NAME

get_robust_list, set_robust_list − get/set the list of robust futexes

SYNOPSIS

#include <linux/futex.h>
#include <syscall.h>

long get_robust_list(int *pid***, struct robust_list_head ******head_ptr***,**
size_t *****len_ptr*);**
long set_robust_list(struct robust_list_head *****head*, size_t** *len***);**

DESCRIPTION

The robust futex implementation needs to maintain per-thread lists of robust futexes which are unlocked when the thread exits. These lists are managed in user space, the kernel is only notified about the location of the head of the list.

get_robust_list returns the head of the robust futex list of the thread with TID defined by the *pid* argument. If *pid* is 0, the returned head belongs to the current thread. *head_ptr* is the pointer to the head of the list of robust futexes. The **get_robust_list** function stores the address of the head of the list here. *len_ptr* is the pointer to the length variable. **get_robust_list** stores **sizeof(******head_ptr***)** here.

set_robust_list sets the head of the list of robust futexes owned by the current thread to *head*. *len* is the size of *******head*.

RETURN VALUE

The **set_robust_list** and **get_robust_list** functions return zero when the operation is successful, an error code otherwise.

ERRORS

The **set_robust_list** function fails with **EINVAL** if the *len* value does not match the size of structure **struct robust_list_head** expected by kernel.

The **get_robust_list** function fails with **EPERM** if the current process does not have permission to see the robust futex list of the thread with the TID *pid*, **ESRCH** if a thread with the TID *pid* does not exist, or **EFAULT** if the head of the robust futex list can't be stored in the space specified by the *head* argument.

APPLICATION USAGE

A thread can have only one robust futex list; therefore applications that wish to use this functionality should use robust mutexes provided by glibc.

The system call is only available for debugging purposes and is not needed for normal operations.

Both system calls are not available to application programs as functions; they can be called using the **syscall**(3) function.

SEE ALSO

futex(2), **pthread_mutexattr_setrobust_np**(3)