Нишки; POSIX функции за работа с нишки

Христо Иванов[1]
hivanov@elsys-bg.org

[1]Технологично училище "Електронни системи"
Технически университет, София

19 февруари 2021 г.

Съдържание

Функция pthread_create()

Описание

```
#include <pthread.h>
int pthread_create(pthread_t *thread,
                   const pthread_attr_t *attr,
                   void *(*start_routine) (void *),
                   void *arg);
```

- Функцията **pthread_create()** стартира нова нишка в извикващият я процес и новата нишка започва да изпълнява функцията **start_routine**.
- Функцията връща 0 при успешно извикване и код за грешка !=0 при грешка. Кодът може да се подаде на strerror(), perror() е неприложима.

Функция pthread_create()

Параметри

- **pthread_t *thread** - указател към променлива, която служи за идентификатор на нишката и държи служебна информация.
- **const pthread_attr_t *attr** - указател към променлива съдържаща атрибути, които искаме да зададем на нишката, може да е NULL.
- **void *(*start_routine) (void *)** - указател към функцията, която нишката ще изпълнява. Връща void* и приема void* за параметър.
- **void *arg** - параметър, който се подава на функцията start_routine, може да е NULL.

Функция pthread_exit()

Описание

```
#include <pthread.h>
void pthread_exit(void *retval);
```

- Функцията **pthread_exit()** прекратява извикващата я нишка и връща стойност чрез параметъра **retval**.
- Функцията **pthread_exit()** не затваря файловете, които са отворени в рамките на нишката.
- Ако първата нишка на процеса, изпълняваща main(), извика **pthread_exit()**, то тя ще излезе, но останалите нишки ще продължат да работят.

Функция pthread_cancel()

Описание

```
#include <pthread.h>
int pthread_cancel(pthread_t thread);
```

- Функцията **pthread_cancel()** изпраща заявка за спиране към нишката подадена чрез параметъра **thread**.
- Това дали нишката ще спре зависи от някои от атрибутите ѝ - *cancelability state* и *cancelability type*.
- *cancelability state* указва дали нишката може да бъде спряна от **pthread_cancel()**, а *cancelability type* указва кога ще стане това - веднага или отложено във времето.
- За повече инфо - man pthread_cancel.

Функция pthread_join()

Описание

```
#include <pthread.h>
int pthread_join(pthread_t thread, void **retval);
```

- Функцията **pthread_join()** блокира извикващата я нишка, докато нишката подадена с параметъра *thread* не завърши. Ако вече е завършила, то **pthread_join()** връща резултат веднага.
- Функцията връща 0 при успешно извикване и код за грешка !=0 при грешка. Кодът може да се подаде на strerror(), perror() е неприложима.

Функция pthread_join()

Параметри

- **pthread_t thread** - идентификатор на нишката, която ще бъде изчакана.
- **void** retval** - указател предаден по адрес, който сочи към стойността върната от нишката чрез pthread_exit(). Може да е NULL.

Mutex-и

Описание

- Mutex е съкращение от "MUTual EXclusion"(взаимно изключване) и е средство за реализация на критична секция за нишки.
- Mutex-ите играят ролята на "ключалка"за споделени ресурси, върху които работят няколко нишки.
- Когато някоя нишка иска да използва ресурс защитен от mutex то тя се опитва да го "вземе"(заключи) и когато успее то само тя може да работи с ресурса.
- За да могат други нишки да работят с mutex-а, то тази която го е заключила трябва да го "пусне"(отключи).
- Mutex-ите могат да се използват за предотвратяване на условия за надпревара (race condition).

Mutex-и

Thread 1	Thread 2	Balance
Read balance: $1000		$1000
	Read balance: $1000	$1000
	Deposit $200	$1000
Deposit $200		$1000
Update balance $1000+$200		$1200
	Update balance $1000+$200	$1200

Фигура: Race condition

Mutex-и

Особености

- За да бъдат защитени споделените ресурси то трябва всички нишки работещи върху тях да използват едни и същи mutex-и свързани с тези ресурси
- При опит за заключване на вече заключен mutex извикващата нишка ще бъде блокирана. Ако това е нежелателно, то съществува и вариант за опит за заключване (trylock).
- Нишките се **"състезават"** помежду си при опитите си за заключване на mutex.
- При използване на повече от 1 mutex може да настъпи т.нар. **deadlock(мъртва хватка)** - това е когато една нишка иска да заключи mutex заключен от друга нишка, докато втората нишка иска да заключи mutex заключен от първата.

Работа с mutex-и

Описание

- В стандарта POSIX mutex са реализирани чрез типа **pthread_mutex_t**.
- За да се създаде mutex първо трябва да се създаде променлива от този тип, след което тя да се инициализира чрез функцията **pthread_mutex_init()**.
- Когато mutex-ът вече не е необходим той се унищожава с функцията **pthread_mutex_destroy()**.
- За заключване на mutex се използват функциите **pthread_mutex_lock()** и **pthread_mutex_trylock()**.
- За отключване на mutex се използва функцията **pthread_mutex_unlock()**.

Функция pthread_mutex_init()

Описание

```
#include <pthread.h>
int pthread_mutex_init(pthread_mutex_t *mutex,
                       const pthread_mutexattr_t *mutexattr);
```

- Функцията **pthread_mutex_init()** инициализира mutex подаден чрез параметъра *mutex* и му задава атрибутите подадени чрез параметъра *mutexattr*. Ако *mutexattr* е NULL за атрибутите се използват стойности по подразбиране.
- Функцията **pthread_mutex_init()** връща винаги 0.

Функция pthread_mutex_destroy()

Описание

```
#include <pthread.h>
int pthread_mutex_destroy(pthread_mutex_t *mutex);
```

- Функцията **pthread_mutex_destroy()** унищожава mutex-а подаден чрез параметъра *mutex* и освобождава ресурсите заети от него. Преди извикването на функцията той трябва да е отключен.
- Функцията **pthread_mutex_destroy()** връща 0 при успех и код за грешка в противен случай.

Функция pthread_mutex_lock()

Описание

```
#include <pthread.h>
int pthread_mutex_lock(pthread_mutex_t *mutex);
```

- Функцията **pthread_mutex_lock()** заключва mutex-а подаден чрез параметъра *mutex*. Ако той е отключен функцията приключва веднага, в противен случай тя блокира извикващата я нишка, докато mutex-ът не бъде отключен.
- Ако mutex-ът е вече заключен от извикващата нишка и е с атрибути по подразбиране, то ще се получи deadlock.
- Функцията **pthread_mutex_lock()** връща 0 при успех и код за грешка в противен случай.

Функция pthread_mutex_unlock()

Описание

```
#include <pthread.h>
int pthread_mutex_unlock(pthread_mutex_t *mutex);
```

- Функцията **pthread_mutex_unlock()** отключва mutex-а подаден чрез параметъра *mutex*.
- Функцията **pthread_mutex_unlock()** връща 0 при успех и код за грешка в противен случай.

Функция pthread_mutex_trylock()

Описание

```
#include <pthread.h>
int pthread_mutex_trylock(pthread_mutex_t *mutex);
```

- Функцията **pthread_mutex_trylock()** се опитва да заключи mutex-а подаден чрез параметъра *mutex*, без да чака да бъде отключен, ако е.
- Ако mutex-ът е вече заключен то функцията връща код за грешка EBUSY.
- Функцията **pthread_mutex_trylock()** връща 0 при успех и код за грешка в противен случай.