

BECKHOFF New Automation Technology

Manual | EN

TE1000

TwinCAT 3 | PLC Library: Tc2_Coupler

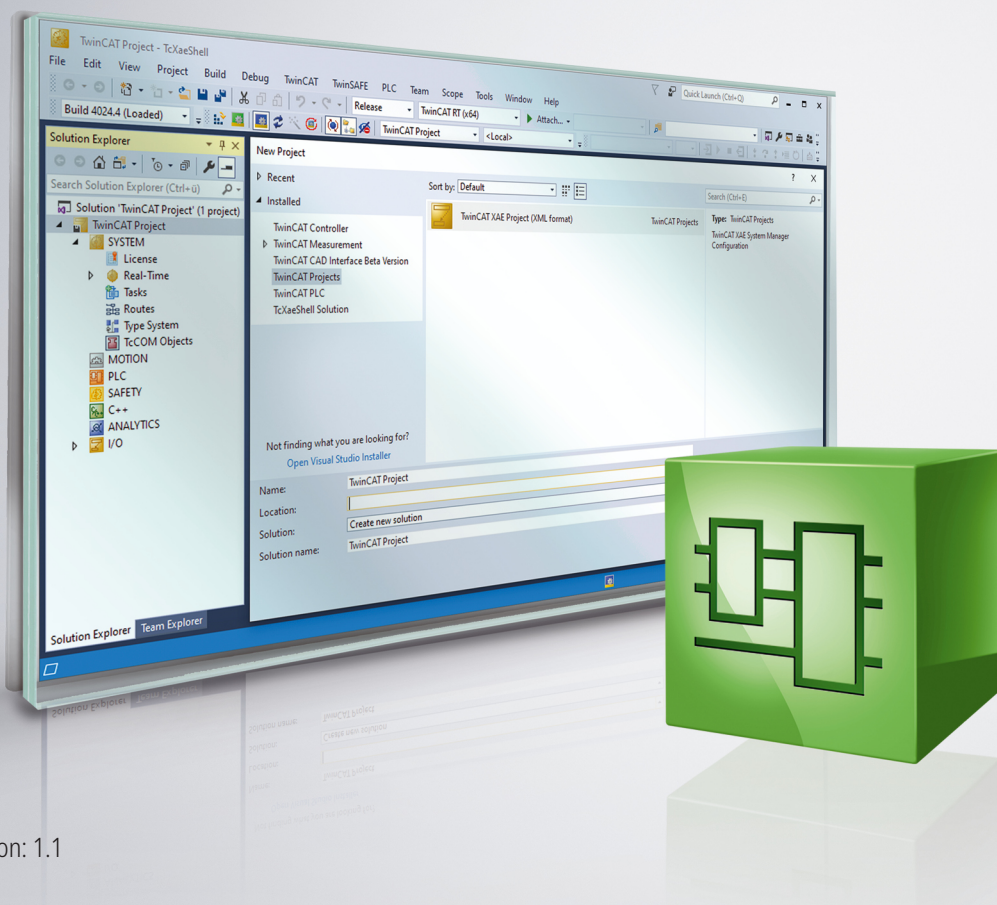


Table of contents

1 Foreword	5
1.1 Notes on the documentation.....	5
1.2 Safety instructions	6
2 Overview	7
3 Function blocks	8
3.1 ReadWriteTerminalReg	8
3.2 CouplerReset.....	10
3.3 FB_ReadCouplerDiag	11
3.4 FB_ReadCouplerRegs	12
3.5 FB_WriteCouplerRegs.....	14
4 [obsolete functions]	16
4.1 F_GetVersionTcPlcCoupler	16
5 Data types	17
5.1 PLCINTFSTRUCT	17
5.2 E_CouplerErrType.....	17
5.3 ST_CouplerDiag	17
5.4 ST_CouplerReg.....	17
5.5 ST_CouplerTable	18
5.6 ST_FlashCode.....	18
6 Global Constants	19
6.1 Library version	19

1 Foreword

1.1 Notes on the documentation

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

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

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

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

Disclaimer

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

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

Trademarks

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

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

Patent Pending

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

EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
with corresponding applications or registrations in various other countries.

EtherCAT®

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

Copyright

© Beckhoff Automation GmbH & Co. KG, Germany.

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

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

1.2 Safety instructions

Safety regulations

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

Exclusion of liability

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

Personnel qualification

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

Description of symbols

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

DANGER

Serious risk of injury!

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

WARNING

Risk of injury!

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

CAUTION

Personal injuries!

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

NOTE

Damage to the environment or devices

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



Tip or pointer

This symbol indicates information that contributes to better understanding.

2 Overview



The Tc2_Coupler library can be used for the following products: BKxxx couplers, KLxxx terminals, KSxxxx terminals and KMxxx modules.

The Tc2_Coupler library contains function blocks that provide convenient access to registers in the terminals via the terminal's control/status byte (register communication) and for communication with the Beckhoff couplers via the 2-byte PLC interface. The function blocks can, for instance, be used for parameterization of the terminals by way of the fieldbus.

Only the intelligent terminals have a register structure. The intelligent terminals include, for example, all the analog input and output terminals. The terminal's status/control byte is only visible in the process image if the terminal has been mapped as a complex terminal. Each terminal channel has its own register structure with a maximum of 64 registers. Under a compact mapping the control/status bytes are not visible in the process image.

For register access via the 2-byte PLC interface it is also necessary for the status and control word variables of the PLC interface to be mapped into the process image. In some fieldbuses (Lightbus, Profibus) this can be configured for the particular coupler in the TwinCAT System Manager, but in others (e.g. Interbus S) special configuration software is required for the job (e.g. KS2000). The status and control variables are linked to the function block's corresponding input and output variables.

If any changes made to the registers are to be stored permanently, the power supply to the coupler must be interrupted.

NOTE

No cyclic access!

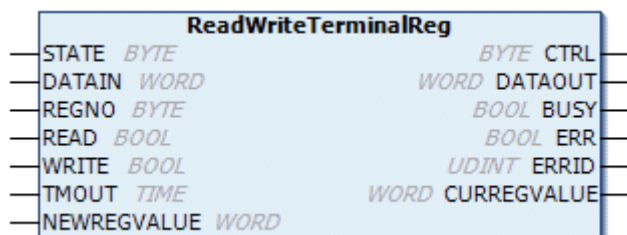
When called, the function blocks of the library carry out write/read access to the registers in the terminals or in the couplers. If they are called cyclically, the EEPROM memory may be destroyed. The function blocks were basically developed to facilitate configuration of the terminals/couplers or fault diagnosis from within the PLC program.

Contents of the Library

Name	Description
ReadWriteTerminalReg [► 8]	Access registers in the terminal via the terminal's control/status byte (register communication)
CouplerReset [► 10]	Reset the coupler via the 2-byte PLC interface
FB_ReadCouplerDiag [► 11]	Read coupler diagnosis (flash code)
FB_ReadCouplerRegs [► 12]	Read coupler registers
FB_WriteCouplerRegs [► 14]	Write coupler registers
F_GetVersionTcPlcCoupler [► 16]	Returns library version info

3 Function blocks

3.1 ReadWriteTerminalReg



The ReadWriteTerminalReg function block permits convenient access to the registers of the terminal via the terminal channel's status/control byte (register communication). In the standard operating mode, the data inputs and outputs of the intelligent terminal (e.g. an analog output terminal) are used to exchange the analog output data. A handshake via the status/control byte permits register access. The data input and output variables are used here to transfer the register values. A rising edge at the READ or WRITE input causes the register with number REGNO to be read or written to. Write protection of the register is disabled by the function block for a write access and enabled once more afterwards. When a register is written to, the new register value is first written to and then read, the read value is available at the CURREGVALUE output. If changes made to the register values are to be stored permanently, the power supply to the coupler must be interrupted. The variables STATE, DATAIN, CTRL and DATAOUT must be linked in the TwinCAT System Manager to the corresponding I/O variables in the terminal channel.

VAR_INPUT

```
VAR_INPUT
  STATE      : BYTE;
  DATAIN   : WORD;
  REGNO     : BYTE;
  READ      : BOOL;
  WRITE     : BOOL;
  TMOUT     : TIME;
  NEWREGVALUE : WORD;
END_VAR
```

STATE: Terminal channel status byte.

DATAIN: Terminal channel data input word.

REGNO: Number of the register that is to be written to or read.

READ: A rising edge at this input activates the block, and the current register value is read. If successful, the register value is available in the output variable CURREGVALUE.

WRITE: A rising edge at this input activates the block, and the value in the input variable NEWREGVALUE is written into the register REGNO. After this, the current value of the register is read, and, if successful, is made available in the output variable CURREGVALUE.

TMOUT: States the length of the timeout that may not be exceeded during execution of the function.

NEWREGVALUE: Data word that is to be written into the register with number REGNO by a write access.

VAR_OUTPUT

```
VAR_OUTPUT
  CTRL      : BYTE;
  DATAOUT  : WORD;
  BUSY      : BOOL;
  ERR       : BOOL;
  ERRID     : UDINT;
  CURREGVALUE : WORD;
END_VAR
```


CTRL: Terminal channel control byte.

DATAOUT: Terminal channel data output word.

BUSY: This output is set when the block is activated, and remains set until execution of the function has been completed.

ERR: If an error should occur during the execution of the function, then this output is set, after the BUSY output has been reset.

ERRID: Supplies the error number when the ERR output is set.

Error number	Error descriptions
0	No error
0x100	Timeout error. The time permitted for execution has been exceeded.
0x200	Parameter error (e.g. an invalid register number).
0x300	The read value differs from the written value (writing not allowed)

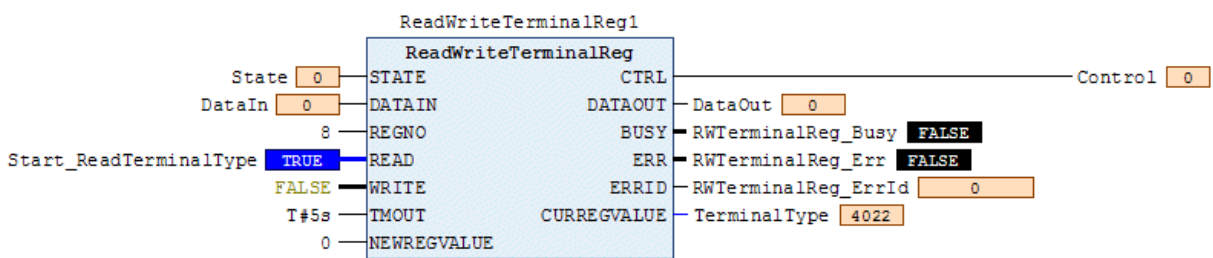
CURREGVALUE: This variable provides the current register value after a successful read or write access.

Examples of calls in FBD:

Example 1

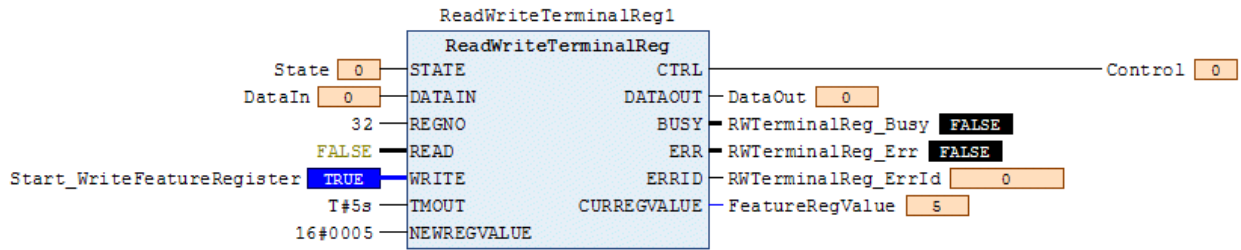
```

VAR
  ReadWriteTerminalReg1      : ReadWriteTerminalReg;
  State AT%I*                : BYTE;
  Control AT%Q*              : BYTE;
  DataIn AT%I*               : WORD;
  DataOut AT%Q*              : WORD;
  Start_ReadTerminalType     : BOOL;
  Start_WriteFeatureRegister : BOOL;
  RWTerminalReg_Busy         : BOOL;
  RWTerminalReg_Err          : BOOL;
  RWTerminalReg_ErrId        : UDINT;
  TerminalType               : WORD;
  FeatureRegValue            : WORD;
END_VAR
  
```



In Example 1 the terminal identifier is read from register 8 of an analog output terminal. The variables *State*, *Control*, *DataIn* and *DataOut* are linked to the terminal's corresponding I/O variables in the TwinCAT System Manager. The terminal identifier is KL4022..

Example 2



In Example 2 the user-scaling is activated in the feature register (register 32) of a KL4022 analog output terminal. The new value in the feature register is then read by the function block, and can be checked through the output variable **CURREGVALUE**.

Requirements

Development environment	Target system type	PLC libraries to include (category group)
TwinCAT v3.1.0	PC or CX (x86, x64, ARM)	Tc2_Coupler (IO)

3.2 CouplerReset



The CouplerReset function block can be used to execute a reset of the coupler via the 2-byte PLC interface. In a coupler reset the current terminal configuration, for example, is read in again by the coupler via the K-bus (the terminal bus), and communication on the K-bus is reinitialized. Existing K-bus error messages for the coupler are reset. The **STATE** and **CONTROL** variables are used to perform a handshake with the coupler while the function block is being executed. These variables must therefore be linked with the status/control I/O variables of the 2-byte PLC interface in the TwinCAT System Manager.

VAR_INPUT

```
VAR_INPUT
    STATE    : PLCINTFSTRUCT;
    START    : BOOL;
    TMOU     : TIME;
END_VAR
```

STATE: Status word of the 2-byte PLC interface. ([PLCINTFSTRUCT \[► 17\]](#))

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

TMOU: States the length of the timeout that may not be exceeded during execution of the function.

VAR_OUTPUT

```
VAR_OUTPUT
    CONTROL  : PLCINTFSTRUCT;
    BUSY     : BOOL;
    ERR      : BOOL;
    ERRID    : UDINT;
END_VAR
```

CONTROL: Control word of the 2-byte PLC interface ([PLCINTFSTRUCT \[► 17\]](#)).

BUSY: This output is set when the block is activated, and remains set until execution of the function has been completed.

ERR: If an error should occur during the execution of the function, then this output is set, after the BUSY output has been reset.

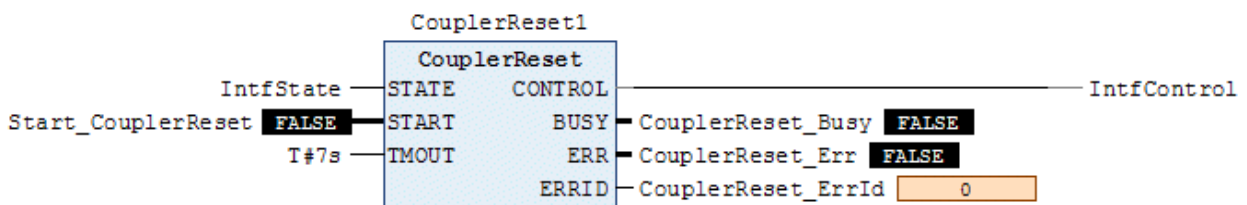
ERRID: Supplies the error number when the ERR output is set.

Error number	Error description
0	no error
0x100	Error at initialisation of the communication via the 2 byte PLC interface
0x200	error during communication
0x300	Timeout-Error. The permitted execution time was exceeded
0x400	Wrong parameter value at register number
0x500	Wrong parameter value at table number

Example of a call in FBD:

```

VAR
  IntfState AT%I*      : PLCINTFSTRUCT;
  IntfControl AT%Q*    : PLCINTFSTRUCT;
  CouplerReset1       : CouplerReset;
  Start_CouplerReset  : BOOL;
  CouplerReset_Busy   : BOOL;
  CouplerReset_Err    : BOOL;
  CouplerReset_ErrId  : UDINT;
END_VAR
    
```

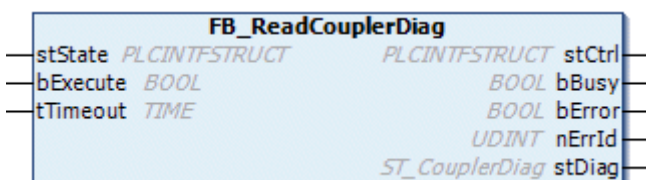


The variables *IntfState* and *IntfControl* are linked with corresponding I/O variables in the TwinCAT System Manager.

Requirements

Development environment	Target system type	PLC libraries to include (category group)
TwinCAT v3.1.0	PC or CX (x86, x64, ARM)	Tc2_Coupler (IO)

3.3 FB_ReadCouplerDiag



The *FB_ReadCouplerDiag* function block allows reading of the first and second flashing sequences of the error LED on the coupler when a terminal bus or coupler error occurs. The data is transferred to the PLC via the 2-byte PLC interface. This, however, only functions if communication over the fieldbus is maintained. It must be possible for the data to be transferred without error from the coupler to the PLC via the fieldbus. In order to detect that a coupler error has occurred, the status byte for the coupler in the PLC can be interrogated cyclically, and the function block activated when an error occurs.

VAR_INPUT

```
VAR_INPUT
    stState      : PLCINTFSTRUCT;
    bExecute     : BOOL;
    tTimeout     : TIME;
END_VAR
```

stState : Status word of the 2-byte PLC interface. ([PLCINTFSTRUCT \[P 17\]](#))

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

tTimeout: States the length of the timeout that may not be exceeded during execution of the function.

VAR_OUTPUT

```
VAR_OUTPUT
    stCtrl       : PLCINTFSTRUCT;
    bBusy        : BOOL;
    bError       : BOOL;
    nErrId       : UDINT;
    stDiag       : ST_CouplerDiag;
END_VAR
```

stCtrl : Control word of the 2-byte PLC interface. ([PLCINTFSTRUCT \[P 17\]](#))

bBusy: This output is set when the block is activated, and remains set until execution of the function has been completed.

bError: If an error should occur during the execution of the function, then this output is set, after the bBusy output has been reset.

nErrId : Supplies the error number when the bError output is set.

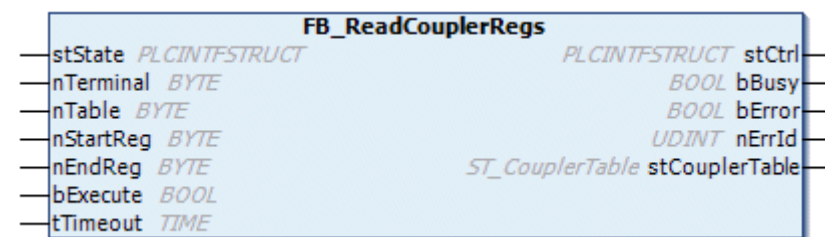
Error number	Error description
0	No error
0x100	Error at initialisation of the communication via the 2 byte PLC interface
0x200	Error during communication
0x300	Timeout-Error. The permitted execution time was exceeded
0x400	Wrong parameter value at register number
0x500	Wrong parameter value at table number

stDiag : Structure containing the coupler's diagnostic information (error type, and the first and second coupler flashing sequences). ([ST_CouplerDiag \[P 17\]](#))

Requirements

Development environment	Target system type	PLC libraries to include (category group)
TwinCAT v3.1.0	PC or CX (x86, x64, ARM)	Tc2_Coupler (IO)

3.4 FB_ReadCouplerRegs



This function block provides read access to the table register in the coupler and the registers of the intelligent terminals. The coupler itself is referred to as terminal 0 (null). All the other terminals in the terminal block, with the exception of passive terminals (such as power feed terminals), are counted in ascending sequence (beginning with 1). It is possible either to read all registers, or only a partial region (between *nStartReg* and *nEndReg*). Several seconds are required in order to read all the registers (0...255) in a table. Register values that have been successfully read are found in the structure *stCouplerTable*. The structure is an array of high and low bytes. Each array element corresponds to a register value (e.g.: `stCouplerTable[5] == Register 5`).

VAR_INPUT

```
VAR_INPUT
  stState      : PLCINTFSTRUCT;
  nTerminal    : BYTE := TERM_COUPLER;
  nTable       : BYTE;
  nStartReg    : BYTE;
  nEndReg      : BYTE;
  bExecute     : BOOL;
  tTimeout     : TIME;
END_VAR
```

stState : Status word of the 2-byte PLC interface. ([PLCINTFSTRUCT \[► 17\]](#))

nTerminal : Terminal number, to whose table register access is to be made. The coupler has terminal number null. Passive terminals are not to be counted.

nTable : Table number whose register values are to be read. Intelligent terminals only have one table for each terminal channel. A 4-channel terminal has the following table numbers: 0-3. An intelligent terminal, however, only possesses a maximum of 64 register values for each terminal channel.

nStartReg : The number of the first register that is to be read.

nEndReg : The number of the last register that is to be read.

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

tTimeout: States the length of the timeout that may not be exceeded during execution of the function.

VAR_OUTPUT

```
VAR_OUTPUT
  stCtrl       : PLCINTFSTRUCT;
  bBusy        : BOOL;
  bError       : BOOL;
  nErrId       : UDINT;
  stCouplerTable: ST_CouplerTable;
END_VAR
```

stCtrl : Control word of the 2-byte PLC interface. ([PLCINTFSTRUCT \[► 17\]](#))

bBusy: This output is set when the block is activated, and remains set until execution of the function has been completed.

bError: If an error should occur during the execution of the function, then this output is set, after the **bBusy** output has been reset.

nErrId : Supplies the error number when the **bError** output is set.

Error number	Error description
0	No error
0x100	Error at initialisation of the communication via the 2 byte PLC interface
0x200	Error during communication
0x300	Timeout-Error. The permitted execution time was exceeded
0x400	Wrong parameter value at register number
0x500	Wrong parameter value at table number

stCouplerTable : Structure containing the register values of the terminal or coupler that have been read (type: [ST_CouplerTable \[► 18\]](#)).

Requirements

Development environment	Target system type	PLC libraries to include (category group)
TwinCAT v3.1.0	PC or CX (x86, x64, ARM)	Tc2_Coupler (IO)

3.5 FB_WriteCouplerRegs



This function block provides write access to the table register in the coupler and the registers of the intelligent terminals. The coupler itself is referred to as terminal 0 (null). All the other terminals in the terminal block, with the exception of passive terminals (such as power feed terminals), are counted in ascending sequence (beginning with 1). It is possible either to write all registers, or only a partial region (between `nStartReg` and `nEndReg`). Several seconds are required in order to write all the registers (0..255) of a table of the coupler. The register values to be written are located in the structure `stCouplerTable`. The structure is an array of high and low bytes. Each array element corresponds to a register value (e.g.: `stCouplerTable[5] == Register 5`).

VAR_INPUT

```

VAR_INPUT
    stState      : PLCINTFSTRUCT;
    nTerminal    : BYTE := TERM_COUPLER;
    nTable       : BYTE;
    nStartReg    : BYTE;
    nEndReg      : BYTE;
    bExecute     : BOOL;
    stCouplerTable : ST_CouplerTable;
    tTimeout     : TIME;
END_VAR

```

stState : Status word of the 2-byte PLC interface. ([PLCINTFSTRUCT](#) [► 17])

nTerminal : Terminal number, to whose table register access is to be made. The coupler has terminal number null. Passive terminals are not to be counted.

nTable : Table number whose register values are to be written. Intelligent terminals only have one table for each terminal channel. A 4-channel terminal has the following table numbers: 0-3. An intelligent terminal, however, only possesses a maximum of 64 register values for each terminal channel!

nStartReg : The number of the first register that is to be written.

nEndReg : The number of the last register that is to be written.

stCouplerTable : Register value array to be written ([ST_CouplerTable](#) [► 18]).

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

tTimeout: States the length of the timeout that may not be exceeded during execution of the function.

VAR_OUTPUT

```

VAR_OUTPUT
    stCtrl      : PLCINTFSTRUCT;
    bBusy       : BOOL;

```

```

    bError    : BOOL;
    nErrId    : UDINT;
END_VAR

```

stCtrl : Control word of the 2-byte PLC interface. (PLCINTFSTRUCT [▶ 17])

bBusy: This output is set when the block is activated, and remains set until execution of the function has been completed.

bError: If an error should occur during the execution of the function, then this output is set, after the bBusy output has been reset.

nErrId : Supplies the error number when the bError output is set.

Error number	Error description
0	No error
0x100	Error at initialisation of the communication via the 2 byte PLC interface
0x200	Error during communication
0x300	Timeout-Error. The permitted execution time was exceeded
0x400	Wrong parameter value at register number
0x500	Wrong parameter value at table number

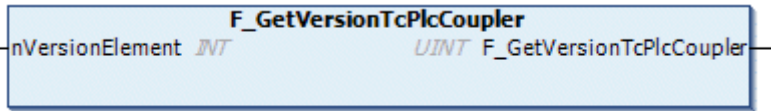
Requirements

Development environment	Target system type	PLC libraries to include (category group)
TwinCAT v3.1.0	PC or CX (x86, x64, ARM)	Tc2_Coupler (IO)

4 [obsolete functions]

4.1 F_GetVersionTcPlcCoupler

This function is obsolete and should not be used any longer. Use the global constant `stLibVersion_Tc2_Coupler` [▶ 19] to read version information from the plc library.



This function reads version information from the plc library.

FUNCTION F_GetVersionTcPlcCoupler : UINT

```
VAR_INPUT
  nVersionElement : INT;
END_VAR
```

nVersionElement : Version element, that is to be read. Possible parameters:

- 1 : major number;
- 2 : minor number;
- 3 : revision number;

Requirements

Development environment	Target system type	PLC libraries to include (category group)
TwinCAT v3.1.0	PC or CX (x86, x64, ARM)	Tc2_Coupler (IO)

5 Data types

5.1 PLCINTFSTRUCT

```

TYPE PLCINTFSTRUCT :
STRUCT
    Byte0    : BYTE;
    Byte1    : BYTE;
END_STRUCT
END_TYPE
    
```

Requirements

Development environment	Target system type	PLC libraries to include (category group)
TwinCAT v3.1.0	PC or CX (x86, x64, ARM)	Tc2_Coupler (IO)

5.2 E_CouplerErrType

```

TYPE E_CouplerErrType :
(
    CPLERR_NONE      := 0, (* No error *)
    CPLERR_FIELDBUS  := 1, (* Fieldbus error *)
    CPLERR_KBUS      := 2, (* Terminal bus error (KBus)*)
    CPLERR_TERM_IO   := 4, (* Terminal IO error *)
    CPLERR_COUPLER   := 8 (* Coupler error *)
);
END_TYPE
    
```

Requirements

Development environment	Target system type	PLC libraries to include (category group)
TwinCAT v3.1.0	PC or CX (x86, x64, ARM)	Tc2_Coupler (IO)

5.3 ST_CouplerDiag

```

TYPE ST_CouplerDiag :
STRUCT
    eErrType      : E_CouplerErrType;
    stFlashCode   : ST_FlashCode;
END_STRUCT
END_TYPE
    
```

eErrType : General Error type (type: [E_CouplerErrType](#) [► 17])

stFlashCode : The first and second sequence of the code of flash (type: [ST_FlashCode](#) [► 18])

Requirements

Development environment	Target system type	PLC libraries to include (category group)
TwinCAT v3.1.0	PC or CX (x86, x64, ARM)	Tc2_Coupler (IO)

5.4 ST_CouplerReg

```

TYPE ST_CouplerReg
STRUCT
    Lo : BYTE;
    
```

```

Hi : BYTE;
END_STRUCT
END_TYPE

```

A Coupler register has the size of one word. The parameterisation and the configuration of the coupler is discarded in the register.

Requirements

Development environment	Target system type	PLC libraries to include (category group)
TwinCAT v3.1.0	PC or CX (x86, x64, ARM)	Tc2_Coupler (IO)

5.5 ST_CouplerTable

```

TYPE ST_CouplerTable : ARRAY[ 0..255 ] OF ST_CouplerReg;
END_TYPE

```

The parameters and configuration of the coupler are stored in the coupler's EEPROM. The memory is divided into tables (Type [ST_CouplerReg](#) | 171). Each table possesses a maximum of 256 registers.

Requirements

Development environment	Target system type	PLC libraries to include (category group)
TwinCAT v3.1.0	PC or CX (x86, x64, ARM)	Tc2_Coupler (IO)

5.6 ST_FlashCode

```

TYPE ST_FlashCode :
STRUCT
  ErrType : WORD;
  ErrLocation : WORD;
END_STRUCT
END_TYPE

```

ErrType : Error type. Corresponds to the coupler's first flash sequence.

ErrLocation : Error location. Corresponds to the coupler's second flash sequence (the position of the last terminal before the error location - passive terminals are not included in the count!).

Requirements

Development environment	Target system type	PLC libraries to include (category group)
TwinCAT v3.1.0	PC or CX (x86, x64, ARM)	Tc2_Coupler (IO)

6 Global Constants

6.1 Library version

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

Global_Version

```
VAR_GLOBAL CONSTANT
    stLibVersion_Tc2_Coupler    : ST_LibVersion;
END_VAR
```

stLibVersion_Tc2_Coupler : version information of the Tc2_Coupler library (type: ST_LibVersion).

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



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

More Information:
www.beckhoff.com/te1000

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

