**Задание для группы 134 на 11.02.2022**

**Задание пересылать мне на электронную почту: param\_e@mail.ru**

**Задание** также можно найти на goole-диске по ссылке:

**https://drive.google.com/drive/folders/1so8k-nqGSs5huXCXUFHUo-Hus25OJGE7?hl=ru (если задания на сайте нет, то ищите на google-диске по расписанию)**

**Конспект лекции высылаете до 20-00 часов 11.02.2022 (конспект в тетради; высылаете фотографии конспекта со своей подписью на каждом листе).**

**Без подписи не принимаю конспект.**

Ответы на вопросы в конце лекции, если они есть (ответы в тетради после конспекта).

Если конспект выслан позже указанного времени, то баллы снижаются за каждый час просрочки на 0,5.

**Напоминание. СРС2 «Технологии повышения производительности процессоров» - самостоятельно в Интернете найти и сделать конспект в тетради. Должны быть описаны следующие технологии: Конвейерная обработка, суперскалярная, Матричные и векторные процессоры, технология Hyper-Threading (HT). Срок выполнения задания – 2 недели (до 22.02.22). Буду проверять на занятиях в техникуме.**

**Лекция. КЭШ-память**

**Назначение и принцип работы КЭШ**

Повышение производительности отдельных компонентов компьютера по мере совершенствования технологии изготовления интегральных элементов идет неравномерно. Быстрее всего растет производительность процессоров, а темпы роста быстродействия модулей памяти им несколько уступают, хотя во многом это объясняется возрастанием информационной емкости отдельного модуля. Из-за этого оказывается, что при обращении процессора напрямую к оперативной памяти за необходимой ему информацией - командами и данными - последняя не успевает выполнять поступающие заявки, и процессору приходится простаивать. В полной мере реализовать потенциальную производительность процессора можно только, согласовав какими-либо методами пропускную способность подсистемы внутренней памяти и процессора.

Наибольшее распространение в современной практике проектирования компьютеров получило включение между оперативной полупроводниковой памятью (как правило, на основе DRAM-модулей) и процессором промежуточной быстродействующей памяти небольшой (по сравнению с общим объемом оперативной памяти) емкости.

Эта промежуточная память – буферное ЗУ, недоступное для программы в том смысле, что никак не может быть адресована машинными командами, от чего и произошло ее название - кэш-память (cache memory). Она разделена на встроенную в процессор (L1) и внешнюю (L2). Концепция использования блока кэш-памяти иллюстрируется схемой на рис. 1.

ЦП

КЭШ

Оперативная память

Передача слова

Передача блока

Рис. 1. Связь между процессором, кэшем и ОП

В составе компьютера имеется оперативная помять довольно большого объема и относительно невысокого быстродействия и блок кэш-памяти, обладающий высоким быстродействием, но значительно меньшей емкостью, чем оперативная память. В кэше временно хранится копия некоторого фрагмента информации из оперативной памяти. Когда процессору требуется получить от подсистемы памяти очередное слово (команду или операнд), первым делом проверяется, не содержится ли уже оно в кэше. Если да (такое событие принято называть кэш-попаданием - cache hit), то это слово считывается из кэша и передается процессору. Если нет (такое событие принято называть кэш-промахом - cache miss), то из оперативной памяти в кэш считывается блок фиксированной длины, в составе которого имеется и требуемое слово, а затем это слово передается процессору. Поскольку для процессора характерна локализация ссылок, то с большой долей вероятности следующие запросы будут адресованы словам из этого же блока.

Отношение кэш-попаданий к общему количеству обращений процессора к данным характеризует эффективность кэш-памяти (hit ratio).

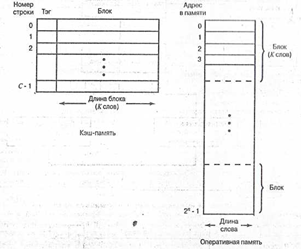


Рис. 2. Структура информации в ОП и в кэш.

На рис. 2 показана структурная организация ОП и кэша. ОП состоит из 2n адресуемых слов, каждому из которых соответствует уникальный n-битовый код адреса. ОП - совокупность множества блоков фиксированной длины в К слов (чтобы сопоставить информацию в ОП с содержимым кэш). Существует показатель информационной емкости ОП в количестве блоков: М=2n/К. Структурно кэш-память состоит из С строк (line) по К слов в каждой. Количество строк значительно меньше блоков в ОП (С<<М). В любой момент времени, в процессе выполнения программы, некоторое подмножество блоков из ОП находится в сроках кэш. Когда требуется передать в кэш какое-либо слово из ОП, в строку кэш копируется весь блок. Нельзя зафиксировать соотношение между строкой кэш и определенным блоком навсегда, т.к. количество блоков ОП >> строк.

Каждая строка содержит специальное поле (тэг - tag), в котором содержится информация о блоке ОП, хранящимся в данной сроке в текущий момент времени. Содержание тэга – это старшие разряды кода адреса в памяти.

Алгоритм операции передачи слова из кэша в процессор показан на рис. 3.

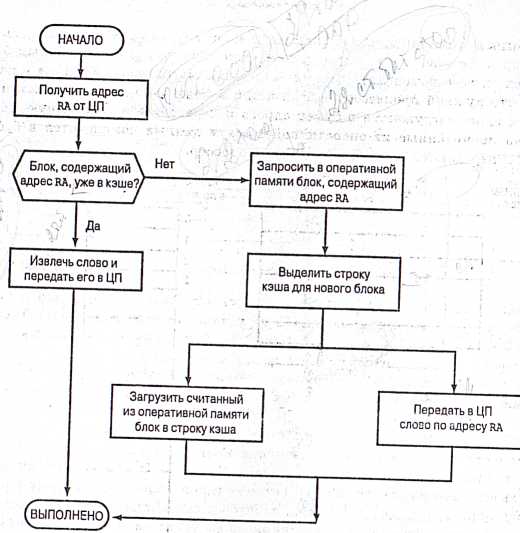


Рис. 3. Алгоритм выполнения чтения слова из кэш.

Процессор формирует адрес RA слова, которое ему необходимо. Если слово уже находится в кэше (обнаружено кэш-попадание), оно сразу же передается процессору. В противном случае в кэш из ОП копируется блок, содержащий это слово, а затем оно передается процессору. На схеме алгоритма показано, что две последних операции выполняются параллельно.

Это обеспечивается структурной организацией блока кэш-памяти (рис. 4).

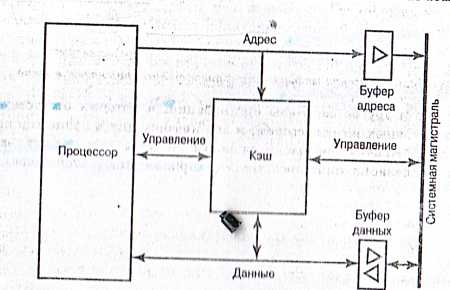


Рис. 4. Структурная схема блока кэш-памяти.

Кэш соединен с процессором линиями адреса, данных и управляющих сигналов. Линии адреса и данных подключены также к буферам адреса и данных, которые имеют выход на системную магистраль, а через нее могут обмениваться информацией с оперативной памятью.

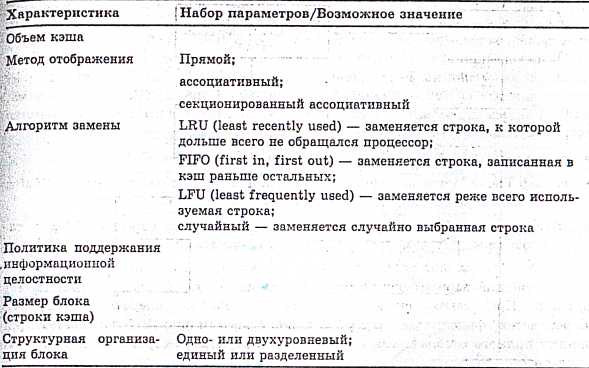
Если интересующая процессор информация уже находится в кэше, буферы адреса и данных блокируются, и весь обмен идет, минуя системную магистраль. Если же оказывается, что нужной процессору информации и кэше нет, затребованный процессором адрес загружается в буфер адреса и передается на системную магистраль, а прочитанные из оперативной памяти данные помещаются в буфер данных и передаются из него в кэш и в процессор.

Существуют и другие варианты организации, в которых отсутствует связь между буфером данных и процессором, и вся информация в процессор поступает только из кэша.

**Конструкция блока кэш-памяти**

Различные варианты конструкций блоков кэш-памяти принято сравнивать на основании набора параметров и функциональных характеристик, перечисленных в таблице 1.

Таблица 1. Характеристики кэш-памяти



**Объем кэша**

Объем кэш должен быть достаточен для того, чтобы подавляющее большинство обращений процессора за информацией в процессе выполнения программы приходилось на данные, уже имеющиеся в кэше. В тоже время она не должна слишком увеличивать относительную стоимость хранения информации, усложнять схему адресации и размещаться в микросхеме или на печатной плате. Достаточно эффективным является использование кэша объемом от 1К до 512К слов. Требования к размеру кэша тесно связаны со спецификой приложений.

**Соответствие между строками кэша и блоками оперативной памяти**

Т.к. количество строк кэша значительно меньше количества блоков ОП, при разработке кэша необходимо выбрать способ, позволяющий установить соответствие между строками кэша и блоками ОП. Способ, позволяющий выяснить, какой блок памяти размещен в той или иной строке кэша называется *функцией отображения* (mapping function). Выбор того или иного варианта функции отображения существенно влияет на структурную организацию блока кэш-памяти.

На сегодняшний день существуют три варианта решения этой функции: прямой, ассоциативный и секционированный ассоциативный. Рассмотрим каждый из них - сначала проанализируем принцип работы алгоритма, а затем опишем способы его реализации. Будем считать, что блок кэш-памяти имеет следующие параметры:

♦ объем кэша 64 Кбайт;

♦ размер блока, которым обмениваются кэш и оперативная память, 4 байта, т.е. кэш состоит из 16К = 214 строк по 4 байт в каждой;

♦ объем оперативной памяти 16 Мбайт, т.е. длина кода адреса - 24 бит; с точки зрения обмена информацией с кэшем оперативная память состоит из 4М блоков по 4 байт в каждом.

Простейший вариант функции отображения - прямое отображение (direct mapping). При таком способе за каждым блоком оперативной памяти “закрепляется” фиксированная строка кэша. Схема на рис. 5 поясняет общий принцип прямого отображения. Для определения номера строки кэша i используется простое соотношение:

i = j mod m

j - номер блока в оперативной памяти;

m - общее количество строк в кэше.

Адрес блока кэш-памяти = (Адрес блока основной памяти) mod (Число блоков в кэш-памяти).

Функция прямого отображения, использующая в качестве исходной информации адрес слова, довольно просто реализуется. Схема поиска информации в кэше рассматривает переданный процессором код адреса слова как состоящий из трех полей. Младшие w бит идентифицируют слово (в нашем случае - байт, поскольку длина слова равна 1 байт) внутри блока оперативной памяти. Старшие s бит определяют один из 2 блоков в оперативной памяти. В схемах управления кэшем эти s бит разбиваются на два поля: старшие s-r бит - поле тэга, а младшие r бит - поле номера строки, которое однозначно задает одну из m = 2r строк кэша.

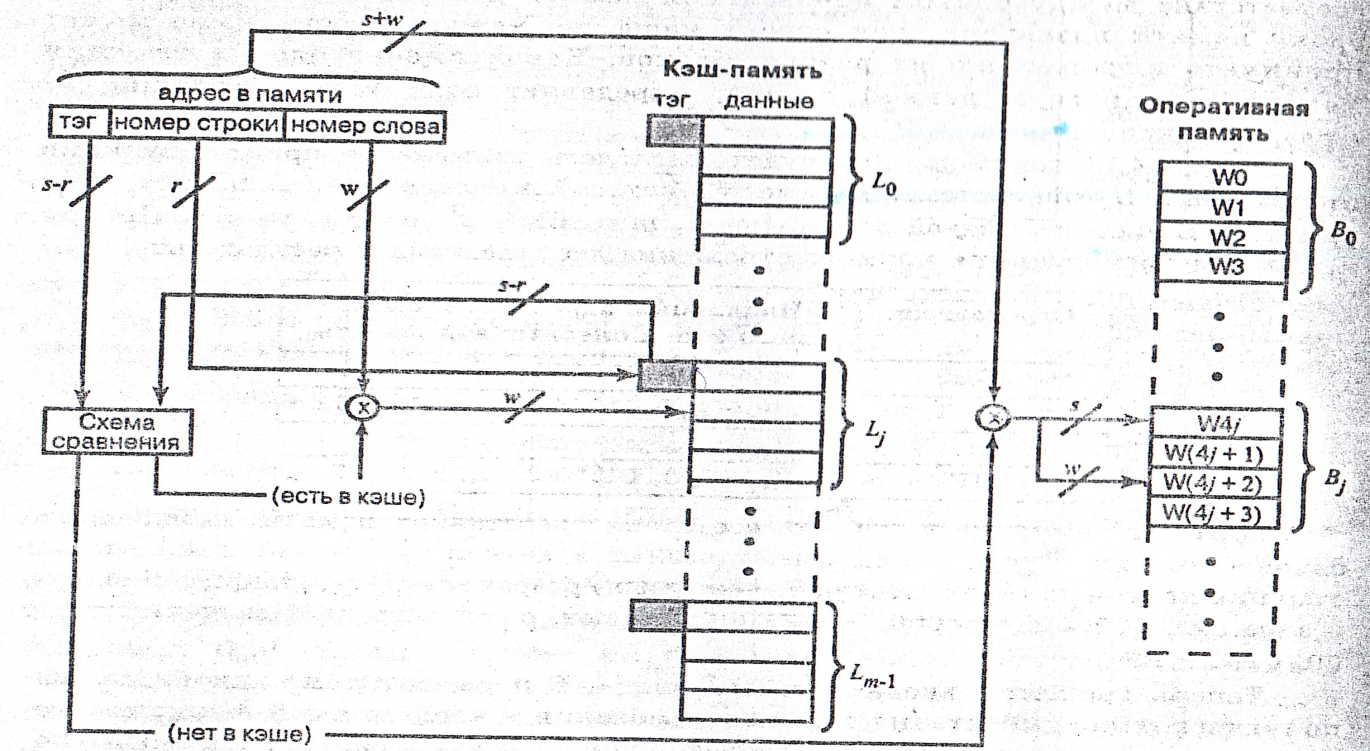


Рис. 5. Схема функционирования кэш при прямом отображении.

Рассмотрим рис. 5 выполнение операции чтения при использовании в кэше прямой функции отображения. От процессора поступает запрос байта, сопровождаемый 24- разрядным кодом адреса. Из него извлекается 14-разрядное поле номера строки, которое однозначно указывает, в какой строке кэша следует искать затребованный байт. Если значение в поле тэга переданного адреса (старшие 8 разрядов) совпадает со значением в поле тэга этой строки, то младшие 2 разряда кода адреса указывают, какой из четырех байтов этой строки следует передать процессору. В противном случае старшие 22 разряда кода адреса (поле тэга + поле номера строки) используются для обращения к оперативной памяти и, дополненные нулями в двух младших разрядах, задают начальный адрес блока размером в 4 байт.

Прямую функцию отображения довольно просто реализовать в схеме управления блоком кэш-памяти, но у нее есть существенный недостаток (который, как это почти всегда бывает, является продолжением ее достоинств). Фиксированное назначение строк кэша блокам оперативной памяти может привести к тому, что одни строки кэша будут обновляться очень часто, в то время как другие вообще не используются, поскольку к ним процессор не обращается. Если в программе имеется несколько повторяющихся обращений к двум разным блокам, отображаемым на одну и ту же строку кэша, эти блоки будут постоянно курсировать между ОП и кэш.

Кэш-память называется полностью ассоциативной, если неко­торый блок основной памяти может располагаться на любом месте кэш-памяти.

Кэш-память называется частично ассоциативной, если некоторый блок основной памяти может располагаться на ограни­ченном множестве мест.

В современных процессорах, как правило, используется либо кэш-память с прямым отображением, либо двух- (четырех-) каналь­ная множественно-ассоциативная кэш-память.

Стратегии замещения информации в кэше определяет блок, подлежащий замещению при возникновении промаха. Простота при использовании кэша с прямым отображением заключается в том, что аппаратные решения здесь наиболее простые: легко реали­зуется сама аппаратура, легко происходит замещение данных. При замещении просто нечего выбирать — на попадание проверяется только один блок и только этот блок может быть замещен. При полностью ассоциативной или множественно-ассоциативной орга­низации кэш-памяти имеются несколько блоков, из которых надо выбрать кандидата в случае промаха.

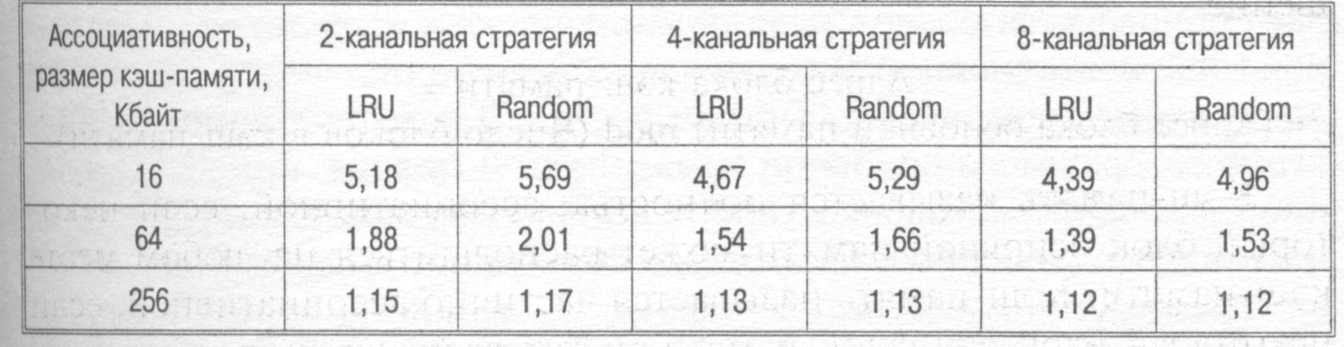
Как правило, для замещения блоков применяются две основные стратегии: случайная и LRU-стратегия.

В первом случае, чтобы иметь равномерное распределение, бло­ки-кандидаты выбираются случайно. В некоторых системах, чтобы получить воспроизводимое поведение, которое особенно полезно во время отладки аппаратуры, используют псевдослучайный алгоритм замещения.

Во втором случае, чтобы уменьшить вероятность выбрасывания информации, которая скоро может потребоваться, все обращения к блокам фиксируются. Заменяется тот блок, который не использо­вался дольше всех *(LRU - Least-Recently Used).*

Достоинство случайного способа заключается в том, что его проще реализовать в аппаратуре. Когда количество блоков увеличи­вается, алгоритм LRU становится все более дорогим и часто только приближенным. В табл. 2 показаны различия в долях промахов при использовании алгоритма замещения LRU и случайного алго­ритма.

Таблица 2.



Из таблицы видно, что при большом размере кэш-памяти стра­тегии замещения несущественно влияют на эффективность памяти.