

Процессы

- Программа — это статическая последовательность команд (инструкций), описывающих решение определенной задачи.
- Процесс — это система действий, реализующая выполнение программы в компьютерной системе.
- Процесс — это контейнер для набора ресурсов, используемых при выполнении экземпляра программы.
- Процесс — это идентифицируемая абстракция совокупности взаимосвязанных системных ресурсов на основе отдельного и независимого виртуального адресного пространства в контексте которой организуется выполнение потоков.

Как хранится информация о процессах ?

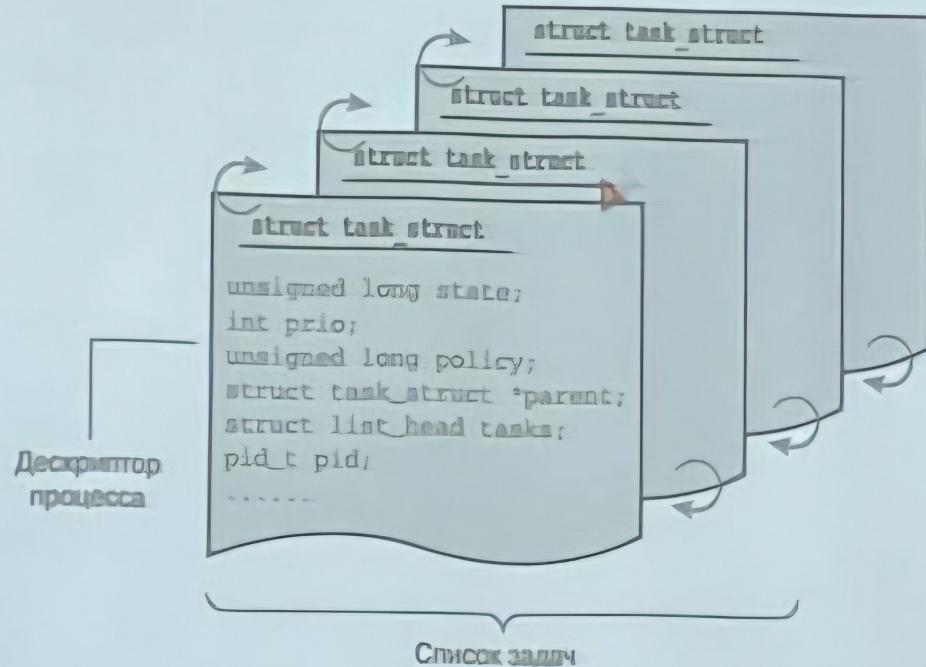
- При управлении процессами операционная система создает три информационные структуры :
 - 1) системный контекст;
 - 2) регистровый контекст;
 - 3) пользовательский контекст.

Системный контекст + регистровый контекст = Блок управления процессом (process control block) PCB или дескриптор процесса.

Что входит в системный контекст

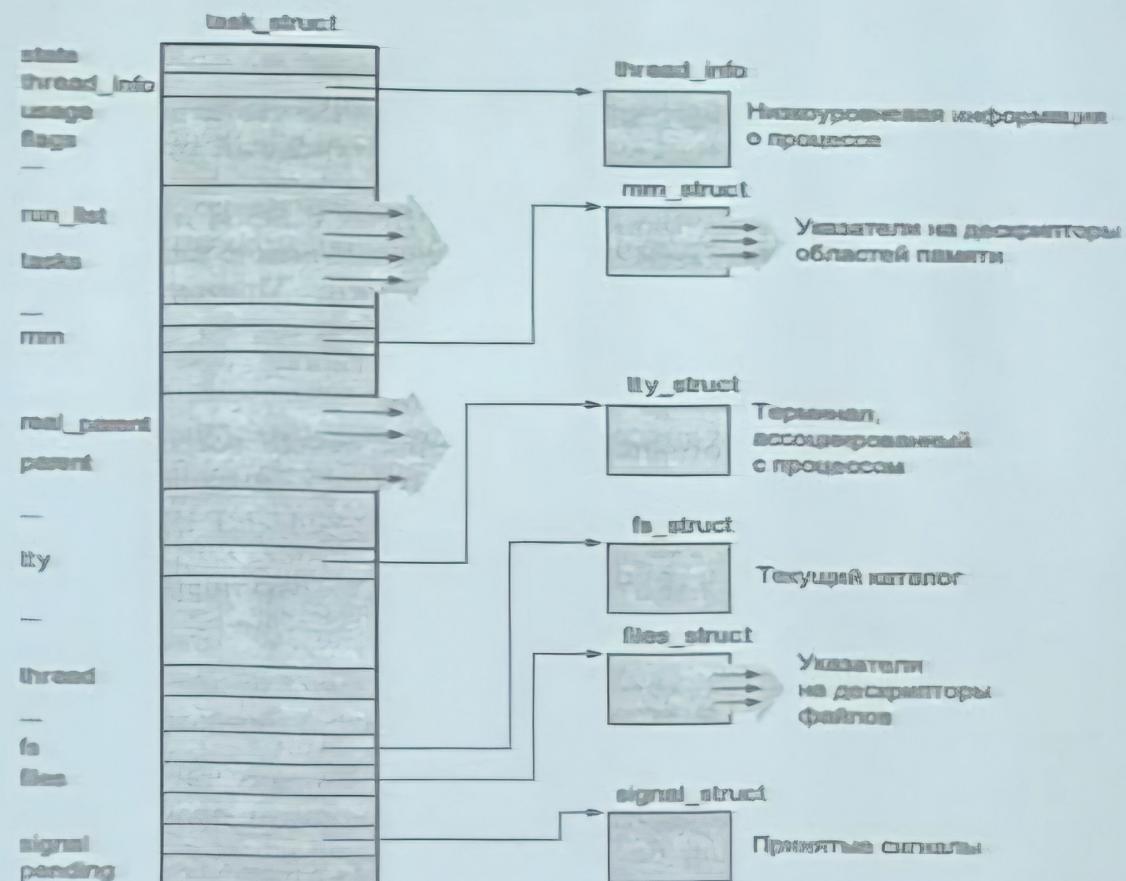
- Информация, которая необходима ОС в течение всего жизненного цикла процесса:
 - время запуска;
 - идентификатор пользователя, создавшего процесс;
 - состояние процесса;
 - расположение процесса в оперативной памяти и на диске
 - приоритет процесса;
 - используемое процессорное время;
 - информация о родственных процессах;
 - параметры планирования;
 - указатели на открытые процессом файлы;
 - информация об операциях ввода-вывода, используемая процессом.

Список процессов



Дескриптор процесса - task_struct

- 1



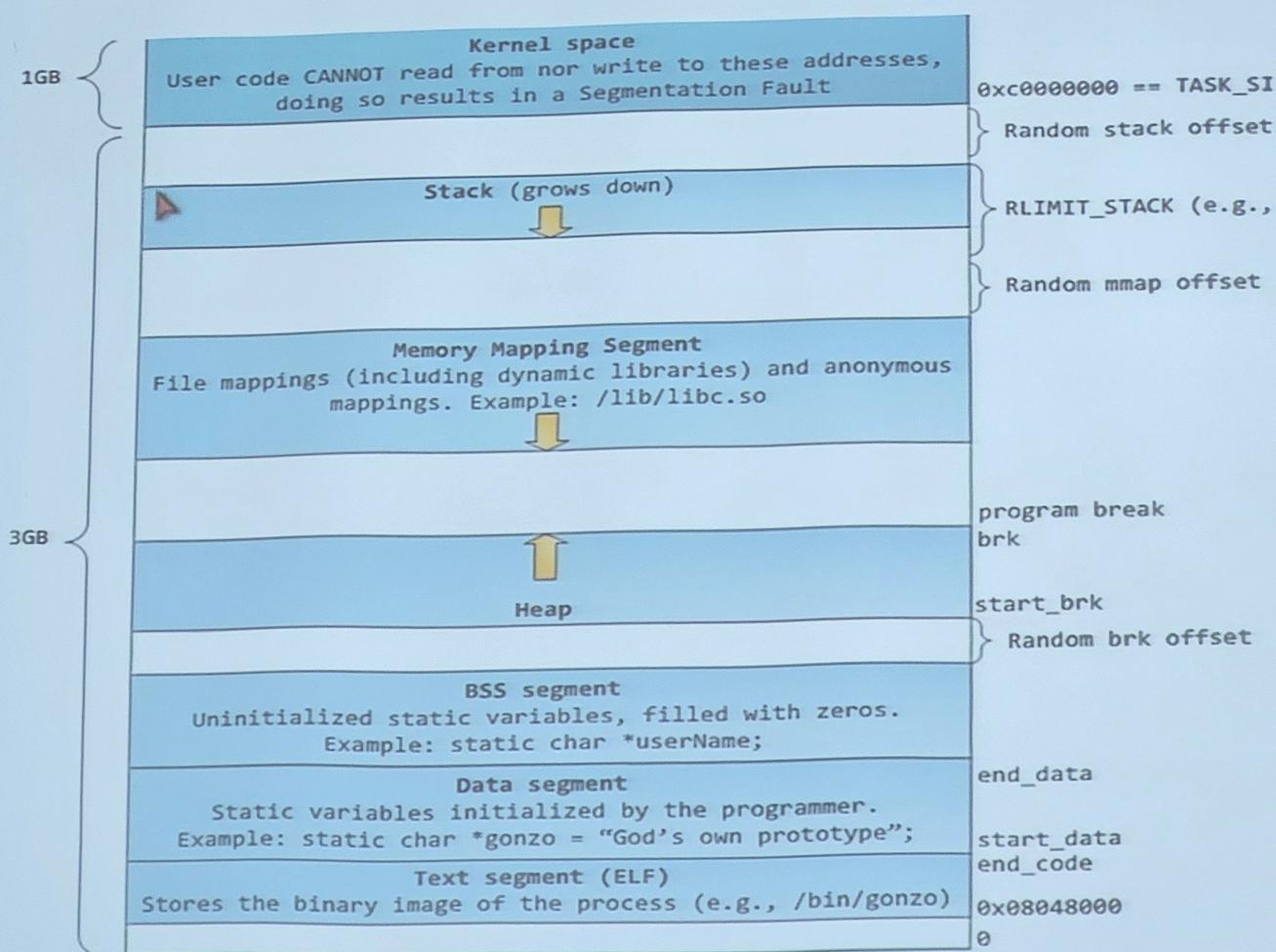
Регистровый контекст

- **Регистровый контекст :**

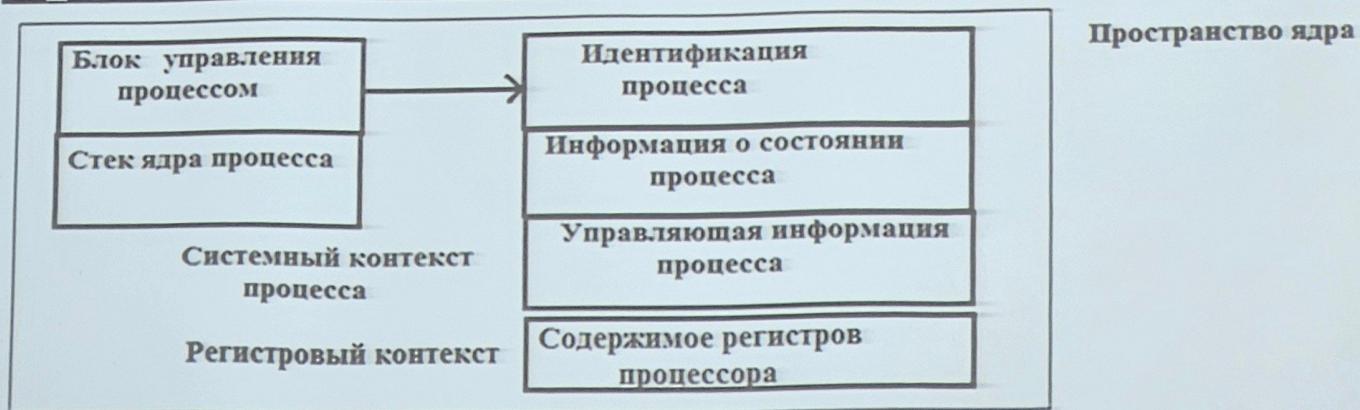
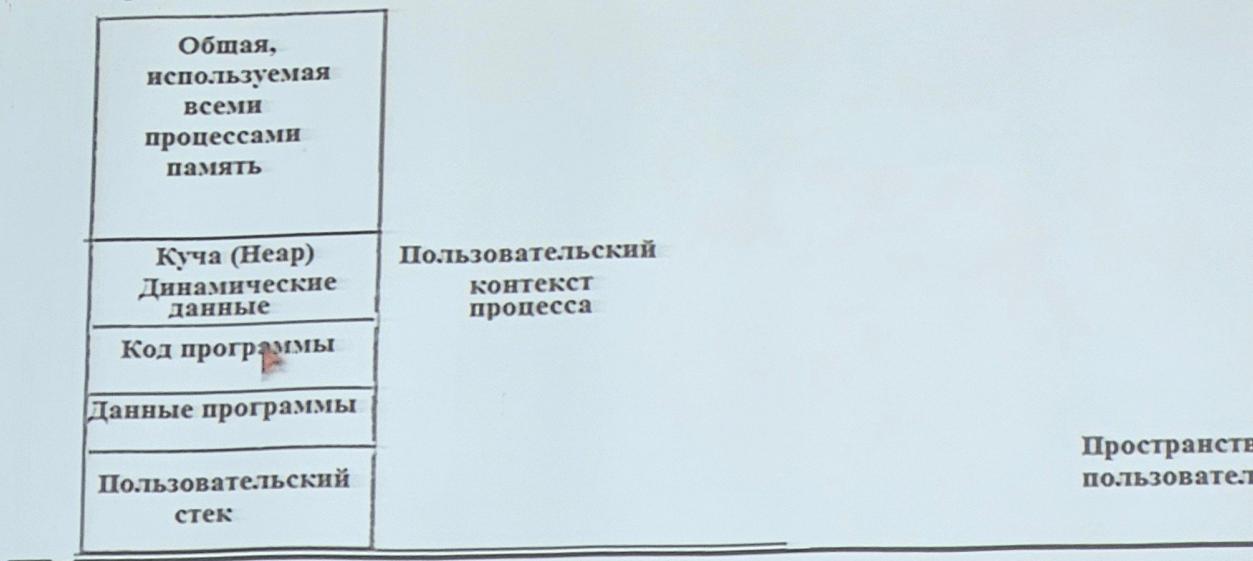
- Сохраняется текущее состояние регистров процессора каждый раз, когда ОС прерывает выполнение процесса
- Извлекается из регистрарного контекста обратно в регистры процессора, когда ОС возобновляет выполнение процесса.
- В архитектуре 80x86 имеется аппаратная поддержка специального типа сегмента памяти для хранения регистрарных контекстов — сегмент состояния задачи, или сегмент TSS (Task State Segment), который может быть выделен каждому процессу. Это позволяет переключать регистрарные контексты аппаратно,

Пользовательский контекст

■ 1



Размещение образа процесса в памяти

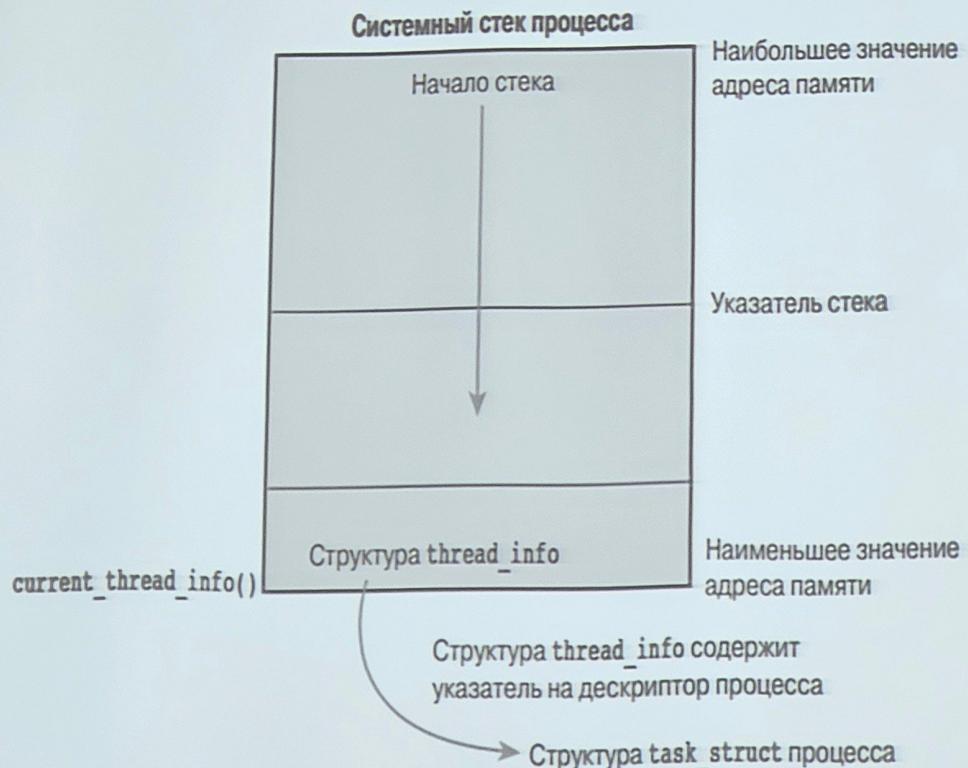


Образ процесса в памяти (все вместе)

В образ процесса в памяти входит следующая информация.

1. Команды программы
2. Данные программы
3. Пользовательский стек.
4. Heap – Куча
5. Общая разделяемая память
6. Блок управления процессом
 - Системный контекст
 - Регистровый контекст
7. Системный стек процесса.

Системный стек процесса



Состояние процесса (поле stat дескриптора)

- R – TASK_RUNNING. Процесс выполняется или находится в очереди готовых к выполнению процессов
- S – TASK_INTERRUPTIBLE. (спящий прерываемый) Процесс приостановлен (находится в состоянии ожидания, *sleeping* до тех пор, пока не будет удовлетворено некоторое условие.
- D – TASK_UNINTERRUPTIBLE (спящий не прерываемый) аналогично состоянию task_interruptible с тем отличием, что доставка сигнала спящему процессу не изменяет его состояние. T – TASK_STOPPED – Процесс переходит в это состояние, когда получает сигнал sigstop, sigtstp, sigttin и sigttou;
- t – TASK_TRACED – выполнение процесса остановлено отладчиком;
- I – TASK_IDLE – процесс бездействует).
- Z – EXIT_ZOMBIE .

Состояние процесса (поле stat дескриптора)

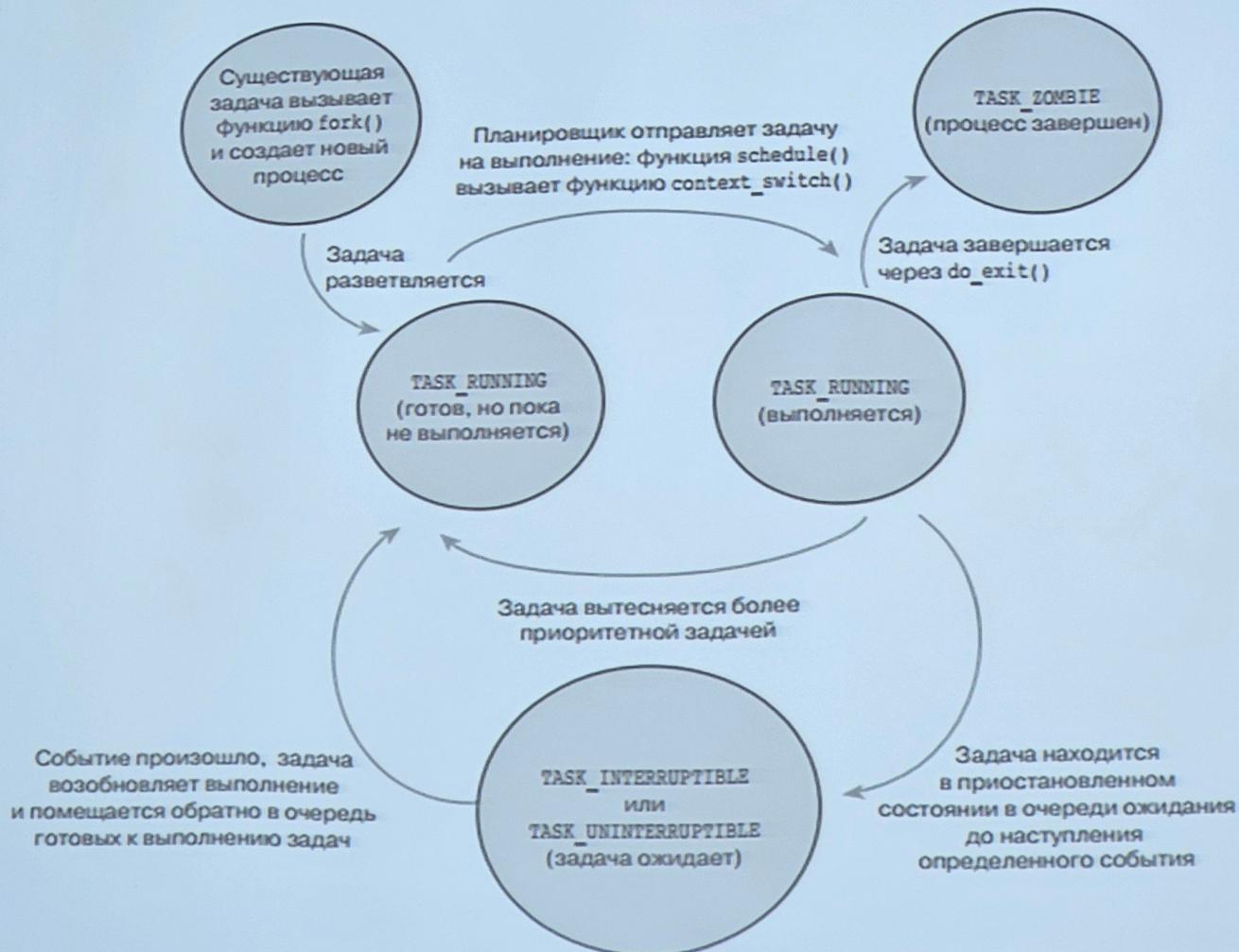
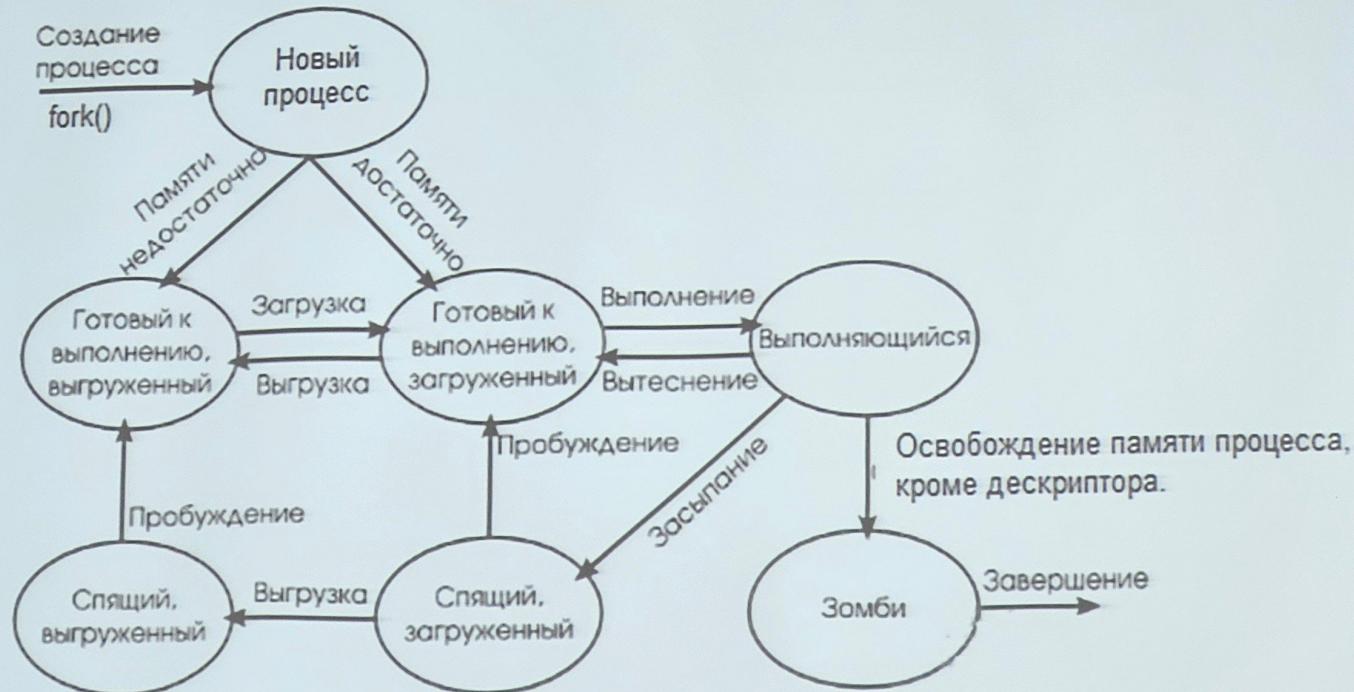


Диаграмма состояний процесса



■ Состояние зомби:

- Дочерний процесс завершился, но родитель не успел сделать системный вызов `wait()`, `waitpid()`
- Родительский процесс завершился, а дочерний нет, тогда он уничтожается процессом `init`

Типы процессов

- Системные процессы являются частью ядра ОС и всегда находятся в оперативной памяти. Они не имеют соответствующих программ в виде исполняемых
- Демоны - не интерактивные процессы, которые запускаются обычным образом (с исполняемого файла) и выполняются в фоновом режиме. Обычно демоны запускаются при инициализации системы
- Прикладные процессы. К ним относятся все остальные процессы, выполняющиеся в системе.

Функции ОС по управлению процессами

- а) Создание и завершение;
- б) Планирование и диспетчеризация;
- в) Переключение;
- г) Синхронизация(взаимодействие);

Планировщик(диспетчер) процессов - та часть операционной системы, которая занимается планированием процессов.

Последовательность создания процесса

1. **Создать** и проинициализировать блок управления процессом PCB
2. **Присвоить** новому процессу уникальный идентификатор (занести новую запись в таблицу процессов);
3. **Выделить** адресное пространство для образа процесса;
4. **Загрузить** часть команд и данных процесса в оперативную память.
5. **Поместить** процесс в очередь “готовых” процессов;

Системный вызов для создания процесса в Linux

```
#include <stdio.h>
#include <unistd.h>
#include<sys/types.h>
int main ()
{
    pid_t pid; /*Pid_t тип данных для ID процесса*/
    printf ("Пока всего один процесс\n");
    pid=fork (); /*Создание нового процесса */
    printf ("Уже два процесса\n");
    if (pid == 0){
        printf ("Это Дочерний процесс его pid = %d\n", getpid());
        printf ("А pid его Родительского процесса=%d\n", getppid());
        // Сюда можно загружать исполняемый файл
        execl("/home/user/test",NULL,NULL);
    }
    else if (pid> 0)
        printf ("Это Родительский процесс pid=%d\n", getpid());
    else
        printf ("Ошибка вызова fork, потомок не создан\n");
}
```

fork() использует механизм COW (copy-on-write)

Дочерний процесс не наследует :

- идентификатора процесса (PID, PPID);
- израсходованного времени ЦП (оно обнуляется);
- сигналов процесса-родителя, требующих ответа;
- блокированных файлов (record locking).