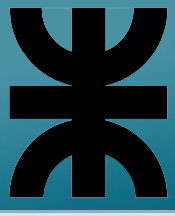
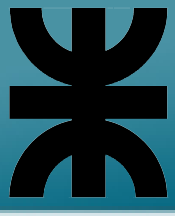


MUTEX



Mutex

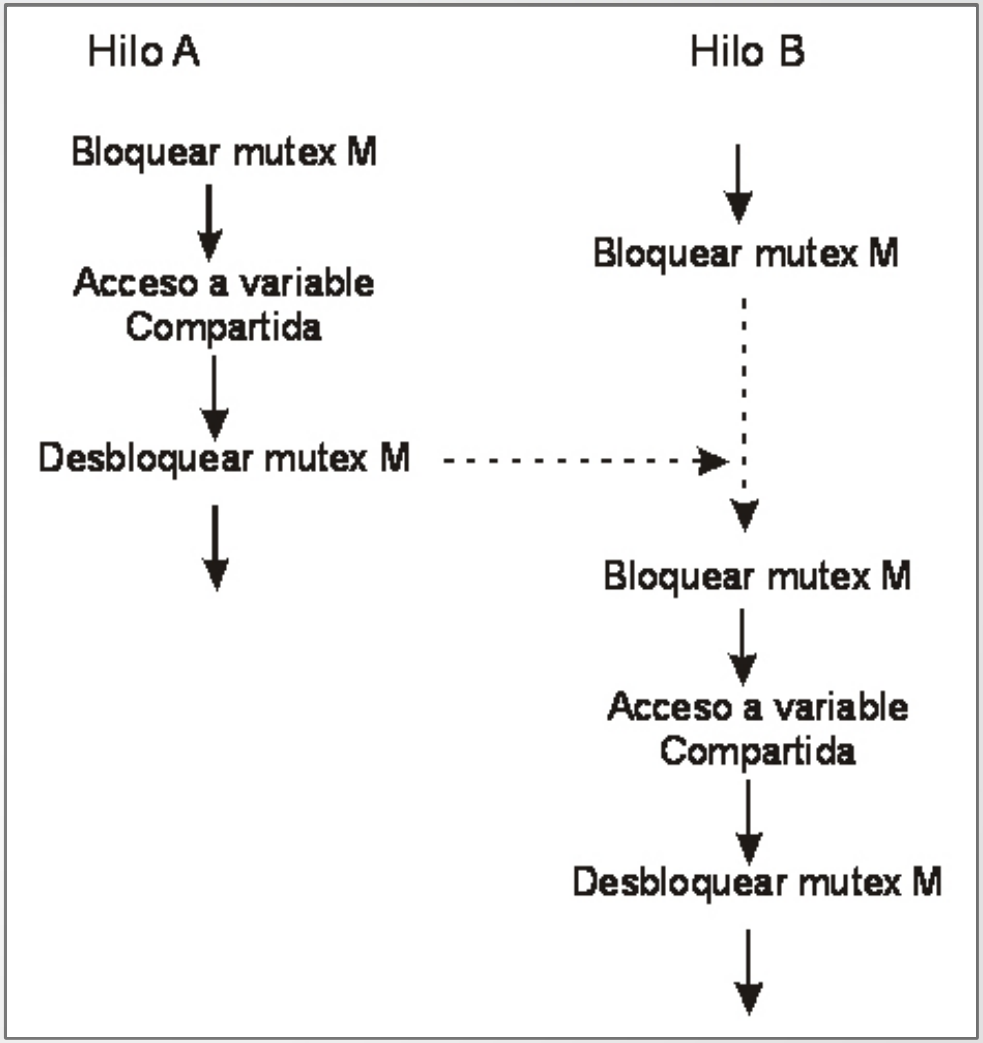
MUTEX

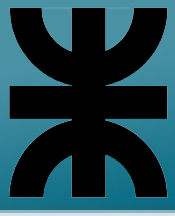
Cuando se tiene acceso a un recurso compartido (sección crítica) la ejecución debe ser atómica, es decir, la ejecución del acceso no debe ser interrumpido por otro hilo que desea acceder simultáneamente al mismo recurso compartido.

- Para sincronizar el uso de variables compartidas están los mutex (abreviatura de la exclusión mutua).
- Un mutex sirve para asegurarse de que sólo un hilo a la vez puede acceder a la variable compartida.
- Un mutex tiene dos estados: bloqueado y desbloqueado.
- Solo un hilo a la vez puede bloquear un mutex.
- Cuando un hilo bloquea un mutex, se convierte en el propietario de dicho mutex.
- Sólo el hilo propietario de un mutex puede desbloquearlo.

Mutex

Si varios hilos intentan ejecutar el bloqueo de un mutex, debido a que un solo hilo puede mantener el bloqueo, los otros hilos permanecen a su vez bloqueados a la espera de la liberación del mutex



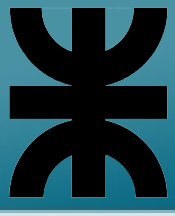


Mutex

Pasos para utilizar el mutex:

* Bloquear el mutex asignado al recurso compartido
* Acceder al recurso compartido
* Desbloquear el mutex

El bloqueo del mutex es de carácter **consultivo**, en vez de obligatorio. Es decir que un hilo es libre de ignorar el uso de un mutex. Con el fin de manejar con seguridad las variables compartidas, todas las hilos deben **cooperar** en el uso de un mutex, respetando las reglas de bloqueo.



Mutex

Inicialización de Mutex estático

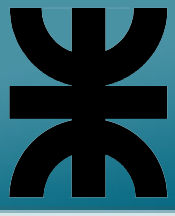
Un mutex es una variable del tipo pthread_mutex_t

Para la asignación estática del mutex hacemos:

pthread_mutex_t mtx = PTHREAD_MUTEX_INITIALIZER;

Antes de bloquear o desbloquear un mutex debemos inicializarlo.

Al inicializar un mutex en forma estática este solo puede tener los atributos por defecto.



Mutex

Inicialización mutex dinámico

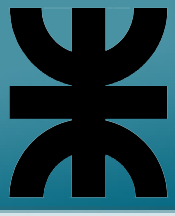
#include <pthread.h>

int pthread_mutex_init(pthread_mutex_t *mutex, const pthread_mutex attr_t *attr);

Devuelve 0 si tiene éxito o un error positivo en caso de error

El argumento de mutex identifica el mutex para ser inicializado.

Si attr es NULL se tomaran los atributos por defecto



Mutex

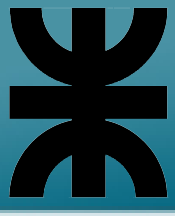
Bloqueo de un mutex

#include <pthread.h>

int pthread_mutex_lock(pthread_mutex_t *mutex);

Devuelve 0 si tuvieron éxito y un número positivo en caso de error

La función pthread_mutex_lock() bloquea un mutex. Si el mutex está desbloqueado, esta llamada bloquea el mutex y vuelve inmediatamente. Si el mutex está bloqueado por otro hilo, la función pthread_mutex_lock() bloquea el hilo hasta que el mutex está desbloqueado, en cuyo momento se bloquea el mutex y la función retorna.



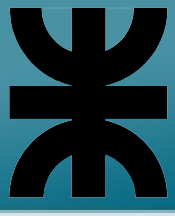
Mutex

Función pthread_mutex_trylock()

La función pthread_mutex_trylock() es la misma que pthread_mutex_lock(), excepto que si el mutex está bloqueado, pthread_mutex_trylock () falla, y vuelve con el error EBUSY.

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

Devuelve 0 si tiene éxito, o el error EBUSY indicando que otro hilo tiene el mutex.



Mutex

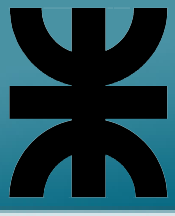
Desbloqueo de un mutex

#include <pthread.h>

int pthread_mutex_unlock(pthread_mutex_t *mutex);

Devuelve 0 si tiene éxito, un número positivo en caso de error.

La función pthread_mutex_unlock () desbloquea un mutex previamente bloqueado por el hilo que la llama .
Es incorrecto desbloquear un mutex que no está bloqueado, o desbloquear un mutex que está bloqueado por otro hilo.



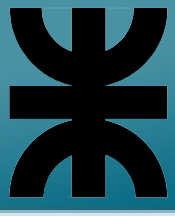
Mutex

Destrucción de un mutex

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

Devuelve 0 si tiene éxito, un número positivo en caso de error

Es seguro destruir un mutex sólo cuando esta desbloqueado, y ningún un hilo tratará subsecuentemente de bloquearlo.
Si el mutex se encuentra en una región de memoria asignada dinámicamente, entonces debe ser destruido antes de la liberación de esa región de la memoria.
Una mutex que ha sido destruido con pthread_mutex_destroy() posteriormente se puede reinicializar con pthread_mutex_init().
No es necesario ejecutar pthread_mutex_destroy() si el mutex se ha inicializado estáticamente (PTHREAD_MUTEX_INITIALIZER)



Mutex

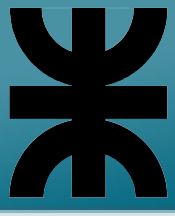
Interbloqueos de Mutex

A veces, un hilo necesita acceder simultáneamente a dos o más recursos compartidos, cada uno de ellos se rige por un mutex por separado.
El bloqueo y desbloqueo de los mutex puede producir interbloqueos

Hilo A
1. pthread_mutex_lock(mutex1);
2. pthread_mutex_lock(mutex2);
Blocks

Hilo B
1. pthread_mutex_lock(mutex2);
2. pthread_mutex_lock(mutex1);
Blocks

Para evitar esto todos los hilos deben bloquear y desbloquear los mutex en el mismo orden. A esto se llama jerarquía de mutex.



Mutex

Atributos Mutex

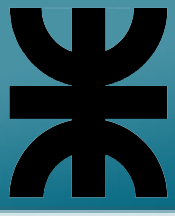
El argumento mtxattr es un puntero a un objeto pthread_mutexattr_t que previamente se debe inicializar para definir los atributos del mutex

```
#include <pthread.h>
pthread_mutexattr_t mtxattr;
```

Inicialización de atributos de un mutex

```
#include <pthread.h>
int pthread_mutexattr_init(pthread_mutexattr_t *attr);
```

Devuelve 0 si tiene éxito, un número positivo en caso de error



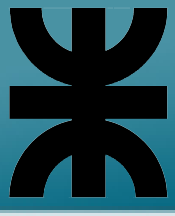
Mutex

Tipos de mutex

PTHREAD_MUTEX_ERRORCHECK: Realiza comprobación de errores en todas las operaciones. Este tipo de mutex es más lento que un mutex normal, pero puede ser útil como una herramienta de depuración.

PTHREAD_MUTEX_RECURSIVE: Cuando un hilo adquiere por primera vez el mutex , el contador de bloqueo se establece en 1. Cada operación posterior de bloqueo hecha por el mismo hilo incrementa el contador de bloqueo, y cada operación de desbloqueo decrementa el contador. El mutex se libera sólo cuando el contador de bloqueos cae a 0.
En mutex estaticos;THREAD_RECURSIVE_MUTEX_INITIALIZER_NP

PTHREAD_MUTEX_DEFAULT: Es el tipo predeterminado de mutex, si usamos PTHREAD_MUTEX_INITIALIZER o especificamos attr como NULL en una llamada a pthread_mutex_init().
En Linux, un mutex PTHREAD_MUTEX_DEFAULT se comporta como un mutex PTHREAD_MUTEX_NORMAL.



Mutex

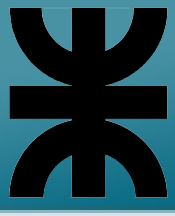
Seteo de tipo en atributos

#include <pthread.h>

int pthread_mutexattr_settype(pthread_mutexattr_t *attr, int type);

Devuelve 0 si tuvo éxito o un error positivo en caso de error

Type:
THREAD_MUTEX_ERRORCHECK
PTHREAD_MUTEX_RECURSIVE
PTHREAD_MUTEX_DEFAULT
PTHREAD_MUTEX_NORMAL



Mutex

Costo de usar un mutex:

Los programas que utilizan mutex consumen más ciclos de reloj, por tanto, insume más tiempo su ejecución.

Errores:

Bloquear un mutex que ya está bloqueado hace que el hilo se bloquee (o falle con un error según la función de bloqueo utilizada), salvo en el caso de mutex recursivos.

Un solo hilo no puede bloquear el mismo mutex dos veces.

Un hilo no puede desbloquear un mutex que no está actualmente bloqueado.