Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'pthread_rwlock_rdlock.3p' command

$ man pthread_rwlock_rdlock.3p

PTHREAD_RWLOCK_RDLOCK(3P) POSIX Programmer's Manual PTHREAD_RWLOCK_RDLOCK(3P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

pthread_rwlock_rdlock, pthread_rwlock_tryrdlock ? lock a read-write lock object for reading

SYNOPSIS

```
#include <pthread.h>
int pthread_rwlock_rdlock(pthread_rwlock_t *rwlock);
int pthread_rwlock_tryrdlock(pthread_rwlock_t *rwlock);
```

DESCRIPTION

The pthread_rwlock_rdlock() function shall apply a read lock to the read-write lock referenced by rwlock. The calling thread acquires the read lock if a writer does not hold the lock and there are no writers blocked on the lock.

If the Thread Execution Scheduling option is supported, and the threads involved in the lock are executing with the scheduling policies SCHED_FIFO or SCHED_RR, the calling thread shall not acquire the lock if a writer holds the lock or if writers of higher or equal priority are blocked on the lock; otherwise, the calling thread shall acquire

the lock.

If the Thread Execution Scheduling option is supported, and the threads involved in the lock are executing with the SCHED_SPORADIC scheduling policy, the calling thread shall not acquire the lock if a writer holds the lock or if writers of higher or equal priority are blocked on the lock; otherwise, the calling thread shall acquire the lock.

If the Thread Execution Scheduling option is not supported, it is im? plementation-defined whether the calling thread acquires the lock when a writer does not hold the lock and there are writers blocked on the lock. If a writer holds the lock, the calling thread shall not acquire the read lock. If the read lock is not acquired, the calling thread shall block until it can acquire the lock. The calling thread may dead? lock if at the time the call is made it holds a write lock.

A thread may hold multiple concurrent read locks on rwlock (that is, successfully call the pthread_rwlock_rdlock() function n times). If so, the application shall ensure that the thread performs matching unlocks (that is, it calls the pthread_rwlock_unlock() function n times).

The maximum number of simultaneous read locks that an implementation guarantees can be applied to a read-write lock shall be implementation-defined. The pthread_rwlock_rdlock() function may fail if this maximum would be exceeded.

The pthread_rwlock_tryrdlock() function shall apply a read lock as in the pthread_rwlock_rdlock() function, with the exception that the func? tion shall fail if the equivalent pthread_rwlock_rdlock() call would have blocked the calling thread. In no case shall the pthread_rwlock_tryrdlock() function ever block; it always either ac? quires the lock or fails and returns immediately.

Results are undefined if any of these functions are called with an uninitialized read-write lock.

If a signal is delivered to a thread waiting for a read-write lock for reading, upon return from the signal handler the thread resumes waiting for the read-write lock for reading as if it was not interrupted.

RETURN VALUE

If successful, the pthread_rwlock_rdlock() function shall return zero; otherwise, an error number shall be returned to indicate the error.

The pthread_rwlock_tryrdlock() function shall return zero if the lock for reading on the read-write lock object referenced by rwlock is acquired. Otherwise, an error number shall be returned to indicate the error.

ERRORS

The pthread_rwlock_tryrdlock() function shall fail if:

EBUSY The read-write lock could not be acquired for reading because a writer holds the lock or a writer with the appropriate priority was blocked on it.

The pthread_rwlock_rdlock() and pthread_rwlock_tryrdlock() functions may fail if:

EAGAIN The read lock could not be acquired because the maximum number of read locks for rwlock has been exceeded.

The pthread_rwlock_rdlock() function may fail if:

EDEADLK

A deadlock condition was detected or the current thread already owns the read-write lock for writing.

These functions shall not return an error code of [EINTR].

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

Applications using these functions may be subject to priority inversion, as discussed in the Base Definitions volume of POSIX.1?2017, Section 3.291, Priority Inversion.

RATIONALE

If an implementation detects that the value specified by the rwlock argument to pthread_rwlock_rdlock() or pthread_rwlock_tryrdlock() does not refer to an initialized read-write lock object, it is recommended that the function should fail and report an [EINVAL] error.

FUTURE DIRECTIONS

None.

SEE ALSO

pthread_rwlock_destroy(), pthread_rwlock_timedrdlock(), pthread_rwlock_timedwrlock(), pthread_rwlock_trywrlock(), pthread_rwlock_unlock()

The Base Definitions volume of POSIX.1?2017, Section 3.291, Priority Inversion, Section 4.12, Memory Synchronization, <pthread.h>

COPYRIGHT