NAME

gettimeofday, settimeofday - get or set the date and time

SYNOPSIS

```
#include <sys/time.h>

int gettimeofday(struct timeval *tp, void *);

int settimeofday(struct timeval *tp, void *);
```

DESCRIPTION

The gettimeofday() function gets and the settimeofday() function sets the system's notion of the current time. The current time is expressed in elapsed seconds and microseconds since 00:00 Universal Coordinated Time, January 1, 1970. The resolution of the system clock is hardware dependent; the time may be updated continuously or in clock ticks.

The tp argument points to a timeval structure, which includes the following members:

```
long    tv_sec;    /* seconds since Jan. 1, 1970 */
long    tv_usec;   /* and microseconds */
```

If tp is a null pointer, the current time information is not returned or set.

The TZ environment variable holds time zone information. See TIMEZONE(4).

The second argument to gettimeofday() and settimeofday() should be a pointer to NULL.

Only the super-user may set the time of day.

RETURN VALUES

Upon successful completion, 0 is returned. Otherwise, -1 is returned and errno is set to indicate the error.

ERRORS

The gettimeofday() function will fail if:

EINVAL The structure pointed to by tp specifies an invalid time.

EPERM A user other than the privileged user attempted to set the time or time zone.

Additionally, the gettimeofday() function will fail for 32-bit interfaces if:

EOVERFLOW The system time has progressed beyond 2038, thus the size of the tv_sec member of the timeval structure pointed to by tp is insufficient to hold the current time in seconds.

USAGE

If the tv_usec member of tp is > 500000, settimeofday() rounds the seconds upward. If the time needs to be set with better than one second accuracy, call settimeofday() for the seconds and then adjtime(2) for finer accuracy.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT-Level	MT-Safe

SEE ALSO

adjtime(2), ctime(3C), TIMEZONE(4), attributes(5)

NAME

adjtime - correct the time to allow synchronization of the system clock

SYNOPSIS

```
#include <sys/time.h>

int adjtime(struct timeval *delta, struct timeval *olddelta);
```

DESCRIPTION

The adjtime() function adjusts the system's notion of the current time as returned by gettimeofday(3C), advancing or retarding it by the amount of time specified in the struct timeval pointed to by delta.

The adjustment is effected by speeding up (if that amount of time is positive) or slowing down (if that amount of time is negative) the system's clock by some small percentage, generally a fraction of one percent. The time is always a monotonically increasing function. A time correction from an earlier call to adjtime() may not be finished when adjtime() is called again.

If delta is 0, then olddelta returns the status of the effects of the previous adjtime() call with no effect on the time correction as a result of this call. If olddelta is not a null pointer, then the structure it points to will contain, upon successful return, the number of seconds and/or microseconds still to be corrected from the earlier call. If olddelta is a null pointer, the corresponding information will not be returned.

This call may be used in time servers that synchronize the clocks of computers in a local area network. Such time servers would slow down the clocks of some machines and speed up the clocks of others to bring them to the average network time.

Only the super-user may adjust the time of day.

The adjustment value will be silently rounded to the resolution of the system clock.

RETURN VALUES

Upon successful completion, adjtime() returns 0. Otherwise, it returns -1 and sets errno to indicate the error.

ERRORS

The adjtime() function will fail if:

EFAULT The delta or olddelta argument points outside the process's allocated address space, or olddelta points to a region of the process's allocated address space that is not writable.

EINVAL The tv_usec member of delta is not within valid range (-1000000 to 1000000).

EPERM The effective user of the calling process is not super-user.

Additionally, the adjtime() function will fail for 32-bit interfaces if:

EOVERFLOW The size of the tv_sec member of the timeval structure pointed to by olddelta is too small to contain the correct number of seconds.

SEE ALSO

date(1), gettimeofday(3C)

NAME

gethrtime, gethrvtime - get high resolution time

SYNOPSIS

```
#include <sys/time.h>

hrtime_t gethrtime(void);

hrtime_t gethrvtime(void);
```

DESCRIPTION

The gethrtime() function returns the current high-resolution real time. Time is expressed as nanoseconds since some arbitrary time in the past; it is not correlated in any way to the time of day, and thus is not subject to resetting or drifting by way of adjtime(2) or settimeofday(3C). The hi-res timer is ideally suited to performance measurement tasks, where cheap, accurate interval timing is required.

The gethrvtime() function returns the current high-resolution LWP virtual time, expressed as total nanoseconds of execution time. This function requires that micro state accounting be enabled with the ptime utility (see proc(1)).

The gethrtime() and gethrvtime() functions both return an hrtime_t, which is a 64-bit (long long) signed integer.

EXAMPLES

The following code fragment measures the average cost of getpid(2):

```
hrtime_t start, end;
int i, iters = 100;

start = gethrtime();
for (i = 0; i < iters; i++)
        getpid();
end = gethrtime();

printf("Avg getpid() time = %lld nsec\n", (end - start) / iters);
```

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT-Level	MT-Safe

SEE ALSO

proc(1), adjtime(2), gettimeofday(3C), settimeofday(3C), attributes(5)

NOTES

Although the units of hi-res time are always the same (nanoseconds), the actual resolution is hardware dependent. Hi-res time is guaranteed to be monotonic (it won't go backward, it won't periodically wrap) and linear (it won't occasionally speed up or slow down for adjustment, like the time of day can), but not necessarily unique: two sufficiently proximate calls may return the same value.

NAME

rtc - provide all real-time clock and GMT-lag management

SYNOPSIS

```
rtc  [ -c ]  [ -z zone-name ]
```

DESCRIPTION

The /usr/sbin/rtc command is used to reconcile the difference in the way time is established between UNIX systems, which utilize Greenwich Mean Time (GMT), and MS-DOS systems, which utilize local time. When used with no arguments, rtc displays the currently configured time zone string which was recorded when rtc -z zone-name ws last run.

The rtc command is not normally run from a shell prompt; it is generally invoked by the system. Commands such as date(1) and rdate(1M), which are used to set the time on a system, invoke /usr/sbin/rtc -c to ensure that daylight savings time (DST) is corrected for properly.

OPTIONS

-c
This option checks for DST and makes corrections if necessary. It is normally run once a day by a cron job.

If there is no RTC time zone or /etc/rtc_config file, this option will do nothing.

-z zone-name
This option, which is normally run by the system at software installation time, is used to specify the time zone in which the RTC is to be maintained. It updates the configuration file /etc/rtc_config with the name of the specified zone and the current GMT lag for that zone. If there is an existing rtc_config file, this command will update it. If not, this command will create it.

FILES

/usr/sbin/rtc
The command used to provide all real-time clock and GMT lag management.

/etc/rtc_config
The data file used to record the time zone and GMT lag. This file is completely managed by /usr/sbin/rtc, and it is read by the kernel.

ATTRIBUTES

See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Architecture	x86
Availability	SUNWcsu

SEE ALSO

date(1), rdate(1M), attributes(5)