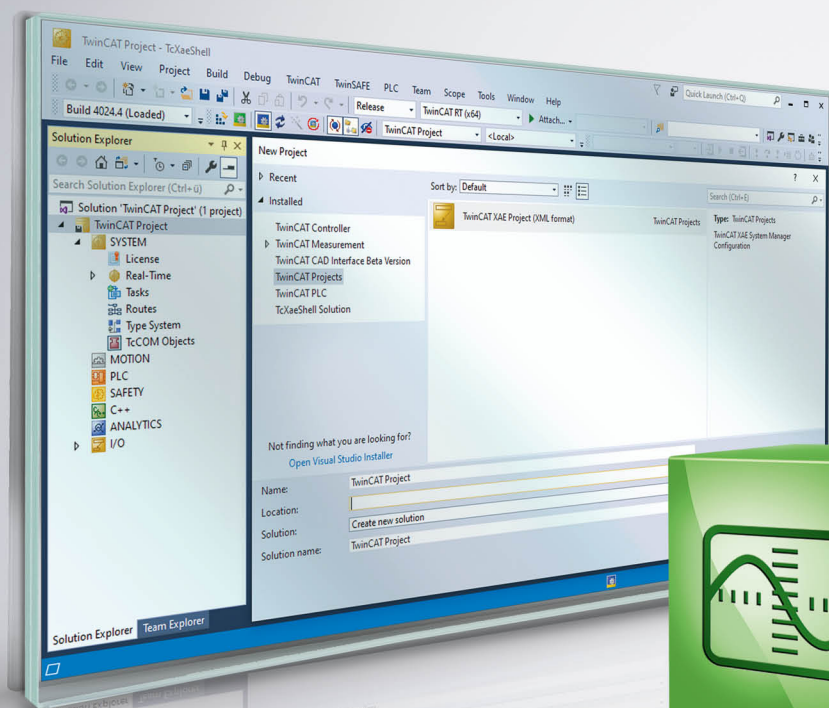


Manual | EN

## TF3900

TwinCAT 3 | Solar Position Algorithm





# Table of contents

**1 Foreword ..... 5**

1.1 Notes on the documentation..... 5

1.2 Safety instructions ..... 6

**2 Overview..... 7**

2.1 Copyright ..... 9

**3 Installation..... 10**

3.1 System requirements..... 10

3.2 Installation ..... 10

3.3 Licensing ..... 13

**4 API..... 16**

4.1 PLC Reference ..... 16

4.1.1 Function blocks..... 16

4.1.2 Data types..... 19

4.1.3 Global constants ..... 21

**5 Example..... 22**



# 1 Foreword

## 1.1 Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with applicable national standards.

It is essential that the documentation and the following notes and explanations are followed when installing and commissioning the components.

It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

### Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.

We reserve the right to revise and change the documentation at any time and without prior announcement. No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

### Trademarks

Beckhoff®, TwinCAT®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.

Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

### Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:

EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702

with corresponding applications or registrations in various other countries.



EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

### Copyright

© Beckhoff Automation GmbH & Co. KG, Germany.

The reproduction, distribution and utilization of this document as well as the communication of its contents to others without express authorization are prohibited.

Offenders will be held liable for the payment of damages. All rights reserved in the event of the grant of a patent, utility model or design.

## 1.2 Safety instructions

### Safety regulations

Please note the following safety instructions and explanations!

Product-specific safety instructions can be found on following pages or in the areas mounting, wiring, commissioning etc.

### Exclusion of liability

All the components are supplied in particular hardware and software configurations appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

### Personnel qualification

This description is only intended for trained specialists in control, automation and drive engineering who are familiar with the applicable national standards.

### Description of symbols

In this documentation the following symbols are used with an accompanying safety instruction or note. The safety instructions must be read carefully and followed without fail!

#### **DANGER**

##### **Serious risk of injury!**

Failure to follow the safety instructions associated with this symbol directly endangers the life and health of persons.

#### **WARNING**

##### **Risk of injury!**

Failure to follow the safety instructions associated with this symbol endangers the life and health of persons.

#### **CAUTION**

##### **Personal injuries!**

Failure to follow the safety instructions associated with this symbol can lead to injuries to persons.

#### **NOTE**

##### **Damage to the environment or devices**

Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.



#### **Tip or pointer**

This symbol indicates information that contributes to better understanding.

## 2 Overview

The TwinCAT PLC Solar Position Algorithm library (SPA) offers an option for calculating the sun position exactly at almost any time.

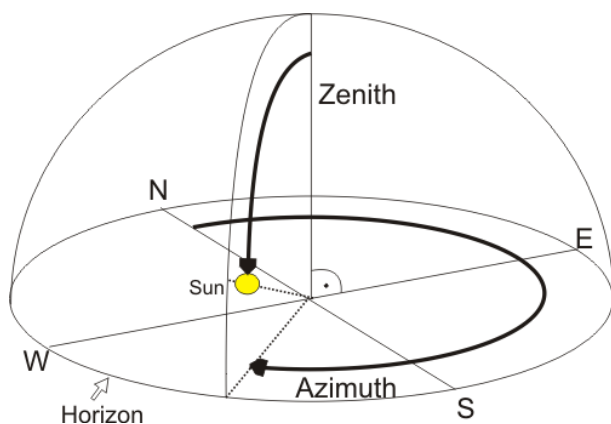
The times for sunrise, solar apex and sunset can also be determined.

In addition to the sun angles an angle of incidence can be issued, if the point of reference has a certain inclination. The sun angles themselves refer to the horizontal at the point of reference.

The algorithm is based on a technical report by the U.S. National Renewable Energy Laboratory (NREL). The theoretical inaccuracy of the sun angles between the year -2000 and 6000 is specified as  $\pm 0.0003^\circ$ . Based on this the function block of the TwinCAT Solar Position Algorithm library assumes an inaccuracy of  $\pm 0.001^\circ$  for the sun angles.

### Sun angles

The position of the sun at a fixed observation point is normally determined by specifying two angles. In order to calculate the sun angles using the TwinCAT Solar Position Algorithm library, the date, time, longitude, latitude and further parameters have to be specified, depending on the required accuracy. The graphic illustrates the meaning of the main terms in this context:



The sun position represented by two angles.

- Zenith** The zenith angle of the sun is defined as the angle between the vertical above the observer and the connecting line between the observer and the sun. In some cases the altitude is used to indicate the sun elevation angle. The following applies:  $90^\circ - \text{zenith angle} = \text{altitude}$
- Azimuth** The azimuth coincides with the horizon. North is  $0^\circ$ , with the value increasing in clockwise direction (east =  $90^\circ$ , south =  $180^\circ$ , west =  $270^\circ$ ).

### Longitude and latitude

The latitude is specified as the distance of a place on the surface of the earth from the equator to the north or to the south in degrees. The latitude can assume a value from  $0^\circ$  (at the equator) to  $\pm 90^\circ$  (at the poles). A positive sign thereby indicates a northern direction and a negative sign a southern direction. The longitude is an angle that can assume values up to  $\pm 180^\circ$  starting from the prime meridian  $0^\circ$  (an artificially determined North-South line). A positive sign indicates a longitude in an eastern direction and a negative sign in a western direction. Examples:

Place	Longitude	Latitude
Sydney, Australia	151.2°	-33.9°
New York, USA	-74.0°	40.7°
London, England	-0.1°	51.5°
Moscow, Russia	37.6°	55.7°
Peking, China	116.3°	39.9°
Dubai, United Arab Emirates	55.3°	25.4°
Rio de Janeiro, Brazil	-43.2°	-22.9°
Hawaii, USA	-155.8°	20.2°
Verl, Germany	8.5°	51.9°

## Time scale

Specification of the correct time is particularly important. Various time scales are in use. The Solar Position Algorithm is based on Universal Time (UT1).

### Universal Time (UT1)

Between 1928 and 1968 was the UT was the accepted world time. It is also referred to as universal solar time. It is determined through astronomic observation of the angle of rotation of the earth and corresponds to the mean local time of the observatory at Greenwich (prime meridian). This parameter is derived from the earth's rotation and takes into account fluctuations and long-term slowdown and is therefore not strictly a uniform measure of time. On the other hand, it is always synchronised with the actual change-over between day and night.

### International Atomic Time (TAI)

The International Atomic Time is specified by more than 50 time institutes worldwide, based on their atomic clocks. An atomic time is based on an atomic standard time that can be assumed to be exactly uniform.

### Coordinated Universal Time (UTC)

The coordinated world time UTC has been used as the standard world time since 1968. This is the time referred to by GMT in everyday usage. Greenwich Mean Time (GMT) was the original world time before 1928.

UTC continues to use the observatory at Greenwich (prime meridian) as point of reference. The earth's time zones are derived from the coordinated world time ( $UTC+1$  = Central European Time). In contrast to UT1, its second cycle matches the exactly uniform second cycle of the International Atomic Time (TAI). Leap seconds are used to compensate the difference between UTC and UT1. The difference between the UT1 reference time is always less than one second.

The coordinated world time UTC is therefore a compromise between UT1 and TAI.

The following formula is used to convert a time from UTC to UT1:  $UT1 = UTC + DUT1$

### Terrestrial Time (TT)

Also referred to as Terrestrial Dynamical Time (TDT). This time is used as the basis for calculating astronomic events and is based on the exactly uniform seconds of the International Atomic Time (TAI). The following applies:  $TT = TAI + 32.184$

### Leap Seconds

To synchronise the coordinated world time UTC with UT1, a leap second is added when required. This additional second is specified by the International Earth Rotation and Reference Systems Service (IERS) at irregular, non-predictable intervals. It ensures that the difference between the two time scales is always less than one second. (In the past such additional leap seconds have always been added on 31 December or 30 June after 23:59:59 UTC.)

DUT1 denotes the remaining difference. The following applies:  $DUT1 = UT1 - UTC$

This value is derived from observations that are continuously reported.

### Delta T



Delta T is the difference between Terrestrial Time and Universal Time. The following applies:  $\Delta t = TT - UT1$

This parameter can be specified as  $f\Delta t$  at the input for function block `FB_SPA` [► 16]. It is derived from observations that are continuously reported. A standard value is 66 seconds.

## 2.1 Copyright

The algorithm is based on the technical report "Solar Position Algorithm for Solar Radiation Application" by I. Reda & A. Andreas, National Renewable Energy Laboratory (NREL), USA (revision 14-JAN-2009).

### NOTICE

Copyright (C) 2007 Alliance for Sustainable Energy, LLC, All Rights Reserved

This computer software was developed by the Alliance for Sustainable Energy, LLC, hereinafter the Contractor, under Contract DE-AC36-08GO28308 (Contract) with the Department of Energy (DOE). The United States Government has been granted for itself and others acting on its behalf a paid-up, non-exclusive, irrevocable, worldwide license in the Software to reproduce, prepare derivative works, and perform publicly and display publicly. Beginning five (5) years after the date permission to assert copyright is obtained from the DOE, and subject to any subsequent five (5) year renewals, the United States Government is granted for itself and others acting on its behalf a paid-up, non-exclusive, irrevocable, worldwide license in the Software to reproduce, prepare derivative works, distribute copies to the public, perform publicly and display publicly, and to permit others to do so. If the Contractor ceases to make this computer software available, it may be obtained from DOE's Office of Scientific and Technical Information's Energy Science and Technology Software Center (ESTSC) at P.O.Box 62, 1 Science Gov Way, Oak Ridge, TN 37831-1020. THIS SOFTWARE IS PROVIDED BY THE CONTRACTOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE CONTRACTOR OR THE U.S. GOVERNMENT BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER, INCLUDING BUT NOT LIMITED TO CLAIMS ASSOCIATED WITH THE LOSS OF DATA OR PROFITS, WHICH MAY RESULT FROM AN ACTION IN CONTRACT, NEGLIGENCE OR OTHER TORTIOUS CLAIM THAT ARISES OUT OF OR IN CONNECTION WITH THE ACCESS, USE OR PERFORMANCE OF THIS SOFTWARE.

## 3 Installation

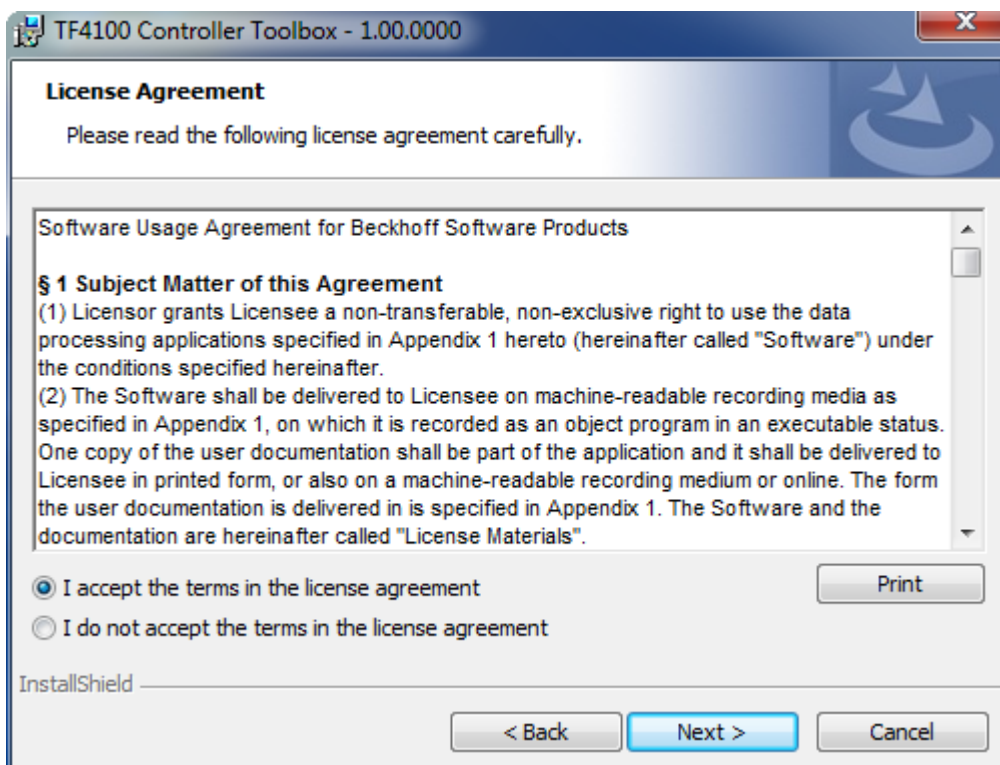
### 3.1 System requirements

- Programming environment:
  - XP, XPe, WES, Win7, WES7;
  - TwinCAT installation: TwinCAT XAE TC3 PLC;
  - TwinCAT System version 3.1.4011 or higher;
  - **Tc2\_SPA** This PLC library must be integrated in the PLC project.
- Target platform
  - PC or CX (x86, x64, ARM): XP, XPe, WES, Win7, WES7, CE7;
  - **Note** In systems without a floating point unit the performance is limited due to the complex internal calculations. In the event of anomalies the cycle time should be checked.
  - TwinCAT PLC runtime system version 3.1.4011 or higher;

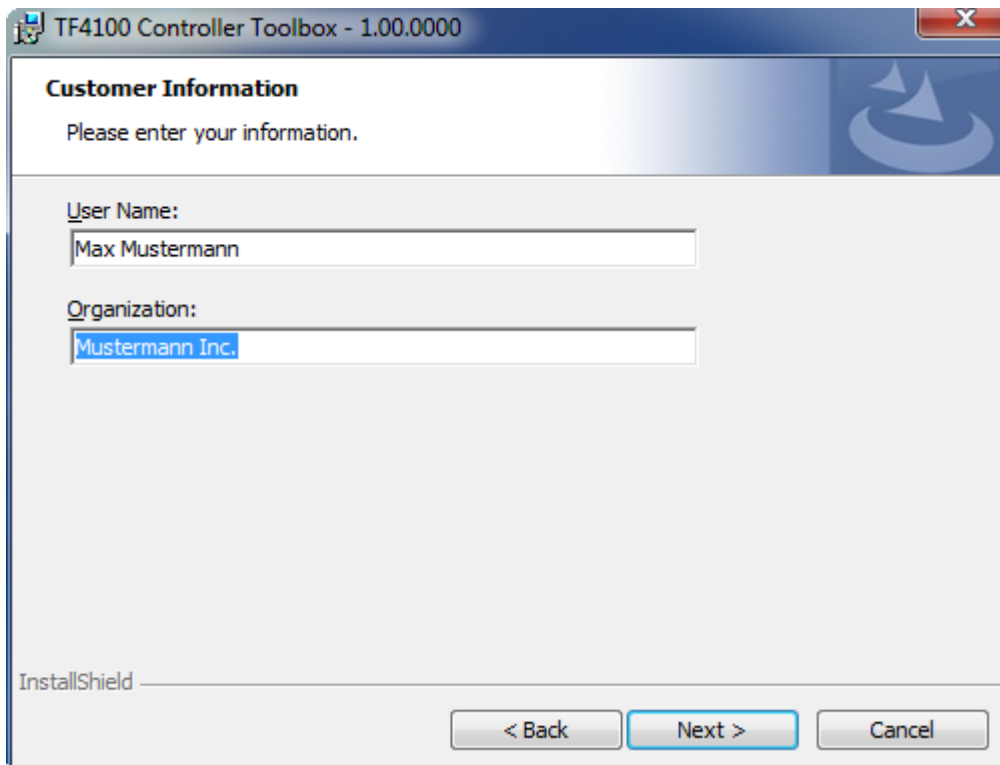
### 3.2 Installation

The following section describes how to install the TwinCAT 3 Function for Windows-based operating systems.

- ✓ The TwinCAT 3 Function setup file was downloaded from the Beckhoff website.
1. Run the setup file as administrator. To do this, select the command **Run as administrator** in the context menu of the file.
    - ⇒ The installation dialog opens.
  2. Accept the end user licensing agreement and click **Next**.



3. Enter your user data.



TF4100 Controller Toolbox - 1.00.0000

**Customer Information**

Please enter your information.

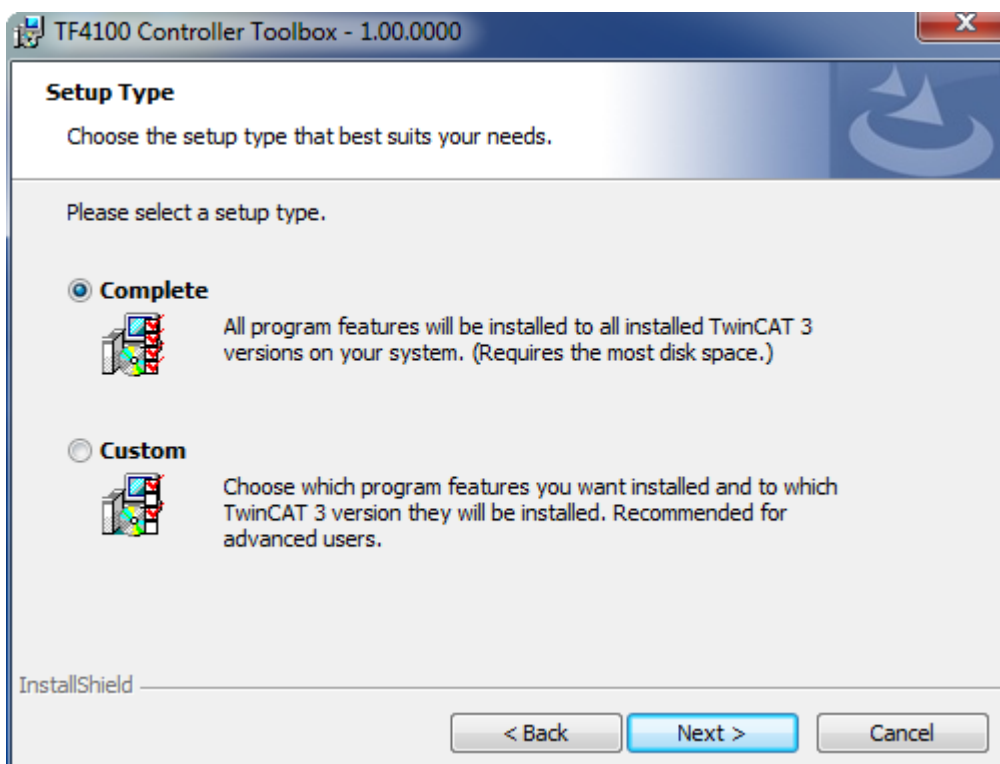
User Name:  
Max Mustermann

Organization:  
Mustermann Inc.

InstallShield

< Back   Next >   Cancel

4. If you want to install the full version of the TwinCAT 3 Function, select **Complete** as installation type. If you want to install the TwinCAT 3 Function components separately, select **Custom**.



TF4100 Controller Toolbox - 1.00.0000

**Setup Type**

Choose the setup type that best suits your needs.

Please select a setup type.

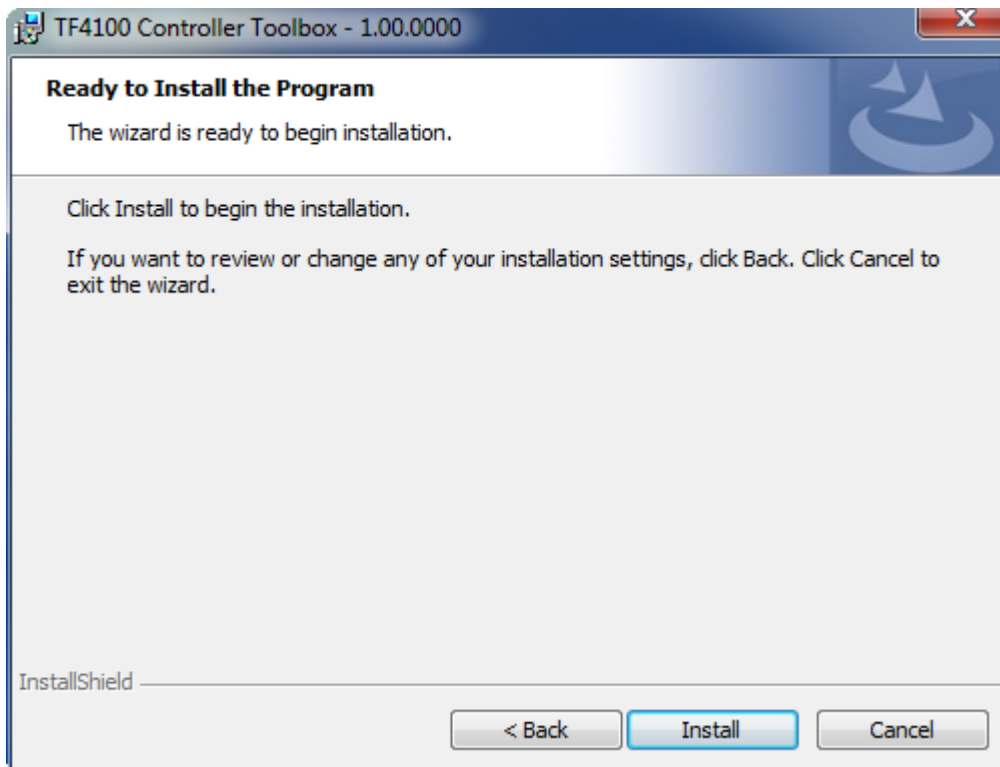
☒ **Complete**  
All program features will be installed to all installed TwinCAT 3 versions on your system. (Requires the most disk space.)

☐ **Custom**  
Choose which program features you want installed and to which TwinCAT 3 version they will be installed. Recommended for advanced users.

InstallShield

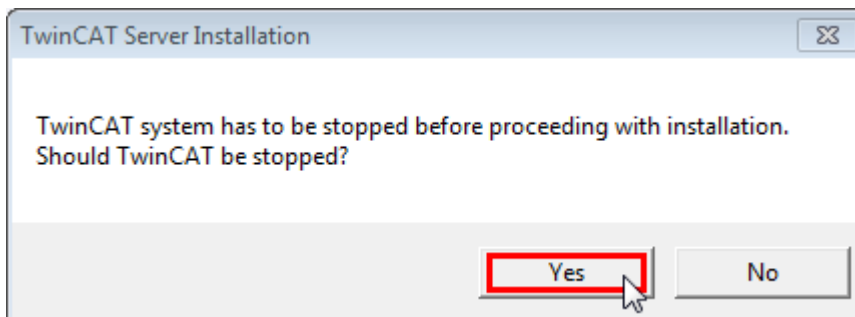
< Back   Next >   Cancel

5. Select **Next**, then **Install** to start the installation.

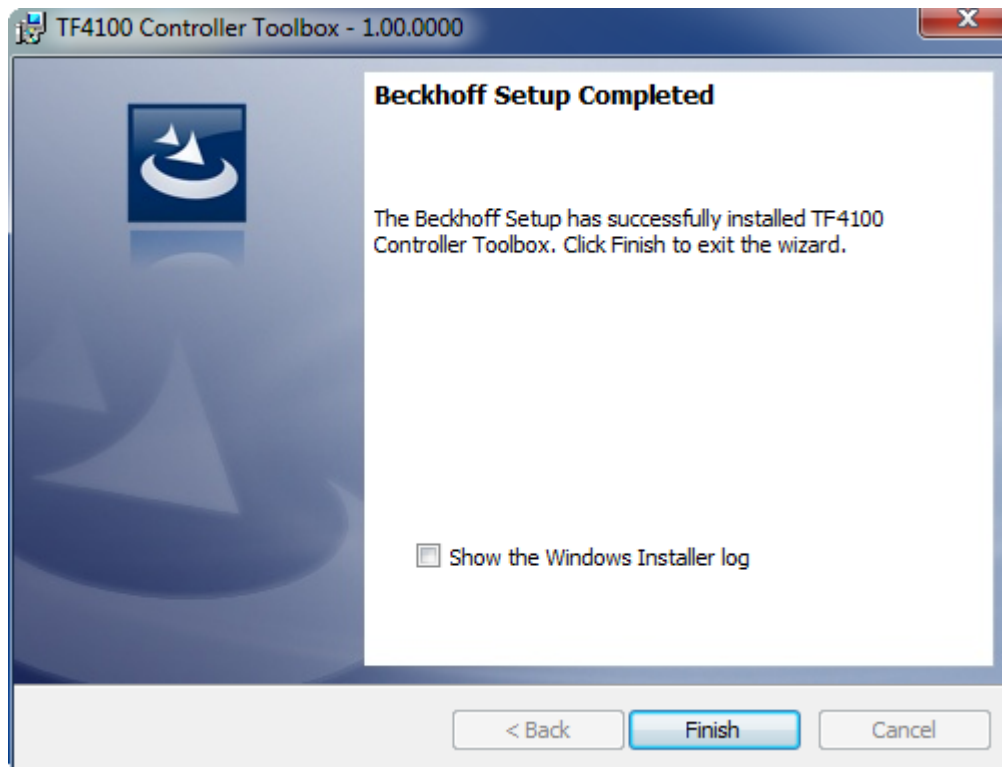


⇒ A dialog box informs you that the TwinCAT system must be stopped to proceed with the installation.

6. Confirm the dialog with **Yes**.



7. Select **Finish** to exit the setup.



⇒ The TwinCAT 3 Function has been successfully installed and can be licensed (see [Licensing](#) [► 13]).

### 3.3 Licensing

The TwinCAT 3 function can be activated as a full version or as a 7-day test version. Both license types can be activated via the TwinCAT 3 development environment (XAE).

#### Licensing the full version of a TwinCAT 3 Function

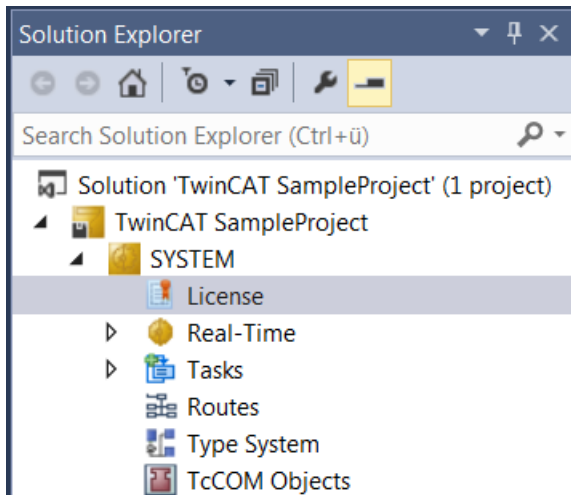
A description of the procedure to license a full version can be found in the Beckhoff Information System in the documentation "[TwinCAT 3 Licensing](#)".

#### Licensing the 7-day test version of a TwinCAT 3 Function

**i** A 7-day test version cannot be enabled for a TwinCAT 3 license dongle.

1. Start the TwinCAT 3 development environment (XAE).
2. Open an existing TwinCAT 3 project or create a new project.
3. If you want to activate the license for a remote device, set the desired target system. To do this, select the target system from the **Choose Target System** drop-down list in the toolbar.
  - ⇒ The licensing settings always refer to the selected target system. When the project is activated on the target system, the corresponding TwinCAT 3 licenses are automatically copied to this system.

4. In the **Solution Explorer**, double-click **License** in the **SYSTEM** subtree.



⇒ The TwinCAT 3 license manager opens.

5. Open the **Manage Licenses** tab. In the **Add License** column, check the check box for the license you want to add to your project (e.g. "TF6420: TC3 Database Server").

Order Information (Runtime) **Manage Licenses** Project Licenses Online Licenses

☐ Disable automatic detection of required licenses for project

Order No	License	Add License
TF3601	TC3 Condition Monitoring Level 2	<input type="checkbox"/> cpu license
TF3650	TC3 Power Monitoring	<input type="checkbox"/> cpu license
TF3680	TC3 Filter	<input type="checkbox"/> cpu license
TF3800	TC3 Machine Learning Inference Engine	<input type="checkbox"/> cpu license
TF3810	TC3 Neural Network Inference Engine	<input type="checkbox"/> cpu license
TF3900	TC3 Solar-Position-Algorithm	<input type="checkbox"/> cpu license
TF4100	TC3 Controller Toolbox	<input checked="" type="checkbox"/> cpu license
TF4110	TC3 Temperature-Controller	<input type="checkbox"/> cpu license
TF4500	TC3 Speech	<input type="checkbox"/> cpu license

6. Open the **Order Information (Runtime)** tab.

⇒ In the tabular overview of licenses, the previously selected license is displayed with the status "missing".

7. Click **7-Day Trial License...** to activate the 7-day trial license.

Order Information (Runtime) | Manage Licenses | Project Licenses | Online Licenses

License Device: Target (Hardware Id) [Add...]

System Id: 27A8E382-5115-364F-CE96-DE17942299A5 | Platform: other (91)

License Request

Provider: Beckhoff Automation [Generate File...]

License Id: | Customer Id: |

Comment: |

License Activation

**7 Days Trial License...** | License Response File...

⇒ A dialog box opens, prompting you to enter the security code displayed in the dialog.

Enter Security Code [X]

Please type the following 5 characters:

Kg8T4

[ ] [ ]

OK | Cancel

8. Enter the code exactly as it is displayed and confirm the entry.
9. Confirm the subsequent dialog, which indicates the successful activation.
- ⇒ In the tabular overview of licenses, the license status now indicates the expiry date of the license.
10. Restart the TwinCAT system.
- ⇒ The 7-day trial version is enabled.

## 4 API

### 4.1 PLC Reference

#### 4.1.1 Function blocks

##### 4.1.1.1 FB\_SPA

FB_SPA	
stTime : ST_SPA_TIMESTRUCT	fZenith : LREAL
fTimezone : LREAL	fAzimuth : LREAL
fDelta_t : LREAL	fAzimuth180 : LREAL
fLongitude : LREAL	fIncidence : LREAL
fLatitude : LREAL	fSuntransit : LREAL
fElevation : LREAL	fSunrise : LREAL
fPressure : LREAL	fSunset : LREAL
fTemperature : LREAL	bError : BOOL
fSlope : LREAL	iErrorCode : UINT
fAzm_rotation : LREAL	
fAtmos_refract : LREAL	
eFunction : E_SPA_FunctionCode	

At the input all available values for the location definition and type of calculation are specified.

The calculation is performed during a function block cycle. The results are immediately available at the output.

Due to the complex internal calculation steps processing takes system performance.

#### VAR\_INPUT

```

VAR_INPUT
    stTime      :ST_SPA_TIMESTRUCT; (* local date and time (year, month, day, hour, minute,
second) *)
    fTimezone   :LREAL;              (* Observer time zone (negative west of Greenwich)      *)
                                      (* valid range: -18 TO 18 hours, error code: 8          *)
    fDelta_t    :LREAL:=66;          (* Difference between earth rotation time and terrestrial time *)
                                      (* It is derived from observation only and is reported in this *)
                                      (* bulletin: http://maia.usno.navy.mil/ser7/ser7.dat,          *)
                                      (* where delta_t = 32.184 + (TAI-UTC) + DUT1                      *)
                                      (* valid range: -8000 to 8000 seconds, error code: 7          *)
    fLongitude  :LREAL;              (* Observer longitude (negative west of Greenwich)      *)
                                      (* valid range: -180 to 180 degrees, error code: 9          *)
    fLatitude   :LREAL;              (* Observer latitude (negative south of equator)         *)
                                      (* valid range: -90 to 90 degrees, error code: 10          *)
    fElevation  :LREAL;              (* Observer elevation [meters]                          *)
                                      (* valid range: -6500000 or higher meters, error code: 11 *)
    fPressure   :LREAL:=1000;        (* Annual average local pressure [millibars]             *)
                                      (* valid range: 0 to 5000 millibars, error code: 12          *)
    fTemperature:LREAL;              (* Annual average local temperature [degrees Celsius]    *)
                                      (* valid range: -273 to 6000 degrees Celsius, error code; 13 *)
    fSlope      :LREAL;              (* Surface slope (measured from the horizontal plane)    *)
                                      (* valid range: -360 to 360 degrees, error code: 14          *)
    fAzm_rotation:LREAL;            (* Surface azimuth rotation (measured from south to projection of
*)
                                      (* surface normal on horizontal plane, negative west)      *)
                                      (* valid range: -360 to 360 degrees, error code: 15          *)
    fAtmos_refract:LREAL:=0.5667;    (* Atmospheric refraction at sunrise and sunset (0.5667 deg is
typ.)*)
                                      (* valid range: -5 to 5 degrees, error code: 16          *)
    eFunction   :E_SPA_FunctionCode:=eSPA_ZA; (* Switch to choose functions for desired
output *)
END_VAR

```



**stTime**

The date and the local time are specified via `stTime`. This structure is of type `ST_SPA_TIMESTRUCT` [► 19].

**fTimezone**

The required date with the corresponding time can be specified in local time via the above variable. The respective time zone is added via `fTimezone`. The time zone is always based on Greenwich (London). (The prime meridian, i.e. 0° geographic longitude, also passes through Greenwich). The following applies in relation to the coordinated world time:

UTC+1 = Central European Time;

UTC+2 = Central European Summer Time.

**fDelta\_t**

The input variable `fDelta_t` is used for balancing the time scales used. A standard value is 66. A more detailed description of the different time scales can be found on the [overview page](#) [► 7].

**fLongitude**

`fLongitude` indicates the longitude in degrees [°]. It is positive to the east of Greenwich.

**fLatitude**

`fLatitude` indicates the latitude in degrees [°]. It is positive to the north of the equator and negative to the south.

**fElevation**

The altitude of the location also has a small effect on the calculation of the sun angles. `fElevation` indicates the height in metres above mean sea level.

**fPressure**

The atmospheric pressure at the location is specified in millibar [mbar] via the input variable `fPressure`. The annual average is specified.

**fTemperature**

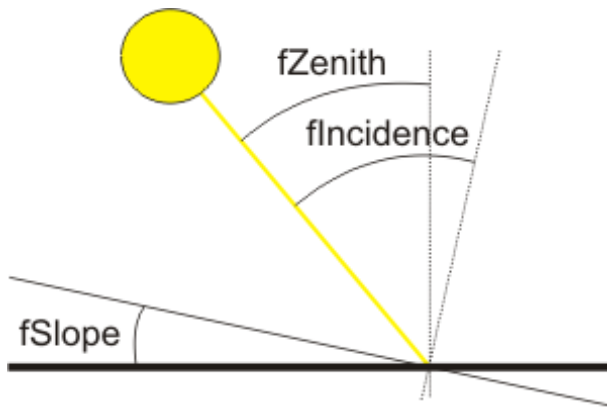
The temperature at the location is specified via the input variable `fTemperature` in °C. The annual average is specified.

**fSlope**

Via `fSlope` a surface inclination can be specified in degrees [°]. It is used for calculating the special angle of incidence `fIncidence`. If `fSlope` is zero, the angle of incidence is the same as the zenith angle.

**fAzm\_rotation**

`fAzm_rotation` can be used to adjust the alignment (in degrees [°]) of the observer or the surface inclined by `fSlope`. For north alignment the value is 0°. From there the alignment angle increases clockwise (positive values, as does the azimuth of the sun angle). It is also used for calculating the special angle of incidence `fIncidence`. If `fSlope` is zero, the angle of incidence is the same as the zenith angle, irrespective of `fAzm_rotation`. If `fAzm_rotation` is the same as the sun angle `fAzimuth`, the following applies:  $fIncidence = fZenith + fSlope$ . This is illustrated the following 2D diagram.



### fAtmos\_refract

Refraction in the atmosphere can have a significant effect on the zenith angle of the sun, particularly for shallow sun angles. The input variable *fAtmos\_refract* is used as a correction factor for the atmospheric distraction at sunrise and sunset. A standard value is 0.5667.

### eFunction

Via this enumeration value ([E\\_SPA\\_FunctionCode](#) [► 20]) the type of calculations can be selected. For example, the calculation can be limited to the sun angles, if information on sunrise etc. is not required.

### VAR\_OUTPUT

```
VAR_OUTPUT
  fZenith      :LREAL;      (* topocentric zenith angle [degrees] *)
  fAzimuth     :LREAL;      (* topocentric azimuth angle (eastward from north) [ 0 to 360 degrees] *)
  fAzimuth180  :LREAL;      (* topocentric azimuth angle (westward from south) [-180 to 180 degrees] *)
  fIncidence   :LREAL;      (* surface incidence angle [degrees] *)
  fSuntransit  :LREAL;      (* local sun transit time (or solar noon) [fractional hour] *)
  fSunrise     :LREAL;      (* local sunrise time (+/- 30 seconds) [fractional hour] *)
  fSunset      :LREAL;      (* local sunset TIME (+/- 30 seconds) [fractional hour] *)
  bError       :BOOL;       (* error flag *)
  iErrorCode   :UINT;       (* error code *)
END_VAR
```

### fZenith

The zenith angle of the sun is defined as the angle between the vertical above the observer (zenith) and the connecting line between the observer and the sun. If the sun is directly vertical above the observer, the zenith angle is 0°.

Sometimes also the sun elevation angle (or altitude) is common. The following applies: 90° - zenith angle = altitude.

### fAzimuth

The azimuth coincides with the horizon. North is 0°, with the value increasing in clockwise direction (east = 90°, south=180°, west=270°). A diagrammatic illustration of the sun angles can be found on the [overview page](#) [► 7].

### fAzimuth180

This value has the same meaning of the azimuth, although with azimuth180 the value 0° is allocated to south. From there the value increases positively in clockwise direction and negatively in counter-clockwise direction (azimuth-180° = azimuth180)

### fIncidence

fIncidence indicates the angle of solar incidence in relation to the surface specified at the input. If the surface is horizontal *fIncidence* matches the value of *fZenith*.

## fSuntransit

fSuntransit indicates the time of the solar apex. It is specified in hours and used the time zone created at the input.

The following typecasting can be used for converting the variable fSuntransit (same procedure for fSunrise and fSunset) to time format:

```
tSuntransit := LREAL_TO_TIME(fbSPA.fSuntransit*60*60*1000);
```

## fSunrise

fSunrise indicates the sunrise time. It is specified in hours and used the time zone created at the input.

## fSunset

fSunset indicates the sunset time. It is specified in hours and used the time zone created at the input.

## bError

bError is TRUE if an error has occurred. In this case *iErrorCode* indicates the respective error code

## iErrorCode

iErrorCode indicates the error value for the calculation. If an error has occurred this value is not equal zero. All possible error values are summarised in the enumeration [E\\_SPA\\_error\\_code](#) [► 20].

## Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT V3.1.4011	PC or CX (x86, x64, ARM)	Tc2_SPA

## 4.1.2 Data types

### 4.1.2.1 Structures

#### 4.1.2.1.1 ST\_SPA\_TIMESTRUCT

```
TYPE ST_SPA_TIMESTRUCT :
STRUCT
  iYear      :INT(-2000..6000); (* 4-digit year, valid range: -2000 TO 6000, error code: 1 *)
  iMonth     :INT(1..12); (* 2-digit month, valid range: 1 to 12 (Jan.= 1), error code: 2 *)
  iDay       :INT(1..31); (* 2-digit day, valid range: 1 to 31, error code: 3 *)
  iHour      :INT(0..24); (* Observer local hour, valid range: 0 to 24, error code: 4 *)
  iMinute    :INT(0..59); (* Observer local minute, valid range: 0 to 59, error code: 5 *)
  iSecond    :INT(0..59); (* Observer local second, valid range: 0 TO 59, error code: 6 *)
END_STRUCT
END_TYPE
```

The structure *ST\_SPA\_TIMESTRUCT* contains information on date and time. It is used at the input for function block [FB\\_SPA](#) [► 16] in order to specify the local time at the location. This local time has seconds as the smallest unit.

Various time scales are in use. The Universal Time (UT1) is used for sun position calculations based on the time specified in *ST\_SPA\_TIMESTRUCT*. If an inaccuracy of  $\pm 0.005^\circ$  is acceptable for the sun angles, the coordinated world time (UTC) may be used as the time. Explanatory notes can be found on the [overview page](#) [► 7].

**Note:** Summer and winter time clock change must not be used. The initiation of summer time (daylight saving time) in the 20. century should only extend the usable number of hours daylight per day. It's not common practice in all countries. For calculating the sun position with this library the Standard Time has to be used. In germany the Standard Time is equal to the winter time.

### 4.1.2.2 Enumerations

#### 4.1.2.2.1 E\_SPA\_FunctionCode

The enumeration *E\_SPA\_FunctionCode* defines constant values for the different functions, which can be executed with the function block FB\_SPA [► 16]. In addition to sun angles, sunrise and sunset can be calculated, depending on the selection. A detailed explanation of the terminology can be found on the [overview page \[► 7\]](#).

#### NOTE

##### Required time

The time required for a calculation strongly depends on the choice of function code.

```
(* enumeration for function codes to select desired final outputs from SPA *)

TYPE E_SPA_FunctionCode : (
    eSPA_ZA,           (*calculate zenith AND azimuth      [default setting] *)
    eSPA_ZA_INC,       (*calculate zenith, azimuth, AND incidence *)
    eSPA_ZA_RTS,       (*calculate zenith, azimuth, AND sun rise/transit/set values *)
    eSPA_ALL           (*calculate all SPA output values *)
);
END_TYPE
```

##### eSPA\_ZA :

If the function code *eSPA\_ZA* is selected only the sun angles (zenith, azimuth, azimuth180) are calculated [DEFAULT].

##### eSPA\_ZA\_INC :

In addition to the sun angles the angle of incidence in relation to the specified surface is issued.

##### eSPA\_ZA\_RTS :

In addition to the sun angles, sunrise, solar apex and sunset is calculated.

##### eSPA\_ALL :

All offered data are calculated and displayed at the output.

#### 4.1.2.2.2 E\_SPA\_ErrorCode

The enumeration *E\_SPA\_ErrorCode* defines constant values for the different errors that can be generated internally in the library.

These values can be found in the output variable *iErrorCode* again, which indicates the associated integer value at the output of the PLC SPA function block FB\_SPA [► 16] in the event of an error.

```
(* enumeration for error codes returned as iErrorCode output of FB_SPA
// Note: A non-zero return error code indicates that one of the //
//       input values did not pass simple bounds tests.         //
//       (input values must be within the range of the specified //
//       input values)                                           *)

TYPE E_SPA_ErrorCode : (
    eSPA_ERR_NoError           := 0,
    eSPA_ERR_InvalidYear,
    eSPA_ERR_InvalidMonth,
    eSPA_ERR_InvalidDay,
    eSPA_ERR_InvalidHour,
    eSPA_ERR_InvalidMinute,
    eSPA_ERR_InvalidSecond,
    eSPA_ERR_InvalidDeltaT,
    eSPA_ERR_InvalidTimezone,
    eSPA_ERR_InvalidLongitude,
    eSPA_ERR_InvalidLatitude,
    eSPA_ERR_InvalidElevation,
    eSPA_ERR_InvalidPressure,
    eSPA_ERR_InvalidTemperature,
```

```
eSPA_ERR_InvalidSlope,  
eSPA_ERR_InvalidAZMRotation,  
eSPA_ERR_InvalidAtmosRefract,  
eSPA_ERR_InvalidFunctionCode  
);  
END_TYPE
```

## 4.1.3 Global constants

### 4.1.3.1 Global\_Version

All libraries have a specific version. This version is shown in the PLC library repository too. A global constant contains the library version information:

```
VAR_GLOBAL CONSTANT  
    stLibVersion_Tc2_SPA : ST_LibVersion;  
END_VAR
```

To compare the existing version to a required version the function F\_CmpLibVersion (defined in Tc2\_System library) is offered.

#### NOTE

##### Use only this query

All other possibilities known from TwinCAT2 libraries to query a library version are obsolete!

## 5 Example

This example offers an introduction into the handling of function block `FB_SPA` [► 16], which is available with the TwinCAT Solar Position Algorithm library.

The objective in this example is to determine the sun position on 4 March 2010 at 14:27:00 at the Cheops pyramid in Egypt.

Time zone: UTC + 2 hours

Latitude: 29.979, [°]

Longitude: 31.134 [°]

Height: 70 [m]

Annual average temperature: 21.7 [°C]

Other locations and times are determined similarly.

### Overview

The following steps are now performed:

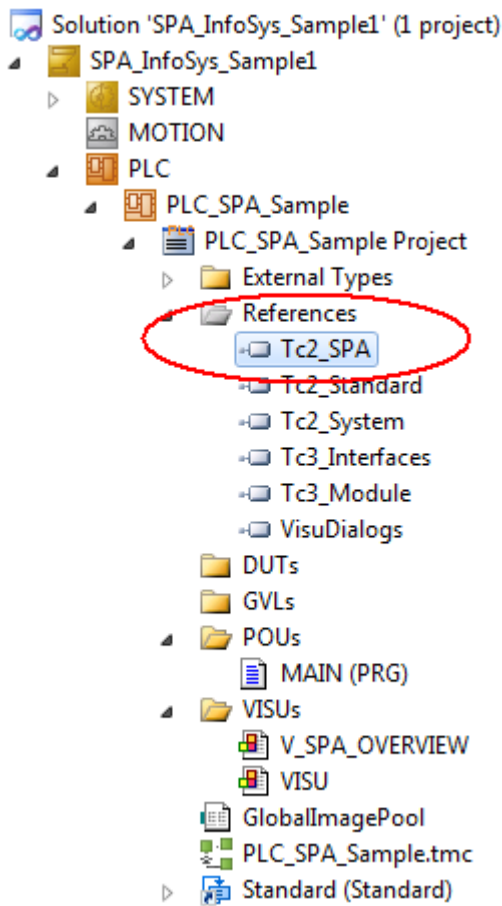
1. Installation of the PLC library
2. Program structure
3. Test

#### 1. Installation of the PLC library

Create a new TwinCAT PLC project and select your target platform.

Your first POU is a program called MAIN and in the programming language ST (Structured Text).

Mark node References and insert the library `Tc2_SPA`.



## 2. Program structure

For sun position calculations you should declare an instance of function block FB\_SPA [► 16] and local variables for allocating the required result values.

The input parameter for the calculation can be directly assigned to the inputs of the function block. In addition to the sun angles the sunrise and sunset is required as output, which means the advanced functionality is required, which is specified via the enumeration value eSPA\_ZA\_RTS of type E\_SPA\_FunctionCode [► 20].

The output values of the function block are assigned to your local variables.

The program section should now look as follows:

```
PROGRAM MAIN
VAR
    fbSPA      : FB_SPA;
    fSunZenith : LREAL;
    fSunAzimuth : LREAL;
    tSunrise   : TIME;
    tSunset    : TIME;
    eErrorCode : E_SPA_ErrorCode;
    bExecute   : BOOL;
    bInit      : BOOL := TRUE;
END_VAR

IF bInit THEN
    bInit      := FALSE;
    fbSPA.stTime.iYear      := 2010;
    fbSPA.stTime.iMonth     := 3;
    fbSPA.stTime.iDay       := 4;
    fbSPA.stTime.iHour      := 14;
    fbSPA.stTime.iMinute    := 27;
    fbSPA.fTimezone        := 2;
    fbSPA.fLongitude        := 31.134;
    fbSPA.fLatitude         := 29.979;
    fbSPA.fElevation        := 70;
    fbSPA.fTemperature      := 21.7;
    fbSPA.eFunction         := eSPA_ZA_RTS;
END_IF

IF bExecute THEN
    fbSPA();
    eErrorCode := fbSPA.iErrorCode;

    fSunZenith := fbSPA.fZenith;
    fSunAzimuth := fbSPA.fAzimuth;
    tSunrise   := LREAL_TO_TIME(fbSPA.fSunrise*60*60*1000);
    tSunset    := LREAL_TO_TIME(fbSPA.fSunset*60*60*1000);
END_IF
```

This sample of the TwinCAT Solar Position Algorithm library contains a visualisation facility that provides a quick overview of current inputs and outputs of function block FB\_SPA [► 16]. It is therefore ideal for test purposes.

## 3. Test

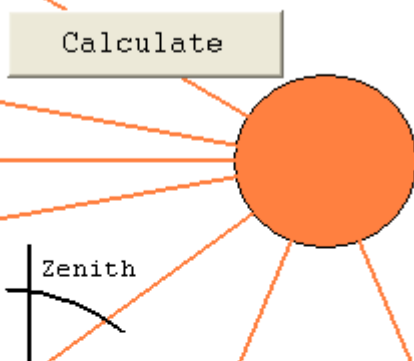
Compile the created PLC program.

Make sure that TwinCAT is in the Run mode on the desired system.

Login to the desired run-time system from TwinCAT PLC Control. Start the PLC program.

The calculation is executed by setting the local variable *bExecute* to TRUE. This can be done via 'online write' or the corresponding button in the visualisation, for example.

The visualisation should now present the following results:

TwinCAT Solar Position Algorithm			
Inputs			
Year:	2010		
Month:	3		
Day:	4		
Hour:	14		
Minute:	27		
Second:	0		
Delta_t:	66.000000		
Timezone:	2.000000		
Longitude:	31.134000		
Latitude:	29.979000		
		Outputs	
Elevation:	70.000000	Zenith:	49.428931
Pressure:	1000.000000	Azimuth:	228.541778
Temperature:	21.700000	Azimuth180:	48.541778
Slope:	0.000000	Incidence:	0.000000
Azm_rotation:	0.000000	Suntransit:	12.119948
Atmos_refract:	0.566700	Sunrise:	6.307221
Function:	eSPA_ZA_RTS	Sunset:	17.939914
		ErrorCode:	0

The sun angles at other locations and at other times within the given value ranges can be calculated accordingly. If an input parameter is invalid, an *eErrorCode* with the corresponding enumeration value for the error is displayed.

Click here to save this example program:

[https://infosys.beckhoff.com/content/1033/TF3900\\_TC3\\_Solar\\_Position\\_Algorithm/Resources/zip/946940683.zip](https://infosys.beckhoff.com/content/1033/TF3900_TC3_Solar_Position_Algorithm/Resources/zip/946940683.zip).





More Information:  
**[www.beckhoff.com/tf3900](http://www.beckhoff.com/tf3900)**

Beckhoff Automation GmbH & Co. KG  
Hülshorstweg 20  
33415 Verl  
Germany  
Phone: +49 5246 9630  
[info@beckhoff.com](mailto:info@beckhoff.com)  
[www.beckhoff.com](http://www.beckhoff.com)

