1. **Особенности ОС с монолитным ядром. Достоинства и недостатки.**

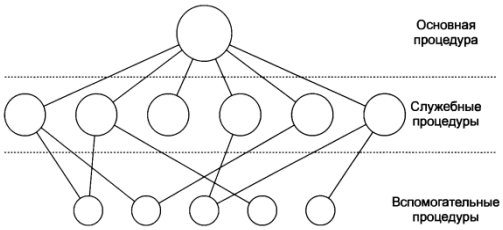
Монолитные - Здесь вся операционная система работает как единая программа в режиме ядра. Операционная система написана в виде набора процедур, связанных вместе в одну большую исполняемую программу При использовании этой технологии каждая процедура может свободно вызвать любую другую процедуру, если та выполняет какое-нибудь полезное действие, в котором нуждается первая процедура. Наличие нескольких тысяч процедур, которые могут вызывать друг друга сколь угодно часто, нередко приводит к громоздкой и непонятной системе. Для построения исполняемого файла монолитной системы необходимо сначала скомпилировать все отдельные процедуры (или файлы, содержащие процедуры), а затем связать их все вместе, воспользовавшись системным компоновщиком. Здесь, по существу, полностью отсутствует сокрытие деталей реализации — каждая процедура видна любой другой процедуре (в отличие от структуры, содержащей модули или пакеты, в которых основная часть информации скрыта внутри модулей, и за пределами модуля его процедуры можно вызвать только через специально определяемые точки входа). Тем не менее даже такие монолитные системы могут иметь некоторую структуру. Службы (системные вызовы), предоставляемые операционной системой, запрашиваются путем помещения параметров в четко определенное место (например, в стек), а затем выполняется инструкция trap. Эта инструкция переключает машину из пользовательского режима в режим ядра и передает управление операционной системе. Затем операционная система извлекает параметры и определяет, какой системный вызов должен быть выполнен. После этого она перемещается по индексу в таблице, которая в строке k содержит указатель на процедуру, выполняющую системный вызов k.

2. Такая организация предполагает следующую базовую структуру операционной системы:

1. Основная программа, которая вызывает требуемую служебную процедуру.

3. 2. Набор служебных процедур, выполняющих системные вызовы.

4. 3. Набор вспомогательных процедур, содействующих работе служебных процедур.



5. В дополнение к основной операционной системе, загружаемой во время запуска компьютера, многие операционные системы поддерживают загружаемые расширения, в числе которых драйверы устройств ввода-вывода и файловые системы. Эти компоненты загружаются по мере надобности.

2. Особенности перехода в заблокированное состояние и выхода из него.

Во время выполнения процесса ему могут понадобиться дополнительные ресурсы, но для их получения необходимо некоторое время. Т.е. процесс должен ожидать наступления некоторого события или окончания выполнения конкретной операции (например, ввода/вывода для получения/выдачи данных). Такой процесс переводится в **заблокированное** состояние.

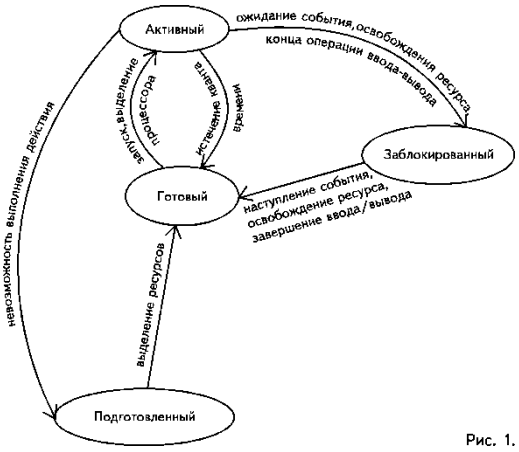
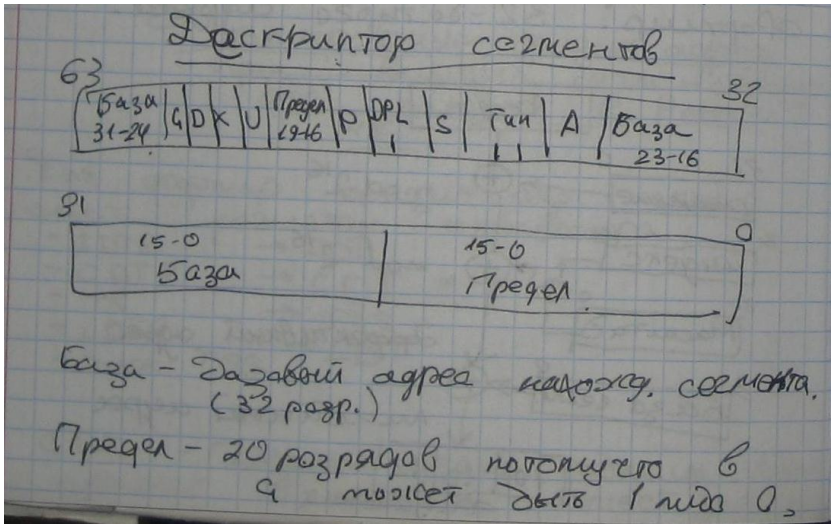


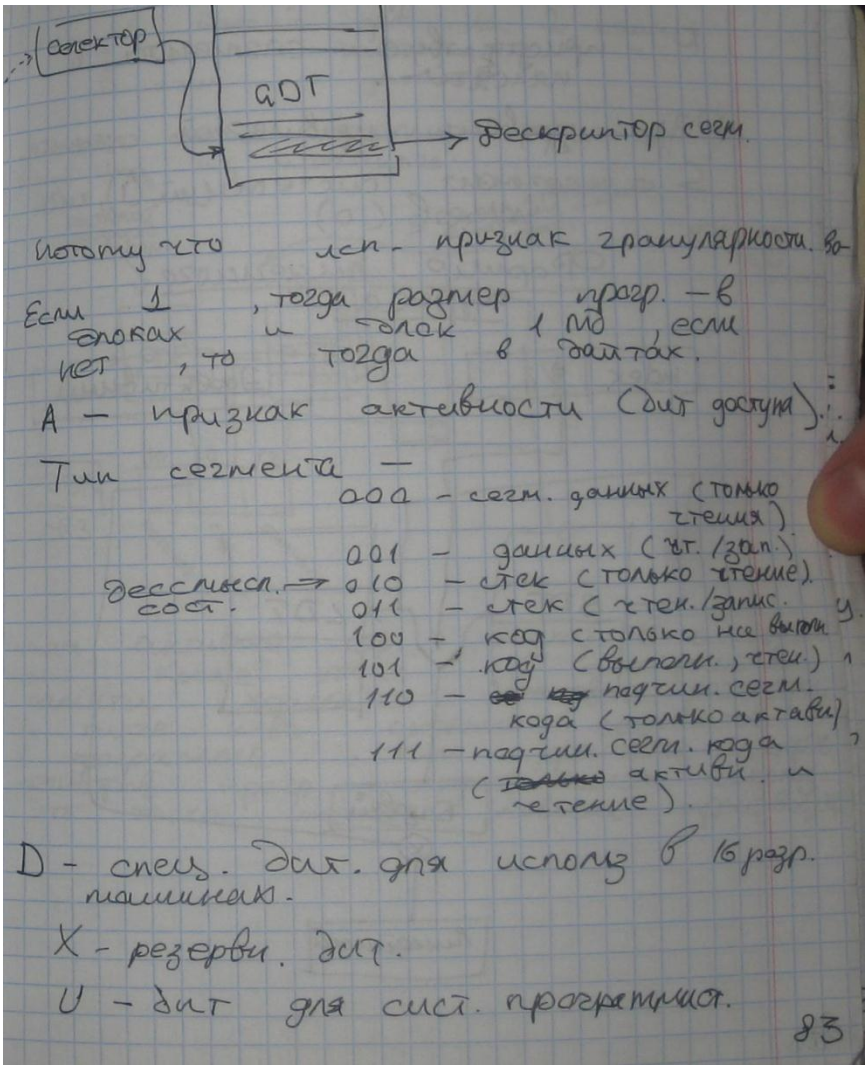
Рис. 1.

Процесс, которому выделяется время процессора, переводиться в активное состояние или состояние выполнения. Во время выполнения процесса ему могут понадобиться дополнительные ресурсы, но для их получения необходимо некоторое время. Т.е. процесс должен ожидать наступления некоторого события или окончания выполнения конкретной операции (например, ввода/вывода для получения/выдачи данных). Такой процесс переводится в заблокированное состояние.

Если активный процесс не выполнился за выделенный ему квант времени, то он переходит снова в готовое состояние.Процесс, который требует дополнительных ресурсов, кроме времени процессора, либо выполнения некоторой операции другим процессом, переводится в заблокированное состояние и ожидает наступления события, которое он потребовал. Как только произошло ожидаемое событие, процесс переводится в готовое состояние.

3. Структура дескриптора сегмента и назначение его частей





4. Алгоритм банкира. Достоинства и недостатки

Алгоритм "банкира".

При использовании алгоритма "банкира" подразумевается, что :

1) системе заранее известно количество имеющихся ресурсов;
2) система знает, сколько ресурсов может максимально потребоваться процессу;
3) число пользователей (процессов), обращающихся к ресурсам, известно и постоянно;
4) система знает, сколько ресурсов занято в данный момент времени этими процессами (в общем и каждым из них);
5) процесс может потребовать ресурсов не больше, чем имеется в системе.

В алгоритме "банкира" введено понятие надежного состояния. Текущее состояние вычислительной машины называется **надежным**, если ОС может обеспечить всем текущим пользователям (процессам) завершение их заданий в течение конечного времени. Иначе состояние считается **ненадежным**. Надежное состояние — это состояние, при котором общая ситуация с ресурсами такова, что все пользователи имеют возможность со временем завершить свою работу. Ненадежное состояние может со временем привести к тупику.

Система удовлетворяет только те запросы, при которых ее состояние остается надежным, т.е. при запросе ресурса система определяет сможет ли она завершить хотя бы один процесс, после этого выделения.

Пример решения задачи выделения ресурсов по алгоритму "банкира".

Имеем 6 ресурсов и 6 процессов (рис .2.17.). В таблице показано состояние занятости ресурсов на данный момент (1 свободный ресурс).

Процесс	Выделенное число ресурсов	Требуемое число ресурсов
A	2	3
B	1	3
C	0	2
D	1	2
E	1	4
F	0	5

Рис. 2.17

Эта таблица отражает надежное состояние, т.к. существует такая последовательность выделений — освобождений ресурсов, при которой все процессы за конечное время будут завершены.

Недостатки алгоритма банкира:

1) Если количество ресурсов большое, то становится достаточно сложно вычислить надежное состояние.
2) Достаточно сложно спланировать, какое количество ресурсов может понадобиться системе.
3) Число работающих пользователей (процессов) остается постоянным.
4) Пользователи должны заранее указывать максимальную потребность в ресурсах, однако, потребность может определяться динамически по мере выполнения процесса.
5) Пользователь всегда заказывает ресурсов больше, чем ему может потребоваться.
6) Алгоритм требует, чтобы процессы возвращали выделенные им ресурсы в течение некоторого конечного времени. Однако, для систем реального времени требуются более конкретные гарантии. Т.е. в системе должно быть задано максимальное время захвата всех ресурсов процессом (через это время ресурсы должны быть освобождены).

5. Приоритетные уровни системы прерываний