NAME

timer_create – create a timer

SYNOPSIS

cc [*flag* ...] *file* ... **–lposix4** [*library* ...]

```
#include <signal.h>
#include <time.h>

int timer_create(clockid_t clock_id, struct sigevent *evp, timer_t *timerid);

struct sigevent {
        int             sigev_notify    /* notification type */
        int             sigev_signo;    /* signal number */
        union sigval    sigev_value;    /* signal value */
};

union sigval {
        int             sival_int;      /* integer value */
        void            *sival_ptr;     /* pointer value */
};
```

MT-LEVEL

MT-Safe with exceptions

DESCRIPTION

timer_create( ) creates a timer using the specified clock, *clock_id,* as the timing base. This timer ID is unique and meaningful only within the calling LWP until the timer is deleted. This timer is initially disarmed upon return from **timer_create( ).**

The timer may be created per-LWP or per-process. Expiration signals for a per-LWP timer will be sent to the creating LWP. Expiration signals for a per-process timer will be sent to the process. A per-LWP timer will be automatically deleted when the creating LWP exits. By default, timers are created per-LWP. If the symbol **_POSIX_PER_PROCESS_TIMERS** is defined or the symbol **_POSIX_C_SOURCE** is defined to have a value greater than **199500L** before the inclusion of **<time.h>**, timers will be created per-process.

If *evp* is non-**NULL**:

> then *evp* points to a **sigevent** structure, allocated by the application, which defines the asynchronous notification that will occur when the timer expires.
>
> If the **sigev_notify** member of *evp* is **SIGEV_SIGNAL,** then the structure also contains the signal number and the application specific data value to be sent to the process. If **SA_SIGINFO** is set for the expiration signal, then the signal and application-defined value specified in the structure will be queued to the process on timer expiration. If **SA_SIGINFO** is not set for the expiration signal, then the signal specified in the structure will be sent upon the timer expiration.
>
> If the **sigev_notify** member is **SIGEV_NONE,** no notification will be sent.

If *evp* is **NULL,** and **SA_SIGINFO** is set for the expiration signal, then the default signal, **SIGALRM,** will be queued to the process and the signal data value will be set to the timer ID.

RETURN VALUES

timer_create( ) returns **0** upon success and creates a **timer_t,** *timerid*, which can be passed to the timer calls; otherwise it returns **-1** and sets **errno** to indicate the error condition.

ERRORS

EAGAIN The system lacks sufficient signal queuing resources to honor the request.

The calling process has already created all of the timers it is allowed by this implementation.

EINVAL The specified clock ID, *clock_id,* is not defined.

ENOSYS **timer_create( )** is not supported by this implementation.

SEE ALSO

exec(2), **fork**(2), **time**(2), **clock_settime**(3R), **timer_delete**(3R), **timer_settime**(3R)

NOTES

Timers are not inherited by a child process across a **fork**(2) and can be disarmed and deleted by an **exec**(2).

In a future release, the ability to create per-LWP timers will be removed, and all calls to **timer_create**() will result in per-process timers.