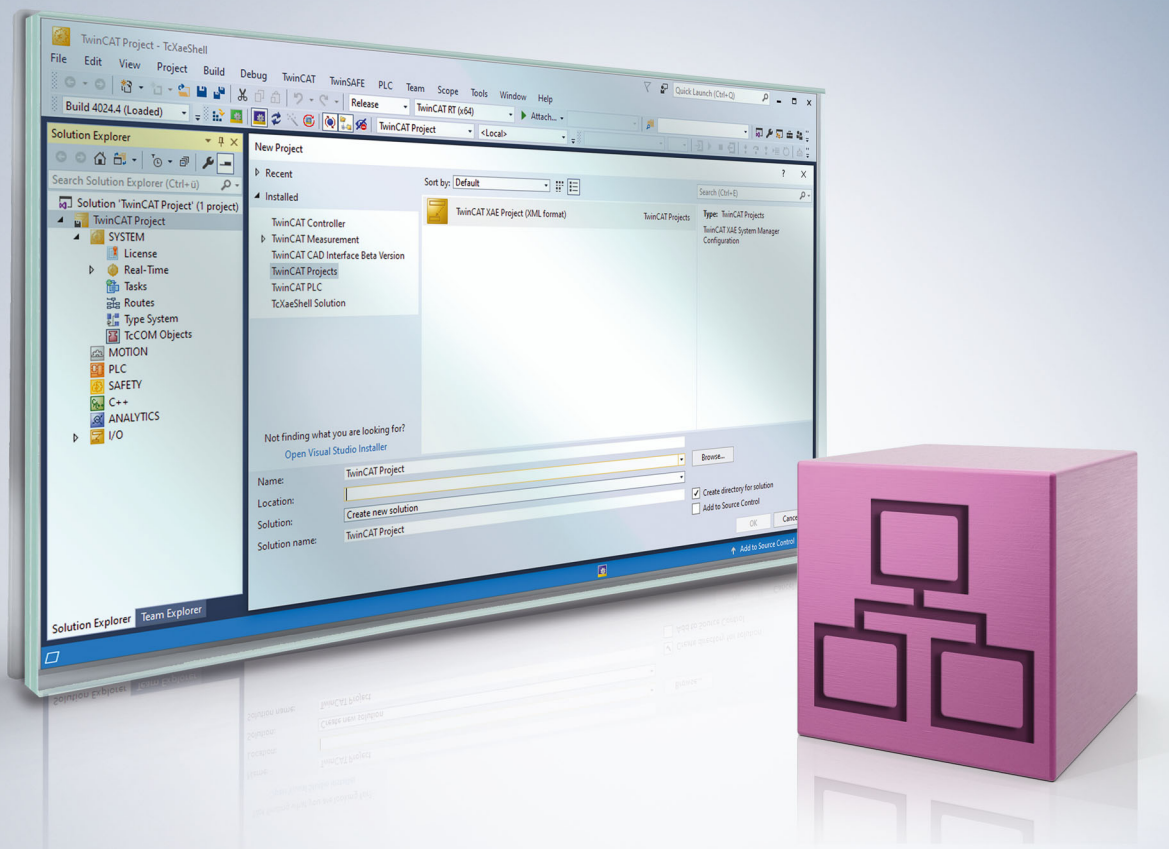
BECKHOFF New Automation Technology

Manual | EN

TF6300

TwinCAT 3 | FTP Client



2024-12-18 | Version: 2.0.0

Table of contents

1 Foreword

1.1 Notes on the documentation

This description is intended exclusively for trained specialists in control and automation technology who are familiar with the applicable national standards.
It is absolutely necessary to comply with the documentation and the following notes and explanations when installing and commissioning the components.
The trained specialists must always use the current valid documentation.

The trained specialists must ensure that the application and use of the products described is in line with all safety requirements, including all relevant laws, regulations, guidelines, and standards.

Disclaimer

The documentation has been compiled with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without notice.
Claims to modify products that have already been supplied may not be made on the basis of the data, diagrams, and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS®, and XPlanar® are registered and licensed trademarks of Beckhoff Automation GmbH.
If third parties make use of the designations or trademarks contained in this publication for their own purposes, this could infringe upon the rights of the owners of the said designations.

Patents

The EtherCAT Technology is covered by the following patent applications and patents, without this constituting an exhaustive list:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
and similar applications and registrations in several other countries.

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

Copyright

Third-party brands

Third-party trademarks and wordmarks are used in this documentation. The trademark endorsements can be found at: https://www.beckhoff.com/trademarks

1.2 For your safety

Safety regulations

Read the following explanations for your safety.
Always observe and follow product-specific safety instructions, which you may find at the appropriate places in this document.

Exclusion of liability

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

Personnel qualification

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

Signal words

The signal words used in the documentation are classified below. In order to prevent injury and damage to persons and property, read and follow the safety and warning notices.

Personal injury warnings

⚠ DANGER
Hazard with high risk of death or serious injury.

⚠ WARNING
Hazard with medium risk of death or serious injury.

⚠ CAUTION
There is a low-risk hazard that could result in medium or minor injury.

Warning of damage to property or environment

NOTICE
The environment, equipment, or data may be damaged.

Information on handling the product

This information includes, for example:
recommendations for action, assistance or further information on the product.

1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secguide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

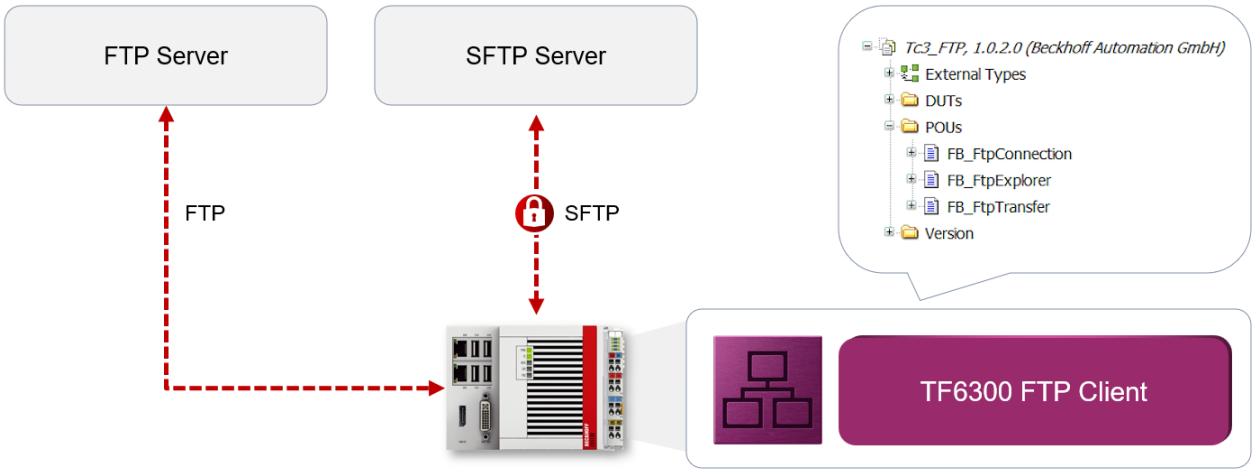
To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

1.4 Documentation issue status

Date	Change
2.0.x	Documentation completely revised, changes and additions in all chapters

2 Overview

The product TF6300 TC3 FTP Client enables the transfer of files between the PLC controller and a server based on the FTP and SFTP protocol. Various functions [▶ 19] are available to enable the efficient exchange of files. The use cases for uploading or downloading files can be diverse, ranging from the transfer of stored log files to the exchange of recipe data.



A PLC library supports the programmer in establishing the connection, as well as all other file or directory operations. Any server application that supports the FTP or SFTP protocol can be used as a remote terminal.

3 Installation

3.1 System Requirements

FTP client version 4.x or higher:

Technical data	Description
Operating system	Windows 10 TwinCAT/BSD
Target platforms	PC architecture (x86, x64)
.NET version	8.0
TwinCAT version	TwinCAT 3.1 Build 4026 or higher
TwinCAT installation level	TwinCAT 3 XAE, XAR
Required TwinCAT license	TF6300 TwinCAT FTP Client

FTP client version up to 3.x:

Technical data	Description
Operating system	Windows Embedded Standard 7 Windows 10
Target platforms	PC architecture (x86, x64)
.NET version	.NET Framework 2.0 SP1
TwinCAT version	TwinCAT 3.0 Build 3102 or higher
TwinCAT installation level	TwinCAT 3 XAE, XAR
Required TwinCAT license	TF6300 TwinCAT FTP Client

.NET 8

The product from version 4.x is based on .NET 8, but is delivered as a "self-contained executable" so that a separate installation of the .NET runtime is not required.

Tc2_FTP and Tc3_FTP

The PLC library Tc3_FTP, which provides support for SFTP, is only available for TwinCAT 3.1 Build 4026 and higher. The previous Tc2_FTP library can still be used under older TwinCAT 3 versions for compatibility reasons.

3.2 Installation

Depending on the TwinCAT version and operating system used, this TwinCAT 3 Function can be installed in different ways, which are described in more detail below.

NOTICE

Update installation

An update installation always uninstalls the previous installation. Please make sure that you have backed up your configuration files beforehand.

TwinCAT Package Manager

If you are using TwinCAT 3.1 Build 4026 (and higher) on the Microsoft Windows operating system, you can install this function via the TwinCAT Package Manager, see Installation documentation.

Normally you install the function via the corresponding workload; however, you can also install the packages contained in the workload individually. This documentation briefly describes the installation process via the workload.

Command line program TcPkg

You can use the TcPkg **C**ommand **L**ine **I**nterface (CLI) to display the available workloads on the system:

```
tcpkg list -t workload
```

You can install the workload of the function using the following command.

```
tcpkg install tf6300.ftpclient.xae
tcpkg install tf6300.ftpclient.xar
```

TwinCAT Package Manager UI

You can use the **U**ser **I**nterface (UI) to display all available workloads and install them if required.
To do this, follow the corresponding instructions in the interface.

NOTICE

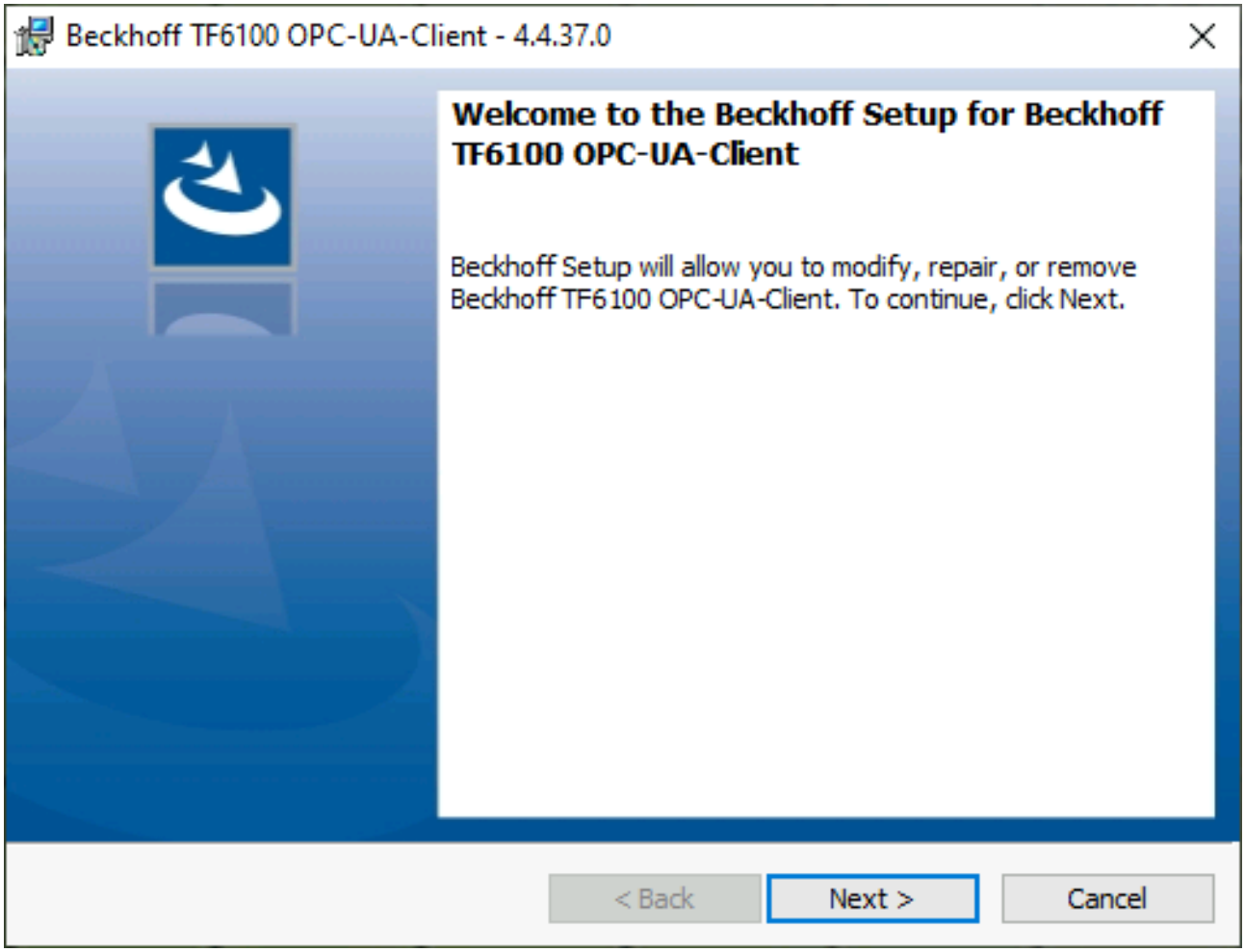
Unprepared TwinCAT restart can cause data loss

The installation of this function may result in a TwinCAT restart.
Make sure that no critical TwinCAT applications are running on the system or shut them down in an orderly manner first.

Setup

If you are using TwinCAT 3.1 Build 4024 on the Microsoft Windows operating system, you can install this function via a setup package, which you can download from the Beckhoff website at https://www.beckhoff.com/download.

Depending on the system on which you need the function, the installation can be done on either the engineering or runtime side. The following screenshot shows an example of the setup interface using the TF6100 TwinCAT OPC UA Client setup.



To complete the installation process, follow the instructions in the Setup dialog.

NOTICE

Unprepared TwinCAT restart can cause data loss

The installation of this function may result in a TwinCAT restart.
Make sure that no critical TwinCAT applications are running on the system or shut them down in an orderly manner first.

3.3 Licensing

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

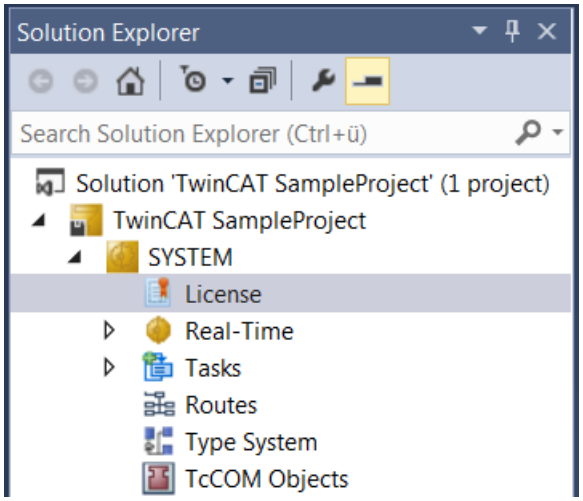
Licensing the full version of a TwinCAT 3 Function

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

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

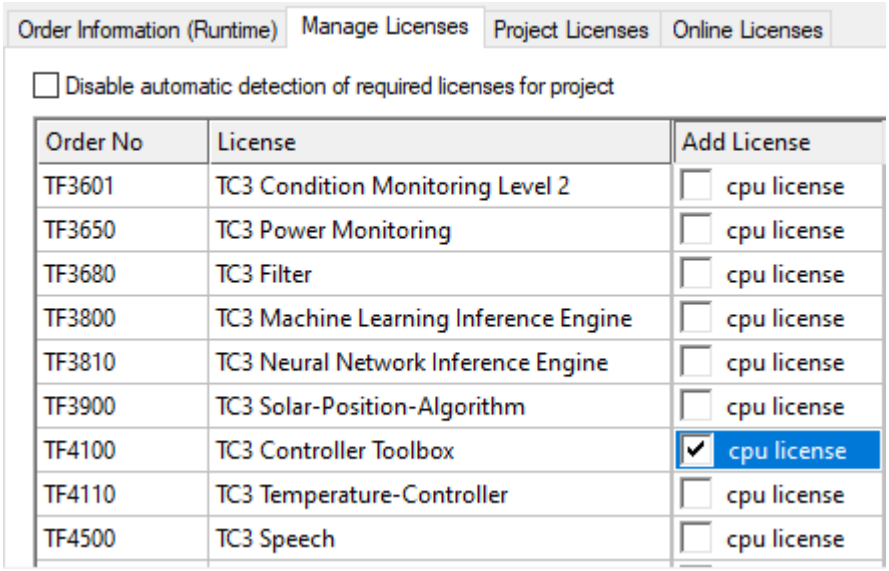
A 7-day test version cannot be enabled for a TwinCAT 3 license dongle.

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

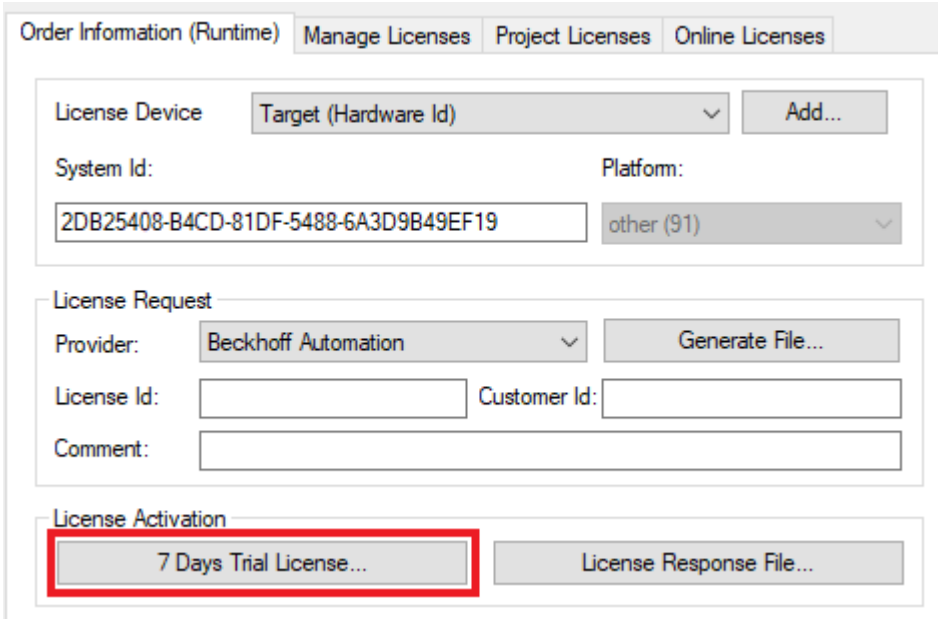


⇨ The TwinCAT 3 license manager opens.

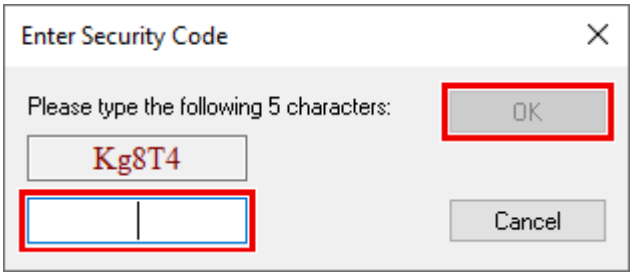
5. Open the **Manage Licenses** tab. In the **Add License** column, check the check box for the license you want to add to your project (e.g. "TF4100 TC3 Controller Toolbox").



6. Open the **Order Information (Runtime)** tab.
   ⇨ In the tabular overview of licenses, the previously selected license is displayed with the status “missing”**.**
7. Click **7-Day Trial License...** to activate the 7-day trial license.



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



8. Enter the code exactly as it is displayed and confirm the entry.
9. Confirm the subsequent dialog, which indicates the successful activation.
   ⇨ In the tabular overview of licenses, the license status now indicates the expiry date of the license.

10. Restart the TwinCAT system.

⇨ The 7-day trial version is enabled.

4 Technical introduction

4.1 Quick start

This documentation article only applies to the TwinCAT FTP Client from version 4.x.

This documentation article is intended to allow you an initial, quick start in how to use this product. You can also download the program code below at Samples [▶ 66]. The program code for a quick start with older product versions can also be found there. Following successful installation and licensing [▶ 11], perform the following steps in order to establish a connection to an FTP/SFTP server and configure variables for the read/write access.

Configuring an FTP/SFTP server

This QuickStart article requires a pre-configured and functioning FTP/SFTP server. In principle, you can use any FTP/SFTP server application for this, e.g. FileZilla, Microsoft IIS, OpenSSH server, Tiny SFTP server, and many more.

In the rest of this article we will use the following settings, please replace the parameters with those of your test setup if necessary:

Property	Description
Server type	SFTP server
Host name	The server is installed on the local system and can therefore be reached via the IP address 127.0.0.1.
Port	23
User name	My_SFTP_User
Password	SomeSecurePassword##1
Root directory	\
Authorizations	User has read/write access to the directory

Using the Tc3_Ftp library for the FileUpload

We now want to establish a connection to the SFTP server and upload a file to the root directory of the above-mentioned user. To do this, create a new TwinCAT project and a PLC project and reference the Tc3_FTP library there. Then go to the MAIN program and add the following lines of code.

Declaration part:

```
PROGRAM MAIN
VAR
    fbSimpleUpload : FB_SimpleUpload;
END_VAR
```

Implementation part:

```
fbSimpleUpload();
```

Then add a POU and name it FB_SimpleUpload. Add the following lines of code there.

Declaration part:

```
FUNCTION_BLOCK FB_SimpleUpload
VAR
    bStart              : BOOL; // Starts the QuickStart sample

    fbConnection        : FB_FtpConnection(1) := (sNetID:= '', tTimeout:= T#60S);
    fbExplorer          : FB_FtpExplorer := (sNetID:= '', tTimeout:= T#60S);
    fbTransfer          : FB_FtpTransfer := (sNetID:= '', tTimeout:= T#60S);

    stTransferInfo      : ST_FtpTransferState;
    stNTDataProfile     : ST_FtpConfigProfile;
```

```
    sPathToOriginFile  : STRING(255) := 'C:\Temp\myFileUpload.txt';
    sPathToTargetDir   : STRING(255) := '\';

    bDone              : BOOL;
    bError             : BOOL;
    sErrorMSG          : STRING(255);
    nCase              : INT;
    nTransferHandle    : UDINT;
    fbTimer            : TON := (PT:=T#1S);
END_VAR
```

Implementation part:

```
CASE nCase OF
    0: (* Start the QuickStart sample. *)
        IF bStart THEN
            bDone := FALSE;
            bStart := FALSE;
            nCase := 10;
        END_IF

    10: (* Connect to server by specifying the profile. Create a new profile if the connection profi
le does not exist, yet. *)
        IF fbConnection.Connect(
            sFtpConfigProfileFileName:='QuickStart'
        ) THEN
            nCase := 15;
        END_IF

    11: (* Connection profile does not exist, yet. Specify connection parameters and store in a new
profile. *)
        stNTDataProfile.stConnection.eAuth := E_FtpConnectionAuth.sftp_user_pwd;
        stNTDataProfile.stConnection.nPort := 23;
        stNTDataProfile.stConnection.sHostname := '127.0.0.1';
        stNTDataProfile.stConnection.sUsername := 'My_SFTP_User';
        stNTDataProfile.stConnection.sPassword := 'SomeSecurePassword##1';
        stNTDataProfile.stConnection.sFileName := 'QuickStart';

        IF fbConnection.AddProfile(
            stFtpConfigProfile := stNTDataProfile
        ) THEN
            nCase := 16;
        END_IF

    15: (* Check connection establishment for any errors. *)
        IF fbConnection.bError THEN
            IF fbConnection.ipResultMessage.EqualsToEventClass(Global.TC_EVENT_CLASSES.TcFtpClientEv
entClass) THEN
                CASE fbConnection.ipResultMessage.nEventId OF
                    E_FtpClientErrorCode.Code_1101_HostConnectionOpen: (* Already connected to the c
orrect host, continue*)
                        nCase := 20;
                    E_FtpClientErrorCode.Code_1100_AnyConnectionOpen: (* Already connected to anothe
r host. Abort.*)
                        nCase := 210;
                    E_FtpClientErrorCode.Code_1102_FileLocationError: (* Profile has not been create
d yet. *)
                        nCase := 11;
                ELSE
                    nCase := 210; (* an unexpected FtpClient error has occured *)
                END_CASE
            ELSE
                nCase := 210; (* an unexpected other error (Ads, Router, ...) has occured*)
            END_IF
        ELSE
            nCase := 20; (* Start file upload. *)
        END_IF

    16: (* Check profile creation for any errors *)
        IF fbConnection.bError THEN
            IF fbConnection.ipResultMessage.EqualsToEventClass(Global.TC_EVENT_CLASSES.TcFtpClientEv
entClass) THEN
                CASE fbConnection.ipResultMessage.nEventId OF
                    E_FtpClientErrorCode.Code_1106_ParametersImproper: (* The profile is configured
improperly (e.g. HostName empty or Port 0) *)
                        nCase := 210;
                ELSE
                    nCase := 210; (* an unexpected FtpClient error has occured *)
                END_CASE
```

```
            ELSE
                nCase := 210; (* an unexpected other error (Ads, Router, ...) has occured*)
            END_IF
        ELSE
            (* now that the profile has been created, try connecting again *)
            nCase := 10;
        END_IF

    20: (* Initiate file upload. *)
        IF fbTransfer.Upload(
            fbFtpConnection         := fbConnection,
            sLocalSourcePath         := sPathToOriginFile,
            sServerTargetDirectory     := sPathToTargetDir,
            nTransferHandle         => nTransferHandle
        ) THEN
            nCase := 25;
        END_IF

    25: (* Check file upload initiation for any errors*)
        IF fbTransfer.bError THEN
            IF fbTransfer.ipResultMessage.EqualsToEventClass(Global.TC_EVENT_CLASSES.TcFtpClientEven
tClass) THEN
                CASE fbTransfer.ipResultMessage.nEventId OF
                    E_FtpClientErrorCode.Code_1106_ParametersImproper: (* Source or Target could be
empty *)
                        nCase := 210;
                    E_FtpClientErrorCode.Code_1110_Busy_TryAgain: (* The connection is busy (ONLY FT
P). Try again later. *)
                        nCase := 210;
                ELSE
                    nCase := 210; (* an unexpected FtpClient error has occured *)
                END_CASE
            ELSE
                nCase := 210; (* an unexpected other error (Ads, Router, ...) has occured*)
            END_IF
        ELSE
            (* now that the transfer has been initialised, check transfer state *)
            nCase := 30;
        END_IF

    30: (* File upload has been started, now cyclically check transfer state. *)
        fbTimer(IN:=TRUE);
        IF fbTimer.Q THEN
            fbTimer(IN:=FALSE);
            nCase := 40;
        END_IF

    40: (* Request transfer state *)
        IF fbTransfer.GetState(
            fbFtpConnection         := fbConnection,
            nTransferHandle         := nTransferHandle,
            stTransferState            => stTransferInfo
        ) THEN
            nCase := 45;
        END_IF

    45: (* Check GetState for any errors*)
        IF fbTransfer.bError THEN
            IF fbTransfer.ipResultMessage.EqualsToEventClass(Global.TC_EVENT_CLASSES.TcFtpClientEven
tClass) THEN
                CASE fbTransfer.ipResultMessage.nEventId OF
                    E_FtpClientErrorCode.Code_1106_ParametersImproper: (* The transfer for the provi
ded handle could not be found *)
                        nCase := 210;
                ELSE
                    nCase := 210; (* an unexpected FtpClient error has occured *)
                END_CASE
            ELSE
                nCase := 210; (* an unexpected other error (Ads, Router, ...) has occured*)
            END_IF
        ELSE
            nCase := 50;
        END_IF

    50: (* Check if file transfer is completed and has not produced any errors *)
        IF (stTransferInfo.eProblem = E_FtpTransferProblemType.None) AND (stTransferInfo.eStep = E_F
tpTransferStep.TransferCompleted) THEN
            nCase := 60; // file transfer completed with no errors
        ELSIF stTransferInfo.eProblem <> E_FtpTransferProblemType.None THEN
            nCase := 210; //file transfer encountered an error
```

```
        ELSE
            nCase := 30; // not completed yet, check state again
        END_IF

    60: (* Disconnect from server. *)
        IF fbConnection.Disconnect() THEN
            nCase := 65;
        END_IF

    65: (* Check disconnect for any errors *)
        IF fbConnection.bError THEN
            nCase := 200; (* continue, but warn user that something happened on disconnect *)
        ELSE
            nCase := 70; (* All clear, connection terminated. *)
        END_IF

    70: (* File upload completed. *)
        bDone := TRUE;
        nCase := 0;

    200: (* File upload completed but unexpected error on disconnect from server. *)
        bDone := TRUE;
        bError := TRUE;
        sErrorMSG := 'Upload was completed successfully, but received error on disconnect. Check [fb
Connection.ipResultMessage] for details.';
        nCase := 0;

    210: (* An error has occured. *)
        bDone := TRUE;
        bError := TRUE;
        sErrorMSG := 'Failed to Upload. Check [fbConnection.ipResultMessage] and [stTransferInfo.ePr
oblem]';
        nCase := 0;

END_CASE
```

Explanation of the program flow

When the program is started (setting the variable bStart to TRUE), the state machine executes a program sequence which first creates a connection profile if this does not yet exist. The system then checks whether a connection to the SFTP server already exists. If no, a connection is initiated with the specified connection parameters. In the next step, the file upload is started and the file "C:\Temp\myFileUpload.txt"' is uploaded to the SFTP server in its root directory. The state machine then immediately jumps one step further and monitors the progress of the file upload at regular intervals. Once the upload is complete, the state machine jumps to the final step and disconnects from the SFTP server. The state machine then returns to the initial step.

4.2 Basics

File Transfer Protocol (FTP)

The File Transfer Protocol (FTP) is based exclusively on TCP-based communication connections. FTP specifies two TCP ports, which are important for data transmission:

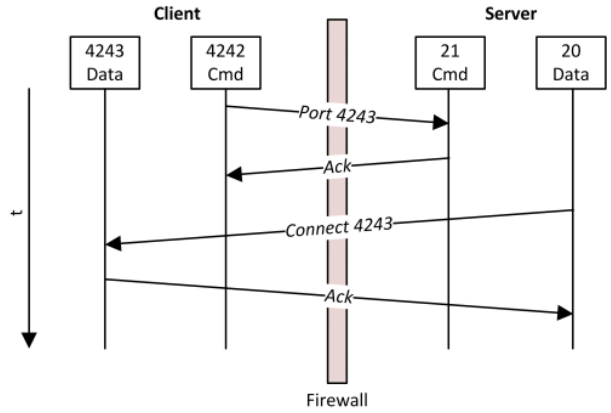
- Port 20/tcp: This port is also referred to as **data port** and is used to send/receive files and directory lists.
- Port 21/tcp: This port is generally referred to as **command port** and is used to exchange status information between the client and server.

Separate TCP connections are used for sending and receiving files (data port) and for transmitting commands (command port). With FTP, two connection modes are available for setting up such connections: "**Active FTP**" and "**Passive FTP**". Depending on the respective connection mode, the two ports mentioned above are initiated differently, which is described in more detail below.

The TwinCAT FTP Client supports passive FTP in every version and active FTP from version 1.0.8.

Active FTP

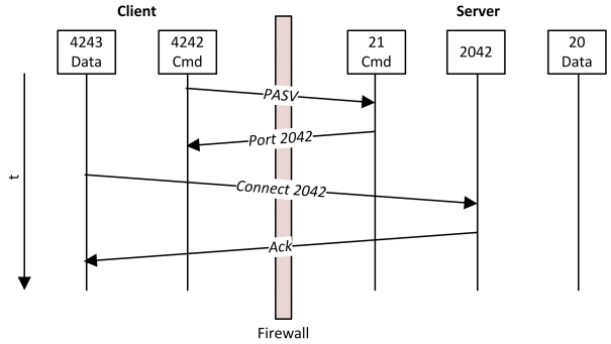
With active FTP, the client connects to the command port of the FTP server. The client uses a random port N, e.g. 4242/tcp, as source port. The client then listens on port N+1 and notifies the server of this port. The server then connects to the client on port N+1 and uses its data port as the source port.



A problem with active FTP is that the client itself does not establish a connection to the server's data port, but communicates a port (N+1) to the server, which then connects to the client via its data port. In the case of firewalls or NAT devices that are located upstream of the client, this could involve additional configuration effort on the client side, since the data port of the client behind the firewall must be accessible to the server (see figure "Connect 4243"). The data ports of the TwinCAT FTP client to be used can be permanently defined, which considerably simplifies the firewall configuration. More accurate information on this can be found at Configuration [▶ 20].

Passive FTP

This method is used when the client is not directly accessible by the server. This is the case, for example, if the client is behind a firewall that uses NAT to rewrite the client's address. With passive FTP, the FTP client initiates a connection via two random TCP ports N (command port) and N+1 (data port). The first port is used to connect to the server's command port. However, instead of the client communicating its port N+1 to the server so that the server can open a connection to it (see active FTP), the client first transmits a so-called PASV command. The server now knows that the connection is via passive FTP. As a result, the server opens a (random) port P as data port and transmits it to the client. The client then initiates a connection with port P and uses port N+1 (data port) as the source port. This connection is then used to transfer the data.



On closer examination it becomes apparent that the firewall problem of active FTP is reversed with passive FTP. On the server side, the firewall should be configured such that the data port of the server can be reached by the client. Many FTP servers offer the option to configure the data ports to be used.

SSH File Transfer Protocol (SFTP)

The SSH File Transfer Protocol or Secure File Transfer Protocol (SFTP) is an alternative to the File Transfer Protocol (FTP) designed for the Secure Shell (SSH) that enables encryption. Unlike FTP, SFTP requires only a single connection between client and server.

SFTP access is possible without access to the Secure Shell itself.

The TwinCAT FTP Client supports SFTP from version 4.x and in connection with the Tc3_FTP [▶ 27] library.

FTP over TLS / FTP over SSL / FTP over SSH (FTPS)

FTP over TLS, FTP over SSL or FTP over SSH are methods for encrypting the File Transfer Protocol (FTP). FTPS and SFTP are not compatible.

The TwinCAT FTP Client does not support FTPS.

Sources

Wikipedia: File Transfer Protocol: https://de.wikipedia.org/wiki/File_Transfer_Protocol

Wikipedia: SSH File Transfer Protocol: https://de.wikipedia.org/wiki/SSH_File_Transfer_Protocol

Wikipedia: FTP over SSL: https: //de.wikipedia.org/wiki/FTP_%C3%BCber_SSL

As of: 26 November 2024

4.3 Utilization concept

The use of the Tc3_FTP library differs from the older Tc2_FTP library due to its enhanced functionality. The connection to an FTP/SFTP server is based on so-called connection profiles. A connection profile is created once to define the access data and connection properties to the FTP/SFTP server and to reference it later in the program. In addition, special function blocks make it possible, for example, to query the status and progress of an ongoing file transfer.

The entire functionality of the Tc3_FTP library can be divided into the following areas, each of which is represented by its own function block:

Area	Function block	Description
Connection	FB_FtpConnection	Contains all functions for establishing a connection with an FTP/SFTP server, as well as for creating a connection profile.
Explorer	FB_FtpExplorer	Contains all functions for file and directory operations, for example creating or deleting a directory.
Transfer	FB_FtpTransfer	Contains all functions for uploading or downloading files.

4.4 Supported functionalities

Starting with version 4.x, the TwinCAT FTP Client has undergone some fundamental optimizations. Among other things, a new PLC library (Tc3_FTP) with extended functionality has been provided. The previous PLC library (Tc2_FTP) will continue to be supplied for compatibility reasons, but will no longer be maintained. We therefore recommend switching to the new library for new projects.

The following table provides an overview of the functionality of both libraries.

Function	Tc2_Ftp	Tc3_Ftp
FTP	x	x
SFTP	-	x
Connection open	x	x
Connection close	x	x
Connection profiles	-	x
Proxy support	-	x
File Upload	x	x
File Download	x	x
Transfer state	-	x
Transfer abort	-	x
Create directory	x	x
Remove directory	x	x
Rename directory	-	x
Move directory	-	x
Check for directory existence	x	x
List files	-	-
Remove file	x	x
Rename file	x	x
Move file	-	x
Check for file existence	x	x

4.5 Configuration

Since version 4.x, the TwinCAT FTP Client can be configured via TcFtpClientConfig.xml, which is located in the following subdirectory of %ProgramData%:

```
Beckhoff\TwinCAT\Functions\TF6300-FTP-Client
```

The settings are divided into categories and have the following meaning:

Category	Configuration entry	Description
Logging	Path	File path of the log
Logging	MaxFileSize	File size of the log in bytes
Logging	MaxLogSize	Size of a log entry in bytes
Logging	Severity	Loglevel
Server	QueueSize	Maximum number of internally buffered data.
LegacyServer	Timeout	Request timeout for Tc2_FTP
LegacyServer	MinPort	Min. port for active Tc2_FTP connections
LegacyServer	MaxPort	Max. port for active Tc2_FTP connections
LegacyServer	PortRangeEnabled	Enabling Min and MaxPort

The TwinCAT FTP Client up to version 3.x can be configured via the registry. Please follow the links below for more detailed information.

4.5.1 Configuration of the data ports for active FTP

This documentation article only applies to the TwinCAT FTP Client up to version 3.x.

In active FTP connection mode the data port for the connection to the FTP server is specified from the client and the server connects to this port. If the TwinCAT FTP Client is to be operated in active FTP mode, a port range can be defined from which a free port is to be used for establishing the connection.

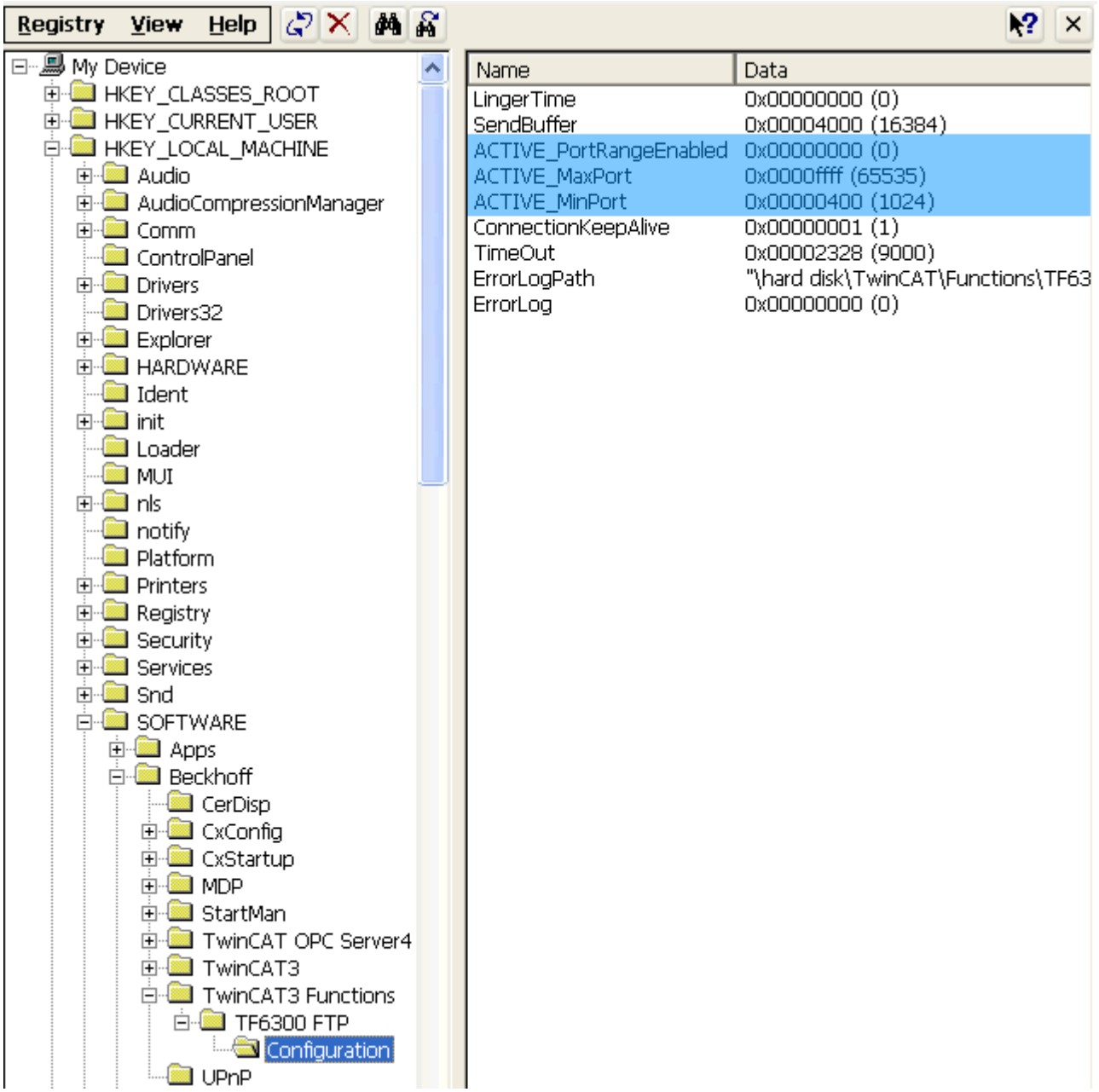
The port range is set using the following three registry keys.

"HKEY_LOCAL_MACHINE\\Software\\Beckhoff\\TwinCAT FTP Client\\Configuration\ACTIVE_PortRangeEnabled"
"HKEY_LOCAL_MACHINE\\Software\\Beckhoff\\TwinCAT FTP Client\\Configuration\ACTIVE_MaxPort"
"HKEY_LOCAL_MACHINE\\Software\\Beckhoff\\TwinCAT FTP Client\\Configuration\ACTIVE_MinPort"

To enable the port range, the registry key "ACTIVE_PortRangeEnabled" must first be set to 1. Otherwise the TwinCAT FTP Client will use any free port to establish the connection.

The registry keys "ACTIVE_MaxPort" and "ACTIVE_MinPort" can be used to define the upper and lower limits of the port range, respectively.

To allow multiple simultaneous FTP connections, the port range must be selected accordingly. Example: If the TwinCAT FTP Client is to connect to three FTP servers simultaneously and start a file transfer via each connection, the port range must contain at least 3 ports.



Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

4.5.2 Activation of an error log file

This documentation article only applies to the TwinCAT FTP Client up to version 3.x.

In order to better localize occurring errors and to obtain a better description in the event of an error, the TwinCAT FTP Client offers the option of logging occurring errors in a text file.

The text file with the name **"TcFTPErrorLog.txt"** is always generated in the folder in which the "EXE" of the TwinCAT FTP Client is located.

You can activate this functionality by creating the following registry key:

32-bit: "HKEY_LOCAL_MACHINE\Software\Beckhoff\TwinCAT3 Functions\TF6300 FTP\Configuration\ErrorLog"

64-bit: "HKEY_LOCAL_MACHINE\Software\Wow6432Node\Beckhoff\TwinCAT3 Functions\TF6300 FTP\Configuration\ErrorLog"

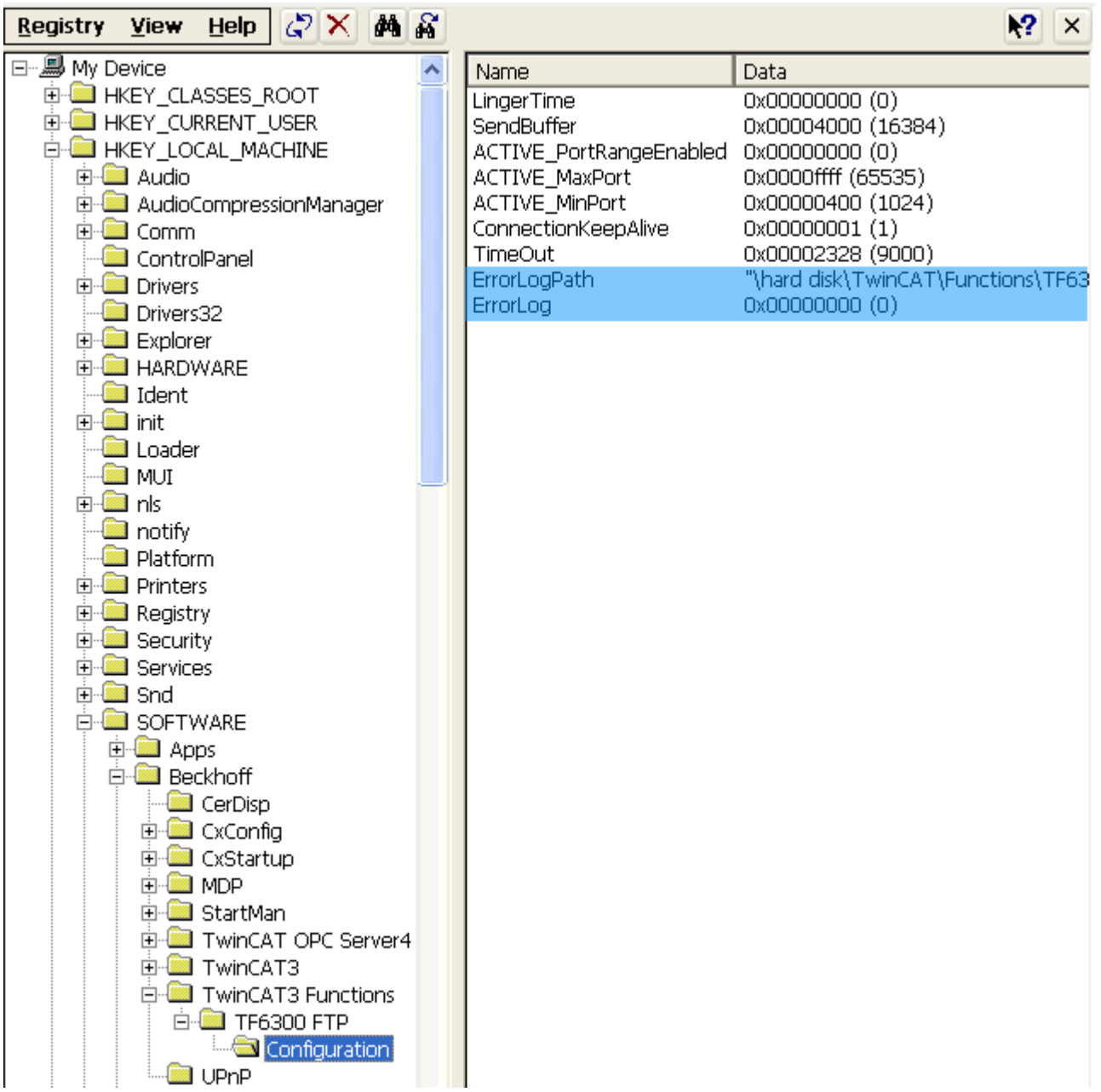


Fig. 1:

With the value 0, the functionality is stopped after a restart of the TwinCAT system.

With the value 1, the functionality is started after a restart of the TwinCAT system and a simple error logging is performed.

With the value 2, the functionality is started after a restart of the TwinCAT system and extended error logging is performed.

NOTICE

Damage to the flash medium

Cyclic writing to the flash medium can damage it. There is also the possibility that the memory of the flash medium is fully written by cyclic writing.

NOTICE

Damage to the flash medium

Only use the error log when testing!

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

4.5.3 Setting the transmit buffer

This documentation article only applies to the TwinCAT FTP Client up to version 3.x.

The transmission speed of the TwinCAT FTP Client can be regulated using the transmit buffer. This can be set in the registry under the following key.

"HKEY_LOCAL_MACHINE\Software\Beckhoff\TwinCAT3 Functions\TF6300 FTP\Configuration\SendBuffer"

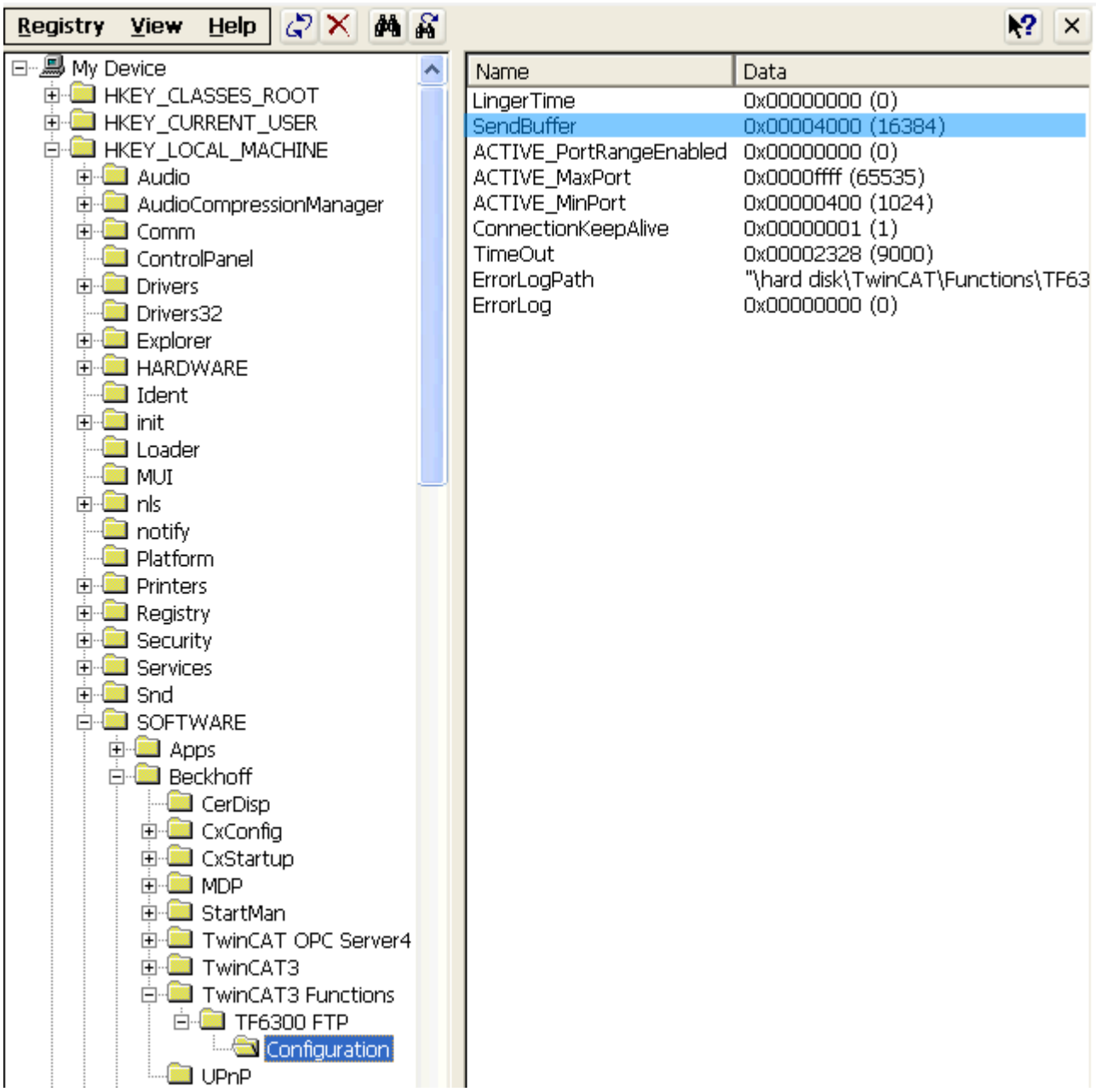


Fig. 2:

The value is specified in bytes. The default value is 16 kB.

NOTICE

Main memory load

The larger the value of the transmit buffer is set, the larger the main memory required by the TwinCAT FTP Client process. If the value is too high, this can lead to storage problems with very small devices.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

4.6 Log file

The TwinCAT FTP Client writes an extended log file, which provides helpful information in the event of diagnostics. From version 4.x, the log file is stored by default in the following subdirectory of %ProgramData%:

```
Beckhoff\TwinCAT\Functions\TF6300-FTP-Client\Log
```

The log file can be opened with any text editor. Settings for the size of the log file and the LogLevel used can be configured. You can find out more about the setting options under Configuration [▶ 20].

4.7 Proxy server

A proxy server is a computer or software that acts as a gateway between devices from your own network and another network (e.g. the Internet) and forwards the data. Proxy servers can also perform other tasks, such as acting as a firewall and/or content filter. The idea here is that all data traffic is routed through the proxy server so that appropriate rules can be used there to monitor the data traffic.

In the case of a PLC controller, depending on the application scenario, it may be necessary to route traffic between the controller and an external FTP/SFTP server via a proxy server.

There are many different software and hardware applications that provide proxy server functionality. A complete list of supported proxy servers would therefore not be practical and would be difficult to maintain. In general, however, the TwinCAT FTP Client supports the following proxy servers, depending on the protocol used for communication between the TwinCAT FTP Client and the proxy server:

Type	Description
SOCKS4	Proxy server based on the SOCKS protocol, version 4.
SOCKS4a	Proxy server based on the SOCKS protocol, version 4a.
SOCKS5	Proxy server based on the SOCKS protocol, version 5.
HttpConnect	Proxy server based on the HTTP protocol.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1 Build 4026	PC or CX (x86)	Tc3_FTP

5 PLC libraries

5.1 Tc3_FTP

5.1.1 Function blocks

5.1.1.1 FB_FtpConnection

This function block can be used to manage a connection to an FTP/SFTP server. Different methods allow you to create a profile, connect to the server, and disconnect from the server.

Inputs

```
VAR_INPUT
    tTimeout : TIME;
    sNetID   : T_AmsNetID;
END_VAR
```

Name	Type	Description
tTimeout	TIME	Indicates the ADS timeout to be used between PLC runtime and the FTP client user mode process (in seconds).
sNetID	T_AmsNetID	AmsNetID of the device on which the FTP client user mode process is running. Normally, the FTP client and the PLC run on the same system, so no value needs to be assigned to this variable.

Outputs

```
VAR_OUTPUT
    bError           : BOOL;
    ipResultMessage  : I_TcMessage;
    bBusy            : BOOL;
    eActiveOperation : E_FtpClientOperation;
    uConnectionState : U_FtpConnectionState;
END_VAR
```

Name	Type	Description
bError	BOOL	TRUE if an error has occurred.
ipResultMessage	I_TcMessage	Output with extended error and diagnostic information based on the TwinCAT 3 EventLogger.
bBusy	BOOL	TRUE as soon as the function block is in use.
eActiveOperation	E_FtpClientOperation [▶ 41]	Indicates the current operation.
uConnectionState	U_FtpConnectionState	Indicates the current connection state. Will be updated automatically.

Properties

Name	Type	Access	Description
nIdentifier	UDINT	Get	Indicates the ID of the connection that was assigned during initialization.

Methods

Name	Description
AddProfile [▶ 28]	Allows you to add a new connection profile.
Connect [▶ 28]	Starts the connection establishment to a server.
Disconnect [▶ 29]	Disconnects from a server.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1 Build 4026	PC or CX (x86)	Tc3_FTP

5.1.1.1.1 AddProfile

Adds a new connection profile.

Syntax

```
METHOD AddProfile : BOOL
VAR_INPUT
    stFtpConfigProfile : ST_FtpConfigProfile;
END_VAR
```

Return value

Name	Type	Description
AddProfile	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.

Inputs

Name	Type	Description
stFtpConfigProfile	ST_FtpConfigProfile [▶ 43]	Structure with further information on the profile to be created.

5.1.1.1.2 Connect

Starts connecting to a server by specifying the appropriate connection profile.

Syntax

```
METHOD Connect : BOOL
VAR_INPUT
    sFtpConfigProfileFileName : STRING;
END_VAR
```

Return value

Name	Type	Description
Connect	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.

Inputs

Name	Type	Description
sFtpConfigProfileFileName	STRING	Indicates the name of the connection profile.

5.1.1.1.3 Disconnect

Disconnects from a server.

Syntax

```
METHOD Disconnect : BOOL
```

Return value

Name	Type	Description
Disconnect	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.

5.1.1.2 FB_FtpExplorer

This function block can be used to perform file and directory operations, e.g. creating new directories on the FTP server or reading existing directories. The corresponding operations are available as methods on the function block.

Inputs

```
VAR_INPUT
    tTimeout : TIME;
    sNetID   : T_AmsNetID;
END_VAR
```

Name	Type	Description
tTimeout	TIME	Indicates the ADS timeout to be used between PLC runtime and the FTP client user mode process (in seconds).
sNetID	T_AmsNetID	AmsNetID of the device on which the FTP client user mode process is running. Normally, the FTP client and the PLC run on the same system, so no value needs to be assigned to this variable.

Outputs

```
VAR_OUTPUT
    bError           : BOOL;
    ipResultMessage  : I_TcMessage;
    bBusy            : BOOL;
    eActiveOperation : E_FtpClientOperation;
END_VAR
```

Name	Type	Description
bError	BOOL	TRUE if an error has occurred.
ipResultMessage	I_TcMessage	Output with extended error and diagnostic information based on the TwinCAT 3 EventLogger.
bBusy	BOOL	TRUE as soon as the function block is in use.
eActiveOperation	E_FtpClientOperation [▶ 41]	Indicates the current transfer operation.

Methods

Name	Description
DirCreate [▶ 30]	Creates a new directory.
DirExists [▶ 31]	Checks whether a directory exists.
DirMove [▶ 31]	Moves a directory.
DirRemove [▶ 32]	Removes a directory.
DirRename [▶ 32]	Renames a directory.
FileExists [▶ 33]	Checks whether a file exists.
FileMove [▶ 34]	Moves a file.
FileRemove [▶ 34]	Removes a file.
FileRename [▶ 35]	Renames a file.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1 Build 4026	PC or CX (x86)	Tc3_FTP

5.1.1.2.1 DirCreate

Creates a new directory on the server.

Syntax

```
METHOD DirCreate : BOOL
VAR_INPUT
    fbFtpConnection : REFERENCE TO FB_FtpConnection;
    sDirectory      : STRING;
END_VAR
```

Return value

Name	Type	Description
DirCreate	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
sDirectory	STRING	Path to the directory.

5.1.1.2.2 DirExists

Checks whether a specified directory already exists on the server.

Syntax

```
METHOD DirExists : BOOL
VAR_INPUT
    fbFtpConnection : REFERENCE TO FB_FtpConnection;
    sDirectory      : STRING;
END_VAR
```

Return value

Name	Type	Description
DirExists	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.
bExists	BOOL	Returns TRUE if the directory already exists.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
sDirectory	STRING	Path to the directory.

5.1.1.2.3 DirMove

Moves a directory on the server.

Syntax

```
METHOD DirMove : BOOL
VAR_INPUT
    fbFtpConnection : REFERENCE TO FB_FtpConnection;
    sDirectory : STRING;
    sTargetDirectory : STRING;
END_VAR
```

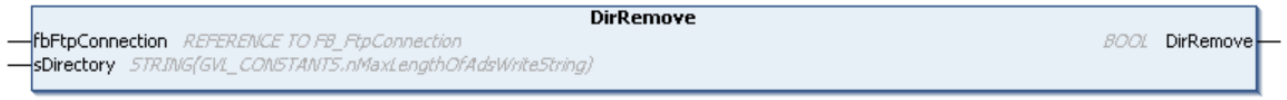
Return value

Name	Type	Description
DirMove	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
sDirectory	STRING	Path to the source directory.
sTargetDirectory	STRING	Path to the target directory to which the source directory is to be moved.

5.1.1.2.4 DirRemove



Removes a directory on the server.

Syntax

```
METHOD DirRemove : BOOL
VAR_INPUT
    fbFtpConnection : REFERENCE TO FB_FtpConnection;
    sDirectory      : STRING;
END_VAR
```

Return value

Name	Type	Description
DirRemove	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
sDirectory	STRING	Path to the directory.

5.1.1.2.5 DirRename

Renames an existing directory on the server.

Syntax

```
METHOD DirRename : BOOL
VAR_INPUT
    fbFtpConnection : REFERENCE TO FB_FtpConnection;
    sDirectory      : STRING;
    sNewName        : STRING;
END_VAR
```

Return value

Name	Type	Description
DirRename	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
sDirectory	STRING	Path to the directory to be renamed.
sNewName	STRING	New directory name.

5.1.1.2.6 FileExists

Checks if a given file exists on the server.

Syntax

```
METHOD FileExists : BOOL
VAR_INPUT
    fbFtpConnection : REFERENCE TO FB_FtpConnection;
    sFilePath       : STRING;
END_VAR
```

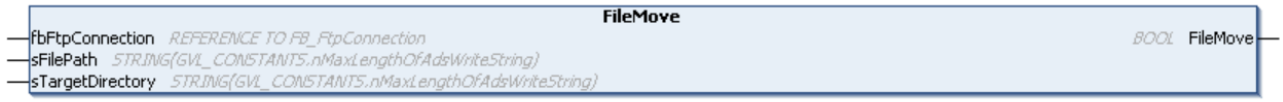
Return value

Name	Type	Description
FileExists	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.
bExists	BOOL	Returns TRUE if the file exists on the server.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
sFilePath	STRING	Path to the file on the server.

5.1.1.2.7 FileMove



Moves a file on the server.

Syntax

```
METHOD FileMove : BOOL
VAR_INPUT
    fbFtpConnection  : REFERENCE TO FB_FtpConnection;
    sFilePath        : STRING;
    sTargetDirectory : STRING;
END_VAR
```

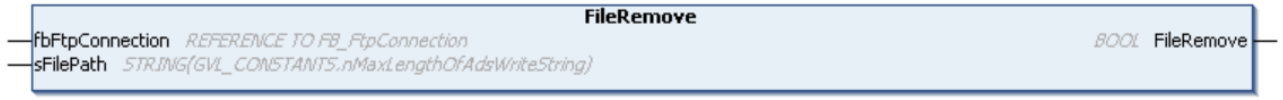
Return value

Name	Type	Description
FileMove	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
sFilePath	STRING	Path to the source file on the server.
sTargetDirectory	STRING	Path to the target directory to which the file is to be moved.

5.1.1.2.8 FileRemove



Removes a file from the server.

Syntax

```
METHOD FileRemove : BOOL
VAR_INPUT
    fbFtpConnection : REFERENCE TO FB_FtpConnection;
    sFilePath       : STRING;
END_VAR
```

Return value

Name	Type	Description
FileRemove	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
sFilePath	STRING	Path to the file on the server.

5.1.1.2.9 FileRename

Renames a file on the server.

Syntax

```
METHOD FileRename : BOOL
VAR_INPUT
    fbFtpConnection : REFERENCE TO FB_FtpConnection;
    sFilePath       : STRING;
    sNewName        : STRING;
END_VAR
```

Return value

Name	Type	Description
FileRename	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
sFilePath	STRING	Path to the file on the server.
sNewName	STRING	New filename.

5.1.1.3 FB_FtpTransfer

This function block can be used to initiate and manage upload and download operations.

Inputs

```
VAR_INPUT
    tTimeout : TIME;
    sNetID   : T_AmsNetID;
END_VAR
```

Name	Type	Description
tTimeout	TIME	Indicates the ADS timeout to be used between PLC runtime and the FTP client user mode process (in seconds).
sNetID	T_AmsNetID	AmsNetID of the device on which the FTP client user mode process is running. Normally, the FTP client and the PLC run on the same system, so no value needs to be assigned to this variable.

Outputs

```
VAR_OUTPUT
    bError           : BOOL;
    ipResultMessage  : I_TcMessage;
    bBusy            : BOOL;
    eActiveOperation : E_FtpClientOperation;
END_VAR
```

Name	Type	Description
bError	BOOL	TRUE if an error has occurred.
ipResultMessage	I_TcMessage	Output with extended error and diagnostic information based on the TwinCAT 3 EventLogger.
bBusy	BOOL	TRUE as soon as the function block is in use.
eActiveOperation	E_FtpClientOperation [▶ 41]	Indicates the current transfer operation.

Methods

Name	Description
Abort [▶ 36]	Aborts a transfer in progress.
Download [▶ 37]	Downloads files or folders from a server.
GetActive [▶ 37]	Returns the number of transfer operations currently in progress.
GetActiveVerbose [▶ 38]	Returns the number of transfer operations currently in progress and a list of transfer handles.
GetState [▶ 39]	Returns the state of a transfer operation currently in progress.
GetStateVerbose [▶ 39]	Returns the detailed state of a transfer operation currently in progress (verbose).
Upload [▶ 40]	Uploads files or folders to a server.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1 Build 4026	PC or CX (x86)	Tc3_FTP

5.1.1.3.1 Abort

Aborts an existing transfer.

Syntax

```
METHOD Abort : BOOL
VAR_INPUT
    fbFtpConnection : REFERENCE TO FB_FtpConnection;
    nTransferHandle : UDINT;
END_VAR
```

Return value

Name	Type	Description
Abort	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
nTransferHandle	UDINT	TransferHandle of the transfer operation that is to be aborted. Is returned when upload or download is called up.

5.1.1.3.2 Download

Starts a download operation.

Syntax

```
METHOD Download : BOOL
VAR_INPUT
    fbFtpConnection       : REFERENCE TO FB_FtpConnection;
    sServerSourcePath     : STRING;
    sLocalTargetDirectory : STRING;
END_VAR
```

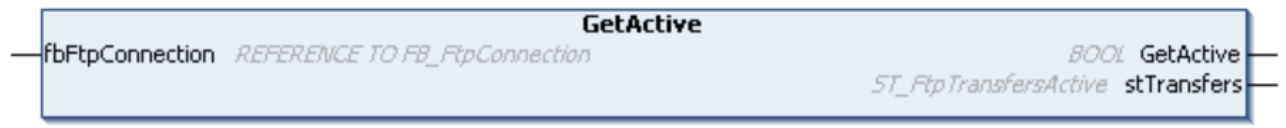
Return value

Name	Type	Description
Download	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.
nTransferHandle	UDINT	Handle for the transfer operation.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
sServerSourcePath	STRING	Path to the source on the server. Can be a file path or a directory path. Supports wildcards. Example: "*.txt" downloads all TXT files in the root directory of the server.
sLocalTargetDirectory	STRING	Path to the local target directory.

5.1.1.3.3 GetActive



Provides an overview of current transfer operations.

Syntax

```
METHOD GetActive : BOOL
VAR_INPUT
    fbFtpConnection : REFERENCE TO FB_FtpConnection;
END_VAR
VAR_OUTPUT
    stTransfers     : ST_FtpTransfersActive;
END_VAR
```

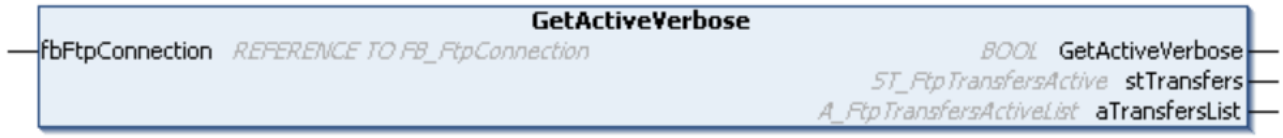
Return value

Name	Type	Description
GetActive	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.
stTransfers	ST_FtpTransfersActive [▶ 44]	Contains the number of transfer operations currently in progress.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.

5.1.1.3.4 GetActiveVerbose



Provides detailed information about transfer operations currently in progress.

Syntax

```
METHOD GetActiveVerbose : BOOL
VAR_INPUT
    fbFtpConnection : REFERENCE TO FB_FtpConnection;
END_VAR
VAR_OUTPUT
    stTransfers     : ST_FtpTransfersActive;
    aTransfersList  : A_FtpTransfersActiveList;
END_VAR
```

Return value

Name	Type	Description
GetActiveVerbose	BOOL	Returns TRUE if the currently active transfer operations have been successfully returned.
stTransfers	ST_FtpTransfersActive [▶ 44]	Contains the number of transfer operations currently in progress.
aTransfersList	A_FtpTransfersActiveList [▶ 41]	Contains a list of TransferHandles of all currently running transfer operations.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.

5.1.1.3.5 GetState

Returns the current state of a specific transfer operation. This method can be used, for example, to check the progress of a file upload/download.

Syntax

```
METHOD GetState : BOOL
VAR_INPUT
    fbFtpConnection : REFERENCE TO FB_FtpConnection;
    nTransferHandle : UDINT;
END_VAR
VAR_OUTPUT
    stTransferState : ST_FtpTransferState;
END_VAR
```

Return value

Name	Type	Description
GetState	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.
stTransferState	ST_FtpTransferState [▶ 44]	Contains the state information for the transfer operation.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
nTransferHandle	UDINT	TransferHandle of the transfer operation.

5.1.1.3.6 GetStateVerbose

Returns detailed information on a specific transfer operation. This method can be used, for example, to check the progress of a file upload/download.

Syntax

```
METHOD GetStateVerbose : BOOL
VAR_INPUT
    fbFtpConnection : REFERENCE TO FB_FtpConnection;
    nTransferHandle : UDINT;
END_VAR
```

```
VAR_OUTPUT
    stTransferStateVerbose : ST_FtpTransferStateVerbose;
END_VAR
```

Return value

Name	Type	Description
GetStateVerbose	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.
stTransferStateVerbose	ST_FtpTransferStateVerbose [▶ 44]	Contains detailed information on the transfer operation.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
nTransferHandle	UDINT	TransferHandle of the transfer operation.

5.1.1.3.7 Upload

Starts an upload operation.

Syntax

```
METHOD Upload : BOOL
VAR_INPUT
    fbFtpConnection        : REFERENCE TO FB_FtpConnection;
    sLocalSourcePath       : STRING;
    sServerTargetDirectory : STRING;
END_VAR
```

Return value

Name	Type	Description
Upload	BOOL	Returns TRUE when the method has completed. Whether the action was successful can be read from the bError variable of the function block.
nTransferHandle	UDINT	Handle for the transfer operation.

Inputs

Name	Type	Description
fbFtpConnection	FB_FtpConnection [▶ 27]	References the server connection to be used for the operation.
sLocalSourcePath	STRING	Path to the local source. Can be a file path or a directory path. Supports wildcards. Example: "C:\Temp*" uploads everything in the Temp folder.
sServerTargetDirectory	STRING	Path to the target directory on the server.

5.1.2 Data types

5.1.2.1 A_FtpTransfersActiveList

```
TYPE A_FtpTransfersActiveList: ARRAY [0..255] OF UDINT; END_TYPE
```

5.1.2.2 E_FtpClientOperation

Indicates the transfer operation currently in progress on the function block FB_FtpTransfer [▶ 35].

```
TYPE E_FtpClientOperation
(
    NONE,
    _conn_Initialize,
    _conn_GetState,
    _GetData,
    conn_Connect,
    conn_Disconnect,
    conn_AddProfile,
    explorer_DirExists,
    explorer_DirCreate,
    explorer_DirRemove,
    explorer_DirRename,
    explorer_DirMove,
    explorer_FileExists,
    explorer_FileRemove,
    explorer_FileRename,
    explorer_FileMove,
    explorer_List,
    explorer_ListVerbose,
    transfer_GetState,
    transfer_GetStateVerbose,
    transfer_Upload,
    transfer_Download,
    transfer_Abort,
    transfer_GetActive,
    transfer_GetActiveVerbose,
    clearError_conn,
    clearError_explorer,
    clearError_transfer
);
END_TYPE
```

5.1.2.3 E_FtpClientErrorCode

Defines return values for the function blocks FB_FtpConnection [▶ 27], FB_FtpExplorer [▶ 29], FB_FtpTransfer [▶ 35].

```
TYPE E_FtpClientErrorCode
(
    None,
    Code_1000_NoError_SUCCESS,
    Code_1001_ConnectFailure,
    Code_1002_ConnectionClosed,
    Code_1003_SocketError,
    Code_1004_NameResolutionFailure,
    Code_1005_Pending,
    Code_1006_ProtocolError,
    Code_1007_ProxyNameResolutionFailure,
    Code_1008_ReceiveFailure,
    Code_1009_OperationAborted,
    Code_1010_UnclassifiableError,
    Code_1011_SendFailure,
    Code_1012_ServerProtocolViolation,
    Code_1013_Timeout,
    Code_1014_AsyncError,
    Code_1015_OperationFailure,
    Code_1016_OutOfBounds,
    Code_1100_AnyConnectionOpen,
    Code_1101_HostConnectionOpen,
    Code_1102_FileLocationError,
    Code_1103_FileContentError,
    Code_1104_NotConfigured,
    Code_1105_ParameterNullError,
```

```
    Code_1106_ParametersImproper,
    Code_1107_AuthenticationError,
    Code_1108_NotConnected,
    Code_1109_UnknownAction,
    Code_1110_Busy_TryAgain,
    Code_1111_SetupFail,
    Code_1112_NotInitialized,
    Code_1113_MultipleCallError,
    Code_1255_UndefinedError,
    TCPCode_10054_ResetByPeer,
    TCPCode_10060_Timeout,
    TCPCode_10061_Refused,
    TCPCode_10066_DirNotEmpty,
    TCPCode_10068_UserQuotaExceeded
);
END_TYPE
```

5.1.2.4 E_FtpConnectionAuth

Specifies the type of authentication when establishing an FTP/SFTP connection.

```
TYPE E_FtpConnectionAuth
(
    ftp_user_pwd,
    ftp_user_pwd_proxy,
    sftp_user_pwd,
    sftp_user_pwd_proxy,
    sftp_public_private_key,
    sftp_x509_certificate
) INT;
END_TYPE
```

5.1.2.5 E_FtpConnectionMode

Specifies the FTP mode (Active/Passive).

```
TYPE E_FtpConnectionMode
(
    NONE,
    ACTIVE,
    PASSIVE
) INT;
END_TYPE
```

5.1.2.6 E_FtpProxyAuth

Specifies the type of proxy authentication.

```
TYPE E_FtpProxyAuth
(
    Basic,
    Ntlm,
    Digest
);
END_TYPE
```

5.1.2.7 E_FtpProxyType

Specifies the proxy type.

```
TYPE E_FtpProxyType
(
    None,
    Socks4,
    Socks4a,
    Socks5,
    HttpConnect,
    FtpSite,
    FtpUser,
    FtpOpen,
    FtpDoubleLogin
);
END_TYPE
```

5.1.2.8 E_FtpTransferAction

```
TYPE E_FtpTransferAction
(
    None,
    Uploading,
    Downloading,
    Deleting,
    Listing
) SINT;
END_TYPE
```

5.1.2.9 E_FtpTransferProblemType

```
TYPE E_FtpTransferProblemType
(
    None,
    FileExists,
    LinkDetected,
    InfiniteLoopDetected,
    CannotCreateDirectory,
    CannotTransferFile,
    CannotReadFromDirectory,
    CannotFindFile,
    FileNameIsInvalidOnTargetFileSystem,
    DirectoryNameIsInvalidOnTargetFileSystem,
    CannotFindDirectory,
    CannotFindLink,
    CannotResolveLink,
    NotFileOrDirectory,
    OperationCanceled,
    UnsupportedFeature,
    CannotDeleteFile,
    CannotDeleteDirectory,
    CannotCalculateChecksum
) SINT;
END_TYPE
```

5.1.2.10 E_FtpTransferStep

```
TYPE E_FtpTransferStep
(
    None,
    DirectoryProcessing,
    FileTransferring,
    FileTransferred,
    DataBlockProcessed,
    TransferCompleted
) SINT;
END_TYPE
```

5.1.2.11 ST_FtpConnectionState

```
TYPE ST_FtpConnectionState :
STRUCT
    bConnected        : BIT;
    bAuthenticated    : BIT;
    bConfigured       : BIT;
    bProxyConfigured  : BIT;
    bUsingFTP         : BIT;
    bSecured          : BIT;
    bBusy             : BIT;
END_STRUCT
END_TYPE
```

5.1.2.12 ST_FtpConfigProfile

```
TYPE ST_FtpConfigProfile :
STRUCT
    stConnection : ST_FtpConfigProfileConnection [▶ 44];
    stProxy      : ST_FtpConfigProfileProxy [▶ 44];
END_STRUCT
END_TYPE
```

5.1.2.13 ST_FtpConfigProfileConnection

```
TYPE ST_FtpConfigProfileConnection :
STRUCT
    eAuth      : E_FtpConnectionAuth [▶ 42];
    nPort      : UINT;
    eMode      : E_FtpConnectionMode [▶ 42];
    sHostname  : STRING;
    sUsername  : STRING;
    sPassword  : STRING;
    sPathToKey : STRING;
    sFileName  : STRING;
END_STRUCT
END_TYPE
```

5.1.2.14 ST_FtpConfigProfileProxy

```
TYPE ST_FtpConfigProfileProxy :
STRUCT
    eAuth          : E_FtpProxyAuth [▶ 42];
    nPort          : DINT;
    eType          : E_FtpProxyType [▶ 42];
    bByPassOnLocal : BOOL;
    sProxyHostname : STRING;
    sProxyUsername : STRING;
    sProxyPassword : STRING;
    sProxyDomain   : STRING;
END_STRUCT
END_TYPE
```

5.1.2.15 ST_FtpTransfersActive

```
TYPE ST_FtpTransfersActive :
STRUCT
    nTotal     : SINT;
    nUploads   : SINT;
    nDownloads : SINT;
    nReserved  : BYTE;
END_STRUCT
END_TYPE
```

5.1.2.16 ST_FtpTransferState

```
TYPE ST_FtpTransferState :
STRUCT
    eAction   : E_FtpTransferAction [▶ 43];
    eStep     : E_FtpTransferStep [▶ 43];
    nProgress : USINT;
    eProblem  : E_FtpTransferProblemType [▶ 43];
END_STRUCT
END_TYPE
```

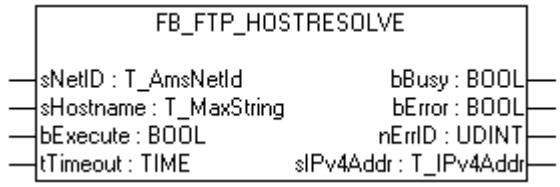
5.1.2.17 ST_FtpTransferStateVerbose

```
TYPE ST_FtpTransferStateVerbose :
STRUCT
    nHandle                       : UDINT;
    eAction                       : E_FtpTransferAction [▶ 43];
    eStep                         : E_FtpTransferStep [▶ 43];
    eProblem                      : E_FtpTransferProblemType [▶ 43];
    fProgressPercentage           : LREAL;
    fCurrentFileProgressPercentage : LREAL;
    nBytesTotal                   : LINT;
    nBytesTransferred             : LINT;
    nBytesSinceLastEvent          : LINT;
    nBytesPerSecond               : LINT;
    nCurrentFileBytesTransferred  : LINT;
    nFilesTotal                   : DINT;
    nFilesProcessed               : DINT;
    nFilesTransferred             : DINT;
END_STRUCT
END_TYPE
```

5.2 Tc2_FTP

5.2.1 Function Blocks

5.2.1.1 FB_FTP_HostResolve



Host names of FTP servers can be resolved with the function block FB_FTP_HostResolve.

Inputs

```
VAR_INPUT
sNetID   : T_AmsNetId := '';
sHostname: T_MaxString:= '';
bExecute : BOOL;
tTimeout : TIME       := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
sHostname	T_MaxString	Host name to resolve to the associated IPv4 address.
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

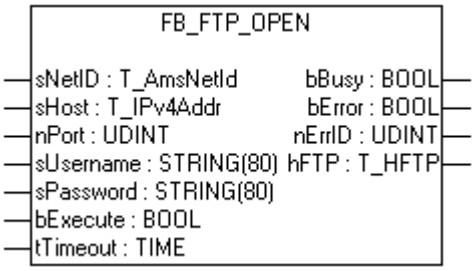
```
VAR_OUTPUT
bBusy    : BOOL;
bError   : BOOL;
nErrID   : UDINT;
sIPv4Addr: T_IPv4Addr;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the corresponding error code in the event of a set bError output.
sIPv4Addr	T_IPv4Addr	Returns the matching IPv4 address to the specified host name.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.2 FB_FTP_Open



With the function block FB_FTP_Open a connection to an FTP server can be established. The handle returned can then be used to perform further actions on the FTP server. The connection type used is Passive FTP.

Inputs

```
VAR_INPUT
sNetID    : T_AmsNetId:= '';
sHost     : T_IPv4Addr:= '127.0.0.1';
nPort     : UDINT     := 21;
sUsername: STRING     := '';
sPassword: STRING     := '';
bExecute : BOOL;
tTimeout : TIME       := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
sHost	T_IPv4Addr	A string containing the IPv4 address of the FTP server.
nPort	UDINT	FTP Port (default 21)
sUsername	STRING	User name for FTP server authentication
sPassword	STRING	Password for FTP server authentication
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

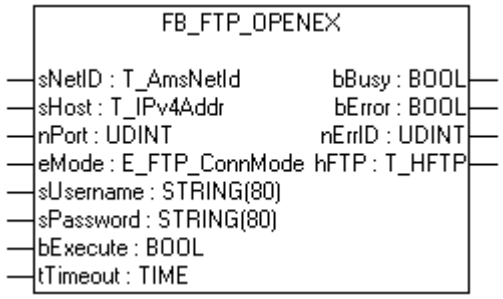
```
VAR_OUTPUT
bBusy   : BOOL;
bError : BOOL;
nErrID : UDINT;
hFTP    : T_HFTP;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the corresponding error code in the event of a set bError output.
hFTP	T_HFTP	Handle of the FTP connection

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.3 FB_FTP_OpenEX



With the function block FB_FTP_OpenEx a connection to an FTP server can be established. The handle returned can then be used to perform further actions on the FTP server. Additionally, the connection type - Passive or Active FTP - can be selected.

Inputs

```
VAR_INPUT
sNetID    : T_AmsNetId     := '';
sHost     : T_IPv4Addr     := '127.0.0.1';
nPort     : UDINT          := 21;
eMode     : E_FTP_ConnMode := eConnMode_PASSIVE;
sUsername : STRING         := '';
sPassword : STRING         := '';
bExecute  : BOOL;
tTimeout  : TIME           := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
sHost	T_IPv4Addr	A string containing the IPv4 address of the FTP server.
nPort	UDINT	FTP Port (default 21).
eMode	E_FTP_ConnMode [▶ 63]	FTP connection type (Active / Passive).
sUsername	STRING	User name for FTP server authentication
sPassword	STRING	Password for FTP server authentication
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

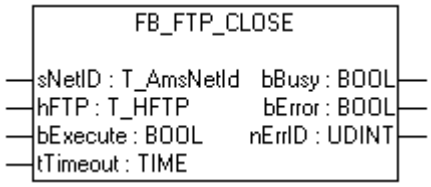
```
VAR_OUTPUT
bBusy  : BOOL;
bError : BOOL;
nErrID : UDINT;
hFTP   : T_HFTP;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the corresponding error code in the event of a set bError output.
hFTP	T_HFTP	Handle of the FTP connection.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.4 FB_FTP_Close



An existing connection to an FTP server can be closed with the function block FB_FTP_Close.

Inputs

```
VAR_INPUT
sNetID   : T_AmsNetId:= '';
hFTP     : T_HFTP;
bExecute: BOOL;
tTimeout: TIME       := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
hFTP	T_HFTP	Handle of the FTP server.
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

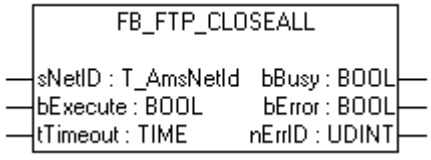
```
VAR_OUTPUT
bBusy   : BOOL;
bError : BOOL;
nErrID : UDINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the corresponding error code in the event of a set bError output.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.5 FB_FTP_CloseAll



All existing connections to an FTP server can be closed with the function block FB_FTP_CloseAll.

Inputs

```
VAR_INPUT
sNetID    : T_AmsNetId := '';
bExecute : BOOL;
tTimeout : TIME       := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

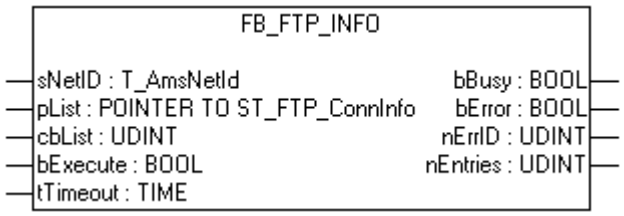
```
VAR_OUTPUT
bBusy  : BOOL;
bError : BOOL;
nErrID : UDINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	BOOL	Returns the corresponding error code in the event of a set bError output.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.6 FB_FTP_Info



Information about the created FTP server connections can be retrieved with the function block FB_FTP_Info.

Inputs

```
VAR_INPUT
sNetID   : T_AmsNetId:= '';
pList    : POINTER TO ST_FTP_ConnInfo;
cbList   : UDINT      := 0;
bExecute : BOOL;
tTimeout : TIME       := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
pList	POINTER TO ST_FTP_ConnInfo	Pointer address to a variable of type ST_FTP_ConnInfo. This variable can also be an array of the type ST_FTP_ConnInfo.
cbList	UDINT	Length of the ST_FTP_ConnInfo instance
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the duration of the timeout.

Outputs

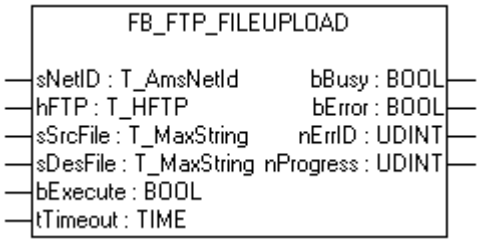
```
VAR_OUTPUT
bBusy    : BOOL;
bError   : BOOL;
nErrID   : UDINT;
nEntries: UDINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the corresponding error code in the event of a set bError output.
nEntries	UDINT	Number of existing connections.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.7 FB_FTP_FileUpload



Files can be transferred from the ADS-Device to an FTP server with the function block FB_FTP_FileUpload. This function block uses an already opened connection to an FTP server

Inputs

```
VAR_INPUT
sNetID   : T_AmsNetId := '';
hFTP     : T_HFTP;
sSrcFile : T_MaxString:= '';
sDesFile : T_MaxString:= '';
bExecute : BOOL;
tTimeout : TIME       := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
hFTP	T_HFTP	Handle of the FTP server.
sSrcFile	T_MaxString	Source file to be copied to the FTP server.
sDesFile	T_MaxString	Destination filename with path specification.
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

```
VAR_OUTPUT
bBusy     : BOOL;
bError    : BOOL;
```

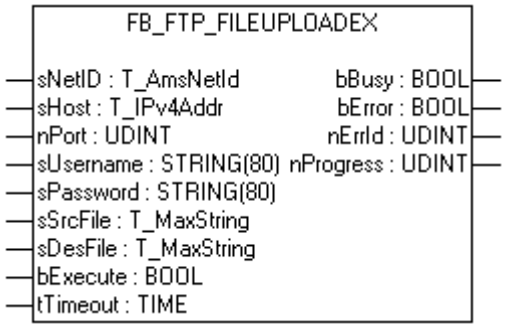
```
nErrID    : UDINT;
nProgress : UDINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the ADS error code if the bError output is set.
nProgress	UDINT	Displays the current status of the transmission in percent.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.8 FB_FTP_FileUploadEx



Files can be transferred from the ADS-Device to an FTP server with the function block FB_FTP_FileUploadEx. The connection type used is Passive FTP.

Inputs

```
VAR_INPUT
sNetID   : T_AmsNetId  := '';
sHost    : T_IPv4Addr  := '127.0.0.1';
nPort    : UDINT       := 21;
sUsername: STRING      := '';
sPassword: STRING      := '';
sSrcFile : T_MaxString := '';
sDesFile : T_MaxString := '';
bExecute : BOOL;
tTimeout : TIME        := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
sHost	T_IPv4Addr	A string containing the IPv4 address of the FTP server.
nPort	UDINT	FTP Port (default 21).
sUsername	STRING	User name for FTP server authentication.
sPassword	STRING	Password for FTP server authentication.
sSrcFile	T_MaxString	Source file to be copied to the FTP server.
sDesFile	T_MaxString	Destination filename with path specification.
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

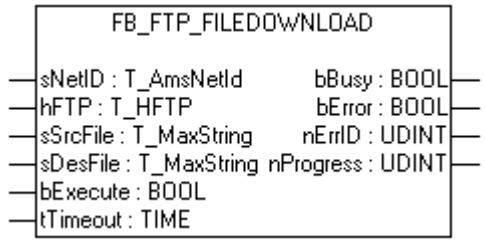
```
VAR_OUTPUT
bBusy    : BOOL;
bError   : BOOL;
nErrID   : UDINT;
nProgress: UDINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the corresponding error code in the event of a set bError output.
nProgress	UDINT	Displays the current status of the transmission in percent.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.9 FB_FTP_FileDownload



Files can be transferred from the FTP server to an ADS-Device with the function block FB_FTP_FileDownload. This function block uses an already open connection to an FTP server.

Inputs

```
VAR_INPUT
sNetID   : T_AmsNetId  := '';
hFTP     : T_HFTP
sSrcFile : T_MaxString := '';
sDesFile : T_MaxString := '';
bExecute : BOOL;
tTimeout : TIME        := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
hFTP	T_HFTP	Handle of the FTP server.
sSrcFile	T_MaxString	Source file to be copied from the FTP server.
sDesFile	T_MaxString	Destination filename with path specification.
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

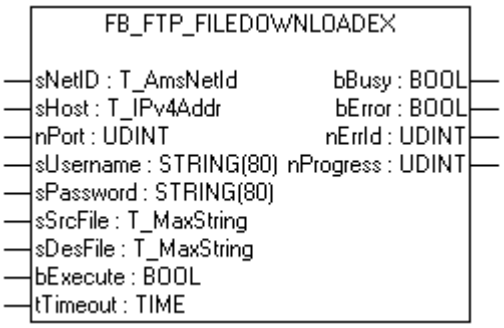
```
VAR_OUTPUT
bBusy    : BOOL;
bError   : BOOL;
nErrID   : UDINT;
nProgress: UDINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the corresponding error code in the event of a set bError output.
nProgress	UDINT	Displays the current status of the transmission in percent. With CE FTP server no intermediate values are returned only 0% and 100%.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.10 FB_FTP_FileDownloadEx



Files can be transferred from an FTP server to a ADS-Device with the function block FB_FTP_FileDownloadEx. The connection type used is Passive FTP.

Inputs

```
VAR_INPUT
sNetID    : T_AmsNetId  := '';
sHost     : T_IPv4Addr  := '127.0.0.1';
nPort     : UDINT       := 21;
sUsername : STRING      := '';
sPassword : STRING      := '';
sSrcFile  : T_MaxString := '';
sDesFile  : T_MaxString := '';
bExecute  : BOOL;
tTimeout  : TIME        := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
sHost	T_IPv4Addr	A string containing the IPv4 address of the FTP server.
nPort	UDINT	FTP Port (default 21).
sUsername	STRING	User name for FTP server authentication.
sPassword	STRING	Password for FTP server authentication.
sSrcFile	T_MaxString	Source file to be copied from the FTP server.
sDesFile	T_MaxString	Destination filename with path specification.
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

```
VAR_OUTPUT
bBusy     : BOOL;
bError    : BOOL;
```

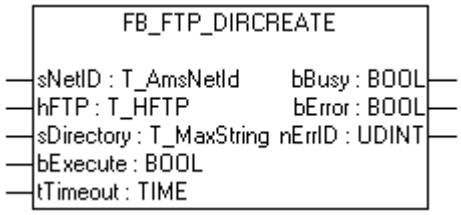
```
nErrID    : UDINT;
nProgress : UDINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the corresponding error code in the event of a set bError output.
nProgress	UDINT	Displays the current status of the transmission in percent. With CE FTP server no intermediate values are returned only 0% and 100%.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.11 FB_FTP_DirCreate



Directories can be created on an FTP server with the function block FB_FTP_DirCreate. This function block uses an already open connection to an FTP server.

Inputs

```
VAR_INPUT
sNetID   : T_AmsNetId := '';
hFTP     : T_HFTP     := 0;
sDirName : T_MaxString:= '';
bExecute : BOOL;
tTimeout : TIME       := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
hFTP	T_HFTP	Handle of the FTP server.
sDirectory	T_MaxString	The name of the directory to be created.
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

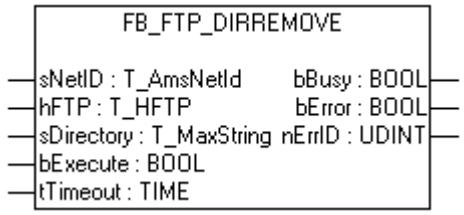
```
VAR_OUTPUT
bBusy  : BOOL;
bError : BOOL;
nErrID : UDINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the corresponding error code in the event of a set bError output.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.12 FB_FTP_DirRemove



Directories can be deleted from the FTP server with the function block FB_FTP_DirRemove. This function block uses an already open connection to an FTP server.

Inputs

```
VAR_INPUT
sNetID    : T_AmsNetId := '';
hFTP      : T_HFTP;
sDirectory: T_MaxString:= '';
bExecute  : BOOL;
tTimeout  : TIME       := T#15s;
END_VAR
```

sNetID : Is a string containing the AMS network ID of the target device to which the ADS command is directed.

hFTP : Handle of the FTP server.

sDirectory : The name of the directory to be deleted.

bExecute : The function block is executed with the rising edge.

tTimeout : Specifies the timeout period.

Outputs

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

bBusy : Command is being transmitted via ADS. No new command will be accepted as long as "bBusy" remains TRUE.

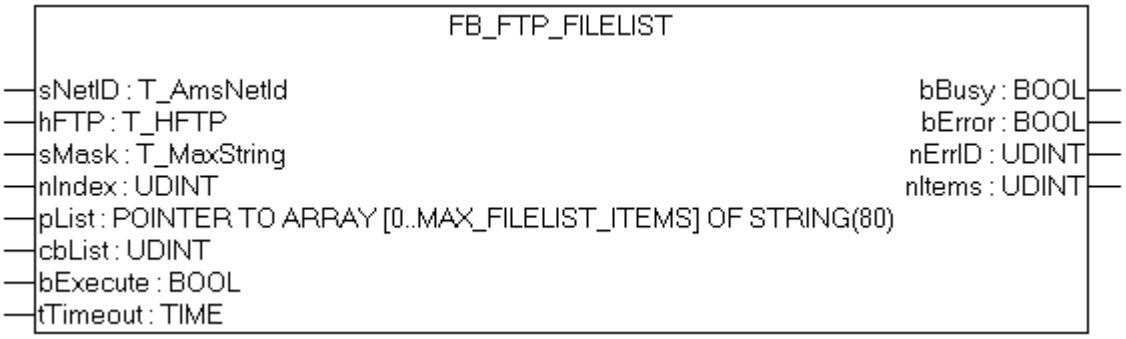
bError: Becomes TRUE, as soon as an error occurs.

nErrID : Returns the associated error code in the event of a set bError output.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.13 FB_FTP_FileList



File and folder names can be read or searched from the FTP server with the function block FB_FTP_FileList. This function block uses an already opened connection to an FTP server

Inputs

```
VAR_INPUT
sNetID  : T_AmsNetId := '';
hFTP     : T_HFTP;
sMask   : T_MaxString:= '';
nIndex  : UDINT;
pList   : POINTER TO ARRAY [0..MAX_FILELIST_ITEMS] OF STRING(80);
cbList  : UDINT;
bExecute: BOOL;
tTimeout: TIME       := T#20s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
hFTP	T_HFTP	Handle of the FTP server.
sMask	T_MaxString	Search mask to filter corresponding files: *.* Reads out all files. * Reads out all files and all folder names. *.txt Reads out all files with the extension "TXT". \Test*.* Reads out all files from the Test subfolder.
nIndex	UDINT	Specifies the index of the first filename to be read.
pList	POINTER TO ARRAY	Specifies the address of the string array into which the filenames are to be written.
cbList	UDINT	Indicates the size of the string array in bytes.
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

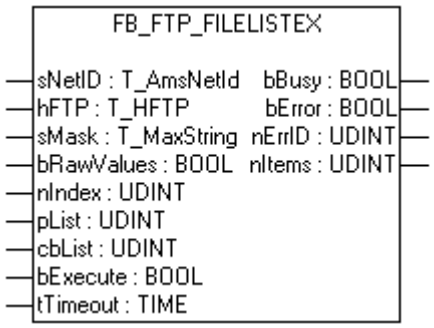
```
VAR_OUTPUT
bBusy : BOOL;
bError: BOOL;
nErrID: UDINT;
nItems: UDINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the ADS error code if the bError output is set.
nItems	UDINT	Returns the total number of file list entries.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.14 FB_FTP_FileListEx



File and folder names can be read or searched from the FTP server with the function block FB_FTP_FileListEx. This function block uses an already open connection to an FTP server. In contrast to the function block FB_FTP_FileList, details of the files/folders are also returned.

Inputs

```
VAR_INPUT
sNetID    : T_AmsNetId := '';
hFTP      : T_HFTP;
sMask     : T_MaxString:= '';
bRawValues: BOOL;
nIndex    : UDINT;
pList     : UDINT;
cbList    : UDINT;
bExecute  : BOOL;
tTimeout  : TIME       := T#20s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
hFTP	T_HFTP	Handle of the FTP server.
sMask	T_MaxString	Search mask to filter corresponding files: *.* Reads out all files. * Reads out all files and all folder names. *.txt Reads out all files with the extension "TXT". \Test*.* Reads out all files from the Test subfolder.
bRawValues	BOOL	Specifies in which form the file list should be returned: **TRUE:** Information is returned in the form of a T_MaxString array. **FALSE:** Information is returned in the form of a ST_FTP_FileDetails [▶ 63] array. (Not supported by every FTP server).
nIndex	UDINT	Specifies the index of the first file to be read.
pList	UDINT	Specifies the address of the array into which the file list is to be written. Maximum array size 256 elements.
cbList	UDINT	Indicates the size of the array in bytes.
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

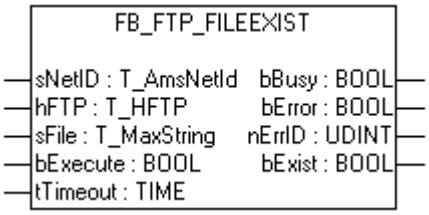
```
VAR_OUTPUT
bBusy : BOOL;
bError: BOOL;
nErrID: UDINT;
nItems: UDINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the ADS error code if the bError output is set.
nItems	UDINT	Returns the total number of file list entries.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.15 FB_FTP_FileExist



It can be checked whether a certain file exists on the FTP server with the function block FB_FTP_FileExist. This function block uses an already open connection to an FTP server.

Inputs

```
VAR_INPUT
sNetID   : T_AmsNetId := '';
hFTP     : T_HFTP;
sFile    : T_MaxString:= '';
bExecute : BOOL;
tTimeout : TIME       := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
hFTP	T_HFTP	Handle of the FTP server.
sFile	T_MaxString	Filename to be searched.
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

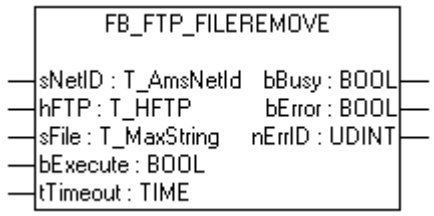
```
VAR_OUTPUT
bBusy : BOOL;
bError: BOOL;
nErrID: UDINT;
bExist: BOOL;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the corresponding error code in the event of a set bError output.
bExist	BOOL	Becomes TRUE if the specified file was found.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.16 FB_FTP_FileRemove



Files on an FTP server can be deleted with the function block FB_FTP_FileRemove. This function block uses an already open connection to an FTP server.

Inputs

```
VAR_INPUT
sNetID  : T_AmsNetId := '';
hFT     : T_HFTP;
sFile   : T_MaxString:= '';
bExecute: BOOL;
tTimeout: TIME       := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
hFTP	T_HFTP	Handle of the FTP server.
sFile	T_MaxString	The name of the file to be deleted.
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

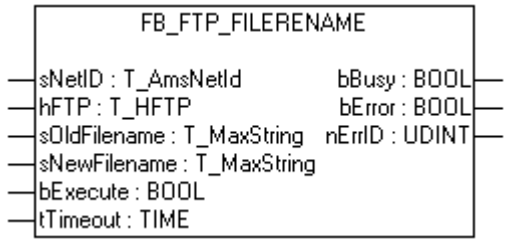
```
VAR_OUTPUT
bBusy : BOOL;
bError: BOOL;
nErrID: UDINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the corresponding error code in the event of a set bError output.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.17 FB_FTP_FileRename



Files on an FTP server can be renamed with the function block FB_FTP_FileRename. This function block uses an already open connection to an FTP server.

Inputs

```
VAR_INPUT
sNetID      : T_AmsNetId := '';
hFTP        : T_HFTP;
sOLDFilename: T_MaxString:= '';
sNEWFilename: T_MaxString:= '';
bExecute    : BOOL;
tTimeout    : TIME       := T#15s;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetId	A string containing the AMS network ID of the target device, at which the ADS command is directed.
hFTP	T_HFTP	Handle of the FTP server.
sOLDFilename	T_MaxString	The old filename of the file to be changed.
sNEWFilename	T_MaxString	The new filename.
bExecute	BOOL	The function block is executed with the rising edge.
tTimeout	TIME	Indicates the timeout period.

Outputs

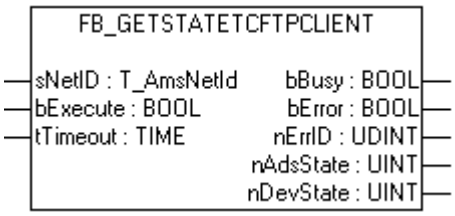
```
VAR_OUTPUT
bBusy : BOOL;
bError: BOOL;
nErrID: UDINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the corresponding error code in the event of a set bError output.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.1.18 FB_GetStateTcFTPClient



The current state of the FTP client can be queried with the function block FB_GetStateTcFTPClient.

Inputs

```
VAR_INPUT
sNetID   : T_AmsNetID;
bExecute: BOOL;
tTimeout: TIME;
END_VAR
```

Name	Type	Description
sNetID	T_AmsNetID	A string containing the AMS network ID of the target device, at which the ADS command is directed.
bExecute	BOOL	The command is executed with the rising edge.
tTimeout	TIME	Indicates the duration of the timeout.

Outputs

```
VAR_OUTPUT
bBusy    : BOOL;
bError   : BOOL;
nErrID   : UDINT;
nAdsState: UINT;
nDevState: UINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The command is in the process of being transmitted by ADS. No new command will be accepted as long as "bBusy" remains TRUE.
bError	BOOL	Becomes TRUE, as soon as an error occurs.
nErrID	UDINT	Returns the corresponding error code in the event of a set bError output.
nAdsState	UINT	Contains the state identification code of the ADS target device. The codes returned here are specified for all ADS servers: ADSSTATE_INVALID =0 ; ADSSTATE_IDLE =1 ; ADSSTATE_RESET =2 ; ADSSTATE_INIT =3 ; ADSSTATE_START =4 ; ADSSTATE_RUN =5 ; ADSSTATE_STOP =6 ; ADSSTATE_SAVECFG =7 ; ADSSTATE_LOADCFG =8 ; ADSSTATE_POWERFAILURE =9 ; ADSSTATE_POWERGOOD =10 ; ADSSTATE_ERROR =11;
nDevState	UINT	Contains the specific state identification code of the ADS target device. The codes returned here are supplementary information specific to the ADS device. 1 = TwinCAT FTP Client started

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.2 Data Types

5.2.2.1 T_HFTP

Syntax

```
TYPE T_HFTP :
STRUCT
hClient:UDINT;
END_STRUCT
END_TYPE
```

Parameter

Name	Type	Description
hClient	UDINT	Specifies the handle of the FTP connection.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.2.2 ST_FTP_ConnInfo

Syntax

```
TYPE ST_FTP_ConnInfo :
STRUCT
sHost    : T_IPv4Addr;
nPort    : UDINT;
hFTP     : T_HFTP;
```

```
sUsername: STRING;
END_STRUCT
END_TYPE
```

Parameter

Name	Type	Description
sHost	T_IPv4Addr	Specifies the IP address of the FTP server.
nPort	UDINT	Specifies the port for the FTP connection.
hFTP	T_HFTP	Specifies the handle of the FTP connection.
sUsername	STRING	Specifies the name of the logged-in user.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.2.3 ST_FTP_FileDetails

Syntax

```
TYPE ST_FTP_FileDetails:
STRUCT
bDir       : BOOL;
sPermission: STRING(10);
nSize      : UDINT;
nFilecode  : UDINT;
tTimestamp : DT;
sOwner     : STRING(79);
sGroup     : STRING(79);
sFilename  : STRING(79);
END_STRUCT
END_TYPE
```

Parameter

Name	Type	Description
bDir	BOOL	Specifies whether it is a file or a directory (not supported by every FTP server).
sPermission	STRING	Specifies the permission parameters of the file/directory (not supported by every FTP server).
nSize	UDINT	Specifies the size of the file.
nFilecode	UDINT	Returns the file code (not supported by every FTP server).
tTimstamp	DT	Specifies the timestamp of the file.
sOwner	STRING	Returns the user (not supported by every FTP server).
sGroup	STRING	Specifies the associated group (not supported by every FTP server).
sFilename	STRING	Specifies the name of the file/directory.

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.2.4 E_FTP_ConnMode

Syntax

```
TYPE E_FTP_ConnMode:(
eConnMode_PASSIVE  := 0,
eConnMode_ACTIVE   := 1
);
END_TYPE
```

Values

Name	Description
eConnMode_PASSIVE	In this mode, the FTP server creates a port and waits for the FTP client to connect. (Firewall is configured on the server side)
eConnMode_ACTIVE	In this mode, the FTP client creates a port and the FTP server connects to the client. (Firewall is configured on the client side)

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

5.2.3 Constants

5.2.3.1 Konstanten

Syntax

```
AMSPORT_FTPADSSRV      : UINT:= 10900;

FTPADS_IGR_CONNOPEN    : UDINT:= 16#100;
FTPADS_IGR_RESOLVEDNS  : UDINT:= 16#101;
FTPADS_IGR_CONNCLOSE   : UDINT:= 16#200;
FTPADS_IGR_CONNCLOSEALL: UDINT:= 16#201;

FTPADS_IGR_CONNINFO    : UDINT:= 16#300;

FTPADS_IGR_FILEUPLOAD  : UDINT:= 16#10000;
FTPADS_IGR_FILEDOWNLOAD: UDINT:= 16#20000;

FTPADS_IGR_DIRREMOV    : UDINT:= 16#30000;
FTPADS_IGR_DIRCREATE   : UDINT:= 16#30001;

FTPADS_IGR_FILEEXIST   : UDINT:= 16#40000;
FTPADS_IGR_FILERENAME  : UDINT:= 16#40001;
FTPADS_IGR_FILEREMOVE  : UDINT:= 16#40002;
FTPADS_IGR_FILELIST    : UDINT:= 16#40003;
FTPADS_IGR_FILELISTEX  : UDINT:= 16#40004;

MAX_FTP_CONNECTIONS    : UDINT:= 25;
MAX_FILELIST_ITEMS     : UDINT:= 255;

DEFAULT_FTP_PORT       : UDINT:= 21;
```

Values

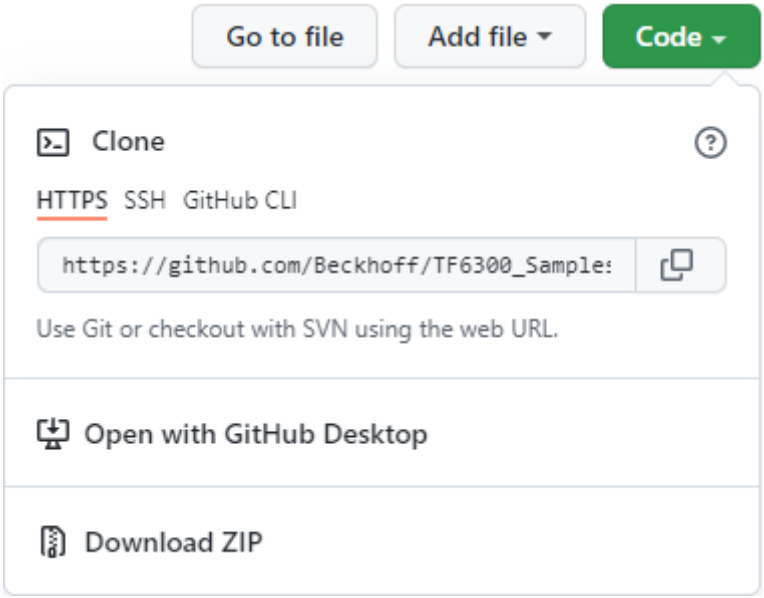
Name	Type	Description
AMSPORT_FTPADSSRV	UINT	The port of the FTP client ADS server.
FTPADS_IGR_CONNOPEN	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
FTPADS_IGR_RESOLVEDNS	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
FTPADS_IGR_CONNCLOSE	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
FTPADS_IGR_CONNCLOSEALL	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
FTPADS_IGR_CONNINFO	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
FTPADS_IGR_FILEUPLOAD	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
FTPADS_IGR_FILEDOWNLOAD	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
FTPADS_IGR_DIRREMOV	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
FTPADS_IGR_DIRCREATE	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
FTPADS_IGR_FILEEXIST	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
FTPADS_IGR_FILERENAME	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
FTPADS_IGR_FILEREMOVE	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
FTPADS_IGR_FILELIST	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
FTPADS_IGR_FILELISTEX	UDINT	The ADS server uses the index group to distinguish the commands coming in via ADS.
MAX_FTP_CONNECTIONS	UDINT	The maximum number of simultaneous FTP connections.
MAX_FILELIST_ITEMS	UDINT	The maximum number of elements contained in a Filelist.
DEFAULT_FTP_PORT	UDINT	The FTP default port

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

6 Samples

Sample code and configurations for this product can be obtained from the corresponding repository on GitHub: https://github.com/Beckhoff/TF6300_Samples. There you have the option to clone the repository or download a ZIP file containing the sample.



The following table also provides an overview of the samples currently available in the documentation:

Tc3_FTP:

TF6300_FTP_Samples | ... | POUs	Description
FB_SimpleUpload	This function block is described in Quick start [▶ 14] and contains sample logic for the entire process of uploading a file to an SFTP server with monitoring of the transfer progress and simple error handling.
FB_SimpleDownload	This function block is basically similar to FB_SimpleUpload, with the difference that a file is downloaded.
FB_ManualOperation	This function block allows you to manually execute individual FtpClient operations.

Tc2_FTP:

TF6300_FTP_Samples	Description
FileUpload	Uploading a file to an FTP server
FileDownload	Downloading a file from an FTP server
GetConnectionInfo	Reading connection information
FileRemove	Removing a file from an FTP server
ReadFileList	Reading a file list on an FTP server

7 Appendix

7.1 Error Codes

The Tc3_FTP library outputs extended error and diagnostic information based on the TwinCAT 3 EventLogger. Different event classes are output depending on the situation.

Online Watch:

fbConnection	FB_FtpConnection	
bError	BOOL	TRUE
ipResultMessage	I_TcMessage	16#FFFFBA0BDF077DB0
SUPER^	I_TcEventBase	
eSeverity	TCEVENTSEVERITY	Error
ipSourceInfo	I_TcSourceInfo	16#FFFFBA0BDF077BF0
nId	UDINT	139468800
sName	STRING((ParameterList.cSourceNameSize - 1))	'MAIN.fbManualOperation.fbConnection'
nEventId	UDINT	1101
sEventClassName	STRING(255)	'TcFtpClientEventClass'
sEventText	STRING(255)	'HostConnection Open'

The Tc3_FTP library returns one instance of an I_TcMessage at each function block. For ADS events there is the TcGeneralAdsEventClass [▶ 67] and for internal events there is the TcFtpClientEventClass [▶ 73]

The Tc2_FTP library returns simple return codes for ADS [▶ 67] and internal errors [▶ 71] as UDINT.

7.1.1 ADS Return Codes

Grouping of error codes:
Global error codes: ADS Return Codes [▶ 67]... (0x9811_0000 ...)
Router error codes: ADS Return Codes [▶ 68]... (0x9811_0500 ...)
General ADS errors: ADS Return Codes [▶ 68]... (0x9811_0700 ...)
RTime error codes: ADS Return Codes [▶ 70]... (0x9811_1000 ...)

Global error codes

Hex	Dec	HRESULT	Name	Description
0x0	0	0x98110000	ERR_NOERROR	No error.
0x1	1	0x98110001	ERR_INTERNAL	Internal error.
0x2	2	0x98110002	ERR_NORTIME	No real time.
0x3	3	0x98110003	ERR_ALLOCLOCKEDMEM	Allocation locked – memory error.
0x4	4	0x98110004	ERR_INSERTMAILBOX	Mailbox full – the ADS message could not be sent. Reducing the number of ADS messages per cycle will help.
0x5	5	0x98110005	ERR_WRONGRECEIVEHMSG	Wrong HMSG.
0x6	6	0x98110006	ERR_TARGETPORTNOTFOUND	Target port not found – ADS server is not started, not reachable or not installed.
0x7	7	0x98110007	ERR_TARGETMACHINENOTFOUND	Target computer not found – AMS route was not found.
0x8	8	0x98110008	ERR_UNKNOWNCMDID	Unknown command ID.
0x9	9	0x98110009	ERR_BADTASKID	Invalid task ID.
0xA	10	0x9811000A	ERR_NOIO	No IO.
0xB	11	0x9811000B	ERR_UNKNOWNAMSCMD	Unknown AMS command.
0xC	12	0x9811000C	ERR_WIN32ERROR	Win32 error.
0xD	13	0x9811000D	ERR_PORTNOTCONNECTED	Port not connected.
0xE	14	0x9811000E	ERR_INVALIDAMSLENGTH	Invalid AMS length.
0xF	15	0x9811000F	ERR_INVALIDAMSNETID	Invalid AMS Net ID.
0x10	16	0x98110010	ERR_LOWINSTLEVEL	Installation level is too low –TwinCAT 2 license error.
0x11	17	0x98110011	ERR_NODEBUGINTAVAILABLE	No debugging available.
0x12	18	0x98110012	ERR_PORTDISABLED	Port disabled – TwinCAT system service not started.
0x13	19	0x98110013	ERR_PORTALREADYCONNECTED	Port already connected.
0x14	20	0x98110014	ERR_AMSSYNC_W32ERROR	AMS Sync Win32 error.
0x15	21	0x98110015	ERR_AMSSYNC_TIMEOUT	AMS Sync Timeout.
0x16	22	0x98110016	ERR_AMSSYNC_AMSERROR	AMS Sync error.
0x17	23	0x98110017	ERR_AMSSYNC_NOINDEXINMAP	No index map for AMS Sync available.
0x18	24	0x98110018	ERR_INVALIDAMSPORT	Invalid AMS port.
0x19	25	0x98110019	ERR_NOMEMORY	No memory.
0x1A	26	0x9811001A	ERR_TCPSEND	TCP send error.
0x1B	27	0x9811001B	ERR_HOSTUNREACHABLE	Host unreachable.
0x1C	28	0x9811001C	ERR_INVALIDAMSFRAGMENT	Invalid AMS fragment.
0x1D	29	0x9811001D	ERR_TLSSEND	TLS send error – secure ADS connection failed.
0x1E	30	0x9811001E	ERR_ACCESSDENIED	Access denied – secure ADS access denied.

Router error codes

Hex	Dec	HRESULT	Name	Description
0x500	1280	0x98110500	ROUTERERR_NOLOCKEDMEMORY	Locked memory cannot be allocated.
0x501	1281	0x98110501	ROUTERERR_RESIZEMEMORY	The router memory size could not be changed.
0x502	1282	0x98110502	ROUTERERR_MAILBOXFULL	The mailbox has reached the maximum number of possible messages.
0x503	1283	0x98110503	ROUTERERR_DEBUGBOXFULL	The Debug mailbox has reached the maximum number of possible messages.
0x504	1284	0x98110504	ROUTERERR_UNKNOWNPORTTYPE	The port type is unknown.
0x505	1285	0x98110505	ROUTERERR_NOTINITIALIZED	The router is not initialized.
0x506	1286	0x98110506	ROUTERERR_PORTALREADYINUSE	The port number is already assigned.
0x507	1287	0x98110507	ROUTERERR_NOTREGISTERED	The port is not registered.
0x508	1288	0x98110508	ROUTERERR_NOMOREQUEUES	The maximum number of ports has been reached.
0x509	1289	0x98110509	ROUTERERR_INVALIDPORT	The port is invalid.
0x50A	1290	0x9811050A	ROUTERERR_NOTACTIVATED	The router is not active.
0x50B	1291	0x9811050B	ROUTERERR_FRAGMENTBOXFULL	The mailbox has reached the maximum number for fragmented messages.
0x50C	1292	0x9811050C	ROUTERERR_FRAGMENTTIMEOUT	A fragment timeout has occurred.
0x50D	1293	0x9811050D	ROUTERERR_TOBEREMOVED	The port is removed.

General ADS error codes

Hex	Dec	HRESULT	Name	Description
0x700	1792	0x98110700	ADSERR_DEVICE_ERROR	General device error.
0x701	1793	0x98110701	ADSERR_DEVICE_SRVNOTSUPP	Service is not supported by the server.
0x702	1794	0x98110702	ADSERR_DEVICE_INVALIDGRP	Invalid index group.
0x703	1795	0x98110703	ADSERR_DEVICE_INVALIDOFFSET	Invalid index offset.
0x704	1796	0x98110704	ADSERR_DEVICE_INVALIDACCESS	Reading or writing not permitted. Several causes are possible. For example, an incorrect password was entered when creating routes.
0x705	1797	0x98110705	ADSERR_DEVICE_INVALIDSIZE	Parameter size not correct.
0x706	1798	0x98110706	ADSERR_DEVICE_INVALIDDATA	Invalid data values.
0x707	1799	0x98110707	ADSERR_DEVICE_NOTREADY	Device is not ready to operate.
0x708	1800	0x98110708	ADSERR_DEVICE_BUSY	Device is busy.
0x709	1801	0x98110709	ADSERR_DEVICE_INVALIDCONTEXT	Invalid operating system context. This can result from use of ADS blocks in different tasks. It may be possible to resolve this through multitasking synchronization in the PLC.
0x70A	1802	0x9811070A	ADSERR_DEVICE_NOMEMORY	Insufficient memory.
0x70B	1803	0x9811070B	ADSERR_DEVICE_INVALIDPARM	Invalid parameter values.
0x70C	1804	0x9811070C	ADSERR_DEVICE_NOTFOUND	Not found (files, ...).
0x70D	1805	0x9811070D	ADSERR_DEVICE_SYNTAX	Syntax error in file or command.
0x70E	1806	0x9811070E	ADSERR_DEVICE_INCOMPATIBLE	Objects do not match.
0x70F	1807	0x9811070F	ADSERR_DEVICE_EXISTS	Object already exists.
0x710	1808	0x98110710	ADSERR_DEVICE_SYMBOLNOTFOUND	Symbol not found.
0x711	1809	0x98110711	ADSERR_DEVICE_SYMBOLVERSIONINVALID	Invalid symbol version. This can occur due to an online change. Create a new handle.
0x712	1810	0x98110712	ADSERR_DEVICE_INVALIDSTATE	Device (server) is in invalid state.
0x713	1811	0x98110713	ADSERR_DEVICE_TRANSMODENOTSUPP	AdsTransMode not supported.
0x714	1812	0x98110714	ADSERR_DEVICE_NOTIFYHNDINVALID	Notification handle is invalid.
0x715	1813	0x98110715	ADSERR_DEVICE_CLIENTUNKNOWN	Notification client not registered.
0x716	1814	0x98110716	ADSERR_DEVICE_NOMOREHDLS	No further handle available.
0x717	1815	0x98110717	ADSERR_DEVICE_INVALIDWATCHSIZE	Notification size too large.
0x718	1816	0x98110718	ADSERR_DEVICE_NOTINIT	Device not initialized.
0x719	1817	0x98110719	ADSERR_DEVICE_TIMEOUT	Device has a timeout.
0x71A	1818	0x9811071A	ADSERR_DEVICE_NOINTERFACE	Interface query failed.
0x71B	1819	0x9811071B	ADSERR_DEVICE_INVALIDINTERFACE	Wrong interface requested.
0x71C	1820	0x9811071C	ADSERR_DEVICE_INVALIDCLSID	Class ID is invalid.
0x71D	1821	0x9811071D	ADSERR_DEVICE_INVALIDOBJID	Object ID is invalid.
0x71E	1822	0x9811071E	ADSERR_DEVICE_PENDING	Request pending.
0x71F	1823	0x9811071F	ADSERR_DEVICE_ABORTED	Request is aborted.
0x720	1824	0x98110720	ADSERR_DEVICE_WARNING	Signal warning.
0x721	1825	0x98110721	ADSERR_DEVICE_INVALIDARRAYIDX	Invalid array index.
0x722	1826	0x98110722	ADSERR_DEVICE_SYMBOLNOTACTIVE	Symbol not active.
0x723	1827	0x98110723	ADSERR_DEVICE_ACCESSDENIED	Access denied. Several causes are possible. For example, a unidirectional ADS route is used in the opposite direction.
0x724	1828	0x98110724	ADSERR_DEVICE_LICENSENOTFOUND	Missing license.
0x725	1829	0x98110725	ADSERR_DEVICE_LICENSEEXPIRED	License expired.
0x726	1830	0x98110726	ADSERR_DEVICE_LICENSEEXCEEDED	License exceeded.
0x727	1831	0x98110727	ADSERR_DEVICE_LICENSEINVALID	Invalid license.
0x728	1832	0x98110728	ADSERR_DEVICE_LICENSESYSTEMID	License problem: System ID is invalid.
0x729	1833	0x98110729	ADSERR_DEVICE_LICENSENOTIMELIMIT	License not limited in time.
0x72A	1834	0x9811072A	ADSERR_DEVICE_LICENSEFUTUREISSUE	Licensing problem: time in the future.
0x72B	1835	0x9811072B	ADSERR_DEVICE_LICENSETIMETOLONG	License period too long.
0x72C	1836	0x9811072C	ADSERR_DEVICE_EXCEPTION	Exception at system startup.
0x72D	1837	0x9811072D	ADSERR_DEVICE_LICENSEDUPLICATED	License file read twice.
0x72E	1838	0x9811072E	ADSERR_DEVICE_SIGNATUREINVALID	Invalid signature.
0x72F	1839	0x9811072F	ADSERR_DEVICE_CERTIFICATEINVALID	Invalid certificate.
0x730	1840	0x98110730	ADSERR_DEVICE_LICENSEOEMNOTFOUND	Public key not known from OEM.
0x731	1841	0x98110731	ADSERR_DEVICE_LICENSERESTRICTED	License not valid for this system ID.

Hex	Dec	HRESULT	Name	Description
0x732	1842	0x98110732	ADSERR_DEVICE_LICENSEDEMODENIED	Demo license prohibited.
0x733	1843	0x98110733	ADSERR_DEVICE_INVALIDFNCID	Invalid function ID.
0x734	1844	0x98110734	ADSERR_DEVICE_OUTOFRANGE	Outside the valid range.
0x735	1845	0x98110735	ADSERR_DEVICE_INVALIDALIGNMENT	Invalid alignment.
0x736	1846	0x98110736	ADSERR_DEVICE_LICENSEPLATFORM	Invalid platform level.
0x737	1847	0x98110737	ADSERR_DEVICE_FORWARD_PL	Context – forward to passive level.
0x738	1848	0x98110738	ADSERR_DEVICE_FORWARD_DL	Context – forward to dispatch level.
0x739	1849	0x98110739	ADSERR_DEVICE_FORWARD_RT	Context – forward to real-time.
0x740	1856	0x98110740	ADSERR_CLIENT_ERROR	Client error.
0x741	1857	0x98110741	ADSERR_CLIENT_INVALIDPARM	Service contains an invalid parameter.
0x742	1858	0x98110742	ADSERR_CLIENT_LISTEMPTY	Polling list is empty.
0x743	1859	0x98110743	ADSERR_CLIENT_VARUSED	Var connection already in use.
0x744	1860	0x98110744	ADSERR_CLIENT_DUPLINVOKEID	The called ID is already in use.
0x745	1861	0x98110745	ADSERR_CLIENT_SYNCTIMEOUT	Timeout has occurred – the remote terminal is not responding in the specified ADS timeout. The route setting of the remote terminal may be configured incorrectly.
0x746	1862	0x98110746	ADSERR_CLIENT_W32ERROR	Error in Win32 subsystem.
0x747	1863	0x98110747	ADSERR_CLIENT_TIMEOUTINVALID	Invalid client timeout value.
0x748	1864	0x98110748	ADSERR_CLIENT_PORTNOTOPEN	Port not open.
0x749	1865	0x98110749	ADSERR_CLIENT_NOAMSADDR	No AMS address.
0x750	1872	0x98110750	ADSERR_CLIENT_SYNCINTERNAL	Internal error in Ads sync.
0x751	1873	0x98110751	ADSERR_CLIENT_ADDHASH	Hash table overflow.
0x752	1874	0x98110752	ADSERR_CLIENT_REMOVEHASH	Key not found in the table.
0x753	1875	0x98110753	ADSERR_CLIENT_NOMORESYM	No symbols in the cache.
0x754	1876	0x98110754	ADSERR_CLIENT_SYNCRESINVALID	Invalid response received.
0x755	1877	0x98110755	ADSERR_CLIENT_SYNCPORTLOCKED	Sync Port is locked.
0x756	1878	0x98110756	ADSERR_CLIENT_REQUESTCANCELLED	The request was canceled.

RTime error codes

Hex	Dec	HRESULT	Name	Description
0x1000	4096	0x98111000	RTERR_INTERNAL	Internal error in the real-time system.
0x1001	4097	0x98111001	RTERR_BADTIMERPERIODS	Timer value is not valid.
0x1002	4098	0x98111002	RTERR_INVALIDTASKPTR	Task pointer has the invalid value 0 (zero).
0x1003	4099	0x98111003	RTERR_INVALIDSTACKPTR	Stack pointer has the invalid value 0 (zero).
0x1004	4100	0x98111004	RTERR_PRIOEXISTS	The request task priority is already assigned.
0x1005	4101	0x98111005	RTERR_NOMORETCB	No free TCB (Task Control Block) available. The maximum number of TCBs is 64.
0x1006	4102	0x98111006	RTERR_NOMORESEMAS	No free semaphores available. The maximum number of semaphores is 64.
0x1007	4103	0x98111007	RTERR_NOMOREQUEUES	No free space available in the queue. The maximum number of positions in the queue is 64.
0x100D	4109	0x9811100D	RTERR_EXTIRQALREADYDEF	An external synchronization interrupt is already applied.
0x100E	4110	0x9811100E	RTERR_EXTIRQNOTDEF	No external sync interrupt applied.
0x100F	4111	0x9811100F	RTERR_EXTIRQINSTALLFAILED	Application of the external synchronization interrupt has failed.
0x1010	4112	0x98111010	RTERR_IRQLNOTLESSOREQUAL	Call of a service function in the wrong context
0x1017	4119	0x98111017	RTERR_VMXNOTSUPPORTED	Intel VT-x extension is not supported.
0x1018	4120	0x98111018	RTERR_VMXDISABLED	Intel VT-x extension is not enabled in the BIOS.
0x1019	4121	0x98111019	RTERR_VMXCONTROLSMISSING	Missing function in Intel VT-x extension.
0x101A	4122	0x9811101A	RTERR_VMXENABLEFAILS	Activation of Intel VT-x fails.

Specific positive HRESULT Return Codes:

HRESULT	Name	Description
0x0000_0000	S_OK	No error.
0x0000_0001	S_FALSE	No error. Example: successful processing, but with a negative or incomplete result.
0x0000_0203	S_PENDING	No error. Example: successful processing, but no result is available yet.
0x0000_0256	S_WATCHDOG_TIMEOUT	No error. Example: successful processing, but a timeout occurred.

TCP Winsock error codes

Hex	Dec	Name	Description
0x274C	10060	WSAETIMEDOUT	A connection timeout has occurred - error while establishing the connection, because the remote terminal did not respond properly after a certain period of time, or the established connection could not be maintained because the connected host did not respond.
0x274D	10061	WSAECONNREFUSED	Connection refused - no connection could be established because the target computer has explicitly rejected it. This error usually results from an attempt to connect to a service that is inactive on the external host, that is, a service for which no server application is running.
0x2751	10065	WSAEHOSTUNREACH	No route to host - a socket operation referred to an unavailable host.
More Winsock error codes: Win32 error codes			

7.1.2 Tc2_FTP: FTP Client Return Codes

The return codes for the Tc2_FTP library are listed below:

Hex	Dec	Description
0x00008001	32768 + 1 := 32769	Internal error TwinCAT FTP Client.
0x00008002	32768 + 2 := 32770	File error (e.g. file not found, access denied)
0x00008003	32768 + 3 := 32771	Transmission error (e.g. connection interrupted)
0x00008004	32768 + 4 := 32772	Error while connecting to the server. Connection could not be established.
0x00008005	32768 + 5 := 32773	No response received from the FTP server.
0x000081F4	32768 + 500 := 33268	Syntax error, command unknown. This may contain the error "Command line too long".
0x000081F5	32768 + 501 := 33269	Syntax error with parameters or arguments.
0x000081F6	32768 + 502 := 33270	Command not implemented.
0x000081F7	32768 + 503 := 33271	Incorrect sequence of commands.
0x000081F8	32768 + 504 := 33272	Command is not implemented for this parameter.
0x00008212	32768 + 530 := 33298	Not logged in.
0x00008214	32768 + 532 := 33300	Requires an account to save files.
0x00008226	32768 + 550 := 33318	Requested action not executed. File not accessible (e.g. file not found, no access).
0x00008227	32768 + 551 := 33319	Requested action aborted, page type unknown.
0x00008228	32768 + 552 := 33320	Requested file action aborted. Allocated memory exceeded (for current directory or file).
0x00008229	32768 + 553 := 33321	Requested action not executed. Filename not allowed.
0x000081A5	32768 + 421 := 33189	Service not available, closes control connections. This can be a response to any command if the service knows that it needs to shut down.
0x000081A9	32768 + 425 := 33193	Cannot open a data connection.
0x000081AA	32768 + 426 := 33194	Connection closed; transfer aborted.
0x000081C2	32768 + 450 := 33218	Requested file action not executed.
0x000081C3	32768 + 451 := 33219	Requested action aborted. Local error during processing.
0x000081C4	32768 + 452 := 33220	Requested action not executed. Insufficient storage space in the system. File not accessible (e.g. file in use).

Requirements

Development Environment	Target Platform	PLC Libraries to include
TwinCAT v3.0 Build 3102	PC or CX (x86)	Tc2_FTP

7.1.3 Tc3_FTP: TcFtpClientEventClass

These are the registered events of the TcFtpClientEventClass:

nEventId (dec)	sEventText
0	None
1000	NoError SUCCESS
1001	Connect Failure
1002	Connection Closed
1003	Socket Error
1004	Nameresolution Failure
1005	Pending
1006	Protocol Error
1007	ProxyName Resolution Failure
1008	Receive Failure
1009	Operation Aborted
1010	Unclassifiable Error
1011	Send Failure
1012	Server Protocol Violation
1013	Timeout
1014	Async Error
1015	Operation Failure
1016	Out Of Bounds
1100	Any Connection Open
1101	Host Connection Open
1102	File Location Error
1103	File Content Error
1104	Not Configured
1105	Parameter Null Error
1106	Parameters Improper
1107	Authentication Error
1108	Not Connected
1109	Unknown Action
1110	Busy Try Again
1111	Setup Fail
1112	Not Initialized
1255	Undefined Error

More Information:
www.beckhoff.com/tf6300

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com