NAME

pthread - POSIX thread functions

LIBRARY

POSIX Threads Library (libpthread, -lpthread)

SYNOPSIS

#include <pthread.h>

DESCRIPTION

POSIX threads are a set of functions that support applications with requirements for multiple flows of control, called *threads,* within a process. Multithreading is used to improve the performance of a program.

The POSIX thread functions are summarized in this section in the following groups:

- Thread Routines
- Attribute Object Routines
- Mutex Routines
- Condition Variable Routines
- Read/Write Lock Routines
- Per-Thread Context Routines
- Cleanup Routines

FreeBSD extensions to the POSIX thread functions are summarized in pthread_np(3).

Thread Routines

int **pthread_create**(*pthread_t *thread, const pthread_attr_t *attr, void *(*start_routine)(void *), void *arg*)

Creates a new thread of execution.

int **pthread_cancel**(*pthread_t thread*)

Cancels execution of a thread.

int **pthread_detach**(*pthread_t thread*)

Marks a thread for deletion.

int **pthread_equal**(*pthread_t t1*, *pthread_t t2*)

Compares two thread IDs.

void **pthread_exit**(*void *value_ptr*)
Terminates the calling thread.

int **pthread_join**(*pthread_t thread*, *void **value_ptr*)
Causes the calling thread to wait for the termination of the specified thread.

int **pthread_kill**(*pthread_t thread*, *int sig*)
Delivers a signal to a specified thread.

int **pthread_once**(*pthread_once_t *once_control*, *void (*init_routine)(void)*)
Calls an initialization routine once.

pthread_t **pthread_self**(*void*)
Returns the thread ID of the calling thread.

int **pthread_setcancelstate**(*int state*, *int *oldstate*)
Sets the current thread's cancelability state.

int **pthread_setcanceltype**(*int type*, *int *oldtype*)
Sets the current thread's cancelability type.

void **pthread_testcancel**(*void*)
Creates a cancellation point in the calling thread.

void **pthread_yield**(*void*)
Allows the scheduler to run another thread instead of the current one.

Attribute Object Routines

int **pthread_attr_destroy**(*pthread_attr_t *attr*)
Destroy a thread attributes object.

int **pthread_attr_getinheritsched**(*const pthread_attr_t *attr*, *int *inheritsched*)
Get the inherit scheduling attribute from a thread attributes object.

int **pthread_attr_getschedparam**(*const pthread_attr_t *attr*, *struct sched_param *param*)
Get the scheduling parameter attribute from a thread attributes object.

int **pthread_attr_getschedpolicy**(*const pthread_attr_t *attr*, *int *policy*)
Get the scheduling policy attribute from a thread attributes object.

int **pthread_attr_getscope**(*const pthread_attr_t *attr*, *int *contentionscope*)
Get the contention scope attribute from a thread attributes object.

int **pthread_attr_getstacksize**(*const pthread_attr_t *attr*, *size_t *stacksize*)
Get the stack size attribute from a thread attributes object.

int **pthread_attr_getstackaddr**(*const pthread_attr_t *attr*, *void **stackaddr*)
Get the stack address attribute from a thread attributes object.

int **pthread_attr_getdetachstate**(*const pthread_attr_t *attr*, *int *detachstate*)
Get the detach state attribute from a thread attributes object.

int **pthread_attr_init**(*pthread_attr_t *attr*)
Initialize a thread attributes object with default values.

int **pthread_attr_setinheritsched**(*pthread_attr_t *attr*, *int inheritsched*)
Set the inherit scheduling attribute in a thread attributes object.

int **pthread_attr_setschedparam**(*pthread_attr_t *attr*, *const struct sched_param *param*)
Set the scheduling parameter attribute in a thread attributes object.

int **pthread_attr_setschedpolicy**(*pthread_attr_t *attr*, *int policy*)
Set the scheduling policy attribute in a thread attributes object.

int **pthread_attr_setscope**(*pthread_attr_t *attr*, *int contentionscope*)
Set the contention scope attribute in a thread attributes object.

int **pthread_attr_setstacksize**(*pthread_attr_t *attr*, *size_t stacksize*)
Set the stack size attribute in a thread attributes object.

int **pthread_attr_setstackaddr**(*pthread_attr_t *attr*, *void *stackaddr*)
Set the stack address attribute in a thread attributes object.

int **pthread_attr_setdetachstate**(*pthread_attr_t *attr*, *int detachstate*)
Set the detach state in a thread attributes object.

Mutex Routines

int **pthread_mutexattr_destroy**(*pthread_mutexattr_t *attr*)
Destroy a mutex attributes object.

int **pthread_mutexattr_getprioceiling**(*const pthread_mutexattr_t *restrict attr*, *int *restrict ceiling*)
Obtain priority ceiling attribute of mutex attribute object.

int **pthread_mutexattr_getprotocol**(*const pthread_mutexattr_t *restrict attr*, *int *restrict protocol*)
Obtain protocol attribute of mutex attribute object.

int **pthread_mutexattr_gettype**(*const pthread_mutexattr_t *restrict attr*, *int *restrict type*)
Obtain the mutex type attribute in the specified mutex attributes object.

int **pthread_mutexattr_init**(*pthread_mutexattr_t *attr*)
Initialize a mutex attributes object with default values.

int **pthread_mutexattr_setprioceiling**(*pthread_mutexattr_t *attr*, *int ceiling*)
Set priority ceiling attribute of mutex attribute object.

int **pthread_mutexattr_setprotocol**(*pthread_mutexattr_t *attr*, *int protocol*)
Set protocol attribute of mutex attribute object.

int **pthread_mutexattr_settype**(*pthread_mutexattr_t *attr*, *int type*)
Set the mutex type attribute that is used when a mutex is created.

int **pthread_mutex_destroy**(*pthread_mutex_t *mutex*)
Destroy a mutex.

int **pthread_mutex_init**(*pthread_mutex_t *mutex*, *const pthread_mutexattr_t *attr*)
Initialize a mutex with specified attributes.

int **pthread_mutex_lock**(*pthread_mutex_t *mutex*)
Lock a mutex and block until it becomes available.

int **pthread_mutex_timedlock**(*pthread_mutex_t *mutex*, *const struct timespec *abstime*)
Lock a mutex and block until it becomes available or until the timeout expires.

int **pthread_mutex_trylock**(*pthread_mutex_t *mutex*)
Try to lock a mutex, but do not block if the mutex is locked by another thread, including the current thread.

int **pthread_mutex_unlock**(*pthread_mutex_t *mutex*)
Unlock a mutex.

Condition Variable Routines

int **pthread_condattr_destroy**(*pthread_condattr_t *attr*)
Destroy a condition variable attributes object.

int **pthread_condattr_init**(*pthread_condattr_t *attr*)
Initialize a condition variable attributes object with default values.

int **pthread_cond_broadcast**(*pthread_cond_t *cond*)
Unblock all threads currently blocked on the specified condition variable.

int **pthread_cond_destroy**(*pthread_cond_t *cond*)
Destroy a condition variable.

int **pthread_cond_init**(*pthread_cond_t *cond*, *const pthread_condattr_t *attr*)
Initialize a condition variable with specified attributes.

int **pthread_cond_signal**(*pthread_cond_t *cond*)
Unblock at least one of the threads blocked on the specified condition variable.

int **pthread_cond_timedwait**(*pthread_cond_t *cond*, *pthread_mutex_t *mutex*, *const struct timespec *abstime*)
Unlock the specified mutex, wait no longer than the specified time for a condition, and then relock the mutex.

int **pthread_cond_wait**(*pthread_cond_t **, *pthread_mutex_t *mutex*)
Unlock the specified mutex, wait for a condition, and relock the mutex.

Read/Write Lock Routines

int **pthread_rwlock_destroy**(*pthread_rwlock_t *lock*)
Destroy a read/write lock object.

int **pthread_rwlock_init**(*pthread_rwlock_t *lock*, *const pthread_rwlockattr_t *attr*)
Initialize a read/write lock object.

int **pthread_rwlock_rdlock**(*pthread_rwlock_t *lock*)
Lock a read/write lock for reading, blocking until the lock can be acquired.

int **pthread_rwlock_tryrdlock**(*pthread_rwlock_t *lock*)
Attempt to lock a read/write lock for reading, without blocking if the lock is unavailable.

int **pthread_rwlock_trywrlock**(*pthread_rwlock_t *lock*)
Attempt to lock a read/write lock for writing, without blocking if the lock is unavailable.

int **pthread_rwlock_unlock**(*pthread_rwlock_t *lock*)
Unlock a read/write lock.

int **pthread_rwlock_wrlock**(*pthread_rwlock_t *lock*)
Lock a read/write lock for writing, blocking until the lock can be acquired.

int **pthread_rwlockattr_destroy**(*pthread_rwlockattr_t *attr*)
Destroy a read/write lock attribute object.

int **pthread_rwlockattr_getpshared**(*const pthread_rwlockattr_t *attr*, *int *pshared*)
Retrieve the process shared setting for the read/write lock attribute object.

int **pthread_rwlockattr_init**(*pthread_rwlockattr_t *attr*)
Initialize a read/write lock attribute object.

int **pthread_rwlockattr_setpshared**(*pthread_rwlockattr_t *attr*, *int pshared*)
Set the process shared setting for the read/write lock attribute object.

Per-Thread Context Routines

int **pthread_key_create**(*pthread_key_t *key*, *void (*routine)(void *)*)
Create a thread-specific data key.

int **pthread_key_delete**(*pthread_key_t key*)
Delete a thread-specific data key.

*void ** **pthread_getspecific**(*pthread_key_t key*)
Get the thread-specific value for the specified key.

int **pthread_setspecific**(*pthread_key_t key*, *const void *value_ptr*)
Set the thread-specific value for the specified key.

Cleanup Routines

int **pthread_atfork**(*void (*prepare)(void)*, *void (*parent)(void)*, *void (*child)(void)*)
Register fork handlers.

void **pthread_cleanup_pop**(*int execute*)
Remove the routine at the top of the calling thread's cancellation cleanup stack and optionally

invoke it.

void **pthread_cleanup_push**(*void (*routine)(void *), void *routine_arg*)

Push the specified cancellation cleanup handler onto the calling thread's cancellation stack.

IMPLEMENTATION NOTES

The current FreeBSD POSIX thread implementation is built into the 1:1 Threading Library (libthr, -lthr) library. It contains thread-safe versions of Standard C Library (libc, -lc) functions and the thread functions. Threaded applications are linked with this library.

SEE ALSO

libthr(3), pthread_atfork(3), pthread_attr(3), pthread_cancel(3), pthread_cleanup_pop(3), pthread_cleanup_push(3), pthread_cond_broadcast(3), pthread_cond_destroy(3), pthread_cond_init(3), pthread_cond_signal(3), pthread_cond_timedwait(3), pthread_cond_wait(3), pthread_condattr_destroy(3), pthread_condattr_init(3), pthread_create(3), pthread_detach(3), pthread_equal(3), pthread_exit(3), pthread_getspecific(3), pthread_join(3), pthread_key_delete(3), pthread_kill(3), pthread_mutex_destroy(3), pthread_mutex_init(3), pthread_mutex_lock(3), pthread_mutex_trylock(3), pthread_mutex_unlock(3), pthread_mutexattr_destroy(3), pthread_mutexattr_getprioceiling(3), pthread_mutexattr_getprotocol(3), pthread_mutexattr_gettype(3), pthread_mutexattr_init(3), pthread_mutexattr_setprioceiling(3), pthread_mutexattr_setprotocol(3), pthread_mutexattr_settype(3), pthread_np(3), pthread_once(3), pthread_rwlock_destroy(3), pthread_rwlock_init(3), pthread_rwlock_rdlock(3), pthread_rwlock_unlock(3), pthread_rwlock_wrlock(3), pthread_rwlockattr_destroy(3), pthread_rwlockattr_getpshared(3), pthread_rwlockattr_init(3), pthread_rwlockattr_setpshared(3), pthread_self(3), pthread_setcancelstate(3), pthread_setcanceltype(3), pthread_setspecific(3), pthread_testcancel(3)

STANDARDS

The functions with the **pthread_** prefix and not **_np** suffix or **pthread_rwlock** prefix conform to ISO/IEC 9945-1:1996 ("POSIX.1").

The functions with the **pthread_** prefix and **_np** suffix are non-portable extensions to POSIX threads.

The functions with the **pthread_rwlock** prefix are extensions created by The Open Group as part of the Version 2 of the Single UNIX Specification ("SUSv2").