Память. Структура памяти.

ОП (Основная память) состоит:

1) ***ПЗУ*** (ROM) – только для чтения, хранит тестовые программы диагностики устройств, программу начальной загрузки ***ОС***, которая считывает из стартового сектора системного диска системный загрузчик, а так же в ***ПЗУ*** находятся драйвера стандартных ПУ и обработчик прерываний.

Flash-***ОЗУ*** - является энергозависимой памятью, но позволяет переписывать информацию, тем самым исправлять ошибки, допущенные в ***ПЗУ***.

CMOS-память – используется для хранения текущей даты и времени.

2) Энергозависимая память – ***ОЗУ*** – главная рабочая область ЗУ, которая хранит программы и данные на сеанс работы.

ОЗУ – адресное пространство, которое позволяет адресовать любой байт кода (RAM). При загрузке программного кода в ***ОЗУ*** ему назначается базовый адрес (индекс). Все команды программного кода имеют относительную адресацию от 0. Т.о. если базовый адрес равен 10000, а код программы составляет 10000 байт, то предельный адрес будет равен 20000. При обработке команд ***МП*** проверяет, не является ли адрес текущей команды больше предельного, если нет – обращается к следующей.

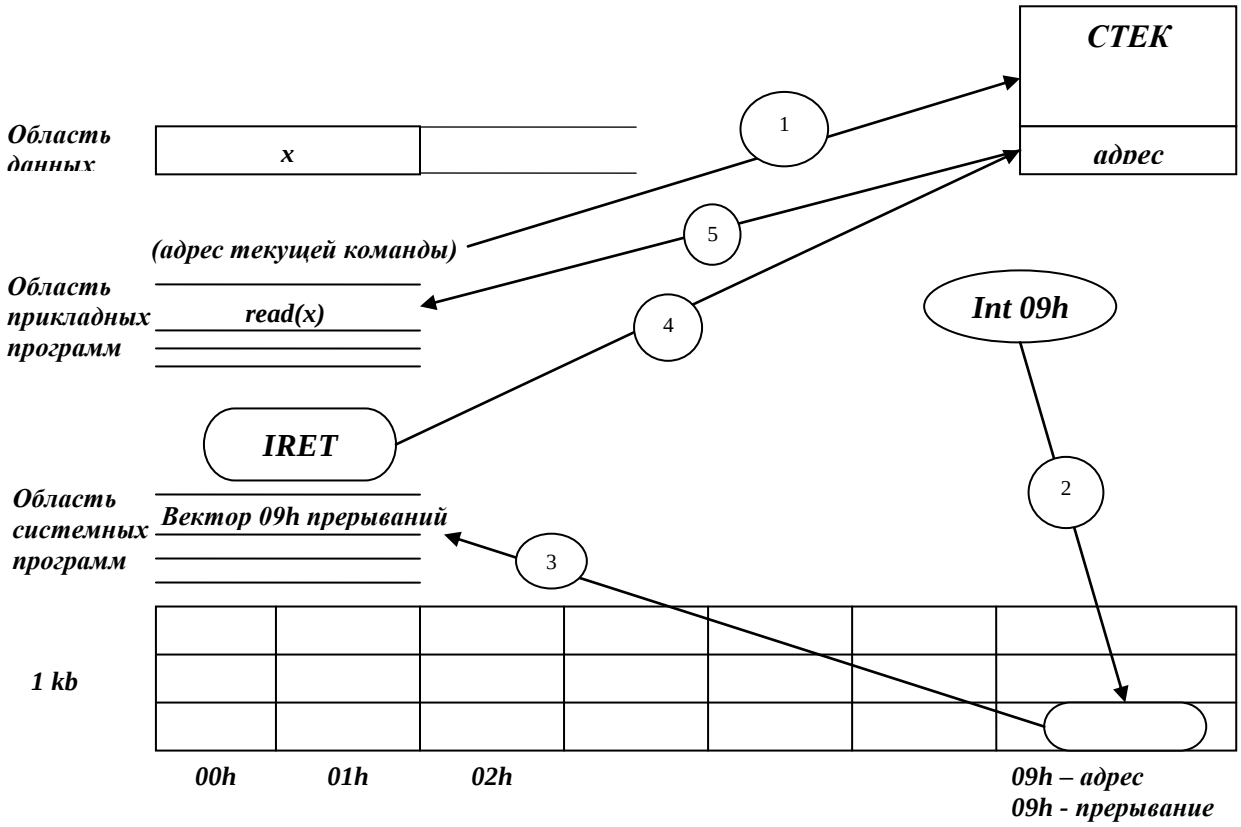
При мультипрограммном режиме необходимо обеспечить защиту данных, а так же перемещение программ в памяти. Этим занимается устройство управления памятью или диспетчер памяти, он находится в схеме микропроцессора, либо между ***МП*** и памятью. Защита данных в памяти обеспечивается таким образом, чтобы базовые адреса программных кодов не перекрывались. Данные программы могут находиться отдельно от программного кода, но диспетчер памяти отслеживает все адреса, относящиеся к данной программе. При переключении на другую задачу адреса прежней программы записываются в стек, а регистры ***МП*** обновляются в соответствии с новыми адресами другой программы.

2.3.2 Понятия прерывания. Система прерываний. Виды прерываний.

Система прерываний (Interrupt, Int) – совокупность аппаратных и программных средств по обработке прерываний.

Прерывание (Int) – возникает в ***МП*** при возникновении какой-либо ситуации, требующей реакции ***ОС***. Каждое прерывание идентифицируется своим номером и каждое из них обрабатывается специальной программой (драйвером прерываний). Различают аппаратные и программные прерывания.

00h - адрес обработчика прерываний
01h вектор прерываний 0000:0000
02h
...
09h



С каждым номером прерывания связана программа обработки данного прерывания, адрес первой команды обработчика – вектора прерываний – хранится в таблице векторов прерываний (IDT – Interrupt Description Table – 1 kb). Под каждый адрес отводится 4 байта: 0000:0004 – 0000:0008 и того всего может быть предусмотрено 256 прерываний.
Прерывания могут быть маскируемыми и немаскируемыми (выполняются всегда). В PSW имеются разряды, в которые можно занести код маски системы, при этом могут быть проигнорированы некоторые ситуации, которые бы вызвали прерывание.
Описание схемы обработки прерываний.

1) Если при выполнении очередной команды ***МП*** встретилась команда, например, READ(***fp***) - читать из файла – то возникнет прерывание. В СТЕК записывается адрес команды, которую ***МП*** должен выполнить следующей, содержимое текущего регистра PSW, а также адрес переменной ***(x)*** и указатель на файл ***fp*** из которого должны быть

прочитаны данные. Затем оператору READ выставляется прерывание с определенным номером. По этому номеру должен быть подключен обработчик системного вызова.

2) ***МП*** обращается в таблицу векторов прерываний и в соответствии с номером прерывания (умноженная на 4 байта) находит соответствующий вектор этого прерывания – адрес первой команды обработчика
3) С помощью специальной программы обслуживания прерывания ISR, ***МП*** переходит в режим ядра и обращается к системному обработчику данного прерывания драйвера устройства. Данная программа из Стека считывает всю необходимую информацию для выполнения указанной операции считывания из файла.
4) Последней командой любого обработчика прерываний является команда IRET (возврат). По данной команде содержимое Стека последовательно переписывается в регистры ***МП***, т.е. восстанавливается содержимое PSW и счетчика адреса команды, PC на прерванной команде.
5) После восстановления содержимого регистров ***МП*** продолжается обычный ход выполнения программы пользователя, при этом ***МП*** опять переходит в режим пользователя.

Обычно ***МП*** может одновременно иметь несколько прерываний: выполнение программы, сигнал от ПУ и т.д. В этом случае ***МП*** выбирает более приоритетное прерывание, а остальное блокирует на некоторое время.

Классы прерываний

1) Программное прерывание - генерируется в результате выполнения команд. Такими ситуациями могут быть – арифметическое переполнение регистров, деление на 0, выполнение некорректной команды, ссылка на область памяти, доступ к которой пользователю запрещен.
2) Прерывание по таймеру - генерируется таймером ***МП***. Данное прерывание позволяет ***ОС*** выполнять некоторые свои функции периодически через заданные промежутки времени.
3) Прерывание ввода-вывода - генерируется контроллером ввода-вывода. Сигнализирует о нормальном завершении операции или о наличии ошибок.
4) Аппаратное прерывание - генерируется при возникновении аварийных ситуаций – падение напряжения в сети, отсутствие четности в памяти.

Если нет аварийной ситуации, то более приоритетными являются программные прерывания, затем прерывания по таймеру или от другого ПК, менее приоритетные прерывания ввода-вывода.

При обработке прерываний возможно два подхода:

1. при получении нескольких прерываний обрабатывать текущее, а остальное запретить, при этом ***МП*** может и должен игнорировать любой новый сигнал прерывания. Все прерывания обрабатываются в строго последовательном порядке.
2. при втором подходе учитывается приоритет прерывания, что позволяет приостановить обработку прерывания с более низким приоритетом (например, устройству ввода-вывода принтера, диску и коммуникационной линии присвоены приоритеты 2, 4, 5) Если в некоторый момент времени ***МП***, обрабатывая прерывания от принтера, получил сигнал от сканера, то ***МП*** приступит к его обработке, при этом вся информация о состоянии принтера будет сохранена до окончания обработки текущего прерывания. Этим занимается программа ISR.

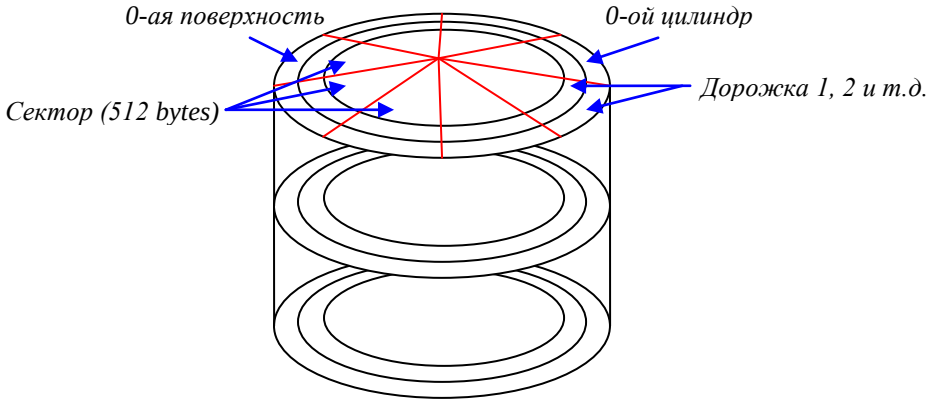
Для поддержки аппаратных прерываний имеется контроллер прерываний – микросхема – IRQ, которая обеспечивает связь с помощью 15 каналов прерываний. Номер прерывания, который присваивается каждому каналу прерываний, определяет ту программу (системный обработчик), которой необходимо передать управление после того, как ***МП*** принял сигнал прерывания. Таким образом, с помощью системы прерывания обеспечивается выполнение всех операций ввода-вывода путем обращения ***ОС*** к драйверу соответствующего ПУ.

2.3.3 Организация ввода-вывода (input-output).

Организация операций IO одна из важнейших функций ОС и обеспечивается комплексом аппаратных и программных средств. Устройства IO состоят из 2-х частей – из контроллера и из самого ПУ.

Контроллер – микросхема или набор микросхем на вставляемой в разъем плате и физически управляющим устройством.

Он принимает команды ОС (например, прочесть сектор 11190 с диска 2), контроллер должен преобразовать линейный номер сектора в номер цилиндра, номер сектора и головки считывающего устройства.



Физически адрес сектора складывается из: *адреса цилиндра, рабочей поверхности* – головки считывающего устройства, *и сектора.*

1 сектор равен 512 bytes (поддерживается BIOS). За один цикл считывается 1 сектор и обмен информации между ОЗУ и МД осуществляется секторами или блоками по 512 байт.

По типу обмена данными между ПУ и ОЗУ различают:

5) блочные ПУ – винчестер, CD-ROM, floppy диски
6) Символьные ПУ – обмен информацией посимвольно (принтер, графопостроитель, монитор, клавиатура, сканер, мышь)

Порты ввода/вывода.

Для организации взаимодействия с контроллером может быть использовано следующее аппаратное средство: COM - **порт.**

В этом случае контроллер или объединенные сетью контроллеры подключаются по протоколам **RS-232**, **RS-422**, **RS-485.**

Портами ввода-вывода (port input/output), вот два их определения из книжки:

1)“**порт ввода/вывода** – это 8,16 или 32-разрядный аппаратный регистр (не путать с регистрами процессора! - прим. от меня), имеющий определенное количество адресов в адресном пространстве ввода-вывода”

2) **порты ввода/вывода** - это пункты системного интерфейса ПК, через которые МикроПроцессоры (МП) обмениваются информацией с другими устройствами.

Всего у МП портов ввода/вывода может быть **65536**. Каждый порт ввода/вывода имеет свой адрес.

Адрес порта ввода/вывода – это **номер порта**, соответствующего адресу ячейки памяти, являющегося частью устройства ввода/вывода, использующего этот порт, а не частью основной памяти компьютера.

“ **Номер порта (number of port)**” - ну как объяснить? Вот десять портов (предположим) - так что им, каждому имя давать? Зачем? Когда их можно просто пронумеровать 0..9. На самом деле их количество ограничено размерностью адр. пр-ва ввода-вывода - FFFFh штук. Например, таймер имеет 4 порта - 40h..43h для различных каналов. (Не буду я тут схему работы таймера разбирать!!!) Фактически, обращаясь к портам i/o, вы общаетесь с устройствами почти напрямую. Для этого используются команды **in** и **out**:

in <аккумулятор - eax/ax/al> - из порта считывается байт/слово/дв. слово в аккумулятор.

out - содержимое аккумулятора пересылается в порт N. Если номер порта **<0FFh**, то можно задавать его непосредственно, иначе – через:

dx:xor ax,ax

mov dx,100h - порт с большим номером.

out dx,ax - выводим в этот порт нолик.

Кроме того, для вывода/ввода не по байтам, а строчками, есть команды:

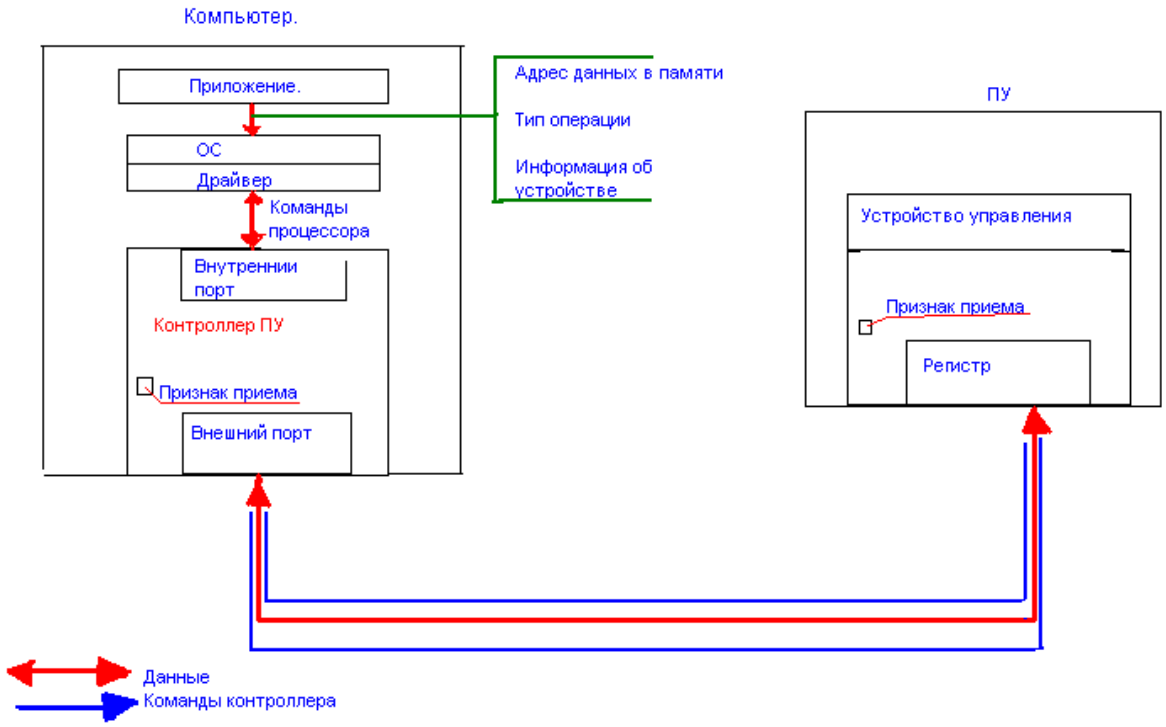
ins,outs (вернее **ins/insb/insw/insd** и ан-но **outs/.**) - перед вызовом их в **dx** вносим номер порта,

в **es:di** - строку для вывода,

а для **ins** - результат будет по адресу: **es:di**.

Контроллеры ввода/вывода.

Программа, выполняемая процессором, может обмениваться данными с помощью команд ввода/вывода с любыми модулями, подключенных к внутренней шине компьютера, в том числе и с **контроллерами** ПУ.



Контроллер ввода/вывода (controller input/ output) - используется для сбора данных с датчиков имеющих открытый коллектор, кнопок, переключателей или через блок гальванической развязки от датчиков другого типа, а также управления светодиодами или силовыми модулями типа Pwr3, PwrR и другими. Контроллер имеет 22 линии двунаправленного ввода/вывода.

Контроллер обеспечивает функции:

1. Управляет механизмом ПУ
2. преобразует коды символов в машинные (двоичные)
3. накапливает передаваемую информацию до полной записи
4. согласует по скорости работу ПУ с МП

Так как все типы контроллеров отличаются друг от друга, для управления ими требуется различное программное обеспечение (ПО). Программа, управляющая работой контроллера (отдает команды, получает ответы) называется драйвером.

Драйвер ввода/вывода (driver input/output) – это специальная резидентная программа, которая дополняет систему ввода и вывода, и обеспечивает обслуживание дополнительных внешних устройств. Он загружается в памяти компьютера при загрузке ОС, и хранится в файлах, имена которых имеют расширение: **.sys.**

Каждый производитель контроллеров должен поставлять драйверы для поддерживаемых им ОС.

Есть три способа установки драйвера в ядро:

1. Unix – заново компонуется ядро с новым драйвером и перезагружается
2. Windows – создается запись во входящим в ОС файле, в котором указывается, что требуется драйвер, если система его не находит при первоначальной загрузке, то он должен быть переписан с диска, после ОС должна быть перезагружена
3. Windows 2000, Windows XP – драйверы устанавливаются без перезагрузки ОС.

Каналы ввода/вывода.

Канал ввода/вывода (канал связи) – является общим звеном любой системы передачи информации. Он реализует процесс передачи информации по информационному каналу. Информационный канал (ИК)– это логический канал. Он устанавливается между двумя ЭВМ, соединенных физическим каналом.

ИК обеспечивает прием и передачу потока данных в виде кадров, в которые упаковываются информационные пакеты, обнаруживает ошибки передачи и реализует алгоритм восстановления информации в случае обнаружения сбоев или потерь данных.

По физической природе каналы связи бывают:

1) механические – обеспечивают передачу материальных носителей информации;
2) акустические – обеспечивают передачу звуковых сигналов;
3) оптические – обеспечивают передачу световых сигналов;
4) электрические – обеспечивают передачу электрические сигналы(Электрические каналы связи бывают проводные и беспроводные).

По форме представления передачи информации каналы связи делятся на:

1) аналоговые – передача информации представляется в непрерывной форме, т.е. в виде непрерывного ряда значений какой-либо физической величины;
2) дискретные – передача информации представляется в виде дискретных (цифровых, импульсных) сигналов той или иной физической природы.

Только возможностями контроллера не удается согласовать работу центральной части ПК и ПУ при обмене данными. Поэтому вводится еще понятие канала ввода-вывода (IO channel).

Понятие канала IO включает в себя: устройства ПУ, контроллер, драйвер, а так же внутренний блок управления ОС для файла или устройства.

Различают 2 типа канала:

1. мультиплексный – для работы с низкоскоростными устройствами (принтерами, сканерами, плоттерами)
2. селекторный – для работы с быстродействующими устройствами (CD-ROM)

Селекторные каналы работают в импульсном режиме, присылая или принимая в импульсе непрерывный поток данных.

ОС для каждого из каналов IO открывает процесс или файл.

Например, при загрузке MS-DOS автоматически открывается 5 стандартных процессов для передачи данных, связанных с файловыми манипуляторами.

0 – стандартный процесс, обеспечивающий ввод информации, связанный с файлом INPUT и связан с клавиатурой (CONsole)

1 – стандартный процесс OUTPUT для вывода информации на монитор (CONsole)

2 – ERROR - для вывода сообщений об ошибке, связан с монитором.

3 – стандартный выход для коммуникационного (последовательного) порта (AUX, COM1)

4 – стандартный выход, связанный с выводом на принтер (PRN, LPT1)

Процесс – это запущенная на выполнение программа с выделенными ей ресурсами. Каждый из перечисленных процессов имеет канал, который связывает указанный процесс с соответствующим устройством.

Процессы 0 и 1 можно переназначить на другие устройства (например, дисковые файлы), другие процессы 3 и 4 также можно переназначаются программными средствами.

К одному дисковому файлу можно открыть одновременно несколько каналов из одного или различных процессов (один канал для последовательного доступа к файлу, другой для прямого). Канал представлен в виде специализированного процессора, который занимается только организацией ввода-вывода.

Задачи канала:

Буферизация данных, подсчет байтов, введение счетчика адресуемой памяти.

Канал обеспечивает связь с центральным процессором с помощью 3-х полей:

1. CAW – Channel Address Word – адресное слово канала – содержит адрес начала программы работы канала.
2. CSW – Channel Status Word – слово состояния канала – содержит адрес данных, счетчик байтов, а также поле состояния, в котором фиксируется информация о занятости устройства, о завершении работы канала, о завершении канала ПУ, т.е. состоянии операции ввода-вывода
3. CCW – Channel Command Word – команда канала – содержит код команды (чтение, запись), адрес данных, определяющий место в ОЗУ, где искать данные или куда записать, поле счетчика байтов.

Связь с каналом реализуется через команды:

1) начать ввод-вывод – SIO
2) остановить ввод-вывод – MIO
3) опросить канал – TCH
4) опросить ввод-вывод – TIO

Это привилегированные команды, т.е. могут использоваться только модулем ОС в режиме ядра.

Порядок организации операций ввода-вывода:

1) Если в прикладной программе (ПП) встретилась команда read или write, то в МП происходит прерывание, организуется передача управления модулю ОС, при этом должен быть передан описатель устройства, с которым должна быть обеспечена связь. Каждая команда IO ориентирована на определение устройства.

2) Перед началом операции ввода-вывода модулем ОС посылается в CAW адрес начала программы канала, а также в CCW заносится адрес данных и значение счетчика байтов.

3) Модуль ОС посылает сигнал опроса через прерывание о состоянии канала для выполнения ввода-вывода.

4) Если канал свободен, то он отвечает соответствующим сигналом, который вносит код состояния в поле CSW.

5) Если канал свободен, то модуль ОС выдает команду SIO и канал запускает операцию ввода-вывода. МП в это время может передать управление другой задаче.

6) После окончания операции ввода-вывода канал через прерывание посылает сигнал МП об окончании операции, приняв данное прерывание, МП возвращается вновь на прерванную программу.

2.3.4 Управление заданиями (процессы, задачи). Очереди.

Процесс – минимальный программный объект, обладающий системными ресурсами.

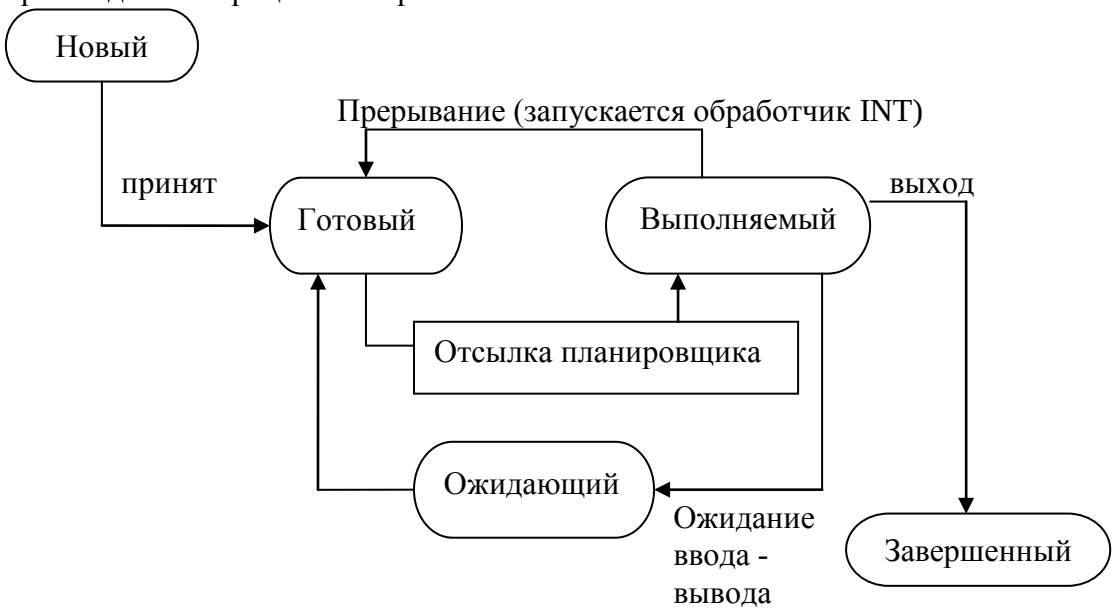
Программа – это план действий, с процессами действий, поэтому понятие процесса включает в себя:

1) программный код
2) данные
3) содержимое СТЭК.
4) Содержимое адресного и других регистров МП.

Для одной программы могут быть созданы несколько процессов в том случае, если с помощью одной программы в МП выполняется несколько несовпадающих последовательностей команд. Различают следующие состояния процесса:

- новый (процесс только что создан)
- выполняемый (команды программы выполняются в МП).
- ожидающий (процесс ожидает завершение некоторого события, например операция ввода – вывода).
- готовый (процесс ожидает освобождения МП).
- завершенный (процесс завершил свою работу).

Каждый процесс представлен в ОС набором данных, называемых таблица управления процессом PCB (Process Control Block). В PCB процесс описывается набором значений, параметров, характеризующих его текущее состояние и используемых ОС для управления прохождением процессом через ПК.



По времени развития процессы делятся на: последовательный, параллельный, комбинируемый.

По месту развития процессы делятся на : внутренние(реализуются на МП), внешние (на внешних процессорах (в сети) сервер).

По принадлежности к ОС: системные и пользовательские процессы.

По связанности различают процессы:

1) взаимосвязанные (информационные, управляющие, пространственно - временные).
2) Изолированные.
3) Информационно - независимые (общие ресурсы у них, но имеют собственные информационные базы).
4) Взаимодействующие.
5) Взаимосвязанные по ресурсам.
6) Конкурирующие.

Процессы могут, находится в отношении:

1) предшествования.
2) Приоритетности (процесса может быть переведен в активное состояние, если нет процессов с более высоким приоритетом).
3) Взаимного исключения (когда используются общий критический ресурс).

Ресурс – средство вычислительной системы, которая может быть выделена процессу на интервал времени.

Классификация ресурсов.

1) Делятся на: физические и виртуальные.
2) По времени существования ресурсы делятся: постоянные и временные.
3) Основные и второстепенные (допускаются альтернативное развитие процесса при их отсутствии).
4) Простые и составные: (доступен или занят – только два состояния).
5) Потребляемые и воспроизводимые (допускают многократное использование и освобождение).
6) Последовательное и параллельно – используемые ресурсы.
7) Жесткие (не допускают копирования) и мягкие (допускают тиражирование).

Управление процессами и распределение ресурсов между ними ОС осуществляет с помощью планировщика. ОС контролирует следующую деятельность, связанную с процессами:

а) создание и удаление.
б) планирование процессов.
в) синхронизация процессов.
г) коммуникация процессов.
д) разрешение тупиковых ситуаций.

Одним из методов планирования процессов, ориентированных на эффективную загрузку ресурсов являются методом очередей ресурсов.

Новые процессы находятся во входной очереди – очередь работ – заданий. Входная очередь располагается во внешней памяти, и ожидают освобождение ресурсов (место в ОЗУ). Готовые к выполнению процессы располагаются в ОЗУ и связаны с очередью готовых процессов и ожидают освобождения ресурса – процессорного времени.

Процесс в состоянии ожидания завершения операции ввода – вывода, находится в одной из очередей к оборудованию ввода – вывода.

Процесс мигрирует между различными очередями под управлением программы, которая называется планировщик.

ОС, обеспечивающий режим мультипрограммирования обычно включает 2 планировщика – долгосрочный и краткосрочный. Долгосрочный планировщик – планировщик заданий. На уровень долгосрочного планирования выносятся редкие системные действия требующие больших затрат системных ресурсов. Объектом является не отдельный процесс, а некоторые объединения процессов по функциональному назначению. Долгосрочный планировщик решает, какой из процессов во входной очереди должен быть переведен в очередь готовых процессов, при освобождении ресурсов памяти.

Краткосрочной планировщик – супервизор решает, какой из процессов в находящейся очереди готовых процессов должен быть передан на выполнение микропроцессоров. На уровне краткосрочного планирования выносятся частные и более короткие процессы. В некоторых ОС долгосрочный планировщик может отсутствовать, например, в системах разделения времени, при этом каждый новый процесс сразу помещается в ОЗУ. Выделение МП процессу производится многократно с целью достижения эффекта мультипрограммирования назначение – диспетчеризация.

Процессы могут взаимодействовать между собой. И называются взаимодействующими либо могут быть независимыми.

При взаимодействии рассматривают процесс производитель, процесс потребитель. При этом создается совместный буфер, заполняемый процессом производителя.

Буфер имеет фиксированный размер, поэтому процессы могут находиться в процессе ожидания, когда:

1) Буфер заполнен – ожидает процесс производитель.
2) Буфер пуст – ожидает процесс потребитель.

Буфер может предоставляться и поддерживаться самой ОС либо должен быть организован программистом, общий участок памяти.

Взаимодействие заключается в передачи данных между процессами или в совместном использовании ресурсов. Обычно реализуется при помощи таких механизмов как:

1) Транспортеры, очереди, сигналы, семафоры, DDE, OLE, Clipboard. Взаимодействие аналогично считыванию или записи в файл. Расположен в памяти, при считывании данных из транспортера реализуется алгоритм, последовательные данные не подлежат повторному чтению данных. ОС контролирует порядок размещения данных на транспортере и наличие свободного места.
2) Очереди – механизм, который обеспечивает передачу или использование общих данных бес перемещения этих данных, а лишь по указателю этого элемента. Очередь используется вместе с механизмом общей памяти. Элемент очереди может быть считан с уничтожением или без уничтожения этого элемента. Чтение элементов очереди осуществляет только создающий очереди процесс, все другие процессы могут только записать элемент в очередь. Имя очереди имеет вид полной спецификации файла. Записывающий процесс осуществляет действия: открыть очередь, записать в очередь, закрыть очередь. Данный механизм позволяет эффективно передавать только элемент очереди, указывающий расположение этих данных.
3) Сигналы – являются механизмом передачи требования от одного процесса к другому на немедленное выполнение действия (является аналогом обработки прерывания). Характер выполняемых действий при возникновении сигнала: обработка системной ошибки, блокирование сигнала, передача управления подпрограмме.
4) Семафоры – являются механизмом передачи сообщений от одного потока к другому о наступлении некоторого события. Различают системные семафоры и семафоры ОЗУ.
5) В , MS Windows существует специальный механизм для взаимодействия процессов в реальном масштабе времени. DDE (динамический обмен данными) он стандартизует процесс обмена командами, сообщениями и объектами для обработки между задачами (организация печати). Другим интерфейсом для обмена данными является OLE – связь (Object Linking and Embedding) связывание и встраивание объектов. Данный интерфейс позволяет хранить объекты одной программой в объектах созданных другой программой, а также редактировать без нарушения цельностей информации и связей (текстовый документ Word и рисунок из Paint).
6) Буфер обмена (самый простой интерфейс межпрограммного обмена данными) Clipboard. Буфер обмена – область ОЗУ, которая может содержать в себе один информационный объект (рисунок и т.д.) с помощью системного вызова процесс может получить копию информации содержащейся в буфере обмена, или сам поместить в буфер.