Министерство образования и науки Украины

Национальный технический университет Украины

«Киевский политехнический институт»

Факультет прикладной математики

Кафедра специализированных компьютерных систем

Лабораторная работа №2

по дисциплине «**Архитектура компьютера**»

Тема: «**Кэш-память**»

Вариант №2

|  |  |
| --- | --- |
| *Выполнили студенты*  группы **КВ-64**  **Подольский Сергей**  **Сокол Владислав** | *Проверил*  \_\_\_\_ \_\_\_\_\_\_\_\_\_\_ 2010г.  **Романкевич А.М.**  ( \_\_\_\_\_\_\_\_\_\_\_\_\_\_ баллов) |

Киев-2010

**Задание**

1. Какие методы отображения Вы знаете?
2. Какие характеристики влияют на попадание в кэш?
3. С помощью программы CacheModule изучите влияние положения кластера в ОП. Приведите примеры статической, динамической и увеличивающейся  области памяти в реальном процессе работы компьютера.
4. Какая величина может служить показателем эффективности доступа к подсистеме памяти?
5. Методы отображения

Поскольку количество строк кэша значительно меньше количества блоков опе­ративной памяти, при разработке кэша необходимо выбрать какой-либо способ, по­зволяющий установить соответствие между строками кэша и блоками оперативной памяти, т.е. позволяющий как можно проще и быстрее выяснить, какой блок памя­ти размещен в той или иной строке кэша. Для краткости такой способ называют 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 проблемы: прямой, ассоциативный и секционированный ассоциативный. Ниже мы 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 байт; это означает, что кэш состоит из 16К = 214строк по 4 байт в каждой;

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

***Прямое отображение.***

Простейший вариант функции отображения — прямое отображение (direct mapping). При таком способе за каждым блоком оперативной памяти "закрепляется" фиксированная строка кэша. Для определения номера строки кэша i исполь­зуется простое соотношение:

i = j mod m,

где

j — номер блока в оперативной памяти;

m — общее количество строк в кэше.

Функция прямого отображения, использующая в качестве исходной информа­ции адрес слова, довольно просто реализуется.

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 другое вообще не используются, поскольку к ним процессор не обращается. Так, если в программе имеется несколько повторяющихся обращений к двум разным строкам, отображаемым на одну и ту же строку кэша, эти блоки будут постоянно "курсировать" между оперативной памятью и кэшем, и проку от него(кэша) в этом случае будет мало.

Этот недостаток устраняется использованием *ассоциативной* функции отображения. Суть данного метода в том, что тэгом являются все старшие разряды кода адреса, т.е. те, которые при прямой функции делились на поле тэга и поле номера строки.

В результате ассоциативная функция отображе­ния разрывает жесткую связь между блоком оперативной памяти и опреде­ленной строкой кэша — теперь любой блок может оказаться в любой строке кэша. Конечно, такое решение серьезно усложняет логику поиска в кэше за­требованного слова. Схема управления кэшем при выполнении операции чтения должна сравнить старшие разряды кода адреса с тэгами всех строк кэша, причем для обеспечения нужного быстродействия сравнение должно осуществляться параллельно по всем строкам.

Соответствие между содержимым строк кэша и блоков оперативной памяти при реализации ассоциативной функции отображения – очевидно. Код адреса в оперативной памяти рассматривается как со­стоящий из двух полей: 22-разрядного поля *тэга* и 2-разрядного поля *номера байта.* Таким образом, в каждой строке кэша нужно хранить помимо блока данных длиной в 4 байт (32 бит) еще и 22-разрядный код тэга этого блока. Так 24-разрядному адресу 16339С будет соответствовать 22-разрядный код тэга 058СЕ7(шестнадцатеричная нотация).

***Ассоциативная функция отображения***

Применение ассоциативной функции отображения обеспечивает известную гибкость при распределении данных между строками кэша, поскольку позволяет применять более сложные алгоритмы при выборе строк, в которые считывается очередной блок из оперативной памяти. Эти алгоритмы позволяют так организовать обновление строк, что вероятность обнаружения нужного слова данных в кэше *(кэш-попадания — hit in cache)* значительно возрастает, а это значит, что значительно реже нужно обращаться за новыми данными к оперативной памяти. Но все это дается немалой кровью — как вы убедились на рассмотренном простом примере, блок кэш-памяти должен располагать ресурсами хранения информации, более чем в пол­тора раза превышающими объем полезной информации. Логические схемы анализа и сравнения тэгов при такой схеме также значительно сложнее, чем при использовании простой функции отображения.

***Секционированная ассоциативная функция отображения***

Третий метод — *секционированная ассоциативная* функция отображения — является комбинацией двух первых. Блок основной памяти может располагаться на ограниченном множестве мест в кэш-памяти, то кэш называется множественно-ассоциативным (set associative). Обычно множество представляет собой группу из двух или большего числа блоков в кэше. Если множество состоит из n блоков, то такое размещение называется множественно-ассоциативным с n каналами (n-way set associative).

В нем сочетаются относительная простота реализации прямой функции отображения с гибкостью, которую обес­печивает ассоциативная функция.

При реализации этого метода весь массив кэш-памяти делится на *v* секций, каждая из которых состоит из *k* строк. Таким образом, имеют место соотношения:

m = *v* x *k*,

*i* = *j* mod v,

где

*j* — номер блока в оперативной памяти

*i* — номер секции кэша;

*m* — общее количество строк в кэше.

В каждой секции используется ассоциативная функция отображения, и блок Bjможет быть передан в любую строку секции *i* . Схемы управления блока кэш-памяти анализируют три поля в составе кода адреса, поступившего от про­цессора: тэг, *номер секции,* и *номер слова* (байта).

Поле номера секции имеет длину *d* разрядов, и задает одну из v = *2d* секций.

Поле тэга длиной *s* разрядов совместно с полем номера секции определяют один из *2s* блоков в оперативной памяти, который может быть помещен в любую строку этой секции.

Рассмотрим схему управления блоком кэш-памяти, в котором использована сек­ционированная ассоциативная функция отображения.

При использовании чисто ассоциативной функции поле тэга в строке кэша имело довольно большую длину и значение тэга из кода адреса, полученного от процессора, нужно было срав­нить со значениями в полях тэга *всех* строк кэша.

При использовании секцио­нированной ассоциативной функции поле тэга в коде адреса имеет несколько меньшую длину, и при анализе адреса это поле нужно сравнивать только с по­лями тэгов строк одной секции (обычно их бывает 2, 4 или 8).

Здесь схематически представлено соответствие между содержимым строк кэша и блоков оперативной памяти при реализации секционированной ас­социативной функции отображения.

Каждая секция включает две строки кэша и, следовательно, номер секции задается 13-разрядным двоичным числом. Блоки в оперативной памяти, номера которых, взятые по модулю 213, совпадают с но­мером секции, будут отображаться на строки этой секции кэша. Так блоки 000000, 00А000, .... FF4000 будут отображаться на строки секции 0. Каждый из этих блоков может быть помещен в любую из двух строк секции 0.

Обратите внимание на то, что ни один из двух блоков, которые отображаются на одну и ту же секцию, не имеют одинаковых кодов тэга.

При выполнении операции чтения 13-разрядный номер секции определяет, какую из двухстрочных секций кэша следует анализировать на предмет того, не содержится ли в ней искомый блок. При этом анализируются поля тэгов обеих строк секции.

Если *v = т, k=* 1, то секционированная ассоциативная функция сводится к прямой функции отображения, а при v = 1, *k = т* секционированная ассоциативная функция сводится к чистой ассоциативной. Разделение кэша на 2-строчные секции (v = m/2, *k =* 2) — самый распространенный вариант использо­вания секционированной ассоциативной функции, который дает наибольший прирост эффективности по сравнению с прямой функцией отображения. Если еще увеличить размер секции — включить в нее 4 строки (v = m/4, k = 4). — то повышение эффективности кэша будет довольно скромным (правда, изатраты на такую модификацию будут невелики*.*

Дальнейшее увеличение размера секции оказывает очень незначительное влияние на повышение эффективности.

1. На попадение в кэш влияют размер кэша, размера строки кэша (блока памяти), а также метод отображения и алгоритм замены строк кеша.
2. Результаты тестов с помощью программы CacheModule (ассоциативная функция отображения):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Смещение области | Опыт №1 | Опыт №2 | Опыт №3 | Среднее значение |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Примером  увеличивающейся  области памяти в реальном процессе работы компьютера может быть, например, загрузка операционной системы, когда количество исполняемого кода в оперативной памяти увеличивается с загрузкой каждого следующего модуля или сервиса. Примером динамической оперативной памяти может быть программа, которая работает поэтапно, подгружая динамические библиотеки библиотеки невыполненного кода и выгружая библиотеки выполненного. Примером статической является детерминированная программа для многократного решения одной конкретной задачи.

1. среднее время доступа к элементу информации в двухуровневой подсистеме. Оно зависит не только от характеристик каждого уровня, но и от вероятности того, что затребованный элемент окажется в *M1*. Итак, получим:

TS=HxT1+(1-H)x(T1+T2)= T1+(1-H)xT2

где

*Ts* — среднее время доступа к подсистеме памяти;

*t1 —* время доступа к памяти *M1* (т.е. кэшу процессора или дисковому кэшу в оперативной памяти);

*Т2 —* время доступа к памяти М2 (т,е. оперативной памяти или диску);

*Н* — эффективность памяти *M1* (отношение количества удовлетворенных заявок к общему количеству заявок).