ОС UNIX, лекция 8: инструменты межпроцессного взаимодействия

Пожидаев М. С.

25 сентября 2020 г.

Мьютексы pthread

Мьютекс — инструмент синхронизации потоков, обеспечивающий вход в защищённый фрагмент кода только одного потока.

- `pthread_mutex_init()` — создание мьютекса;
- `pthread_mutex_destroy()` — уничтожение мьютекса;
- `pthread_mutex_lock()` — блокировка мьютекса;
- `pthread_mutex_unlock()` — освобождение мьютекса.

Условные переменные pthread

Условная переменная — способ приостановить выполнение потока до получения уведомления от другого потока.

- `pthread_cond_init()` — создание условной переменной;
- `pthread_cond_destroy()` — уничтожение условной переменной;
- `pthread_cond_wait()` — начать ожидание уведомления;
- `pthread_cond_signal()`/`pthread_cond_broadcast()` — уведомить потоки, ожидающие уведомления на условной переменной.

Понятие монитора

Монитор — способ безопасного и вычислительноэффективного уведомления о наступлении события в параллельно выполняющихся потоках с возможностью передачи произвольных вспомогательных данных.

Примеры применения мониторов:

1. Передача части работы потоку из пула обработчиков.
2. Выполнение части действий в другом контексте безопасности.
3. Ожидание освобождения ресурса.
4. Ожидание команды пользователя от потока-обработчика устройства ввода.

Составные части монитора

Монитор имеет смысл только в контексте двух параллельно выполняющихся потоков, которые условно назовём поток-поставщик и поток-потребитель. В общем случае потребителей может быть несколько.

Элементы монитора, общие для обоих потоков:

1. Формальный критерий проверки наступления события.
2. Разделяемые данные, передаваемые от потока-поставщика к потоку-потребителю.
3. Инструмент блокировки участков кода, имеющих доступ к разделяемым данным.
4. Возможность остановить выполнение потока-потребителя до получения уведомления от потока-поставщика.

Схема функции-поставщика

1. Блокировка мьютекса.
2. Выполнение действий необходимых для наступления события.
3. Уведомление потока-потребителя о наступившем событии.
4. Освобождение мьютекса.

Схема функции-потребителя

1. Блокировка мьютекса.
2. Проверка наступления события.
3. Ожидание наступления события с временным освобождением мьютекса.
4. Обработка события.
5. Окончательное освобождение мьютекса.

Функция-поставщик для Java

```
void synchronized provide()
{
   if (ready)
     return;
   ready = true;
   notify ();
}
```

Функция-потребитель для Java

```
void synchronized consume()
{
  while(!ready)
    wait();
  ready = false;
}
```

Переменные для pthread

```
pthread_cond_t cond1 = PTHREAD_COND_INITIALIZER;
pthread_mutex_t lock = PTHREAD_MUTEX_INITIALIZER;
int ready = 0;
```

Функция-поставщик для pthread

```
pthread_mutex_lock(&lock);
if (ready == 1)
{
  pthread_mutex_unlock(&lock);
  continue;
}
ready = 1;
printf ("provided\n");
pthread_cond_signal(&cond1);
pthread_mutex_unlock(&lock);
```

Функция-потребитель для pthread

```
pthread_mutex_lock(&lock);
while (ready == 0)
{
   pthread_cond_wait(&cond1, &lock);
    printf ("awoke\n");
}
ready = 0;
 printf ("consumed\n");
pthread_mutex_unlock(&lock);
```

Понятие трубы

Труба (pipe) — пара файловых дескрипторов, один из которых служит для записи данных, второй — для чтения.

1. Создаётся при помощи системного вызова `pipe()`.
2. Переходит в режим ожидания при отсутствии данных или при исчерпании ресурса буфера.

Семафоры

Семафор — обобщение понятия мьютекса на случай, когда возможно вхождение в защищённый фрагмент кода более одного процесса.

Поддерживаются два варианта семафоров:

1. Семафоры System V.
2. Семафоры POSIX.

Семафоры System V

1. Предоставляют возможность работать сразу со множеством семафоров.
2. Создаются при помощи функции `semget()`.
3. Инициализируются при помощи функции `semctl()` (отдельная инициализация является серьёзной проблемой).
4. Операции выполняются при помощи функции `semop()`.

Семафоры POSIX

1. Являются именованными и идентифицируются символьными строками.
2. Создаются при помощи функции `sem_open()`.
3. Операции выполняются при помощи функций `sem_wait()` и `sem_post()`.

Разделяемая память

1. Самый быстрый способ межпроцессного взаимодействия.
2. Реализуется на уровне трансляции страниц памяти в ядре операционной системы.
3. Создаётся парой функций `shmget()` и `shmat()`.

Очереди сообщений и файловые блокировки

1. Очереди сообщений (message queues) позволяют передать некоторый фрагмент данных с указанием типа сообщения из одного процесса в другой.
2. Файловые блокировки (file locks) позволяют синхронизировать работу нескольких процессов с одним и тем же открытым файлом (но не запрещают работу с ним).

Спасибо за внимание!

Веб-сайт: http://marigostra.ru/
E-mail: msp@luwrain.org