NAME

ntp_adjtime, **ntp_gettime** **-** Network Time Protocol (NTP) daemon interface system calls

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <sys/timex.h>

int
ntp_adjtime(*struct timex **);

int
ntp_gettime(*struct ntptimeval **);

DESCRIPTION

The two system calls **ntp_adjtime**() and **ntp_gettime**() are the kernel interface to the Network Time Protocol (NTP) daemon ntpd(8).

The **ntp_adjtime**() function is used by the NTP daemon to adjust the system clock to an externally derived time. The time offset and related variables which are set by **ntp_adjtime**() are used by **hardclock**() to adjust the phase and frequency of the phase- or frequency-lock loop (PLL resp. FLL) which controls the system clock.

The **ntp_gettime**() function provides the time, maximum error (sync distance) and estimated error (dispersion) to client user application programs.

In the following, all variables that refer PPS are only relevant if the *PPS_SYNC* option is enabled in the kernel.

ntp_adjtime() has as argument a *struct timex ** of the following form:

```
struct timex {
        unsigned int modes;         /* clock mode bits (wo) */
        long offset;                /* time offset (us) (rw) */
        long freq;          /* frequency offset (scaled ppm) (rw) */
        long maxerror;              /* maximum error (us) (rw) */
        long esterror;              /* estimated error (us) (rw) */
        int status;         /* clock status bits (rw) */
        long constant;              /* pll time constant (rw) */
        long precision;              /* clock precision (us) (ro) */
        long tolerance;              /* clock frequency tolerance (scaled
                                      * ppm) (ro) */
        /*
         * The following read-only structure members are implemented
         * only if the PPS signal discipline is configured in the
         * kernel.
         */
        long ppsfreq;                /* pps frequency (scaled ppm) (ro) */
        long jitter;                 /* pps jitter (us) (ro) */
        int shift;          /* interval duration (s) (shift) (ro) */
        long stabil;                 /* pps stability (scaled ppm) (ro) */
        long jitcnt;                 /* jitter limit exceeded (ro) */
        long calcnt;                 /* calibration intervals (ro) */
        long errcnt;                 /* calibration errors (ro) */
        long stbcnt;                 /* stability limit exceeded (ro) */
};
```

The members of this struct have the following meanings when used as argument for **ntp_adjtime**():

modes Defines what settings should be changed with the current **ntp_adjtime**() call (write-only). Bitwise OR of the following:

MOD_OFFSET	set time offset
MOD_FREQUENCY	set frequency offset
MOD_MAXERROR	set maximum time error
MOD_ESTERROR	set estimated time error
MOD_STATUS	set clock status bits
MOD_TIMECONST	set PLL time constant
MOD_CLKA	set clock A
MOD_CLKB	set clock B

offset Time offset (in microseconds), used by the PLL/FLL to adjust the system time in small increments (read-write).

freq Frequency offset (scaled ppm) (read-write).

maxerror Maximum error (in microseconds). Initialized by an **ntp_adjtime**() call, and increased by the kernel once each second to reflect the maximum error bound growth (read-write).

esterror Estimated error (in microseconds). Set and read by **ntp_adjtime**(), but unused by the kernel (read-write).

status System clock status bits (read-write). Bitwise OR of the following:

STA_PLL	Enable PLL updates (read-write).
STA_PPSFREQ	Enable PPS freq discipline (read-write).

STA_PPSTIME	Enable PPS time discipline (read-write).
STA_FLL	Select frequency-lock mode (read-write).
STA_INS	Insert leap (read-write).
STA_DEL	Delete leap (read-write).
STA_UNSYNC	Clock unsynchronized (read-write).
STA_FREQHOLD	Hold frequency (read-write).
STA_PPSSIGNAL	PPS signal present (read-only).
STA_PPSJITTER	PPS signal jitter exceeded (read-only).
STA_PPSWANDER	PPS signal wander exceeded (read-only).
STA_PPSERROR	PPS signal calibration error (read-only).
STA_CLOCKERR	Clock hardware fault (read-only).

constant	PLL time constant, determines the bandwidth, or "stiffness", of the PLL (read-write).
precision	Clock precision (in microseconds). In most cases the same as the kernel tick variable (see hz(9)). If a precision clock counter or external time-keeping signal is available, it could be much lower (and depend on the state of the signal) (read-only).
tolerance	Maximum frequency error, or tolerance of the CPU clock oscillator (scaled ppm). Ordinarily a property of the architecture, but could change under the influence of external time-keeping signals (read-only).
ppsfreq	PPS frequency offset produced by the frequency median filter (scaled ppm) (read-only).
jitter	PPS jitter measured by the time median filter in microseconds (read-only).
shift	Logarithm to base 2 of the interval duration in seconds (PPS, read-only).
stabil	PPS stability (scaled ppm); dispersion (wander) measured by the frequency median filter (read-only).
jitcnt	Number of seconds that have been discarded because the jitter measured by the time median filter exceeded the limit *MAXTIME* (PPS, read-only).
calcnt	Count of calibration intervals (PPS, read-only).
errcnt	Number of calibration intervals that have been discarded because the wander exceeded the limit *MAXFREQ* or where the calibration interval jitter exceeded two ticks (PPS, read-only).
stbcnt	Number of calibration intervals that have been discarded because the frequency wander exceeded the limit *MAXFREQ*/4 (PPS, read-only).

After the **ntp_adjtime**() call, the *struct timex ** structure contains the current values of the corresponding variables.

ntp_gettime() has as argument a *struct ntptimeval ** with the following members:

```
struct ntptimeval {
        struct timeval time; /* current time (ro) */
        long maxerror;         /* maximum error (us) (ro) */
        long esterror;         /* estimated error (us) (ro) */
};
```

These have the following meaning:

time	Current time (read-only).
maxerror	Maximum error in microseconds (read-only).
esterror	Estimated error in microseconds (read-only).

RETURN VALUES

ntp_adjtime() and **ntp_gettime**() return the current state of the clock on success, or any of the errors of copyin(9) and copyout(9). **ntp_adjtime**() may additionally return EPERM if the user calling **ntp_adjtime**() does not have sufficient permissions.

Possible states of the clock are:

TIME_OK	Everything okay, no leap second warning.
TIME_INS	"insert leap second" warning. At the end of the day, a leap second will be inserted after 23:59:59.
TIME_DEL	"delete leap second" warning. At the end of the day, second 23:59:59 will be skipped.
TIME_OOP	Leap second in progress.
TIME_WAIT	Leap second has occurred within the last few seconds.
TIME_ERROR	Clock not synchronized.

ERRORS

The **ntp_adjtime**() system call may return EPERM if the caller does not have sufficient permissions.

SEE ALSO

options(4), ntpd(8), hardclock(9), hz(9)

http://www.bipm.fr/enus/5_Scientific/c_time/time_1.html

http://www.boulder.nist.gov/timefreq/general/faq.htm

ftp://time.nist.gov/pub/leap-seconds.list

BUGS

Take note that this API is extremely complex and stateful. Users should not attempt modification without first reviewing the ntpd(8) sources in depth.