13. Кэш-память.Способы отображения ОП на кэш-память. Алгоритмы замещения информации в заполненной кэш-памяти.

Кэш-память

В качестве элементной базы основной памяти в большинстве ВМ служат микросхемы динамических ОЗУ, на порядок уступающие по быстродействию центральному процессору. В результате процессор вынужден простаивать несколько тактовых периодов, пока информация из ИМС памяти установится на шине данных ВМ. Если ОП выполнить на быстрых микросхемах статической памяти, стоимость ВМ возрастет весьма существенно. Экономически приемлемое решение этой проблемы было предложено М. Уилксом в 1965 году в процессе разработки ВМ Atlas и заключается оно в использовании двухуровневой памяти, когда между ОП и процессором размещается небольшая, но быстродействующая буферная память. В процессе работы такой системы в буферную память копируются участки ОП, К которым производится обращение со стороны процессора. В общепринятой терминологии - производится *отображение* участков ОП на буферную память. Выигрыш достигается за счет ранее рассмотренного свойства локальности – если отобразить участок ОП в более быстродействующую буферную память и переадресовать на нее все обращения в пределах скопированного участка, можно добиться существенного повышения производительности ВМ.

Уилкс называл рассматриваемую буферную память подчиненной (slave memory). Позже распространение получил термин *кэш-память* (от английского слова *cache* — убежище, тайник), поскольку такая память обычно скрыта от программа миста в том смысле, что он не может ее адресовать и может даже вообще не знать о ее существовании. Впервые кэш-системы появились в машинах модели 85 семейства IBM 360.

В общем виде использование кэш-памяти поясним следующим образом. Когда ЦП пытается прочитать слово из основной памяти, сначала осуществляется поиске копии этого слова в кэше. Если такая копия существует, обращение к ОП не производится, а в ЦП передается слово, извлеченное из кэш-памяти. Данную ситуацию принято называть успешным обращением или *попаданием* (hit). При *отсутствии* слова в кэше, то есть при неуспешном обращении — *промахе* (miss), — требуемое слово передается в ЦП из основной памяти, но одновременно из ОП в кэш-память пересылается блок данных, содержащий это слово.

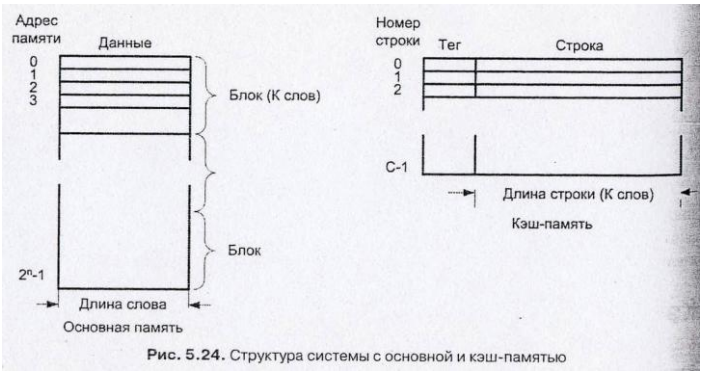

Рис. 5.24. Структура системы с основной и кэш-памятью

На **рис. 5.24** приведена структура системы с основной и кэш-памятью. ОП состоит из 2^n адресуемых слов, где каждое слово имеет уникальный n-разрядный адрес. При взаимодействии с кэшем эта память рассматривается как M блоков фиксированной длины по K слов в каждом $(M = 2^n/K)$. Кэш-память состоит из C блоков аналогичного размера (блоки в кэш-памяти принято называть *строками),* причем их число значительно меньше числа блоков в основной памяти ($C << M$). При считывании слова из какого-либо блока ОП этот блок копируется в одну из строк кэша. Поскольку число блоков ОП больше числа строк, отдельная строка не может быть выделена постоянно одному и тому же блоку ОП. По этой причине каждой строке кэш-памяти соответствует *тег* (признак), содержащий сведения о том, копия какого юлока ОП в данный момент хранится в данной строке. В качестве тега обычно используется часть адреса ОП.

На эффективность применения кэш-памяти в иерархической системе памяти влияет целый ряд моментов. К наиболее существенным из них можно отнести:

- емкость кэш-памяти;

- размер строки;
- способ отображения основной памяти на кэш-память;
- алгоритм замещения информации в заполненной кэш-памяти;
- алгоритм согласования содержимого основной и кэш-памяти;
- число уровней кэш-памяти.

Способы отображения оперативной памяти на кэш-память

Сущность отображения блока основной памяти на кэш-память состоит в копировании этого блока в какую-то строку кэш-памяти, после чего все обращения к блоку в ОП должны переадресовываться на соответствующую строку кэш-памяти. Удачным может быть признан лишь такой способ отображения, который одновременно отвечает трем требованиям: обеспечивает быструю проверку кэш-памяти на наличие в ней копии блока основной памяти; обеспечивает быстрое преобразование адреса блока ОП в адрес строки кэша; реализует достижение первых двух требований наиболее экономными средствами.

Для облегчения понимания комплекса вопросов, возникающих при выборе способа отображения оперативной памяти на кэш-память, будем рассматривать систему, состоящую из основной памяти емкостью 256 Кслов, и кэш-памяти емкостью 2 Кслова. Для адресации каждого слова основной памяти необходим 18-разрядный адрес (2^{18} - 256К). ОП разбивается на блоки по 16 слов в каждом, следовательно ее удобно рассматривать как линейную последовательность из 16 384 = 2^n блоков. При такой организации 18-разрядный адрес можно условно разделить на две части: младшие 4 разряда определяют адрес слова в пределах блока, а старшие 14 - номер одного из 16 384 блоков. Эти старшие 14 разрядов в дальнейшем будем называть адресом *блока основной памяти*. В свою очередь, для адресации любого слова в кэш-памяти требуется 11-разрядный адрес (2^n - 2К). Кэш-память разбита на строки такого же размера, что и в ОП (вместо слова «блок» принято использовать термин «строка»), то есть содержит 128 = 2^n строк. 11-разрядный адрес слова в кэш-памяти также можно представить состоящим из двух частей: адреса слова в строке (4 младших разряда) и *адреса строки кэш-памяти* (7 старших разрядов).

Поскольку ЦП всегда обращается к ОП (кэш-память для ЦП невидима) и формирует для этого 18-разрядный адрес, необходим механизм преобразования такого адреса в 11-разрядный адрес слова в кэше. Так как расположение слов в блоке и строке кэш-памяти идентично, для доступа к конкретному слову в блоке ОП или в строке кэш-памяти можно использовать младшие 4 разряда 18-разрядного адреса. Следовательно, остается лишь задача преобразования 14-разрядного адреса блока основной памяти в 7-разрядный адрес строки кэша.

Известные варианты отображения основной памяти на кэш можно свести к трем видам: прямому, полностью ассоциативному и частично-ассоциативному, причем последний имеет две модификации — множественно-ассоциативное отображение и отображение секторов.

Алгоритмы замещения информации в заполненной кэш-памяти

Когда кэш-память заполнена, занесение в нее нового блока связано с замещением содержимого одной из строк. При прямом отображении каждому блоку основной памяти соответствует только одна определенная строка в кэш-памяти, и никакой иной выбор удаляемой строки здесь невозможен. При полностью и частично ассоциативных способах отображения требуется какой-либо алгоритм замещения (выбора удаляемой из кэш-памяти строки).

Основная цель стратегии замещения — удерживать в кэш-памяти строки, к которым наиболее вероятны обращения в ближайшем будущем, и заменять строки доступ к которым произойдет в более отдаленном времени или вообще не случится. Очевидно, что оптимальным будет алгоритм, который замещает ту строку, обращение к которой в будущем произойдет позже, чем к любой другой строке кэш. К сожалению, такое предсказание практически нереализуемо, и приходится привлекать алгоритмы, уступающие

оптимальному. Вне зависимости от используемого алгоритма замещения для достижения высокой скорости он должен быть реализован аппаратными средствами.

Среди множества возможных алгоритмов замещения наиболее распространенными являются четыре, рассматриваемые в порядке уменьшения их относительной эффективности.

Наиболее эффективным является алгоритм замещения на основе *наиболее давнего использования* (LRU — Least Recently Used), при котором замещается та строка кэш-памяти, к которой дольше всего не было обращения. Проводившие исследования показали,что алгоритм LRU, который «смотрит» назад, работает, достаточно хорошо в сравнении с оптимальным алгоритмом, «смотрящим» вперед.

Наиболее известны два способа аппаратурной реализации этого алгоритма.

В первом из них с каждой строкой кэш-памяти ассоциируют счетчик. К содержимому всех счетчиков через определенные интервалы времени добавляется единица. При обращении к строке ее счетчик обнуляется. Таким образом, наибольшее число будет в счетчике той строки, к которой дольше всего не было обращений и эта строка — первый кандидат на замещение.

Второй способ реализуется с помощью очереди, куда в порядке заполнения кэш-памяти заносятся ссылки на эти строки. При каждом обращении к строке ссылка на нее перемещается в конец очереди. В итоге первой в очереди каждый раз оказывается ссылка на строку, к которой дольше всего не было обращений. Именно эта строка прежде всего и заменяется.

Другой возможный алгоритм замещения — алгоритм, работающий по принципу *первый вошел, первый вышел* (FIFO — First In First Out). Здесь заменяет строка, дольше всего находившаяся в кэш-памяти. Алгоритм легко реализуется с помощью рассмотренной ранее очереди, с той лишь разницей, что после обращения к строке положение соответствующей ссылки в очереди не меняется.

Еще один алгоритм — замена *наименее часто использовавшейся* строки (LFU Least Frequently Used). Заменяется та строка в кэш-памяти, к которой было меньше всего обращений. Принцип можно воплотить на практике, связав каждую строку со счетчиком обращений, к содержимому которого после каждого обращения добавляется единица. Главным претендентом на замещение является строка, счетчик которой содержит наименьшее число.

Простейший алгоритм - *произвольный выбор* строки для замены. Замещаемая строка выбирается случайным образом. Реализовано это может быть, например, с помощью счетчика, содержимое которого увеличивается на единицу с каждым тактовым импульсом, вне зависимости от того, имело место попадание или промах. Значение в счетчике определяет заменяемую строку к полностью ассоциативной кэш-памяти или строку в пределах модуля для множественно-ассоциативной кэш-памяти. Данный алгоритм используется крайне редко.

Среди известных в настоящее время систем с кэш-памятью наиболее встречаемым является алгоритм LRU.