



System i
Programming
Date and Time APIs

Version 6 Release 1





System i
Programming
Date and Time APIs

Version 6 Release 1

Note

Before using this information and the product it supports, read the information in "Notices," on page 31.

This edition applies to version 6, release 1, modification 0 of IBM i5/OS (product number 5761-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, 2008. All rights reserved.

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

Contents

Date and Time APIs.	1
APIs	1
Adjust Time (QWCADJTM) API	1
Authorities and Locks	2
Required Parameter Group	2
Format of Adjustment Variable	2
ADJT0100	3
Field Descriptions	3
Error Messages	3
Convert Date and Time Format (QWCCVTDI) API	3
Authorities and Locks	4
Required Parameter Group	4
Optional Parameter Group 1	5
Optional Parameter Group 2	6
Input and Output Variable Formats.	6
16-Byte Character Date and Time Value Structure	7
17-Byte Character Date and Time Value Structure	7
19-Byte Character Date and Time Value Structure	7
20-Byte Character Date and Time Value Structure	7
DOSGetDateTime Value Structure	8
Time Zone Information Value Structure	8
Field Descriptions	9
Usage Notes	9
Error Messages	11
Qp0zCvtToMITime()-Convert Timeval Structure to _MI_Time	11
Parameters	12
Authorities and Locks	12
Return Value	12
Error Conditions.	12
Error Messages	13
Usage Notes	13
Related Information	13
Example	13
Qp0zCvtToTimeval()-Convert _MI_Time to Timeval Structure	14
Parameters	14
Authorities and Locks	15

Return Value	15
Error Conditions.	15
Error Messages	15
Usage Notes	16
Related Information	16
Example	16
Retrieve System Time Information (QWCRTVTM) API	17
Authorities and Locks	17
Required Parameter Group	17
Format of Receiver Variable	17
RTTM0100.	18
Field Descriptions	18
Valid Keys.	18
Key Field Descriptions.	19
Error Messages	19
Retrieve Time Zone Description (QWCRTVTZ) API	20
Authorities and Locks	20
Required Parameter Group	20
Format RTMZ0100	21
Format RTMZ0200	22
Field Descriptions	23
Error Messages	27
Set System Time (QWCSETTM) API	28
Authorities and Locks	28
Required Parameter Group	28
Input Variable Formats	28
20-Byte Character Date and Time Value Structure	28
Usage Notes	29
Error Messages	29
Code license and disclaimer information.	29

Appendix. Notices.	31
Programming interface information	32
Trademarks	33
Terms and conditions	34

Date and Time APIs

The date and time APIs provide support for working with date and time values. Using these APIs, you can do the following tasks:

- Access the current date and time.
- Adjust the current time of your system.
- Determine if a time adjustment is currently in progress.
- Convert date and time values from one format to another format.
- Retrieve information related to time zone descriptions.

The date and time APIs include:

- “Adjust Time (QWCADJTM) API” (QWCADJTM) adjusts the time-of-day clock.
- “Qp0zCvtToTimeval()-Convert `_MI_Time` to Timeval Structure” on page 14 (Qp0zCvtToTimeval()) converts a machine timestamp (or timestamp offset), represented by an `_MI_Time` data type, to a corresponding timeval structure.
- “Convert Date and Time Format (QWCCVTDT) API” on page 3 (QWCCVTDT) allows you to convert date and time formats from one format to another format.
- “Qp0zCvtToMTime()-Convert Timeval Structure to `_MI_Time`” on page 11 (Qp0zCvtToMTime()) converts a UNIX-type timestamp (or timestamp offset), represented by a timeval structure, to a corresponding `_MI_Time` data type.
- “Retrieve System Time Information (QWCRTVTM) API” on page 17 (QWCRTVTM) retrieves the current Coordinated Universal Time and time adjustment information.
- “Retrieve Time Zone Description (QWCRTVTZ) API” on page 20 (QWCRTVTZ) retrieves information about one or more time zone descriptions.
- “Set System Time (QWCSETTM) API” on page 28 (QWCSETTM) sets the Coordinated Universal Time (UTC) for the system.

In addition to the date and time APIs above, the following APIs also work with date and time values:

ILE CEE Date and Time APIs

ILE C/C++ Run-Time Library Functions

Machine Interface Instructions

UNIX-Type Time APIs

[Top](#) | [APIs by category](#)

APIs

These are the APIs for this category.

Adjust Time (QWCADJTM) API

Required Parameter Group:

1	Adjustment variable	Input	Char(*)
2	Length of adjustment variable	Input	Binary(4)

3	Adjustment format name	Input	Char(8)
4	Error Code	I/O	Char(*)

Default Public Authority: *USE
 Threadsafe: YES

The Adjust Time (QWCADJTM) API will adjust the time by increasing or decreasing the time-of-day clock to a maximum change of plus or minus two hours.

Note: A time-of-day adjustment will remain active until completed unless one of the following occurs:

- A new time-of-day clock adjustment is started for the system.
- The time-of-day clock for the system is changed.
- The system is powered off.

The Retrieve System Time Information(QWCRTVTM) API can be used to determine if an adjustment to the time-of-day clock is allowed and can be used to retrieve information on any time adjustment that may be currently active.

Authorities and Locks

User Special Authority
 *ALLOBJ

Required Parameter Group

Adjustment variable

INPUT; CHAR(*)

The variable that is used to specify the time adjustment details.

Length of adjustment variable

INPUT; BINARY(4)

The length of the adjustment variable. Minimum length is 9 bytes.

Adjustment format name

INPUT; CHAR(8)

The format name of the adjustment variable. The possible format name is:

ADJT0100

Basic time adjustment details.

See "Format of Adjustment Variable" for more information.

Error code

I/O; CHAR(*)

The structure in which to return error information. For the format of the structure, see Error code parameter.

Format of Adjustment Variable

The following table describes the order and format of the data that is specified in the adjustment variable. For detailed descriptions of the fields in this table, see "Field Descriptions" on page 3.

ADJT0100

Offset		Type	Field
Dec	Hex		
0	0	BINARY(8), UNSIGNED	Time adjustment amount
8	8	CHAR(1)	Time adjustment direction

Field Descriptions

Time adjustment amount. The time value which specifies the amount of time in microseconds by which the time-of-day clock will be increased or decreased.

Note: The maximum value for the time adjustment interval is two hours.

Time adjustment direction. The direction of the time-of-day clock adjustment for the system.

0 Increase time of day.
1 Decrease time of day.

Error Messages

The following messages may be sent from this function:

Message ID	Error Message Text
CPF1890 E	*ALLOBJ authority required for requested operation.
CPF18C5 E	Time adjustment not valid.
CPF24B4 E	Severe error while addressing parameter list.
CPF3C12 E	Length of data is not valid.
CPF3C21 E	Format name &1 is not valid.
CPF3C36 E	Number of parameters, &1, entered for this API was not valid.
CPF3C3C E	Value for parameter &1 not valid.
CPF3CF1 E	Error code parameter not valid.
CPF3CF2 E	Error(s) occurred during running of &1 API.
CPF9821 E	Not authorized to program &1 in library &2.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.

API introduced: V5R3

[Top](#) | ["Date and Time APIs," on page 1](#) | [APIs by category](#)

Convert Date and Time Format (QWCCVTDT) API

Required Parameter Group:

1	Input format	Input	Char(10)
2	Input variable	Input	Char(*)
3	Output format	Input	Char(10)
4	Output variable	Output	Char(*)
5	Error code	I/O	Char(*)

Optional Parameter Group 1:

6	Input time zone	Input	Char(10)
7	Output time zone	Input	Char(10)
8	Time zone information	Output	Char(*)
9	Length of time zone information	Input	Bin(4)
10	Precision indicator	Input	Char(1)

Optional Parameter Group 2:

11	Input time indicator	Input	Char(1)
----	----------------------	-------	---------

Default Public Authority: *USE
 Threadsafe: Yes

The Convert Date and Time Format (QWCCVTDT) API converts date and time values from one format to another format. The QWCCVTDT API lets you:

- Convert a time-stamp (*DTS, for system time-stamp) value to character format
- Convert a character date and time value to time-stamp format
- Convert a date from one character format to another
- Convert a date and time based on input and output time zone values and return the time zone information that is associated with the converted output
- Specify a precision of milliseconds or microseconds for your input and output variables
- Retrieve a current clock time based on the output time zone and return it based on the output format you specify

For additional information on converting dates and times, see “Usage Notes” on page 9.

Authorities and Locks

None.

Required Parameter Group

Input format

INPUT; CHAR(10)

The format of the data you give QWCCVTDT to convert. Valid values are:

*CURRENT	The current system time.
*DTS	System time-stamp.
*JOB	The format given in the DATFMT job attribute.
*SYSVAL	The format given in the QDATFMT system value.
*YMD	YYMMDD (year, month, day) format.
*YYMD	YYYYMMDD (4-digit year, month, day) format.
*MDY	MMDDYY (month, day, year) format.
*MDYY	MMDDYYYY (month, day, 4-digit year) format.
*DMY	DDMMYY (day, month, year) format.
*DMYY	DDMMYYYY (day, month, 4-digit year) format.
*JUL	Julian format (YYDDD (year, day of year)).
*LONGJUL	Long Julian format (YYYYDDD (4-digit year, day of year)).

Input variable

INPUT; CHAR(*)

The data to be converted. If the input format is *CURRENT, then this parameter is not used. See “Input and Output Variable Formats” on page 6 to determine the structure of the input variable for all other input formats.

Output format

INPUT; CHAR(10)

The format to convert the data to. Valid values are:

<i>*DTS</i>	System time-stamp.
<i>*JOB</i>	The format given in the DATFMT job attribute
<i>*SYSVAL</i>	The format given in the QDATFMT system value
<i>*YMD</i>	YYMMDD format
<i>*YYMD</i>	YYYYMMDD format
<i>*MDY</i>	MMDDYY format
<i>*MDYY</i>	MMDDYYYY format
<i>*DMY</i>	DDMMYY format
<i>*DMYY</i>	DDMMYYYY format
<i>*JUL</i>	Julian format (YYDDD)
<i>*LONGJUL</i>	Long Julian format (YYYYDDD)
<i>*DOS</i>	DOSGetDateTime format. The <i>*DOS</i> value can be specified only when <i>*CURRENT</i> or <i>*DTS</i> is specified for the input format parameter.

Output variable

OUTPUT; CHAR(*)

The converted data. If the output format is **DOS*, the first 11 characters of this parameter are used. For details, see “DOSGetDateTime Value Structure” on page 8. See “Input and Output Variable Formats” on page 6 to determine the structure of the output variable for all other output formats.

Error code

I/O; CHAR(*)



The structure in which to return error information. For the format of the structure, see Error code parameter.

Optional Parameter Group 1

Input time zone

INPUT; CHAR(10)



Specifies the time zone associated with the input variable. If the input format is **CURRENT*, then this parameter is not used. The default value is **SYS*. Valid values are:

<i>*SYS</i>	The input variable is a local system time value and the associated time zone is specified by the time zone system value.
<i>*UTC</i>	The input variable is a  Gregorian  Coordinated Universal Time (UTC) value.
<i>*JOB</i>	The input variable is a local job time value and the associated time zone is specified by the time zone job attribute.
<i>Time zone name</i>	Specifies the name of a time zone description (<i>*TIMZON</i>) object.

Output time zone

INPUT; CHAR(10)

Specifies the time zone associated with the output variable. The default value is **SYS*. Valid values are:

<i>*SYS</i>	The output variable is a local system time value and the associated time zone is specified by the time zone system value.
<i>*UTC</i>	The output variable is a  Gregorian  Coordinated Universal Time (UTC) value.
<i>*JOB</i>	The output variable is a local job time value and the associated time zone is specified by the time zone job attribute.

Time zone name Specifies the name of a time zone description (*TIMZON) object.

Time zone information

OUTPUT; CHAR(*)

Specifies the time zone information associated with the output time zone. If 0 is specified for the length of time zone information, then this parameter is not used. For the format of the structure, see “Time Zone Information Value Structure” on page 8.

Length of time zone information

INPUT; BIN(4)

Specifies the length of the time zone information to be returned. The minimum length is 0 which indicates to not return any time zone information.

Precision indicator

INPUT; CHAR(1)

Specifies the precision of the input and output variables. The default value is 0 or milliseconds. Valid values are:

- 0 The input and output variables will have a precision in milliseconds.
- 1 The input and output variables will have a precision in microseconds.

Optional Parameter Group 2

Input time indicator

INPUT; CHAR(1)

Specifies which segment of time to use when the input variable has a date and time value that matches a repeated time. Otherwise, this parameter is not used. Repeated times occur when time changes from Daylight Saving Time (DST) to Standard Time (ST). For example, if DST ends on a given day at 02:00AM, then the segment of time from 01:00:00.000000 to 01:59:59.999999 on that day repeats. The first segment of time is considered in DST and the second segment is considered in ST. The default value is 1 or use the DST segment. For additional information on this parameter, see “Usage Notes” on page 9.

- 0 The input variable contains a date and time value that is contained in the second or Standard Time segment.
- 1 The input variable contains a date and time value that is contained in the first or Daylight Saving Time segment.

Input and Output Variable Formats

This table shows the format used for the input or output variable parameters.

Input or Output Format	Input or Output Variable
*DTS	System time-stamp. The first 8 characters are used.
*YYMD, *MDYY, *DMYY, *LONGJUL in milliseconds	The first 17 characters are used. See “17-Byte Character Date and Time Value Structure” on page 7.
All other character formats in milliseconds	The first 16 characters are used. See “16-Byte Character Date and Time Value Structure” on page 7.
*YYMD, *MDYY, *DMYY, *LONGJUL in microseconds	The first 20 characters are used. See “20-Byte Character Date and Time Value Structure” on page 7.
All other character formats in microseconds	The first 19 characters are used. See “19-Byte Character Date and Time Value Structure” on page 7.

16-Byte Character Date and Time Value Structure

This table shows the structure used for the input and output variables when the format is *JOB, *SYSVAL, *YMD, *MDY, *DMY, and *JUL and the precision indicator specifies milliseconds.

Offset	Description
0	Century. Possible values are 0, which indicates years 19xx, 1, which indicates years 20xx and so forth through 9, which indicates years 28xx.
1-6	Date, left-justified. This value cannot be all blanks or all zeros. Left-justify Julian dates, using blanks to fill the space.
7-12	Time, in HHMMSS (hours, minutes, seconds) format.
13-15	Milliseconds. This value cannot be blanks.

17-Byte Character Date and Time Value Structure

This table shows the structure used for the input and output variables when the format is *YYMD, *MDYY, *DMYY, and *LONGJUL and the precision indicator specifies milliseconds.

Offset	Description
0-7	Date, left-justified. This value cannot be all blanks or all zeros. Left-justify Julian dates, using blanks to fill the space.
8-13	Time, in HHMMSS (hours, minutes, seconds) format.
14-16	Milliseconds. This value cannot be blanks.

19-Byte Character Date and Time Value Structure

This table shows the structure used for the input and output variables when the format is *JOB, *SYSVAL, *YMD, *MDY, *DMY, and *JUL and the precision indicator specifies microseconds.

Offset	Description
0	Century. Possible values are 0, which indicates years 19xx, 1, which indicates years 20xx and so forth through 9, which indicates years 28xx.
1-6	Date, left-justified. This value cannot be all blanks or all zeros. Left-justify Julian dates, using blanks to fill the space.
7-12	Time, in HHMMSS (hours, minutes, seconds) format.
13-18	Microseconds. This value cannot be blanks.

20-Byte Character Date and Time Value Structure

This table shows the structure used for the input and output variables when the format is *YYMD, *MDYY, *DMYY, and *LONGJUL and the precision indicator specifies microseconds..

Offset	Description
0-7	Date, left-justified. This value cannot be all blanks or all zeros. Left-justify Julian dates, using blanks to fill the space.
8-13	Time, in HHMMSS (hours, minutes, seconds) format.
14-19	Microseconds. This value cannot be blanks.

DOSGetDateTime Value Structure

This table shows the structure used for the output variable.

Offset	Description
0	Hours (0-23) ¹
1	Minutes (0-59) ¹
2	Seconds (0-59) ¹
3	Hundredths of seconds (0-99) ¹
4	Day (1-31) ¹
5	Month (1-12) ¹
6-7	Year (for example, 1995) ²
8-9	Time zone offset (in minutes) ^{2, 3}
10	Day of the week, where 0 is Sunday (0-6) ¹
Notes:	
¹	A 1-byte integer.
²	A 2-byte integer.
³	This is the negative value of the offset associated with the specified output time zone. If *UTC is specified for the output time zone, then this value will be 0. If an output time zone is not specified, then this is the negative value of the system value QUTCOFFSET.

Time Zone Information Value Structure

This table shows the structure used for the time zone information output parameter. If *UTC is specified for the output time zone, or if the input and output time zone parameter values are the same and the input variable contains a date that is outside the supported date range (from August 25, 1928, 00:00:00.000000 to May 09, 2071, 00:00:00.000000), then all binary fields will be set to 0 and all character fields will be set to blanks.

Offset		Type	Field
Dec	Hex		
0	0	BINARY(4)	Bytes returned
4	4	BINARY(4)	Bytes available
8	8	CHAR(10)	Time zone description name
18	12	CHAR(1)	Reserved
19	13	CHAR(1)	Current Daylight Saving Time indicator
20	14	BINARY(4)	Current offset
24	18	CHAR(50)	Current full name
74	4A	CHAR(10)	Current abbreviated name
84	54	CHAR(7)	Current message identifier
91	5B	CHAR(10)	Message file name
101	65	CHAR(10)	Message file library
➤ 111	6F	CHAR(1)	Reserved
112	70	BINARY(4)	Year offset ⏪

Field Descriptions

Bytes available. The number of bytes of data available to be returned. All available data is returned if enough space is provided.

Bytes returned. The number of bytes of data returned.

Current abbreviated name. The abbreviated, or short, name for the time zone. This field will contain either the Standard Time or Daylight Saving Time abbreviated name depending on whether or not Daylight Saving Time is in effect. If the time zone description uses a message to specify the current abbreviated name and the message cannot be retrieved, this field returns *N. This can occur when the caller of the API is not authorized to the message file or its library, the message file cannot be found or the message does not exist in the message file.

Current Daylight Saving Time indicator. Indicates whether or not the output date and time (output variable converted based on the output time zone) is observing Daylight Saving Time or not. Valid values that are returned are:

- 0 Daylight Saving Time is not being observed (Standard Time).
- 1 Daylight Saving Time is being observed.

Current full name. The full, or long, name for the time zone. This field will contain either the Standard Time or Daylight Saving Time full name depending on whether or not Daylight Saving Time is in effect. If the time zone description uses a message to specify the current full name and the message cannot be retrieved, this field returns *N. This can occur when the caller of the API is not authorized to the message file or its library, the message file cannot be found or the message does not exist in the message file.

Current message identifier. The identifier of the message that contains the current full and abbreviated names. This field will be *NONE if a message was not specified when the time zone description was created.

Current offset. The time difference, in minutes, between the output time zone and Coordinated Universal Time (UTC). This value has been adjusted for Daylight Saving Time, if necessary.

Message file library. The name of the library that contains the message file. The field will contain all blanks if the current message identifier is *NONE.

Message file name. The name of the file that contains the current message. The field will contain *NONE if the current message identifier is *NONE.

Reserved. An unused field.

Time zone description name. The name of the time zone description that is associated with the output time zone. If *SYS or *JOB was specified for the output time zone and a time zone has not been set for the Time zone (QTIMZON) system value, this field returns *N.

➤ **Year offset.** The number of years that the current year in the calendar system used with this time zone differs from the current Gregorian year. The range is -140 to 140. ⚡

Usage Notes

When converting an input date from a 2-digit year format to a *DTS time-stamp format without time zone conversion, the supported date range is from August 23, 1928, 12:03:06.314752 (.315 for milliseconds) to May 10, 2071, 11:56:53.685240 (.685 for milliseconds). Converting an input date that is outside this range will result in an output date within this range.

When converting an input date from a 4-digit year format to a *DTS time-stamp format without time zone conversion, the supported date range is from August 24, 1928, 00:00:00.000000 to May 09, 2071, 23:59:59.999999 (.999 for milliseconds). Converting an input date that is outside this range will result in error message CPF1060.

When converting an input date from a 4-digit year format to a 2-digit year format without time zone conversion, the supported date range is from January 1, 1900, 00:00:00.000000 to December 31, 2899, 23:59:59.999999 (.999 for milliseconds). Converting an input date that is outside this range will result in error message CPF1060.

When converting an input date from a 4-digit year format to a 4-digit year format without time zone conversion, the supported date range is from January 1, 0001, 00:00:00.000000 to December 31, 9999, 23:59:59.999999 (.999 for milliseconds). Converting an input date that is outside this range will result in error message CPF1060.

When converting an input date from 2-digit year format to a 2-digit year format without time zone conversion, the supported date range is from January 1, 1900, 00:00:00.000000 to December 31, 2899, 23:59:59.999999 (.999 for milliseconds). The century digit of the input variable is copied into the output variable without validation.

When converting an input date from 2-digit year format to a 4-digit year format without time zone conversion, the supported date range is from January 1, 1900, 00:00:00.000000 to December 31, 2899, 23:59:59.999999 (.999 for milliseconds).

When converting an input date from a *DTS time-stamp format to an output date of any format without time zone conversion, the supported date range is from August 23, 1928, 12:03:06.314752 (.315 for milliseconds) to May 10, 2071, 11:56:53.685240 (.685 for milliseconds).

When converting an input date of any format to an output date of any format that involves time zone conversion as well, the supported date range is from August 25, 1928, 00:00:00.000000 to May 08, 2071, 23:59:59.999999 (.999 for milliseconds). Converting an input date that is outside this range will result in error message CPF1060.

When moving from Standard Time (ST) to Daylight Saving Time (DST) there is a window of time (1 hour) that does not occur. Any time zone conversion where the input variable date and time value is within this window will result in error message CPF1060.

When moving from Daylight Saving Time (DST) to Standard Time there is a window of time (1 hour) that repeats. For example, if DST ends on a given day at 02:00AM, then the segment of time from 01:00:00.000000 to 01:59:59.999999 on that day repeats. The first segment of repeated time is the DST segment. The second segment of repeated time is the Standard Time segment. It is possible using time zone conversion to have the output variable date and time value end up in either segment. If you are retrieving time zone information, the current Daylight Saving Time indicator will be set accordingly. By default, for any time zone conversion the input variable that is within this window of time that repeats is considered part of the DST segment. However, you can use the optional Input time indicator parameter to cause the input variable to be considered within the Standard Time segment. You can copy the resultant current DST indicator into the Input time indicator parameter when converting back and forth between time zones. For example, when converting a date and time value from *UTC to time zone A, the resultant time is 01:15:00 AM and the current DST indicator returned is 0, which means the resultant time is Standard Time. In order to obtain the original *UTC value when converting back to *UTC from time zone A, the current DST indicator value should be copied to the Input time indicator parameter. This will cause the date and time value to be treated as Standard Time rather than as the default, Daylight Saving Time.

You can convert any input format except *CURRENT to the same output format without receiving an error (time zone conversion is not specified, or if specified, the input and output time zone parameter

values must be the same and the time zone information length must be 0 as well). For these cases, the input variable is copied into the output variable without validation.

When converting one character date format (that is, anything other than *CURRENT, the current machine-clock time, *DTS, the system time-stamp, or any specified time zone conversion) to another character date format, the date information is validated and converted. However, the time portion of the input variable is copied into the output variable without validation.

When requesting time zone conversion with different input and output time zone values, or when requesting time zone information, the time portion is validated and converted as well as the date portion.

When converting a character date and time value to *DTS and back to character format using microseconds precision, there is a rounding error of minus 1 to minus 7 microseconds. If you specify a precision of microseconds, it is recommended that you use a microsecond value that is evenly divisible by 8.

» The system has the ability to manage a year offset that specifies the number of years that the current year in the calendar system used with this time zone differs from the current Gregorian year. When the input time format is *CURRENT, the date returned will be in the non-Gregorian year which takes the year offset into account. When specifying an input time zone value that is not *UTC and an output time zone value of *UTC, the year offset is removed along with the time zone offset. The converted output will be in the UTC time zone (based on the Greenwich meridian) and will be in the Gregorian year. To get a result that includes the year offset, specify an output time zone that defines Greenwich Mean Time such as the IBM-supplied Q0000GMT and that also contains the desired year offset. The converted output will be in the GMT time zone (same as UTC time zone) and will also be in the non-Gregorian year. «

Error Messages

Message ID	Error Message Text
CPF1060 E	Date not valid.
CPF1061 E	Time not valid.
CPF1848 E	Century digit &1 not valid.
CPF1849 E	Millisecond or microsecond value not valid.
CPF1850 E	Format &1 not valid
CPF24B4 E	Severe error while addressing parameter list.
CPF3C36 E	Number of parameters, &1, for API not valid.
CPF3C3C E	Value for parameter &1 not valid.
CPF3C90 E	Literal value cannot be changed.
CPF3CF1 E	Error code parameter not valid.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.

API introduced: V2R1

[Top](#) | [“Date and Time APIs,” on page 1](#) | [Miscellaneous APIs](#) | [APIs by category](#)

Qp0zCvtToMITime()-Convert Timeval Structure to _MI_Time

Syntax:

```
#include <qp0z1170.h>

int Qp0zCvtToMITime (_MI_Time to,
                    const struct timeval *from,
                    int option);
```

Service Program Name: QP0ZCPA
Default Public Authority: *USE
Threadsafe: Yes

The `Qp0zCvtToMITime()` function converts a UNIX[®]-type timestamp (or a timestamp offset), represented by a `timeval` structure, to a corresponding `_MI_Time` data type. The job's time zone offset from UTC and epoch-1970 are optionally taken into account by this conversion. Only timestamps or timestamp offsets in the following ranges can be converted:

- Timestamps that are later than or equal to 1 January 1970, 00:00:00 UTC (epoch-1970) and less than 19 January 2038, 03:14:08 UTC.
- Timestamp offsets that are greater than or equal to 0 and less than 2,147,483,648 seconds.

Note: This function uses a header (include) file from the library `QSYSINC`, which is optionally installable. Make sure `QSYSINC` is installed on your system before using this function. See Header Files for UNIX-Type Functions) for the file and member name of each header file.

Parameters

to (Output) The `_MI_Time` data type to contain the converted timestamp (or timestamp offset).

from (Input) The address of the `timeval` structure to be converted.

option (Input) The conversion option.

The **option** parameter must be one of the following constants:

`QP0Z_CVTTIME_TO_OFFSET`

Do the conversion as a timestamp offset, not factoring in the UTC offset from the current time zone of the job or epoch-1970.

`QP0Z_CVTTIME_TO_TIMESTAMP`

Do the conversion as a timestamp, factoring in the UTC offset from the current time zone of the job and epoch-1970.

`QP0Z_CVTTIME_FACTOR_EPOCH_ONLY`

Do the conversion as a timestamp, but factor in epoch-1970 only.

`QP0Z_CVTTIME_FACTOR_UTCOFFSET_ONLY`

Do the conversion as a timestamp, but factor in the UTC offset from the current time zone of the job only.

Authorities and Locks

None.

Return Value

- 0 `Qp0zCvtToMITime()` was successful. The value referenced by the **to** parameter is the converted timestamp (or timestamp offset).
- 1 `Qp0zCvtToMITime()` was not successful. The `errno` variable is set to indicate the error.

Error Conditions

If `Qp0zCvtToMITime()` 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]

The value specified for the argument is not correct.

A function was passed incorrect argument values, or an operation was attempted on an object and the operation specified is not supported for that type of object.

An argument value is not valid, out of range, or NULL.

[EFAULT]

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.

[ERANGE]

A range error occurred.

The value of an argument is too small, or a result too large.

[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

1. `Qp0zCvtToMITime()`, when called with **option** equal to `QP0Z_CVTTIME_TO_OFFSET`, will convert the number of seconds and microseconds given in the **from** parameter to an equivalent machine timestamp offset, similar to what the `mitime()` API does.
2. `Qp0zCvtToMITime()`, when called with **option** equal to `QP0Z_CVTTIME_TO_TIMESTAMP`, will convert the number of seconds and microseconds given in the **from** parameter to an equivalent machine timestamp.

Related Information

- The `<qp0z1170.h>` file (see Header Files for UNIX-Type Functions)
- “`Qp0zCvtToTimeval()-Convert _MI_Time to Timeval Structure`” on page 14

Example

The following example converts a timestamp.

Note: By using the code examples, you agree to the terms of the “Code license and disclaimer information” on page 29.

```
#include <qp0z1170.h>
#include <stdio.h>
#include <errno.h>

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

    tv.tv_sec=867422292;
    tv.tv_usec=52992;

    printf("timeval timestamp: %u.%06u\n",
           tv.tv_sec, tv.tv_usec);
    rc = Qp0zCvtToMITime(mt, &tv,
```

```

        QP0Z_CVTTIME_TO_TIMESTAMP);

if(rc==0) {
    printf("mi timestamp: %08X%08X\n",
        *((unsigned *)&mt[0]),
        *((unsigned *)&mt[4]));
}
else {
    printf("Qp0zCvtToMITime() failed, errno = %d\n",
        errno);
    return -1;
}

return 0;
}

```

Example Output:

```

timeval timestamp: 867422292.052992
mi timestamp: 7B7E9425EAC00000

```

API introduced: V4R2

Top | "Date and Time APIs," on page 1 | APIs by category

Qp0zCvtToTimeval()-Convert `_MI_Time` to Timeval Structure

Syntax:

```

#include <qp0z1170.h>

int Qp0zCvtToTimeval (struct timeval *to,
                    const _MI_Time from,
                    int option);

```

Service Program Name: QP0ZCPA
 Default Public Authority: *USE
 Threadsafe: Yes

The `Qp0zCvtToTimeval()` function converts a machine timestamp (or a machine timestamp offset), represented by an `_MI_Time` data type, to a corresponding structure `timeval` value. The job's time zone offset from UTC and epoch-1970 are optionally taken into account by this conversion. Only timestamps or timestamp offsets in the following ranges can be converted:

- Timestamps that are later than or equal to 1 January 1970, 00:00:00 UTC (epoch-1970) and less than 19 January 2038, 03:14:08 UTC.
- Timestamp offsets that are greater than or equal to 0 and less than 2,147,483,648 seconds.

Note: This function uses a header (include) file from the library `QSYSINC`, which is optionally installable. Make sure `QSYSINC` is installed on your system before using this function. See Header Files for UNIX®-Type Functions) for the file and member name of each header file.

Parameters

- to** (Output) The address of the `timeval` structure to contain the converted timestamp (or timestamp offset).
- from** (Input) The `_MI_Time` data type to be converted.
- option** (Input) The conversion option.

The **option** parameter must be one of the following constants:

QP0Z_CVTTIME_TO_OFFSET

Do the conversion as a timestamp offset, not factoring in the UTC offset from the current time zone of the job or epoch-1970.

QP0Z_CVTTIME_TO_TIMESTAMP

Do the conversion as a timestamp, factoring in the UTC offset from the current time zone of the job and epoch-1970.

QP0Z_CVTTIME_FACTOR_EPOCH_ONLY

Do the conversion as a timestamp, but factor in epoch-1970 only.

QP0Z_CVTTIME_FACTOR_UTCOFFSET_ONLY

Do the conversion as a timestamp, but factor in the UTC offset from the current time zone of the job only.

Authorities and Locks

None.

Return Value

- 0 `Qp0zCvtToTimeval()` was successful. The value referenced by the `to` parameter is the converted timestamp (or timestamp offset).
- 1 `Qp0zCvtToTimeval()` was not successful. The `errno` variable is set to indicate the error.

Error Conditions

If `Qp0zCvtToTimeval()` 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]

The value specified for the argument is not correct.

A function was passed incorrect argument values, or an operation was attempted on an object and the operation specified is not supported for that type of object.

An argument value is not valid, out of range, or NULL.

[EFAULT]

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.

[ERANGE]

A range error occurred.

The value of an argument is too small, or a result too large.

[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

1. `Qp0zCvtToTimeval()`, when called with **option** equal to `QP0Z_CVTTIME_TO_OFFSET`, will convert the machine timestamp offset given in the **from** parameter to an equivalent number of seconds and microseconds. This could be used to calculate a time delay.
2. `Qp0zCvtToTimeval()`, when called with **option** equal to `QP0Z_CVTTIME_TO_TIMESTAMP`, will convert the machine timestamp given in the **from** parameter to an equivalent number of seconds and microseconds. This could be used as a UNIX-type timestamp.

Related Information

- The `<qp0z1170.h>` file (see Header Files for UNIX-Type Functions)
- “`Qp0zCvtToMTime()-Convert Timeval Structure to _MI_Time`” on page 11

Example

The following example converts a timestamp.

Note: By using the code examples, you agree to the terms of the “Code license and disclaimer information” on page 29.

```
#include <qp0z1170.h>
#include <mimchint.h>
#include <stdio.h>
#include <errno.h>

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

    mattod(mt);

    printf("mi timestamp: %08X%08X\n",
           *((unsigned *)&mt[0]),
           *((unsigned *)&mt[4]));

    rc = Qp0zCvtToTimeval(&tv, mt, QP0Z_CVTTIME_TO_TIMESTAMP);

    if(rc==0) {
        printf("timeval timestamp: %u.%06u\n",
              tv.tv_sec, tv.tv_usec);
    }
    else {
        printf("Qp0zCvtToTimeval() failed, errno = %d\n",
              errno);
        return -1;
    }

    return 0;
}
```

Example Output:

```
mi timestamp: 7B7E9425EAC00000
timeval timestamp: 867422292.052992
```

API introduced: V4R2

Top | “Date and Time APIs,” on page 1 | APIs by category

Retrieve System Time Information (QWCRTVTM) API

Required Parameter Group:

1	Receiver variable	Output	Char(*)
2	Length of receiver variable	Input	Binary(4)
3	Format name	Input	Char(8)
4	Number of fields to return	Input	Binary(4)
5	Key of fields to return	Input	Array(*) of Binary(4)
6	Error Code	I/O	Char(*)

Default Public Authority: *USE
Threadsafe: Yes

The Retrieve System Time Information (QWCRTVTM) API retrieves the current Coordinated Universal Time and time adjustment information.

Authorities and Locks

None

Required Parameter Group

Receiver variable

OUTPUT; CHAR(*)

The variable that is used to return the time information.

Length of receiver variable

INPUT; BINARY(4)

The length of the receiver variable. This must be a minimum of 8 bytes.

Format name

INPUT; CHAR(8)

The format of the information to be returned in the receiver variable. The possible format name is:

RTTM0100 Basic time information. See "Format of Receiver Variable" for more information.

Number of fields to return

INPUT; BINARY(4)

The number of fields to return in the specified format.

Key of fields to return

INPUT; ARRAY(*) of BINARY(4)

The list of fields to be returned in the specified format. For a list of valid fields, see "Valid Keys" on page 18.

Error code

I/O; CHAR(*)

The structure in which to return error information. For the format of the structure, see Error code parameter.

Format of Receiver Variable

The following table describes the order and format of the data that is returned in the receiver variable. For detailed descriptions of the fields in this table, see "Field Descriptions" on page 18.

RTTM0100

Offset		Type	Field
Dec	Hex		
0	0	BINARY(4)	Bytes returned
4	4	BINARY(4)	Bytes available
8	8	BINARY(4)	Offset to key fields
12	C	BINARY(4)	Number of fields returned
16	10	CHAR(*)	Reserved
These fields repeat, in the order listed, for the number of key fields returned.		BINARY(4)	Length of field information returned
		BINARY(4)	Key field
		CHAR(1)	Type of data
		CHAR(3)	Reserved
		BINARY(4)	Length of data
		CHAR(*)	Data
		CHAR(*)	Reserved

Field Descriptions

Bytes available. The number of bytes of data available to be returned. All available data is returned if enough space is provided.

Bytes returned. The number of bytes of data returned.

Data. The data returned for the key field.

Key field. The field returned. See “Valid Keys” for the list of valid keys.

Length of data. The length of the data returned for the field.

Length of field information returned. The total length of information returned for this field. This value is used to increment to the next field in the list.

Number of fields returned. The number of fields returned to the application.

Offset to key fields. The offset in bytes from the beginning of the receiver to the key fields array entry.

Reserved. An unused field.

Type of data. The type of data returned.

C The data is returned in character format.

B The data is returned in binary format.

Valid Keys

The following indicates the valid keys for the formats specified. See “Key Field Descriptions” on page 19 for the descriptions of the valid key attributes.

Key	Type	Description
101	CHAR(8)	Coordinated Universal Time

Key	Type	Description
201	CHAR(1)	Time adjustment status
202	CHAR(1)	Time adjustment direction
203	BINARY(8), UNSIGNED	Time adjustment amount
204	BINARY(8), UNSIGNED	Time adjustment duration
205	CHAR(1)	Time adjustment supported

Key Field Descriptions

Coordinated Universal Time. The value of the time-of-day clock which is returned as the Coordinated Universal Time (UTC) for the system. The UTC is returned as a system time-stamp.

Time adjustment amount. The time value which specifies the remaining amount of time, in microseconds, by which the time-of-day clock will be increased or decreased. If a time adjustment is not active, this field will be hex zeros.

Time adjustment direction. The direction of the time-of-day clock adjustment for the system. If a time adjustment is not active, this field will be blank.

0 Increase time of day.

1 Decrease time of day.

Time adjustment duration. The time value which provides an estimate of the amount of time, in microseconds, required in order to complete the time-of-day clock adjustment. If a time adjustment is not active, this field will be hex zeros.

Time adjustment status. The status of the time-of-day clock adjustment.

0 Time-of-day clock adjustment not active.

1 Time-of-day clock adjustment active.

Time adjustment supported. The availability of the time-of-day clock adjustments for the system.

0 Time adjustments not supported.

1 Time adjustments supported.

Error Messages

The following messages may be sent from this function:

Message ID	Error Message Text
CPF1866 E	Value &1 for number of fields to return not valid.
CPF1867 E	Value &1 in list not valid.
CPF24B4 E	Severe error while addressing parameter list.
CPF3C21 E	Format name &1 is not valid.
CPF3C24 E	Length of the receiver variable is not valid.
CPF3C36 E	Number of parameters, &1, entered for this API was not valid.
CPF3CF1 E	Error code parameter not valid.
CPF3CF2 E	Error(s) occurred during running of &1 API.
CPF9821 E	Not authorized to program &1 in library &2.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.

Retrieve Time Zone Description (QWCRTVTZ) API

Required Parameter Group:

1	Receiver variable	Output	Char(*)
2	Length of receiver variable	Input	Binary(4)
3	Format name	Input	Char(8)
4	Time zone description	Input	Char(10)
5	Error Code	I/O	Char(*)

Default Public Authority: *USE

Threadsafe: Yes

The Retrieve Time Zone Description (QWCRTVTZ) API retrieves information about one or more time zone descriptions. A time zone description contains information that is used to calculate a local time.

Authorities and Locks

Time Zone Description Authority

*USE

Message File Authority

*USE is required to retrieve the abbreviated and full names if a message is specified in the time zone description.

QSYS Library Authority

*USE

Message File Library Authority

*EXECUTE is required to retrieve the abbreviated and full names if a message is specified in the time zone description.

Time Zone Description Lock

*SHRNUP

Required Parameter Group

Receiver variable

OUTPUT; CHAR(*)

The variable to receive the information about the time zone descriptions.

Length of receiver variable

INPUT; BINARY(4)

The length of the receiver variable. The length must be at least 8 bytes.

Format name

INPUT; CHAR(8)

The format of the time zone description information. The possible format name is:

- RTMZ0100* ➤ Returns the properties of the time zone description. ⚡ See "Format RTMZ0100" on page 21 for details on the time zone description information returned.
- RTMZ0200* ➤ Returns the TZ string derived from the time zone description properties that are associated with the current date and time of the job. ⚡ See "Format RTMZ0200" on page 22 for details on the time zone description information returned.

Time zone description

INPUT; CHAR(10)

The name of the time zone description for which information is to be retrieved. This name can be a simple object name, a generic name or the following special value:

*ALL Retrieve information for all time zone descriptions.

Error code

I/O; CHAR(*)

The structure in which to return error information. For the format of the structure, see Error code parameter.

Format RTMZ0100

The RTMZ0100 format returns the following information for the specified time zone description.

Offset		Type	Field
Dec	Hex		
0	0	BINARY(4)	Bytes returned
4	4	BINARY(4)	Bytes available
8	8	BINARY(4)	Number of time zone descriptions available
12	C	BINARY(4)	Offset to time zone descriptions
16	10	BINARY(4)	Number of time zone descriptions returned
20	14	BINARY(4)	Length of a time zone description entry
24	18	CHAR(*)	Reserved

Offset		Type	Field
Dec	Hex		
These fields repeat, in the order listed, for the number of time zone descriptions returned.		CHAR(10)	Time zone description name
		CHAR(1)	Local system time indicator
		CHAR(1)	Daylight Saving Time indicator
		BINARY(4)	Offset from UTC
		CHAR(10)	Standard Time abbreviated name
		CHAR(50)	Standard Time full name
		CHAR(10)	Daylight Saving Time abbreviated name
		CHAR(50)	Daylight Saving Time full name
		CHAR(7)	Standard Time message
		CHAR(7)	Daylight Saving Time message
		CHAR(10)	Message file name
		CHAR(10)	Message file library name
		CHAR(2)	Daylight Saving Time start - month
		CHAR(1)	Daylight Saving Time start - day
		CHAR(1)	Daylight Saving Time start - relative day of month
		CHAR(6)	Daylight Saving Time start - time
		CHAR(2)	Daylight Saving Time end - month
		CHAR(1)	Daylight Saving Time end - day
		CHAR(1)	Daylight Saving Time end - relative day of month
		CHAR(6)	Daylight Saving Time end - time
		CHAR(50)	Text description
		» BINARY(4)	Daylight Saving Time shift
		BINARY(4)	Year offset
	CHAR(128)	Alternate name «	
	CHAR(*)	Reserved	

Format RTMZ0200

The RTMZ0200 format returns the following information for the specified time zone description.

Offset		Type	Field
Dec	Hex		
0	0	BINARY(4)	Bytes returned
4	4	BINARY(4)	Bytes available
8	8	BINARY(4)	Number of time zone descriptions available
12	C	BINARY(4)	Offset to time zone descriptions
16	10	BINARY(4)	Number of time zone descriptions returned
20	14	CHAR(*)	Reserved

Offset		Type	Field
Dec	Hex		
These fields repeat, in the order listed, for the number of time zone descriptions returned.		BINARY(4)	Length of this time zone description entry
		BINARY(4)	Displacement to TZ string
		BINARY(4)	Length of TZ string
		CHAR(10)	Time zone description name
		CHAR(*)	Reserved
		CHAR(*)	TZ string

Field Descriptions

» **Alternate name.** The descriptive name that provides additional information about the time zone description. If the time zone description does not have an alternate name, this field is blank. «

Bytes available. The number of bytes of data available to be returned. All available data is returned if enough space is provided.

Bytes returned. The number of bytes of data returned.

Daylight Saving Time abbreviated name. The abbreviated name used with the time zone when Daylight Saving Time is being observed. If the time zone description does not support Daylight Saving Time, this field is blank. If the time zone description uses a message to specify the abbreviated name and the message cannot be retrieved, this field returns *N. This can occur when the caller of the API is not authorized to the message file or its library, the message file cannot be found or the message does not exist in the specified message file.

Daylight Saving Time end - day. The day of the week on which Daylight Saving Time ends. The possible values are:

<i>blank</i>	This time zone description does not support Daylight Saving Time.
1	Daylight Saving Time ends on a Monday.
2	Daylight Saving Time ends on a Tuesday.
3	Daylight Saving Time ends on a Wednesday.
4	Daylight Saving Time ends on a Thursday.
5	Daylight Saving Time ends on a Friday.
6	Daylight Saving Time ends on a Saturday.
7	Daylight Saving Time ends on a Sunday.

Daylight Saving Time end - month. The month in which Daylight Saving Time ends. The possible values are:

<i>blank</i>	This time zone description does not support Daylight Saving Time.
01	Daylight Saving Time ends in January.
02	Daylight Saving Time ends in February.
03	Daylight Saving Time ends in March.
04	Daylight Saving Time ends in April.
05	Daylight Saving Time ends in May.
06	Daylight Saving Time ends in June.
07	Daylight Saving Time ends in July.
08	Daylight Saving Time ends in August.
09	Daylight Saving Time ends in September.

- 10 Daylight Saving Time ends in October.
- 11 Daylight Saving Time ends in November.
- 12 Daylight Saving Time ends in December.

Daylight Saving Time end - relative day of month. The relative day of the month on which Daylight Saving Time ends. The possible values are:

- blank* This time zone description does not support Daylight Saving Time.
- 1 Daylight Saving Time ends on the first occurrence of the specified day of the week.
- 2 Daylight Saving Time ends on the second occurrence of the specified day of the week.
- 3 Daylight Saving Time ends on the third occurrence of the specified day of the week.
- 4 Daylight Saving Time ends on the fourth occurrence of the specified day of the week.
- L Daylight Saving Time ends on the last occurrence of the specified day of the week.

Daylight Saving Time end - time. The time of day at which Daylight Saving Time ends. The time is specified in the form hhmmss, where hh = hours, mm = minutes and ss = seconds. If the time zone description does not support Daylight Saving Time, this field is blank.

Daylight Saving Time full name. The full name of the time zone when Daylight Saving Time is being observed. If the time zone description does not support Daylight Saving Time, this field is blank. If the time zone description uses a message to specify the full name and the message cannot be retrieved, this field returns *N. This can occur when the caller of the API is not authorized to the message file or its library, the message file cannot be found or the message does not exist in the specified message file.

Daylight Saving Time indicator. Indicates whether or not the time zone description supports Daylight Saving Time.

- 0 This time zone description does not support Daylight Saving Time.
- 1 This time zone description supports Daylight Saving Time.

Daylight Saving Time message. The message that contains the abbreviated and full names of the time zone when Daylight Saving Time is being observed. If the time zone description does not support Daylight Saving Time, this field is blank. If a message was not specified when the time zone description was created or last changed, this field returns *NONE.

» **Daylight Saving Time shift.** The number of minutes that local time moves forward when Daylight Saving Time starts or moves backward when Daylight Saving Time ends. If the time zone description does not support Daylight Saving Time, this field returns -1. «

Daylight Saving Time start - day. The day of the week on which Daylight Saving Time starts. The possible values are:

- blank* This time zone description does not support Daylight Saving Time.
- 1 Daylight Saving Time starts on a Monday.
- 2 Daylight Saving Time starts on a Tuesday.
- 3 Daylight Saving Time starts on a Wednesday.
- 4 Daylight Saving Time starts on a Thursday.
- 5 Daylight Saving Time starts on a Friday.
- 6 Daylight Saving Time starts on a Saturday.
- 7 Daylight Saving Time starts on a Sunday.

Daylight Saving Time start - month. The month in which Daylight Saving Time starts. The possible values are:

<i>blank</i>	This time zone description does not support Daylight Saving Time.
01	Daylight Saving Time starts in January.
02	Daylight Saving Time starts in February.
03	Daylight Saving Time starts in March.
04	Daylight Saving Time starts in April.
05	Daylight Saving Time starts in May.
06	Daylight Saving Time starts in June.
07	Daylight Saving Time starts in July.
08	Daylight Saving Time starts in August.
09	Daylight Saving Time starts in September.
10	Daylight Saving Time starts in October.
11	Daylight Saving Time starts in November.
12	Daylight Saving Time starts in December.

Daylight Saving Time start - relative day of month. The relative day of the month on which Daylight Saving Time starts. The possible values are:

<i>blank</i>	This time zone description does not support Daylight Saving Time.
1	Daylight Saving Time starts on the first occurrence of the specified day of the week.
2	Daylight Saving Time starts on the second occurrence of the specified day of the week.
3	Daylight Saving Time starts on the third occurrence of the specified day of the week.
4	Daylight Saving Time starts on the fourth occurrence of the specified day of the week.
L	Daylight Saving Time starts on the last occurrence of the specified day of the week.

Daylight Saving Time start - time. The time of day at which Daylight Saving Time starts. The time is specified in the form hhmmss, where hh = hours, mm = minutes and ss = seconds. If the time zone description does not support Daylight Saving Time, this field is blank.

Displacement to TZ string. The displacement, in bytes, from the beginning of this time zone description to the **TZ string**.

Length of a time zone description entry. The length of a single time zone description entry in the receiver variable.

Length of this time zone description entry. The length of this time zone description entry. This value is the number of bytes from the start of the time zone description entry to the start of the next entry (if any) in the receiver variable.

Length of TZ string. The length in bytes of the **TZ string**, excluding the terminating null at the end of the string.

Local system time indicator. Indicates whether or not the time zone description is currently specified in the Time zone (QTIMZON) system value. The time zone description specified in the QTIMZON system value is used to calculate local system time and cannot be deleted.

0	This time zone description is not currently specified in the QTIMZON system value.
1	This time zone description is currently specified in the QTIMZON system value.

Message file library name. The name of the library containing the message file used to retrieve the Standard Time message and the Daylight Saving Time message. The field may contain *LIBL which means that the library list is searched to locate the message file. The library name is left-justified and padded with blanks on the right. If the message file name specifies *NONE, this field returns blanks.

Message file name. The name of the message file used to retrieve the Standard Time message and the Daylight Saving Time message. The message file name is left-justified and padded with blanks on the right. If a message was not specified when the time zone description was created or last changed, this field returns *NONE.

Number of time zone descriptions available. The number of time zone descriptions that match the time zone description name specified on the call to this API. This is the number of time zone descriptions which the caller of the API has *USE authority to.

Number of time zone descriptions returned. The number of time zone descriptions returned in the receiver variable.

Offset from UTC. The time difference, in minutes, between this time zone and Coordinated Universal Time (UTC). This value is subtracted from local time to obtain UTC time. A negative difference indicates that the time zone is west of UTC and a positive difference indicates that the time zone is east of UTC.

Offset to time zone descriptions. The offset in bytes from the beginning of the receiver variable to the first time zone description.

Reserved. An unused field.

Standard Time abbreviated name. The abbreviated name used with the time zone when Daylight Saving Time is not being observed. If the time zone description uses a message to specify the abbreviated name and the message cannot be retrieved, this field returns *N. This can occur when the caller of the API is not authorized to the message file or its library, the message file cannot be found or the message does not exist in the specified message file.

Standard Time full name. The full name of the time zone when Daylight Saving Time is not being observed. If the time zone description uses a message to specify the full name and the message cannot be retrieved, this field returns *N. This can occur when the caller of the API is not authorized to the message file or its library, the message file cannot be found or the message does not exist in the specified message file.

Standard Time message. The message that contains the abbreviated and full names of the time zone when Daylight Saving Time is not being observed. If a message was not specified when the time zone description was created or last changed, this field returns *NONE.

Text description. The user text, if any, used to briefly describe the time zone description.

Time zone description name. The name of the time zone description object.

TZ string. A null-terminated character string that describes the time zone in a format compatible with industry standards. The returned string may be used to set i5/OS® PASE environment variable TZ, and has this format (spaces inserted for readability):

```
std offset dst offset , start/time , end/time
```

An example of a TZ string for time zone QN0600CST (USA Central time) is

» "<CST>6<CDT>,M4.1.0,M10.5.0". The TZ string for time zone QP1245UTC2 (Chatham Islands time) is "<UTC+12x45S>-12:45<UTC+12x45D>,M10.1.0/02:45:00,M3.3.0/03:45:00". « Daylight Saving Time specifications (dst, offset, start/time, and end/time) are omitted if the time zone does not use Daylight Saving Time.

std This is the **Standard Time abbreviated name**. » The first character shall be the less-than ('<') character and the last character shall be the greater-than ('>') character. All characters between these delimiting characters shall be alphanumeric characters, the plus-sign ('+') character, or the

minus-sign ('-') character. Trailing blanks are removed, and any character not allowed by standards is replaced with a lower case x ('x'). The value for this field does not include the delimiting characters. <<

offset Hours and minutes behind Coordinated Universal Time (UTC). Minutes are omitted if the time zone is an integral number of hours behind UTC. **offset** is formatted as hh:mm if minutes are included, and has a leading minus sign if the value is negative. A negative value indicates the time zone is east of UTC, which is the opposite of the **Offset from UTC** field (in format **RTMZ0100**). >> The offset following **std** is required. If no offset follows **dst**, Daylight Saving Time is assumed to be one hour ahead of Standard Time. <<

dst This is the **Daylight Saving Time abbreviated name**. >> The first character shall be the less-than ('<') character and the last character shall be the greater-than ('>') character. All characters between these delimiting characters shall be alphanumeric characters, the plus-sign ('+') character, or the minus-sign ('-') character. Trailing blanks are removed, and any character not allowed by standards is replaced with a lower case x ('x'). The value for this field does not include the delimiting characters. <<

start/time

Specifies when Daylight Saving Time starts. **start** includes the month (1-12), week number (1-5), and day (0-6, for Sunday-Saturday) in the form Mm.n.d. **time** is formatted as hh:mm:ss, but is omitted if Daylight Saving Time starts at 2:00:00.

end/time

Specifies when Daylight Saving Time ends. **end** includes the month (1-12), week number (1-5), and day (0-6, for Sunday-Saturday) in the form Mm.n.d. **time** is formatted as hh:mm:ss, but is omitted if Daylight Saving Time ends at 2:00:00.

>> **Year offset**. The number of years that the current year in the calendar system used with this time zone differs from the current Gregorian year. <<

Error Messages

Message ID	Error Message Text
CPF24B4 E	Severe error while addressing parameter list.
CPF3CF1 E	Error code parameter not valid.
CPF3CF2 E	Error(s) occurred during running of &1 API.
CPF3C19 E	Error occurred with receiver variable specified.
CPF3C21 E	Format name &1 is not valid.
CPF3C24 E	Length of the receiver variable is not valid.
CPF3C90 E	Literal value cannot be changed.
CPF8130 E	Time zone description &1 damaged.
CPF9801 E	Object &2 in library &3 not found.
CPF9802 E	Not authorized to object &2 in &3.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.

API introduced: V5R3

Top | "Date and Time APIs," on page 1 | APIs by category

Set System Time (QWCSETTM) API

Required Parameter Group:

1	Input format	Input	Char(8)
2	Input variable	Input	Char(*)
3	Error Code	I/O	Char(*)

Default Public Authority: *USE
Threadsafe: YES

The Set System Time (QWCSETTM) API sets the Coordinated Universal Time (UTC) for the system.

For additional information on setting the time, see “Usage Notes” on page 29.

Authorities and Locks

User Special Authority
*ALLOBJ

Required Parameter Group

Input format

INPUT; CHAR(8)

The format of the date specified for the input variable. Valid values are:

*YYMD YYYYMMDD (4-digit year, month, day) format.
*MDYY MMDDYYYY (month, day, 4-digit year) format.
*DMYY DDMMYYYY (day, month, 4-digit year) format.

Input variable

INPUT; CHAR(*)

The date as Coordinated Universal Time (UTC) that is used to set the time-of-day clock. See “Input Variable Formats” to determine the structure of the input variable.

Error code

I/O; CHAR(*)

The structure in which to return error information. For the format of the structure, see Error code parameter.

Input Variable Formats

This table shows the formats used for the input variable parameter.

Input Format	Input Variable
*YYMD, *MDYY, *DMYY in microseconds	The first 20 characters are used. See “20-Byte Character Date and Time Value Structure.”

20-Byte Character Date and Time Value Structure

This table shows the structure used for the input variable when the format is *YYMD, *MDYY, and *DMYY.

Offset	Description
0-7	Date, left-justified. This value cannot be all blanks or all zeros.
8-13	Time, in HHMMSS (hours, minutes, seconds) format.
14-19	Microseconds. This value cannot be blanks.

Usage Notes

The supported date range is from August 23, 1928, 12:03:06.314752 to May 10, 2071, 11:56:53.685240. Setting an input date that is outside this range will result in a date within this range. If you specify a precision of microseconds, it is recommended that you use a microsecond value that is evenly divisible by 8.

Error Messages

The following messages may be sent from this function:

Message ID	Error Message Text
CPF1060 E	Date not valid.
CPF1061 E	Time not valid.
CPF1890 E	*ALLOBJ authority required for requested operation.
CPF24B4 E	Severe error while addressing parameter list.
CPF3C21 E	Format name &1 is not valid.
CPF3C36 E	Number of parameters, &1, entered for this API was not valid.
CPF3CF1 E	Error code parameter not valid.
CPF3CF2 E	Error(s) occurred during running of &1 API.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.

API introduced: V5R3

[Top](#) | ["Date and Time APIs," on page 1](#) | [APIs by category](#)

Code license and disclaimer information

IBM grants you a nonexclusive copyright license to use all programming code examples from which you can generate similar function tailored to your own specific needs.

SUBJECT TO ANY STATUTORY WARRANTIES WHICH CANNOT BE EXCLUDED, IBM, 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. DIRECT, 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 DIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, SO SOME OR ALL OF THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU.

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 document 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.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

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.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

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

Programming interface information

This API descriptions publication documents intended Programming Interfaces that allow the customer to write programs to obtain the services of IBM i5/OS.

Trademarks

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

Advanced 36
Advanced Function Presentation
Advanced Peer-to-Peer Networking
AFP
AIX
AnyNet
AS/400
BCOCA
C/400
COBOL/400
Common User Access
CUA
DB2
DB2 Universal Database
Distributed Relational Database Architecture
Domino
DPI
DRDA
Enterprise Storage Server
eServer
FlashCopy
GDDM
i5/OS
IBM
IBM (logo)
InfoColor
Infoprint
Integrated Language Environment
Intelligent Printer Data Stream
IPDS
Lotus
Lotus Notes
MO:DCA
MVS
Net.Data
NetServer
Notes
OfficeVision
Operating System/2
Operating System/400
OS/2
OS/400
PartnerWorld
POWER5+
PowerPC
Print Services Facility
PrintManager
PROFS
RISC System/6000
RPG/400
RS/6000

SAA
SecureWay
SOM
System i
System i5
System Object Model
System/36
System/38
System/390
TotalStorage
VisualAge
WebSphere
xSeries
z/OS

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

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, or service names may be trademarks or service marks of others.

Terms and conditions

Permissions for the use of these publications is granted subject to the following terms and conditions.

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

Commercial Use: You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications 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 publications or any information, 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 publications 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 THESE PUBLICATIONS. THE PUBLICATIONS ARE 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.



Printed in USA