ОПЕРАЦИОННЫЕ СИСТЕМЫ

Лекция 3

Гурген Аракелов

20 октября 2016 г.

Лаборатория Касперского

потоки

потоки

Несколько потоков управления в одном адресном пространстве.

потоки

Нужны ли потоки в одно-ядерных системах?

потоки

Причины использования потоков:

потоки

Причины использования потоков:

1. Несколько действий внутри приложения

потоки

Причины использования потоков:

1. Несколько действий внутри приложения
2. Легкость создания

потоки

Причины использования потоков:

1. Несколько действий внутри приложения
2. Легкость создания
3. Увеличение производительности, даже в одно-процессорных системах

потоки

Причины использования потоков:

1. Несколько действий внутри приложения
2. Легкость создания
3. Увеличение производительности, даже в одно-процессорных системах
4. Многопроцессорные системы

Потоки.Классическая модель

Наличие нескольких потоков — аналогия наличия нескольких процессов.

ПОТОКИ.КЛАССИЧЕСКАЯ МОДЕЛЬ

Наличие нескольких потоков — аналогия наличия нескольких процессов.
Потоки добавляют к модели процесса возможность реализации нескольких независимых задач в единой среде выполнения.

Потоки.Классическая модель

Потоки как «облегченные процессы»

Потоки и много-поточный режим в одно-ядерных системах

Потоки.Классическая модель

Различные потоки одного процесса не обладают той независимостью, которая есть у процессов.

ПОТОКИ

Элементы, присущие каждому процессу	Элементы, присущие каждому потоку
Адресное пространство	Счетчик команд
Глобальные переменные	Регистры
Открытые файлы	Стек
Дочерние процессы	Состояние
Необработанные аварийные сигналы	
Сигналы и обработчики сигналов	
Учетная информация	

ПОТОКИ

thread_exit, thread_join, thread_yield

ПОТОКИ. ПРОБЛЕМЫ

Потоки. Проблемы

Вызов `fork()`.

ПОТОКИ. ПРОБЛЕМЫ

Вызов `fork()`.
Если у родительского процесса есть несколько потоков, должны ли они быть у дочернего?

Потоки. Проблемы

Совместно используемые структуры данных.

ПОТОКИ. ПРОБЛЕМЫ

Переключение потоков.

ПОТОКИ. POSIX. PTHREADS

`Pthreads`—стандарт POSIX для реализации потоков.

ПОТОКИ. POSIX. PTHREADS

`Pthreads`—стандарт POSIX для реализации потоков.
Pthreads поддерживается большинством UNIX-систем.

Основные вызовы стандарта Pthreads:

ПОТОКИ. POSIX. PTHREADS

Основные вызовы стандарта Pthreads:

1. `pthread_create`

ПОТОКИ. POSIX. PTHREADS

Основные вызовы стандарта Pthreads:

1. `pthread_create`
2. `pthread_exit`

ПОТОКИ. POSIX. PTHREADS

Основные вызовы стандарта Pthreads:

1. `pthread_create`
2. `pthread_exit`
3. `pthread_join`

ПОТОКИ. POSIX. PTHREADS

Основные вызовы стандарта Pthreads:

1. `pthread_create`
2. `pthread_exit`
3. `pthread_join`
4. `pthread_yield`

ПОТОКИ. РЕАЛИЗАЦИЯ

Два способа реализации потоков:

ПОТОКИ. РЕАЛИЗАЦИЯ

Два способа реализации потоков:

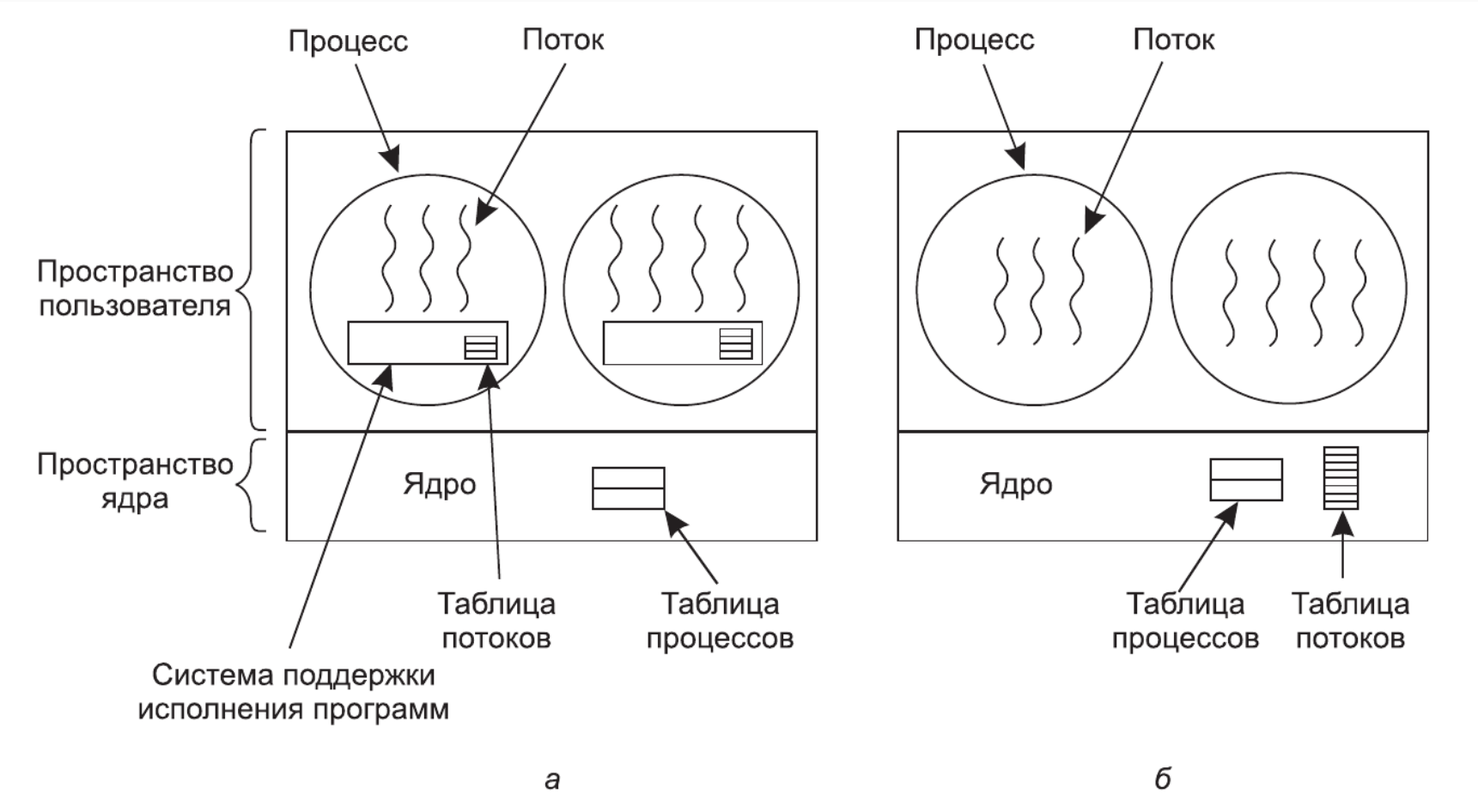
1. В пользовательском пространстве

ПОТОКИ. РЕАЛИЗАЦИЯ

Два способа реализации потоков:

1. В пользовательском пространстве
2. В пространстве ядра.

ПОТОКИ. РЕАЛИЗАЦИЯ



Потоки. Реализация в пользовательском пространстве

Способ, при котором весь набор потоков помещается в пользовательское пространство.

Потоки. Реализация в пользовательском пространстве

Способ, при котором весь набор потоков помещается в пользовательское пространство.
В этом случае ядро ничего не знает про потоки.

Потоки. Реализация в пользовательском пространстве

Способ, при котором весь набор потоков помещается в пользовательское пространство.
В этом случае ядро ничего не знает про потоки.
Поддержка потоков реализуется на уровне библиотеки.

Потоки. Реализация в пользовательском пространстве

Способ, при котором весь набор потоков помещается в пользовательское пространство.
В этом случае ядро ничего не знает про потоки.
Поддержка потоков реализуется на уровне библиотеки.
Каждый процесс при данном подходе должен обладать собственной **таблицей потоков**.

Потоки. Реализация в пользовательском пространстве

Локальный планировщик и эффективность.

Потоки. Реализация в пользовательском пространстве

Преимущество потоков, реализованных в пользовательском пространстве:

Потоки. Реализация в пользовательском пространстве

Преимущество потоков, реализованных в пользовательском пространстве:

1. Эффективность управления

Потоки. Реализация в пользовательском пространстве

Преимущество потоков, реализованных в пользовательском пространстве:

1. Эффективность управления
2. Каждый процесс может иметь свои настройки планирования

Потоки. Реализация в пользовательском пространстве

Преимущество потоков, реализованных в пользовательском пространстве:

1. Эффективность управления
2. Каждый процесс может иметь свои настройки планирования
3. Не требуется дополнительная память в пространстве ядра.

Потоки. Реализация в пользовательском пространстве

Проблемы потоков, реализованных в пользовательском пространстве:

Потоки. Реализация в пользовательском пространстве

Проблемы потоков, реализованных в пользовательском пространстве:

1. Блокирующие системные вызовы

Потоки. Реализация в пользовательском пространстве

Проблемы потоков, реализованных в пользовательском пространстве:

1. Блокирующие системные вызовы
2. Каждый поток должен добровольно уступать время другим потокам внутри процесса, потому, что внутри процесса нет прерываний по таймеру.

Потоки. Реализация в пользовательском пространстве

Проблемы потоков, реализованных в пользовательском пространстве:

1. Блокирующие системные вызовы
2. Каждый поток должен добровольно уступать время другим потокам внутри процесса, потому, что внутри процесса нет прерываний по таймеру.

ПОТОКИ. РЕАЛИЗАЦИЯ В ЯДРЕ

Потоки. Реализация в ядре

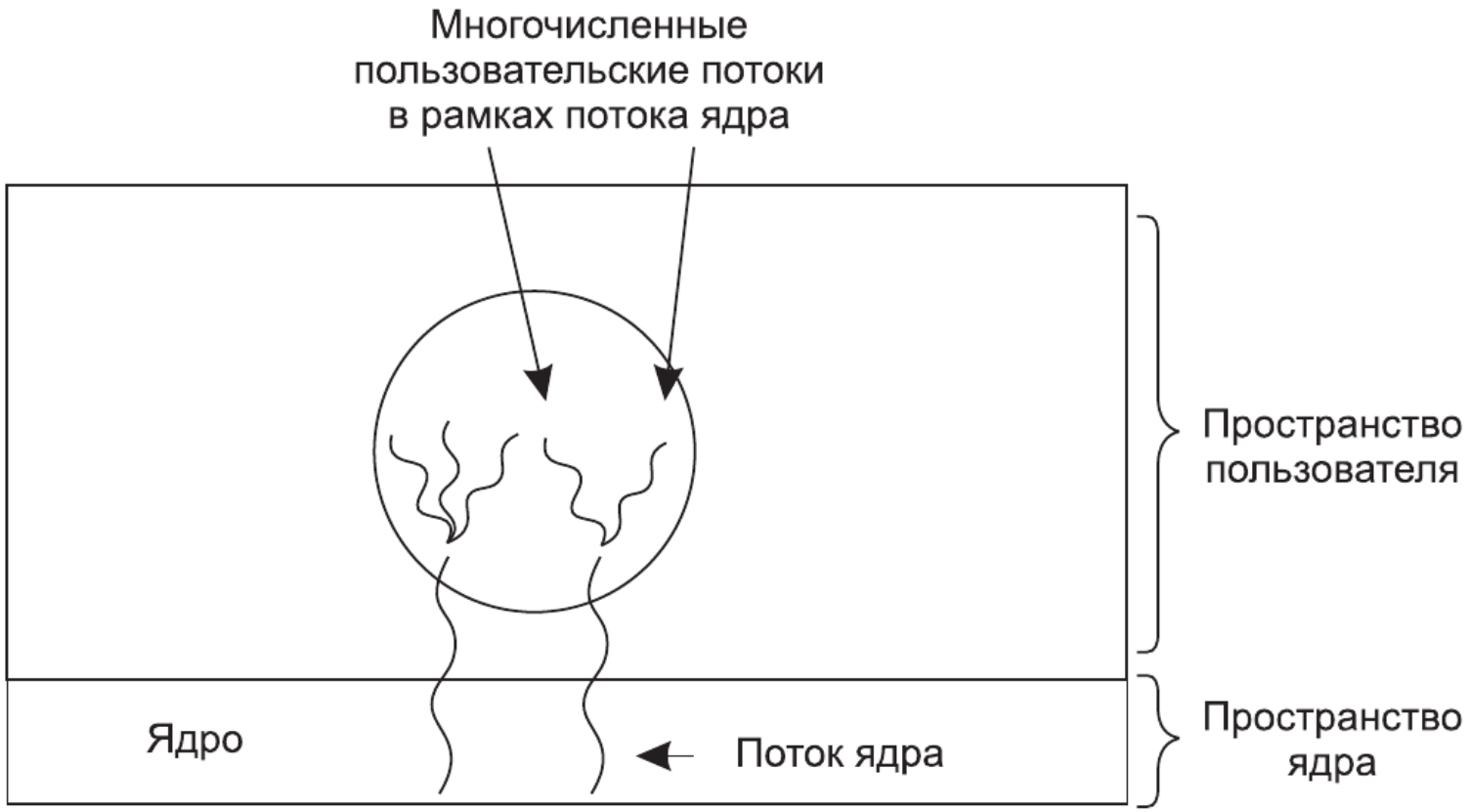
В этом случае в ядре есть таблица потоков, с помощью которой отслеживается информация обо всех потоках в системе.

Потоки. Реализация в ядре

Проблемы:

1. Разветвление много-поточного процесса
2. Сигналы

ПОТОКИ. ГИБРИДНАЯ РЕАЛИЗАЦИЯ



Вопросы?