NAME

pthread.h − threads

SYNOPSIS

#include <pthread.h>

DESCRIPTION

The *<pthread.h>* header shall define the following symbols:

PTHREAD_BARRIER_SERIAL_THREAD

PTHREAD_CANCEL_ASYNCHRONOUS
PTHREAD_CANCEL_ENABLE
PTHREAD_CANCEL_DEFERRED
PTHREAD_CANCEL_DISABLE
PTHREAD_CANCELED
PTHREAD_COND_INITIALIZER
PTHREAD_CREATE_DETACHED
PTHREAD_CREATE_JOINABLE
PTHREAD_EXPLICIT_SCHED
PTHREAD_INHERIT_SCHED

PTHREAD_MUTEX_DEFAULT
PTHREAD_MUTEX_ERRORCHECK

PTHREAD_MUTEX_INITIALIZER

PTHREAD_MUTEX_NORMAL
PTHREAD_MUTEX_RECURSIVE

PTHREAD_ONCE_INIT

PTHREAD_PRIO_INHERIT

PTHREAD_PRIO_NONE

PTHREAD_PRIO_PROTECT

PTHREAD_PROCESS_SHARED
PTHREAD_PROCESS_PRIVATE

PTHREAD_SCOPE_PROCESS
PTHREAD_SCOPE_SYSTEM

The following types shall be defined as described in *<sys/types.h>* :

pthread_attr_t

pthread_barrier_t
pthread_barrierattr_t

pthread_cond_t
pthread_condattr_t
pthread_key_t

pthread_mutex_t
pthread_mutexattr_t
pthread_once_t
pthread_rwlock_t
pthread_rwlockattr_t

pthread_spinlock_t

pthread_t

The following shall be declared as functions and may also be defined as macros. Function prototypes shall be provided.

```
int   pthread_atfork(void (*)(void), void (*)(void),
          void(*)(void));
int   pthread_attr_destroy(pthread_attr_t *);
int   pthread_attr_getdetachstate(const pthread_attr_t *, int *);

int   pthread_attr_getguardsize(const pthread_attr_t *restrict,
          size_t *restrict);


int   pthread_attr_getinheritsched(const pthread_attr_t *restrict,
          int *restrict);

int   pthread_attr_getschedparam(const pthread_attr_t *restrict,
          struct sched_param *restrict);

int   pthread_attr_getschedpolicy(const pthread_attr_t *restrict,
          int *restrict);


int   pthread_attr_getscope(const pthread_attr_t *restrict,
          int *restrict);


int   pthread_attr_getstack(const pthread_attr_t *restrict,
          void **restrict, size_t *restrict);


int   pthread_attr_getstackaddr(const pthread_attr_t *restrict,
          void **restrict);


int   pthread_attr_getstacksize(const pthread_attr_t *restrict,
          size_t *restrict);

int   pthread_attr_init(pthread_attr_t *);
int   pthread_attr_setdetachstate(pthread_attr_t *, int);

int   pthread_attr_setguardsize(pthread_attr_t *, size_t);


int   pthread_attr_setinheritsched(pthread_attr_t *, int);
int   pthread_attr_setschedparam(pthread_attr_t *restrict,
          const struct sched_param *restrict);

int   pthread_attr_setschedpolicy(pthread_attr_t *, int);
int   pthread_attr_setscope(pthread_attr_t *, int);


int   pthread_attr_setstack(pthread_attr_t *, void *, size_t);


int   pthread_attr_setstackaddr(pthread_attr_t *, void *);


int   pthread_attr_setstacksize(pthread_attr_t *, size_t);


int   pthread_barrier_destroy(pthread_barrier_t *);
int   pthread_barrier_init(pthread_barrier_t *restrict,
          const pthread_barrierattr_t *restrict, unsigned);
int   pthread_barrier_wait(pthread_barrier_t *);
int   pthread_barrierattr_destroy(pthread_barrierattr_t *);


int   pthread_barrierattr_getpshared(
          const pthread_barrierattr_t *restrict, int *restrict);


int   pthread_barrierattr_init(pthread_barrierattr_t *);


int   pthread_barrierattr_setpshared(pthread_barrierattr_t *, int);

int   pthread_cancel(pthread_t);
void  pthread_cleanup_push(void (*)(void *), void *);
void  pthread_cleanup_pop(int);
int   pthread_cond_broadcast(pthread_cond_t *);
int   pthread_cond_destroy(pthread_cond_t *);
int   pthread_cond_init(pthread_cond_t *restrict,
          const pthread_condattr_t *restrict);
int   pthread_cond_signal(pthread_cond_t *);
int   pthread_cond_timedwait(pthread_cond_t *restrict,
          pthread_mutex_t *restrict, const struct timespec *restrict);
int   pthread_cond_wait(pthread_cond_t *restrict,
          pthread_mutex_t *restrict);
int   pthread_condattr_destroy(pthread_condattr_t *);

int   pthread_condattr_getclock(const pthread_condattr_t *restrict,
          clockid_t *restrict);


int   pthread_condattr_getpshared(const pthread_condattr_t *restrict,
          int *restrict);

int   pthread_condattr_init(pthread_condattr_t *);
int   pthread_condattr_setclock(pthread_condattr_t *, clockid_t);


int   pthread_condattr_setpshared(pthread_condattr_t *, int);

int   pthread_create(pthread_t *restrict, const pthread_attr_t *restrict,
          void *(*)(void *), void *restrict);
int   pthread_detach(pthread_t);
int   pthread_equal(pthread_t, pthread_t);
void  pthread_exit(void *);

int   pthread_getconcurrency(void);


int   pthread_getcpuclockid(pthread_t, clockid_t *);


int   pthread_getschedparam(pthread_t, int *restrict,
          struct sched_param *restrict);

void *pthread_getspecific(pthread_key_t);
int   pthread_join(pthread_t, void **);
int   pthread_key_create(pthread_key_t *, void (*)(void *));
int   pthread_key_delete(pthread_key_t);
int   pthread_mutex_destroy(pthread_mutex_t *);

int   pthread_mutex_getprioceiling(const pthread_mutex_t *restrict,
          int *restrict);

int   pthread_mutex_init(pthread_mutex_t *restrict,
          const pthread_mutexattr_t *restrict);
int   pthread_mutex_lock(pthread_mutex_t *);

int   pthread_mutex_setprioceiling(pthread_mutex_t *restrict, int,
          int *restrict);


int   pthread_mutex_timedlock(pthread_mutex_t *,
          const struct timespec *);

int   pthread_mutex_trylock(pthread_mutex_t *);
int   pthread_mutex_unlock(pthread_mutex_t *);
int   pthread_mutexattr_destroy(pthread_mutexattr_t *);

int   pthread_mutexattr_getprioceiling(
          const pthread_mutexattr_t *restrict, int *restrict);


int   pthread_mutexattr_getprotocol(const pthread_mutexattr_t *restrict,
          int *restrict);


int   pthread_mutexattr_getpshared(const pthread_mutexattr_t *restrict,
          int *restrict);
int   pthread_mutexattr_gettype(const pthread_mutexattr_t *restrict,
          int *restrict);

int   pthread_mutexattr_init(pthread_mutexattr_t *);

int   pthread_mutexattr_setprioceiling(pthread_mutexattr_t *, int);


int   pthread_mutexattr_setprotocol(pthread_mutexattr_t *, int);


int   pthread_mutexattr_setpshared(pthread_mutexattr_t *, int);


int   pthread_mutexattr_settype(pthread_mutexattr_t *, int);

int   pthread_once(pthread_once_t *, void (*)(void));
int   pthread_rwlock_destroy(pthread_rwlock_t *);
int   pthread_rwlock_init(pthread_rwlock_t *restrict,
          const pthread_rwlockattr_t *restrict);
int   pthread_rwlock_rdlock(pthread_rwlock_t *);

int   pthread_rwlock_timedrdlock(pthread_rwlock_t *restrict,
          const struct timespec *restrict);
int   pthread_rwlock_timedwrlock(pthread_rwlock_t *restrict,
          const struct timespec *restrict);

int   pthread_rwlock_tryrdlock(pthread_rwlock_t *);
int   pthread_rwlock_trywrlock(pthread_rwlock_t *);
int   pthread_rwlock_unlock(pthread_rwlock_t *);
int   pthread_rwlock_wrlock(pthread_rwlock_t *);
int   pthread_rwlockattr_destroy(pthread_rwlockattr_t *);

int   pthread_rwlockattr_getpshared(
          const pthread_rwlockattr_t *restrict, int *restrict);

int   pthread_rwlockattr_init(pthread_rwlockattr_t *);

int   pthread_rwlockattr_setpshared(pthread_rwlockattr_t *, int);

pthread_t
      pthread_self(void);
int   pthread_setcancelstate(int, int *);
int   pthread_setcanceltype(int, int *);

int   pthread_setconcurrency(int);


int   pthread_setschedparam(pthread_t, int,
          const struct sched_param *);


int   pthread_setschedprio(pthread_t, int);
int   pthread_setspecific(pthread_key_t, const void *);

int   pthread_spin_destroy(pthread_spinlock_t *);
int   pthread_spin_init(pthread_spinlock_t *, int);
int   pthread_spin_lock(pthread_spinlock_t *);
int   pthread_spin_trylock(pthread_spinlock_t *);
int   pthread_spin_unlock(pthread_spinlock_t *);

void  pthread_testcancel(void);
```

Inclusion of the *<pthread.h>* header shall make symbols defined in the headers *<sched.h>* and *<time.h>* visible.

The following sections are informative.

APPLICATION USAGE

None.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

<sched.h>, *<sys/types.h>*, *<time.h>*, the System Interfaces volume of IEEE Std 1003.1-2001, *pthread_attr_getguardsize*(), *pthread_attr_init*(), *pthread_attr_setscope*(), *pthread_barrier_destroy*(), *pthread_barrier_init*(), *pthread_barrier_wait*(), *pthread_barrierattr_destroy*(), *pthread_barrierattr_getpshared*(), *pthread_barrierattr_init*(), *pthread_barrierattr_setpshared*(), *pthread_cancel*(), *pthread_cleanup_pop*(), *pthread_cond_init*(), *pthread_cond_signal*(), *pthread_cond_wait*(), *pthread_condattr_getclock*(), *pthread_condattr_init*(), *pthread_condattr_setclock*(), *pthread_create*(), *pthread_detach*(), *pthread_equal*(), *pthread_exit*(), *pthread_getconcurrency*(), *pthread_getcpuclockid*(), *pthread_getschedparam*(), *pthread_join*(), *pthread_key_create*(), *pthread_key_delete*(), *pthread_mutex_init*(), *pthread_mutex_lock*(), *pthread_mutex_setprioceiling*(), *pthread_mutex_timedlock*(), *pthread_mutexattr_init*(), *pthread_mutexattr_gettype*(), *pthread_mutexattr_setprotocol*(), *pthread_once*(), *pthread_rwlock_destroy*(), *pthread_rwlock_init*(), *pthread_rwlock_rdlock*(), *pthread_rwlock_timedrdlock*(), *pthread_rwlock_timedwrlock*(), *pthread_rwlock_tryrdlock*(), *pthread_rwlock_trywrlock*(), *pthread_rwlock_unlock*(), *pthread_rwlock_wrlock*(), *pthread_rwlockattr_destroy*(), *pthread_rwlockattr_getpshared*(), *pthread_rwlockattr_init*(), *pthread_rwlockattr_setpshared*(), *pthread_self*(), *pthread_setcancelstate*(), *pthread_setspecific*(), *pthread_spin_destroy*(), *pthread_spin_init*(), *pthread_spin_lock*(), *pthread_spin_trylock*(), *pthread_spin_unlock*()

COPYRIGHT