**Тезаурусный словарь**

Поток данных (англ. stream) в программировании — абстракция, используемая для чтения или записи файлов, сокетов и т. п. в единой манере.

Потоки являются удобным унифицированным программным интерфейсом для чтения или записи файлов (в том числе специальных и, в частности, связанных с устройствами), сокетов и передачи данных между процессами поддержка потоков включена в большинство языков программирования и едва ли не во все современные (на 2008 год) операционные системы.

Ядро (англ. kernel) — центральная часть операционной системы (ОС), обеспечивающая приложениям координированный доступ к ресурсам компьютера, таким как процессорное время, память, внешнее аппаратное обеспечение, внешнее устройство ввода и вывода информации. Также обычно ядро предоставляет сервисы файловой системы и сетевых протоколов.

**Ментальная карта**

**Реализация потоков в ядре**

Потоки

Ядра

# Реализация потоков в ядре

Существует несколько способов реализации потоков:

* в пространстве пользователя
* в ядре
* смешанная реализация.

При реализации потоков в ядре, ядро знает о существовании потоков и управляет ими.

В этом случае=

1. система поддержки исполнения программ не нужна.
2. Нет необходимости и в наличии таблицы потоков в каждом процессе, вместо этого есть единая таблица потоков, отслеживающая все потоки системы.
3. Если потоку необходимо создать новый поток или завершить имеющийся, он выполняет запрос ядра, который создает или завершает поток, внося изменения в таблицу потоков.

Таблица потоков, находящаяся в ядре содержит регистры, состояние и другую информацию о каждом потоке.

Информация та же, что и в случае управления потоками на уровне пользователя, только теперь она располагается не в пространстве пользователя (внутри системы поддержки исполнения программ), а в ядре.

Эта информация является подмножеством информации, которую традиционное ядро хранит о каждом из своих однопоточных процессов (то есть подмножеством состояния процесса). Дополнительно ядро содержит обычную таблицу процессов, отслеживающую все процессы системы.

Все запросы, которые могут блокировать поток, реализуются как системные запросы, что требует значительно больших временных затрат, чем вызов процедуры системы поддержки исполнения программ.

Когда поток блокируется, ядро по желанию запускает другой поток из этого же процесса (если есть поток в состоянии готовности) либо поток из другого процесса. При управлении потоками на уровне пользователя система поддержки исполнения программ запускает потоки из одного процесса, пока ядро не передает процессор другому процессу (или пока не кончаются потоки, находящиеся в состоянии готовности).

Поскольку создание и и завершение потоков в ядре требуют относительно больших расходов, некоторые системы используют повторное использование потоков.

После завершения поток помечается как не функционирующий, но в остальном его структура данных, хранящаяся в ядре, не затрагивается. Позже, когда нужно создать новый поток, реактивируется отключенный поток, что помогает сэкономить на некоторых накладных расходах.

При управлении потоками на уровне пользователя повторное использование потоков тоже возможно, но поскольку накладных расходов, связанных с управлением потоками, в этом случае существенно меньше, то и смысла в этом меньше.

Управление потоками в ядре не требует новых не блокирующих системных вызовов. Более того, если один поток вызвал ошибку из-за отсутствия страницы, ядро легко может проверить, есть ли в этом процессе потоки в состоянии готовности, и запустить один из них, пока требуемая страница считывается с диска.