

# SoMachine

## Getting & Setting Real Time Clock SysTimeRtc and SysTimeCore Library Guide

06/2017

EIO00000006667.08

[www.schneider-electric.com](http://www.schneider-electric.com)



---

The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products contained herein. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any such user or integrator to perform the appropriate and complete risk analysis, evaluation and testing of the products with respect to the relevant specific application or use thereof. Neither Schneider Electric nor any of its affiliates or subsidiaries shall be responsible or liable for misuse of the information contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

No part of this document may be reproduced in any form or by any means, electronic or mechanical, including photocopying, without express written permission of Schneider Electric.

All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When devices are used for applications with technical safety requirements, the relevant instructions must be followed.

Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

© 2017 Schneider Electric. All Rights Reserved.

---

# Table of Contents

---



	<b>Safety Information</b> .....	<b>5</b>
	<b>About the Book</b> .....	<b>7</b>
<b>Chapter 1</b>	<b>General Information</b> .....	<b>11</b>
	General Information .....	<b>11</b>
<b>Chapter 2</b>	<b>Getting Started</b> .....	<b>13</b>
	Adding the Libraries .....	<b>14</b>
	Get the Controller Date and Time .....	<b>15</b>
	Set the Controller Date and Time .....	<b>17</b>
<b>Chapter 3</b>	<b>SysTimeCore Library</b> .....	<b>19</b>
3.1	Functions Descriptions .....	<b>20</b>
	SysTimeGetMs .....	<b>21</b>
	SysTimeGetNs .....	<b>22</b>
	SysTimeGetUs .....	<b>23</b>
<b>Chapter 4</b>	<b>SysTimeRtc Library</b> .....	<b>25</b>
4.1	Functions Descriptions .....	<b>26</b>
	SysTimeRtcControl .....	<b>27</b>
	SysTimeRtcGet .....	<b>28</b>
	SysTimeRtcHighResGet .....	<b>29</b>
	SysTimeRtcSet .....	<b>30</b>
	SysTimeRtcHighResSet .....	<b>31</b>
	SysTimeRtcConvertDateToUtc .....	<b>32</b>
	SysTimeRtcConvertDateToHighRes .....	<b>33</b>
	SysTimeRtcConvertUtcToDate .....	<b>34</b>
	SysTimeRtcConvertHighResToDate .....	<b>35</b>
	SysTimeRtcGetTimezone .....	<b>36</b>
	SysTimeRtcSetTimezone .....	<b>37</b>
	SysTimeRtcConvertLocalToUtc .....	<b>38</b>
	SysTimeRtcConvertLocalToHighRes .....	<b>39</b>
	SysTimeRtcConvertUtcToLocal .....	<b>40</b>
	SysTimeRtcConvertHighResToLocal .....	<b>41</b>
4.2	Data Type Descriptions .....	<b>42</b>
	SYSTIMEDATE .....	<b>43</b>
	RTS_SYSTIMEDATE .....	<b>43</b>
	TimezoneInformation .....	<b>44</b>
<b>Appendices</b>	.....	<b>45</b>

---

<b>Appendix A</b>	<b>Function and Function Block Representation</b> .....	<b>47</b>
	Differences Between a Function and a Function Block .....	<b>48</b>
	How to Use a Function or a Function Block in IL Language .....	<b>49</b>
	How to Use a Function or a Function Block in ST Language .....	<b>53</b>
<b>Glossary</b>	.....	<b>57</b>
<b>Index</b>	.....	<b>61</b>

---

# Safety Information

---



## Important Information

### NOTICE

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.



The addition of this symbol to a “Danger” or “Warning” safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.



This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

## **DANGER**

**DANGER** indicates a hazardous situation which, if not avoided, **will result in** death or serious injury.

## **WARNING**

**WARNING** indicates a hazardous situation which, if not avoided, **could result in** death or serious injury.

## **CAUTION**

**CAUTION** indicates a hazardous situation which, if not avoided, **could result** in minor or moderate injury.

## **NOTICE**

**NOTICE** is used to address practices not related to physical injury.

---

## PLEASE NOTE

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

# About the Book



## At a Glance

### Document Scope

This documentation describes the libraries SysTimeRtc and SysTimeCore.

The SysTimeRtc library implements functions for the real time clock (RTC) management of the runtime system.

The SysTimeCore library provides access to high resolution time routines for time calculations.

The following table shows the list of the functions splitted in the two libraries and the controllers that support them:

Function Name	Library Name	M238	M241, M251, M258 LMC058, LMC078	XBT-GC XBT-GK XBT-GT HMISCU	ATV IMC
SysTimeGetMs	SysTimeCore	x	x	x	x
SysTimeGetNs	SysTimeCore	-	x	x	x
SysTimeGetUs	SysTimeCore	x	x	x	x
SysTimeRtcConvertDateToHighRes	SysTimeRtc	-	x	x	-
SysTimeRtcConvertHighResToDate	SysTimeRtc	-	x	x	-
SysTimeRtcHighResGet	SysTimeRtc	-	x	x	-
SysTimeRtcHighResSet	SysTimeRtc	-	x	x	-
SysTimeRtcControl	SysTimeRtc	-	only supported by LMC078	x	x
SysTimeRtcConvertDateToUtc	SysTimeRtc	x	x	x	x
SysTimeRtcConvertUtcToDate	SysTimeRtc	x	x	x	x
SysTimeRtcConvertLocalToUtc	SysTimeRtc	-	-	-	-
SysTimeRtcConvertUtcToLocal	SysTimeRtc	-	-	-	-
SysTimeRtcGet	SysTimeRtc	x	x	x	x
SysTimeRtcSet	SysTimeRtc	x	x	x	x
SysTimeRtcConvertHighResToLocal	SysTimeRtc	-	-	-	-
SysTimeRtcConvertLocalToHighRes	SysTimeRtc	-	-	-	-
SysTimeRtcGetTimezone	SysTimeRtc	x	x	x	x
<b>x</b> Supported - Not supported					

Function Name	Library Name	M238	M241, M251, M258 LMC058, LMC078	XBT-GC XBT-GK XBT-GT HMISCU	ATV IMC
SysTimeRtcSetTimezone	SysTimeRtc	x	x	x	x
<b>x</b> Supported <b>-</b> Not supported					

### Validity Note

This document has been updated for the release of SoMachine V4.3.

### Related Documents

Document title	Reference
Modicon M238 Logic Controller - System Functions and Variables - PLCSystem Library Guide	<a href="#">EIO000000364 (ENG);</a> <a href="#">EIO000000757 (FRE);</a> <a href="#">EIO000000758 (GER);</a> <a href="#">EIO000000759 (SPA);</a> <a href="#">EIO000000760 (ITA);</a> <a href="#">EIO000000761 (CHS)</a>
Modicon M241 Logic Controller - System Functions and Variables - PLCSystem Library Guide	<a href="#">EIO0000001438 (ENG);</a> <a href="#">EIO0000001439 (FRE);</a> <a href="#">EIO0000001440 (GER);</a> <a href="#">EIO0000001441 (SPA);</a> <a href="#">EIO0000001442 (ITA);</a> <a href="#">EIO0000001443 (CHS)</a>
Modicon M251 Logic Controller - System Functions and Variables - PLCSystem Library Guide	<a href="#">EIO0000001468 (ENG);</a> <a href="#">EIO0000001469 (FRE);</a> <a href="#">EIO0000001470 (GER);</a> <a href="#">EIO0000001471 (SPA);</a> <a href="#">EIO0000001472 (ITA);</a> <a href="#">EIO0000001473 (CHS)</a>
Modicon M258 Logic Controller - System Functions and Variables - PLCSystem Library Guide	<a href="#">EIO0000000584 (ENG);</a> <a href="#">EIO0000000585 (FRE);</a> <a href="#">EIO0000000586 (GER);</a> <a href="#">EIO0000000587 (ITA);</a> <a href="#">EIO0000000588 (SPA);</a> <a href="#">EIO0000000589 (CHS)</a>
Altivar ATV IMC Drive Controller - System Functions and Variables - PLCSystem Library Guide	<a href="#">EIO0000000596 (ENG);</a> <a href="#">EIO0000000597 (FRE);</a> <a href="#">EIO0000000598 (GER);</a> <a href="#">EIO0000000599 (SPA);</a> <a href="#">EIO0000000600 (ITA);</a> <a href="#">EIO0000000601 (CHS)</a>



Document title	Reference
Modicon LMC058 Motion Controller - System Functions and Variables - PLCSystem Library Guide	<a href="#">EIO0000000566 (ENG);</a> <a href="#">EIO0000000567 (FRE);</a> <a href="#">EIO0000000568 (GER);</a> <a href="#">EIO0000000569 (ITA);</a> <a href="#">EIO0000000570 (SPA);</a> <a href="#">EIO0000000571 (CHS);</a>
Modicon LMC078 Motion Controller - System Functions and Variables - PLCSystem Library Guide	<a href="#">EIO0000001917 (ENG);</a> <a href="#">EIO0000001918 (FRE);</a> <a href="#">EIO0000001919 (GER);</a> <a href="#">EIO0000001920 (SPA);</a> <a href="#">EIO0000001921 (ITA);</a> <a href="#">EIO0000001922 (CHS);</a>
Magelis XBTGC, XBTGT, XBTGK HMI Controller - System Functions and Variables - PLCSystem Library Guide	<a href="#">EIO0000000626 (ENG);</a> <a href="#">EIO0000000627 (FRE);</a> <a href="#">EIO0000000628 (GER);</a> <a href="#">EIO0000000629 (ITA);</a> <a href="#">EIO0000000630 (SPA);</a> <a href="#">EIO0000000631 (CHS);</a>
Magelis SCU HMI Controller - System Functions and Variables - PLCSystem Library Guide	<a href="#">EIO0000001246 (ENG);</a> <a href="#">EIO0000001247 (FRE);</a> <a href="#">EIO0000001248 (GER);</a> <a href="#">EIO0000001249 (SPA);</a> <a href="#">EIO0000001250 (ITA);</a> <a href="#">EIO0000001251 (CHS);</a>

You can download these technical publications and other technical information from our website at <http://www.schneider-electric.com/en/download>.

---

## Product Related Information

### WARNING

#### LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.<sup>1</sup>
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

**Failure to follow these instructions can result in death, serious injury, or equipment damage.**

<sup>1</sup> For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

### WARNING

#### UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

**Failure to follow these instructions can result in death, serious injury, or equipment damage.**

---

# Chapter 1

## General Information

---

### General Information

#### Overview

Depending on the runtime system, the real time clock (RTC) of the controller is provided in standard or high resolution time stamp. That is, the time stamp represents the seconds (standard resolution) or milliseconds (high resolution) since January 1st, 1970 00:00:00. It is also known as UNIX format.

The RTC in standard resolution is supported for all SoMachine controllers. Refer to the table (*see page 7*) in the Document Scope chapter to verify for which controller supports the high resolution RTC.



---

# Chapter 2

## Getting Started

---

### Overview

This chapter provides essential information to start using the `SysTime` functions.

### What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Adding the Libraries	14
Get the Controller Date and Time	15
Set the Controller Date and Time	17

## Adding the Libraries

### Procedure

To have access to the functions, it is necessary to manually add one or both libraries:

Step	Action
1	Double-click the <b>Library Manager</b> node of the controller <b>Application</b> node in the <b>Tools tree</b> .
2	Click <b>Add Library</b> .
3	Browse to <b>System</b> → <b>SysLibs</b> , visible when the <b>Company</b> filter is set to <b>System</b> or <b>All Company</b> . <b>Result:</b> The System libraries list is displayed.
4	Select the SysTimeRtc or SysTimeCore library and press <b>OK</b> . <b>Result:</b> The library is added to the <b>Library Manager</b> list.

## Get the Controller Date and Time

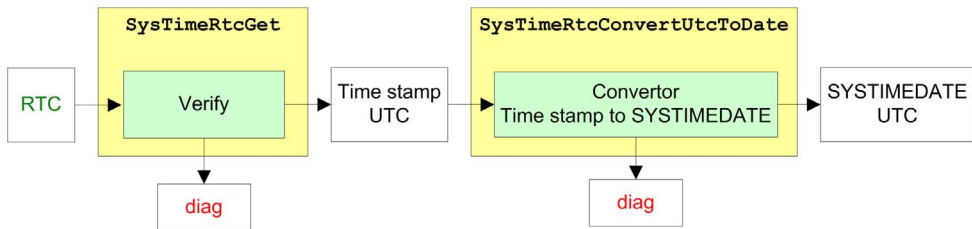
### Overview

To get the RTC of the controller in a structured and ergonomic format, you have to use 2 different functions.

1. Read the RTC, using the functions `SysTimeRtcGet` ([see page 28](#)) or `SysTimeRtcHighResGet` ([see page 29](#)).
2. Convert the time stamp in UNIX format to the SYSTIMEDATE format with the use of the function `SysTimeRtcConvertUtcToDate` ([see page 34](#)) or `SysTimeRtcConvertHighResToDate` ([see page 35](#)).

**NOTE:** Due to the fact that only the UTC (Coordinated Universal Time) time is globally unique, on most systems only the UTC time is stored and processed.

### Principle Diagram - Get the RTC of the Controller in Standard Resolution



### Example

This program example can be used to get the controller date and time.

#### Variable declaration:

```

VAR
    uidResultRtcGet: UDINT
    stGetDate: SysTimeRtc.RTS_SYSTIMEDATE;
    uiGetYear: UINT;
    uiGetMonth: UINT;
    uiGetDay: UINT;
    uiGetHour: UINT;
    uiGetMinute: UINT;
    uiGetSecond: UINT;
    uiGetMSecond: UINT;
  
```

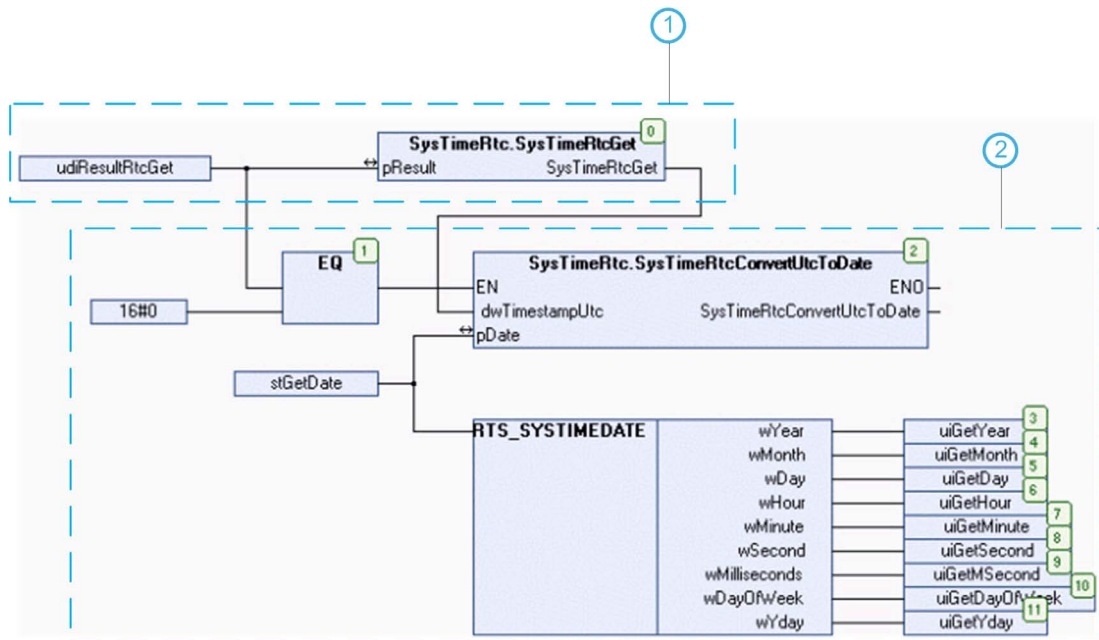
```

uiGetDayOfWeek: UINT;
uiGetYday: UINT;
uidResultConvertToDate: UDINT;

```

END\_VAR

**POU program:**



- 1 Get the RTC of the controller as time stamp value.
- 2 Convert the time stamp value in SYSTIMEDATE format.



## Set the Controller Date and Time

### Overview

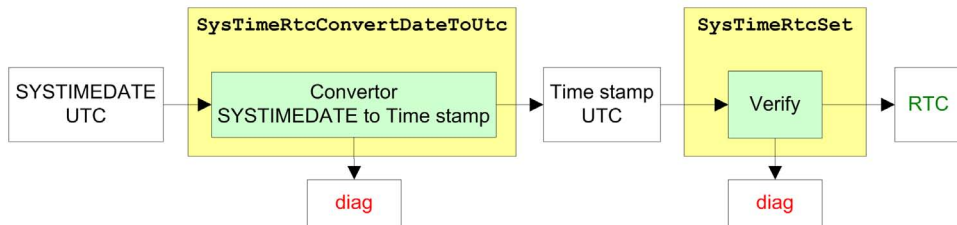
To set the RTC of the controller based on a structured and ergonomic format, you have to use 2 different functions.

1. Convert the SYSTIMEDATE format to the time stamp in UNIX format with the use of the function `SysTimeRtcConvertDateToUtc` (see page 32) or `SysTimeRtcConvertDateToHighRes` (see page 33).
2. Write the RTC, using the functions `SysTimeRtcSet` (see page 30) or `SysTimeRtcHighResSet` (see page 31).

**NOTE:** Some controllers support a function for a weekly automatic correction of the real time clock. The name of this function is `SetRTCDrift`. The use of this function could be an alternative to using the `SysTimeRtcSet` function for the continuously readjustment of the RTC. Refer to the PLCSystem Library Guide (see page 8) of your controller to verify whether the function is supported and to get further information about this function.

**NOTE:** Due to the fact that only the UTC (Coordinated Universal Time) time is globally unique, on most systems only the UTC time is stored and processed.

### Principle Diagram - Set the RTC of the Controller in Standard Resolution



### Example

This program example can be used to set the controller real time clock with a user date and time.

#### Variable declaration:

```

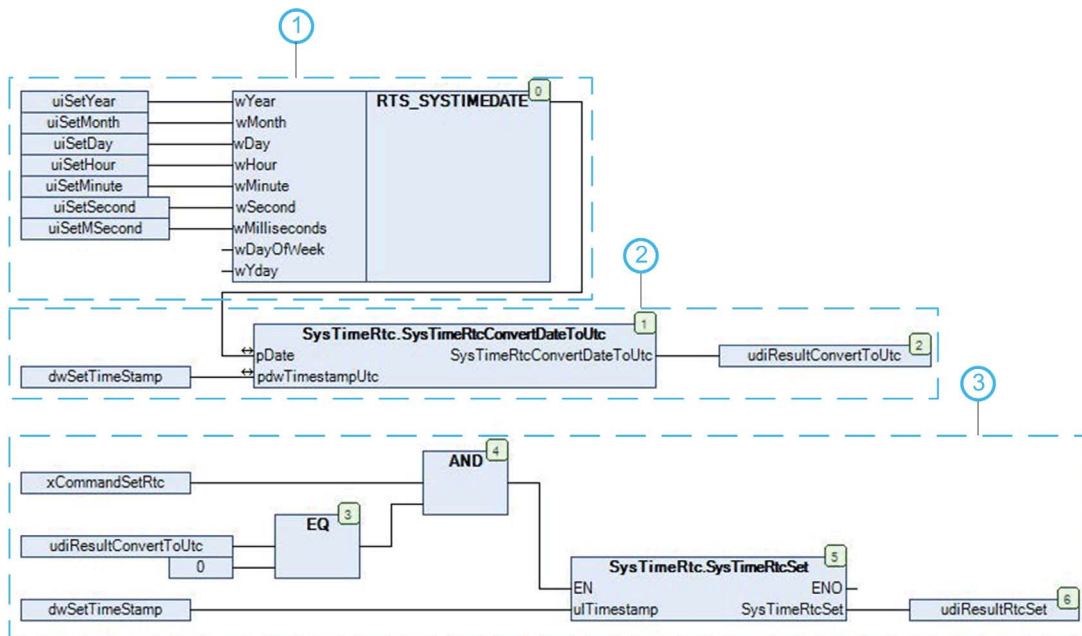
VAR
    uiSetYear: UINT;
    uiSetMonth: UINT;
    uiSetDay: UINT;
    uiSetHour: UINT;
    uiSetMinute: UINT;
  
```

```

uiSetSecond: UINT;
uiSetMSecond: UINT
udiResultConvertToUtc: UDINT;
dwSetTimeStamp: DWORD;
xCommandSetRtc: BOOL;
uidResultRtcSet: UDINT;
    
```

END\_VAR

**POU program:**



- 1 Assign the date and time parameter to the structure.
- 2 Convert the SYSTIMEDATE format to a time stamp value.
- 3 Set the controller RTC with new time stamp if xCommandSetRtc = TRUE and conversion was successful.

---

# Chapter 3

## SysTimeCore Library

---

# Section 3.1

## Functions Descriptions

---

### Overview

This section describes the functions of the SysTimeCore library.

### What Is in This Section?

This section contains the following topics:

Topic	Page
SysTimeGetMs	21
SysTimeGetNs	22
SysTimeGetUs	23

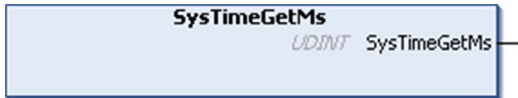
## SysTimeGetMs

### Function Description

Returns a monotonic rising millisecond counter. This value can be used for timeout and relative time measurements. The counter is reset at each boot-up of the controller.

**NOTE:** The real time clock does not influence this counter.

### Graphical Representation



### I/O Variables Description

Output	Type	Description
SysTimeGetMs	UDINT	Value of the millisecond counter, which is reset on boot-up.

## SysTimeGetNs

### Function Description

Returns a monotonic rising nanosecond counter. This value can be used for timeout and high resolution time measurements.

The counter is reset at each boot-up of the controller.

**NOTE:** The real time clock does not influence this counter.

### Graphical Representation



### I/O Variables Description

Output	Type	Description
SysTimeGetNs	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

Input/Output	Type	Description
pUsTime	SYSTIME	Value of the nanosecond counter, which is reset on boot-up.

**NOTE:** SYSTIME is an alias type based on the data type ULINT.

## SysTimeGetUs

### Function Description

Returns a monotonic rising microsecond counter. This value can be used for timeout and high resolution time measurements.

The counter is reset at each boot-up of the controller.

**NOTE:** The real time clock does not influence this counter.

### Graphical Representation



### I/O Variables Description

Output	Type	Description
SysTimeGetUs	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

Input/Output	Type	Description
pUsTime	SYSTEM	Value of the microsecond counter, which is reset on boot-up.

**NOTE:** SYSTEM is an alias type based on the data type ULINT.





---

# Chapter 4

## SysTimeRtc Library

---

### What Is in This Chapter?

This chapter contains the following sections:

Section	Topic	Page
4.1	Functions Descriptions	26
4.2	Data Type Descriptions	42

# Section 4.1

## Functions Descriptions

### Overview

This section describes the functions of the SysTimeRtc library.

The function for the real time clock handle two different time types:

**UTC time:** coordinated universal time; has replaced the Greenwich Mean Time. The time zones are given as positive or negative deviation from UTC. For example, UTC+1 corresponds to the Central European Time (CET) and UTC+2 corresponds to the Central European Summer Time (CEST).

**Local time:** Local time, including daylight saving time (summer time) and time zone shift.

**NOTE:** Due to the fact that only the UTC (Coordinated Universal Time) time is globally unique, on most systems only the UTC time is stored and processed.

### What Is in This Section?

This section contains the following topics:

Topic	Page
SysTimeRtcControl	27
SysTimeRtcGet	28
SysTimeRtcHighResGet	29
SysTimeRtcSet	30
SysTimeRtcHighResSet	31
SysTimeRtcConvertDateToUtc	32
SysTimeRtcConvertDateToHighRes	33
SysTimeRtcConvertUtcToDate	34
SysTimeRtcConvertHighResToDate	35
SysTimeRtcGetTimezone	36
SysTimeRtcSetTimezone	37
SysTimeRtcConvertLocalToUtc	38
SysTimeRtcConvertLocalToHighRes	39
SysTimeRtcConvertUtcToLocal	40
SysTimeRtcConvertHighResToLocal	41

## SysTimeRtcControl

### Function Description

This function is used to read the hardware status information related to the real time clock (RTC).

### Graphical Representation



### I/O Variables Description

Input	Type	Description
iControlTag	DINT	0 = to verify the battery of the RTC 1 = to verify the hour format of the RTC

Input/Output	Type	Description
pdiControlResult	DINT	Result of the verification. Hour format verification: 0 = 12-hour format 1 = 24-hour format Battery verification: 0 = indicates that the battery needs to be replaced 1 = battery is ok

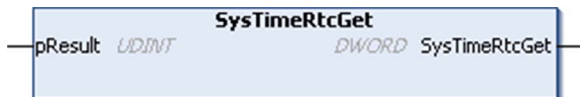
Output	Type	Description
SysTimeRtcControl	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

## SysTimeRtcGet

### Function Description

This function is used to read the value of the real time clock (RTC) of the controller. The RTC is provided as time stamp which indicates the number of seconds since January 1st, 1970 00:00:00.

### Graphical Representation



### I/O Variables Description

Input/Output	Type	Description
pResult	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

Output	Type	Description
SysTimeRtcGet	DWORD	RTC of the controller (number of seconds since January 1st, 1970 00:00:00)

**NOTE:** An example using this function is provided in this document. ([see page 15](#))

## SysTimeRtcHighResGet

### Function Description

This function is used to read the value of the real time clock (RTC) of the controller. The RTC is provided as high resolution time stamp which indicates the number of milliseconds since January 1st, 1970 00:00:00:000.

### Graphical Representation



### I/O Variables Description

Input/Output	Type	Description
pTimestamp	SYSTIME	Time in milliseconds since January 1st, 1970 00:00:00:000

Output	Type	Description
SysTimeRtcHighResGet	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

**NOTE:** SYSTIME is an alias type based on the data type ULINT.

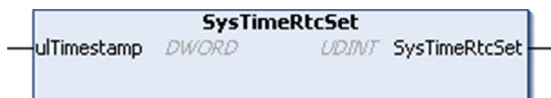
## SysTimeRtcSet

### Function Description

This function is used to set the real time clock of the controller by a provided time stamp which indicates the numbers of seconds since January 1st, 1970 00:00:00.

**NOTE:** Setting the RTC of the controller generates entries into the controller log file. Therefore, for automatic adjustments, do not use this function more than once a day.

### Graphical Representation



### I/O Variables Description

Input	Type	Description
ulTimestamp	DWORD	Time stamp value (number of seconds since January 1st, 1970 00:00:00)

Output	Type	Description
SysTimeRtcSet	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

**NOTE:** An example using this function is provided in this document. ([see page 17](#))

## SysTimeRtcHighResSet

### Function Description

This function is used to set the real time clock (RTC) of the controller by a provided high resolution time stamp value which indicates the number of milliseconds since January 1st, 1970 00:00:00:000.

**NOTE:** Setting the RTC of the controller generates entries into the controller log file. Therefore, for automatic adjustments, do not use this function more than once a day.

### Graphical Representation



### I/O Variables Description

Input/Output	Type	Description
pTimestamp	SYSTIME	Time stamp value (number in milliseconds since January 1st, 1970 00:00:00:000).

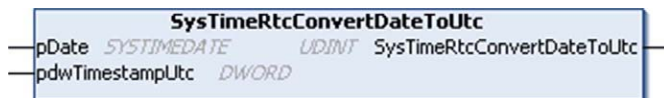
Output	Type	Description
SysTimeRtcHighResSet	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

## SysTimeRtcConvertDateToUtc

### Function Description

This function converts a date and time in SYSTIMEDATE format (*see page 43*) into the corresponding time stamp value. The time stamp indicates the number of seconds since January 1st, 1970 00:00:00.

### Graphical Representation



### I/O Variables Description

Input/Output	Type	Description
pDate	SYSTIMEDATE <i>(see page 43)</i>	Date and time to be converted.
pdwTimestampUtc	DWORD	Time stamp calculated from pDate.

Output	Type	Description
SysTimeRtcConvertDateToUtc	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

**NOTE:** An example using this function is provided in this document. (*see page 17*)



## SysTimeRtcConvertDateToHighRes

### Function Description

This function converts a date and time in SYSTIMEDATE (*see page 43*) format into the corresponding high resolution time stamp value. The time stamp indicates the number of milliseconds since January 1st, 1970 00:00:00:000.

### Graphical Representation



### I/O Variables Description

Input/Output	Type	Description
pDate	SYSTIMEDATE ( <i>see page 43</i> )	The time to be converted.
pTimestamp	SYSTIME	Time stamp calculated from pDate.

Output	Type	Description
SysTimeRtcConvertDateToHighRes	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

**NOTE:** SYSTIME is an alias type based on the data type ULINT.

## SystemRtcConvertUtcToDate

### Function Description

This function converts a time stamp value into the corresponding date and time in SYSTIMEDATE format. (see page 43) The time stamp indicates the number of seconds since January 1st, 1970 00:00:00.

### Graphical Representation



### I/O Variables Description

Input	Type	Description
dwTimestampUtc	DWORD	Time stamp to be converted.

Input/Output	Type	Description
pDate	SYSTIMEDATE (see page 43)	Date and time calculated from input value.

Output	Type	Description
SystemRtcConvertUtcToDate	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

**NOTE:** An example using this function is provided in this document. (see page 15)

## SysTimeRtcConvertHighResToDate

### Function Description

This function converts a high resolution time stamp value into the corresponding date and time in SYSTIMEDATE (*see page 43*) format. The time stamp indicates the number of milliseconds since January 1st, 1970 00:00:00:000.

### Graphical Representation



### I/O Variables Description

Input/Output	Type	Description
pTimestamp	SYSTIME	Time stamp to be converted.
pDate	SYSTIMEDATE ( <i>see page 43</i> )	Date and time calculated from the input value.

Output	Type	Description
SysTimeRtcConvertHighResToDate	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

**NOTE:** SYSTIME is an alias type based on the data type ULINT.

## SysTimeRtcGetTimezone

### Function Description

This function is used to read the timezone settings of the controller.

### Graphical Representation



### I/O Variables Description

Input/Output	Type	Description
pTimezone	TimezoneInformation <i>(see page 44)</i>	Timezone settings of the controller.

Output	Type	Description
SysTimeRtcGetTimezone	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

## SysTimeRtcSetTimezone

### Function Description

This function is used to set the specified timezone settings.

### Graphical Representation



### I/O Variables Description

Input/Output	Type	Description
pTimezone	TimezoneInformation <i>(see page 44)</i>	Timezone settings to be set for the controller.

Output	Type	Description
SysTimeRtcSetTimezone	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

## SystemRtcConvertLocalToUtc

### Function Description

This function calculates the UTC (Coordinated Universal Time) from a local time considering the timezone setting of the runtime system. Both, the UTC and the local time stamp indicate the number of seconds since January 1st, 1970 00:00:00.

### Graphical Representation



### I/O Variables Description

Input	Type	Description
dwTimestampLocal	DWORD	Local time stamp

Input/Output	Type	Description
pdwTimestampUTC	DWORD	UTC calculated from the input.

Output	Type	Description
SysTimeRtcConvertLocalToUtc	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

## SysTimeRtcConvertLocalToHighRes

### Function Description

This function converts a date and time in SYSTIMEDATE (*see page 43*) format into the corresponding high resolution time stamp value considering the timezone settings of the runtime system.

The value pDate represents the local time and will be converted to the UTC value, which is provided on pTimestamp. The value pTimestamp indicates the number of milliseconds since January 1st, 1970 00:00:00:000.

### Graphical Representation



### I/O Variables Description

Input/Output	Type	Description
pDate	SYSTIMEDATE ( <i>see page 43</i> )	Local time to be converted.
pTimestamp	SYSTIME	Time stamp calculated from pDate.

Output	Type	Description
SysTimeRtcConvertLocalToHighRes	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected

**NOTE:** SYSTIME is an alias type based on the data type ULINT.

## SystemRtcConvertUtcToLocal

### Function Description

This function calculates the local time from the UTC (Coordinated Universal Time) considering the time zone setting of the runtime system. Both, the UTC and the local time stamp indicate the number of seconds since January 1st, 1970 00:00:00.

### Graphical Representation



### I/O Variables Description

Input	Type	Description
dwTimestampUtc	DWORD	UTC time stamp

Input/Output	Type	Description
pdwTimestampLocal	DWORD	Local time stamp calculated from the input.

Output	Type	Description
SysTimeRtcConvertUtcToLocal	UDINT	Runtime system error code (refer to CmpErrors.library): 0 = no error detected



## SysTimeRtcConvertHighResToLocal

### Function Description

This function converts a high resolution time stamp value into the corresponding date and time in SYSTIMEDATE (*see page 43*) format considering the timezone settings of the runtime system. The value `pTimestamp` represents the UTC and will be converted to the local time, which is provided on `pDate`.

### Graphical Representation



### I/O Variables Description

Input/Output	Type	Description
<code>pTimestamp</code>	SYSTIME	Time stamp to be converted.
<code>pDate</code>	SYSTIMEDATE ( <i>see page 43</i> )	Local time calculated from <code>pTimestamp</code> .

Output	Type	Description
<code>SysTimeRtcConvertHighResToLocal</code>	UDINT	Runtime system error code (refer to <code>CmpErrors.library</code> ): 0 = no error detected

**NOTE:** SYSTIME is an alias type based on the data type ULINT.

# Section 4.2

## Data Type Descriptions

---

### What Is in This Section?

This section contains the following topics:

Topic	Page
SYSTIMEDATE	43
RTS_SYSTIMEDATE	43
TimezoneInformation	44

## SYSTIMEDATE

### Data Type Description

The data type SYSTIMEDATE is an alias type based on the data type RTS\_SYSTIMEDATE.

## RTS\_SYSTIMEDATE

### Data Type Description

The data type RTS\_SYSTIMEDATE is a structure which implements the elements needed to represent a high resolution date and time value.

This structure contains detailed information on the date and time, presented in a readable format (in contrast to the time stamp).

Name	Type	Description
wYear	UINT	Year
wMonth	UINT	Month, values 1 to12 (1=January, 12=December)
wDay	UINT	Day of month, values 1 to 31
wHour	UINT	Hours of current day, values 0 to 23
wMinute	UINT	Minutes of current hour, values 0 to 59
wSecond	UINT	Seconds of current minute, values 0 to 59
wMilliseconds	UINT	Milliseconds of current second, values 0 to 999
wDayOfWeek	UINT	Day of the week, values 1 to 7 (1=Monday, 7=Sunday)
wYday	UINT	Day of the year, values 1 to 366 (1=January 1 <sup>st</sup> , 365 or 366=December 31 <sup>st</sup> )

## TimezoneInformation

### Data Type Description

This information describes a local timezone with standard- and daylight-saving-time (also known as summer and winter time).

Name	Type	Description	Unit
ulStandardDate	UDINT	Date to switch to standard time (from summer to winter time).	Number of seconds since January 1st, 1970 00:00:00.
ulDaylightDate	UDINT	Date to switch to daylight saving time (from winter to summer time).	
szStandardName	STRING (32)	Standard name of the timezone (winter time).	String with maximum 32 characters
szDaylightName	STRING (32)	Daylight saving time name (summer time).	String with maximum 32 characters
iBias	INT	UTC = local time + bias	minutes
iStandardBias	INT	Additional offset at standard time (winter time).	minutes
iDaylightBias	INT	Additional offset at daylight saving time (summer time).	minutes

---

# Appendices

---





---

# Appendix A

## Function and Function Block Representation

---

### Overview

Each function can be represented in the following languages:

- IL: Instruction List
- ST: Structured Text
- LD: Ladder Diagram
- FBD: Function Block Diagram
- CFC: Continuous Function Chart

This chapter provides functions and function blocks representation examples and explains how to use them for IL and ST languages.

### What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Differences Between a Function and a Function Block	48
How to Use a Function or a Function Block in IL Language	49
How to Use a Function or a Function Block in ST Language	53

## Differences Between a Function and a Function Block

### Function

A function:

- is a POU (Program Organization Unit) that returns one immediate result.
- is directly called with its name (not through an instance).
- has no persistent state from one call to the other.
- can be used as an operand in other expressions.

**Examples:** boolean operators (AND), calculations, conversion (BYTE\_TO\_INT)

### Function Block

A function block:

- is a POU (Program Organization Unit) that returns one or more outputs.
- needs to be called by an instance (function block copy with dedicated name and variables).
- each instance has a persistent state (outputs and internal variables) from one call to the other from a function block or a program.

**Examples:** timers, counters

In the example, Timer\_ON is an instance of the function block TON:

```
1  PROGRAM MyProgram_ST
2  VAR
3      Timer_ON: TON; // Function Block Instance
4      Timer_RunCd: BOOL;
5      Timer_PresetValue: TIME := T#5S;
6      Timer_Output: BOOL;
7      Timer_ElapsedTime: TIME;
8  END_VAR

1  Timer_ON(
2      IN:=Timer_RunCd,
3      PT:=Timer_PresetValue,
4      Q=>Timer_Output,
5      ET=>Timer_ElapsedTime);
```



## How to Use a Function or a Function Block in IL Language

### General Information

This part explains how to implement a function and a function block in IL language.

Functions `IsFirstMastCycle` and `SetRTCDrift` and Function Block `TON` are used as examples to show implementations.

### Using a Function in IL Language

This procedure describes how to insert a function in IL language:

Step	Action
1	Open or create a new POU in Instruction List language. <b>NOTE:</b> The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POU's ( <i>see SoMachine, Programming Guide</i> ).
2	Create the variables that the function requires.
3	If the function has 1 or more inputs, start loading the first input using LD instruction.
4	Insert a new line below and: <ul style="list-style-type: none"> <li>type the name of the function in the operator column (left field), or</li> <li>use the <b>Input Assistant</b> to select the function (select <b>Insert Box</b> in the context menu).</li> </ul>
5	If the function has more than 1 input and when Input Assistant is used, the necessary number of lines is automatically created with ??? in the fields on the right. Replace the ??? with the appropriate value or variable that corresponds to the order of inputs.
6	Insert a new line to store the result of the function into the appropriate variable: type ST instruction in the operator column (left field) and the variable name in the field on the right.

To illustrate the procedure, consider the Functions `IsFirstMastCycle` (without input parameter) and `SetRTCDrift` (with input parameters) graphically presented below:

Function	Graphical Representation
without input parameter: <code>IsFirstMastCycle</code>	
with input parameters: <code>SetRTCDrift</code>	

In IL language, the function name is used directly in the operator column:

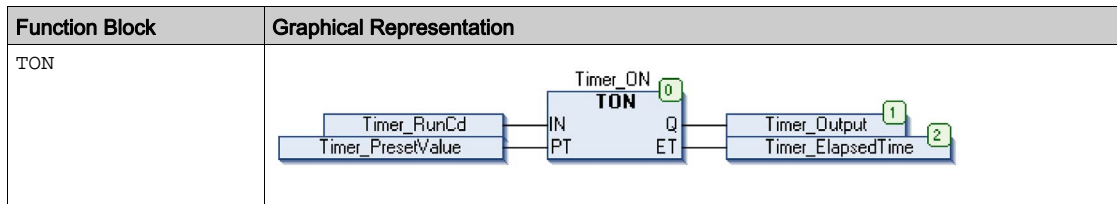
Function	Representation in POU IL Editor															
IL example of a function without input parameter: IsFirstMastCycle	<pre> 1  PROGRAM MyProgram_IL 2  VAR 3      FirstCycle: BOOL; 4  END_VAR                     </pre> <hr/> <table border="1" data-bbox="391 467 975 578"> <tr> <td data-bbox="391 467 426 500">1</td> <td data-bbox="426 467 738 500"><b>IsFirstMastCycle</b></td> <td data-bbox="738 467 975 500"></td> </tr> <tr> <td data-bbox="391 500 426 532"></td> <td data-bbox="426 500 738 532"><b>ST</b></td> <td data-bbox="738 500 975 532">FirstCycle</td> </tr> </table>	1	<b>IsFirstMastCycle</b>			<b>ST</b>	FirstCycle									
1	<b>IsFirstMastCycle</b>															
	<b>ST</b>	FirstCycle														
IL example of a function with input parameters: SetRTCDrift	<pre> 1  PROGRAM MyProgram_IL 2  VAR 3      myDrift: SINT (-29..29) := 5; 4      myDay: DAY_OF_WEEK := SUNDAY; 5      myHour: HOUR := 12; 6      myMinute: MINUTE; 7      myDiag: RTCSETDRIFT_ERROR; 8  END_VAR                     </pre> <hr/> <table border="1" data-bbox="391 987 927 1170"> <tr> <td data-bbox="391 987 426 1019">1</td> <td data-bbox="426 987 683 1019"><b>LD</b></td> <td data-bbox="683 987 927 1019">myDrift</td> </tr> <tr> <td data-bbox="391 1019 426 1052"></td> <td data-bbox="426 1019 683 1052"><b>SetRTCDrift</b></td> <td data-bbox="683 1019 927 1052">myDay</td> </tr> <tr> <td data-bbox="391 1052 426 1084"></td> <td data-bbox="426 1052 683 1084"></td> <td data-bbox="683 1052 927 1084">myHour</td> </tr> <tr> <td data-bbox="391 1084 426 1117"></td> <td data-bbox="426 1084 683 1117"></td> <td data-bbox="683 1084 927 1117">myMinute</td> </tr> <tr> <td data-bbox="391 1117 426 1149"></td> <td data-bbox="426 1117 683 1149"><b>ST</b></td> <td data-bbox="683 1117 927 1149">myDiag</td> </tr> </table>	1	<b>LD</b>	myDrift		<b>SetRTCDrift</b>	myDay			myHour			myMinute		<b>ST</b>	myDiag
1	<b>LD</b>	myDrift														
	<b>SetRTCDrift</b>	myDay														
		myHour														
		myMinute														
	<b>ST</b>	myDiag														

## Using a Function Block in IL Language

This procedure describes how to insert a function block in IL language:

Step	Action
1	Open or create a new POU in Instruction List language. <b>NOTE:</b> The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POU's ( <i>see SoMachine, Programming Guide</i> ).
2	Create the variables that the function block requires, including the instance name.
3	Function Blocks are called using a <b>CAL</b> instruction: <ul style="list-style-type: none"> <li>● Use the <b>Input Assistant</b> to select the FB (right-click and select <b>Insert Box</b> in the context menu).</li> <li>● Automatically, the <b>CAL</b> instruction and the necessary I/O are created.</li> </ul> Each parameter (I/O) is an instruction: <ul style="list-style-type: none"> <li>● Values to inputs are set by " :=".</li> <li>● Values to outputs are set by " =&gt;".</li> </ul>
4	In the <b>CAL</b> right-side field, replace ??? with the instance name.
5	Replace other ??? with an appropriate variable or immediate value.

To illustrate the procedure, consider this example with the **TON** Function Block graphically presented below:



In IL language, the function block name is used directly in the operator column:

Function Block	Representation in POU IL Editor
TON	<pre>1  PROGRAM MyProgram_IL 2  VAR 3      Timer_ON: TON; // Function Block instance declaration 4      Timer_RunCd: BOOL; 5      Timer_PresetValue: TIME := T#5S; 6      Timer_Output: BOOL; 7      Timer_ElapsedTime: TIME; 8  END_VAR 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000</pre>

## How to Use a Function or a Function Block in ST Language

### General Information

This part explains how to implement a Function and a Function Block in ST language.

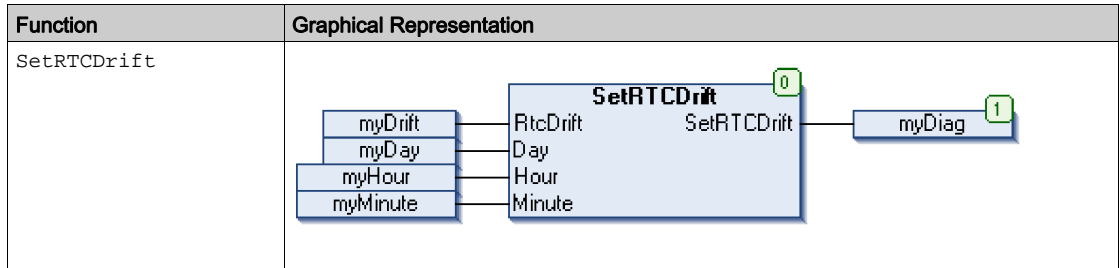
Function `SetRTCDrift` and Function Block `TON` are used as examples to show implementations.

### Using a Function in ST Language

This procedure describes how to insert a function in ST language:

Step	Action
1	Open or create a new POU in Structured Text language. <b>NOTE:</b> The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POU's ( <i>see SoMachine, Programming Guide</i> ).
2	Create the variables that the function requires.
3	Use the general syntax in the <b>POU ST Editor</b> for the ST language of a function. The general syntax is: <code>FunctionResult := FunctionName(VarInput1, VarInput2, .. VarInputx);</code>

To illustrate the procedure, consider the function `SetRTCDrift` graphically presented below:



The ST language of this function is the following:

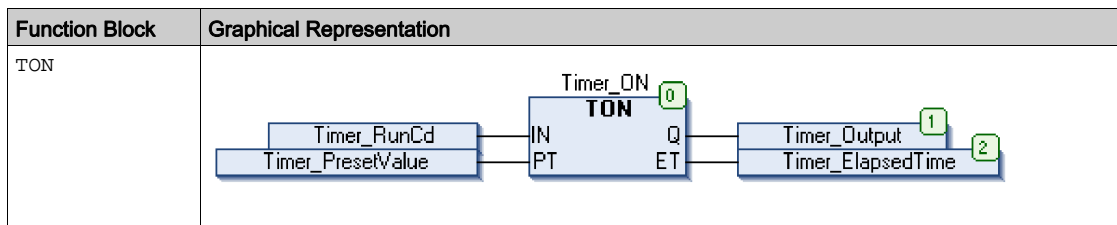
Function	Representation in POU ST Editor
SetRTCDrift	<pre>PROGRAM MyProgram_ST VAR myDrift: SINT(-29..29) := 5; myDay: DAY_OF_WEEK := SUNDAY; myHour: HOUR := 12; myMinute: MINUTE; myRTCAdjust: RTCDRIFT_ERROR; END_VAR myRTCAdjust := SetRTCDrift(myDrift, myDay, myHour, myMinute);</pre>

### Using a Function Block in ST Language

This procedure describes how to insert a function block in ST language:

Step	Action
1	Open or create a new POU in Structured Text language. <b>NOTE:</b> The procedure to create a POU is not detailed here. For more information on adding, declaring and calling POUs, refer to the related documentation (see <i>SoMachine, Programming Guide</i> ).
2	Create the input and output variables and the instance required for the function block: <ul style="list-style-type: none"> <li>• Input variables are the input parameters required by the function block</li> <li>• Output variables receive the value returned by the function block</li> </ul>
3	Use the general syntax in the <b>POU ST Editor</b> for the ST language of a Function Block. The general syntax is: <pre>FunctionBlock_InstanceName ( Input1:=VarInput1, Input2:=VarInput2, ... Ouput1=&gt;VarOutput1, Ouput2=&gt;VarOutput2, ... );</pre>

To illustrate the procedure, consider this example with the TON function block graphically presented below:



This table shows examples of a function block call in ST language:

Function Block	Representation in POU ST Editor
TON	<pre> 1  PROGRAM MyProgram_ST 2  VAR 3      Timer_ON: TON; // Function Block Instance 4      Timer_RunCd: BOOL; 5      Timer_PresetValue: TIME := T#5S; 6      Timer_Output: BOOL; 7      Timer_ElapsedTime: TIME; 8  END_VAR  Timer_ON(     IN:=Timer_RunCd,     PT:=Timer_PresetValue,     Q=&gt;Timer_Output,     ET=&gt;Timer_ElapsedTime); </pre>





---

# Glossary

---



## A

### **application**

A program including configuration data, symbols, and documentation.

## B

### **byte**

A type that is encoded in an 8-bit format, ranging from 00 hex to FF hex.

## C

### **CFC**

*(continuous function chart)* A graphical programming language (an extension of the IEC 61131-3 standard) based on the function block diagram language that works like a flowchart. However, no networks are used and free positioning of graphic elements is possible, which allows feedback loops. For each block, the inputs are on the left and the outputs on the right. You can link the block outputs to the inputs of other blocks to create complex expressions.

### **configuration**

The arrangement and interconnection of hardware components within a system and the hardware and software parameters that determine the operating characteristics of the system.

### **controller**

Automates industrial processes (also known as programmable logic controller or programmable controller).

## E

### **expansion bus**

An electronic communication bus between expansion I/O modules and a controller.

## F

### **FB**

*(function block)* A convenient programming mechanism that consolidates a group of programming instructions to perform a specific and normalized action, such as speed control, interval control, or counting. A function block may comprise configuration data, a set of internal or external operating parameters and usually 1 or more data inputs and outputs.

**function block diagram**

One of the 5 languages for logic or control supported by the standard IEC 61131-3 for control systems. Function block diagram is a graphically oriented programming language. It works with a list of networks where each network contains a graphical structure of boxes and connection lines representing either a logical or arithmetic expression, the call of a function block, a jump, or a return instruction.

I

**I/O**

*(input/output)*

**IL**

*(instruction list)* A program written in the language that is composed of a series of text-based instructions executed sequentially by the controller. Each instruction includes a line number, an instruction code, and an operand (refer to IEC 61131-3).

**INT**

*(integer)* A whole number encoded in 16 bits.

L

**LD**

*(ladder diagram)* A graphical representation of the instructions of a controller program with symbols for contacts, coils, and blocks in a series of rungs executed sequentially by a controller (refer to IEC 61131-3).

P

**POU**

*(program organization unit)* A variable declaration in source code and a corresponding instruction set. POU's facilitate the modular re-use of software programs, functions, and function blocks. Once declared, POU's are available to one another.

**program**

The component of an application that consists of compiled source code capable of being installed in the memory of a logic controller.

S

**ST**

*(structured text)* A language that includes complex statements and nested instructions (such as iteration loops, conditional executions, or functions). ST is compliant with IEC 61131-3.

## U

### UTC

*(universal time coordinated)* The primary time standard by which the world regulates clocks and time.

## V

### variable

A memory unit that is addressed and modified by a program.





## C

converting

- `SysTimeRtcConvertDateToUtc`, 32
- `SysTimeRtcConvertUtcToDate`, 34

## D

data and time of the controller

getting, 15

data types

- `RTS_SYSTIMEDATE`, 43
- `SYSTIMEDATE`, 43
- `SysTimeRtc`, 42
- `TimezoneInformation`, 44

date

- `SysTimeRtcConvertDateToUtc`, 32
- `SysTimeRtcConvertUtcToDate`, 34

date and time of the controller

setting, 17

## E

examples

- getting the controller data and time, 15
- setting data and time of the controller, 17

## F

functions

- differences between a function and a function block, 48
- how to use a function or a function block in IL language, 49
- how to use a function or a function block in ST language, 53
- `SysTimeCore`, 20
- `SysTimeRtc`, 26

## G

getting controller RTC, 15

examples, 15

## H

high resolution time stamp

`SysTimeRtc`, 26

## R

real time clock

- `SysTimeRtcControl`, 27
- `SysTimeRtcConvertDateToHighRes`, 33
- `SysTimeRtcConvertHighResToDate`, 35
- `SysTimeRtcConvertHighResToLocal`, 41
- `SysTimeRtcConvertLocalToHighRes`, 39
- `SysTimeRtcGet`, 28
- `SysTimeRtcGetTimezone`, 36
- `SysTimeRtcHighResGet`, 29
- `SysTimeRtcHighResSet`, 31
- `SysTimeRtcSet`, 30
- `SysTimeRtcSetTimezone`, 37

RTC

- `SysTimeRtcControl`, 27
- `SysTimeRtcConvertDateToHighRes`, 33
- `SysTimeRtcConvertHighResToDate`, 35
- `SysTimeRtcConvertHighResToLocal`, 41
- `SysTimeRtcConvertLocalToHighRes`, 39
- `SysTimeRtcGet`, 28
- `SysTimeRtcHighResGet`, 29
- `SysTimeRtcHighResSet`, 31

RTC (real time clock), 13

- `SysTimeRtcGetTimezone`, 36
- `SysTimeRtcSet`, 30
- `SysTimeRtcSetTimezone`, 37

`RTS_SYSTIMEDATE`, 43

**S**

- setting controller RTC, *17*
  - examples, *17*
- standard resolution time stamp
  - SysTimeRtc, *26*
- SysTime functions
  - SysTimeCore library, *13*
  - SysTimeRtc library, *13*
- SysTimeCore
  - functions, *20*
  - general information, *11*
  - SysTimeGetMs, *21*
  - SysTimeGetNs, *22*
  - SysTimeGetUs, *23*
  - time meters, *20*
- SYSTIMEDATE, *43*
- SysTimeGetMs, *21*
- SysTimeGetNs, *22*
- SysTimeGetUs, *23*
- SysTimeRtc
  - functions, *26*
  - general information, *11*
  - RTS\_SYSTIMEDATE, *43*
  - SYSTIMEDATE, *43*
  - SysTimeRtcControl, *27*
  - SysTimeRtcConvertDateToHighRes, *33*
  - SysTimeRtcConvertDateToUtc, *32*
  - SysTimeRtcConvertHighResToDate, *35*
  - SysTimeRtcConvertHighResToLocal, *41*
  - SysTimeRtcConvertLocalToHighRes, *39*
  - SysTimeRtcConvertUtcToDate, *34*
  - SysTimeRtcConvertUtcToLocal, *40*
  - SysTimeRtcGet, *28*
  - SysTimeRtcGetTimezone, *36*
  - SysTimeRtcHighResGet, *29*
  - SysTimeRtcHighResSet, *31*
  - SysTimeRtcSet, *30*
  - SysTimeRtcSetTimezone, *37*
  - TimezoneInformation, *44*
- SysTimeRtcControl, *27*
- SysTimeRtcConvertDateToHighRes, *33*
- SysTimeRtcConvertDateToUtc, *32*
- SysTimeRtcConvertHighResToDate, *35*
- SysTimeRtcConvertHighResToLocal, *41*
- SysTimeRtcConvertLocalToHighRes, *39*

- SysTimeRtcConvertLocalToUtc, *38*
  - SysTimeRtcConvertLocalToUtc, *38*
- SysTimeRtcConvertUtcToDate, *34*
- SysTimeRtcConvertUtcToLocal, *40*
- SysTimeRtcGet, *28*
- SysTimeRtcGetTimezone, *36*
- SysTimeRtcHighResGet, *29*
- SysTimeRtcHighResSet, *31*
- SysTimeRtcSet, *30*
- SysTimeRtcSetTimezone, *37*

**T**

- time
  - SysTimeGetMs, *21*
  - SysTimeGetNs, *22*
  - SysTimeGetUs, *23*
- time meters
  - SysTimeCore, *20*
- time stamp
  - SysTimeRtc, *26*
  - SysTimeRtcConvertDateToUtc, *32*
  - SysTimeRtcConvertUtcToDate, *34*
- TimezoneInformation, *44*

**U**

- universal time coordinated (UTC)
  - SysTimeRtcConvertLocalToUtc, *38*
  - SysTimeRtcConvertUtcToLocal, *40*
- UTC
  - SysTimeRtcConvertLocalToUtc, *38*
  - SysTimeRtcConvertUtcToDate, *34*
  - SysTimeRtcConvertUtcToLocal, *40*
- UTC (universal time coordinated), *13*