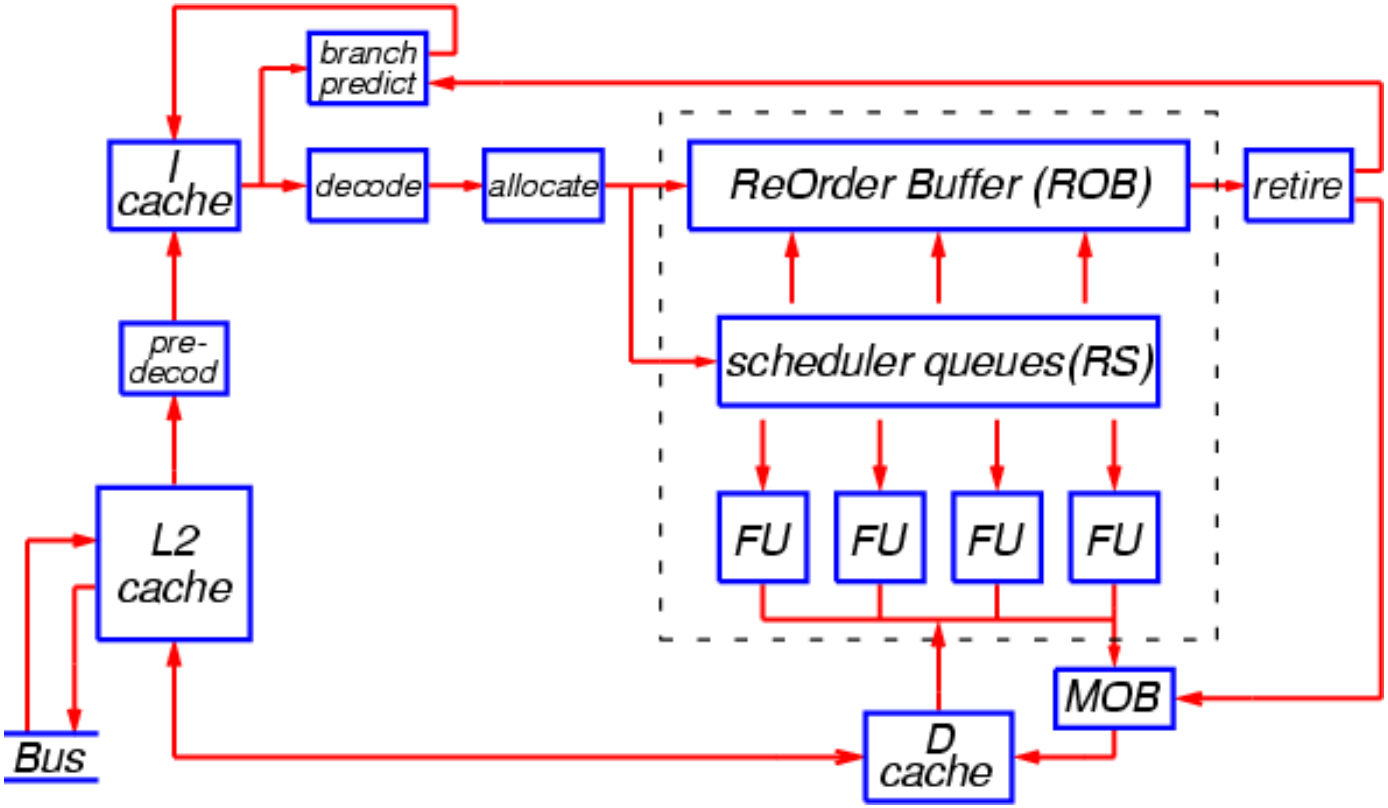

Рисунок 6.1

I-cashe – кэш инструкций (I-кэш); **L2-cashe** – кэш 2-го уровня (L2-кэш);
pre-decod – предварительный декодер; **decoder** – декодер инструкций;
branch predict – предсказатель переходов;
Rename/Allocate – устройство переименования регистров и выделения ресурсов;
ReOrder Buffer (ROB) – специальная очередь МОПов, носящая название «буфер переупорядочения»;
retire – передача результата;
Reservation Station (RS) – «пункт резервирования», «резервация»;
FU – специализированное функциональное устройство;
MOB – очередь приема результатов; **D-cashe** – кэш данных.

I-cashe

1. Подмножество кода программы, наиболее «активно» исполняемое в данный отрезок времени, размещено в кэше инструкций (I-кэше).

2. В зависимости от организации процессора, инструкции в этом кэше могут храниться в исходном неизменном виде, либо в частично «предекодированном» виде, либо в полностью «декодированном» виде (то есть в виде готовых МОПов).

3. В случае отсутствия в данном кэше нужных инструкций (исходных либо преобразованных) они считываются из кэша 2-го уровня (L2-кэша), при необходимости подвергаясь предварительному декодированию перед помещением в I-кэш.

decode

1. Инструкции считываются из I-кэша блоками, с опережающей предвыборкой.

2. Текущий блок инструкций отправляется сразу в два устройства – декодер инструкций и предсказатель переходов.

3. Декодер преобразует исходные (либо частично декодированные) инструкции в микрооперации (МОПы), а предсказатель переходов определяет, есть ли в обрабатываемом блоке инструкции перехода, будут ли эти переходы совершены и по каким адресам.

4. Если какой-либо переход предсказывается как «совершённый», то немедленно считывается блок инструкций, находящийся по предсказанному адресу.

rename/allocate

1. Тем временем вновь порождённая декодером группа МОПов поступает в устройство переименования регистров и выделения ресурсов (Rename/Allocate).

2. Переименование (или переназначение) регистров — это выделение данному МОПу нового экземпляра внутреннего регистра процессора, куда будут помещены результаты выполнения этого МОПа.

3. Все дальнейшие операции, зависящие от результатов данного МОПа, будут использовать этот регистр в качестве операнда.

4. «Переименованный» регистр ставится в соответствие регистру, который указан в машинной инструкции (так называемому «архитектурному регистру»).

5. Необходимость в переназначении регистров связана с тем, что архитектурных регистров обычно очень мало, и при использовании столь ограниченного числа регистров невозможно эффективно исполнить поток операций.

6. Каждая новая операция, использующая определённый регистр, была бы вынуждена ждать завершения всех предыдущих операций, которые к нему обращаются (даже если ей не нужны результаты этих операций).

7. Наличие большого числа машинных (физических) регистров и механизма переименования позволяет обойти эту проблему. Информация о соответствии регистров хранится в специальных таблицах. В момент отставки МОПа будет произведено обратное преобразование физического регистра в архитектурный.

ROB операции

1. После преобразования и подготовки регистров поступившая группа МОПов записывается в конец специальной очереди, носящей название «буфер переупорядочения» (ReOrder Buffer, ROB).

2. Эта структура является ключевой в организации внеочередного исполнения операций. В ней хранятся все МОПы и необходимые вспомогательные данные от момента завершения декодирования и выделения ресурсов до момента отставки.

3. Таким образом, длина буфера ROB ограничивает число операций, которые одновременно могут находиться в обрабатывающей части процессора (подсистеме внеочередного исполнения) — от самой «старой», которая ещё не завершена и поэтому не может «уйти в отставку», до самой «новой», которая только что поступила из декодера.

4. В случае переполнения буфера переупорядочения, работа декодера приостанавливается до тех пор, пока не произойдёт отставка операций в начале очереди и освобождение места для новых МОПов.

1. Одновременно с попаданием в ROB новая группа МОПов передается в другую структуру данных, откуда МОПы будут отсылаться на исполнение непосредственно в функциональные устройства.

2. Данная структура, известная под названием «пункт резервирования», «резервация» (Reservation Station, RS), представляет собой один или несколько буферов, к которым подсоединены эти функциональные устройства.

3. В каждом такте в этих буферах производится поиск операций, которые готовы к исполнению (то есть аргументы которых уже вычислены либо вычисляются и будут готовы к моменту попадания операции в функциональное устройство) вне зависимости от порядка, в котором они записывались в буфера.

4. Устройство, которое осуществляет этот поиск и запуск на исполнение, обычно называют планировщиком, а сами буфера — очередями планировщика.

5. Планировщик отслеживает зависимости между операциями по данным и прогнозирует готовность операций к исполнению в устройствах.

6. Таким образом, поиск МОПов для внеочередного исполнения всегда производится только в пределах такого буфера планировщика (единого для всех функциональных устройств, либо специфичного для каждой группы устройств)

7. Этот буфер выглядит как «окно», в котором (при необходимости) происходит изменение порядка выполнения операций.

8. Очередь планировщика представляет собой полностью ассоциативный буфер, с точки зрения поиска операций для исполнения.

9. Но с точки зрения помещения новых МОПов, она ведет себя как обычная очередь.

10. Все МОПы, находящиеся в какой-то момент в очередях планировщика, одновременно находятся и в буфере ROB.

11. При запуске операции на исполнение в функциональном устройстве соответствующий ей МОП удаляется из очереди планировщика, а в момент завершения операции делается пометка в соответствующем элементе буфера ROB.

12. Когда все МОПы, предшествующие данному, успешно выполнятся и отправятся в отставку, данный МОП также сможет быть отставлен и удалён из буфера ROB.

13. Однако может оказаться, что МОП попал в ошибочную (спекулятивную) ветвь исполнения из-за неверного предсказания перехода — в этом случае вся ветвь будет удалена из буфера переупорядочения после правильного исполнения и отставки данной инструкции перехода.

14. Когда МОП, запущенный на исполнение в устройстве, удаляется из очереди планировщика, в ней освобождается место для приёма нового МОПа.

15. Это означает, что эффективный размер окна для изменения порядка операций превышает длину данной очереди и ограничивается вместимостью буфера ROB.

16. К каждой очереди планировщика подсоединено одно или несколько специализированных функциональных устройств.

17. Число операций, которые могут быть запущены на исполнение в каждом такте, обычно заметно превышает ширину остальных трактов процессора и варьируется от 5 (P-III) до 10 (PPC970), что позволяет сглаживать пиковую нагрузку и обеспечить высокую пропускную способность при неоднородной загрузке устройств.

18. Помимо арифметико-логических и адресных функциональных устройств, в каждом процессоре имеются также устройства загрузки и выгрузки (Load/Store), которые производят доступ к кэшам данных и к оперативной памяти.

19. Эти устройства работают асинхронно от других, и их обычно не изображают на блок-схемах.

20. Логически эти устройства связаны с устройствами вычисления адресов чтения/записи (AGU).

21. Устройства загрузки и выгрузки конвейеризованы и могут одновременно обслуживать большое количество запросов.

22. Эти устройства также осуществляют предварительную выборку из оперативной памяти (копирование в кэши тех данных, использование которых ожидается в ближайшее время).

23. На рисунке 6.1 группа блоков процессора, образующих подсистему внеочередного исполнения, обведена пунктирной линией.

6.2 Внешние интерфейсы процессоров

Посредством внешних интерфейсов процессоры осуществляют доступ к оперативной памяти и внешним устройствам, а также обмен с другими процессорами в составе многопроцессорной (многоядерной) системы.

Внешний интерфейс Intel

Наиболее традиционным способом организации внешнего интерфейса является шина **FSB (Front Side Bus),** используемая в процессорах компании Intel.

В процессорах P-M, P-M2, P-4 и P8 используется одинаковая (по организации и протоколу) 64-битная шина с «учетверенной» скоростью передачи данных **QDR (Quad Data Rate)**.

Шина FSB соединяет один или два процессора (иногда больше) и контроллер, обеспечивающий доступ к оперативной памяти и внешним устройствам.

Этот контроллер входит в состав набора **системной логики (чипсета)**, его обычно называют **«Northbridge» («Северный мост»)**.

В каждом такте синхронизации шины по ней может быть передана команда либо четыре порции данных по 64 бита (8 байт).

Частота синхронизации шины находится в диапазоне 200-266 МГц для процессоров P-4/P-4E, и 266-333 МГц для процессоров P8.

Это соответствует частоте передачи 8-байтных порций данных, равной 800-1066 МГц и 1066-1333 МГц, и предельной скорости передачи данных из памяти (в память) 6.4-8.5 Гбайт/с и 8.5-10.6 Гбайт/с.

Шинная организация системы имеет свои недостатки. В системе, содержащей два или более процессоров (процессорных ядер), шина FSB ограничивает пропускную способность при доступе в память.

Использование режима передачи данных QDR с относительно невысокой частотой синхронизации приводит к дополнительным потерям времени при обращении в память.

На обращение тратится несколько тактов синхронизации шины, что может увеличить задержку доступа на 20-25 нс и более.

Наконец, шина FSB может являться узким местом при наличии внешних устройств с очень высоким темпом передачи данных (например, коммуникационных контроллеров в кластерах).

Для ослабления ограничений такого рода иногда используют многошинную организацию систем, когда, например, две пары процессоров группируются на двух независимых шинах, которые объединяются (управляются) специальным контроллером.

Внешний интерфейс IBM PPC970

Несколько иначе организован внешний интерфейс в процессоре IBM PPC970. Он соединяется с контроллером памяти через два однонаправленных 32-битных канала (последовательные шины) — один для чтения из памяти, другой для записи.

Данные передаются **с «удвоенной» скоростью (DDR, Double Data Rate)**. Предельная скорость передачи данных по каждому каналу может достигать 5 Гбайт/с (для процессора с частотой 2.5 ГГц), однако в силу последовательной организации реально достигается скорость на уровне 90% от предельной.

Достаточно высокая частота синхронизации позволяет несколько снизить потери времени на доступ, однако скорость канала недостаточна для чтения данных из памяти в необходимом темпе.

Внешний интерфейс K8

Наиболее интересным образом организованы интерфейсы в процессоре AMD K8. На кристалле этого процессора содержится **встроенный контроллер оперативной памяти, непосредственно управляющий сигналами, посылаемыми на микросхемы памяти**.

Поскольку управляющая логика такого контроллера непосредственно связана с ядром процессора и может работать на высокой частоте, исключаются потери времени на доступ к данным.

В сравнении с процессорами на шине FSB, время доступа к памяти может оказаться примерно **в полтора раза ниже** (например, 45-50 нс против 70 нс), что повышает производительность процессора за счёт снижения потерь на ожидание данных.

Реализация контроллера памяти, встроенного в процессор, имеет **много преимуществ** для использования в десктопных компьютерах.

Она способствует ощутимому повышению производительности и упрощению структуры системы.

Взаимодействие с контроллером внешних устройств осуществляется через **специальные высокоскоростные последовательные каналы HyperTransport (HT)**. Всего в процессоре может быть три таких канала с частотой синхронизации 1 ГГц при «удвоенной» скорости передачи (DDR).

Однако встроенный контроллер памяти имеет и **свои недостатки**.

В связи с тем, что повышаются электрические требования к процессору с встроенным контроллером, бывает **затруднено** использование конфигураций с **большим количеством чипов памяти**, либо в таких конфигурациях происходит снижение скоростных характеристик памяти.

6.1 Эволюция и ближайшие перспективы развития процессорных архитектур

В настоящее время эволюционное развитие архитектур современных процессорных ядер идет по пути:

1. Дальнейшая эволюция конвейеров;
2. Развитие суперскалярной организации;
3. Развитие механизмов внеочередного исполнения команд;
4. Увеличение производительности за счет увеличения тактовой частоты (за счет усовершенствования технологии изготовления) и количества инструкций, выполняемых за один машинный такт:

[Производительность] = [Тактовая частота] x [Количество инструкций, выполняемых за один такт]

5. Увеличение числа ядер на одном кристалле процессора.
6. Развитие системы команд:

6.1. **MMX (Multimedia Extensions — мультимедийные расширения)** – коммерческое название (**Intel MMX**) дополнительного набора инструкций, выполняющих характерные для процессов кодирования/декодирования потоковых аудио/видео данных действия за одну машинную инструкцию.

6.2. **3DNow!** — дополнительное расширение MMX для процессоров AMD, начиная с AMD K6 3D. Причиной создания 3DNow! послужило стремление завоевать превосходство над процессорами производства компании Intel в области обработки мультимедийных данных.

Хотя это расширение является разработкой AMD, его также интегрировали в свои процессоры IBM Cyrix и другие.

6.3. **SSE (англ. Streaming SIMD Extensions, потоковое SIMD-расширение процессора)** — это **SIMD (англ. Single Instruction, Multiple Data, Одна инструкция — множество данных)** набор инструкций, разработанный Intel и впервые представленный в процессорах серии Pentium III как ответ на аналогичный набор инструкций 3DNow! от AMD, который был представлен годом раньше.

Первоначально названием этих инструкций было KNI, что расшифровывалось как Katmai New Instructions (Katmai — название первой версии ядра процессора Pentium III).

Также существуют расширения SSE3, 3DnowExtended, SSE4 **(рассмотреть самостоятельно).**