



@server

iSeries

UNIX-Type -- Time APIs

Version 5 Release 3





@server

iSeries

UNIX-Type -- Time APIs

Version 5 Release 3

Note

Before using this information and the product it supports, be sure to read the information in "Notices," on page 25.

Sixth Edition (August 2005)

This edition applies to version 5, release 3, modification 0 of Operating System/400 (product number 5722-SS1) and to all subsequent releases and modifications until otherwise indicated in new editions. This version does not run on all reduced instruction set computer (RISC) models nor does it run on CISC models.

© Copyright International Business Machines Corporation 1998, 2005. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Time APIs	1	Parameters	11
System Clock APIs	1	Authorities and Locks	11
APIs	1	Return Value	11
adjtime()—Adjust System Clock	2	Error Conditions.	11
Parameters	2	Error Messages	11
Authorities and Locks	2	Usage Notes	12
Return Value	2	Related Information	12
Error Conditions	2	Example	12
Error Messages	3	Example Output:	12
Related Information	3	Qp0zGetTimeofday()—Get Current Software Clock	
Example	3	Time.	13
ftime()—Get Date and Time	4	Parameters	13
Parameters	4	Authorities and Locks	13
Authorities and Locks	4	Return Value	13
Return Value	4	Error Conditions.	13
Error Conditions	4	Error Messages	14
Error Messages	5	Usage Notes	14
Related Information	5	Related Information	14
Example	5	Example	14
gettimeofday()—Get Current UTC Time	6	Example Output:	14
Parameters	6	Qp0zSetTimeofday()—Set Software Clock	15
Authorities and Locks	6	Parameters	15
Return Value	6	Authorities and Locks	15
Error Conditions	6	Return Value	15
Error Messages	7	Error Conditions.	15
Usage Notes	7	Error Messages	16
Related Information	7	Usage Notes	16
Example	7	Related Information	16
settimeofday()—Set System Clock	8	Example	16
Parameters	8	Example Output.	17
Authorities and Locks	8	Concepts	17
Return Value	8	Header Files for UNIX-Type Functions	17
Error Conditions	8	Errno Values for UNIX-Type Functions	20
Error Messages	9		
Related Information	9		
Example	9		
Software Clock APIs	10		
APIs.	10		
Qp0zAdjTime()—Adjust Software Clock	10		
		Appendix. Notices	25
		Trademarks	26
		Terms and conditions for downloading and printing	
		publications	27
		Code disclaimer information.	28

Time APIs

» The Time APIs include two sets of APIs:

- “System Clock APIs”
- “Software Clock APIs” on page 10

Both the System Clock and Software Clock APIs use a `timeval` structure that is the number of seconds and microseconds since 1 January 1970, 00:00:00 Universal Coordinated Time (UTC).

The System Clock APIs allow you to work with the system clock. The system clock is a system facility that maintains the time of the system. The `gettimeofday()` and `ftime()` APIs get the current time from the system clock. The `adjtime()` and `settimeofday()` APIs set the system clock. See the Time Management topic for more information about the system clock.

The Software Clock APIs allow you to work with the software clock. The software clock is a system facility that maintains a time that is independent from the system clock. The `Qp0zGetTimeofday()` API gets the current time from the software clock. The `Qp0zAdjTime()` and `Qp0zSetTimeofday()` APIs set the software clock. In previous releases, the software clock was the only way to get a UTC time in seconds and microseconds. System components do not base their timestamps on the software clock, but use the system clock instead. The software clock will be removed in a future release and its use is discouraged.◀◀

[Top](#) | [UNIX-Type APIs](#) | [APIs by category](#)

System Clock APIs

The System Clock APIs are:

- “`adjtime()`—Adjust System Clock” on page 2 (Adjust system clock) makes small adjustments to the software clock, either slowing it down or speeding it up by the time specified in the delta parameter.
- » “`ftime()`—Get Date and Time” on page 4 (Get Date and Time) retrieves the current Coordinated Universal Time (UTC).◀◀
- “`gettimeofday()`—Get Current UTC Time” on page 6 (Get Current UTC Time) retrieves the current Coordinated Universal Time (UTC) and places it in the `timeval` structure pointed to by `tp`.
- “`settimeofday()`—Set System Clock” on page 8 (Set System Clock) sets the system clock to the Coordinated Universal Time (UTC) contained in the `timeval` structure pointed to by `tp`.

Note: These functions use header (include) files from the library `QSYSINC`, which is optionally installable. Make sure `QSYSINC` is installed on your system before using any of the functions. See for the file and member name of each header file.



[Top](#) | [UNIX-Type APIs](#) | [APIs by category](#)

APIs

These are the APIs for this category.

adjtime()—Adjust System Clock

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

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

Service Program Name: >> QWCTZUTC
<<

Default Public Authority: *USE

Threadsafe: Yes

>> The **adjtime()** function makes small adjustments to the system clock, either slowing it down or speeding it up by the time specified in the *delta* parameter up to a maximum of two hours. If *delta* is negative, the clock is slowed down by incrementing it more slowly than normal until the correction is complete. If *delta* is positive, the clock is sped up by incrementing it more quickly than normal until the correction is complete. If *olddelta* is not NULL, the amount of time still to be corrected from a previous **adjtime()** call is returned in the structure it points to. <<

Parameters

delta (Input)

A pointer to a `timeval` structure that contains the amount of time for adjusting the clock.

olddelta

(Output)

A pointer to a `timeval` structure that contains the amount of time still to be corrected from a previous call to **adjtime()**.

Authorities and Locks

Special Authority

*ALLOBJ

Return Value

- 0 **adjtime()** was successful. The requested adjustment was initiated and the value returned in the structure pointed to by the *olddelta* parameter is the amount of time still to be corrected from a previous **adjtime()**.
- 1 **adjtime()** was not successful. The *errno* variable is set to indicate the error.

Error Conditions

If **adjtime()** is not successful, *errno* usually indicates one of the following errors. Under some conditions, *errno* could indicate an error other than those listed here.

[EACCES] Permission denied.

An attempt was made to access an object in a way forbidden by its object access permissions.

[EINVAL]	An invalid parameter was found.
[EFAULT]	A parameter passed to this function is not valid. The address used for an argument is not correct. In attempting to use an argument in a call, the system detected an address that is not valid. While attempting to access a parameter passed to this function, the system detected an address that is not valid.
[EPERM]	Operation not permitted. You must have appropriate privileges or be the owner of the object or other resource to do the requested operation.
» [ENOTSUP]	Operation not supported. The operation is not supported. «
[EUNKNOWN]	Unknown system state. The operation failed because of an unknown system state. See any messages in the job log and correct any errors that are indicated, then retry the operation.

Error Messages

None.

Related Information

- The <sys/time.h> file (see “Header Files for UNIX-Type Functions” on page 17)
- “gettimeofday()—Get Current UTC Time” on page 6
- “settimeofday()—Set System Clock” on page 8

Example

See Code disclaimer information for information pertaining to code examples.

The following example initiates a system clock adjustment:

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

int main(int argc, char *argv[])
{
    struct timeval adj, old;
    int rc;

    /* Speed up the clock by 1.5 seconds. */
    adj.tv_sec=1;
    adj.tv_usec=500000;

    rc=adjtime(&adj, &old);
    if(rc==0) {
        printf("adjtime() successful. "
            "Olddelta = %u.%06u\n",
            old.tv_sec, old.tv_usec);
    }
    else {
        printf("adjtime() failed, errno = %d\n",errno);
        return -1;
    }

    return 0;
}
```

Example Output:

```
adjtime() successful. Olddelta = 0.000000
```

API introduced: V4R2

Top | UNIX-Type APIs | APIs by category

ftime()—Get Date and Time

```
Syntax
#include <sys/timeb.h>

int ftime (struct timeb *tp);

Service Program Name: QWCTZUTC

Default Public Authority: *USE

Threadsafe: Yes
```

The **ftime()** function retrieves the current Coordinated Universal Time (UTC) and places it in `timeb` structure pointed to by `tp`.

Parameters

tp (Output) A pointer to a `timeb` structure that contains the time in seconds and milliseconds since 1 January 1970, 00:00:00 UTC (epoch-1970), the local time zone (measured in minutes west of Greenwich), and a flag that, if nonzero, indicates Daylight Saving Time is currently in effect.

Authorities and Locks

None

Return Value

0 **ftime()** was successful.
-1 **ftime()** was not successful. The `errno` variable is set to indicate the error.

Error Conditions

If **ftime()** is not successful, `errno` usually indicates one of the following errors. Under some conditions, `errno` could indicate an error other than those listed here.

[EINVAL] An invalid parameter was found.

[EFAULT] A parameter passed to this function is not valid.
The address used for an argument is not correct.

In attempting to use an argument in a call, the system detected an address that is not valid.

While attempting to access a parameter passed to this function, the system detected an address that is not valid.

[EUNKNOWN] Unknown system state.

The operation failed because of an unknown system state. See any messages in the job log and correct any errors that are indicated, then retry the operation.

Error Messages

None.

Related Information

- The <sys/timeb.h> file (see “Header Files for UNIX-Type Functions” on page 17)
- “gettimeofday()—Get Current UTC Time” on page 6

Example

See Code disclaimer information for information pertaining to code examples.

The following example gets the current date and time:

```
#include <sys/timeb.h>
#include <stdio.h>
#include <errno.h>

int main(int argc, char *argv[])
{
    struct timeb now;
    int rc;

    rc=ftime(&now);
    if(rc==0) {
        printf("ftime() successful.\n");
        printf("time = %u.%03u, "
              "timezone = %d, "
              "dstflag = %d\n",
              now.time, now.millitm,
              now.timezone,
              now.dstflag );
    }
    else {
        printf("ftime() failed, errno = %d\n",
              errno);
        return -1;
    }

    return 0;
}
```

Example Output:

```
ftime() successful.
time = 1019833362.226 timezone = 360, dstflag = 1
```



API introduced: V5R3

gettimeofday()—Get Current UTC Time

Syntax

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

```
int gettimeofday (struct timeval *tp,  
                 struct timezone *tzp);
```

Service Program Name: **»** QWCTZUTC

«

Default Public Authority: *USE

Threadsafe: Yes

» The `gettimeofday()` function retrieves the current Coordinated Universal Time (UTC) and places it in the `timeval` structure pointed to by `tp`. If `tzp` is not NULL, the time zone information is returned in the time zone structure pointed to by `tzp`. **«**

Parameters

- tp** (Output) A pointer to a `timeval` structure that contains the time in seconds and microseconds since 1 January 1970, 00:00:00 UTC (epoch-1970).
- tzp** (Output) A pointer to a time zone structure that contains the local time zone (measured in minutes west of Greenwich) and a flag that, if nonzero, indicates Daylight Saving Time applies locally during the appropriate part of the year.

Authorities and Locks

None

Return Value

- 0 `gettimeofday()` was successful.
- 1 `gettimeofday()` was not successful. The `errno` variable is set to indicate the error.

Error Conditions

If `gettimeofday()` is not successful, `errno` usually indicates one of the following errors. Under some conditions, `errno` could indicate an error other than those listed here.

- [EINVAL] An invalid parameter was found.
- [EFAULT] A parameter passed to this function is not valid.
The address used for an argument is not correct.
- In attempting to use an argument in a call, the system detected an address that is not valid.
- While attempting to access a parameter passed to this function, the system detected an address that is not valid.

➤ [EUNKNOWN] Unknown system state.

The operation failed because of an unknown system state. See any messages in the job log and correct any errors that are indicated, then retry the operation. ⏪

Error Messages

None.

Usage Notes

- For the best performance, specify NULL for the *tzp* parameter.

Related Information

- The <sys/time.h> file (see “Header Files for UNIX-Type Functions” on page 17)
- “adjtime()—Adjust System Clock” on page 2
- “ftime()—Get Date and Time” on page 4
- Qp0zCvtToMITime()—Convert Timeval Structure to _MI_Time
- “settimeofday()—Set System Clock” on page 8

Example

See Code disclaimer information for information pertaining to code examples.

The following example gets the current UTC time:

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

int main(int argc, char *argv[])
{
    struct timeval now;
    int rc;

    rc=gettimeofday(&now, NULL);
    if(rc==0) {
        printf("gettimeofday() successful.\n");
        printf("time = %u.%06u\n",
              now.tv_sec, now.tv_usec);
    }
    else {
        printf("gettimeofday() failed, errno = %d\n",
              errno);
        return -1;
    }

    return 0;
}
```

Example Output:

```
gettimeofday() successful.
time = 866208142.290944
```

API introduced: V4R2

settimeofday()—Set System Clock

Syntax

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

```
int settimeofday (struct timeval *tp,  
                 struct timezone *tzp);
```

Service Program Name: **»** QWCTZUTC

«

Default Public Authority: *USE

Threadsafe: Yes

» The `settimeofday()` function sets the system clock to the Coordinated Universal Time (UTC) contained in the `timeval` structure pointed to by `tp`. The `tzp` parameter is not used. **«**

Parameters

tp (Input) A pointer to a `timeval` structure that contains the time in seconds and microseconds since 1 January 1970, 00:00:00 UTC (epoch-1970).

tzp **»** This parameter is not used. **«**

Authorities and Locks

Special Authority
*ALLOBJ

Return Value

0 `settimeofday()` was successful.

-1 `settimeofday()` was not successful. The `errno` variable is set to indicate the error.

Error Conditions

If `settimeofday()` is not successful, `errno` usually indicates one of the following errors. Under some conditions, `errno` could indicate an error other than those listed here.

[EACCES]

Permission denied.

[EINVAL]

An attempt was made to access an object in a way forbidden by its object access permissions.

An invalid parameter was found.

[EFAULT]

A parameter passed to this function is not valid. The address used for an argument is not correct.

In attempting to use an argument in a call, the system detected an address that is not valid.

While attempting to access a parameter passed to this function, the system detected an address that is not valid.

[EPERM]

Operation not permitted.

You must have appropriate privileges or be the owner of the object or other resource to do the requested operation. Unknown system state.

[EUNKNOWN]

The operation failed because of an unknown system state. See any messages in the job log and correct any errors that are indicated, then retry the operation.

Error Messages

None.

Related Information

- The <sys/time.h> file (see “Header Files for UNIX-Type Functions” on page 17)
- “adjtime()—Adjust System Clock” on page 2
- “gettimeofday()—Get Current UTC Time” on page 6

Example

See Code disclaimer information for information pertaining to code examples.

The following example sets the system clock:

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

int main(int argc, char *argv[])
{
    struct timeval now;
    int rc;

    now.tv_sec=866208142;
    now.tv_usec=290944;

    rc=settimeofday(&now, NULL);
    if(rc==0) {
        printf("settimeofday() successful.\n");
    }
    else {
        printf("settimeofday() failed, "
            "errno = %d\n",errno);
        return -1;
    }

    return 0;
}
```

Example Output:

```
settimeofday() successful.
```

API introduced: V4R2

Software Clock APIs

» The Software Clock APIs are:

- » “Qp0zAdjTime()—Adjust Software Clock” (Adjust Software Clock) makes small adjustments to the software clock, either slowing it down or speeding it up by the time specified in the delta parameter.◀
- » “Qp0zGetTimeofday()—Get Current Software Clock Time” on page 13 (Get Current Software Clock Time) retrieves the current software clock time and places it in the timeval structure pointed to by tp.◀
- » “Qp0zSetTimeofday()—Set Software Clock” on page 15 (Set Software Clock) sets the software clock to the time contained in the timeval structure pointed to by tp.◀

Note: These functions use header (include) files from the library QSYSINC, which is optionally installable. Make sure QSYSINC is installed on your system before using any of the functions. See “Header Files for UNIX-Type Functions” on page 17 for the file and member name of each header file.

The Software Clock is a system facility that determines the current Universal Coordinated Time (UTC) in seconds and microseconds since 1 January 1970, 00:00:00 (epoch-1970). The current time is determined by keeping an internal ‘time-delta’, which is an offset from the system clock. When **Qp0zGetTimeofday()** is called to retrieve the software clock time, the time returned is calculated by taking the current system time, subtracting the QUTCOFFSET system value, then adding the internal time-delta.

The **Qp0zSetTimeofday()** API sets or changes the time-delta, without affecting the system clock or the QUTCOFFSET system value. The **Qp0zAdjtime()** API slowly changes the time-delta, without affecting the system clock either. Adjustments are made at a rate of approximately 1 second of adjustment for every 100 seconds of elapsed time.

System components do not base their timestamps on the software clock, but use the system clock instead. The software clock will be removed in a future release and its use is discouraged. ◀

Top | UNIX-Type APIs | APIs by category

APIs

These are the APIs for this category.

Qp0zAdjTime()—Adjust Software Clock

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

int Qp0zAdjTime (struct timeval *delta,
                struct timeval *olddelta);
```

Service Program Name: QP0ZSETC

Default Public Authority: *USE

Threadsafe: Yes

The **Qp0zAdjTime()** function makes small adjustments to the software clock, either slowing it down or speeding it up by the time specified in the *delta* parameter. If *delta* is negative, the clock is slowed down by incrementing it more slowly than normal until the correction is complete. If *delta* is positive, the clock

is sped up by incrementing it more quickly than normal until the correction is complete. If *olddelta* is not NULL, the amount of time still to be corrected from a previous **Qp0zAdjTime()** call is returned in the structure it points to.

The software clock maintains a time that can be set independently of the system clock. It is not integrated with the system and will be removed in a future release. The “adjtime()—Adjust System Clock” on page 2 function should be used instead.

Parameters

delta (Input) A pointer to a `timeval` structure that contains the amount of time for adjusting the software clock.

olddelta (Output) A pointer to a `timeval` structure that contains the amount of time still to be corrected from a previous call to **Qp0zAdjTime()**

Authorities and Locks

QSYS/QP0ZXCPA Service Program Authority
*USE

Return Value

- 0 **Qp0zAdjTime()** was successful. The requested adjustment was initiated and the value returned in the structure pointed to by the *olddelta* parameter is the amount of time still to be corrected from a previous **Qp0zAdjTime()**.
- 1 **Qp0zAdjTime()** was not successful. The *errno* variable is set to indicate the error.

Error Conditions

If **Qp0zAdjTime()** is not successful, *errno* usually indicates one of the following errors. Under some conditions, *errno* could indicate an error other than those listed here.

- [EINVAL] An invalid parameter was found.
- [EFAULT] A parameter passed to this function is not valid.
The address used for an argument is not correct.
In attempting to use an argument in a call, the system detected an address that is not valid.
While attempting to access a parameter passed to this function, the system detected an address that is not valid.
- [EPERM] Operation not permitted.
You must have appropriate privileges or be the owner of the object or other resource to do the requested operation.
- [EUNKNOWN] Unknown system state.
The operation failed because of an unknown system state. See any messages in the job log and correct any errors that are indicated, then retry the operation.

Error Messages

None.

Usage Notes

If the value of the environment variable `QIBM_USE_SFWCLK` is "N", `Qp0zAdjTime()` calls `adjtime()` to adjust the system clock.

Related Information

- The `<sys/time.h>` file (see "Header Files for UNIX-Type Functions" on page 17)
- "adjtime()—Adjust System Clock" on page 2
- "Qp0zGetTimeofDay()—Get Current Software Clock Time" on page 13
- "Qp0zSetTimeofDay()—Set Software Clock" on page 15

Example

See Code disclaimer information for information pertaining to code examples.

The following example initiates a software clock adjustment:

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

int main(int argc, char *argv[])
{
    struct timeval adj, old;
    int rc;

    /* Speed up the software clock by 1.5 seconds. */
    adj.tv_sec=1;
    adj.tv_usec=500000;

    rc=Qp0zAdjTime(&adj, &old);
    if(rc==0) {
        printf("Qp0zAdjTime() successful. "
              "Olddelta = %u.%06u\n",
              old.tv_sec, old.tv_usec);
    }
    else {
        printf("Qp0zAdjTime() failed, errno = %d\n",errno);
        return -1;
    }

    return 0;
}
```

Example Output:

```
Qp0zAdjTime() successful. Olddelta = 0.000000
```



API introduced: V5R3

[Top](#) | [UNIX-Type APIs](#) | [APIs by category](#)

Qp0zGetTimeofday()—Get Current Software Clock Time

Syntax

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

```
int Qp0zGetTimeofday (struct timeval *tp,  
                      struct timezone *tzp);
```

Service Program Name: QP0ZCPA

Default Public Authority: *USE

Threadsafe: Yes

The **Qp0zGetTimeofday()** function retrieves the current software clock time and places it in the `timeval` structure pointed to by `tp`. If `tzp` is not `NULL`, the time zone information is returned in the `timezone` structure pointed to by `tzp`.

The software clock maintains a time that can be set independently of the system clock. It is not integrated with the system and will be removed in a future release. The “`gettimeofday()—Get Current UTC Time`” on page 6 function should be used instead.

Parameters

- tp** (Output) A pointer to a `timeval` structure that contains the time in seconds and microseconds since 1 January 1970, 00:00:00 UTC (epoch-1970).
- tzp** (Output) A pointer to a `timezone` structure that contains the local time zone (measured in minutes west of Greenwich) and a flag that, if nonzero, indicates daylight saving time applies locally during the appropriate part of the year.

Authorities and Locks

None

Return Value

- 0 **Qp0zGetTimeofday()** was successful.
- 1 **Qp0zGetTimeofday()** was not successful. The `errno` variable is set to indicate the error.

Error Conditions

If **Qp0zGetTimeofday()** is not successful, `errno` usually indicates one of the following errors. Under some conditions, `errno` could indicate an error other than those listed here.

[EINVAL] An invalid parameter was found.

[EFAULT] A parameter passed to this function is not valid.
The address used for an argument is not correct.

In attempting to use an argument in a call, the system detected an address that is not valid.

While attempting to access a parameter passed to this function, the system detected an address that is not valid.

Error Messages

None.

Usage Notes

- For the best performance, specify NULL for the *tzp* parameter.
- If the value of the environment variable QIBM_USE_SFWCLK is "N", `Qp0zGetTimeofday()` calls `gettimeofday()` to get the current UTC time from the system clock.

Related Information

- The `<sys/time.h>` file (see "Header Files for UNIX-Type Functions" on page 17)
- "gettimeofday()—Get Current UTC Time" on page 6
- "Qp0zAdjTime()—Adjust Software Clock" on page 10
- "Qp0zSetTimeofday()—Set Software Clock" on page 15

Example

See Code disclaimer information for information pertaining to code examples.

The following example gets the current software clock time:

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

int main(int argc, char *argv[])
{
    struct timeval now;
    int rc;

    rc=Qp0zGetTimeofday(&now, NULL);
    if(rc==0) {
        printf("Qp0zGetTimeofday() successful.\n");
        printf("time = %u.%06u\n",
              now.tv_sec, now.tv_usec);
    }
    else {
        printf("Qp0zGetTimeofday() failed, errno = %d\n",
              errno);
        return -1;
    }

    return 0;
}
```

Example Output:

```
Qp0zGetTimeofday() successful.
time = 866208142.290944
```



API introduced: V5R3

Top | UNIX-Type APIs | APIs by category

Qp0zSetTimeofday()—Set Software Clock

Syntax

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

```
int Qp0zSetTimeofday (struct timeval *tp,  
                      struct timezone *tzp);
```

Service Program Name: QP0ZSETC

Default Public Authority: *USE

Threadsafe: Yes

The **Qp0zSetTimeofday()** function sets the software clock to the time contained in the `timeval` structure pointed to by `tp`. If `tzp` is not `NULL`, the time zone information is also set.

The software clock maintains a time that can be set independently of the system clock. It is not integrated with the system and will be removed in a future release. The “`settimeofday()—Set System Clock`” on page 8 function should be used instead.

Parameters

- tp** (Input) A pointer to a `timeval` structure that contains the time in seconds and microseconds since 1 January 1970, 00:00:00 UTC (epoch-1970).
- tzp** (Input) A pointer to a `timezone` structure that contains the local time zone (measured in minutes west of Greenwich) and a flag that, if nonzero, indicates daylight saving time applies locally during the appropriate part of the year.

Authorities and Locks

QSYS/QP0ZXCPA Service Program Authority
*USE

Return Value

- 0 **Qp0zSetTimeofday()** was successful.
- 1 **Qp0zSetTimeofday()** was not successful. The `errno` variable is set to indicate the error.

Error Conditions

If **Qp0zSetTimeofday()** is not successful, `errno` usually indicates one of the following errors. Under some conditions, `errno` could indicate an error other than those listed here.

[EINVAL] An invalid parameter was found.

[EFAULT] A parameter passed to this function is not valid.
The address used for an argument is not correct.

In attempting to use an argument in a call, the system detected an address that is not valid.

While attempting to access a parameter passed to this function, the system detected an address that is not valid.

[EPERM] Operation not permitted.

You must have appropriate privileges or be the owner of the object or other resource to do the requested operation.

[EUNKNOWN] Unknown system state.

The operation failed because of an unknown system state. See any messages in the job log and correct any errors that are indicated, then retry the operation.

Error Messages

None.

Usage Notes

If the value of the environment variable QIBM_USE_SFWCLK is "N", `Qp0zSetTimeofday()` calls `settimeofday()` to adjust the system clock.

Related Information

- The `<sys/time.h>` file (see "Header Files for UNIX-Type Functions" on page 17)
- "settimeofday()—Set System Clock" on page 8
- "Qp0zAdjTime()—Adjust Software Clock" on page 10
- "Qp0zGetTimeofday()—Get Current Software Clock Time" on page 13

Example

See Code disclaimer information for information pertaining to code examples.

The following example sets the software clock:

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

int main(int argc, char *argv[])
{
    struct timeval now;
    int rc;

    time.tv_sec=866208142;
    time.tv_usec=290944;

    rc=Qp0zSetTimeofday(&now, NULL);
    if(rc==0) {
        printf("Qp0zSetTimeofday() successful.\n");
    }
    else {
        printf("Qp0zSetTimeofday() failed, "
            "errno = %d\n",errno);
        return -1;
    }

    return 0;
}
```

Example Output

```
Qp0zSetTimeofday() successful.
```



API introduced: V5R3

[Top](#) | [UNIX-Type APIs](#) | [APIs by category](#)

Concepts

These are the concepts for this category.

Header Files for UNIX-Type Functions

Programs using the UNIX^(R)-type functions must include one or more header files that contain information needed by the functions, such as:

- Macro definitions
- Data type definitions
- Structure definitions
- Function prototypes

The header files are provided in the QSYSINC library, which is optionally installable. Make sure QSYSINC is on your system before compiling programs that use these header files. For information on installing the QSYSINC library, see [Include files](#) and the [QSYSINC Library](#).

The table below shows the file and member name in the QSYSINC library for each header file used by the UNIX-type APIs in this publication.

Name of Header File	Name of File in QSYSINC	Name of Member
arpa/inet.h	ARPA	INET
arpa/nameser.h	ARPA	NAMESER
bse.h	H	BSE
bsedos.h	H	BSEDOS
bseerr.h	H	BSEERR
dirent.h	H	DIRENT
errno.h	H	ERRNO
fcntl.h	H	FCNTL
grp.h	H	GRP
inttypes.h	H	INTTYPES
limits.h	H	LIMITS
mman.h	H	MMAN
netdbh.h	H	NETDB
netinet/icmp6.h	NETINET	ICMP6
net/if.h	NET	IF
netinet/in.h	NETINET	IN
netinet/ip_icmp.h	NETINET	IP_ICMP
netinet/ip.h	NETINET	IP

Name of Header File	Name of File in QSYSINC	Name of Member
netinet/ip6.h	NETINET	IP6
netinet/tcp.h	NETINET	TCP
netinet/udp.h	NETINET	UDP
netns/idp.h	NETNS	IDP
netns/ipx.h	NETNS	IPX
netns/ns.h	NETNS	NS
netns/sp.h	NETNS	SP
net/route.h	NET	ROUTE
nettel/tel.h	NETTEL	TEL
os2.h	H	OS2
os2def.h	H	OS2DEF
pwd.h	H	PWD
Qlg.h	H	QLG
» qp0lchsg.h	H	QP0LCHSG «
qp0lflop.h	H	QP0LFLOP
qp0ljrnl.h	H	QP0LJRNL
qp0lrord.h	H	QP0LROR
» qp0lrro.h	H	QP0LRRO «
» qp0lrtsg.h	H	QP0LRTSG «
» qp0lscan.h	H	QP0LSCAN «
Qp0lstdi.h	H	QP0LSTDI
qp0wpid.h	H	QP0WPID
qp0zdipc.h	H	QP0ZDIPC
qp0zipc.h	H	QP0ZIPC
qp0zolip.h	H	QP0ZOLIP
qp0zolsm.h	H	QP0ZOLSM
qp0zripc.h	H	QP0ZRIPC
qp0ztrc.h	H	QP0ZTRC
qp0ztrml.h	H	QP0ZTRML
qp0z1170.h	H	QP0Z1170
qsoasync.h	H	QSOASYNC
qtnxaapi.h	H	QTNXAAPI
qtnxadtp.h	H	QTNXADTP
qtomeapi.h	H	QTOMEAPI
qtossapi.h	H	QTOSSAPI
resolv.h	H	RESOLVE
semaphore.h	H	SEMAPHORE
signal.h	H	SIGNAL
spawn.h	H	SPAWN
ssl.h	H	SSL
sys/errno.h	H	ERRNO

Name of Header File	Name of File in QSYSINC	Name of Member
sys/ioctl.h	SYS	IOCTL
sys/ipc.h	SYS	IPC
sys/layout.h	H	LAYOUT
sys/limits.h	H	LIMITS
sys/msg.h	SYS	MSG
sys/param.h	SYS	PARAM
sys/resource.h	SYS	RESOURCE
sys/sem.h	SYS	SEM
sys/setjmp.h	SYS	SETJMP
sys/shm.h	SYS	SHM
sys/signal.h	SYS	SIGNAL
sys/socket.h	SYS	SOCKET
sys/stat.h	SYS	STAT
sys/statvfs.h	SYS	STATVFS
sys/time.h	SYS	TIME
sys/types.h	SYS	TYPES
sys/uio.h	SYS	UIO
sys/un.h	SYS	UN
sys/wait.h	SYS	WAIT
ulimit.h	H	ULIMIT
unistd.h	H	UNISTD
utime.h	H	UTIME

You can display a header file in QSYSINC by using one of the following methods:

- Using your editor. For example, to display the **unistd.h** header file using the Source Entry Utility editor, enter the following command:
STRSEU SRCFILE(QSYSINC/H) SRCMBR(UNISTD) OPTION(5)
- Using the Display Physical File Member command. For example, to display the **sys/stat.h** header file, enter the following command:
DSPPFM FILE(QSYSINC/SYS) MBR(STAT)

You can print a header file in QSYSINC by using one of the following methods:

- Using your editor. For example, to print the **unistd.h** header file using the Source Entry Utility editor, enter the following command:
STRSEU SRCFILE(QSYSINC/H) SRCMBR(UNISTD) OPTION(6)
- Using the Copy File command. For example, to print the **sys/stat.h** header file, enter the following command:
CPYF FROMFILE(QSYSINC/SYS) TOFILE(*PRINT) FROMMBR(STAT)

Symbolic links to these header files are also provided in directory /QIBM/include.

Errno Values for UNIX-Type Functions

Programs using the UNIX^(R)-type functions may receive error information as *errno* values. The possible values returned are listed here in ascending *errno* value sequence.

Name	Value	Text
EDOM	3001	A domain error occurred in a math function.
ERANGE	3002	A range error occurred.
ETRUNC	3003	Data was truncated on an input, output, or update operation.
ENOTOPEN	3004	File is not open.
ENOTREAD	3005	File is not opened for read operations.
EIO	3006	Input/output error.
ENODEV	3007	No such device.
ERECIO	3008	Cannot get single character for files opened for record I/O.
ENOTWRITE	3009	File is not opened for write operations.
ESTDIN	3010	The stdin stream cannot be opened.
ESTDOUT	3011	The stdout stream cannot be opened.
ESTDERR	3012	The stderr stream cannot be opened.
EBADSEEK	3013	The positioning parameter in fseek is not correct.
EBADNAME	3014	The object name specified is not correct.
EBADMODE	3015	The type variable specified on the open function is not correct.
EBADPOS	3017	The position specifier is not correct.
ENOPOS	3018	There is no record at the specified position.
ENUMMBRS	3019	Attempted to use ftell on multiple members.
ENUMRECS	3020	The current record position is too long for ftell.
EINVAL	3021	The value specified for the argument is not correct.
EBADFUNC	3022	Function parameter in the signal function is not set.
ENOENT	3025	No such path or directory.
ENOREC	3026	Record is not found.
EPERM	3027	The operation is not permitted.
EBADDATA	3028	Message data is not valid.
EBUSY	3029	Resource busy.
EBADOPT	3040	Option specified is not valid.
ENOTUPD	3041	File is not opened for update operations.
ENOTDLT	3042	File is not opened for delete operations.
EPAD	3043	The number of characters written is shorter than the expected record length.
EBADKEYLN	3044	A length that was not valid was specified for the key.
EPUTANDGET	3080	A read operation should not immediately follow a write operation.
EGETANDPUT	3081	A write operation should not immediately follow a read operation.
EIOERROR	3101	A nonrecoverable I/O error occurred.
EIORECERR	3102	A recoverable I/O error occurred.
EACCES	3401	Permission denied.

Name	Value	Text
ENOTDIR	3403	Not a directory.
ENOSPC	3404	No space is available.
EXDEV	3405	Improper link.
EAGAIN	3406	Operation would have caused the process to be suspended.
EWOULDBLOCK	3406	Operation would have caused the process to be suspended.
EINTR	3407	Interrupted function call.
EFAULT	3408	The address used for an argument was not correct.
ETIME	3409	Operation timed out.
ENXIO	3415	No such device or address.
EAPAR	3418	Possible APAR condition or hardware failure.
ERECURSE	3419	Recursive attempt rejected.
EADDRINUSE	3420	Address already in use.
EADDRNOTAVAIL	3421	Address is not available.
EAFNOSUPPORT	3422	The type of socket is not supported in this protocol family.
EALREADY	3423	Operation is already in progress.
ECONNABORTED	3424	Connection ended abnormally.
ECONNREFUSED	3425	A remote host refused an attempted connect operation.
ECONNRESET	3426	A connection with a remote socket was reset by that socket.
EDESTADDRREQ	3427	Operation requires destination address.
EHOSTDOWN	3428	A remote host is not available.
EHOSTUNREACH	3429	A route to the remote host is not available.
EINPROGRESS	3430	Operation in progress.
EISCONN	3431	A connection has already been established.
EMSGSIZE	3432	Message size is out of range.
ENETDOWN	3433	The network currently is not available.
ENETRESET	3434	A socket is connected to a host that is no longer available.
ENETUNREACH	3435	Cannot reach the destination network.
ENOBUFS	3436	There is not enough buffer space for the requested operation.
ENOPROTOPT	3437	The protocol does not support the specified option.
ENOTCONN	3438	Requested operation requires a connection.
ENOTSOCK	3439	The specified descriptor does not reference a socket.
ENOTSUP	3440	Operation is not supported.
EOPNOTSUPP	3440	Operation is not supported.
EPFNOSUPPORT	3441	The socket protocol family is not supported.
EPROTONOSUPPORT	3442	No protocol of the specified type and domain exists.
EPROTOTYPE	3443	The socket type or protocols are not compatible.
ERCVDERR	3444	An error indication was sent by the peer program.
ESHUTDOWN	3445	Cannot send data after a shutdown.
ESOCKTNOSUPPORT	3446	The specified socket type is not supported.
ETIMEDOUT	3447	A remote host did not respond within the timeout period.

Name	Value	Text
EUNATCH	3448	The protocol required to support the specified address family is not available at this time.
EBADF	3450	Descriptor is not valid.
EMFILE	3452	Too many open files for this process.
ENFILE	3453	Too many open files in the system.
EPIPE	3455	Broken pipe.
ECANCEL	3456	Operation cancelled.
EEXIST	3457	File exists.
EDEADLK	3459	Resource deadlock avoided.
ENOMEM	3460	Storage allocation request failed.
EOWNERTERM	3462	The synchronization object no longer exists because the owner is no longer running.
EDESTROYED	3463	The synchronization object was destroyed, or the object no longer exists.
ETERM	3464	Operation was terminated.
ENOENT1	3465	No such file or directory.
ENOEQFLOG	3466	Object is already linked to a dead directory.
EEMPTYDIR	3467	Directory is empty.
EMLINK	3468	Maximum link count for a file was exceeded.
ESPIPE	3469	Seek request is not supported for object.
ENOSYS	3470	Function not implemented.
EISDIR	3471	Specified target is a directory.
EROFS	3472	Read-only file system.
EUNKNOWN	3474	Unknown system state.
EITERBAD	3475	Iterator is not valid.
EITERSTE	3476	Iterator is in wrong state for operation.
EHRICLSBAD	3477	HRI class is not valid.
EHRICLBAD	3478	HRI subclass is not valid.
EHRITYPBAD	3479	HRI type is not valid.
ENOTAPPL	3480	Data requested is not applicable.
EHRIREQTYP	3481	HRI request type is not valid.
EHRINAMEBAD	3482	HRI resource name is not valid.
EDAMAGE	3484	A damaged object was encountered.
ELOOP	3485	A loop exists in the symbolic links.
ENAMETOOLONG	3486	A path name is too long.
ENOLCK	3487	No locks are available.
ENOTEMPTY	3488	Directory is not empty.
ENOSYSRSC	3489	System resources are not available.
ECONVERT	3490	Conversion error.
E2BIG	3491	Argument list is too long.
EILSEQ	3492	Conversion stopped due to input character that does not belong to the input codeset.

Name	Value	Text
ETYPE	3493	Object type mismatch.
EBADDIR	3494	Attempted to reference a directory that was not found or was destroyed.
EBADOBJ	3495	Attempted to reference an object that was not found, was destroyed, or was damaged.
EIDXINVAL	3496	Data space index used as a directory is not valid.
ESOFTDAMAGE	3497	Object has soft damage.
ENOTENROLL	3498	User is not enrolled in system distribution directory.
EOffline	3499	Object is suspended.
EROOBJ	3500	Object is a read-only object.
EEAHDDSI	3501	Hard damage on extended attribute data space index.
EEASDDSI	3502	Soft damage on extended attribute data space index.
EEAHDDS	3503	Hard damage on extended attribute data space.
EEASDDS	3504	Soft damage on extended attribute data space.
EEADUPRC	3505	Duplicate extended attribute record.
ELOCKED	3506	Area being read from or written to is locked.
EFBIG	3507	Object too large.
EIDRM	3509	The semaphore, shared memory, or message queue identifier is removed from the system.
ENOMSG	3510	The queue does not contain a message of the desired type and (msgflg logically ANDed with IPC_NOWAIT).
EFILECVT	3511	File ID conversion of a directory failed.
EBADFID	3512	A file ID could not be assigned when linking an object to a directory.
ESTALE	3513	File handle was rejected by server.
ESRCH	3515	No such process.
ENOTSIGINIT	3516	Process is not enabled for signals.
ECHILD	3517	No child process.
EBADH	3520	Handle is not valid.
ETOOMANYREFS	3523	The operation would have exceeded the maximum number of references allowed for a descriptor.
ENOTSAFE	3524	Function is not allowed.
EOverflow	3525	Object is too large to process.
EJRNDAMAGE	3526	Journal is damaged.
EJRNINACTIVE	3527	Journal is inactive.
EJRNRCVSPC	3528	Journal space or system storage error.
EJRNRMt	3529	Journal is remote.
ENEWJRNRCV	3530	New journal receiver is needed.
ENEWJRN	3531	New journal is needed.
EJOURNALED	3532	Object already journaled.
EJRNENTTOOLONG	3533	Entry is too large to send.
EDATALINK	3534	Object is a datalink object.

Name	Value	Text
ENOTAVAIL	3535	IASP is not available.
ENOTTY	3536	I/O control operation is not appropriate.
EFBIG2	3540	Attempt to write or truncate file past its sort file size limit.
ETXTBSY	3543	Text file busy.
EASPGRPNOTSET	3544	ASP group not set for thread.
ERESTART	3545	A system call was interrupted and may be restarted.
↳ ESCANFAILURE	3546	An object has been marked as a scan failure due to processing by an exit program associated with the scan-related integrated file system exit points. ⚡

Top | UNIX-Type APIs | APIs by category

Appendix. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Software Interoperability Coordinator, Department YBWA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, IBM License Agreement for Machine Code, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

Advanced 36
Advanced Function Printing
Advanced Peer-to-Peer Networking
AFP
AIX
AS/400
COBOL/400
CUA
DB2
DB2 Universal Database
Distributed Relational Database Architecture
Domino
DPI

DRDA
eServer
GDDM
IBM
Integrated Language Environment
Intelligent Printer Data Stream
IPDS
iSeries
Lotus Notes
MVS
Netfinity
Net.Data
NetView
Notes
OfficeVision
Operating System/2
Operating System/400
OS/2
OS/400
PartnerWorld
PowerPC
PrintManager
Print Services Facility
RISC System/6000
RPG/400
RS/6000
SAA
SecureWay
System/36
System/370
System/38
System/390
VisualAge
WebSphere
xSeries

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Terms and conditions for downloading and printing publications

Permissions for the use of the information you have selected for download are granted subject to the following terms and conditions and your indication of acceptance thereof.

Personal Use: You may reproduce this information for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative works of this information, or any portion thereof, without the express consent of IBM^(R).

Commercial Use: You may reproduce, distribute and display this information solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of this information, or reproduce, distribute or display this information or any portion thereof outside your enterprise, without the express consent of IBM.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the information or any data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the information is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations. IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THIS INFORMATION. THE INFORMATION IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

All material copyrighted by IBM Corporation.

By downloading or printing information from this site, you have indicated your agreement with these terms and conditions.

Code disclaimer information

This document contains programming examples.

SUBJECT TO ANY STATUTORY WARRANTIES WHICH CANNOT BE EXCLUDED, IBM^(R), ITS PROGRAM DEVELOPERS AND SUPPLIERS MAKE NO WARRANTIES OR CONDITIONS EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT, REGARDING THE PROGRAM OR TECHNICAL SUPPORT, IF ANY.

UNDER NO CIRCUMSTANCES IS IBM, ITS PROGRAM DEVELOPERS OR SUPPLIERS LIABLE FOR ANY OF THE FOLLOWING, EVEN IF INFORMED OF THEIR POSSIBILITY:

1. LOSS OF, OR DAMAGE TO, DATA;
2. SPECIAL, INCIDENTAL, OR INDIRECT DAMAGES, OR FOR ANY ECONOMIC CONSEQUENTIAL DAMAGES; OR
3. LOST PROFITS, BUSINESS, REVENUE, GOODWILL, OR ANTICIPATED SAVINGS.

SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO SOME OR ALL OF THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU.



Printed in USA