# Table of contents

1 Foreword .................................................................................................................................................. 5  
1.1 Notes on the documentation............................................................................................................. 5  
1.2 Safety instructions ............................................................................................................................. 6  
2 Overview .................................................................................................................................................. 7  
3 Installation ................................................................................................................................................ 8  
3.1 System requirements .......................................................................................................................... 8  
3.2 Installation ........................................................................................................................................... 8  
3.3 Installation Windows CE ..................................................................................................................... 11  
3.4 Licensing ............................................................................................................................................. 13  
4 TwinCAT SMTP ........................................................................................................................................ 16  
4.1 Configuration ....................................................................................................................................... 16  
4.2 PLC API ................................................................................................................................................. 17  
4.2.1 Function blocks ................................................................................................................................. 18  
4.3 Samples ............................................................................................................................................... 31  
4.3.1 Send mail sample ............................................................................................................................... 31  
4.3.2 SmtpFull sample with features ......................................................................................................... 31  
4.4 Appendix ............................................................................................................................................... 32  
4.4.1 Trouble-Shooting ............................................................................................................................... 32  
4.4.2 Error Codes ....................................................................................................................................... 33  
4.4.3 Windows Socket Error Codes ........................................................................................................... 34  
5 TwinCAT SMS .......................................................................................................................................... 39  
5.1 PLC API ............................................................................................................................................... 39  
5.1.1 Function blocks ................................................................................................................................. 39  
5.1.2 Functions .......................................................................................................................................... 41  
5.1.3 Global constants ............................................................................................................................... 41  
5.2 Samples ............................................................................................................................................... 42  
5.2.1 Sending an SMS ................................................................................................................................. 42  
5.3 Appendix ............................................................................................................................................... 42  
5.3.1 Fault Finding ....................................................................................................................................... 42  
6 Appendix .................................................................................................................................................... 44  
6.1 ADS Return Codes ............................................................................................................................... 44  
6.2 Support and Service ............................................................................................................................. 48
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®, TwinCAT/BSD®, TC/BSD®, 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:


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!

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="DANGER" /></td>
<td>Serious risk of injury! Failure to follow the safety instructions associated with this symbol directly endangers the life and health of persons.</td>
</tr>
<tr>
<td><img src="image" alt="WARNING" /></td>
<td>Risk of injury! Failure to follow the safety instructions associated with this symbol endangers the life and health of persons.</td>
</tr>
<tr>
<td><img src="image" alt="CAUTION" /></td>
<td>Personal injuries! Failure to follow the safety instructions associated with this symbol can lead to injuries to persons.</td>
</tr>
<tr>
<td><img src="image" alt="NOTE" /></td>
<td>Damage to the environment or devices Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.</td>
</tr>
</tbody>
</table>

Tip or pointer

This symbol indicates information that contributes to better understanding.
2 Overview

The function provides the possibility to send alerts and stati directly from TwinCAT:

**SMTP Server:**
The TwinCAT SMTP Server is used to send E-Mail messages with TwinCAT.

_TwinCAT SMTP Overview [16]

**SMS Library:**
The TwinCAT SMS Library is used to send SMS messages with TwinCAT via GSM modem.

_TwinCAT SMS Overview [39]
3 Installation

3.1 System requirements

<table>
<thead>
<tr>
<th>Technical data</th>
<th>TF6350 TwinCAT3 SMS/SMTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target system</td>
<td>WinXP, WES, Win7, WES7</td>
</tr>
<tr>
<td></td>
<td>IPC or CX, (x86)</td>
</tr>
<tr>
<td>Min. TwinCAT version</td>
<td>3.1.4000</td>
</tr>
<tr>
<td>Min. TwinCAT level</td>
<td>TC1200 TC3</td>
</tr>
</tbody>
</table>

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.1.4000</td>
<td>PC or CX (x86)</td>
<td>Tc2_Smtp, Tc2_Sms</td>
</tr>
</tbody>
</table>

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.

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

![Dialog Box](image)

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

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

![Dialog Box](image)
7. Select **Finish** to exit the setup.

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

### 3.3 Installation Windows CE

This part of the documentation describes, how you can install the TwinCAT 3 Function TF6350 SMS SMTP on a Beckhoff Embedded PC Controller based on Windows CE.

The setup process consists of four steps:

- Downloading the setup file
- Installation on a host computer
- Transfering the executable to the Windows CE device
- Software installation

#### Downloading the setup file

The CAB installation files for Windows CE are part of the TF6350 SMS SMTP setup. Therefore you only need to download one setup file from [www.beckhoff.com](http://www.beckhoff.com) which contains binaries for Windows XP, Windows 7 and Windows CE (x86 and ARM).

#### Installation on a host computer

After installation, the install folder (e.g. C:\TwinCAT\Functions\TF6350-SMS-SMTP) contains three directories - each one for a different hardware platform:

- **CE-ARM**: ARM-based Embedded Controllers running Windows CE, e.g. CX8090, CX9020
- **CE-X86**: X86-based Embedded Controllers running Windows CE, e.g. CX50xx, CX20x0
- **Win32**: Embedded Controllers running Windows XP, Windows 7 or Windows Embedded Standard
The CE-ARM and CE-X86 folders contain the TF6350 CAB-File for Windows CE - corresponding to the hardware platform of your Windows CE device. This file needs to be transferred to the Windows CE device, see next chapter.

**Transferring the executable to the Windows CE device**

Transfer the corresponding CAB-File to your Windows CE device. This can be done via one of the following ways:

- via a Shared Folder
- via the integrated FTP-Server
- via a USB Stick, CF card or SD-Card

**Software installation**

After the file has been transferred via one of the above methods, you need to execute the file and acknowledge the following dialog with "Ok". Restart your Windows CE device after the installation has finished.

After the restart has been completed, the TF6350 SMTP server and configuration will be automatically started in background and is now available to use.

The software will be installed in the following directory on the CE device: \Hard Disk\TwinCAT\Functions\TF6350-SMS-SMTP

**Software upgrade**

If you already have a version of TF6350 installed on your Windows CE device, you need to perform the following steps on the Windows CE device to upgrade to a newer version:

- Open the CE Explorer by clicking on Start --> Run and entering "explorer"
- Navigate to \Hard Disk\TwinCAT\Functions\TF6350-SMS-SMTP\Server
- Rename TcMailSrv.exe
- Restart the Windows CE device
- Transfer the new CAB-File to the CE device
- Execute the CAB-File and install the new version of TF6350
- Delete the old (renamed) files
- Restart the Windows CE device

After the restart is complete, the new version is active.

After a successful installation the TC3 Function needs to be licensed.
3.4 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**

- 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. "TF4100 TC3 Controller Toolbox").

![Manage Licenses](image)

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)](image)

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

   ![Enter Security Code](image)

   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 TwinCAT SMTP

The TwinCAT SMTP Server enables to send eMails directly out of the PLC. The PLC SMTP library and the TwinCAT SMTP Server will be provided by the setup.

Mails can be send from a local system or remote system which is connected via network (TCP/IP). Therefore several machines can be observed and provide information (e.g. status, alerts, attached measured values) via mail worldwide.

The TwinCAT SMTP Server can communicate with a local mailserver (e.g. Exchange, SendMail, Notes) or external mail providers (e.g. GoogleMail, Hotmail, GMX). SMTP and encrypted communication via SSL/STARTTLS is supported.

4.1 Configuration

TC SMTP Server uses a XML-based configuration, which is located in the installation folder of the supplement (TwinCAT\Functions\TF6350-SMS-SMTP).

The default configuration is:

```xml
<TcSmtpConfig>
    <!-- EnableLogFile: 0 (Disabled), 1 (Enabled), 2 (Verbose) -->
    <EnableLogFile>0</EnableLogFile>
    <!-- LogSize: in Byte, 0 = use Default -->
    <LogSize>20000</LogSize>
    <!-- Authentication: 0 (NONE), 1 (AUTO), 2 (LOGIN), 3 (NTLM), 4 (PLAIN) -->
    <Authentication>1</Authentication>
    <!-- Port: 0 (use default ports) -->
    <Port>0</Port>
    <!-- ContentEncoding: 0 (7BIT), 1 (8BIT), 2 (BINARY), 3 (BASE64), 4 (QUOTED_PRINTABLE) -->
    <ContentEncoding>0</ContentEncoding>
    <!-- Timeout for the socket connection -->
    <Timeout>15000</Timeout>
    <!-- Charset for the message content -->
    <Charset>iso-8859-1</Charset>
    <Reconnects>5</Reconnects>
</TcSmtpConfig>
```
Notes regarding the XML configuration file

**EnableLogFile**: Enable logging only for diagnose issues.

**Authentication**: Method for authenticate. Use option 1.

**Port**: Option 0 uses the default ports.

**ContentEncoding**: Defines the content encoding.

**Timeout**: Timeout for mail delivery in ms.

**Charset**: Defines the character set.

**Reconnects**: Number of retries.

### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Smtp</td>
</tr>
</tbody>
</table>

### 4.2 PLC API

The TwinCAT library provides function blocks for sending e-mails to the TwinCAT SMTP server.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FB_SmtpV3 [18]</td>
<td>Function block to send emails to the TwinCAT SMTP Server</td>
</tr>
<tr>
<td>FB_SmtpV3_Full [20]</td>
<td>Function block with additional properties (e.g. send attachments, set priorities, send HTML messages)</td>
</tr>
</tbody>
</table>

### Obsolete function blocks

The function blocks, which are described in the chapter obsolete are obsolete. Their usage is deprecated. Please use the function blocks FB_SmtpV3 and FB_SmtpV3_Full which should provide the same functionality.

### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Smtp</td>
</tr>
</tbody>
</table>
4.2.1 Function blocks

4.2.1.1 FB_SmtpV3

The block sends a byte stream to a remote ADS device via ADS. The TwinCAT ADS Smtp service must be running on the remote ADS device, so that the byte stream can be received and processed into an e-mail. Once the byte stream has been processed the e-mail is sent.

VAR_INPUT

VAR_INPUT
sNetID : T_AmsNetID; (* AmsNetID *)
sSmtpServer : T_MaxString; (* Smtp Server address ( IP or Name ) *)
sUsername : T_MaxString; (* Smtp Username *)
sPassword : T_MaxString; (* Smtp Password *)
nEncryption : UDINT; (* 0=NONE, 1=STARTTLS, 2=SSL *)
sFrom : T_MaxString; (* Sender string *)
sTo : T_MaxString; (* To recipient string *)
sCc : T_MaxString; (* Cc recipient string *)
sBcc : T_MaxString; (* Bcc recipient string *)
sSubject : T_MaxString; (* Subject string *)
pMessage : DWORD; (* Pointer to the message *)
cbMessage : UDINT; (* Message length in byte to send *)
bExecute : BOOL;
tTimeout : TIME := T#20s;
END_VAR

sNetID: AmsNetID on which the TwinCAT Smtp server runs.

sSmtpServer: Name or IP of the Smtp server.

sUsername: Username for the Smtp Server.

sPassword: Password for the Smtp Server.

nEncryption: Smtp encryption type:
0 = NONE
1 = STARTTLS
2 = SSL

sFrom: A string containing the e-mail address of the sender. A sender must be specified. The string is limited to 255 characters.

sTo: A string containing the e-mail address of the recipient. Several addresses can be specified, separated by semicolon. At least one recipient has to be specified. The string is limited to 255 characters.
sCc: A string containing an e-mail address of a further recipient (cc=carbon copy). This string can also be empty. A copy of the e-mail is sent to this recipient. The e-mail address of this recipient is visible to other recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.

sBcc: A string containing the e-mail address of a further recipient (Bcc = blind carbon copy). This string can also be empty. A copy of the e-mail is sent to this/these recipient/s. The e-mail address of this recipient is not visible to other recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.

sSubject: A string containing the subject line for the e-mail. The e-mail may be sent without subject, in which case the name of the sending computer is automatically entered in the subject line (e.g. "Mail sent from: CX_00762C"). The string for the subject line is limited to 255 characters.

pMessage: The address (a pointer) to a null-terminated string containing the e-mail text. The e-mail may be sent without body text, in which case the date and time are entered automatically (e.g. "Mail sent at: Thu, 23 Mar 2006 02:31:44 -0800"). The address of the string can be determined with the ADR operator.

cbMessage: Length of the e-mail text. The length can be determined through the LEN operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the command.

VAR_OUTPUT

| VAR_OUTPUT | bBusy : BOOL; |
| VAR_OUTPUT | bError : BOOL; |
| VAR_OUTPUT | nErrId : UDINT; |
| END_VAR |

bBusy: This output remains TRUE until the block has executed a command, but at the longest for the duration supplied to the tTimeOut input.

bError: This output is switched to TRUE if an error occurs during the execution of a command. The command-specific error code is contained in iErrId.

nErrId: Contains the command-specific error code of the most recently executed command (see table [33]).

Notes: Make sure, that you don't use \o within byte-arrays. Otherwise the Message will be cut. The maximum amount of characters, that can be used in a message, is 510.725 - you have 1275 characters for From, To, Cc, Bcc and Subject.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Smtp</td>
</tr>
</tbody>
</table>
This function block communicates over ADS with the TwinCAT SMTP Server. It offers a wide range of mail functionalities as for example the prioritization of emails out of the PLC. The individual parameters will be described in detail in this documentation.

**VAR_INPUT**

- **sNetId**: AmsNetID on which the TwinCAT SMTP server runs.
- **sSmtpServer**: Name or IP of the SMTP server.
- **sUsername**: Username for the SMTP server.
- **sPassword**: Password for the SMTP server.
- **sFrom**: Sender string.
- **sTo**: To recipient string.
- **sCc**: Cc recipient string.
- **sBcc**: Bcc recipient string.
- **sDispositionNotification**: Disposition notification recipient string.
- **sReturnReceipt**: Return recipient string.
- **nPriority**: Priority value.
- **nSensitivity**: Sensitivity value.
- **nPort**: Communication port.
- **nContentType**: Content type.
- **sSubject**: Subject string.
- **pMessage**: Pointer to the message.
- **cbMessage**: Message length in bytes to send.
- **sAttachments**: Different attachments.
- **bExecute**: Trigger flag.
- **tTimeout**: Communication timeout.

**FB_SmtpV3_Full**

- **sNetId**: T_AmsNetID
- **sSmtpServer**: T_MaxString
- **sUsername**: T_MaxString
- **sPassword**: T_MaxString
- **nEncryption**: UDINT
- **sFrom**: T_MaxString
- **sTo**: T_MaxString
- **sCc**: T_MaxString
- **sBcc**: T_MaxString
- **sDispositionNotification**: T_MaxString
- **sReturnReceipt**: T_MaxString
- **nPriority**: UDINT
- **nSensitivity**: UDINT
- **nPort**: UDINT
- **nContentType**: UDINT
- **sSubject**: T_MaxString
- **pMessage**: DWORD
- **cbMessage**: UDINT
- **sAttachments**: ARRAY [0..32] OF STRING
- **bExecute**: BOOL
- **tTimeout**: TIME := T#20s
nEncryption: Smtp encryption type:
0 = NONE
1 = STARTTLS
2 = SSL

sFrom: A string containing the email address of the sender. A sender must be specified. The string is limited to 255 characters.

sTo: A string containing the email address of the recipient. Several addresses can be specified, separated by semicolon. At least one recipient has to be specified. The string is limited to 255 characters.

sCc: A string containing an email address of a further recipient (cc=carbon copy). This string can also be empty. A copy of the email is sent to this recipient. The email address of this recipient is visible to other recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.

sBcc: A string containing the email address of a further recipient (Bcc = blind carbon copy). This string can also be empty. A copy of the email is sent to this recipient. The email address of this recipient is not visible to other recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.

sDispositionNotification: The mail address which is given to this parameter receives an return receipt of the recipients under sTo and sCc. The condition precedent is that the return receipt will be send by the recipients.

sReturnReceipt: An acknowledgment of transfer will be send to this mail address.

nPriority: With this parameter you can set the priority of the mail:
1 = Highest
2 = not used
3 = Normal
4 = not used
5 = Lowest

nSensitivity: With this parameter you can set the confidentiality of the message:
0 = Private
1 = Personal
2 = Normal
3 = Confidential

nPort: You can choose the communication-port here. If you do not enter an own port it will be accessed to the default-port 25.

nContentType: With this parameter it is possible to make a HTML-code which is given per pointer (pMessage) and size (cbMessage) to a string variable readable in the mail.

sSubject: A string containing the subject line for the e-mail. The email may be sent without subject, in which case the name of the sending computer is automatically entered in the subject line (e.g. "Mail sent from: CX_00762C"). The string for the subject line is limited to 255 characters.

pMessage: The address (a pointer) to a null-terminated string containing the email text. The email may be sent without body text, in which case the date and time are entered automatically (e.g. "Mail sent at: Thu, 23 Mar 2006 02:31:44 -0800"). The address of the string can be determined with the ADR operator.

cbMessage: Length of the email text. The length can be determined through the LEN operator.

bExecute: The function block is activated by a rising edge at this input.

sAttachments: Array of filenames

tTimeout: Maximum time allowed for the execution of the command.
VAR_OUTPUT

VAR_OUTPUT
bBusy : BOOL;
bError : BOOL;
nErrId : UDINT;
END_VAR

bBusy: This output remains TRUE until the block has executed a command, but at the longest for the duration supplied to the tTimeOut input.

bError: This output is switched to TRUE if an error occurs during the execution of a command. The command-specific error code is contained in iErrorId.

nErrId: Contains the command-specific error code of the most recently executed command (see table [33]).

Notes: Make sure, that you don’t use \o within byte-arrays. Otherwise the Message will be cut. The maximum amount of characters, that can be used in a message, is 510.725 - you have 1275 characters for From, To, Cc, Bcc and Subject.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Ssmtp</td>
</tr>
</tbody>
</table>

4.2.1.3 [obsolete functions]

4.2.1.3.1 FB_Ssmtp

The block sends a byte stream to a remote ADS device via ADS. The TwinCAT ADS Smtp service must be running on the remote ADS device, so that the byte stream can be received and processed into an e-mail. Once the byte stream has been processed the e-mail is sent.

Note that password checking must be disabled on the SMTP server, since the TwinCAT ADS Smtp service does not register on the server via password checking.
**VAR_INPUT**

VAR_INPUT

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sNetId</td>
<td>AmsNetID on which the TwinCAT SMS server runs.</td>
</tr>
<tr>
<td>sSmtpServer</td>
<td>Name or IP of the Smtp server.</td>
</tr>
<tr>
<td>sFrom</td>
<td>A string containing the e-mail address of the sender. A sender must be specified. The string is limited to 255 characters.</td>
</tr>
<tr>
<td>sTo</td>
<td>A string containing the e-mail address of the recipient. Several addresses can be specified, separated by semicolon. At least one recipient has to be specified. The string is limited to 255 characters.</td>
</tr>
<tr>
<td>sCc</td>
<td>A string containing an e-mail address of a further recipient (cc=carbon copy). This string can also be empty. A copy of the e-mail is sent to this recipient. The e-mail address of this recipient is visible to other recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.</td>
</tr>
<tr>
<td>sBcc</td>
<td>A string containing the e-mail address of a further recipient (Bcc = blind carbon copy). This string can also be empty. A copy of the e-mail is sent to this/these recipient/s. The e-mail address of this recipient is not visible to other recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.</td>
</tr>
<tr>
<td>sSubject</td>
<td>A string containing the subject line for the e-mail. The e-mail may be sent without subject, in which case the name of the sending computer is automatically entered in the subject line (e.g. &quot;Mail sent from: CX_00762C&quot;). The string for the subject line is limited to 255 characters.</td>
</tr>
<tr>
<td>pMessage</td>
<td>The address (a pointer) to a null-terminated string containing the e-mail text. The e-mail may be sent without body text, in which case the date and time are entered automatically (e.g. &quot;Mail sent at: Thu, 23 Mar 2006 02:31:44 -0800&quot;). The address of the string can be determined with the ADR operator.</td>
</tr>
<tr>
<td>cbMessage</td>
<td>Length of the e-mail text. The length can be determined through the LEN operator.</td>
</tr>
<tr>
<td>bExecute</td>
<td>The function block is activated by a rising edge at this input.</td>
</tr>
<tr>
<td>tTimeout</td>
<td>Maximum time allowed for the execution of the command.</td>
</tr>
</tbody>
</table>

**VAR_OUTPUT**

VAR_OUTPUT

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bBusy</td>
<td>This output remains TRUE until the block has executed a command, but at the longest for the duration supplied to the tTimeOut input.</td>
</tr>
<tr>
<td>bError</td>
<td>This output is switched to TRUE if an error occurs during the execution of a command. The command-specific error code is contained in nErrId.</td>
</tr>
<tr>
<td>nErrId</td>
<td>Contains the command-specific error code of the most recently executed command (see table [33]).</td>
</tr>
</tbody>
</table>

sNetId: AmsNetId on which the TwinCAT SMS server runs.

sSmtpServer: Name or IP of the Smtp server.

sFrom: A string containing the e-mail address of the sender. A sender must be specified. The string is limited to 255 characters.

sTo: A string containing the e-mail address of the recipient. Several addresses can be specified, separated by semicolon. At least one recipient has to be specified. The string is limited to 255 characters.

sCc: A string containing an e-mail address of a further recipient (cc=carbon copy). This string can also be empty. A copy of the e-mail is sent to this recipient. The e-mail address of this recipient is visible to other recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.

sBcc: A string containing the e-mail address of a further recipient (Bcc = blind carbon copy). This string can also be empty. A copy of the e-mail is sent to this/these recipient/s. The e-mail address of this recipient is not visible to other recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.

sSubject: A string containing the subject line for the e-mail. The e-mail may be sent without subject, in which case the name of the sending computer is automatically entered in the subject line (e.g. "Mail sent from: CX_00762C"). The string for the subject line is limited to 255 characters.

pMessage: The address (a pointer) to a null-terminated string containing the e-mail text. The e-mail may be sent without body text, in which case the date and time are entered automatically (e.g. "Mail sent at: Thu, 23 Mar 2006 02:31:44 -0800"). The address of the string can be determined with the ADR operator.

cbMessage: Length of the e-mail text. The length can be determined through the LEN operator.

bExecute: The function block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the command.
Notes:

Make sure, that you don't use \o within byte-arrays. Otherwise the Message will be cut.
The maximum amount of characters, that can be used in a message, is 510.725 - you have 1275 characters for From, To, Cc, Bcc and Subject.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Smtp</td>
</tr>
</tbody>
</table>

4.2.1.3.2 FB_SmtpFull

This function block communicates over ADS with the TwinCAT SMTP Server. It offers a wide range of mail functionalities as for example the prioritization of emails out of the PLC. The individual parameters will be described in detail in this documentation.

VAR_INPUT

<table>
<thead>
<tr>
<th>VAR_INPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>sNetId             : T_AmsNetID; (* AmsNetID *)</td>
</tr>
<tr>
<td>sSmtpServer        : T_MaxString; (* Smtp Server address (IP or Name) *)</td>
</tr>
<tr>
<td>sUsername          : T_MaxString; (* Smtp Username *)</td>
</tr>
<tr>
<td>sPassword          : T_MaxString; (* Smtp Password *)</td>
</tr>
<tr>
<td>nAuth              : UDINT; (* Smtp Auth Type*)</td>
</tr>
<tr>
<td>sFrom              : T_MaxString; (* Sender string *)</td>
</tr>
<tr>
<td>sTo                : T_MaxString; (* To recipient string *)</td>
</tr>
<tr>
<td>sCc                : T_MaxString; (* Cc recipient string *)</td>
</tr>
<tr>
<td>sBcc               : T_MaxString; (* Bcc recipient string *)</td>
</tr>
<tr>
<td>sDispositionNotification : T_MaxString; (* Disposition notification recipient string *)</td>
</tr>
<tr>
<td>sReturnReceipt     : T_MaxString; (* Return recipient string *)</td>
</tr>
<tr>
<td>nPriority          : UDINT; (* Priority value *)</td>
</tr>
<tr>
<td>nSensitivity       : UDINT; (* Sensitivity value *)</td>
</tr>
<tr>
<td>nPort              : UDINT; (* Communication port *)</td>
</tr>
<tr>
<td>nContentType       : UDINT; (* Content type *)</td>
</tr>
<tr>
<td>sSubject           : T_MaxString; (* Subject string *)</td>
</tr>
</tbody>
</table>
sNetId: AmsNetID on which the TwinCAT SMTP server runs.

sSmtpServer: Name or IP of the SMTP server.

sUsername: Username for the SMTP server.

sPassword: Password for the SMTP server.

nAuth: Smtp Auth Type:
0 = AUTH NONE
1 = RESERVED
2 = AUTH LOGIN
3 = AUTH NTLM
4 = AUTH PLAIN

sFrom: A string containing the email address of the sender. A sender must be specified. The string is limited to 255 characters.

sTo: A string containing the email address of the recipient. Several addresses can be specified, separated by semicolon. At least one recipient has to be specified. The string is limited to 255 characters.

sCc: A string containing an email address of a further recipient (cc=carbon copy). This string can also be empty. A copy of the email is sent to this recipient. The email address of this recipient is visible to other recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.

sBcc: A string containing the email address of a further recipient (Bcc = blind carbon copy). This string can also be empty. A copy of the email is sent to this/these recipient/s. The email address of this recipient is not visible to other recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.

sDispositionNotification: The mail address which is given to this parameter receives an return receipt of the recipients under sTo and sCc. The condition precedent is that the return receipt will be send by the recipients.

sReturnReceipt: An acknowledgment of transfer will be send to this mail address.

nPriority: With this parameter you can set the priority of the mail:
1 = Highest
2 = not used
3 = Normal
4 = not used
5 = Lowest

nSensitivity: With this parameter you can set the confidentiality of the message:
0 = Private
1 = Personal
2 = Normal
3 = Confidential

nPort: You can choose the communication-port here. If you do not enter an own port it will be accessed to the default-port 25.

nContentType: With this parameter it is possible to make a HTML-code which is given per pointer (pMessage) and size (cbMessage) to a string variable readable in the mail.

sSubject: A string containing the subject line for the e-mail. The email may be sent without subject, in which
case the name of the sending computer is automatically entered in the subject line (e.g. "Mail sent from: CX_00762C"). The string for the subject line is limited to 255 characters.

**pMessage:** The address (a pointer) to a null-terminated string containing the email text. The email may be sent without body text, in which case the date and time are entered automatically (e.g. "Mail sent at: Thu, 23 Mar 2006 02:31:44 -0800"). The address of the string can be determined with the ADR operator.

**cbMessage:** Length of the email text. The length can be determined through the LEN operator.

**bExecute:** The function block is activated by a rising edge at this input.

**sAttachments:** Array of filenames

**tTimeout:** Maximum time allowed for the execution of the command.

### VAR_OUTPUT

<table>
<thead>
<tr>
<th>VAR_OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>bBusy : BOOL;</td>
</tr>
<tr>
<td>bError : BOOL;</td>
</tr>
<tr>
<td>nErrId : UDINT;</td>
</tr>
</tbody>
</table>

**bBusy:** This output remains TRUE until the block has executed a command, but at the longest for the duration supplied to the tTimeOut input.

**bError:** This output is switched to TRUE if an error occurs during the execution of a command. The command-specific error code is contained in iErrId.

**nErrId:** Contains the command-specific error code of the most recently executed command (see table [33]).

### Notes:

Make sure, that you don't use \o within byte-arrays. Otherwise the Message will be cut. The maximum amount of characters, that can be used in a message, is 510.725 - you have 1275 characters for From, To, Cc, Bcc and Subject.

### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Smtp</td>
</tr>
</tbody>
</table>


The block sends a byte stream to a remote ADS device via ADS. The TwinCAT ADS Smtp service must be running on the remote ADS device, so that the byte stream can be received and processed into an e-mail. Once the byte stream has been processed the e-mail is sent.

**VAR_INPUT**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sNetId</td>
<td>T_AmsNetID</td>
<td>AmsNetID on which the TwinCAT SMS server runs.</td>
</tr>
<tr>
<td>sSmtpServer</td>
<td>T_MaxString</td>
<td>Name or IP of the Smtp server.</td>
</tr>
<tr>
<td>sUsername</td>
<td>T_MaxString</td>
<td>Username for the Smtp Server.</td>
</tr>
<tr>
<td>sPassword</td>
<td>T_MaxString</td>
<td>Password for the Smtp Server.</td>
</tr>
<tr>
<td>nAuth</td>
<td>UDINT</td>
<td>Smtp Auth Type:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = AUTH NONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = RESERVEDorative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 = AUTH LOGIN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 = AUTH NTLM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 = AUTH PLAIN</td>
</tr>
<tr>
<td>sFrom</td>
<td>T_MaxString</td>
<td>A string containing the e-mail address of the sender. A sender must be specified. The string is limited to 255 characters.</td>
</tr>
<tr>
<td>sTo</td>
<td>T_MaxString</td>
<td>A string containing the e-mail address of the recipient. Several addresses can be specified, separated by semicolon. At least one recipient has to be specified. The string is limited to 255 characters.</td>
</tr>
<tr>
<td>sCc</td>
<td>T_MaxString</td>
<td>A string containing the e-mail address of the copy recipient.</td>
</tr>
<tr>
<td>sBcc</td>
<td>T_MaxString</td>
<td>A string containing the e-mail address of the blind copy recipient.</td>
</tr>
<tr>
<td>sSubject</td>
<td>T_MaxString</td>
<td>A string containing the subject of the e-mail.</td>
</tr>
<tr>
<td>pMessage</td>
<td>DWORD</td>
<td>Pointer to the message.</td>
</tr>
<tr>
<td>cbMessage</td>
<td>UDINT</td>
<td>Message length in byte to send.</td>
</tr>
<tr>
<td>sAttachments</td>
<td>ARRAY [0..32] OF STRING</td>
<td>Array of file attachments to be sent.</td>
</tr>
<tr>
<td>bExecute</td>
<td>BOOL</td>
<td>Execute flag.</td>
</tr>
<tr>
<td>tTimeout</td>
<td>TIME</td>
<td>Timeout for the operation.</td>
</tr>
</tbody>
</table>

sNetId: AmsNetID on which the TwinCAT SMS server runs.

sSmtpServer: Name or IP of the Smtp server.

sUsername: Username for the Smtp Server.

sPassword: Password for the Smtp Server.

nAuth: Smtp Auth Type:

0 = AUTH NONE
1 = RESERVED
2 = AUTH LOGIN
3 = AUTH NTLM
4 = AUTH PLAIN

sFrom: A string containing the e-mail address of the sender. A sender must be specified. The string is limited to 255 characters.

sTo: A string containing the e-mail address of the recipient. Several addresses can be specified, separated by semicolon. At least one recipient has to be specified. The string is limited to 255 characters.
sCc: A string containing an e-mail address of a further recipient (cc=carbon copy). This string can also be empty. A copy of the e-mail is sent to this recipient. The e-mail address of this recipient is visible to other recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.

sBcc: A string containing the e-mail address of a further recipient (Bcc = blind carbon copy). This string can also be empty. A copy of the e-mail is sent to this/these recipient(s). The e-mail address of this recipient is not visible to other recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.

sSubject: A string containing the subject line for the e-mail. The e-mail may be sent without subject, in which case the name of the sending computer is automatically entered in the subject line (e.g. "Mail sent from: CX_00762C"). The string for the subject line is limited to 255 characters.

pMessage: The address (a pointer) to a null-terminated string containing the e-mail text. The e-mail may be sent without body text, in which case the date and time are entered automatically (e.g. "Mail sent at: Thu, 23 Mar 2006 02:31:44 -0800"). The address of the string can be determined with the ADR operator.

cbMessage: Length of the e-mail text. The length can be determined through the LEN operator.

bExecute: The function block is activated by a rising edge at this input.

sAttachments: Array containing filenames

tTimeout: Maximum time allowed for the execution of the command.

VAR_OUTPUT

VAR_OUTPUT
  bBusy : BOOL;
  bError : BOOL;
  nErrId : UDINT;
END_VAR

bBusy: This output remains TRUE until the block has executed a command, but at the longest for the duration supplied to the tTimeOut input.

bError: This output is switched to TRUE if an error occurs during the execution of a command. The command-specific error code is contained in iErrId.

nErrId: Contains the command-specific error code of the most recently executed command (see table [33]).

Notes:

Make sure, that you don't use \o within byte-arrays. Otherwise the Message will be cut. The maximum amount of characters, that can be used in a message, is 510.725 - you have 1275 characters for From, To, Cc, Bcc and Subject.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Smtp</td>
</tr>
</tbody>
</table>

28  Version: 1.2  TF6350
4.2.1.3.4 FB_SmtpV2

The block sends a byte stream to a remote ADS device via ADS. The TwinCAT ADS Smtp service must be running on the remote ADS device, so that the byte stream can be received and processed into an e-mail. Once the byte stream has been processed the e-mail is sent.

VAR_INPUT

VAR_INPUT
sNetId : T_AmsNetID; (* AmsNetID *)
sSmtpServer : T_MaxString; (* Smtp Server address (IP or Name) *)
sUsername : T_MaxString; (* Smtp Username *)
sPassword : T_MaxString; (* Smtp Password *)
nAuth : UDINT; (* Smtp Auth Type *)
sFrom : T_MaxString; (* Sender string *)
sTo : T_MaxString; (* To recipient string *)
sCc : T_MaxString; (* Cc recipient string *)
sBcc : T_MaxString; (* Bcc recipient string *)
sSubject : T_MaxString; (* Subject string *)
pMessage : DWORD; (* Pointer to the message *)
cbMessage : UDINT; (* Message length in byte to send *)
bExecute : BOOL;
tTimeout : TIME := T#20s;
END_VAR

sNetId: AmsNetID on which the TwinCAT Smtp server runs.

sSmtpServer: Name or IP of the Smtp server.

sUsername: Username for the Smtp Server.

sPassword: Password for the Smtp Server.

nAuth: Smtp Auth Type:
0 = AUTH NONE
1 = RESERVED
2 = AUTH LOGIN
3 = AUTH NTLM
4 = AUTH PLAIN

sFrom: A string containing the e-mail address of the sender. A sender must be specified. The string is limited to 255 characters.

sTo: A string containing the e-mail address of the recipient. Several addresses can be specified, separated by semicolon. At least one recipient has to be specified. The string is limited to 255 characters.

sCc: A string containing an e-mail address of a further recipient (cc=carbon copy). This string can also be empty. A copy of the e-mail is sent to this recipient. The e-mail address of this recipient is visible to other
recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.

**sBcc:** A string containing the e-mail address of a further recipient (Bcc = blind carbon copy). This string can also be empty. A copy of the e-mail is sent to this/these recipient/s. The e-mail address of this recipient is not visible to other recipients. It is possible to enter multiple recipient addresses separated by semicolons. The string is limited to 255 characters.

**sSubject:** A string containing the subject line for the e-mail. The e-mail may be sent without subject, in which case the name of the sending computer is automatically entered in the subject line (e.g. "Mail sent from: CX_00762C"). The string for the subject line is limited to 255 characters.

**pMessage:** The address (a pointer) to a null-terminated string containing the e-mail text. The e-mail may be sent without body text, in which case the date and time are entered automatically (e.g. "Mail sent at: Thu, 23 Mar 2006 02:31:44 -0800"). The address of the string can be determined with the ADR operator.

**cbMessage:** Length of the e-mail text. The length can be determined through the LEN operator.

**bExecute:** The function block is activated by a rising edge at this input.

**tTimeout:** Maximum time allowed for the execution of the command.

### VAR_OUTPUT

<table>
<thead>
<tr>
<th>VAR_OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>bBusy : BOOL;</td>
</tr>
<tr>
<td>bError : BOOL;</td>
</tr>
<tr>
<td>nErrId : UDINT;</td>
</tr>
</tbody>
</table>

**bBusy:** This output remains TRUE until the block has executed a command, but at the longest for the duration supplied to the tTimeOut input.

**bError:** This output is switched to TRUE if an error occurs during the execution of a command. The command-specific error code is contained in iErrId.

**nErrId:** Contains the command-specific error code of the most recently executed command (see table [33]).

### Notes:

Make sure, that you don't use \o within byte-arrays. Otherwise the Message will be cut. The maximum amount of characters, that can be used in a message, is 510.725 - you have 1275 characters for From, To, Cc, Bcc and Subject.

### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Smtp</td>
</tr>
</tbody>
</table>

Version: 1.2
4.3 Samples

4.3.1 Send mail sample

The function block FB_SmtpV3 enables to send mails from your PLC.

Description

A message will be sent if the variable bStart is triggered.

ToDo: Configure mailserver and credentials.

Download Sample1 TC3 project

Program-variables

```plaintext
PROGRAM MAIN
VAR
fbSendMail: FB_SmtpV3;
sMessage: STRING := 'Hello Beckhoff';
R_Edge: R_TRIG;
bStart: BOOL;
bBusy: BOOL;
bError: BOOL;
nErrID: UDINT;
bSend: BOOL;
nErr: UDINT;
nMailCounter: UDINT;
END_VAR
```

Program-code

```plaintext
fbSendMail(
 sNetId:= '',
sSmtpServer:= 'mail.company.com',
sUsername:= 'TestUser',
sPassword:= 'TestPwd',
sFrom:= 'TestUser@company.com',
sTo:= 'service@company.com',
sSubject:= 'Email from your Beckhoff PLC',
pMessage:= ADR(sMessage),
chMessage:= SIZEOF(sMessage),
bExecute:= bStart,
bError=> bError,
bBusy=> bBusy,
nErrId=> nErrId);
IF NOT bBusy AND NOT bError AND bStart THEN
 bStart := FALSE;
END_IF
```

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Smtp</td>
</tr>
</tbody>
</table>

4.3.2 SmtpFull sample with features

With the function block FB_SmtpV3_Full more email functionalities are available.

Description

In this example the following possibilities will be described:

- The email text can provide HTML code, which offers new formatting possibilities.
• Furthermore it is possible to set the **priority** of an email
• Several files can be sent as **attachment**

A message will be sent if the variable `bStart` is triggered.

**ToDo:** Configure mailserver address, credentials and adjust file paths of the attachments.

Download Sample2 TC3 project

**Program-variables**

```plaintext
VAR
 fbSmtpFull : FB_SmtpV3_Full;
 sMessage_HTML : STRING(500) := <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
 "http://www.w3.org/TR/html4/loose.dtd"><html><head><title>HTML-Test</title><body><h3>Dear SMTP-
User,<\\n32 this e-mail was sent in HTML from <u>your PLC!</u><br/><br/>Best regards,<br/>your
Beckhoff-Team</body></html>";
 nPriority : UDINT;
 R_Edge : R_TRIG;
 bStart : BOOL;
 bError : BOOL;
 udErrId : UDINT;
 nMailCounter : UDINT := 0;
 sFiles : ARRAY [0..32] OF STRING(80) := ['c:\Temperatures.txt','c:\Log.txt']; (* TODO: Adjust
file paths*)END_VAR
```

**Program-code**

```plaintext
fbSmtpFull(
 sNetId:= '',
 sSmtpServer:= 'mail.company.com',
 sUsername:= 'TestUser',
 sPassword:= 'TestPwd',
 sFrom:= 'TestUser@company.com',
 sTo:= 'service@company.com',
 sSubject:= 'Email from your Beckhoff PLC',
 nContentType:= 2, (* 2 = HTML *)
 nPriority:= 1, (* 1 = HIGH *)
 sAttachments:= sFiles,
 pMessage:= ADR(sMessage_HTML),
 cbMessage:= SIZEOF(sMessage_HTML),
 bExecute:= bStart,
 bError=> bError,
 bBusy=>bBusy,
 nErrId=> nErrId);

IF NOT bBusy AND NOT bError AND bStart THEN
 nState := 0;
 bStart := FALSE;
END_IF
```

**Requirements**

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Smtp</td>
</tr>
</tbody>
</table>

### 4.4 Appendix

#### 4.4.1 Trouble-Shooting

The following list provides basic help in case of errors and should be read *before* contacting our support department.

1. **Check if one of the PLC function blocks returns an error code:**
   
   Please consult the list of error codes:
   
   [SMTP Error codes](#33)
2. **Activate the logging option in the TcSmtpConfig.xml**

   The TcSmtpConfig.xml is located in \TwinCAT\Functions\TF6350-SMS-SMTP

   Enable verbose logging by setting the EnableLogFile = 2 and restart TwinCAT.
   `<TcSmtpConfig> <!-- EnableLogFile: 0 (Disabled), 1 (Enabled), 2 (Verbose) -->
   <EnableLogFile>2</EnableLogFile> ... <Reconnects>5</Reconnects> </TcSmtpConfig>

   The log file will be generate in \TwinCAT\Functions\TF6350-SMS-SMTP .

   If the troubleshooting checklist does not help, please contact our support department and provide the following information:

   **General system information**
   - What kind of hardware is being used on the computer running TF6350 SMS/SMTP?
     - Beckhoff IPC or Embedded PC: Which product number does the PC have?
     - Which Operating System image version is currently installed on that computer?

   **Product-related system information**
   - Which version of TF6350 SMS/STMP is being used?
   - Which function blocks of the Tc2_SMTP library are being used in the PLC program?
   - Which SMTP Server is being used?
     - Microsoft Exchange Server
     - Axigen
     - PostFix
     - WebMail Provider (e.g. GMAIL, Hotmail, GMX)
   - Please provide the SMTP log file (see 2. of the troubleshooting list)
   - Please provide an exact description of the environment in which the product TF6350 SMS/SMTP is being used
     - Where is the computer running TF6350 SMS/SMTP located?
     - Where is the SMTP-Server located? (Local network, Internet)
     - Which encryption is in use? (NON, STARTTLS, SSL)
     - What are the IP settings of the Mail-Server and the computer running TF6350 SMS/SMTP? (IP address, subnet mask, Ports)
     - In case that there is a firewall between both computers:
       - What kind of Firewall system is being used (please provide vendor information)?
       - Has the Firewall been configured to allow SMTP connections?

   **4.4.2 Error Codes**

   This list gives possible error codes for the TwinCAT SMTP Server supplement product. If an error code is generated that is not included in the list, please refer to the ADS Return Codes [34] or the WinSockErrorCodes [34] list.
Error code (hex) | Error code (dec) | Description
--- | --- | ---
< 0x8000 | < 32778 | ADS return code
0x800A | 32778 | Not connected
0x800B | 32779 | Sender expected
0x800C | 32780 | Recipients expected
0x800D | 32781 | Send FROM command failed
0x800E | 32782 | Send DATA command failed
0x800F | 32783 | Send mail header failed
0x8010 | 32784 | Send mail body failed
0x8011 | 32785 | Send "end of mail indicator" failed
0x8012 | 32786 | Send "RCPT" command failed
0x8013 | 32787 | Server Response got no username request
0x8014 | 32788 | Server Response got no password request
0x8015 | 32789 | Unable to create socket connection
0x8016 | 32790 | Authentication type not supported by smtp server
0x8017 | 32791 | Wrong username or password
0x8018 | 32792 | Not supported
0x8019 | 32793 | Invalid hostname
0x801A | 32794 | Unable to send attachment
0x801B | 32795 | File not found
0x801C | 32796 | Invalid Version (New SMTP Server with old SMTP PLC library)
0x801D | 32797 | Unable to connect (Connection error => sometimes wrong port or wrong server)
0x801E | 32798 | Unable to create socket
0x801F | 32799 | WSA startup failed
0x8020 | 32800 | Invalid hostname
0x8021 | 32801 | Unexpected response from server
0x8022 | 32802 | Error while receiving data
0x8023 | 32803 | No supported authentication methods found
0x8024 | 32804 | Invalid parameter
0x80A0 | 32928 | Security interface not found
0x80A1 | 32929 | Unable to call security interface
0x80A2 | 32930 | Security initialization failed
0x80A4 | 32932 | Unable to create credentials
0x80A5 | 32933 | SSL-handshake failed
0x80A6 | 32934 | Invalid server credentials
0x80A7 | 32935 | Unable to verify server
0x80A8 | 32936 | Unable to encrypt message
0x80A9 | 32937 | Unable to decrypt message

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target platform</th>
<th>PLC libraries to include</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.1.4000</td>
<td>PC or CX (x86)</td>
<td>Tc2_Smtp</td>
</tr>
</tbody>
</table>

4.4.3 Windows Socket Error Codes

The following table describes the possible error codes, returned by the WSAGetLastError function. The errors are sorted in alphabetical order. Some error codes that are defined in Winsock2.h are not returned. They are not included in the list.
<table>
<thead>
<tr>
<th>Return Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSAEINTR10004</td>
<td>Interrupted function call. Blocking operation was interrupted by a call to WSACancelBlockingCall.</td>
</tr>
<tr>
<td>WSAEACCES 10013</td>
<td>Permission denied. An attempt was made to access a socket in a way forbidden by its access permissions. An example is using a broadcast address for sendto without broadcast permission being set using setsockopt(SO_BROADCAST). Another possible reason for the WSAEACCES error is that when the bind function is called (on Windows NT 4 SP4 or later), another application, service, or kernel mode driver is bound to the same address with exclusive access. Such exclusive access is a new feature of Windows NT 4 SP4 and later, and is implemented by using the SO_EXCLUSIVEADDRUSE option.</td>
</tr>
<tr>
<td>WSAEFAULT 10014</td>
<td>Bad address. The system detected an invalid pointer address in attempting to use a pointer argument of a call. This error occurs if an application passes an invalid pointer value, or if the length of the buffer is too small. For instance, if the length of an argument, which is a sockaddr structure, is smaller than the sizeof(sockaddr).</td>
</tr>
<tr>
<td>WSAEINVAL 10022</td>
<td>Invalid argument. Some invalid argument was supplied (for example, specifying an invalid level to the setsockopt function). In some instances, it also refers to the current state of the socket—for instance, calling accept on a socket that is not listening.</td>
</tr>
<tr>
<td>WSAEFAULT 10024</td>
<td>Too many open files. Too many open sockets. Each implementation may have a maximum number of socket handles available, either globally, per process, or per thread.</td>
</tr>
<tr>
<td>WSAEWOULDBLOCK 10035</td>
<td>Resource temporarily unavailable. This error is returned from operations on nonblocking sockets that cannot be completed immediately, for example recv when no data is queued to be read from the socket. It is a nonfatal error, and the operation should be retried later. It is normal for WSAEWOULDBLOCK to be reported as the result from calling connect on a nonblocking SOCK_STREAM socket, since some time must elapse for the connection to be established.</td>
</tr>
<tr>
<td>WSAEPROGRESS 10036</td>
<td>Operation now in progress. A blocking operation is currently executing. Windows Sockets only allows a single blocking operation—per- task or thread—to be outstanding, and if any other function call is made (whether or not it references that or any other socket) the function fails with the WSAEPROGRESS error.</td>
</tr>
<tr>
<td>WSAEALREADY 10037</td>
<td>Operation already in progress. An operation was attempted on a nonblocking socket with an operation already in progress—that is, calling connect a second time on a nonblocking socket that is already connecting, or canceling an asynchronous request (WSAAsyncGetXbyY) that has already been canceled or completed.</td>
</tr>
<tr>
<td>WSAENOTSOCK 10038</td>
<td>Socket operation on nonsocket. An operation was attempted on something that is not a socket. Either the socket handle parameter did not reference a valid socket, or for select, a member of an fd_set was not valid.</td>
</tr>
<tr>
<td>WADESTADDRREQ 10039</td>
<td>Destination address required. A required address was omitted from an operation on a socket. For example, this error is returned if sendto is called with the remote address of ADDR_ANY.</td>
</tr>
<tr>
<td>WSAEMSGSIZE 10040</td>
<td>Message too long. A message sent on a datagram socket was larger than the internal message buffer or some other network limit, or the buffer used to receive a datagram was smaller than the datagram itself.</td>
</tr>
<tr>
<td>WSAEPROTOTYPE 10041</td>
<td>Protocol wrong type for socket. A protocol was specified in the socket function call that does not support the semantics of the socket type requested. For example, the ARPA Internet UDP protocol cannot be specified with a socket type of SOCK_STREAM.</td>
</tr>
<tr>
<td>WSAENOPROTOOPT 10042</td>
<td>Bad protocol option. An unknown, invalid or unsupported option or level was specified in a getsockopt or setsockopt call.</td>
</tr>
<tr>
<td>WSAEPROTONOSUPPORT 10043</td>
<td>Protocol not supported. The requested protocol has not been configured into the system, or no implementation for it exists. For example, a socket call requests a SOCK_DGRAM socket, but specifies a stream protocol.</td>
</tr>
<tr>
<td>WSAESOCKTNOSUPPORT 10044</td>
<td>Socket type not supported. The support for the specified socket type does not exist in this address family. For example, the optional type SOCK_RAW might be selected in a socket call, and the implementation does not support SOCK_RAW sockets at all.</td>
</tr>
<tr>
<td>Return Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>WSAEOPNOTSUPP</td>
<td>Operation not supported. The attempted operation is not supported for the type of object referenced. Usually this occurs when a socket descriptor to a socket that cannot support this operation is trying to accept a connection on a datagram socket.</td>
</tr>
<tr>
<td>P 10045</td>
<td>Protocol family not supported. The protocol family has not been configured into the system or no implementation for it exists. This message has a slightly different meaning from WSAEAFNOSUPPORT. However, it is interchangeable in most cases, and all Windows Sockets functions that return one of these messages also specify WSAEAFNOSUPPORT.</td>
</tr>
<tr>
<td>WSAEADDRINUSE</td>
<td>Address family not supported by protocol family. An address incompatible with the requested protocol was used. All sockets are created with an associated address family (that is, AF_INET for Internet Protocols) and a generic protocol type (that is, SOCK_STREAM). This error is returned if an incorrect protocol is explicitly requested in the socket call, or if an address of the wrong family is used for a socket, for example, in sendto.</td>
</tr>
<tr>
<td>10048</td>
<td>Address already in use. Typically, only one usage of each socket address (protocol/IP address/port) is permitted. This error occurs if an application attempts to bind a socket to an IP address/port that has already been used for an existing socket, or a socket that was not closed properly, or one that is still in the process of closing. For server applications that need to bind multiple sockets to the same port number, consider using setsockopt (SO_REUSEADDR). Client applications usually need not call bind at all—connect chooses an unused port automatically. When bind is called with a wildcard address (involving ADDR_ANY), a WSAEADDRINUSE error could be delayed until the specific address is committed. This could happen with a call to another function later, including connect, listen, WSAConnect, or WSAJoinLeaf.</td>
</tr>
<tr>
<td>WSAEADDRNOTAVAL</td>
<td>Cannot assign requested address. The requested address is not valid in its context. This normally results from an attempt to bind to an address that is not valid for the local computer. This can also result from connect, sendto, WSAConnect, WSAJoinLeaf, or WSASendTo when the remote address or port is not valid for a remote computer (for example, address or port 0).</td>
</tr>
<tr>
<td>10049</td>
<td>Network is down. A socket operation encountered a dead network. This could indicate a serious failure of the network system (that is, the protocol stack that the Windows Sockets DLL runs over), the network interface, or the local network itself.</td>
</tr>
<tr>
<td>WSAENETUNREACH</td>
<td>Network is unreachable. A socket operation was attempted to an unreachable network. This usually means the local software knows no route to reach the remote host.</td>
</tr>
<tr>
<td>10051</td>
<td>Network dropped connection on reset. The connection has been broken due to keep-alive activity detecting a failure while the operation was in progress. It can also be returned by setsockopt if an attempt is made to set SO_KEEPALIVE on a connection that has already failed.</td>
</tr>
<tr>
<td>WSAECONNABORTED</td>
<td>Software caused connection abort. An established connection was aborted by the software in your host computer, possibly due to a data transmission time-out or protocol error.</td>
</tr>
<tr>
<td>10053</td>
<td>Connection reset by peer. An existing connection was forcibly closed by the remote host. This normally results if the peer application on the remote host is suddenly stopped, the host is rebooted, the host or remote network interface is disabled, or the remote host uses a hard close (see setsockopt for more information on the SO_LINGER option on the remote socket). This error may also result if a connection was broken due to keep-alive activity detecting a failure while one or more operations are in progress. Operations that were in progress fail with WSAENETRESET. Subsequent operations fail with WSAECONNRESET.</td>
</tr>
<tr>
<td>WSAENOBUFFS</td>
<td>No buffer space available. An operation on a socket could not be performed because the system lacked sufficient buffer space or because a queue was full.</td>
</tr>
<tr>
<td>10055</td>
<td>Socket is already connected. A connect request was made on an already-connected socket. Some implementations also return this error if sendto is called on a connected SOCK_DGRAM socket (for SOCK_STREAM sockets, the to parameter in sendto is ignored) although other implementations treat this as a legal occurrence.</td>
</tr>
<tr>
<td>Return Value</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>WSAENOTCONN 10057</td>
<td>Socket is not connected. A request to send or receive data was disallowed because the socket is not connected and (when sending on a datagram socket using sendto) no address was supplied. Any other type of operation might also return this error—for example, setssockopt setting SO_KEEPALIVE if the connection has been reset.</td>
</tr>
<tr>
<td>WSAESHUTDOWN 10058</td>
<td>Cannot send after socket shutdown. A request to send or receive data was disallowed because the socket had already been shut down in that direction with a previous shutdown call. By calling shutdown a partial close of a socket is requested, which is a signal that sending or receiving, or both have been discontinued.</td>
</tr>
<tr>
<td>WSAETIMEDOUT 10060</td>
<td>Connection timed out. A connection attempt failed because the connected party did not properly respond after a period of time, or the established connection failed because the connected host has failed to respond.</td>
</tr>
<tr>
<td>WSECONNNREFUSED 10061</td>
<td>Connection refused. No connection could be made because the target computer actively refused it. This usually results from trying to connect to a service that is inactive on the foreign host—that is, one with no server application running.</td>
</tr>
<tr>
<td>WSAEHOSTDOWN 10064</td>
<td>Host is down. A socket operation failed because the destination host is down. A socket operation encountered a dead host. Networking activity on the local host has not been initiated. These conditions are more likely to be indicated by the error WSAETIMEDOUT.</td>
</tr>
<tr>
<td>WSAEHOSTUNREACH 10065</td>
<td>No route to host. A socket operation was attempted to an unreachable host. See WSAENETUNREACH.</td>
</tr>
<tr>
<td>WSAEPROCLIM 10067</td>
<td>Too many processes. A Windows Sockets implementation may have a limit on the number of applications that can use it simultaneously. WSAStartup may fail with this error if the limit has been reached.</td>
</tr>
</tbody>
</table>
| WSAYESNOTREADY 10091     | Network subsystem is unavailable. This error is returned by WSAStartup if the Windows Sockets implementation cannot function at this time because the underlying system it uses to provide network services is currently unavailable. Users should check:  
  • That the appropriate Windows Sockets DLL file is in the current path.  
  • That they are not trying to use more than one Windows Sockets implementation simultaneously. If there is more than one Winsock DLL on your system, be sure the first one in the path is appropriate for the network subsystem currently loaded.  
  • The Windows Sockets implementation documentation to be sure all necessary components are currently installed and configured correctly. |
<p>| WSAVERNOT_SUPPORTED 10092| Winsock.dll version out of range. The current Windows Sockets implementation does not support the Windows Sockets specification version requested by the application. Check that no old Windows Sockets DLL files are being accessed.                                                                                           |
| WSANOTINITIALIZED 10093  | Successful WSAStartup not yet performed. Either the application has not called WSAStartup or WSAStartup failed. The application may be accessing a socket that the current active task does not own (that is, trying to share a socket between tasks), or WSACleanup has been called too many times.     |
| WSAEDISCON 10101         | Graceful shutdown in progress. Returned by WSARcv and WSARcvFrom to indicate that the remote party has initiated a graceful shutdown sequence.                                                                                                                                                                                                    |
| WSIATYPE_NOTFOUND 10109  | Class type not found. The specified class was not found.                                                                                                                                                                                                                                                                                       |
| WSASALISCON 11001        | Host not found. No such host is known. The name is not an official host name or alias, or it cannot be found in the database(s) being queried. This error may also be returned for protocol and service queries, and means that the specified name could not be found in the relevant database.                                                                                                    |
| WSATRY_AGAIN 11002       | Nonauthoritative host not found. This is usually a temporary error during host name resolution and means that the local server did not receive a response from an authoritative server. A retry at some time later may be successful.                                                                                                                   |</p>
<table>
<thead>
<tr>
<th>Return Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSANO_RECOVERY 11003</td>
<td>This is a nonrecoverable error. This indicates that some sort of nonrecoverable error occurred during a database lookup. This may be because the database files (for example, BSD-compatible HOSTS, SERVICES, or PROTOCOLS files) could not be found, or a DNS request was returned by the server with a severe error.</td>
</tr>
<tr>
<td>WSANO_DATA 11004</td>
<td>Valid name, no data record of requested type. The requested name is valid and was found in the example database, but it does not have the correct associated data being resolved for. The usual example for this is a host name-to-address translation attempt (using gethostbyname or WSAAsyncGetHostByName) which uses the DNS (Domain Name Server). An MX record is returned but no A record—indicating the host itself exists, but is not directly reachable.</td>
</tr>
<tr>
<td>WSA_INVALID_HANDLE</td>
<td>Specified event object handle is invalid. An application attempts to use an event object, but the specified handle is not valid.</td>
</tr>
<tr>
<td>WSA_INVALID_PARAMETER</td>
<td>One or more parameters are invalid. An application used a Windows Sockets function which directly maps to a Windows function. The Windows function is indicating a problem with one or more parameters.</td>
</tr>
<tr>
<td>WSA_IO_INCOMPLETE</td>
<td>Overlapped I/O event object not in signaled state. The application has tried to determine the status of an overlapped operation which is not yet completed. Applications that use WSAGetOverlappedResult (with the fWait flag set to FALSE) in a polling mode to determine when an overlapped operation has completed, get this error code until the operation is complete.</td>
</tr>
<tr>
<td>WSA_IO_PENDING</td>
<td>Overlapped operations will complete later. The application has initiated an overlapped operation that cannot be completed immediately. A completion indication will be given later when the operation has been completed.</td>
</tr>
<tr>
<td>WSA_NOT_ENOUGH_MEMORY</td>
<td>Insufficient memory available. An application used a Windows Sockets function that directly maps to a Windows function. The Windows function is indicating a lack of required memory resources.</td>
</tr>
<tr>
<td>WSA_OPERATION_ABORTED</td>
<td>Overlapped operation aborted. An overlapped operation was canceled due to the closure of the socket, or the execution of the SIO_FLUSH command in WSAIoctl.</td>
</tr>
<tr>
<td>WSAINVALIDPROC_TABLE</td>
<td>Invalid procedure table from service provider. A service provider returned a bogus procedure table to Ws2_32.dll. (This is usually caused by one or more of the function pointers being null.)</td>
</tr>
<tr>
<td>WSAINVALIDPROVIDER</td>
<td>Invalid service provider version number. A service provider returned a version number other than 2.0.</td>
</tr>
<tr>
<td>WSA_PROVIDERFAILED</td>
<td>Unable to initialize a service provider. Either a service provider's DLL could not be loaded (LoadLibrary failed) or the provider's WSPStartup/NSPStartup function failed.</td>
</tr>
<tr>
<td>WSASYSCALLFAILURE</td>
<td>System call failure. Generic error code, returned under various conditions. Returned when a system call that should never fail does fail. For example, if a call to WaitForMultipleEvents fails or one of the registry functions fails trying to manipulate the protocol/namespace catalogs. Returned when a provider does not return SUCCESS and does not provide an extended error code. Can indicate a service provider implementation error.</td>
</tr>
</tbody>
</table>

**Requirements**

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Smtp</td>
</tr>
</tbody>
</table>
5 TwinCAT SMS

The TwinCAT SMS library contains library for sending SMS messages directly from the PLC. The SMS library is based on the 'Serial Communication' library, which will be installed, too. This makes it possible to communicate with the PC's serial interface and with the serial terminal (EL6xxx and KL6xxx) in the same way.

Additionally it is possible to provide status or alerts from other TwinCAT system, which are connected via network.

5.1 PLC API

5.1.1 Function blocks

5.1.1.1 SendSMS Function Block

The SendSMS function block allows an SMS to be sent via a connected GSM modem. The function block is based on the 'Serial Communication' library.

Because the block only communicates via the ComBuffer structure in the 'Serial Communication' library, instances can be formed, and it can be applied to every kind of serial interface.
**VAR_INPUT**

- **Send**: BOOL;
- **Number**: String;
- **Text**: String(160);

**Send**: The function block is activated by a positive edge at this input.

**Number**: telephone number to be dialled in national format (e.g.: 0170123456)

**Text**: The SMS message to be sent

**VAR_OUTPUT**

- **Busy**: BOOL;
- **Error**: INT;

**Busy**: This output is set when there is a rising edge at the Send input, and remains set until the SMS has been sent to the modem or until an error has occurred.

**Error**: If an error occurs while the SMS is being transferred, the Busy output is reset, and an error code is made available at the Error output. If the Error output is 0, the transfer was successful.

The function block can return the following errors:

<table>
<thead>
<tr>
<th>Number</th>
<th>Meaning</th>
<th>Cause</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Communication with the modem is not possible.</td>
<td>Is the terminal correctly configured? Has the appropriate ComLib library been used?</td>
</tr>
<tr>
<td>2</td>
<td>Modem reports an error during configuration.</td>
<td>Is a compatible GSM modem connected?</td>
</tr>
<tr>
<td>3</td>
<td>Modem can not send SMS.</td>
<td>Is the SIM card working properly? Can the card be used without entering the PIN? Is the modem connected to the network? Is a compatible modem connected?</td>
</tr>
<tr>
<td>4</td>
<td>Communication error.</td>
<td>Has the correct transmission speed been set?</td>
</tr>
</tbody>
</table>

**VAR_IN_OUT**

- **RXbuffer**: ComBuffer;
- **TXbuffer**: ComBuffer;

**RXbuffer**: Structure for communication with the serial interface. An interface-specific block in the ‘Serial Communication’ library fills this buffer with the data for the interface.

**TXbuffer**: Structure for communication with the serial interface. An interface-specific block in the ‘Serial Communication’ library transfers the data from this buffer to the interface.

These structures, and their usage, are described in more detail in the documentation for the ‘Serial Communication’ library. The SendSMS block is here connected to a SendString or ReceiveString block.

**Requirements**

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Sms</td>
</tr>
</tbody>
</table>
5.1.2 Functions

5.1.2.1 FUNCTION Get_TcPlcSMS_Version

The function returns library version info.

FUNCTION Get_TcPlcSMS_Version: STRING(20)

VAR_INPUT

bGet : BOOL;

bGet: The compiler requires at least one input parameter for functions. You can set this parameter to TRUE or FALSE.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Sms</td>
</tr>
</tbody>
</table>

5.1.3 Global constants

5.1.3.1 Library 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:

Global_Version

VAR_GLOBAL CONSTANT

   stLibVersion_Tc2_SMS_SMTP : ST_LibVersion;
END_VAR

ST_LibVersion

To compare the existing version to a required version the function F_CmpLibVersion (defined in Tc2_System library) is offered.

Compare versions

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

5.2.1 Sending an SMS

It is possible to communicate with the PC’s serial interface and with the serial terminal (EL6xxx and KL6xxx) in the same way.

1. Add the Tc2_SMS and Tc2_SerialCom Library to your PLC project
2. Follow the tutorial for the ‘Serial Communication’ library to configure the communication
3. Check the documentation of your used GSM modem to disable the PIN request (the SIM card should not be protected by a PIN)

Supported devices:
- Westermo GS-01 (communication parameters: 9600 baud, 8 data bits, no parity bit, one stop bit)
- Maestro 100 (communication parameters: 9600 baud, 8 data bits, no parity bit, one stop bit)
- For further device tests please contact support@beckhoff.com

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_Sms, Tc2_SerialCom Library</td>
</tr>
</tbody>
</table>

5.3 Appendix

5.3.1 Fault Finding

There are a number of reasons why an SMS may fail to be sent with the SendSMS function block or SMS COM Server:

- no connection to the GSM modem
- incorrectly configured communication settings of SMS COM Server
- incorrect call to the ADS service
- the use of an unsupported GSM modem
- incorrectly configured serial terminal (Advanced or Standard, 3 byte / 5 byte, speed, ...)
- incorrect telephone number
- PIN required (the SIM card must not be protected by a PIN)
- Serial terminal not initialised (call KL6Init)
- incorrect in GSM network

A variety of tools are available to look for these errors:

Using the Log File

Keeping records in a log file can be activated with the TwinCAT SMS Server Configurator. Once this has been done, all the messages sent and the errors are written into the TcSmsSrvCfg.xml file. The file can be found in the TwinCAT installation directory.

NT Event Log

Errors when sending messages are also always recorded in the NT Event Log. The Event Log can be opened through the TwinCAT icon on the task bar.
ADS Error Messages

If the call to an ADS Function fails, the error is coded in the function's return value. A list of these error codes can be found under ADS Return Codes.[44]

Configuration of the Terminal

The serial terminal can be configured in different ways. Terminals that have been differently configured, have to some extent a different representation in the process image (3 byte /5 byte terminals, advanced/standard). It must be noted that the ComLib library must be appropriate for the terminal configuration. See also the documentation for the KL6xxx and the ComLib documentation:

It is also important that the terminal transmission speed be matched to that of the modem in use.

Sending a Test SMS

A test SMS can easily be sent with the Visual Basic example program, to find out whether an error lies with the ADS call or in the configuration of the SMS Server.

Sending a Test SMS using a Mobile Telephone

To find out whether the SIM card is correctly configured, it can be inserted into an ordinary mobile phone and used to send an SMS. It should not be necessary to enter a PIN number here.

Network Selection with the Westermo GS-01

GS-01 has a number of variations for the various networks in Europe and in the USA. The lamp on the front of the modem indicates whether a network is available. The lamp flashes if the modem is connected to a network. If the lamp is continuously illuminated, the fault finding section should be consulted in the Westermo manual.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Target system type</th>
<th>PLC libraries to be linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT v3.0.0</td>
<td>PC or CX (x86)</td>
<td>Tc2_</td>
</tr>
</tbody>
</table>
6 Appendix

6.1 ADS Return Codes

Grouping of error codes: 0x000...44, 0x500...44, 0x700...45, 0x1000...47...

Global error codes

<table>
<thead>
<tr>
<th>Hex</th>
<th>Dec</th>
<th>HRESULT</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0</td>
<td>0</td>
<td>0x9811 0000</td>
<td>ERR_NOERROR</td>
<td>No error.</td>
</tr>
<tr>
<td>0x1</td>
<td>1</td>
<td>0x9811 0001</td>
<td>ERR_INTERNAL</td>
<td>Internal error.</td>
</tr>
<tr>
<td>0x2</td>
<td>2</td>
<td>0x9811 0002</td>
<td>ERR_NORTIME</td>
<td>No real-time.</td>
</tr>
<tr>
<td>0x3</td>
<td>3</td>
<td>0x9811 0003</td>
<td>ERR_ALLOCCLOCKEDMEM</td>
<td>Allocation locked – memory error.</td>
</tr>
<tr>
<td>0x4</td>
<td>4</td>
<td>0x9811 0004</td>
<td>ERR_INSERTMAILBOX</td>
<td>Mailbox full – the ADS message could not be sent. Reducing the number of ADS messages per cycle will help.</td>
</tr>
<tr>
<td>0x5</td>
<td>5</td>
<td>0x9811 0005</td>
<td>ERR_WRONGRECEIVEHMSG</td>
<td>Wrong HMSG.</td>
</tr>
<tr>
<td>0x6</td>
<td>6</td>
<td>0x9811 0006</td>
<td>ERR_TARGETPORTNOTFOUND</td>
<td>Target port not found – ADS server is not started or is not reachable.</td>
</tr>
<tr>
<td>0x7</td>
<td>7</td>
<td>0x9811 0007</td>
<td>ERR_TARGETMACHINENOTFOUND</td>
<td>Target computer not found – AMS route was not found.</td>
</tr>
<tr>
<td>0x8</td>
<td>8</td>
<td>0x9811 0008</td>
<td>ERR_UNKNOWNCMDID</td>
<td>Unknown command ID.</td>
</tr>
<tr>
<td>0x9</td>
<td>9</td>
<td>0x9811 0009</td>
<td>ERR_BADTASKID</td>
<td>Invalid task ID.</td>
</tr>
<tr>
<td>0xA</td>
<td>10</td>
<td>0x9811 000A</td>
<td>ERR_NOIO</td>
<td>No IO.</td>
</tr>
<tr>
<td>0xB</td>
<td>11</td>
<td>0x9811 000B</td>
<td>ERR_UNKNOWNAMSCMD</td>
<td>Unknown AMS command.</td>
</tr>
<tr>
<td>0xC</td>
<td>12</td>
<td>0x9811 000C</td>
<td>ERR_WIN32ERROR</td>
<td>Win32 error.</td>
</tr>
<tr>
<td>0xD</td>
<td>13</td>
<td>0x9811 000D</td>
<td>ERR_PORTNOTCONNECTED</td>
<td>Port not connected.</td>
</tr>
<tr>
<td>0xE</td>
<td>14</td>
<td>0x9811 000E</td>
<td>ERR_INVALIDAMSLENGTH</td>
<td>Invalid AMS length.</td>
</tr>
<tr>
<td>0xF</td>
<td>15</td>
<td>0x9811 000F</td>
<td>ERR_INVALIDAMSNETID</td>
<td>Invalid AMS Net ID.</td>
</tr>
<tr>
<td>0x10</td>
<td>16</td>
<td>0x9811 0010</td>
<td>ERR_LOWINSTLEVEL</td>
<td>Installation level is too low – TwinCAT 2 license error.</td>
</tr>
<tr>
<td>0x11</td>
<td>17</td>
<td>0x9811 0011</td>
<td>ERR_NODEBUGINTAVAILABLE</td>
<td>No debugging available.</td>
</tr>
<tr>
<td>0x12</td>
<td>18</td>
<td>0x9811 0012</td>
<td>ERR_PORTDISABLED</td>
<td>Port disabled – TwinCAT system service not started.</td>
</tr>
<tr>
<td>0x13</td>
<td>19</td>
<td>0x9811 0013</td>
<td>ERR_PORTALREADYCONNECTED</td>
<td>Port already connected.</td>
</tr>
<tr>
<td>0x14</td>
<td>20</td>
<td>0x9811 0014</td>
<td>ERR_AMSSYNC_W32ERROR</td>
<td>AMS Sync Win32 error.</td>
</tr>
<tr>
<td>0x15</td>
<td>21</td>
<td>0x9811 0015</td>
<td>ERR_AMSSYNC_TIMEOUT</td>
<td>AMS Sync Timeout.</td>
</tr>
<tr>
<td>0x16</td>
<td>22</td>
<td>0x9811 0016</td>
<td>ERR_AMSSYNC_AMSERROR</td>
<td>AMS Sync error.</td>
</tr>
<tr>
<td>0x17</td>
<td>23</td>
<td>0x9811 0017</td>
<td>ERR_AMSSYNC_NOINDEXINMAP</td>
<td>No index map for AMS Sync available.</td>
</tr>
<tr>
<td>0x18</td>
<td>24</td>
<td>0x9811 0018</td>
<td>ERR_INVALIDAMSPORT</td>
<td>Invalid AMS port.</td>
</tr>
<tr>
<td>0x19</td>
<td>25</td>
<td>0x9811 0019</td>
<td>ERR_NOMEMORY</td>
<td>No memory.</td>
</tr>
<tr>
<td>0x1A</td>
<td>26</td>
<td>0x9811 001A</td>
<td>ERR_TCPSEND</td>
<td>TCP send error.</td>
</tr>
<tr>
<td>0x1B</td>
<td>27</td>
<td>0x9811 001B</td>
<td>ERR_HOSTUNREACHABLE</td>
<td>Host unreachable.</td>
</tr>
<tr>
<td>0x1C</td>
<td>28</td>
<td>0x9811 001C</td>
<td>ERR_INVALIDAMSFRAAGMENT</td>
<td>Invalid AMS fragment.</td>
</tr>
<tr>
<td>0x1D</td>
<td>29</td>
<td>0x9811 001D</td>
<td>ERR_TLSSEND</td>
<td>TLS send error – secure ADS connection failed.</td>
</tr>
<tr>
<td>0x1E</td>
<td>30</td>
<td>0x9811 001E</td>
<td>ERR_ACCESSDENIED</td>
<td>Access denied – secure ADS access denied.</td>
</tr>
</tbody>
</table>

Router error codes
<table>
<thead>
<tr>
<th>Hex</th>
<th>Dec</th>
<th>HRESULT</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x500</td>
<td>1280</td>
<td>0x9811 0500</td>
<td>ROUTERERR_NOLOCKEDMEMORY</td>
<td>Locked memory cannot be allocated.</td>
</tr>
<tr>
<td>0x501</td>
<td>1281</td>
<td>0x9811 0501</td>
<td>ROUTERERR_RESIZEMEMORY</td>
<td>The router memory size could not be changed.</td>
</tr>
<tr>
<td>0x502</td>
<td>1282</td>
<td>0x9811 0502</td>
<td>ROUTERERR_MAILBOXFULL</td>
<td>The mailbox has reached the maximum number of possible messages.</td>
</tr>
<tr>
<td>0x503</td>
<td>1283</td>
<td>0x9811 0503</td>
<td>ROUTERERR_DEBUGBOXFULL</td>
<td>The Debug mailbox has reached the maximum number of possible messages.</td>
</tr>
<tr>
<td>0x504</td>
<td>1284</td>
<td>0x9811 0504</td>
<td>ROUTERERR_UNKNOWNPORTTYPE</td>
<td>The port type is unknown.</td>
</tr>
<tr>
<td>0x505</td>
<td>1285</td>
<td>0x9811 0505</td>
<td>ROUTERERR_NOTINITIALIZED</td>
<td>The router is not initialized.</td>
</tr>
<tr>
<td>0x506</td>
<td>1286</td>
<td>0x9811 0506</td>
<td>ROUTERERR_PORTALREADYINUSE</td>
<td>The port number is already assigned.</td>
</tr>
<tr>
<td>0x507</td>
<td>1287</td>
<td>0x9811 0507</td>
<td>ROUTERERR_NOTREGISTERED</td>
<td>The port is not registered.</td>
</tr>
<tr>
<td>0x508</td>
<td>1288</td>
<td>0x9811 0508</td>
<td>ROUTERERR_NOMOREQUEUES</td>
<td>The maximum number of ports has been reached.</td>
</tr>
<tr>
<td>0x509</td>
<td>1289</td>
<td>0x9811 0509</td>
<td>ROUTERERR_INVALIDPORT</td>
<td>The port is invalid.</td>
</tr>
<tr>
<td>0x50A</td>
<td>1290</td>
<td>0x9811 050A</td>
<td>ROUTERERR_NOTACTIVATED</td>
<td>The router is not active.</td>
</tr>
<tr>
<td>0x50B</td>
<td>1291</td>
<td>0x9811 050B</td>
<td>ROUTERERR_FRAGMENTBOXFULL</td>
<td>The mailbox has reached the maximum number for fragmented messages.</td>
</tr>
<tr>
<td>0x50C</td>
<td>1292</td>
<td>0x9811 050C</td>
<td>ROUTERERR_FRAGMENTTIMEOUT</td>
<td>A fragment timeout has occurred.</td>
</tr>
<tr>
<td>0x50D</td>
<td>1293</td>
<td>0x9811 050D</td>
<td>ROUTERERR_TOBEREMOVED</td>
<td>The port is removed.</td>
</tr>
</tbody>
</table>

**General ADS error codes**
<table>
<thead>
<tr>
<th>Hex</th>
<th>Dec</th>
<th>HRESULT</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x700</td>
<td>1792</td>
<td>0x9811 0700</td>
<td>ADSERR_DEVICE_ERROR</td>
<td>General device error.</td>
</tr>
<tr>
<td>0x701</td>
<td>1793</td>
<td>0x9811 0701</td>
<td>ADSERR_DEVICE_SRVNOTSUPP</td>
<td>Service is not supported by the server.</td>
</tr>
<tr>
<td>0x702</td>
<td>1794</td>
<td>0x9811 0702</td>
<td>ADSERR_DEVICE_INVALIDGRP</td>
<td>Invalid index group.</td>
</tr>
<tr>
<td>0x703</td>
<td>1795</td>
<td>0x9811 0703</td>
<td>ADSERR_DEVICE_INVALIDOFFSET</td>
<td>Invalid index offset.</td>
</tr>
<tr>
<td>0x704</td>
<td>1796</td>
<td>0x9811 0704</td>
<td>ADSERR_DEVICE_INVALIDACCESS</td>
<td>Reading or writing not permitted.</td>
</tr>
<tr>
<td>0x705</td>
<td>1797</td>
<td>0x9811 0705</td>
<td>ADSERR_DEVICE_INVALIDIDSIZE</td>
<td>Parameter size not correct.</td>
</tr>
<tr>
<td>0x706</td>
<td>1798</td>
<td>0x9811 0706</td>
<td>ADSERR_DEVICE_INVALIDDATA</td>
<td>Invalid data values.</td>
</tr>
<tr>
<td>0x707</td>
<td>1799</td>
<td>0x9811 0707</td>
<td>ADSERR_DEVICE_NOTREADY</td>
<td>Device is not ready to operate.</td>
</tr>
<tr>
<td>0x708</td>
<td>1800</td>
<td>0x9811 0708</td>
<td>ADSERR_DEVICE_BUSY</td>
<td>Device is busy.</td>
</tr>
<tr>
<td>0x709</td>
<td>1801</td>
<td>0x9811 0709</td>
<td>ADSERR_DEVICE_INVALIDCONTEXT</td>
<td>Invalid operating system context. This can result from use of ADS function blocks in different tasks. It may be possible to resolve this through Multi-task data access synchronization in the PLC.</td>
</tr>
<tr>
<td>0x70A</td>
<td>1802</td>
<td>0x9811 070A</td>
<td>ADSERR_DEVICE_NOMEMORY</td>
<td>Insufficient memory.</td>
</tr>
<tr>
<td>0x70B</td>
<td>1803</td>
<td>0x9811 070B</td>
<td>ADSERR_DEVICE_INVALIDPARM</td>
<td>Invalid parameter values.</td>
</tr>
<tr>
<td>0x70C</td>
<td>1804</td>
<td>0x9811 070C</td>
<td>ADSERR_DEVICE_NOTFOUND</td>
<td>Not found (files, ...).</td>
</tr>
<tr>
<td>0x70D</td>
<td>1805</td>
<td>0x9811 070D</td>
<td>ADSERR_DEVICE_SYNTAX</td>
<td>Syntax error in file or command.</td>
</tr>
<tr>
<td>0x70E</td>
<td>1806</td>
<td>0x9811 070E</td>
<td>ADSERR_DEVICE_INCOMPATIBLE</td>
<td>Objects do not match.</td>
</tr>
<tr>
<td>0x70F</td>
<td>1807</td>
<td>0x9811 070F</td>
<td>ADSERR_DEVICE_EXISTS</td>
<td>Object already exists.</td>
</tr>
<tr>
<td>0x710</td>
<td>1808</td>
<td>0x9811 0710</td>
<td>ADSERR_DEVICE_SYMBOLNOTFOUND</td>
<td>Symbol not found.</td>
</tr>
<tr>
<td>0x711</td>
<td>1809</td>
<td>0x9811 0711</td>
<td>ADSERR_DEVICE_SYMBOLVERSIONINVALID</td>
<td>Invalid symbol version. This can occur due to an online change. Create a new handle.</td>
</tr>
<tr>
<td>0x712</td>
<td>1810</td>
<td>0x9811 0712</td>
<td>ADSERR_DEVICEINVALIDSTATE</td>
<td>Device (server) is in invalid state.</td>
</tr>
<tr>
<td>0x713</td>
<td>1811</td>
<td>0x9811 0713</td>
<td>ADSERR_DEVICE_TRANSMODENOTSUPP</td>
<td>AdsTransMode not supported.</td>
</tr>
<tr>
<td>0x714</td>
<td>1812</td>
<td>0x9811 0714</td>
<td>ADSERR_DEVICE_NOTIFYHIGHINVALID</td>
<td>Notification handle is invalid.</td>
</tr>
<tr>
<td>0x715</td>
<td>1813</td>
<td>0x9811 0715</td>
<td>ADSERR_DEVICE_CLIENTUNKNOWN</td>
<td>Notification client not registered.</td>
</tr>
<tr>
<td>0x716</td>
<td>1814</td>
<td>0x9811 0716</td>
<td>ADSERR_DEVICE_NOMOREHDLSS</td>
<td>No further handle available.</td>
</tr>
<tr>
<td>0x717</td>
<td>1815</td>
<td>0x9811 0717</td>
<td>ADSERR_DEVICE_INVALIDWATCHSIZE</td>
<td>Notification size too large.</td>
</tr>
<tr>
<td>0x718</td>
<td>1816</td>
<td>0x9811 0718</td>
<td>ADSERR_DEVICE_NOTINIT</td>
<td>Device not initialized.</td>
</tr>
<tr>
<td>0x719</td>
<td>1817</td>
<td>0x9811 0719</td>
<td>ADSERR_DEVICE_TIMEOUT</td>
<td>Device has a timeout.</td>
</tr>
<tr>
<td>0x71A</td>
<td>1818</td>
<td>0x9811 071A</td>
<td>ADSERR_DEVICE_NOTINTERFACE</td>
<td>Interface query failed.</td>
</tr>
<tr>
<td>0x71B</td>
<td>1819</td>
<td>0x9811 071B</td>
<td>ADSERR_DEVICE_INVALIDINTERFACE</td>
<td>Wrong interface requested.</td>
</tr>
<tr>
<td>0x71C</td>
<td>1820</td>
<td>0x9811 071C</td>
<td>ADSERRDEVICE_INVALIDIDCLASS</td>
<td>Class ID is invalid.</td>
</tr>
<tr>
<td>0x71D</td>
<td>1821</td>
<td>0x9811 071D</td>
<td>ADSERR_DEVICE_INVALIDOBJID</td>
<td>Object ID is invalid.</td>
</tr>
<tr>
<td>0x71E</td>
<td>1822</td>
<td>0x9811 071E</td>
<td>ADSERR_DEVICE_PENDIND</td>
<td>Request pending.</td>
</tr>
<tr>
<td>0x71F</td>
<td>1823</td>
<td>0x9811 071F</td>
<td>ADSERR_DEVICE_ABORTED</td>
<td>Request is aborted.</td>
</tr>
<tr>
<td>0x720</td>
<td>1824</td>
<td>0x9811 0720</td>
<td>ADSERR_DEVICE_WARNING</td>
<td>Signal warning.</td>
</tr>
<tr>
<td>0x721</td>
<td>1825</td>
<td>0x9811 0721</td>
<td>ADSERR_DEVICE_INVALIDARRAYIDX</td>
<td>Invalid array index.</td>
</tr>
<tr>
<td>0x722</td>
<td>1826</td>
<td>0x9811 0722</td>
<td>ADSERR_DEVICE_SYMBOLNOTACTIVE</td>
<td>Symbol not active.</td>
</tr>
<tr>
<td>0x723</td>
<td>1827</td>
<td>0x9811 0723</td>
<td>ADSERR_DEVICE_ACCESSDENIED</td>
<td>Access denied.</td>
</tr>
<tr>
<td>0x724</td>
<td>1828</td>
<td>0x9811 0724</td>
<td>ADSERR_DEVICE_LICENSENOTFOUND</td>
<td>Missing license.</td>
</tr>
<tr>
<td>0x725</td>
<td>1829</td>
<td>0x9811 0725</td>
<td>ADSERR_DEVICE_LICENSEEXPIRED</td>
<td>License expired.</td>
</tr>
<tr>
<td>0x726</td>
<td>1830</td>
<td>0x9811 0726</td>
<td>ADSERR_DEVICE_LICENSEEXCEEDED</td>
<td>License exceeded.</td>
</tr>
<tr>
<td>0x727</td>
<td>1831</td>
<td>0x9811 0727</td>
<td>ADSERR_DEVICE_LICENSEINVALID</td>
<td>Invalid license.</td>
</tr>
<tr>
<td>0x728</td>
<td>1832</td>
<td>0x9811 0728</td>
<td>ADSERR_DEVICE_LICENSESYSTEMID</td>
<td>License problem: System ID is invalid.</td>
</tr>
<tr>
<td>0x729</td>
<td>1833</td>
<td>0x9811 0729</td>
<td>ADSERR_DEVICE_LICENSENOSTIMELIMIT</td>
<td>License not limited in time.</td>
</tr>
<tr>
<td>0x72A</td>
<td>1834</td>
<td>0x9811 072A</td>
<td>ADSERR_DEVICE_LICENSEFUTUREISSUE</td>
<td>License problem: Time in the future.</td>
</tr>
<tr>
<td>0x72B</td>
<td>1835</td>
<td>0x9811 072B</td>
<td>ADSERR_DEVICE_LICENSETIMELOMLONG</td>
<td>License period too long.</td>
</tr>
<tr>
<td>0x72C</td>
<td>1836</td>
<td>0x9811 072C</td>
<td>ADSERRDEVICE_EXCEPTION</td>
<td>Exception at system startup.</td>
</tr>
<tr>
<td>0x72D</td>
<td>1837</td>
<td>0x9811 072D</td>
<td>ADSERR_DEVICE_LICENSEDUPPLICATED</td>
<td>License file read twice.</td>
</tr>
<tr>
<td>0x72E</td>
<td>1838</td>
<td>0x9811 072E</td>
<td>ADSERR_DEVICE_SIGNATUREINVALID</td>
<td>Invalid signature.</td>
</tr>
<tr>
<td>0x72F</td>
<td>1839</td>
<td>0x9811 072F</td>
<td>ADSERR_DEVICE_CERTIFICATEINVALID</td>
<td>Invalid certificate.</td>
</tr>
<tr>
<td>0x730</td>
<td>1840</td>
<td>0x9811 0730</td>
<td>ADSERR_DEVICE_LICENSEOEMNOTFOUND</td>
<td>Public key not known from OEM.</td>
</tr>
<tr>
<td>0x731</td>
<td>1841</td>
<td>0x9811 0731</td>
<td>ADSERR_DEVICE_LICENSERESTRICTED</td>
<td>License not valid for this system ID.</td>
</tr>
<tr>
<td>0x732</td>
<td>1842</td>
<td>0x9811 0732</td>
<td>ADSERR_DEVICE_LICENSEDEMODENIED</td>
<td>Demo license prohibited.</td>
</tr>
<tr>
<td>0x733</td>
<td>1843</td>
<td>0x9811 0733</td>
<td>ADSERR_DEVICE_INVALIDFNCID</td>
<td>Invalid function ID.</td>
</tr>
<tr>
<td>0x734</td>
<td>1844</td>
<td>0x9811 0734</td>
<td>ADSERR_DEVICE_OUTOF Range</td>
<td>Outside the valid range.</td>
</tr>
<tr>
<td>0x735</td>
<td>1845</td>
<td>0x9811 0735</td>
<td>ADSERR_DEVICE_INVALIDALIGNMENT</td>
<td>Invalid alignment.</td>
</tr>
</tbody>
</table>

**Appendix**

**TF6350**

Version: 1.2
### TCP Winsock error codes

<table>
<thead>
<tr>
<th>Hex</th>
<th>Dec</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x274C</td>
<td>10060</td>
<td>WSAETIMEOUT</td>
<td>A connection timeout has occurred - error while establishing the connection, because the remote terminal did not respond properly after a certain period of time, or the established connection could not be maintained because the connected host did not respond.</td>
</tr>
<tr>
<td>0x274D</td>
<td>10061</td>
<td>WSAECONNREFUSED</td>
<td>Connection refused - no connection could be established because the target computer has explicitly rejected it. This error usually results from an attempt to connect to a service that is inactive on the external host, that is, a service for which no server application is running.</td>
</tr>
<tr>
<td>0x2751</td>
<td>10065</td>
<td>WSAEHOSTUNREACH</td>
<td>No route to host - a socket operation referred to an unavailable host.</td>
</tr>
</tbody>
</table>

More Winsock error codes: Win32 error codes
6.2 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

Beckhoff’s branch offices and representatives

Please contact your Beckhoff branch office or representative for local support and service on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on her internet pages: https://www.beckhoff.com

You will also find further documentation for Beckhoff components there.

Beckhoff Support

Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline: +49 5246 963 157
Fax: +49 5246 963 9157
e-mail: support@beckhoff.com

Beckhoff Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline: +49 5246 963 460
Fax: +49 5246 963 479
e-mail: service@beckhoff.com

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG
Huelshorstweg 20
33415 Verl
Germany

Phone: +49 5246 963 0
Fax: +49 5246 963 198
e-mail: info@beckhoff.com
web: https://www.beckhoff.com