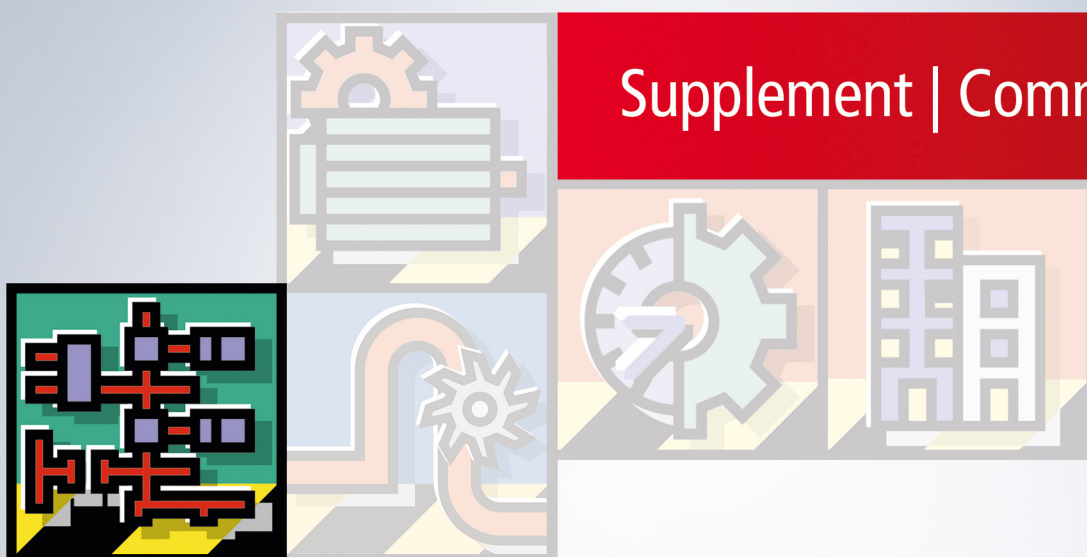


Handbuch | DE

# TS6350

TwinCAT 2 | SMS/SMTP Server





# Inhaltsverzeichnis

<b>1</b>	<b>Vorwort</b>	<b>5</b>
1.1	Hinweise zur Dokumentation	5
1.2	Zu Ihrer Sicherheit	6
1.3	Hinweise zur Informationssicherheit	7
<b>2</b>	<b>Übersicht</b>	<b>8</b>
<b>3</b>	<b>SMS Server</b>	<b>9</b>
3.1	SPS-Bibliotheken	9
3.1.1	SendSMS	10
3.1.2	Get_TcPlcSMS_Version	11
3.1.3	Get_TcPlsSMSBC_Version	11
3.1.4	Beispiele	12
3.2	ADS-Gerät	17
3.2.1	Konfiguration	17
3.2.2	ADS-Interface	19
3.2.3	Beispiele	24
3.3	Fehlersuche	26
3.4	Serielles Kabel für KL6001	27
3.5	7 Bit GSM default alphabet coding	27
3.6	Syntax der Geräte-Steuerzeichenkette	33
<b>4</b>	<b>SMTP Server</b>	<b>35</b>
4.1	Konfiguration	35
4.2	Funktionsbausteine	36
4.2.1	FB_Smtp	36
4.2.2	FB_SmtpAttach	37
4.2.3	FB_SmtpFull	39
4.2.4	FB_SmtpV2	42
4.2.5	FB_SmtpV3	44
4.2.6	FB_SmtpV3_Full	46
4.3	Beispiele	48
4.3.1	How to - Best practice	48
4.3.2	Beispiel: Mailversand aus der SPS	49
4.3.3	Beispiel: Versenden von Mails in HTML	50
4.4	Anhang	51
4.4.1	Fehlersuche	51
4.4.2	Fehlercodes	51
4.4.3	Windows Socket Fehler Codes	53



# 1 Vorwort

## 1.1 Hinweise zur Dokumentation

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist.

Zur Installation und Inbetriebnahme der Komponenten ist die Beachtung der Dokumentation und der nachfolgenden Hinweise und Erklärungen unbedingt notwendig.

Das Fachpersonal ist verpflichtet, für jede Installation und Inbetriebnahme die zu dem betreffenden Zeitpunkt veröffentlichte Dokumentation zu verwenden.

Das Fachpersonal hat sicherzustellen, dass die Anwendung bzw. der Einsatz der beschriebenen Produkte alle Sicherheitsanforderungen, einschließlich sämtlicher anwendbaren Gesetze, Vorschriften, Bestimmungen und Normen erfüllt.

### Disclaimer

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte werden jedoch ständig weiter entwickelt.

Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.

Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

### Marken

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® und XPlanar® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH.

Die Verwendung anderer in dieser Dokumentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Bezeichnungen führen.

### Patente

Die EtherCAT-Technologie ist patentrechtlich geschützt, insbesondere durch folgende Anmeldungen und Patente:

EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702

mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.



EtherCAT® ist eine eingetragene Marke und patentierte Technologie lizenziert durch die Beckhoff Automation GmbH, Deutschland

### Copyright

© Beckhoff Automation GmbH & Co. KG, Deutschland.

Weitergabe sowie Vervielfältigung dieses Dokuments, Verwertung und Mitteilung seines Inhalts sind verboten, soweit nicht ausdrücklich gestattet.

Zu widerhandlungen verpflichten zu Schadenersatz. Alle Rechte für den Fall der Patent-, Gebrauchsmuster- oder Geschmacksmustereintragung vorbehalten.

## 1.2 Zu Ihrer Sicherheit

### Sicherheitsbestimmungen

Lesen Sie die folgenden Erklärungen zu Ihrer Sicherheit.  
Beachten und befolgen Sie stets produktspezifische Sicherheitshinweise, die Sie gegebenenfalls an den entsprechenden Stellen in diesem Dokument vorfinden.

### Haftungsausschluss

Die gesamten Komponenten werden je nach Anwendungsbestimmungen in bestimmten Hard- und Software-Konfigurationen ausgeliefert. Änderungen der Hard- oder Software-Konfiguration, die über die dokumentierten Möglichkeiten hinausgehen, sind unzulässig und bewirken den Haftungsausschluss der Beckhoff Automation GmbH & Co. KG.

### Qualifikation des Personals

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs-, Automatisierungs- und Antriebstechnik, das mit den geltenden Normen vertraut ist.

### Signalwörter

Im Folgenden werden die Signalwörter eingeordnet, die in der Dokumentation verwendet werden. Um Personen- und Sachschäden zu vermeiden, lesen und befolgen Sie die Sicherheits- und Warnhinweise.

### Warnungen vor Personenschäden

#### **GEFAHR**

Es besteht eine Gefährdung mit hohem Risikograd, die den Tod oder eine schwere Verletzung zur Folge hat.

#### **WARNUNG**

Es besteht eine Gefährdung mit mittlerem Risikograd, die den Tod oder eine schwere Verletzung zur Folge haben kann.

#### **VORSICHT**

Es besteht eine Gefährdung mit geringem Risikograd, die eine mittelschwere oder leichte Verletzung zur Folge haben kann.

### Warnung vor Umwelt- oder Sachschäden

#### **HINWEIS**

Es besteht eine mögliche Schädigung für Umwelt, Geräte oder Daten.

### Information zum Umgang mit dem Produkt



Diese Information beinhaltet z. B.:  
Handlungsempfehlungen, Hilfestellungen oder weiterführende Informationen zum Produkt.

## 1.3 Hinweise zur Informationssicherheit

Die Produkte der Beckhoff Automation GmbH & Co. KG (Beckhoff) sind, sofern sie online zu erreichen sind, mit Security-Funktionen ausgestattet, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen. Trotz der Security-Funktionen sind die Erstellung, Implementierung und ständige Aktualisierung eines ganzheitlichen Security-Konzepts für den Betrieb notwendig, um die jeweilige Anlage, das System, die Maschine und die Netzwerke gegen Cyber-Bedrohungen zu schützen. Die von Beckhoff verkauften Produkte bilden dabei nur einen Teil des gesamtheitlichen Security-Konzepts. Der Kunde ist dafür verantwortlich, dass unbefugte Zugriffe durch Dritte auf seine Anlagen, Systeme, Maschinen und Netzwerke verhindert werden. Letztere sollten nur mit dem Unternehmensnetzwerk oder dem Internet verbunden werden, wenn entsprechende Schutzmaßnahmen eingerichtet wurden.

Zusätzlich sollten die Empfehlungen von Beckhoff zu entsprechenden Schutzmaßnahmen beachtet werden. Weiterführende Informationen über Informationssicherheit und Industrial Security finden Sie in unserem <https://www.beckhoff.de/secguide>.

Die Produkte und Lösungen von Beckhoff werden ständig weiterentwickelt. Dies betrifft auch die Security-Funktionen. Aufgrund der stetigen Weiterentwicklung empfiehlt Beckhoff ausdrücklich, die Produkte ständig auf dem aktuellen Stand zu halten und nach Bereitstellung von Updates diese auf die Produkte aufzuspielen. Die Verwendung veralteter oder nicht mehr unterstützter Produktversionen kann das Risiko von Cyber-Bedrohungen erhöhen.

Um stets über Hinweise zur Informationssicherheit zu Produkten von Beckhoff informiert zu sein, abonnieren Sie den RSS Feed unter <https://www.beckhoff.de/secinfo>.

## 2 Übersicht

### SMS Server:

Der TwinCAT SMS Server dient zum Verschicken von SMS Nachrichten mit TwinCAT über ein GSM Modem.

Das Produkt besteht aus zwei Hauptkomponenten:

1. [TwinCAT SPS Bibliotheken: SMS / SMS BC \[► 9\]](#) (ermöglichen das Versenden der SMS-Nachrichten direkt aus der SPS)
2. [TwinCAT ADS Gerät: SMS COM Server \[► 17\]](#) (ermöglicht das Versenden der SMS-Nachrichten über ADS (z.B. aus einer Visual Basic Applikation))

### SMTP Server:

Der TwinCAT SMTP Server dient zum Verschicken von E-Mail Nachrichten mit TwinCAT über ADS. Eine ausführliche Dokumentation finden Sie [hier \[► 35\]](#).



## 3 SMS Server

Der TwinCAT SMS Server dient zum Verschicken von SMS Nachrichten mit TwinCAT über ein GSM Modem.

Das Produkt besteht aus zwei Hauptkomponenten:

1. [TwinCAT SPS Bibliotheken: SMS / SMS BC \[► 9\]](#) (ermöglichen das Versenden der SMS-Nachrichten direkt aus der SPS)
2. [TwinCAT ADS Gerät: SMS COM Server \[► 17\]](#) (ermöglicht das Versenden der SMS-Nachrichten über ADS (z.B. aus einer Visual Basic Applikation))

### 3.1 SPS-Bibliotheken

Die TwinCAT SMS Bibliotheken enthalten einen Baustein zum Verschicken von SMS-Nachrichten direkt aus der SPS. Die SMS-Bibliothek basiert auf der 'Serial Communication'-Bibliothek. Damit ist es möglich, die serielle Schnittstelle des PCs und die serielle Klemme (KL6xxx) gleichartig anzusprechen. Nähere Informationen dazu befinden sich in der Dokumentation zur '[Serial Communication](#)' Bibliothek.

#### Produkteigenschaften:

- Versenden einer SMS über ein GSM-Modem direkt aus der TwinCAT SPS
- Das GSM-Modem kann über ein serielles Datenkabel an die serielle Schnittstelle des TwinCAT PC's oder an die seriellen Klemmen KL6xxx angeschlossen werden
- Versenden von SMS-Nachrichten bis zu 160 Zeichen
- Verfügbar für PC und BC (Bus Controller)

#### Systemvoraussetzungen:

- Installiertes TwinCAT System: Installationslevel: TwinCAT PLC oder höher
- TwinCAT SPS-Laufzeitsystem auf dem PC oder BC
- Ein geeignetes GSM-Modem mit Datenkabel

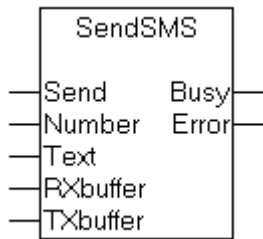
#### Unterstützten Geräte:

- Westermo GS-01 (Kommunikationsparameter: 9600 Baud, 8 Datenbits, kein Paritätsbit, ein Stopbit)
- Siemens S35i (Kommunikationsparameter: 19200 Baud, 8 Datenbits, kein Paritätsbit, ein Stopbit)
- Nokia 6210 (Kommunikationsparameter: 19200 Baud, 8 Datenbits, kein Paritätsbit, ein Stopbit)
- Maestro 100 (Kommunikationsparameter: 9600 Baud, 8 Datenbits, kein Paritätsbit, ein Stopbit)

Folgende SPS-Bibliotheken werden während der Installation in den `..\TwinCAT\PLC\LIB`-Ordner hineinkopiert:

- `TcPlcSMS.Lib` (SMS-Bibliothek für das PC-Laufzeitsystem)
- `TcPlcSMSBC.lb6` (SMS-Bibliothek für das BC-Laufzeitsystem)
- `COMlib.lib`, `COMlibBC5B.lb6`, `COMlibBCext.lb6`, `ChrAsc.Lib` und `ChrAsc.obj` (Serial Communication Bibliothek für PC und BC)
- `COMlibV2.lib`, `COMlibV2lb6` (Serial Communication Bibliothek für PC und BC v2.0 ). Nur TwinCAT SMS Server v2.0 und höher!

### 3.1.1 SendSMS



Mit dem Funktionsbaustein **SendSMS** wird eine SMS über ein angeschlossenes GSM Modem verschickt. Der Funktionsbaustein basiert auf der 'Serial Communcation' Bibliothek.

Da der Baustein nur über die **ComBuffer** Struktur der 'Serial Communcation' Bibliothek kommuniziert ist er instanzierbar und auf jede Art von serieller Schnittstelle anwendbar.

#### VAR\_INPUT

```
Send      : BOOL;
Number    : String;
Text      : String(160);
```

**Send:** Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

**Number:** Anzuwählende Telefonnummer im nationalen Format (z.B.: 0170123456)

**Text:** Die zu verschickende SMS Nachricht

#### VAR\_OUTPUT

```
Busy      : BOOL;
Error     : INT;
```

**Busy:** Bei der steigenden Flanke des Send Eingangs wird dieser Ausgang gesetzt und bleibt gesetzt, bis die SMS und das Modem übermittelt wurde oder ein Fehler aufgetreten ist.

**Error:** Sollte ein Fehler bei der Übertragung der SMS auftreten, wird der Busy Ausgang zurückgesetzt und am Error Ausgang liegt ein Fehlercode an.

Fehlercode	Bedeutung	Ursache
0	Kein Fehler	Die SMS wurde erfolgreich übertragen.
1	Keine Kommunikation zum Modem möglich.	Ist die Klemme korrekt konfiguriert? Wird die passende ComLib Bibliothek verwendet?
2	Modem meldet Fehler beim Konfigurieren.	Ist ein kompatibles GSM Modem angeschlossen?
3	Modem kann SMS nicht verschicken.	Ist die SIM-Karte in Ordnung? Kann die Karte ohne Eingabe der PIN verwendet werden? Hat das Modem eine Verbindung zum Netz? Ist ein kompatibles Modem angeschlossen?
4	Kommunikationsfehler.	Ist die richtige Übertragungsgeschwindigkeit eingestellt?

#### VAR\_IN\_OUT

```
RXbuffer  : ComBuffer;
TXbuffer  : ComBuffer;
```

**RXbuffer:** Struktur für die Kommunikation mit der seriellen Schnittstelle. Ein schnittstellenspezifischer Baustein der 'Serial Communcation' Bibliothek füllt diesen Buffer mit den Daten der Schnittstelle.

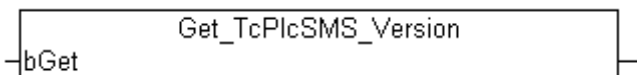
**TXbuffer:** Struktur für die Kommunikation mit der seriellen Schnittstelle. Ein schnittstellenspezifischer Baustein der 'Serial Communcation' Bibliothek überträgt die Daten dieses Buffers an die Schnittstelle.

Diese Strukturen und ihre Verwendung sind in der Dokumentation der 'Serial Communcation' Bibliothek näher beschrieben. Der SendSMS Baustein wird dabei wie ein SendString oder ReceiveString Baustein angeschlossen.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v2.7.0 und höher	PC (i386)	TcPlcSMS.Lib, ChrAsc.Lib, COMLib.Lib, Standard.Lib, PlcHelper.Lib
TwinCAT v2.8.0 und höher	PC (i386)	TcPlcSMS.Lib, ChrAsc.Lib, COMLib.Lib, TcSystem.Lib, ( Standard.Lib; TcBase.Lib; werden automatisch eingebunden )
TwinCAT v2.7.0 und höher	BCxxxx (165)	TcPlcSMSBC.Lb6, Standard.Lb6, PlcHelperBC.Lb6, ChrAsc.Lb6, COMLibBC5B.Lb6

**3.1.2 Get\_TcPlcSMS\_Version**



Mit dieser Funktion können Versionsinformationen der SPS-Bibliothek ausgelesen werden.

**FUNCTION Get\_TcPlcSMS\_Version: STRING(20)**

**VAR\_INPUT**

bGet : BOOL;

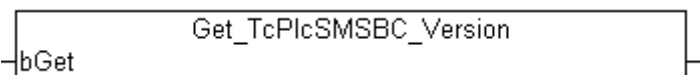
**bGet:** Dieser Parameter hat für die Funktionsausführung keine Bedeutung und kann beliebig gesetzt werden.

Der Rückgabeparameter ist ein String (z.B.: "2.000.000" ).

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v2.7.0 und höher	PC (i386)	TcPlcSMS.Lib, ChrAsc.Lib, COMLib.Lib, Standard.Lib, PlcHelper.Lib
TwinCAT v2.8.0 und höher	PC (i386)	TcPlcSMS.Lib, ChrAsc.Lib, COMLib.Lib, TcSystem.Lib, ( Standard.Lib; TcBase.Lib; werden automatisch eingebunden )

**3.1.3 Get\_TcPlsSMSBC\_Version**



Mit dieser Funktion können Versionsinformationen der SPS-Bibliothek ausgelesen werden.

**FUNCTION Get\_TcPlcSMSBC\_Version: STRING(20)****VAR\_INPUT**

bGet : BOOL;

**bGet:** Dieser Parameter hat für die Funktionsausführung keine Bedeutung und kann beliebig gesetzt werden.

Der Rückgabeparameter ist ein String (z.B.: "2.000.000" ).

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v2.7.0 und höher	BCxxxx (165)	TcPlcSMSBC.Lb6, Standard.Lb6, PlcHelperBC.Lb6, ChrAsc.Lb6, COMLibBC5B.Lb6

**3.1.4 Beispiele****3.1.4.1 Verschicken einer SMS mit Funktionsbaustein auf BC über KL6001**

Quelltext: <https://infosys.beckhoff.com/content/1031/tcsmssmtprv/Resources/11386387595.exe> (nicht vergessen die Telefonnummer zu ändern)

**Aufgabe**

Einfaches Programm, das mit der dem SMS-Funktionsbaustein auf einem BC9000 eine SMS über die serielle Klemme KL6001 verschickt.

**Beschreibung**

Die serielle Schnittstelle wird zunächst mit dem KL6Init Baustein initialisiert.

Danach kann mit der steigenden Flanke Send Eingangs das Verschicken initiiert werden.

Implementation

The screenshot displays the TwinCAT PLC Control interface for a project named 'TcPlcSMSBCTest.pr6 - [MAIN (PRG-FBD)]'. The main window shows a ladder logic program with the following structure:

```

0001 PROGRAM MAIN
0002 VAR
0003   SmsTestInst: SendSMS;
0004   TxBuff, RxBuff: ComBuffer;
0005   bSend: BOOL;
0006   bBusy: BOOL;
0007   Error: INT;
0008   Kl6Control: KL6Control5B;
0009   SerInit: KL6Init;
0010   blnit: BOOL;
0011 END_VAR
0012
0001 [Ladder Logic Step 1]
    SerInit (KL6Init)
    Inputs: blnit (Start), comIn.Status (SerStatus), comOut.Ctrl (SerCtrl)
    Outputs: Busy (to LAB_EXIT), Ready
0002 [Ladder Logic Step 2]
    SmsTestInst (SendSMS)
    Inputs: bSend (Send), '0175111111' (Number), 'TestTest' (Text), RxBuff (RXbuffer), TxBuff (TXbuffer)
    Outputs: Busy (to bBusy), Error
0003 [Ladder Logic Step 3]
    Kl6Control (KL6Control5B)
    Inputs: comIn (COMin), comOut (COMout), TxBuff (TxBuffer), RxBuff (RxBuffer)
0004 LAB_EXIT:
    TRUE (blnit)
    
```

The interface includes a menu bar (File, Edit, Project, Insert, Extras, Online, Window, Help), a toolbar with various icons, and a project tree on the left showing 'POUs' and 'MAIN (PRG)'. At the bottom, there are status indicators for 'ONLINE', 'OV', and 'READ'.

### **3.1.4.2 Verschicken einer SMS mit Funktionsbaustein auf PC über die serielle Schnittstelle des PCs**

Quelltext: <https://infosys.beckhoff.com/content/1031/tcsmssmtprv/Resources/11386389003.exe> (nicht vergessen die Telefonnummer zu ändern)

#### **Aufgabe**

Einfaches Programm, das mit dem SMS-Funktionsbaustein auf einem PC eine SMS über dessen serielle Schnittstelle verschickt.

#### **Beschreibung**

Wie in der Dokumentation zur 'Serial Communication' Bibliothek beschreiben, wird in einer schnellen Task die serielle Schnittstelle bedient, während das Verschicken der SMS Nachricht in einer langsameren Task erfolgt.

Mit der steigenden Flanke des Send Eingangs wird das Verschicken initiiert. An der Busy Variablen ist zu erkennen, wann die SMS-Nachricht an das Modem übertragen wurde.

#### **Konfiguration im System Manager**

Für die Ansteuerung der seriellen Schnittstelle muss diese im Systemmanager als E/A Gerät eingefügt werden. Die Schnittstelle muss für den KL6xx1 Betrieb mit 64 Datenbytes konfiguriert werden. Die Eingänge und Ausgänge des COM Port Geräts müssen dann mit den Variablen des SPS Programms (SerInData und SerOutData) verbunden werden.

Implementation des Programms in der langsamen Task zum Verschicken der SMS

The screenshot displays the TwinCAT PLC Control environment for the project 'TcPlcSM5PCTest.pro\*'. The main window shows the implementation of the 'SmsTestPg' program. On the left, the project tree lists 'POUs' containing 'FastSerPg (PRG)' and 'SmsTestPg (PRG)'. The central window shows the variable declaration for the program:

```

0001 PROGRAM SmsTestPg
0002 VAR
0003     SMS1 : SendSMS;
0004     Busy : BOOL;
0005     Err : INT;
0006     bSend : BOOL;
0007 END_VAR
0008
0009
0010

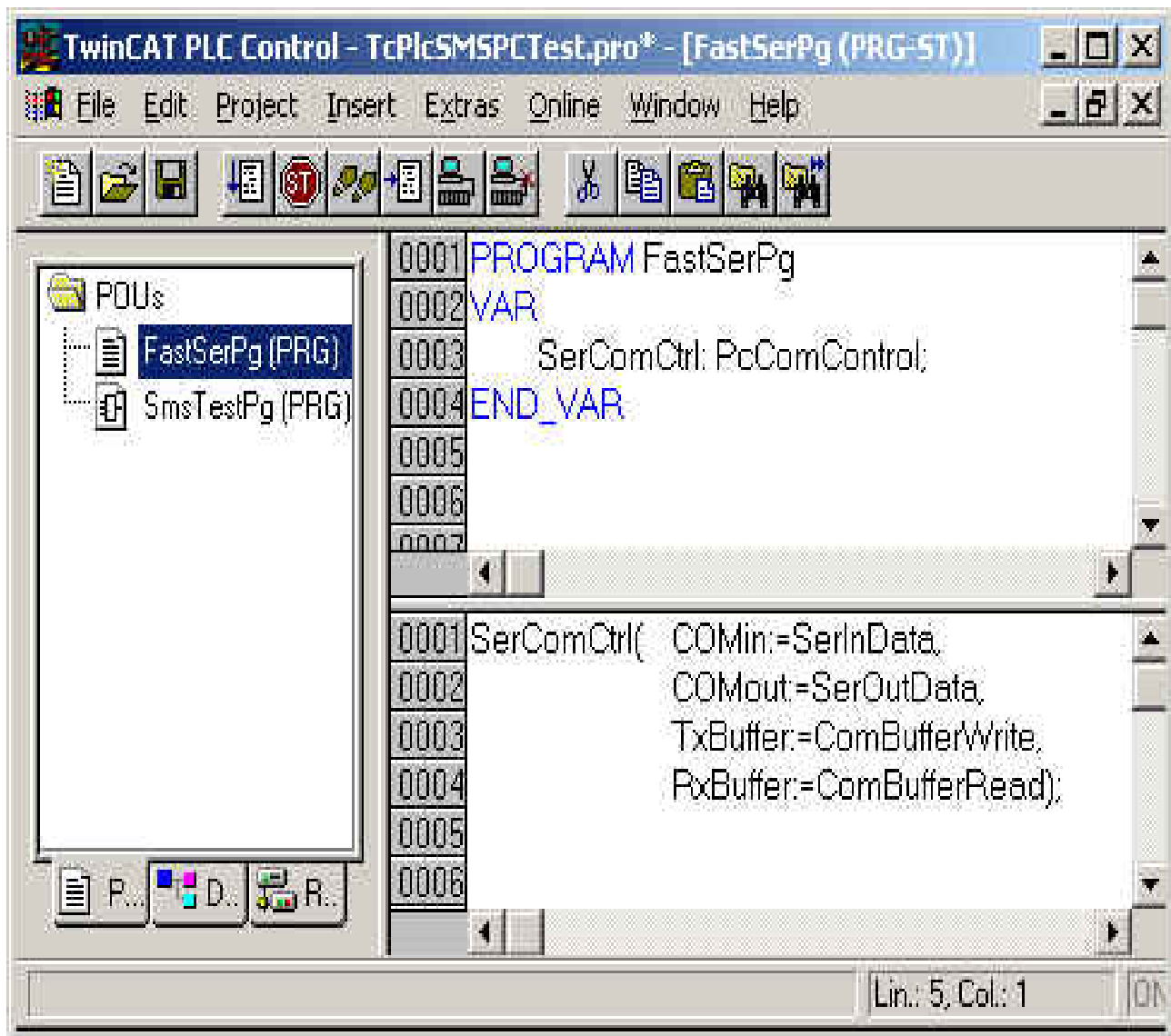
```

The right window shows a ladder logic network (0001) implementing the 'SendSMS' function block. The inputs and outputs are as follows:

- Inputs: bSend (Send), '01751111111' (Number), 'hello' (Text), ComBufferRead (RXbuffer), ComBufferWrite (TXbuffer).
- Outputs: Busy (Busy), Error (Err).

At the bottom right of the interface, there are status buttons for 'ONLINE', 'OV', and 'READ'.

## Implementation des Programms in der schnellen Task zum Bedienen der seriellen Schnittstelle

**Globale Variablen****VAR\_GLOBAL**

```

ComBufferRead : ComBuffer;
ComBufferWrite : ComBuffer;
SerInData AT %IB100 : PcComInData;
SerOutData AT %QB100 : PcComOutData;

```

**END\_VAR****Task Konfiguration**

- ☐ Task Configuration
  - ☐ Standard (PRIORITY := 0, INTERVAL := T#10ms)
    - └─ SmsTestPg
  - ☐ FastSerTask (PRIORITY := 1, INTERVAL := T#1ms)
    - └─ FastSerPg



## 3.2 ADS-Gerät

TwinCAT ADS Gerät: SMS COM Server ist ein Software Treiber, der SMS Nachrichten über ein GSM Modem verschicken kann. Der SMS COM Server kann, da es sich um einen reinen Software-Treiber handelt, als virtuelles Feldgerät (Automation Device) beschrieben werden. Er stellt daher für andere Kommunikationspartner (z.B. SPS oder Visual Basic Programme) eine Beckhoff ADS (Automation Device Specification) Schnittstelle zur Verfügung. Die Verwendung von ADS standardisiert den Zugriff auf das TwinCAT SMS-Gerät und reiht es in die Gruppe der verfügbaren virtuellen Feldgeräte ein.

### Produkteigenschaften:

- Implementation als NT COM Service.
- Ein SPS Programm ist nicht erforderlich.
- Implementation als Beckhoff ADS Gerät mit der ADS Port Nummer 10400.
- Wird zusammen mit TwinCAT gestartet und gestoppt.
- Nach der Installation muss die COM Port Verbindung mit dem [SMS COM Server Konfigurator \[► 17\]](#) konfiguriert werden.
- Das GSM Modem muss über ein serielles Datenkabel an die serielle Schnittstelle des TwinCAT PC's angeschlossen werden.
- Verschicken von SMS-Nachrichten bis zu 160 Zeichen.
- 50 SMS-Demoversion: Wenn bei der Installation kein gültiger Lizenzschlüssel eingegeben wurde, wird nur eine Test Version installiert. Diese hat den vollen Funktionsumfang, ist jedoch auf 50 Nachrichten begrenzt.

### Systemvoraussetzungen:

- Installiertes TwinCAT System, Installationslevel: TwinCAT CP oder höher.
- Ein geeignetes GSM Modem mit Datenkabel.

### Unterstützten Geräte:

- Westermo GS-01 (COM Parameter: 9600,N,8,1)
- Siemens S35i (COM Parameter: 19200,N,8,1)
- Nokia 6210 (COM Parameter: 19200,N,8,1)
- Maestro 100 (COM Parameter: 9600,N,8,1)

Folgende Dateien werden während der Installation in den ..\TwinCAT\SMS-Ordner hineinkopiert:

- TcSmsSrv.exe (TwinCAT ADS Gerät: SMS COM Server).
- TcSmsSrvCfg.exe (SMS COM Server Konfigurator. Programm zum [Konfigurieren des SMS COM Servers \[► 17\]](#)).
- SmsSrvTest.exe (Visual Basic Test-Applikation zum Verschicken der SMS-Nachrichten).
- TcSmsSrvSetup.txt (Informationen zur Installation und Konfiguration)

### 3.2.1 Konfiguration

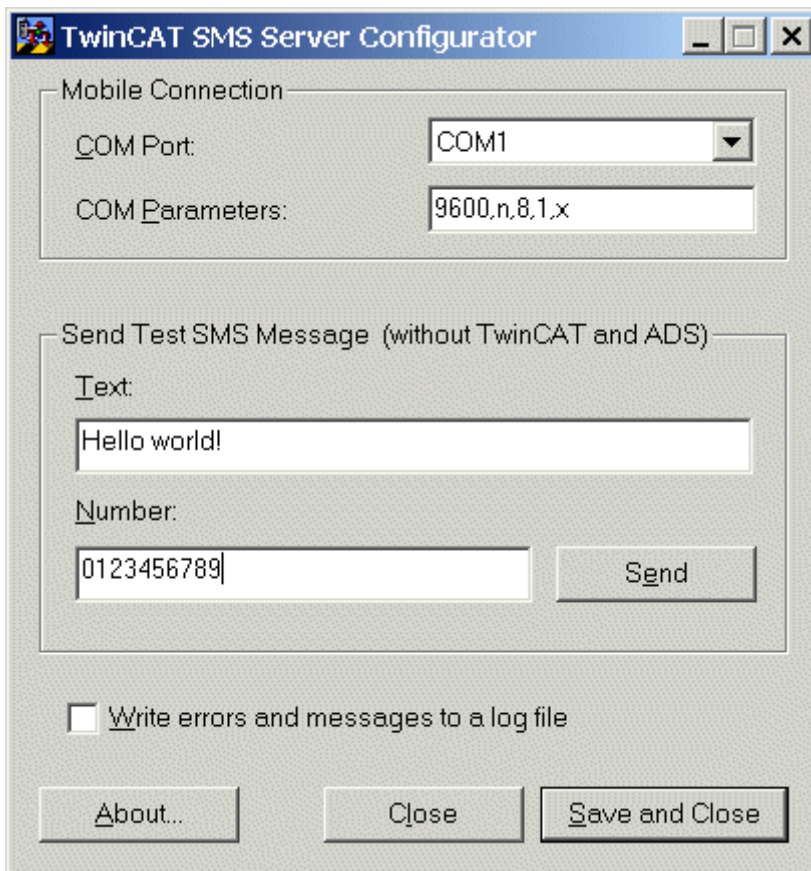
Für die Konfiguration des SMS COM Servers wird der TwinCAT SMS-Server Konfigurator verwendet. Mit dem Konfigurator kann die verwendete serielle Schnittstelle und die zu verwendenden Kommunikationsparameter eingestellt werden. Auch das Protokollieren in eine Log-Datei wird mit dem Konfigurator aktiviert.

Ein verbundenes GSM Modem ist für die Konfiguration nicht erforderlich. Sie können also die Konfiguration vornehmen, bevor Sie das Gerät in Betrieb nehmen. Die Konfigurationsdaten werden in der Datei Default.tps im ..\TwinCAT Ordner gespeichert. Die Konfiguration kann also mit dieser Datei gesichert oder auf einen Zielrechner kopiert werden.

1. Nach der Installation und vor der eigentlichen Konfiguration muss der SMS COM Server **einmalig** als TwinCAT Gerät registriert werden. Das TwinCAT System muss sich dafür im Stop-Zustand befinden (rotes Icon). In der Kommandozeile geben Sie folgendes ein und bestätigen mit Return:

### C:\TwinCAT\SMS\TcSmsSrv.exe /RegTcServer

2. Jetzt können Sie die eigentliche Konfiguration vornehmen. Zu beachten ist, dass für die Verwendung des Konfigurators das TwinCAT System ebenfalls gestoppt sein muss. Starten Sie die **TcSmsSrvCfg.exe**



#### Auswahl der seriellen Schnittstelle

In der 'COM Port' Auswahlbox muss die Schnittstelle eingestellt werden an der das GSM Modem angeschlossen ist.

#### Konfigurieren der Kommunikationsparameter

Die Kommunikationsparameter der seriellen Schnittstelle müssen entsprechend der Herstellerangaben des GSM Modems eingestellt werden.

In den meisten Fällen sollte die Standard Einstellung '19200,n,8,1' ausreichen ([Device configuration string syntax](#) [► 33]).

#### Einschalten der Log Funktion

Um genauere Fehler Informationen zu bekommen oder um ein Protokoll der verschickten Nachrichten zu erhalten, kann die Log-Funktion eingeschaltet werden.

Bei eingeschalteter Log- Funktion wird im ..\TwinCAT- oder im ..\Windows\System32-Verzeichnis eine Datei 'TcSmsSrvLog.xml' erstellt, in der alle verschickten Nachrichten und alle Fehler protokolliert werden.

#### Verschicken einer Test SMS

Um herauszufinden, ob alles richtig eingerichtet ist, sollte eine Test SMS verschickt werden. Eine einfache Möglichkeit ist die Verwendung des Visual Basic Beispiel Programms.

#### Sehen Sie dazu auch

- [Syntax der Geräte-Steuerzeichenkette](#) [► 33]

### 3.2.2 ADS-Interface

Der TwinCAT SMS-Server ist ein Software Treiber, der SMS Nachrichten über ein angeschlossenes GSM Modem verschicken kann. Er stellt daher für andere Kommunikationspartner (z.B. SPS - oder Visual Basic Programme) eine Beckhoff ADS (Automation Device Specification) Schnittstelle zur Verfügung. Die Verwendung von ADS standardisiert den Zugriff auf das TwinCAT SMS-Gerät und reiht es in die Gruppe der verfügbaren virtuellen Feldgeräte ein.

Die READ und WRITE Operationen auf der Schnittstelle erfolgen, wie durch ADS festgelegt, über zwei Zahlen: Der Index-Group und dem Index-Offset.

Auf den nächsten Seiten wird die ADS-Schnittstelle des SMS Servers hinsichtlich der Gruppen- und Offsetindizes genauer beschrieben.

#### Spezifikationen "Index-Group" der SPS

Die zwei globalen Bereiche eines ADS-Gerätes werden für den SMS Server in den Index-Groups wie folgt abgebildet:

Index-Group (0x = hex)	Index Group Beschreibung
0x00003000	<a href="#">Konfigurationsbereich [▶ 19]</a>
0x00004000	<a href="#">Bereich für SMS Dienste [▶ 19]</a>

#### 3.2.2.1 "Index-Group/Offset" Spezifikation für TwinCAT SMS Server Dienste

Dieser Abschnitt beschreibt ADS-Dienste zum Versenden von SMS Nachrichten mit dem TwinCAT SMS Server.

Index Group	Index Off-set	Zugriff	Datentyp	Phys. Einheit	Definitions- Bereich	Beschreibung	Anmerkung
0x00004000 0	0x00000000 1	W	UINT8[n]	Zeichenkette	Nullterminierte Zeichenkette	<a href="#">Die XML Zeichenkette beschreibt den Inhalt und das Ziel der SMS Nachricht. [▶ 20]</a>	

#### 3.2.2.2 "Index-Group/Offset" Spezifikation für TwinCAT SMS Server Konfiguration

Dieser Abschnitt beschreibt ADS-Dienste zur Konfiguration des TwinCAT SMS Servers.

Index Group	Index Off-set	Zugriff	Datentyp	Phys. Einheit	Definitions- Bereich	Beschreibung	Anmerkung
0x00003000 0	0x00000000 1	W	UINT8	1	0/1	Log-Datei ein- / ausschalten (1 = Log-Datei wird geschrieben)	

### 3.2.2.3 Sende SMS

Dieser ADS Dienst verschickt eine SMS an ein oder mehrere Empfänger.

Verwenden Sie hierfür den ADS-Dienst "AdsWriteReq".

Name	Beschreibung
ADS Port	10400
IndexGroup	0x00004000
IndexOffset	0x00000001
cbLength	Länge der XML Zeichenkette inklusive der terminierenden Null.
pWriteData	XML Zeichenkette mit folgendem Aufbau: <b>&lt;Msg&gt;</b> <b>&lt;Targets&gt;&lt;Target&gt;&lt;No&gt;&lt;/No&gt;&lt;/Target&gt;&lt;/Targets&gt;</b> <b>&lt;Body&gt;&lt;![CDATA[]]&gt;&lt;/Body&gt;</b> <b>&lt;/Msg&gt;</b> <i>Ziel Nachricht</i>
<b>&lt;No&gt;&lt;/No&gt;</b> <i>Ziel</i>	Die Zeichenfolge <b>&lt;No&gt;&lt;/No&gt;</b> <i>Ziel</i> kann für das versenden an mehrere Empfänger wiederholt werden.
<i>Ziel</i>	<b>Ziel</b> ist die anzuwählende Nummer. Sie kann entweder national oder international angegeben werden. Nationale Nummern haben eine führende 0 wie z.B.: 0170111111. Internationale Nummern beginnen mit einem + Zeichen gefolgt vom der Länderkennung wie z.B.: +49170111111.
<i>Nachricht</i>	<i>Nachricht</i> ist der zu verschickende Text. Die Nachricht darf nicht mehr als 80 Zeichen haben!

#### Beispiel für die XML Zeichenkette:

```
<Msg>
  <Targets>
    <Target>
      <No>0170111111</No>
      <No>0170222222</No>
    </Target>
  </Targets>

  <Body>
    <![CDATA[Hallo, dies ist eine SMS]]>
  </Body>
</Msg>
```

### 3.2.2.4 ADS Return Codes

Error codes: [0x000... \[▶ 20\]](#), [0x500... \[▶ 20\]](#), [0x700... \[▶ 20\]](#), [0x1000... \[▶ 20\]](#), [0x274C... \[▶ 20\]](#)

#### Global Error Codes

Hex	Dec	Description	Possible Causes	Solution
0x0	0	no error		
0x1	1	Internal error		
0x2	2	No Rtime		
0x3	3	Allocation locked memory error		
0x4	4	Insert mailbox error	No ADS mailbox was available to process this message.	Reduce the number of ADS calls (e.g <a href="#">ADS-Sum commands</a> or <a href="#">Max Delay Parameter</a> )
0x5	5	Wrong receive HMSG		
0x6	6	target port not found	ADS Server not started	
0x7	7	target machine not found	Missing ADS routes	
0x8	8	Unknown command ID		
0x9	9	Bad task ID		

Hex	Dec	Description	Possible Causes	Solution
0xA	10	No IO		
0xB	11	Unknown ADS command		
0xC	12	Win 32 error		
0xD	13	Port not connected		
0xE	14	Invalid ADS length		
0xF	15	Invalid AMS Net ID		
0x10	16	Low Installation level		
0x11	17	No debug available		
0x12	18	Port disabled		
0x13	19	Port already connected		
0x14	20	ADS Sync Win32 error		
0x15	21	ADS Sync Timeout		
0x16	22	ADS Sync AMS error		
0x17	23	ADS Sync no index map		
0x18	24	Invalid ADS port		
0x19	25	No memory		
0x1A	26	TCP send error		
0x1B	27	Host unreachable		
0x1C	28	Invalid AMS fragment		

**Router Error Codes**

Hex	Dec	Description	Possible Causes	Solution
0x500	1280	ROUTERERR_NOLOCKEDMEMORY	No locked memory can be allocated	
0x501	1281	ROUTERERR_RESIZEMEMORY	The size of the router memory could not be changed	
0x502	1282	ROUTERERR_MAILBOXFULL	The mailbox has reached the maximum number of possible messages. The current sent message was rejected	Check the connection between the communication partners
0x503	1283	ROUTERERR_DEBUGBOXFULL	The mailbox has reached the maximum number of possible messages. The sent message will not be displayed in the debug monitor	Check the connection to the debug monitor
0x504	1284	ROUTERERR_UNKNOWNPORTTYPE	The port type is unknown	
0x505	1285	ROUTERERR_NOTINITIALIZED	Router is not initialised	
0x506	1286	ROUTERERR_PORTALREADYINUSE	The desired port number is already assigned	
0x507	1287	ROUTERERR_NOTREGISTERED	Port not registered	
0x508	1288	ROUTERERR_NOMOREQUEUES	The maximum number of Ports reached	
0x509	1289	ROUTERERR_INVALIDPORT	The port is invalid.	
0x50A	1290	ROUTERERR_NOTACTIVATED	TwinCAT Router not active	

Hex	Dec	Description	Possible Causes	Solution
0x50B	1291	ROUTERERR_FRAGMENTBOXFULL		
0x50C	1292	ROUTERERR_FRAGMENTTIMEOUT		
0x50D	1293	ROUTERERR_TOBEREMOVED		

### General ADS Error Codes

Hex	Dec	Description	Possible Causes	Solution
0x700	1792	error class <device error>		
0x701	1793	Service is not supported by server		
0x702	1794	invalid index group		
0x703	1795	invalid index offset		
0x704	1796	reading/writing not permitted		
0x705	1797	parameter size not correct		
0x706	1798	invalid parameter value(s)		
0x707	1799	device is not in a ready state		
0x708	1800	device is busy		
0x709	1801	invalid context (must be in Windows)		
0x70A	1802	out of memory		
0x70B	1803	invalid parameter value(s)		
0x70C	1804	not found (files, ...)		
0x70D	1805	syntax error in command or file		
0x70E	1806	objects do not match		
0x70F	1807	object already exists		
0x710	1808	symbol not found		
0x711	1809	symbol version invalid	Onlinechange	Release handle and get a new one
0x712	1810	server is in invalid state		
0x713	1811	AdsTransMode not supported		
0x714	1812	Notification handle is invalid	Onlinechange	Release handle and get a new one
0x715	1813	Notification client not registered		
0x716	1814	no more notification handles		
0x717	1815	size for watch too big		
0x718	1816	device not initialized		
0x719	1817	device has a timeout		
0x71A	1818	query interface failed		
0x71B	1819	wrong interface required		
0x71C	1820	class ID is invalid		
0x71D	1821	object ID is invalid		
0x71E	1822	request is pending		
0x71F	1823	request is aborted		
0x720	1824	signal warning		
0x721	1825	invalid array index		
0x722	1826	symbol not active	Onlinechange	Release handle and get a new one
0x723	1827	access denied		
0x724	1828	missing license		Activate license for TwinCAT 3 function

Hex	Dec	Description	Possible Causes	Solution
0x72c	1836	exception occurred during system start		Check each device transistions
0x740	1856	Error class <client error>		
0x741	1857	invalid parameter at service		
0x742	1858	polling list is empty		
0x743	1859	var connection already in use		
0x744	1860	invoke ID in use		
0x745	1861	timeout elapsed		Check ADS routes of sender and receiver and your <u>firewall setting</u>
0x746	1862	error in win32 subsystem		
0x747	1863	Invalid client timeout value		
0x748	1864	ads-port not opened		
0x750	1872	internal error in ads sync		
0x751	1873	hash table overflow		
0x752	1874	key not found in hash		
0x753	1875	no more symbols in cache		
0x754	1876	invalid response received		
0x755	1877	sync port is locked		

**RTime Error Codes**

Hex	Dec	Description	Possible Causes
0x1000	4096	RTERR_INTERNAL	Internal fatal error in the TwinCAT real-time system
0x1001	4097	RTERR_BADTIMERPERIODS	Timer value not vaild
0x1002	4098	RTERR_INVALIDTASKPTR	Task pointer has the invalid value ZERO
0x1003	4099	RTERR_INVALIDSTACKPTR	Task stack pointer has the invalid value ZERO
0x1004	4100	RTERR_PRIOEXISTS	The demand task priority is already assigned
0x1005	4101	RTERR_NOMORETCB	No more free TCB (Task Control Block) available. Maximum number of TCBs is 64
0x1006	4102	RTERR_NOMORESEMAS	No more free semaphores available. Maximum number of semaphores is 64
0x1007	4103	RTERR_NOMOREQUEUEJES	No more free queue available. Maximum number of queue is 64
0x1008	4104	TwinCAT reserved.	
0x1009	4105	TwinCAT reserved.	
0x100A	4106	TwinCAT reserved.	
0x100B	4107	TwinCAT reserved.	
0x100C	4108	TwinCAT reserved.	
0x100D	4109	RTERR_EXTIRQALREADYDEF	An external synchronisation interrupt is already applied
0x100E	4110	RTERR_EXTIRQNOTDEF	No external synchronisation interrupt applied
0x100F	4111	RTERR_EXTIRQINSTALLFAILED	The apply of the external synchronisation interrupt failed
0x1010	4112	RTERR_IRQNOTLESSOREQUAL	Call of a service function in the wrong context
0x1017	4119	RTERR_VMXNOTSUPPORTED	Intel VT-x extension is not supported.

Hex	Dec	Description	Possible Causes
0x1018	4120	RTERR_VMXDISABLED	Intel VT-x extension is not enabled in BIOS.
0x1019	4121	RTERR_VMXCONTROLSMISSING	Missing feature in Intel VT-x extension.
0x101A	4122	RTERR_VMXENABLEFAILS	Enabling Intel VT-x fails.

### TCP Winsock Error Codes

Hex	Dec	Description	Possible Causes	Solution
0x274c	10060	A socket operation was attempted to an unreachable host	Host unreachable	Check network connection via ping
0x274d	10061	A connection attempt failed because the connected party did not properly respond after a period of time, or established connection failed because connected host has failed to respond.	Host unreachable	Check network connection via ping
0x2751	10065	No connection could be made because the target machine actively refused it		
		Further Winsock error codes: Win32 Error Codes		

## 3.2.3 Beispiele

### 3.2.3.1 SMS verschicken mit Visual Basic

Quelltext: <https://infosys.beckhoff.com/content/1031/tcsmssmtpsrv/Resources/11386390411.exe>

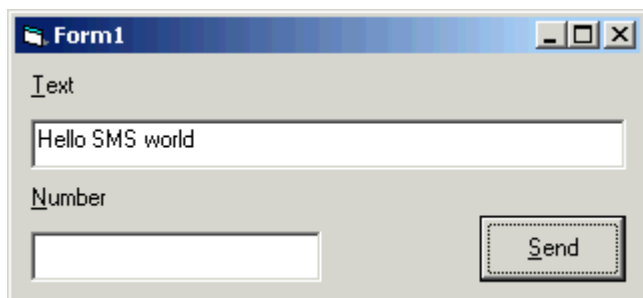
#### Aufgabe

Einfaches Visual Basic Programm zum Verschicken einer SMS.

#### Beschreibung

In diesem Beispiel wird mit dem ADS-OCX eine ADS Verbindung zum TwinCAT SMS-Server aufgebaut. In einem Formular kann der SMS-Text und die Telefonnummer angegeben werden. Die Nachricht wird dann auf Knopfdruck verschickt.

#### Formular



Die Eingabefelder haben die Namen ebText und ebNumber und der Knopf hat den Namen btSend.

Auf das Formular muss zusätzlich das ADS-OCX gezogen werden. Genauere Information dazu finden Sie in der Dokumentation des ADS-OCXes.



**Visual Basic Programm**

```
Option Explicit

Private Sub Form_Load()
    ' port of the TwinCAT SMS server
    AdsOCX1.AdsAmsServerPort = 10400
End Sub

Private Sub btSend_Click()
    On Error GoTo ERRORHANDLER
    Dim message As String

    ' create the XML message
    message = "<Msg><Targets><Target><No>" & _
        ebNumber & _
        "</No></Target></Targets><Body><![CDATA[" & _
        ebText & _
        "]]></Body></Msg>"

    AdsOCX1.AdsSyncWriteStringReq &H4000, 1, LenB(message), message

Exit Sub
ERRORHANDLER:
    MsgBox "Error " & Err.Number & ": " & Err.Description
End Sub
```

**3.2.3.2 SMS Funktionsbaustein in ST**

Quelltext: <https://infosys.beckhoff.com/content/1031/tcsmssmtpsrv/Resources/11386391819.exe>

**Aufgabe**

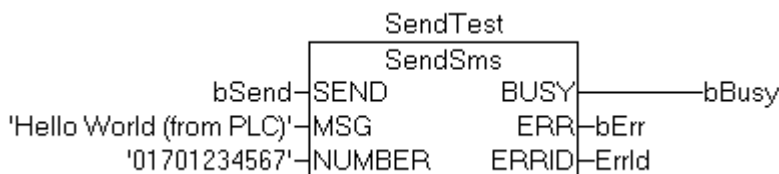
Einfacher Funktionsbaustein zum Verschicken einer SMS.

**Beschreibung**

Die Nachricht und die anzuwählende Telefonnummer soll über Eingänge anzugeben sein.

Die Steigende Flanke des SEND Eingangs soll das Verschicken initiieren.

**Aufruf des SMS Funktionsbausteins**



**Implementation**

Für den ADS Baustein muss die Bibliothek PlcSystem.lib eingebunden werden.

**Deklaration der Variablen**

```

FUNCTION_BLOCK SendSms
VAR_INPUT
    SEND: BOOL;
    MSG: STRING(80);
    NUMBER: STRING;
END_VAR
VAR_OUTPUT
    BUSY: BOOL;
    ERR: BOOL;
    ERRID: UDINT;
END_VAR
VAR
    AdsWr: ADSWRITE;
    XmlMsg: STRING(250);
END_VAR

```

**ST Kode des Funktionsbausteins**

```

XmlMsg := CONCAT('<Msg><Targets><Target><No>',
    CONCAT(NUMBER,
    CONCAT('</No></Target></Targets><Body><![CDATA[',
    CONCAT(MSG,
        ']]></Body></Msg>'))));

AdsWr.PORT := 10400;
AdsWr.NETID := ''; (* local Net ID*)
AdsWr.IDXGRP := 16#4000;
AdsWr.IDXOFFS := 1;
AdsWr.LEN := SIZEOF(XmlMsg);
AdsWr.SRCADDR := ADR(XmlMsg);
AdsWr.TMOU := T#5s;
AdsWr.WRITE := SEND;

AdsWr(); (* call the ADS function block *)

BUSY := AdsWr.BUSY;
ERR := AdsWr.ERR;
ERRID := AdsWr.ERRID;

```

### 3.3 Fehlersuche

Falls das Verschicken einer SMS mit dem SendSMS Funktionsbaustein oder über den SMS COM Server nicht funktioniert, kann das verschiedene Gründe haben:

- keine Verbindung zum GSM Modem
- Verwendung eines nicht unterstützten GSM Modems
- falsche Rufnummer
- falsches GSM-Netz
- PIN erforderlich (die SIM Karte darf nicht mit einer PIN geschützt sein)
- nicht initialisierte serielle Klemme (KL6Init aufrufen)
- Fehlerhafter Aufruf des ADS Dienstes

Um diese Fehler zu finden gibt es verschiedene Hilfsmittel:

**Log Datei verwenden**

Mit dem SMS COM Server Konfigurator können Sie das Protokollieren in eine Log-Datei aktivieren. Danach werden alle verschickten Nachrichten und Fehler in die Datei TcSmsSrvCfg.xml geschrieben.

**NT Event Log überprüfen**

Fehler bei der Versendung von Nachrichten werden zusätzlich immer im NT Event Log protokolliert. Das Event Log erreichen Sie über das TwinCAT Icon in der Taskleiste.

**ADS-Fehlermeldungen auswerten**

Falls der Aufruf einer ADS-Funktion fehlschlägt, wird der Fehler im Rückgabewert der Funktion codiert. Eine Liste dieser Fehlercodes finden Sie unter [ADS Return Codes](#).

**Konfiguration der Klemme**

Die serielle Klemme kann auf verschiedene Arten konfiguriert werden. Unterschiedlich konfigurierte Klemmen werden zum Teil auch unterschiedlich im Prozessabbild abgebildet (3 Byte / 5 Byte Klemme, Advanced / Standard). Dabei ist zu beachten, dass die 'Serial Communication' Bibliothek zu der Konfiguration der Klemme passen muss. Siehe auch Dokumentation zur KL6xxx und Dokumentation der ComLib:

Wichtig ist auch die Übertragungsgeschwindigkeit der Klemme auf das verwendete Modem abzustimmen.

**Verschicken einer Test SMS**

Um herauszufinden, ob der Fehler im ADS Aufruf oder in der Konfiguration des SMS COM Servers liegt, können Sie mit dem Visual Basic Beispiel Programm einfach eine Test SMS verschicken.

**Verschicken einer Test SMS mit einem Handy**

Um herauszufinden, ob die SIM-Karte korrekt konfiguriert ist, können Sie sie in ein handelsübliches Handy stecken und damit eine SMS verschicken. Dabei darf es nicht erforderlich sein, eine PIN-Nummer einzugeben.

**Netzauswahl beim Westermo GS-01**

Für die verschiedenen Netze in Europa und den USA gibt es verschiedene Varianten des GS-01. Die Lampe an der Vorderseite des Modems zeigt an, ob ein Netz verfügbar ist. Wenn die Lampe blinkt, hat das Modem Verbindung zu einem Netz. Wenn die Lampe dauernd leuchtet, sollte das Westermo Handbuch zur Fehlersuche verwendet werden.

**3.4 Serielles Kabel für KL6001**

Ein Kabel zwischen GSM Modem und der serielle Klemme KL6001 sollte folgende Anschlussbelegung haben. Bei Verwendung eines Adapterkabels kann die Belegung eventuell unterschiedlich sein. Beim Nokia Handy ist oft noch eine Brücke zwischen Pin 4 und Pin 9 (DSR und DTR) der SUB-D Buchse erforderlich.

Das hier beschriebene Kabel verwendet Hardware Handshake (RTS / CTS). Es wurde mit dem Westermo GS-01 und dem Siemens S53i getestet.

Anschluss	9 polige SUB-D Buchse	KL6001
RxD	2	5
TxD	3	1
GND	5	3
RTS	7	2
CTS	8	6

**3.5 7 Bit GSM default alphabet coding**

Der TwinCAT SMS Server codiert und versendet die SMS-Nachrichten nach dem 7 Bit Standard-Alphabet. Die Sonder- oder nicht druckbaren ASCII-Zeichen (0x00..0x31) werden aber nicht automatisch in die entsprechenden 7-bit Codes konvertiert. Um diese Zeichen versenden zu können muss der SMS-String vorher entsprechend der unterstehenden Tabelle formatiert werden.

**Beispiel:**

Eine SMS mit folgendem Text soll aus der SPS gesendet werden:

**'Total: 100.89€, SmsSrv@Beckhoff.com'**

Der SPS-String muss folgendes Format haben:

**'Total: 100.89\$1B\$65, SmsSrv\$80Beckhoff.com'**

7 bit default GSM alphabet as specified by GSM 03.38.		8 bit ANSI alphabet		
Character	Character name	Hex	Dec	TwinCAT PLC string constant
@	COMMERCIAL AT	0x00	0	\$80 or every other number >= 0x80H
£	POUND SIGN	0x01	1	\$01
\$	DOLLAR SIGN	0x02	2	\$02
¥	YEN SIGN	0x03	3	\$03
è	LATIN SMALL LETTER E WITH GRAVE	0x04	4	\$04
é	LATIN SMALL LETTER E WITH ACUTE	0x05	5	\$05
ù	LATIN SMALL LETTER U WITH GRAVE	0x06	6	\$06
ì	LATIN SMALL LETTER I WITH GRAVE	0x07	7	\$07
ò	LATIN SMALL LETTER O WITH GRAVE	0x08	8	\$08
Ç	LATIN CAPITAL LETTER C WITH CEDILLA	0x09	9	\$09
	LINE FEED	0x0A	10	\$0A or \$N
Ø	LATIN CAPITAL LETTER O WITH STROKE	0x0B	11	\$0B
ø	LATIN SMALL LETTER O WITH STROKE	0x0C	12	\$0C
	CARRIAGE RETURN	0x0D	13	\$0D or \$R
Å	LATIN CAPITAL LETTER A WITH RING ABOVE	0x0E	14	\$0E
å	LATIN SMALL LETTER A WITH RING ABOVE	0x0F	15	\$0F
Δ	GREEK CAPITAL LETTER DELTA	0x10	16	\$10
_	LOW LINE	0x11	17	\$11
Φ	GREEK CAPITAL LETTER PHI	0x12	18	\$12

7 bit default GSM alphabet as specified by GSM 03.38.		8 bit ANSI alphabet		
Character	Character name	Hex	Dec	TwinCAT PLC string constant
Γ	GREEK CAPITAL LETTER GAMMA	0x13	19	\$13
Λ	GREEK CAPITAL LETTER LAMBDA	0x14	20	\$14
Ω	GREEK CAPITAL LETTER OMEGA	0x15	21	\$15
Π	GREEK CAPITAL LETTER PI	0x16	22	\$16
Ψ	GREEK CAPITAL LETTER PSI	0x17	23	\$17
Σ	GREEK CAPITAL LETTER SIGMA	0x18	24	\$18
Θ	GREEK CAPITAL LETTER THETA	0x19	25	\$19
Ξ	GREEK CAPITAL LETTER XI	0x1A	26	\$1A
	ESCAPE TO EXTENSION TABLE	0x1B	27	\$1B
	FORM FEED	0x1B 0x0A	27 10	\$1B\$0A
^	CIRCUMFLEX ACCENT	0x1B 0x14	27 20	\$1B\$14
{	LEFT CURLY BRACKET	0x1B 0x28	27 40	\$1B\$28
}	RIGHT CURLY BRACKET	0x1B 0x29	27 41	\$1B\$29
\	REVERSE SOLIDUS (BACKSLASH)	0x1B 0x2F	27 47	\$1B\$2F
[	LEFT SQUARE BRACKET	0x1B 0x3C	27 60	\$1B\$3C
~	TILDE	0x1B 0x3D	27 61	\$1B\$3D
]	RIGHT SQUARE BRACKET	0x1B 0x3E	27 62	\$1B\$3E
	VERTICAL BAR	0x1B 0x40	27 64	\$1B\$40
€	EURO SIGN	0x1B 0x65	27 101	\$1B\$65
Æ	LATIN CAPITAL LETTER AE	0x1C	28	\$1C
æ	LATIN SMALL LETTER AE	0x1D	29	\$1D
ß	LATIN SMALL LETTER SHARP S (German)	0x1E	30	\$1E
É	LATIN CAPITAL LETTER E WITH ACUTE	0x1F	31	\$1F
	SPACE	0x20	32	\$20 or ' '
!	EXCLAMATION MARK	0x21	33	!
"	QUOTATION MARK	0x22	34	"

7 bit default GSM alphabet as specified by GSM 03.38.		8 bit ANSI alphabet		
Character	Character name	Hex	Dec	TwinCAT PLC string constant
#	NUMBER SIGN	0x23	35	#
¤	CURRENCY SIGN	0x24	36	\$24 or \$\$
%	PERCENT SIGN	0x25	37	%
&	AMPERSAND	0x26	38	&
'	APOSTROPHE	0x27	39	\$27 or \$'
(	LEFT PARENTHESIS	0x28	40	(
)	RIGHT PARENTHESIS	0x29	41	)
*	ASTERISK	0x2A	42	*
+	PLUS SIGN	0x2B	43	+
,	COMMA	0x2C	44	,
-	HYPHEN-MINUS	0x2D	45	-
.	FULL STOP	0x2E	46	.
/	SOLIDUS (SLASH)	0x2F	47	/
0	DIGIT ZERO	0x30	48	0
1	DIGIT ONE	0x31	49	1
2	DIGIT TWO	0x32	50	2
3	DIGIT THREE	0x33	51	3
4	DIGIT FOUR	0x34	52	4
5	DIGIT FIVE	0x35	53	5
6	DIGIT SIX	0x36	54	6
7	DIGIT SEVEN	0x37	55	7
8	DIGIT EIGHT	0x38	56	8
9	DIGIT NINE	0x39	57	9
:	COLON	0x3A	58	:
;	SEMICOLON	0x3B	59	;
<	LESS-THAN SIGN	0x3C	60	<
=	EQUALS SIGN	0x3D	61	=
>	GREATER-THAN SIGN	0x3E	62	>
?	QUESTION MARK	0x3F	63	?
¡	INVERTED EXCLAMATION MARK	0x40	64	\$40
A	LATIN CAPITAL LETTER A	0x41	65	A
B	LATIN CAPITAL LETTER B	0x42	66	B
C	LATIN CAPITAL LETTER C	0x43	67	C
D	LATIN CAPITAL LETTER D	0x44	68	D
E	LATIN CAPITAL LETTER E	0x45	69	E
F	LATIN CAPITAL LETTER F	0x46	70	F

7 bit default GSM alphabet as specified by GSM 03.38.		8 bit ANSI alphabet		
Character	Character name	Hex	Dec	TwinCAT PLC string constant
G	LATIN CAPITAL LETTER G	0x47	71	G
H	LATIN CAPITAL LETTER H	0x48	72	H
I	LATIN CAPITAL LETTER I	0x49	73	I
J	LATIN CAPITAL LETTER J	0x4A	74	J
K	LATIN CAPITAL LETTER K	0x4B	75	K
L	LATIN CAPITAL LETTER L	0x4C	76	L
M	LATIN CAPITAL LETTER M	0x4D	77	M
N	LATIN CAPITAL LETTER N	0x4E	78	N
O	LATIN CAPITAL LETTER O	0x4F	79	O
P	LATIN CAPITAL LETTER P	0x50	80	P
Q	LATIN CAPITAL LETTER Q	0x51	81	Q
R	LATIN CAPITAL LETTER R	0x52	82	R
S	LATIN CAPITAL LETTER S	0x53	83	S
T	LATIN CAPITAL LETTER T	0x54	84	T
U	LATIN CAPITAL LETTER U	0x55	85	U
V	LATIN CAPITAL LETTER V	0x56	86	V
W	LATIN CAPITAL LETTER W	0x57	87	W
X	LATIN CAPITAL LETTER X	0x58	88	X
Y	LATIN CAPITAL LETTER Y	0x59	89	Y
Z	LATIN CAPITAL LETTER Z	0x5A	90	Z
Ä	LATIN CAPITAL LETTER A WITH DIAERESIS	0x5B	91	\$5B
Ö	LATIN CAPITAL LETTER O WITH DIAERESIS	0x5C	92	\$5C
Ñ	LATIN CAPITAL LETTER N WITH TILDE	0x5D	93	\$5D
Ü	LATIN CAPITAL LETTER U WITH DIAERESIS	0x5E	94	\$5E

7 bit default GSM alphabet as specified by GSM 03.38.		8 bit ANSI alphabet		
Character	Character name	Hex	Dec	TwinCAT PLC string constant
§	SECTION SIGN	0x5F	95	\$5F
¿	INVERTED QUESTION MARK	0x60	96	\$60
a	LATIN SMALL LETTER A	0x61	97	a
b	LATIN SMALL LETTER B	0x62	98	b
c	LATIN SMALL LETTER C	0x63	99	c
d	LATIN SMALL LETTER D	0x64	100	d
e	LATIN SMALL LETTER E	0x65	101	e
f	LATIN SMALL LETTER F	0x66	102	f
g	LATIN SMALL LETTER G	0x67	103	g
h	LATIN SMALL LETTER H	0x68	104	h
i	LATIN SMALL LETTER I	0x69	105	i
j	LATIN SMALL LETTER J	0x6A	106	j
k	LATIN SMALL LETTER K	0x6B	107	k
l	LATIN SMALL LETTER L	0x6C	108	l
m	LATIN SMALL LETTER M	0x6D	109	m
n	LATIN SMALL LETTER N	0x6E	110	n
o	LATIN SMALL LETTER O	0x6F	111	o
p	LATIN SMALL LETTER P	0x70	112	p
q	LATIN SMALL LETTER Q	0x71	113	q
r	LATIN SMALL LETTER R	0x72	114	r
s	LATIN SMALL LETTER S	0x73	115	s
t	LATIN SMALL LETTER T	0x74	116	t
u	LATIN SMALL LETTER U	0x75	117	u
v	LATIN SMALL LETTER V	0x76	118	v
w	LATIN SMALL LETTER W	0x77	119	w
x	LATIN SMALL LETTER X	0x78	120	x



7 bit default GSM alphabet as specified by GSM 03.38.		8 bit ANSI alphabet		
Character	Character name	Hex	Dec	TwinCAT PLC string constant
y	LATIN SMALL LETTER Y	0x79	121	y
z	LATIN SMALL LETTER Z	0x7A	122	z
ä	LATIN SMALL LETTER A WITH DIAERESIS	0x7B	123	\$7B
ö	LATIN SMALL LETTER O WITH DIAERESIS	0x7C	124	\$7C
ñ	LATIN SMALL LETTER N WITH TILDE	0x7D	125	\$7D
ü	LATIN SMALL LETTER U WITH DIAERESIS	0x7E	126	\$7E
à	LATIN SMALL LETTER A WITH GRAVE	0x7F	127	\$7F

### 3.6 Syntax der Geräte-Steuerzeichenkette

Die Geräte-Steuerzeichenkette verwendet die Syntax des Befehls **mode**. Die Zeichenkette muss die gleiche Form haben wie die Befehlszeilenargumente des Befehls **mode**. Weitere Informationen zur Syntax des Befehls **mode** finden Sie in der Endbenutzerdokumentation für Ihr Betriebssystem.

#### Syntax

```
modemcomm[:] [baud=b] [parity=p] [data=d] [stop=s] [to={on|off}] [xon={on|off}] [odsr={on|off}] [octs={on|off}] [dtr={on|off|hs}] [rts={on|off|hs|tg}] [idsr={on|off}]
```

#### Parameter

<b>com m[:]</b>	Gibt die Nummer des Ports für die asynchrone Kommunikation (COM) an.
<b>baud= b</b>	Gibt die Übertragungsrates in Bits pro Sekunde an. In der folgenden Tabelle sind die gültigen Abkürzungen für <i>b</i> und die zugehörigen Baudraten aufgeführt.
<b>parity= p</b>	Gibt an, wie das System das Paritätsbit zur Prüfung auf Übertragungsfehler verwendet. In der folgenden Tabelle sind die gültigen Werte aufgeführt: <i>p</i> . Der Standardwert ist <b>e</b> . Nicht alle Computer unterstützen die Werte <b>m</b> und <b>s</b> .
<b>data= d</b>	Gibt die Anzahl der Datenbits in einem Zeichen an. Gültige Werte für <b>d</b> liegen im Bereich von 5 bis 8. Der Standardwert ist 7. Nicht alle Computer unterstützen die Werte 5 und 6.
<b>stop= s</b>	Gibt die Anzahl der Stoppbits an, die das Ende eines Zeichens definieren: 1, 1,5 oder 2. Bei einer Baudrate von 110 ist der Standardwert 2, ansonsten ist der Standardwert 1. Nicht alle Computer unterstützen den Wert 1,5.
<b>to={on off}</b>	Gibt an, ob die Verarbeitung der endlosen Zeitüberschreitung ein- oder ausgeschaltet ist. Der Standardwert ist aus.
<b>xon={on off}</b>	Gibt an, ob das xon- oder xoff-Protokoll für die Datenflusssteuerung ein- oder ausgeschaltet ist.
<b>odsr={on off}</b>	Gibt an, ob das Ausgabehandshake, das das DSR (Data Set Ready)-Signal verwendet, ein- oder ausgeschaltet ist.

<b>octs={on off}</b>	Gibt an, ob das Ausgabehandshake, das das CTS (Clear To Send)-Signal verwendet, ein- oder ausgeschaltet ist.
<b>dtr={on off hs}</b>	Gibt an, ob das DTR (Data Terminal Ready)-Signal ein- oder ausgeschaltet oder auf Handshake eingestellt ist.
<b>rts={on off hs tg}</b>	Legt fest, ob das RTS (Request To Send)-Signal auf Ein, Aus, Handshake oder Toggle eingestellt ist.
<b>idsr={on off}</b>	Gibt an, ob das DSR-Signal verwendet wird oder nicht.

**Wissenswertes:**

- Für eine Zeichenkette wie **96,n,8,1** oder jede andere Zeichenkette in älterer Form **Modus**, die nicht mit einem **x** oder einem **p** endet:
  - fInX**, **fOutX**, **fOutXDsrFlow**, und **fOutXCtsFlow** sind alle auf FALSE gesetzt
  - fDtrControl** ist auf DTR\_CONTROL\_ENABLE gesetzt
  - fRtsControl** ist auf RTS\_CONTROL\_ENABLE gesetzt
- Für eine Zeichenkette wie **96,n,8,1,x** oder jede andere Zeichenkette der älteren Form **mode**, die mit einem **x** endet:
  - fInX** und **fOutX** sind beide auf TRUE gesetzt
  - fOutXDsrFlow** und **fOutXCtsFlow** sind beide auf FALSE gesetzt
  - fDtrControl** ist auf DTR\_CONTROL\_ENABLE gesetzt
  - fRtsControl** ist auf RTS\_CONTROL\_ENABLE gesetzt
- Für eine Zeichenkette wie **96,n,8,1,p** oder eine andere Zeichenkette der älteren Form **mode**, die mit einem **p** endet:
  - fInX** und **fOutX** sind beide auf FALSE gesetzt
  - fOutXDsrFlow** und **fOutXCtsFlow** sind beide auf TRUE gesetzt
  - fDtrControl** ist auf DTR\_CONTROL\_HANDSHAKE gesetzt
  - fRtsControl** ist auf RTS\_CONTROL\_HANDSHAKE gesetzt

## 4 SMTP Server

Mit dem TwinCAT SMTP Server können Emails direkt aus der SPS versendet werden. Dazu wird mit abgeschlossener Installation ein entsprechender Server zusammen mit TwinCAT gestartet. Der Server wird über ADS aus der SPS heraus angesprochen. Mehrere Bausteine stehen für den Versand von Emails in der SPS zur Verfügung:

- [FB\\_Smtp \[► 36\]](#)
- [FB\\_SmtpV2 \[► 42\]](#)
- [FB\\_SmtpAttach \[► 37\]](#)
- [FB\\_SmtpFull \[► 39\]](#)
- [FB\\_SmtpV3 \[► 37\]](#)
- [FB\\_SmtpV3\\_Full \[► 39\]](#)

### Verschlüsselung:

Der TC SMTP Server unterstützt die Verschlüsselungsmethoden via Secure Socket Layer (SSL) oder STARTTLS ab Version 1.0.14.

## 4.1 Konfiguration

Ab der Version 1.0.14 des TC Smtip Servers wird eine XML-basierende Konfiguration verwendet. Die **TcSmtipConfig.xml** befindet sich im Installationsverzeichnis des Supplements.

```
<TcSmtipConfig>
  <!-- EnableLogFile: 0 (Disabled), 1 (Enabled), 2 (Verbose) -->
  <EnableLogFile>0</EnableLogFile>
  <!-- LogSize: in Byte, 0 = use Default -->
  <LogSize>0</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>8000</Timeout>
  <!-- Charset for the message content -->
  <Charset>iso-8859-1</Charset>
</TcSmtipConfig>
```

### Wissenswertes zur XML Konfigurationdatei (ab Version 1.0.14)

**EnableLogFile:** Aktiviert das Logging. Bitte nur für Diagnosezwecke aktivieren.

**Authentication:** Auswahl der Authentifizierungsmethode.

**Port:** Option 0 verwendet die Standardports für den Mailversand.

**ContentEncoding:** Definiert das Entschlüsseln des Inhaltes.

**Timeout:** Timeout zum Mailversand in ms.

**Charset:** Definiert die Zeichenkodierung.

### Wissenswertes zur Registry Konfiguration (Version < 1.0.14)

#### Log-File aktivieren:

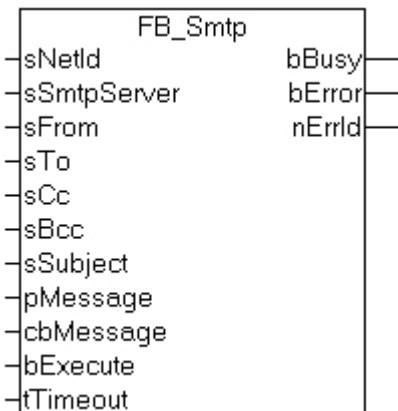
Der TwinCAT SMTP Server erzeugt eine Logdatei unter "\\TwinCAT\\SMS SMTP", wenn folgenden Eintrag in der Registry gesetzt wurde.

**Registry path:** HKEY\_LOCAL\_MACHINE\\SOFTWARE\\Beckhoff\\TwinCAT SMS / SMTP Server

**DWORD:** EnableLogFile = 1

## 4.2 Funktionsbausteine

### 4.2.1 FB\_Smtp



Der Baustein schickt per ADS einen Bytestream an ein ADS-Remotegerät. Auf dem ADS-Remotegerät muss der TwinCAT ADS Smtip Service laufen, um den Bytestream entgegen zu nehmen und zu einer Email zu verarbeiten. Nach der Bytesteam-Verarbeitung wird dann die E-Mail versendet.

Zu beachten ist, dass der SMTP-Server keine Passwort-Überprüfung unterstützt, da der TwinCAT ADS Smtip Service sich nicht per Passwort-Überprüfung am Server anmeldet.

#### VAR\_INPUT

```

VAR_INPUT
  sNetId      : T_AmsNetID;      (* AmsNetID *)
  sSmtpServer : T_MaxString;    (* Smtip-Server address (IP or Name) *)
  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;         (* Messagelenght to send *)
  bExecute    : BOOL;
  tTimeout    : TIME := T#20s;
END_VAR

```

**sNetId:** AmsNetID worauf der TwinCAT Mail Servers läuft.

**sSmtpServer:** Name oder IP des Smtip-Servers.

**sFrom:** Ein String, der die Emailadresse des Absenders enthält. Es muss ein Absender angegeben werden. Der String ist auf 255 Zeichen begrenzt.

**sTo:** Ein String, der die Emailadressen der Empfänger beinhaltet. Es können mehrere Adressen angegeben werden, diese müssen dann mit einem Semikolon getrennt werden. Es muss mindestens ein Empfänger angegeben werden. Der String ist auf 255 Zeichen begrenzt.

**sCc:** Ein String mit einer E-Mail-Adresse eines weiteren Empfängers (Cc=Carbon Copy). Es kann auch ein Leerstring angegeben werden. Eine Kopie der E-Mail wird an diesen Empfänger gesendet. Die E-Mail-Adresse dieses Empfängers wird bei anderen Empfängern **sichtbar**. Es dürfen auch mehrere Empfängeradressen getrennt durch Semikolons angegeben werden. Der String ist auf 255 Zeichen begrenzt.

**sBcc:** Ein String mit einer E-Mail-Adresse eines weiteren Empfängers (Bcc=Blind Carbon Copy). Es kann auch ein Leerstring angegeben werden. Eine Kopie der E-Mail wird an diese Empfänger gesendet. Die E-Mail-Adresse dieses Empfängers wird bei anderen Empfängern nicht sichtbar. Es dürfen auch mehrere Empfängeradressen getrennt durch Semikolons angegeben werden. Der String ist auf 255 Zeichen begrenzt.

**sSubject:** Ein String mit dem Betreff der E-Mail. Die E-Mail kann auch ohne Betreff versendet werden. Sie bekommt dann automatisch den Namen vom versendenden Computer in den Betreff eingetragen (z.B. "Mail send from: CX\_00762C"). Der String für den Betreff ist auf 255 Zeichen begrenzt.

**pMessage:** Die Adresse (Pointer) eines nullterminierten Strings mit dem E-Mail-Text. Die E-Mail kann auch ohne Bodytext versendet werden, sie bekommt dann automatisch einen kleinen Satz mit dem Sendedatum und Uhrzeit eingetragen (z.B. "Mail send at: Thu, 23 Mar 2006 02:31:44 -0800"). Die Adresse des Strings kann mit dem ADR-Operator ermittelt werden.

**cbMessage:** Die Länge des E-Mail-Textes. Die Länge kann durch den LEN-Operator ermittelt werden.

**bExecute:** Eine steigende Flanke an diesem Eingang aktiviert den Funktionsbaustein.

**tTimeout:** Maximale Zeit, die bei der Ausführung des Befehls nicht überschritten werden darf.

**VAR\_OUTPUT**

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

**bBusy:** Dieser Ausgang bleibt solange auf TRUE, bis der Baustein eine Befehlsanforderung ausführt, längstens aber für die Dauer der an dem tTimeOut-Eingang angelegten Zeit.

**bError:** Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlspezifische Fehlercode ist in iErrorId enthalten.

**nErrId:** Enthält den befehlspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle [▶ 511](#)).

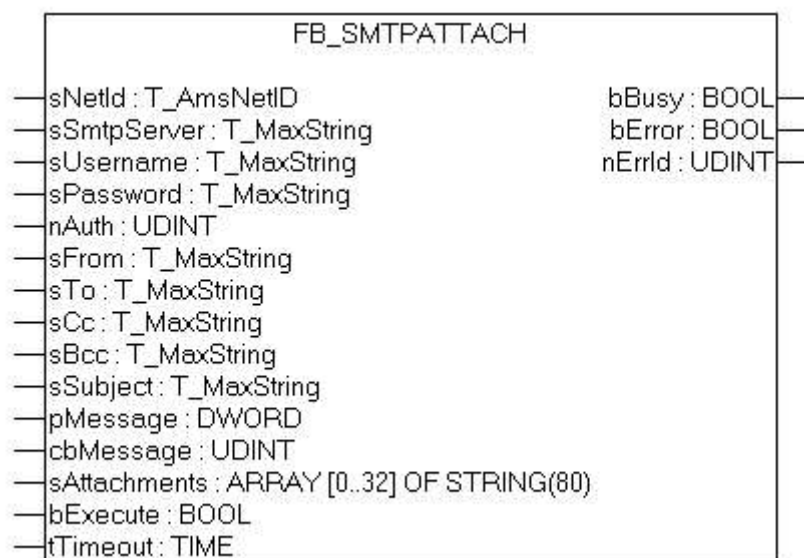


- Stellen sie sicher, dass Sie innerhalb des byte-Arrays keine \0 Stellen haben, da die Message dort abgebrochen wird.
- Die maximale Anzahl der Zeichen, die in einer Message verwendet werden dürfen, ist 510.725 - für From, To, Cc, Bcc und Subject stehen 1275 Zeichen zur Verfügung.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.8.0 und höher		TcSmtplib

**4.2.2 FB\_SmtpAttach**



Der Baustein sendet einen Datenstrom zu einem entfernten ADS Gerät via ADS. Der TwinCAT ADS SMTP Service muss auf dem entfernten ADS Gerät laufen, so dass der Datenstrom empfangen und in eine eMail verarbeitet werden kann. Sobald der Datenstrom verarbeitet wurde, wird die eMail versendet.

## VAR\_INPUT

```
VAR_INPUT
  sNetId      : T_AmsNetID;    (* AmsNetID *)
  sSmtServer  : T_MaxString;   (* Smt Server address ( IP or Name) *)
  sUsername   : T_MaxString;   (* Smt Username *)
  sPassword   : T_MaxString;   (* Smt Password *)
  nAuth       : UDINT;         (* Smt 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;        (* Messagelenght in byte to send *)
  sAttachments : ARRAY [0..32] OF STRING;
  bExecute    : BOOL;
  tTimeout    : TIME := T#20s;
END_VAR
```

**sNetId:** AmsNetID auf dem der TwinCAT SMS Server läuft.

**sSmtServer:** Name oder IP des SMTP Servers.

**sUsername:** Benutzername des SMTP Servers.

**sPassword:** Passwort für den SMTP Server.

**nAuth:** Smt Auth Type:

0 = AUTH NONE

1 = RESERVED

2 = AUTH LOGIN

3 = AUTH NTLM

4 = AUTH PLAIN

**sFrom:** Ein String, der die E-Mail-Adresse des Absenders enthält. Der Absender muss festgelegt werden. Der String ist auf 255 Zeichen limitiert.

**sTo:** Ein String, der die E-Mail-Adresse des Empfängers enthält. Mindestens ein Empfänger muss eingetragen werden. Es ist aber auch möglich mehrere Adressen einzutragen. Diese müssen per Simicolon getrennt werden. Der String ist auf 255 Zeichen limitiert.

**sCc:** Ein String, der die E-Mail-Adresse von weiteren Empfängern enthält (cc=carbon copy). Es ist möglich mehrere Adressen von Empfängern einzutragen, diese müssen dann durch ein Simicolon getrennt werden. Der String kann aber auch leer bleiben. Dem/den Empfänger(n) wird eine Kopie der eMail zugeschickt. Die E-Mail-Adresse des Empfängers ist für andere Empfänger **sichtbar**. Der String ist auf 255 Zeichen limitiert.

**sBcc:** Ein String der die E-Mail-Adressen von weiteren Empfängern enthält (Bcc = blind carbon copy). Es ist möglich mehrere Adressen von Empfängern einzutragen, diese müssen dann durch ein Simicolon getrennt werden. Der String kann aber auch leer bleiben. Dem/den Empfänger(n) wird eine Kopie der E-Mail zugeschickt. Die E-Mail-Adresse der Empfänger ist für andere Empfänger **nicht sichtbar**. Der String ist auf 255 Zeichen limitiert.

**sSubject:** Dieser String enthält den Betreff der E-Mail. Falls die E-Mail ohne Betreff gesendet wird, wird automatisch der Computername des Absenders in die Betreffzeile geschrieben (z.B. "eMail gesendet von: CX\_00762C"). Der String der Betreffzeile ist auf 255 Zeichen limitiert.

**pMessage:** Dieser Parameter gibt die Adresse des Strings, welcher den Nachrichtentext enthält, an. Falls die eMail ohne Text gesendet wird, wird automatisch das Datum und die Uhrzeit eingesetzt (z.B. "Mail send at Thu, 23 Mar 2006 02:31:44 -0800"). Die Adresse des Strings kann mit dem ADR Operator festgelegt werden.

**cbMessage:** Länge des E-Mail-Textes. Die Länge kann durch den LEN Operator festgelegt werden.

**bExecute:** Der Funktionsbaustein wird durch eine steigende Flanke an dieser Eingangsvariablen aktiviert.

**sAttachments:** Auflistung von Dateinamen

**tTimeout:** Die maximale, erlaubte Zeit, um einen Befehl auszuführen.

**VAR\_OUTPUT**

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

**bBusy:** Die Ausgangsvariable bleibt TRUE, bis der Block einen Befehl ausgeführt hat, allerdings nur bis zum Ablauf von tTimeOut.

**bError :** Die Ausgangsvariable wird auf TRUE umgeschaltet sobald ein Fehler bei der Ausführung des Befehls auftaucht. Der Befehls-spezifische Fehler ist in iErrorId enthalten.

**nErrId:** Beinhaltet den befehlspezifischen Fehler Code des zuletzt ausgeführten Befehls ([siehe Tabelle \[► 51\]](#)).

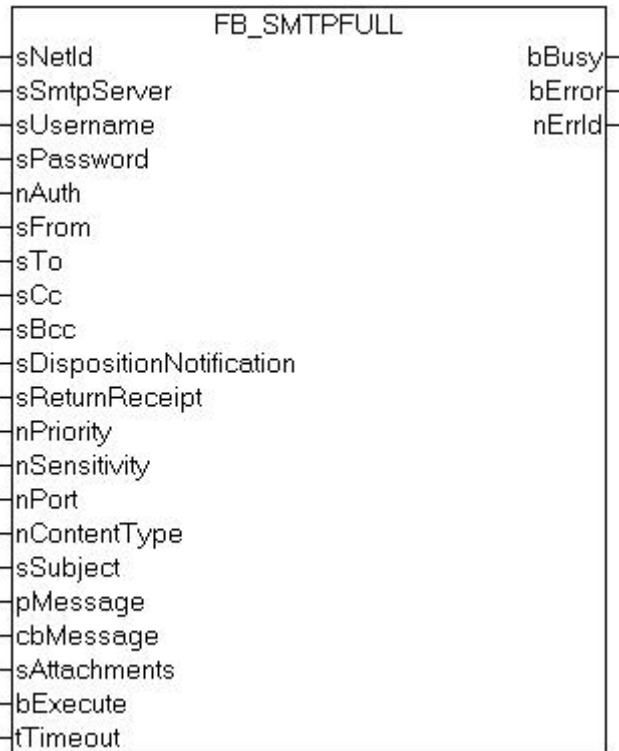


- Stellen Sie sicher, dass Sie \o nicht innerhalb der Byte-Anordnung verwenden, ansonsten wird die Nachricht gestoppt.
- Die maximale Anzahl von Zeichen in einer Nachricht beträgt 510.725 - insgesamt haben Sie 1275 Zeichen für From, To, Cc, Bcc and Subject.

**Voraussetzungen**

Entwicklungsumgebung	Zielsystem	Einzubindende SPS Bibliotheken
TwinCAT v2.8.0 und höher		TcSmtplib

**4.2.3 FB\_SmtpFull**



Dieser Funktionsbaustein kommuniziert über ADS mit dem TwinCAT SMTP Server. Er bietet sehr umfangreiche Mail-Funktionalitäten, wie zum Beispiel die Priorisierung von Emails aus der SPS heraus. Im Detail werden die einzelnen Parameter in dieser Dokumentation erläutert.

**VAR\_INPUT**

```

VAR_INPUT
  sNetId          : T_AmsNetID;          (* AmsNetID *)
  sSmtServer      : T_MaxString;        (* Smt Server address ( IP or Name) *)
  sUsername       : T_MaxString;        (* Smt Username *)
  sPassword       : T_MaxString;        (* Smt Password *)
  nAuth           : UDINT;              (* Smt Auth Type*)
  sFrom           : T_MaxString;        (* Sender stzring *)
  sTo             : T_MaxString;        (* To recipient string *)
  sCc             : T_MaxString;        (* Cc recipient string *)
  sBcc           : T_MaxString;        (* Bcc recipient string *)
  sDispositionNotification : T_MaxString; (* Disposition notification recipient string *)
  sReturnReceipt  : T_MaxString;        (* Return recipient string *)
  nPriority        : UDINT;              (* Priority value *)
  nSensitivity     : UDINT;              (* Sensitivity value *)
  nPort           : UDINT;              (* Communication port *)
  nContentType     : UDINT;              (* Content type *)
  sSubject        : T_MaxString;        (* Subject string *)
  pMessage        : DWORD;              (* Pointer to the message *)
  cbMessage       : UDINT;              (* Messagelenght in byte to send *)
  sAttachments    : ARRAY [0..32] OF STRING; (* Different attachments *)
  bExecute        : BOOL;               (* Trigger flag *)
  tTimeout        : TIME := T#20s;      (* Communication timeout *)
END_VAR

```

**sNetId:** AmsNetID, auf dem der TwinCAT SMTP Server läuft.

**sSmtServer:** Name oder IP des SMTP Servers

**sUsername:** Benutzername des SMTP Servers

**sPassword:** Passwort für den SMTP Server.

**nAuth:** Smt Auth Type:

0 = AUTH NONE

1 = RESERVED

2 = AUTH LOGIN

3 = AUTH NTLM

4 = AUTH PLAIN

**sFrom:** Ein String, der die E-Mail-Adresse des Absenders enthält. Der Absender muss festgelegt werden. Der String ist auf 255 Zeichen limitiert.

**sTo:** Ein String, der die E-Mail-Adresse des Empfängers enthält. Mindestens ein Empfänger muss eingetragen werden. Es ist aber auch möglich mehrere Adressen einzutragen. Diese müssen per Semicolon getrennt werden. Der String ist auf 255 Zeichen limitiert.

**sCc:** Ein String, der die E-Mail-Adresse von weiteren Empfängern enthält (cc=carbon copy). Es ist möglich mehrere Adressen von Empfängern einzutragen, diese müssen dann durch ein Semicolon getrennt werden. Der String kann aber auch leer bleiben. Dem/den Empfänger(n) wird eine Kopie der E-Mail zugeschickt. Die E-Mail-Adresse des Empfängers ist für andere Empfänger **sichtbar**. Der String ist auf 255 Zeichen limitiert.

**sBcc:** Ein String der die E-Mail-Adressen von weiteren Empfängern enthält (Bcc = blind carbon copy). Es ist möglich mehrere Adressen von Empfängern einzutragen, diese müssen dann durch ein Semicolon getrennt werden. Der String kann aber auch leer bleiben. Dem/den Empfänger(n) wird eine Kopie der E-Mail zugeschickt. Die E-Mail-Adresse der Empfänger ist für andere Empfänger **nicht sichtbar**. Der String ist auf 255 Zeichen limitiert.

**sDispositionNotification:** Die hier angegebene Mail-Adresse, erhält eine Lesebestätigung der Empfänger von sTo und sCc. Voraussetzung dafür ist jedoch, dass diese auch von den Empfängern gesendet wird.

**sReturnReceipt:** An die hier angegebene Mail-Adresse wird eine Übertragungsbestätigung der gesendeten Mail geschickt.

**nPriority:** Mit diesem Parameter können Sie die Priorität der Mail einstellen:

1 = Highest

2 = not used

3 = Normal

4 = not used

5 = Lowest



**nSensitivity:** Mit diesem Parameter können Sie die Vertraulichkeit der Nachricht einstellen:

- 0 = Private
- 1 = Personal
- 2 = Normal
- 3 = Confidential

**nPort:** Hier können Sie den Kommunikations-Port auswählen. Sollten Sie keinen eigenen Port eintragen, so wird auf den Default-Port 25 zurückgegriffen.

**nContentType:** Durch diesen Parameter ist es möglich beispielsweise HTML-Code, welcher per Pointer (pMessage) und Größe (cbMessage) einer Stringvariablen dem Baustein übergeben wird, in der eMail lesbar zu machen.

**sSubject:** Dieser String enthält den Betreff der eMail. Falls die eMail ohne Betreff gesendet wird, wird automatisch der Computername des Absenders in die Betreffzeile geschrieben (z.B. "eMail gesendet von: CX\_00762C"). Der String der Betreffzeile ist auf 255 Zeichen limitiert.

**pMessage:** Dieser Parameter gibt die Adresse des Strings, welcher den Nachrichtentext enthält, an. Falls die E-Mail ohne Text gesendet wird, wird automatisch das Datum und die Uhrzeit eingesetzt (z.B. "Mail send at Thu, 23 Mar 2006 02:31:44 -0800"). Die Adresse des Strings kann mit dem ADR Operator ermittelt werden.

**cbMessage:** Länge des E-Mail-Textes. Die Länge kann durch den LEN Operator festgelegt werden.

**bExecute:** Der Funktionsbaustein wird durch eine steigende Flanke an dieser Eingangsvariablen aktiviert.

**sAttachments:** Auflistung von Dateinamen

**tTimeout:** Die erlaubte, maximale Zeit, um einen Befehl auszuführen.

**VAR\_OUTPUT**

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

**bBusy:** Der Output bleibt TRUE, bis der Block einen Befehl ausgeführt hat, allerdings nur bis zum Ablauf von tTimeOut.

**bError :** Der Output wird auf TRUE umgeschaltet sobald ein Fehler bei der Ausführung des Befehls auftaucht. Der Befehls-spezifische Fehler ist in iErrorId enthalten.

**nErrId:** Beinhaltet den befehlspezifischen Fehler Code des zuletzt ausgeführten Befehls ([siehe Tabelle \[► 51\]](#)).

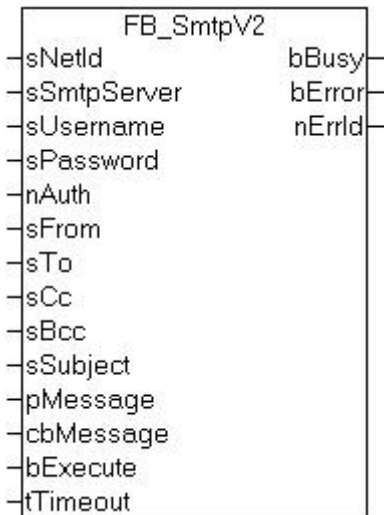


- Stellen Sie sicher, dass Sie \o nicht innerhalb der Byte-Anordnung verwenden, ansonsten wird die Nachricht gestoppt.
- Die maximale Anzahl von Zeichen in einer Nachricht beträgt 510.725 - insgesamt haben Sie 1275 Zeichen für From, To, Cc, Bcc and Subject.

**Voraussetzungen**

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.10.0 and above		TcSntp.lib

## 4.2.4 FB\_SmtpV2



Der Baustein sendet einen Datenstrom zu einem entfernten ADS Gerät via ADS. Der TwinCAT ADS SMTP Service muss auf dem entfernten ADS Gerät laufen, so dass der Datenstrom empfangen und in eine E-Mail verarbeitet werden kann. Sobald der Datenstrom verarbeitet wurde, wird die E-Mail versendet.

### VAR\_INPUT

```

VAR_INPUT
sNetId      : T_AmsNetID;      (* AmsNetID *)
sSmtpServer : T_MaxString;    (* Smtip Server address ( IP or Name) *)
sUsername   : T_MaxString;    (* Smtip Username *)
sPassword   : T_MaxString;    (* Smtip Password *)
nAuth       : UDINT;          (* Smtip 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 lenght in byte to send *)
bExecute    : BOOL;
tTimeout    : TIME := T#20s;
END_VAR

```

**sNetId:** AmsNetID auf dem der TwinCAT SMS-Server läuft.

**sSmtpServer:** Name oder IP des SMTP Servers.

**sUsername:** Benutzername des SMTP Servers.

**sPassword:** Passwort für den SMTP Server.

**nAuth:** Smtip Auth Type:

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

**sFrom:** Ein String, der die E-Mail-Adresse des Absenders enthält. Der Absender muss festgelegt werden. Der String ist auf 255 Zeichen limitiert.

**sTo:** Ein String, der die E-Mail-Adresse des Empfängers enthält. Mindestens ein Empfänger muss eingetragen werden. Es ist aber auch möglich mehrere Adressen einzutragen. Diese müssen per Simicolon getrennt werden. Der String ist auf 255 Zeichen limitiert.

**sCc:** Ein String, der die E-Mail-Adresse von weiteren Empfängern enthält (cc=carbon copy). Es ist möglich mehrere Adressen von Empfängern einzutragen, diese müssen dann durch ein Simicolon getrennt werden. Der String kann aber auch leer bleiben. Dem/den Empfänger(n) wird eine Kopie der eMail zugeschickt. Die E-Mail-Adresse des Empfängers ist für andere Empfänger **sichtbar**. Der String ist auf 255 Zeichen limitiert.

**sBcc:** Ein String der die E-Mail-Adressen von weiteren Empfängern enthält (Bcc = blind carbon copy). Es ist möglich mehrere Adressen von Empfängern einzutragen, diese müssen dann durch ein Simicolon getrennt werden. Der String kann aber auch leer bleiben. Dem/den Empfänger(n) wird eine Kopie der eMail zugeschickt. Die E-Mail-Adresse der Empfänger ist für andere Empfänger **nicht sichtbar**. Der String ist auf 255 Zeichen limitiert.

**sSubject:** Dieser String enthält den Betreff der E-Mail. Falls die E-Mail ohne Betreff gesendet wird, wird automatisch der Computername des Absenders in die Betreffzeile geschrieben (z.B. "eMail gesendet von: CX\_00762C"). Der String der Betreffzeile ist auf 255 Zeichen limitiert.

**pMessage:** Dieser Parameter gibt die Adresse des Strings, welcher den Nachrichtentext enthält, an. Falls die E-Mail ohne Text gesendet wird, wird automatisch das Datum und die Uhrzeit eingesetzt (z.B. "Mail send at Thu, 23 Mar 2006 02:31:44 -0800"). Die Adresse des Strings kann mit dem ADR Operator festgelegt werden.

**cbMessage:** Länge des E-Mail-Textes. Die Länge kann durch den LEN Operator festgelegt werden.

**bExecute:** Der Funktionsbaustein wird durch eine steigende Flanke an dieser Eingangsvariablen aktiviert.

**tTimeout:** Die maximale, erlaubte Zeit, um einen Befehl auszuführen.

**VAR\_OUTPUT**

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

**bBusy:** Die Ausgangsvariable bleibt TRUE, bis der Block einen Befehl ausgeführt hat, allerdings nur bis zum Ablauf von tTimeOut.

**bError :** Die Ausgangsvariable wird auf TRUE umgeschaltet sobald ein Fehler bei der Ausführung des Befehls auftaucht. Der Befehls-spezifische Fehler ist in iErrorId enthalten.

**nErrId:** Beinhaltet den befehlspezifischen Fehler Code des zuletzt ausgeführten Befehls ([siehe Tabelle \[p. 51\]](#)).



- Stellen Sie sicher, dass Sie \0 nicht innerhalb der Byte-Anordnung verwenden, ansonsten wird die Nachricht gestoppt.
- Die maximale Anzahl von Zeichen in einer Nachricht beträgt 510.725 - insgesamt haben Sie 1275 Zeichen für From, To, Cc, Bcc and Subject.

**Voraussetzungen**

Entwicklungsumgebung	Zielsystem	Einzubindende SPS Bibliotheken
TwinCAT v2.8.0 und höher		TcSmtplib

**Beispiel in ST:**

```
PROGRAM MAIN
VAR
  FB_SmtpV2 : FB_SmtpV2;
  bSend     : BOOL;
  bBusy     : BOOL;
  bError    : BOOL;
  nErrID    : UDINT;
  sServer   : STRING := 'smtpmail.mustermann.com';
  sMsg      : STRING := 'TcSmtplib is working properly';
  sSubject  : STRING := 'TcSmtplib Service Test';
  sUser     : STRING := 'username';
  sPassword : STRING := 'password';
  sFrom     : STRING := 'm.mustermann@mustermann.com';
END_VAR
```

```

    sTo      : STRING := 'm.muster@muster.com';
    nAuth    : INT := 2;
END_VAR

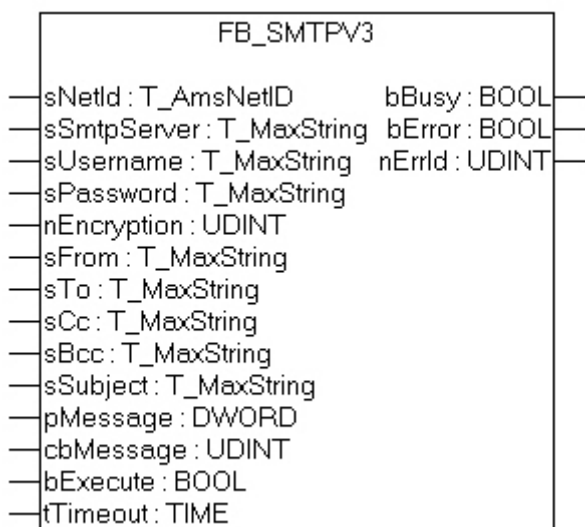
```

```

FB_SmtpV2 (      sNetId      := '',
                sSmtpServer := sServer,
                sUsername   := sUser,
                sPassword   := sPassword,
                nAuth       := nAuth,
                sFrom       := sFrom,
                sTo         := sTo,
                sSubject    := sSubject,
                pMessage    := ADR(sMsg) ,
                cbMessage   := LEN(sMsg)+1,
                bExecute    := bSend,
                tTimeout    := t#20s,
                bBusy       => bBusy,
                bError      => bError,
                nErrId     => nErrID);

```

## 4.2.5 FB\_SmtpV3



Der Block sendet einen Bytestream zu einem entfernten ADS Gerät. Der TwinCAT ADS Smtip dient muss auf dem Zielsystem laufen, damit der Stream entgegengenommen und als E-Mail verschickt wird.

### VAR\_INPUT

```

VAR_INPUT
    sNetId      : T_AmsNetID;    (* AmsNetID *)
    sSmtpServer : T_MaxString;   (* Smtip Server address ( IP or Name) *)
    sUsername   : T_MaxString;   (* Smtip Username *)
    sPassword   : T_MaxString;   (* Smtip 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;        (* Messagelenght in byte to send *)
    bExecute    : BOOL;
    tTimeout    : TIME := T#20s;
END_VAR

```

**sNetId:** AmsNetID on desTwinCAT Smtip server.

**sSmtpServer:** Name oder IP des Smtip server.

**sUsername:** Username für den Smtip Server.

**sPassword:** Password für den Smtip Server.

**nEncryption:** SmtP Verschlüsselungstyp:

0 = keine Verschlüsselung

1 = STARTTLS

2 = SSL

**sFrom:** Ein String, der die E-Mail-Adresse des Absenders enthält. Der Absender muss festgelegt werden. Der String ist auf 255 Zeichen limitiert.

**sTo:** Ein String, der die E-Mail-Adresse des Empfängers enthält. Mindestens ein Empfänger muss eingetragen werden. Es ist aber auch möglich mehrere Adressen einzutragen. Diese müssen per Semicolon getrennt werden. Der String ist auf 255 Zeichen limitiert.

**sCc:** Ein String, der die E-Mail-Adresse von weiteren Empfängern enthält (cc=carbon copy). Es ist möglich mehrere Adressen von Empfängern einzutragen, diese müssen dann durch ein Semicolon getrennt werden. Der String kann aber auch leer bleiben. Dem/den Empfänger(n) wird eine Kopie der eMail zugeschickt. Die E-Mail-Adresse des Empfängers ist für andere Empfänger **sichtbar**. Der String ist auf 255 Zeichen limitiert.

**sBcc:** Ein String der die E-Mail-Adressen von weiteren Empfängern enthält (Bcc = blind carbon copy). Es ist möglich mehrere Adressen von Empfängern einzutragen, diese müssen dann durch ein Semicolon getrennt werden. Der String kann aber auch leer bleiben. Dem/den Empfänger(n) wird eine Kopie der E-Mail zugeschickt. Die E-Mail-Adresse der Empfänger ist für andere Empfänger **nicht sichtbar**. Der String ist auf 255 Zeichen limitiert.

**sSubject:** Dieser String enthält den Betreff der E-Mail. Falls die E-Mail ohne Betreff gesendet wird, wird automatisch der Computernamen des Absenders in die Betreffzeile geschrieben (z.B. "eMail gesendet von: CX\_00762C"). Der String der Betreffzeile ist auf 255 Zeichen limitiert.

**pMessage:** Dieser Parameter gibt die Adresse des Strings, welcher den Nachrichtentext enthält, an. Falls die eMail ohne Text gesendet wird, wird automatisch das Datum und die Uhrzeit eingesetzt (z.B. "Mail send at Thu, 23 Mar 2006 02:31:44 -0800"). Die Adresse des Strings kann mit dem ADR Operator festgelegt werden.

**cbMessage:** Länge des E-Mail-Textes. Die Länge kann durch den LEN Operator festgelegt werden.

**bExecute:** Der Funktionsbaustein wird durch eine steigende Flanke an dieser Eingangsvariablen aktiviert.

**tTimeout:** Die maximale, erlaubte Zeit, um einen Befehl auszuführen.

**VAR\_OUTPUT**

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

**bBusy:** Die Ausgangsvariable bleibt TRUE, bis der Block einen Befehl ausgeführt hat, allerdings nur bis zum Ablauf von tTimeOut.

**bError :** Die Ausgangsvariable wird auf TRUE umgeschaltet sobald ein Fehler bei der Ausführung des Befehls auftaucht. Der Befehls-spezifische Fehler ist in nErrId enthalten.

**nErrId:** Beinhaltet den befehls-spezifischen Fehler Code des zuletzt ausgeführten Befehls ([siehe Tabelle \[► 51\]](#)).

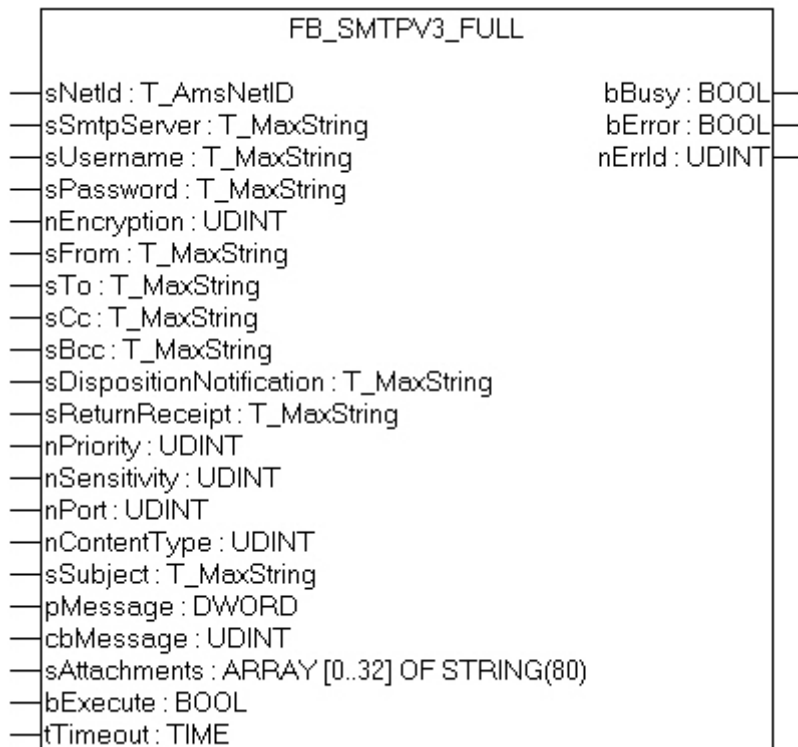


- Stellen Sie sicher, dass Sie \0 nicht innerhalb der Byte-Anordnung verwenden, ansonsten wird die Nachricht gestoppt.
- Die maximale Anzahl von Zeichen in einer Nachricht beträgt 510.725 - insgesamt haben Sie 1275 Zeichen für From, To, Cc, Bcc and Subject.

**Voraussetzungen**

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.8.0 und größer		TcSmtplib

## 4.2.6 FB\_SmtpV3\_Full



Dieser Funktionsbaustein kommuniziert über ADS mit dem TwinCAT SMTP Server. Er bietet sehr umfangreiche Mail-Funktionalitäten, wie zum Beispiel die Priorisierung von Emails aus der SPS heraus. Im Detail werden die einzelnen Parameter in dieser Dokumentation erläutert.

### VAR\_INPUT

```

VAR_INPUT
  sNetId      : T_AmsNetID;      (* AmsNetID *)
  sSmtpServer : T_MaxString;    (* Smtip Server address ( IP or Name) *)
  sUsername   : T_MaxString;    (* Smtip Username *)
  sPassword   : T_MaxString;    (* Smtip Password *)
  nEncryption : UDINT;         (* 0=NONE, 1=TLS, 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 *)
  sDispositionNotification: T_MaxString; (* Disposition notification recipient string *)
  sReturnReceipt : T_MaxString; (* Return recipient string *)
  nPriority    : UDINT;         (* Priority value *)
  nSensitivity : UDINT;         (* Sensitivity value *)
  nPort       : UDINT;         (* Communication port *)
  nContentType : UDINT;         (* Content type *)
  sSubject    : T_MaxString;    (* Subject string *)
  pMessage    : DWORD;         (* Pointer to the message *)
  cbMessage   : UDINT;         (* Messagelenght in byte to send *)
  sAttachments : ARRAY [0..32] OF STRING; (* Different attachments *)
  bExecute    : BOOL;         (* Trigger flag *)
  tTimeout    : TIME := T#20s; (* Communication timeout *)
END_VAR

```

**sNetId:** AmsNetID, auf dem der TwinCAT SMTP Server läuft.

**sSmtpServer:** Name oder IP des SMTP Servers

**sUsername:** Benutzername des SMTP Servers

**sPassword:** Passwort für den SMTP Server.

**nEncryption:** SmtP Verschlüsselungstyp:

0 = NONE

1 = STARTTLS

2 = SSL

**sFrom:** Ein String, der die E-Mail-Adresse des Absenders enthält. Der Absender muss festgelegt werden. Der String ist auf 255 Zeichen limitiert.

**sTo:** Ein String, der die E-Mail-Adresse des Empfängers enthält. Mindestens ein Empfänger muss eingetragen werden. Es ist aber auch möglich mehrere Adressen einzutragen. Diese müssen per Semicolon getrennt werden. Der String ist auf 255 Zeichen limitiert.

**sCc:** Ein String, der die E-Mail-Adresse von weiteren Empfängern enthält (cc=carbon copy). Es ist möglich mehrere Adressen von Empfängern einzutragen, diese müssen dann durch ein Semicolon getrennt werden. Der String kann aber auch leer bleiben. Dem/den Empfänger(n) wird eine Kopie der eMail zugeschickt. Die E-Mail-Adresse des Empfängers ist für andere Empfänger **sichtbar**. Der String ist auf 255 Zeichen limitiert.

**sBcc:** Ein String der die E-Mail-Adressen von weiteren Empfängern enthält (Bcc = blind carbon copy). Es ist möglich mehrere Adressen von Empfängern einzutragen, diese müssen dann durch ein Semicolon getrennt werden. Der String kann aber auch leer bleiben. Dem/den Empfänger(n) wird eine Kopie der eMail zugeschickt. Die E-Mail-Adresse der Empfänger ist für andere Empfänger **nicht sichtbar**. Der String ist auf 255 Zeichen limitiert.

**sDispositionNotification:** Die hier angegebene Mail-Adresse, erhält eine Lesebestätigung der Empfänger von sTo und sCc. Voraussetzung dafür ist jedoch, dass diese auch von den Empfängern gesendet wird.

**sReturnReceipt:** An die hier angegebene Mail-Adresse wird eine Übertragungsbestätigung der gesendeten Mail geschickt.

**nPriority:** Mit diesem Parameter können Sie die Priorität der Mail einstellen:

1 = Highest

2 = not used

3 = Normal

4 = not used

5 = Lowest

**nSensitivity:** Mit diesem Parameter können Sie die Vertraulichkeit der Nachricht einstellen:

0 = Private

1 = Personal

2 = Normal

3 = Confidential

**nPort:** Hier können Sie den Kommunikations-Port auswählen. Sollten Sie keinen eigenen Port eintragen, so wird auf den Default-Port 25 zurückgegriffen.

**nContentType:** Durch diesen Parameter ist es möglich beispielsweise HTML-Code, welcher per Pointer (pMessage) und Größe (cbMessage) einer Stringvariablen dem Baustein übergeben wird, in der eMail lesbar zu machen.

**sSubject:** Dieser String enthält den Betreff der eMail. Falls die eMail ohne Betreff gesendet wird, wird automatisch der Computername des Absenders in die Betreffzeile geschrieben (z.B. "eMail gesendet von: CX\_00762C"). Der String der Betreffzeile ist auf 255 Zeichen limitiert.

**pMessage:** Dieser Parameter gibt die Adresse des Strings, welcher den Nachrichtentext enthält, an. Falls die eMail ohne Text gesendet wird, wird automatisch das Datum und die Uhrzeit eingesetzt (z.B. "Mail send at Thu, 23 Mar 2006 02:31:44 -0800"). Die Adresse des Strings kann mit dem ADR Operator ermittelt werden.

**cbMessage:** Länge des eMail Textes. Die Länge kann durch den LEN Operator festgelegt werden.

**bExecute:** Der Funktionsbaustein wird durch eine steigende Flanke an dieser Eingangsvariablen aktiviert.

**sAttachments:** Auflistung von Dateinamen

**tTimeout:** Die erlaubte, maximale Zeit, um einen Befehl auszuführen.

## VAR\_OUTPUT

```

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

```

**bBusy:** Der Output bleibt TRUE, bis der Block einen Befehl ausgeführt hat, allerdings nur bis zum Ablauf von tTimeOut.

**bError :** Der Output wird auf TRUE umgeschaltet sobald ein Fehler bei der Ausführung des Befehls auftaucht. Der Befehls-spezifische Fehler ist in iErrorId enthalten.

**nErrId:** Beinhaltet den befehlspezifischen Fehler Code des zuletzt ausgeführten Befehls ([siehe Tabelle \[▶ 51\]](#)).



- Stellen Sie sicher, dass Sie \o nicht innerhalb der Byte-Anordnung verwenden, ansonsten wird die Nachricht gestoppt.
- Die maximale Anzahl von Zeichen in einer Nachricht beträgt 510.725 - insgesamt haben Sie 1275 Zeichen für From, To, Cc, Bcc and Subject.

### Voraussetzungen

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.10. and above		TcSntp.lib

## 4.3 Beispiele

### 4.3.1 How to - Best practice

In diesem Bereich werden häufige Anforderungen und deren Lösung aufgezeigt, um Ihnen die Arbeit mit TwinCAT SMTP zu erleichtern.

#### Versenden von Anhängen

Verwenden Sie den Baustein [FB SntpV3 Full \[▶ 46\]](#) und übergeben den Dateipfad am Eingang **sAttachments**.



Beachten Sie die unterschiedlichen Pfadangaben von Windows und Windows CE.

#### Windows:

```
sAttachments: ARRAY [0..32] OF STRING := 'C:\Data.csv', 'C:\Config.xml';
```

#### Windows CE:

```
sAttachments: ARRAY [0..32] OF STRING := '\\Hard Disk\Data.csv', '\\Hard Disk\Config.xml';
```

#### Mails verschlüsseln

Verwenden Sie den Baustein [FB SntpV3 \[▶ 44\]](#) oder [FB SntpV3 Full \[▶ 46\]](#) und aktivieren die Verschlüsselung über den Eingang **nEncryption**.

0 = NONE

1 = STARTTLS

2 = SSL



**Mehr als ein Empfänger angeben**

Verwenden Sie den Baustein [FB\\_SmtpV3 \[▶ 44\]](#) oder [FB\\_SmtpV3\\_Full \[▶ 46\]](#) und trennen mehrerer Empfänger (To, Cc, Bcc) mit einem Komma.

```
sTo:= 'service@customer.com, support@integrator.com';
```


**Mails mit dem CX9000 versenden**

Der CX9000 wird nur mit einem LF-Image (Low-Footprint) ausgeliefert. Dieses wird nur bis zur Version 1.0.13 unterstützt.

**Troubleshoot**

Bitte lesen Sie unseren Troubleshoot-Artikel.

**Sehen Sie dazu auch**

 [Fehlersuche \[▶ 51\]](#)

**4.3.2 Beispiel: Mailversand aus der SPS**

Anhand einer steigenden Flanke an bStart wird eine Mail versendet.

<https://infosys.beckhoff.com/content/1031/tcsmssmtprv/Resources/11386393227.zip>



Die Mailadressen und die Daten des SMTP Servers müssen vorher angepasst werden.

**Programmvariablen**

```
PROGRAM MAIN
VAR
    fbSendMail: FB_SmtpV3;
    sMessage: STRING := 'Hello Beckhoff';
    bStart: BOOL;
    bBusy: BOOL;
    bError: BOOL;
    nErrId: UDINT;
    nMails: UINT;
END_VAR
```

**Programm-code**

```
fbSendMail(sNetId:= '',
    sSmtpServer:= 'mail.company.com',
    sUsername:= '',
    sPassword:= '',
    nEncryption:= 0,
    sFrom:= 'machine@company.com',
    sTo:= 'service@customer.com',
    sSubject:= 'Mail sent via TwinCAT SMTP',
    pMessage:= ADR(sMessage),
    cbMessage:= SIZEOF(sMessage),
    bExecute:= bStart,
    bBusy=> bBusy,
    bError=> bError,
    nErrId=> nErrId);

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

**Voraussetzungen**

Development environment	Target system	PLC libraries to include
TwinCAT v2.10.0 or higher with (x86)	x86 or ARM	TcSmtp.Lib

Development environment	Target system	PLC libraries to include
		( Standard.Lib; TcBase.Lib; TcSystem.Lib will be included automatically )

### 4.3.3 Beispiel: Versenden von Mails in HTML

Mit dem FB\_SmtpV3Full stehen sehr umfangreiche E-Mail-Funktionalitäten für die SPS zur Verfügung. Es wird unter anderem der E-Mail-Text in HTML Code übergeben, was ganz neue Möglichkeiten der Formatierung bietet. So können sehr leicht aktuelle Messwert o.ä. in einer strukturierten Form übertragen werden.

Download: <https://infosys.beckhoff.com/content/1031/tcsmssmtpsrv/Resources/11386394635.zip>



Die Mailadressen und die Daten des SMTP Servers müssen vorher angepasst werden.

#### Programm-Variablen

```
PROGRAM MAIN
VAR
fbSmtpFull: FB_SmtpV3_Full;
sMessage_HTML: STRING := '<!DOCTYPE html><html><body><p>Sent by TwinCAT SMTP</p></body></html>';
bStart: BOOL;
bBusy: BOOL;
bError: BOOL;
nErrId: UDINT;
END_VAR
```

#### Programm-Code

```
fbSmtpFull(
sNetId:= '',
sSmtpServer:= 'mail.company.com',
sUsername:= '',
sPassword:= '',
nEncryption:= 0,
sFrom:= 'machine@company.com',
sTo:= 'service@customer.com',
nContentType:= 2,
sSubject:= 'Email from your Beckhoff PLC',
pMessage:= ADR(sMessage_HTML),
cbMessage:= SIZEOF(sMessage_HTML),
bExecute:= bStart,
bBusy=> bBusy,
bError=> bError,
nErrId=> nErrId);

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

#### Voraussetzungen

#### Voraussetzungen

Entwicklungsumgebung	Zielsystem	Benötigte SPS-Bibliotheken
TwinCAT v2.10.0 oder höher mit (x86)	x86 oder ARM	TcSmtp.Lib ( Standard.Lib; TcBase.Lib; TcSystem.Lib werden automatisch hinzugefügt)

## 4.4 Anhang

### 4.4.1 Fehlersuche

Die folgende Liste bietet grundlegende Hilfe bei Fehlern und sollte gelesen werden, bevor Sie sich an unsere Supportabteilung wenden.

1. Prüfen Sie, ob einer der SPS-Funktionsbausteine einen Fehlercode zurückgibt:

[SMTP-Fehlercodes](#)

2. Aktivieren Sie die Protokollierungsoption in der Datei `TcSmtpConfig.xml` [[▶ 17](#)] [[▶ 17](#)]

Aktivieren Sie die ausführliche Protokollierung durch Setzen von `EnableLogFile = 2` und starten Sie TwinCAT neu. Die Protokolldatei wird in `\TwinCAT\Functions\TF6350-SMS-SMTP` erzeugt.

3. SMTP-Verbindung über Telnet prüfen

Verwenden Sie einen Telnet-Client (z. B. PuTTY), versuchen Sie, eine Verbindung zum SMTP-Port (Standard 25) des Mailserver herzustellen, und senden Sie einen HELO-Befehl  
`220 mail.company.com Microsoft ESMTP MAIL Service readyHELO250 mail.company.com Hello [192.168.0.123]` Der Mailserver sollte mit Ihrer IP-Adresse antworten.

Wenn die Checkliste zur Fehlersuche nicht hilft, wenden Sie sich bitte an unsere Supportabteilung und geben Sie die folgenden Informationen an:

Allgemeine Systeminformationen

- Welche Art von Hardware wird auf dem Computer verwendet, auf dem TF6350 SMS/SMTP läuft?
- Beckhoff IPC oder Embedded-PC: Welche Produktnummer hat der PC?
- Welche Version des Betriebssystem-Images ist derzeit auf diesem Computer installiert?

Produktbezogene Systeminformationen

- Welche TF6350 SMS/SMTP-Version wird verwendet?
- Welche Funktionsbausteine der Bibliothek `Tc2_SMTP` werden im SPS-Programm verwendet?
- Welcher SMTP-Server wird verwendet?
- Bitte stellen Sie die SMTP-Protokolldatei zur Verfügung (siehe 2. der Fehlersuchliste)

**Bitte geben Sie eine genaue Beschreibung der Umgebung, in der das Produkt TF6350 SMS/SMTP eingesetzt wird**

- Wo befindet sich der Computer, auf dem TF6350 SMS/SMTP läuft?
- Wo befindet sich der SMTP-Server? (Lokales Netzwerk, Internet)
- Welche Verschlüsselung ist im Einsatz? (NON, STARTTLS, SSL)
- Wie lauten die IP-Einstellungen des Mail-Servers und des Computers, auf dem TF6350 SMS/SMTP läuft? (IP-Adresse, Subnetzmaske, Ports)

### 4.4.2 Fehlercodes

Diese Liste gibt mögliche Fehlercodes für das Supplementprodukt TwinCAT SMTP Server an. Sollten Sie einen Fehlercode bekommen, welchen Sie nicht in der Liste finden, dann schauen Sie bitte in den [ADS Return Codes](#) [[▶ 20](#)] oder in der Liste der [WinSockErrorCodes](#) [[▶ 53](#)] nach.

Fehlercode (hex)	Fehlercode (dec)	Beschreibung
< 0x8000	< 32778	ADS return code [ <a href="#">▶ 20</a> ]
0x800A	32778	Not connected
0x800B	32779	Sender expected
0x800C	32780	Recipients expected
0x800D	32781	Send FROM command failed

Fehlercode (hex)	Fehlercode (dec)	Beschreibung
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

In älteren Versionen des Servers (< 1.0.14) können außerdem folgende Fehler auftreten:

Fehlercode (hex)	Fehlercode (dec)	Beschreibung
0x000A	10	Not connected
0x000B	11	Sender expected
0x000C	12	Recipients expected
0x000D	13	Send FROM command failed
0x000E	14	Send DATA command failed
0x000F	15	Send mail header failed
0x0010	16	Send mail body failed
0x0011	17	Send "end of mail indicator" failed
0x0012	18	Send "RCPT" command failed
0x0064	100	General error
0x0065	101	Invalid parameter

Fehlercode (hex)	Fehlercode (dec)	Beschreibung
0x0066	102	Funtion not loaded
0x0067	103	Dll not loaded
0x0068	104	TcSmtpDll.dll cannot load. Check the installation from the TcSmtpDll.dll.
0x80D3	211	System status, or system help reply
0x80D6	214	Help message [Information on how to use the receiver or the meaning of a particular non-standard command; this reply is useful only to the human user]
0x80FB	251	User not local; will forward to <forward-path>
0x8163	354	Start mail input; end with <CRLF>.<CRLF>
0x81A5	421	<domain> Service not available, closing transmission channel [This may be a reply to any command if the service knows it must shut down]
0x81C2	450	Requested mail action not taken: mailbox unavailable [E.g., mailbox busy]
0x81C3	451	Requested action aborted: error in processing
0x81C4	452	Requested action not taken: insufficient system storage
0x81F4	500	Syntax error, command unrecognized [This may include errors such as command line too long]
0x81F5	501	Syntax error in parameters or arguments.
0x81F6	502	Command not implemented.
0x81F7	503	Bad sequence of commands.
0x8504	504	Command parameter not implemented
0x8226	550	Requested action not taken: mailbox unavailable [E.g., mailbox not found, no access]
0x8227	551	User not local; please try <forward-path>
0x8228	552	Requested mail action aborted: exceeded storage allocation
0x8229	553	Requested action not taken: mailbox name not allowed [E.g., mailbox syntax incorrect]
0x8224	554	Transaction failed

### 4.4.3 Windows Socket Fehler Codes

Die folgende Liste beschreibt die möglichen Fehler Codes, die durch die WSAGetLastError Funktion zurückkommen könnten. Die Fehler sind in alphabetischer Reihenfolge, nach Fehler-Makros sortiert, aufgelistet. Einige Fehler Codes, die in Winsock2.h definiert sind, sind nicht zurückgekommen - diese sind nicht in der Liste enthalten.

Rückgabewert	Beschreibung
WSAEINTR10004	Interrupted function call.blocking operation was interrupted by a call to WSACancelBlockingCall.
WSAEACCES 10013	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.

Rückgabewert	Beschreibung
WSAEFAULT 10014	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).
WSAEINVAL 10022	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.
WSAEMFILE 10024	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.
WSAEWOULDBLOCK 10035	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.
WSAEINPROGRESS 10036	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 WSAEINPROGRESS error.
WSAEALREADY 10037	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.
WSAENOTSOCK 10038	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.
WSAEDESTADDRREQ 10039	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.
WSAEMSGSIZE 10040	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.
WSAEPROTOTYPE 10041	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.
WSAENOPROTOOPT 10042	Bad protocol option. An unknown, invalid or unsupported option or level was specified in a getsockopt or setsockopt call.
WSAEPROTONOSUPPORT 10043	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.
WSAESOCKTNOSUPPORT 10044	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.
WSAEOPNOTSUPP 10045	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.

Rückgabewert	Beschreibung
WSAEPFNOSUPPORT 10046	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.
WSAEAFNOSUPPORT 10047	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.
WSAEADDRINUSE 10048	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.
WSAEADDRNOTAVAIL 10049	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).
WSAENETDOWN 10050	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.
WSAENETUNREACH 10051	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.
WSAENETRESET 10052	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.
WSAECONNABORTED 10053	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.
WSAECONNRESET 10054	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.
WSAENOBUFS 10055	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.
WSAEISCONN 10056	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.

Rückgabewert	Beschreibung
WSAENOTCONN 10057	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, setsockopt setting SO_KEEPALIVE if the connection has been reset.
WSAESHUTDOWN 10058	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.
WSAETIMEDOUT 10060	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.
WSAECONNREFUSED 10061	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.
WSAEHOSTDOWN 10064	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.
WSAEHOSTUNREACH 10065	No route to host. A socket operation was attempted to an unreachable host. See WSAENETUNREACH.
WSAEPROCLIM 10067	Too many processes. A Windows Sockets implementation may have a limit on the number of applications that can use it simultaneously. WSASStartup may fail with this error if the limit has been reached.
WSASYSNOTREADY 10091	Network subsystem is unavailable. This error is returned by WSASStartup 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: <ul style="list-style-type: none"> <li>• That the appropriate Windows Sockets DLL file is in the current path.</li> <li>• 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.</li> <li>• The Windows Sockets implementation documentation to be sure all necessary components are currently installed and configured correctly.</li> </ul>
WSAVERNOTSUPPORTED 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.
WSANOTINITIALISED 10093	Successful WSASStartup not yet performed. Either the application has not called WSASStartup or WSASStartup 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.
WSATYPE_NOT_FOUND 10109	Class type not found. The specified class was not found.
WSAHOST_NOT_FOUND 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.



Rückgabewert	Beschreibung
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.
WSANO_RECOVERY 11003	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.
WSANO_DATA 11004	Valid name, no data record of requested type. The requested name is valid and was found in the 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.
WSA_INVALID_HANDLE OS dependent	Specified event object handle is invalid. An application attempts to use an event object, but the specified handle is not valid.
WSA_INVALID_PARAMETER OS dependent	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.
WSA_IO_INCOMPLETE OS dependent	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.
WSA_IO_PENDING OS dependent	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.
WSA_NOT_ENOUGH_MEMORY OS dependent	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.
WSA_OPERATION_ABORTED OS dependent	Overlapped operation aborted. An overlapped operation was canceled due to the closure of the socket, or the execution of the SIO_FLUSH command in WSALocI.
WSA_INVALIDPROCEDURETABLE OS dependent	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.)
WSA_INVALIDPROVIDER OS dependent	Invalid service provider version number. A service provider returned a version number other than 2.0.
WSA_PROVIDER_FAILEDINIT OS dependent	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.
WSASYS_CALL_FAILURE OS dependent	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.



Mehr Informationen:  
**[www.beckhoff.de/ts6350](http://www.beckhoff.de/ts6350)**

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

