



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

\$ man pthread_getconcurrency.3p

PTHREAD_GETCONCURRENCY(3P) POSIX Programmer's ManualPTHREAD_GETCONCURRENCY(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_getconcurrency, pthread_setconcurrency ? get and set the level of concurrency

SYNOPSIS

```
#include <pthread.h>

int pthread_getconcurrency(void);

int pthread_setconcurrency(int new_level);
```

DESCRIPTION

Unbound threads in a process may or may not be required to be simultaneously active. By default, the threads implementation ensures that a sufficient number of threads are active so that the process can continue to make progress. While this conserves system resources, it may not produce the most effective level of concurrency.

The pthread_setconcurrency() function allows an application to inform the threads implementation of its desired concurrency level, new_level.

The actual level of concurrency provided by the implementation as a result of this function call is unspecified.

If `new_level` is zero, it causes the implementation to maintain the concurrency level at its discretion as if `pthread_setconcurrency()` had never been called.

The `pthread_getconcurrency()` function shall return the value set by a previous call to the `pthread_setconcurrency()` function. If the `pthread_setconcurrency()` function was not previously called, this function shall return zero to indicate that the implementation is maintaining the concurrency level.

A call to `pthread_setconcurrency()` shall inform the implementation of its desired concurrency level. The implementation shall use this as a hint, not a requirement.

If an implementation does not support multiplexing of user threads on top of several kernel-scheduled entities, the `pthread_setconcurrency()` and `pthread_getconcurrency()` functions are provided for source code compatibility but they shall have no effect when called. To maintain the function semantics, the `new_level` parameter is saved when `pthread_setconcurrency()` is called so that a subsequent call to `pthread_getconcurrency()` shall return the same value.

RETURN VALUE

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

The `pthread_getconcurrency()` function shall always return the concurrency level set by a previous call to `pthread_setconcurrency()`. If the `pthread_setconcurrency()` function has never been called, `pthread_getconcurrency()` shall return zero.

ERRORS

The `pthread_setconcurrency()` function shall fail if:

EINVAL The value specified by `new_level` is negative.

EAGAIN The value specified by `new_level` would cause a system resource to be exceeded.

The `pthread_setconcurrency()` function shall not return an error code of `[EINTR]`.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

Application developers should note that an implementation can always ignore any calls to `pthread_setconcurrency()` and return a constant for `pthread_getconcurrency()`. For this reason, it is not recommended that portable applications use this function.

RATIONALE

None.

FUTURE DIRECTIONS

These functions may be removed in a future version.

SEE ALSO

The Base Definitions volume of POSIX.1-2017, `<pthread.h>`

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1-2017, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 7, 2018 Edition, Copyright (C) 2018 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.opengroup.org/unix/online.html>.

Any typographical or formatting errors that appear in this page are most likely to have been introduced during the conversion of the source files to man page format. To report such errors, see https://www.kernel.org/doc/man-pages/reporting_bugs.html.

IEEE/The Open Group 2017 PTHREAD_GETCONCURRENCY(3P)