1. Мультипрограммный режим работы компьютера

Возможность компьютерной системы выполнять несколько действий (процессов) одновременно. А системах с несколькими логическими или физическими ядрами (или даже процессорами), код действительно выполняется параллельно, а однопроцессорных же системах создаётся иллюзия этого за счёт переключения между процессами, когда один из простаивает.

1. Организация конвейера операций.

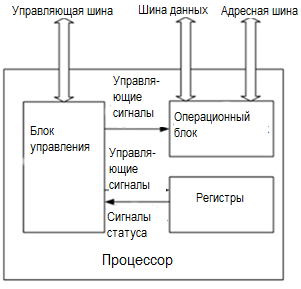
Способ организации вычислений, используемый в современных [процессорах](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80) целью повышения их производительности (увеличения числа инструкций, выполняемых в единицу времени — эксплуатация [параллелизма на уровне инструкций](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D0%B8%D0%B7%D0%BC_%D0%BD%D0%B0_%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5_%D0%BA%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4)).

Идея заключается в параллельном выполнении нескольких инструкций процессора. Сложные инструкции процессора представляются в виде последовательности более простых стадий. Вместо выполнения инструкций последовательно (ожидания завершения конца одной инструкции и перехода к следующей), следующая инструкция может выполняться через несколько стадий выполнения первой инструкции.



Пример конвейера. За единицу времени (зелёная область) выполняется несколько стадий.

1. Каноническая структура процессора.



Управляющий блок – декодирует команды и отвечает за взаимодействие процессора с памятью. Операционных блок – выполняет арифметические операции над числами с фиксированной и плавающей точкой с помощью АЛУ и FPU соответственно. Регистры – блоки памяти для служебных целей, к примеру, хранения адреса следующей выполняемом команды или промежуточных результатов вычислений (аккумулятор). Также процессор может содержать кэш – быструю память для хранения повторяющихся команд, что снижает количество обращений к медленной памяти.

Тактовый генератор, или генератор тактовых импульсов, задаёт рабочую частоту процессора. С помощью тактовых импульсов выполняется синхронизация для внутренних команд процессора и остальных устройств. Тактовый генератор вырабатывает (генерирует) прямоугольные импульсы, которые следуют с определённой частотой (для разных процессоров частота разная).

1. Смешанное кодирование операционных микрокоманд. Достоинства и недостатки.

Смешанное кодирование микрокоманд призвано устранить недостатки вертикального и горизонтального кодирования, такие как: неэффективное использование памяти при горизонтальном кодировании и сложные дешифраторы при вертикальном.

При смешанном кодировании множество микроопераций V разбивается на k

подмножеств (или полей):

k

V = UVi .

i =1

Подмножества Vi кодируются горизонтальным, а микрооперации внутри каждого из этих подмножеств вертикальным способами. В этом случае каждому подмножеству Vi выделяется отдельное поле в операционной части микрокоманды. Длина n операционной части микрокоманды равна:

k

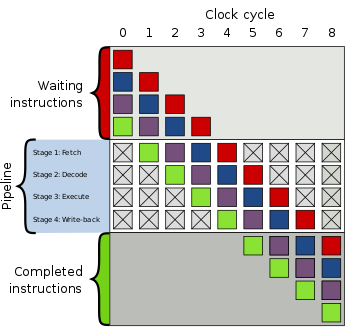
n = ∑ ] log 2 mi ,

i =1

Дешифраторы дешифрирующие код микроопераций отдельных полей, вырабатывают управляющие слова, реализуемые в течение одного микрокомандного цикла. Таким образом, применение вертикального метода кодирования позволяет оптимизировать емкость ПМК, а горизонтального способа – сохранить принципы параллельной работы операционных блоков микропроцессора.

1. Организация конвейера команд в процессоре Intel Pentium.

Пример конвейера

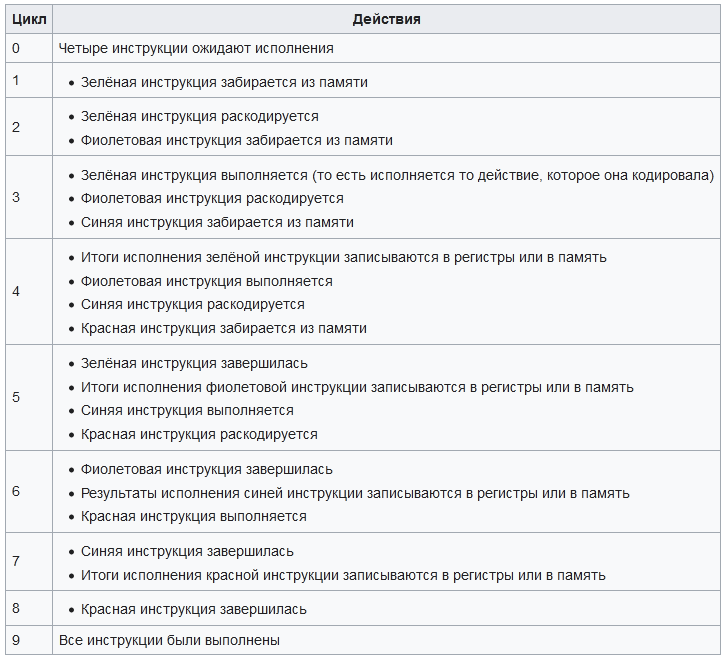


Конвейер делится на 4 стадии обработки:

1. Получение
2. Раскодирование
3. Выполнение
4. Запись результата

Каждая из стадий, в идеале, должна выполнятся параллельно с другой:

Пример:



1. Особенности EPIC-архитектуры

EPIC - архитектура процессора с явным параллелизмом команд. EPIC позволяет микропроцессору выполнять инструкции параллельно, опираясь на работу компилятора, а не выявляя возможность параллельной работы инструкций при помощи специальных схем. Это могло упростить масштабирование вычислительной мощности процессора без увеличения тактовой частоты.

В архитектуре EPIC, предполагается наличие в процессоре ста двадцати восьми 64-разрядных регистров общего назначения и ста двадцати восьми 80-разрядных регистров с плавающей запятой. Команды упаковываются компилятором в сверхдлинную команду длиною в 128 разрядов.

Особенностями архитектуры EPIC являются:

* большое количество регистров;
* масштабируемость архитектуры до большого количества функциональных блоков.
* явный параллелизм в машинном коде. Поиск зависимостей между командами осуществляет не процессор, а компилятор;
* предикация - команды из разных ветвей условного предложения снабжаются полями условий и запускаются параллельно;
* предварительная загрузка - данные из медленной основной памяти загружаются заранее.

1. Функции системы прерываний

1.       определение наиболее приоритетного незамаскированного запроса на прерывание (если одновременно поступило несколько запросов);

2.       определение типа выбранного запроса;

3.       сохранение текущего состояния счетчика команд и регистра флагов;

4.       определение адреса обработчика прерывания по типу прерывания и передача управления первой команде этого обработчика;

5.       выполнение программы - обработчика прерывания;

6.       восстановление сохраненных значений счетчика команд и регистра флагов прерванной программы;

7.       продолжение выполнения прерванной программы.

1. Синхронные и асинхронные прерывания

Синхронные прерывания (или исключения). Эта прерывания возникают в CPU или в текущей программе. Исключения могут возникнуть по разным причинам: вследствие программной ошибки во время выполнения или из-за аномального поведения программы, которое требует «внешней» инструкции для процессора. В некоторых случаях ядро информирует приложение, что произошла особая ситуация. Это дает возможность процессу скорректировать появление ошибки и продолжить выполнение программы или закончить ее корректно.

асинхронные прерывания вызываются периферийными устройствами в произвольное время. В отличие от синхронных прерываний, асинхронные не связаны каким-нибудь процессом. Они возникают в любое время независимо от состояния системы. Например, сигнал с сетевого адаптера о получении очередного пакета данных из сети будет вызывать соответствующее прерывание для сохранения этих данных. Поскольку невозможно предсказать момент прихода этого пакета в систему, ядро должно гарантировать, что прерывание произойдет максимально быстро, а пакет будет доставлен.

1. Концепция кэш-памяти.

Небольшая, но очень быстрая (от того и дорогая память), предназначенная для хранения наиболее частых команд или данных поступивших недавно команд для минимизации обращений процессора к медленной памяти. Также кэш находится физической близко к процессору, что уменьшает время обращения к нему. К современных процессорах делится на 3 уровня, где с каждым уровнем увеличивается объём, но уменьшается скорость. Может быть общим как для всех ядер, так и отдельным.

1. Пространственная локальность в отношении команд и данных.

Основная идея кэширования опирается на свойство локальности данных и инструкций: если происходит обращение по некоторому адресу, то велика вероятность, что в ближайшее время произойдет обращение к памяти по тому же адресу либо по соседним адресам.

1. Кэш-память с прямым отображением – достоинства и недостатки

Логически кэш-память представляет собой набор кэш-линий. Каждая кэш-линия хранит блок данных определенного размера и дополнительную информацию. Под размером кэш-линии понимают обычно размер блока данных, который в ней хранится.

Принцип работы кэша с прямым отображением, следующий: при поступлении команды, анализируется адрес данных. Затем сравнивается адрес, если он совпадает с размешенной в кэше командой, происходит кэш-попадание и команда считывается из кэша, иначе кэш-промах и обращение к медленной памяти.

1. Стратегии замещения блоков в кэш-памяти

При использовании любой стратегии отображения, кроме принципа прямого отображения, между некоторым множеством блоков ОП существует конкуренция на захват места в КЭШ памяти. В связи с этим возникает прблема выбора блока, кандидата на удаления из КЭШ памяти, при очередном КЭШ промахе. Эта проблема решается с использованием одной из стратегий замещения блока. В зависимости от используемого принципа отображения, выбор кандидата на удаление из КЭШ памяти осуществляется одним из способов:

1 Случайное замещение (RAND) не требует никакой истории, самый простой в реализации.

2 Стратегия с использованием принципа FIFO, удалению подлежит блок, который наиболее давно был помещен в КЭШ память.

3 LFU – Least Frequenly Used наименее часто используемый блок. Собирается статистика по частоте обращения между двумя последовательными КЭШ промахами.

4 LRU – Least Resentely Used В соответствии с ней удалению подлежит блок, к которому наиболее долго не было обращений.

1. Стратегия WT обновления основной памяти

По WT обычно обновляется слово, хранящееся в озу. Если в кэш-памяти есть копия этого слова, то она также обновляется. иначе, то либо из озу в кэш-память пересылается блок, содержащий это слово (WTWA), либо этого не делается (WTNWA). При любом обращении к памяти по записи, обращение идет к озу, то есть её содержимое остается актуальным.

1. На основе какого типа запоминающих элементов строится кэш-память?

На основе SRAM - полупроводниковая оперативная память, в которой каждый двоичный или троичный разряд хранится в схеме с положительной обратной связью, позволяющей поддерживать состояние без регенерации, необходимой в динамической памяти. Тем не менее сохранять данные без перезаписи SRAM может, только пока есть питание, то есть она энергозависима.

1. Выборка широким словом

Один из способов сокращения числа обращений к озу состоит в организации. выборки широким словом. При выборке широким словом за одно обращение к ОП производится одновременное считывание (или запись) нескольких команд или слов данных из широкой ячейки. Широкое слово заносится в буферную память (кэш-память или регистр), где оно расформировывается на отдельные команды (или слова данных), которые могут (последовательно) использоваться процессором уже без дополнительных обращений к ОП.

Реализация выборки широким словом вызывает необходимость мультиплексирования данных между кэш-памятью и процессором, поскольку основной единицей обработки данных в процессоре остается слово. Кэш-память второго уровня позволяет смягчить эту проблему, так как в этом случае мультиплексоры могут располагаться между двумя уровнями кэш-памяти, и вносимая ими задержка не столь критична.

1. Распределение памяти разделами переменной величины

В этом случае память машины не делится на разделы заранее. Сначала вся память свободна. Каждой поступающей команде выделяется необходимая ей память, и если достаточный объём отсутствует, то задача ставится в очередь. После завершения задачи память освобождается. Таким образом, в произвольный момент времени оперативная память представляет собой случайную последовательность занятых и свободных участков произвольного размера.

1. Организация защиты памяти по ключам.

Каждому присваивается уникальный ключ. Все выделенные для него страницы отмечаются ключом. В качестве ключа защиты обычно указывается двоичный код номера программы. При обращении к озу производится сравнение ключа процесса с ключами соответствующих страниц памяти. Обращение разрешается только при совпадении сравниваемых кодов ключей.

1. Достоинства и недостатки стратегии RAND для выбора блока-кандидата на удаление из кэш.

При использовании принципа случайного выбора не предполагается выполнения анализа предыстории блоков, находящихся в кэш-памяти. Согласно стратегии RAND блок – кандидат на удаление – выбирается случайным образом (т.е. удален может быть любой из допустимого множества блоков кэш-памяти).

1. Понятие латентности памяти.

Латентность (задержка) - это время, которое затрачивается на чтение из памяти одного слова данных (восьми байт) . Чем ниже латентность оперативной памяти, тем меньше центральный процессор будет находиться в состоянии простоя. Полная латентность складывается из программной и аппаратной составляющих. Значения латентности измеряются в микросекундах. Латентность измеряется как время, необходимое на передачу сигнала, или сообщения нулевой длины.

1. Может ли FPU самостоятельно обращаться к памяти?

Нет, все операции с памятью должны проходить через блок управления процессора.