NAME

getcontext, **getcontextx**, **setcontext** - get and set user thread context

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <ucontext.h>

int
getcontext(*ucontext_t *ucp*);

*ucontext_t **
getcontextx(*void*);

int
setcontext(*const ucontext_t *ucp*);

DESCRIPTION

The **getcontext**() function saves the current thread's execution context in the structure pointed to by *ucp*. This saved context may then later be restored by calling **setcontext**().

The **getcontextx**() function saves the current execution context in the newly allocated structure *ucontext_t*, which is returned on success. If architecture defines additional CPU states that can be stored in extended blocks referenced from the *ucontext_t*, the memory for them may be allocated and their context also stored. Memory returned by **getcontextx**() function shall be freed using **free**(*3*).

The **setcontext**() function makes a previously saved thread context the current thread context, i.e., the current context is lost and **setcontext**() does not return. Instead, execution continues in the context specified by *ucp*, which must have been previously initialized by a call to **getcontext**(), makecontext(3), or by being passed as an argument to a signal handler (see sigaction(2)).

If *ucp* was initialized by **getcontext**(), then execution continues as if the original **getcontext**() call had just returned (again).

If *ucp* was initialized by makecontext(3), execution continues with the invocation of the function specified to makecontext(3). When that function returns, *ucp->uc_link* determines what happens next: if *ucp->uc_link* is NULL, the process exits; otherwise, **setcontext**(*ucp->uc_link*) is implicitly invoked.

If *ucp* was initialized by the invocation of a signal handler, execution continues at the point the thread

was interrupted by the signal.

RETURN VALUES

If successful, **getcontext**() returns zero and **setcontext**() does not return; otherwise -1 is returned. The **getcontextx**() returns pointer to the allocated and initialized context on success, and *NULL* on failure.

ERRORS

No errors are defined for **getcontext**() or **setcontext**(). The **getcontextx**() may return the following errors in *errno*:

[ENOMEM] No memory was available to allocate for the context or some extended state.

SEE ALSO

sigaction(2), sigaltstack(2), makecontext(3), ucontext(3)

STANDARDS

The **getcontext**() and **setcontext**() functions conform to X/Open System Interfaces and Headers Issue 5 ("XSH5") and IEEE Std 1003.1-2001 ("POSIX.1"). The *errno* indications are an extension to the standard.

The IEEE Std 1003.1-2004 ("POSIX.1") revision marked the functions **getcontext**() and **setcontext**() as obsolete, citing portability issues and recommending the use of POSIX threads instead. The IEEE Std 1003.1-2008 ("POSIX.1") revision removed the functions from the specification.

HISTORY

The **getcontext**() and **setcontext**() functions first appeared in AT&T System V Release 4 UNIX.