Technische Information

SICK
CoLa Kommunikationsbaustein

CCOM_TCP Funktionsbaustein für
Siemens S7-Steuerungen

Ausgabestand: 15. Mrz. 2012

SICK
Sensor Intelligence.

Inhaltsverzeichnis

1 Zu diesem Dokument

Bitte lesen Sie dieses Kapitel sorgfältig, bevor Sie mit dieser Betriebsanleitung und den SICK CoLa Kommunikationsbaustein arbeiten.

1.1 Funktion dieses Dokuments

Diese Betriebsanleitung beschreibt den Umgang mit dem SICK CCOM_TCP Funktionsbaustein. Sie leitet das technische Personal des Maschinenherstellers bzw. Maschinenbetreibers zur Projektierung und Inbetriebnahme des Funktionsbausteins an.

1.2 Zielgruppe

Diese Betriebsanleitung richtet sich an fachkundiges Personal wie z.B. Techniker oder Ingenieure.

2 Allgemeines

Der Funktionsbaustein CCOM_TCP unterstützt beim Datenaustausch zwischen SICK Geräten und S7-Steuerungen. Es werden nur S7-300/S7-400 Steuerungen unterstützt, die mit einer integrierten Industrial Ethernet (IE) Schnittstelle ausgerüstet sind.

Mit diesem Baustein können die folgenden SICK Sensoren angesteuert werden:
- CLV6xx
- LECTOR62x
- RFH62x
- RFU63x

Die folgende Abbildung zeigt die Darstellung des Funktionsbausteins in der Funktionsplan Ansicht (FUP).

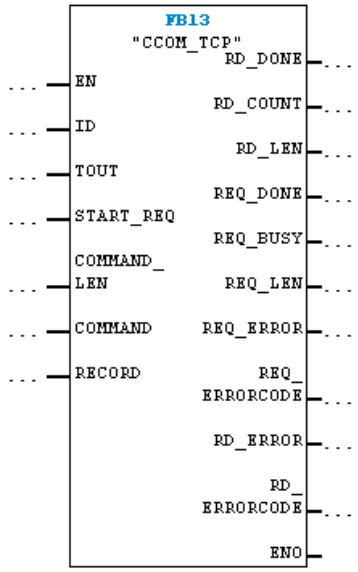

Abbildung 1: Darstellung des Funktionsbausteins in der FUP-Ansicht

Optionale Funktionen:
- Senden von CoLa[1] Kommandos an ein SICK Sensor
- Empfangen von CoLa Antworten eines SICK Sensor
- Empfangen von geräteseitig gesendeten Telegrammen (im SOPAS[2] Ausgabeformat konfigurierbar)

[1] Die command language (CoLa) ist ein SICK internes Protokoll zur Kommunikation mit SOPAS Geräten
[2] SOPAS-ET ist ein Engineering Tool zum parametrieren von SICK Sensoren

3 Hardwarekonfiguration

3.1 Unterstützte SPS-Steuerungen

Der Funktionsbaustein darf nur mit Simatic S7-Steuerungen der 300er und der 400er Familie betrieben werden. Es werden nur Steuerungen mit integrierter IE-Schnittstelle unterstützt. TCP Verbindungen, die über einen Kommunikationsprozessor (CP-Baugruppe) projektiert wurden werden nicht unterstützt.

3.2 Verbindungsaufbau

Bevor der Funktionsbaustein verwendet werden kann, muss in eine TCP-Verbindung zum Sensor hergestellt werden. Für S7-Steuerungen mit integrierter IE-Schnittstelle stellt Siemens die folgenden Kommunikationsbausteine zur Verfügung (Standard Library → Communication Blocks):

- FB65 (TCON): Zum aufbauen einer TCP-Verbindung
- FB66 (TDISCON): Zum abbauen einer TCP-Verbindung
- FB63 (TSEND): Zum senden von Daten
- FB64 (TRCV): Zum empfangen von Daten

Abbildung 2 zeigt den Aufruf des FB65 mit der zugehörigen Instanz (DB65) im OB1. Beim Anlauf der Steuerung wird einmalig der OB100 ausgeführt. Im OB100 wird das Startbit (REQ) des FB65 gesetzt, um die Verbindung über den FB65 aufzubauen. Ein erfolgreicher Verbindungsaufbau wird durch das Bit DONE = TRUE signalisiert. Die Verbindungsparameter zum Aufbau der Verbindung sind in einer Datenstruktur (UDT65) gespeichert.

Der ID Parameter des TCON Bausteins und der instanziierten UDT-Struktur muss identisch mit der ID des FB CCOM_TCP sein.

```
Network 1: HERSTELLEN EINER TCP VERBINDUNG | OPEN TCP CONNECTION
Herstellen einer TCP Verbindung mittels FB65 (TCON)
---
Open TCP connection via FB65 (TCON)

      CALL  "TCON" , "INSTANCE_FB65"                      FB65 / DB65
       REQ    :="TCON_PARMETER".CONNECT.REQ               DB2.DBX64.0
       ID     :=W#16#1
       DONE   :="TCON_PARMETER".CONNECT.DONE              DB2.DBX64.1
       BUSY   :="TCON_PARMETER".CONNECT.BUSY              DB2.DBX64.2
       ERROR  :="TCON_PARMETER".CONNECT.ERROR             DB2.DBX64.3
       STATUS :="TCON_PARMETER".CONNECT.STATUS            DB2.DBW66
       CONNECT:="TCON_PARMETER".CONNECT.TCON_PARAMETER    P#DB2.DBX0.0
```

Abbildung 2: Aufbau einer TCP-Verbindung mit Hilfe des FB65 (TCON)

Die folgende Tabelle zeigt eine Beispielkonfiguration des UDT65.

Byte	Parameter	Datentyp	Startwert	Beschreibung
0 - 1	block_length	WORD	W#16#0040	Länge des UDT65: 64 Bytes (fest)
2 - 3	id	WORD	W#16#0001	Referenz auf die TCP-Verbindung. Dieser Wert muss identisch zum ID Parameter am TCON (FB65) und zum CCOM_TCP (FB13) Baustein sein.
4	connection_type	BYTE	B#16#11	Verbindungstyp = TCP
5	active_est	BOOL	TRUE	Aktiver Verbindungsaufbau

Byte	Parameter	Datentyp	Startwert	Beschreibung
6	local_device_id	BYTE	B#16#02	Typ der TCP Verbindung In diesem Fall: Kommunikation über die integrierte Ethernet Schnittstelle bei den CPUs 315-2 PN/DP und 317-2 PN/DP
7	local_tsap_id_len	BYTE	B#16#02	Für den Verbindungstyp B#16#11 und einem passiven Endpunkt
8	rem_subnet_id_ len	BYTE	B#16#00	Wird nicht verwendet
9	rem_staddr_len	BYTE	B#16#04	Länge der IP-Adresse des Teilnehmers (SICK Gerät)
10	rem_tsap_id_len	BYTE	B#16#02	Fix für den Verbindungstyp B#16#11
11	Next_staddr_len	BYTE	B#16#00	Verwendete Länge des Parameters next_staddr (Wird nicht verwendet)
12 - 27	Local_tsap_id	Array [1..16] of BYTE	-	Nummer des verwendeten Lokalports: [1] = High byte der verwendeten Port-nummer in hexadezimaler Dar-stellung [2] = Low byte der verwendeten Port-nummer in hexadezimaler Dar-stellung [3..16] = B#16#00
28 - 33	rem_subnet_id	Array [1..6] of BYTE	-	Wird nicht verwendet [1..16] = B#16#0
34 - 39	Rem_staddr	Array [1..6] of BYTE	-	IP-Adresse des SICK Geräts Zum Beispiel: 192.168.10.15 [1] = B#16#C0 (192) [2] = B#16#A8 (168) [3] = B#16#0A (10) [4] = B#16#0F (15) [5-6] = B#16#00
40 - 55	Rem_tsap_id	Array [1..16] of BYTE	-	Portnummer des angeschlossenen SICK Geräts. SICK Kommunikationsport: 2112 (de-zimal) [1] = B#16#08 (High byte) [2] = B#16#40 (Low byte) [3..16] = B#16#0
56 - 61	Next_staddr	Array [1..6] of BYTE	-	Wird nicht verwendet [1..6] = B#16#0
62 - 63	spare	WORD	W#16#0000	Wird nicht verwendet

Eine genaue Beschreibung der UDT Parameter entnehmen Sie bitte dem Step7 Hilfesystem.

Vorrausetzung für den CCOM_TCP Baustein ist eine aktive TCP-Verbindung zum SICK Gerät. Abbildung 3 zeigt die Step7 Diagnose der offenen Kommunikation über Industrial Ethernet. Diese ist erreichbar mittels „Baugruppenzustand → Diagnose → Belegte Verbindungsresourcen“.

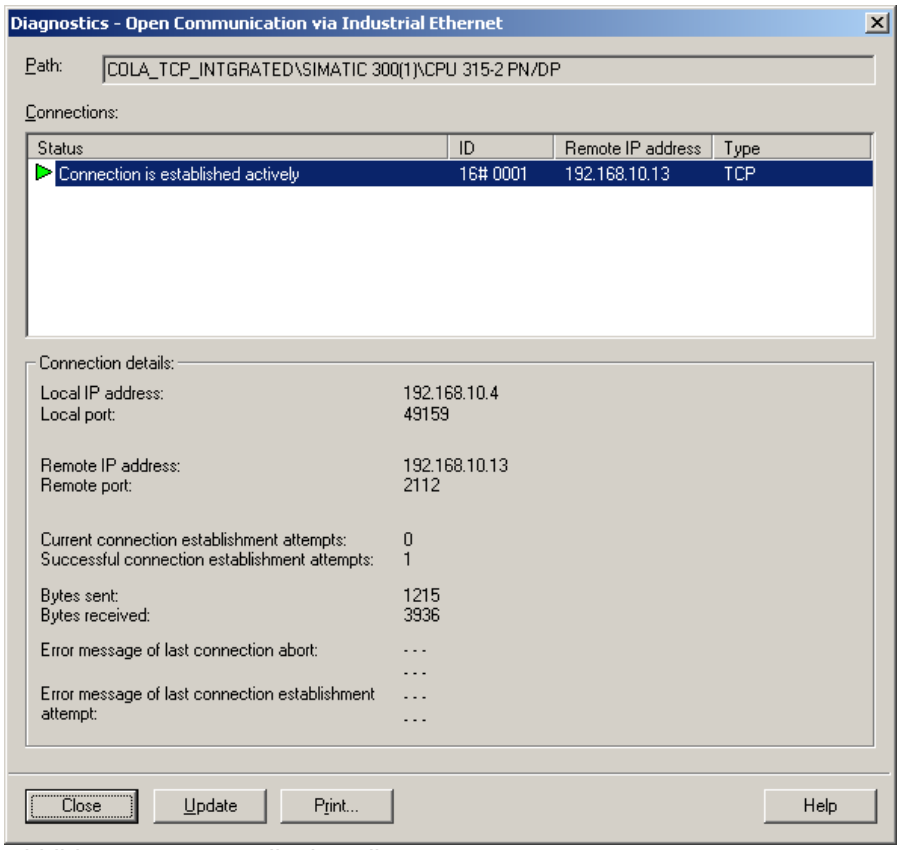

Abbildung 3: Kommunikationsdiagnose

4 Bausteinbeschreibung

Der Funktionsbaustein CCOM_TCP (FB13) vereinfacht die Benutzung von SICK Sensoren an S7-Steuerungen. Der Baustein ermöglicht das Senden und Empfangen von CoLa Telegrammen über eine mit dem FB65 projektierte TCP-Verbindung.

Der Baustein unterstützt bei folgenden Aufgaben:
- Senden von CoLa Kommandos an ein SICK Sensor
- Empfangen von CoLa Antworten eines SICK Sensor
- Empfangen von geräteseitig gesendeten Telegrammen (im SOPAS Ausgabeformat konfigurierbar)

Der Funktionsbaustein ist ein asynchron arbeitender FB d.h. die Bearbeitung erstreckt sich über mehrere FB-Aufrufe. Der Funktionsbaustein muss hierfür zyklisch im Anwenderprogramm aufgerufen werden.

Der CCOM_TCP Baustein (FB13) kapselt die Siemens Funktionsbausteine TSEND (FB63) und TRCV (FB64), die für die die Kommunikation zwischen SPS und Sensor verwendet werden.

4.1 Bausteinspezifikationen

Bausteinnummer:	FB13
Bausteinname:	CCOM_TCP
Version:	1.0
Aufgerufene Bausteine:	FB63 (TSEND) FB64 (TRCV) SFB4 (TON)
Verwendete Datenbausteine:	-
Bausteinaufruf:	Zyklisch
Verwendete Merker:	keine
Verwendete Zähler:	keine
Verwendetet Register:	AR1, AR2 (für Multiinstanzen)
Erstellsprache:	Step7-AWL

Die Bausteine FB63 und FB64 werden von der Siemens Bausteinbibliothek bereitgestellt.

4.2 Arbeitsweise

Um den CCOM_TCP Baustein einsetzten zu können, müssen zunächst die folgenden Parameter angegeben werden.

ID: Verbindungs-ID der TCP-Verbindung. Hier muss der gleiche Wert wie der ID Parameter des TCON Bausteins und der instanziierten UDT-Struktur angegeben werden. Siehe auch Abbildung 3.

COMMAND: Der Pointer zeigt auf den Datenbereich, in den das CoLa Kommando abgelegt ist. Der Datenbereich muss vom Programmierer selbst erstellt werden (z.B. Datenbaustein mit einem Array of CHAR). Die Kommandos müssen immer mit [STX] / [ETX] Rahmung angegeben werden.

COMMAND_LEN: Zeichenlänge des zu übertragenden CoLa Kommandos.

RECORD: Der Pointer zeigt auf den Datenbereich, in den die vom Gerät gesendeten Telegrame abgelegt werden. Der Datenbereich muss vom Programmierer selbst erstellt werden (z.B. Datenbaustein mit einem Array of BYTE).

4.2.1 Empfangen von Leseergebnissen (RD)

Daten die geräteseitig gesendet werden (RD), werden in den Record geschrieben, sobald der Funktionsbaustein neue Daten empfangen hat. Das Bit RD_DONE zeigt für einen SPS Zyklus den Empfang neuer Daten an. Sobald neue Daten empfangen wurden, wird der Zähler RD_COUNT inkrementiert. Die jeweilige Bytelänge des zuletzt empfangenden Telegramms kann dem Parameter RD_LEN entnommen werden.

4.2.2 Gerätekommunikation über CoLa Kommandos (REQ)

Bei der Kommunikation via CoLa Kommandos, wird das im COMMAND definierte Kommando zum Gerät übertragen. Die resultierende Antwort wird in den vom Pointer RECORD definierten Bereich abgelegt. CoLa Kommandos werden immer mit Steuerzeichen ([ETX], [STX] Rahmung) übertragen.

Sie starten die Übertragung, indem Sie den Parameter START_REQ mit einer positiven Flanke antriggern. Solange noch keine gültige Antwort auf das gesendete CoLa Kommando eingetroffen ist, wird dies über den Parameter REQ_BUSY signalisiert. Sollte innerhalb der Timeout Zeit (TOUT) keine Antwort eingetroffen sein, wird die Bearbeitung mit einem Timeout Fehler (REQ_ERRORCODE) abgebrochen. Der Ausgangsparameter REQ_DONE zeigt an, dass auf ein CoLa Kommando eine Antwort empfangen wurde (REQ_DONE = TRUE).

4.2.3 Timing

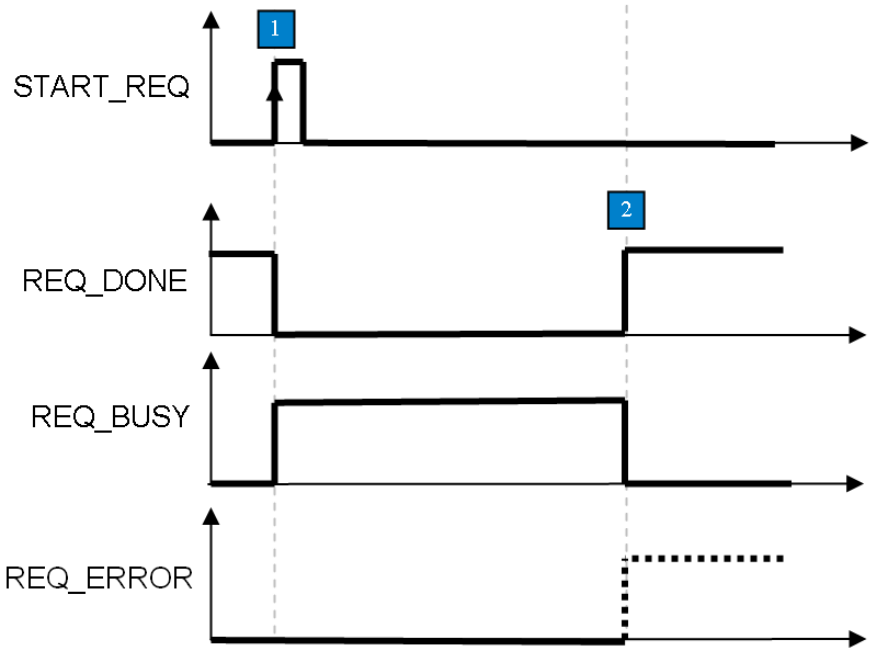

Abbildung 4: Timing Diagramm

1: Anforderung durch eine positve Flanke an START_REQ. Das vom Parameter COMMAND referenzierte CoLa Kommando wird zum Sensor gesendet. Es kann immer nur ein Kommando zeitgleich gesendet werden.

2: Wenn das Kommando versendet ist und die Antwort empfangen wurden, wird die Aktion mit „REQ_DONE“ beendet. Wenn die Aktion fehlerhaft verläuft, wird mit „REQ_ERROR“ beendet. Bei Abbruch mit „REQ_ERROR“ enthält „REQ_ERRORCODE“ den aufgetretenen Fehlercode.

4.3 Verhalten im Fehlerfall

Im Fehlerfall signalisieren die Errorbits REQ_ERROR oder RD_ERROR, dass ein Fehler aufgetreten ist. In diesem Fall wird über den Parameter REQ_ERRORCODE bzw. RD_ERRORCODE ein Fehlercode ausgegeben. Das Errorbit REQ_ERROR bleibt solange gesetzt, bis ein neuer Auftrag gestartet wird. Der Parameter RD_ERROR ist immer nur für einen SPS-Zyklus aktiv und wird danach zurückgesetzt, sollte der Fehler nicht weiter bestehen.

4.4 Parameter

Parameter	Deklaration	Datentyp	Speicherbereich	Beschreibung
EN	INPUT	BOOL	E,M,D,L, Konst.	Enable Eingang (KOP und FUP).
ID	INPUT	INT	E,M,D,L, Konst.	Verbindungs-ID zur projektierten TCP-Verbindung (siehe Kommunikationsdiagnose Abbildung 3 oder ID Parameter TCON-FB)
TOUT	INPUT	TIME	E,M,D,L, Konst.	Zeit, nachdem ein Timeout-Fehler ausgelöst wird. Wenn dieser Parameter nicht beschaltet ist, beträgt die Timeout Zeit standardmäßig 5Sekunden. Bitte beachten Sie, dass einige CoLa Kommandos eine längere Bearbeitungszeit benötigen (z.B. Speicherkommandos).
START_REQ	INPUT	BOOL	E,M,D,L	Positive Flanke: Senden ein CoLa Kommando und wartet auf die entsprechende Antwort.
COMMAND	INPUT	ANY	D	Pointer auf den Bereich, welcher das zu übertragende CoLa Kommando enthält. Es ist nur das Datentyp BYTE zulässig. Die Kommandos müssen immer mit [STX] / [ETX] Rahmung angegeben werden (ASCII-Steuerzeichen). <u>Hinweis:</u> Beachten Sie, dass der Parameter immer die vollständige Angabe der DB-Parameter erfordert (Bsp.: P#DB13.DBX0.0 BYTE 100). Das Weglassen einer expliziten DB-Nr. ist unzulässig und führt zu einem Bausteinfehler.
COMMAND_LEN	INPUT	INT	E,M,D,L, Konst.	Anzahl der Bytes, die das zu sendende CoLa Kommando enthält, worauf der Pointer #COMMAND referenziert.

Parameter	Deklaration	Datentyp	Speicherbereich	Beschreibung
RECORD	INPUT	ANY	D	Pointer auf den Bereich, in denen die vom gerät gesendeten Telegramme abgelegt werden sollen. Es ist nur der Datentyp BYTE zulässig. Hinweis: Beachten Sie, dass der Parameter immer die vollständige Angabe der DB-Parameter erfordert (Bsp.: P#DB13.DBX0.0 BYTE 100). Das Weglassen einer expliziten DB-Nr. ist unzulässig und führt zu einem Bausteinfehler.
RD_DONE	OUTPUT	BOOL	A,M,D,L	Positive Flanke: Ein vom Gerät gesendetes Leseergebnis wurde empfangen (Formatierung siehe SOPAS Ausgabeformat). Wenn ein Leseergebnis empfangen wurde, ist das Bit für jeweils einen SPS-Zyklus gesetzt. Das Leseergebnis steht im vom Parameter #RECORD referenzierten Speicherbereich zur Verfügung.
RD_COUNT	OUTPUT	BYTE	A,M,D,L	Zählt die Anzahl der Leseergebnisse die empfangen wurden. Der Zähler zählt von 0…255 (Dez.). Bei einem Überlauf fängt der Zähler wieder bei null an.
RD_LEN	OUTPUT	INT	A,M,D,L	Gibt die Bytelänge des empfangenden Leseergebnisses an.
REQ_DONE	OUTPUT	BOOL	A,M,D,L	Zeigt an, ob ein CoLa Kommando abgeschickt und eine Antwort empfangen wurde. TRUE: Bearbeitung abgeschlossen FALSE: Bearbeitung noch nicht abgeschlossen Die Kommandoantwort steht im, vom Parameter #RECORD referenzierten, Speicherbereich zur Verfügung.
REQ_BUSY	OUTPUT	BOOL	A,M,D,L	REQ Auftrag ist in Bearbeitung.
REQ_LEN	OUTPUT	INT	A,M,D,L	Länge eines Antworttelegramms in BYTE.
REQ_ ERROR	OUTPUT	BOOL	A,M,D,L	REQ Fehler Status: 0: Kein Fehler 1: Abbruch mit Fehler
RD_ERROR	OUTPUT	BOOL	A,M,D,L	RD Fehler Status: 0: Kein Fehler 1: Abbruch mit Fehler

Parameter	Deklaration	Datentyp	Speicherbereich	Beschreibung
REQ_ERROR CODE	OUTPUT	WORD	A,M,D,L	REQ Fehlerstatus (siehe Fehlercodes).
RD_ERROR CODE	OUTPUT	WORD	A,M,D,L	RD Fehlerstatus (siehe Fehlercodes).
ENO	OUTPUT	BOOL	A,M,D,L	Enable Ausgang (KOP und FUP).

4.5 Fehlercodes

Die Parameter REQ_ERRORCODE und RD_ERRORCODE enthalten die folgenden Fehlerinformationen:

Fehlercode	Kurzbeschreibung	Beschreibung
W#16#0000	Kein Fehler	Kein Fehler
W#16#0001	Ungültiger Speicherbereich #RECORD Pointer angegeben	Ungültiger Speicherbereich des angegebenen ANY-Pointers. Dem Pointer muss ein DB zugeordnet werden.
W#16#0002	Ungültige Pointerlänge #RECORD angegeben	Die Länge des referenzierten Datenbaustein ist kürzer als die vom Pointer vorgegebene Länge.
W#16#0003	Ungültiger Speicherbereich #COMMAND Pointer angegeben	Ungültiger Speicherbereich des angegebenen ANY-Pointers. Dem Pointer muss ein DB zugeordnet werden.
W#16#0004	Ungültige Pointerlänge #COMMAND angegeben	Die Länge des referenzierten Datenbaustein ist kürzer als die vom Pointer vorgegebene Länge.
W#16#0005	Timeout	Der Auftrag konnte innerhalb der gewählten Timeoutzeit nicht ausgeführt werden. Dies könnte folgende Ursache haben: - Gerät ist nicht mit der SPS Verbunden - Kommunikationsparameter fehlerhaft - Verwendung von CoLa Kommandos, die keine Antwort (Echo) zurücksenden - Bearbeitungszeit des Kommandos > Timeout Zeit
W#16#0006	Ungültige Kommandolänge	Die Länge des zu sendenden Kommandos ist länger als die angegebene Kommandolänge (COMMAND_LEN).
W#16#0007	Ungültiges CoLa Kommando	Das angegebene CoLa Kommando hat keine [STX] [ETX] Rahmung.
W#16#000A	Empfangendes Telegramm > #RECORD Länge	Das empfangende Telegram ist länger als die angegebene #RECORD Länge.
W#16#000B	Telegramm ohne Steuerzeichen empfangen	- Es wurde ein Telegramm ohne [STX] [ETX] Rahmung empfangen. - Das empfangende Telegram ist länger als die angegebene #RECORD Länge.
W#16#7XXX - W#16#8XXX	FB63 / FB64 Fehler	Fehlerbeschreibung siehe Step7 Hilfe.

5 Beispiel

Abbildung 7 zeigt eine Beispielbeschaltung des CCOM_TCP FBs. Die TCP Verbindung zum SICK Sensor wird mit dem FB65 (TCON) während des SPS Anlaufs hergestellt (siehe Abbildung 5 / Abbildung 6).

Programmaufruf:

```
OB100 :  "Complete Restart"
Comment:

Network 1: TCP VERBINDUNG HERSTELLEN | ESTABLISHing A TCP CONNECTION
Herstellen einer TCP Verbindung nach jedem SPS restart.
---
Open TCP connection after every PLC restart.

      SET
      =     "TCON_PARMETER".CONNECT.REQ    DB2.DBX64.0
```

Abbildung 5: Start des Verbindungsaufbaus im OB100

```
OB1 :  "Main Program Sweep (Cycle)"
Bitte beachten:
Es werden nur S7-300/S7-400 Steuerungen mit integrierter TCP-Schnittstelle
unterstützt
---
Please note:
This function block my only be operated with S7-300/S7-400 controllers with
integrated TCP interfaces

Network 1: HERSTELLEN EINER TCP VERBINDUNG | OPEN TCP CONNECTION
Herstellen einer TCP Verbindung mittels FB65 (TCON)
---
Open TCP connection via FB65 (TCON)

      CALL  "TCON" , "INSTANCE_FB65"                      FB65 / DB65
       REQ     :="TCON_PARMETER".CONNECT.REQ              DB2.DBX64.0
       ID      :=W#16#1
       DONE    :="TCON_PARMETER".CONNECT.DONE             DB2.DBX64.1
       BUSY    :="TCON_PARMETER".CONNECT.BUSY             DB2.DBX64.2
       ERROR   :="TCON_PARMETER".CONNECT.ERROR            DB2.DBX64.3
       STATUS  :="TCON_PARMETER".CONNECT.STATUS           DB2.DBW66
       CONNECT:="TCON_PARMETER".CONNECT.TCON_PARAMETER    P#DB2.DBX0.0

      CLR
      =     "TCON_PARMETER".CONNECT.REQ                  DB2.DBX64.0
```

Abbildung 6: Aufruf des FB65 (TCON) zum anlegen einer TCP-Verbindung

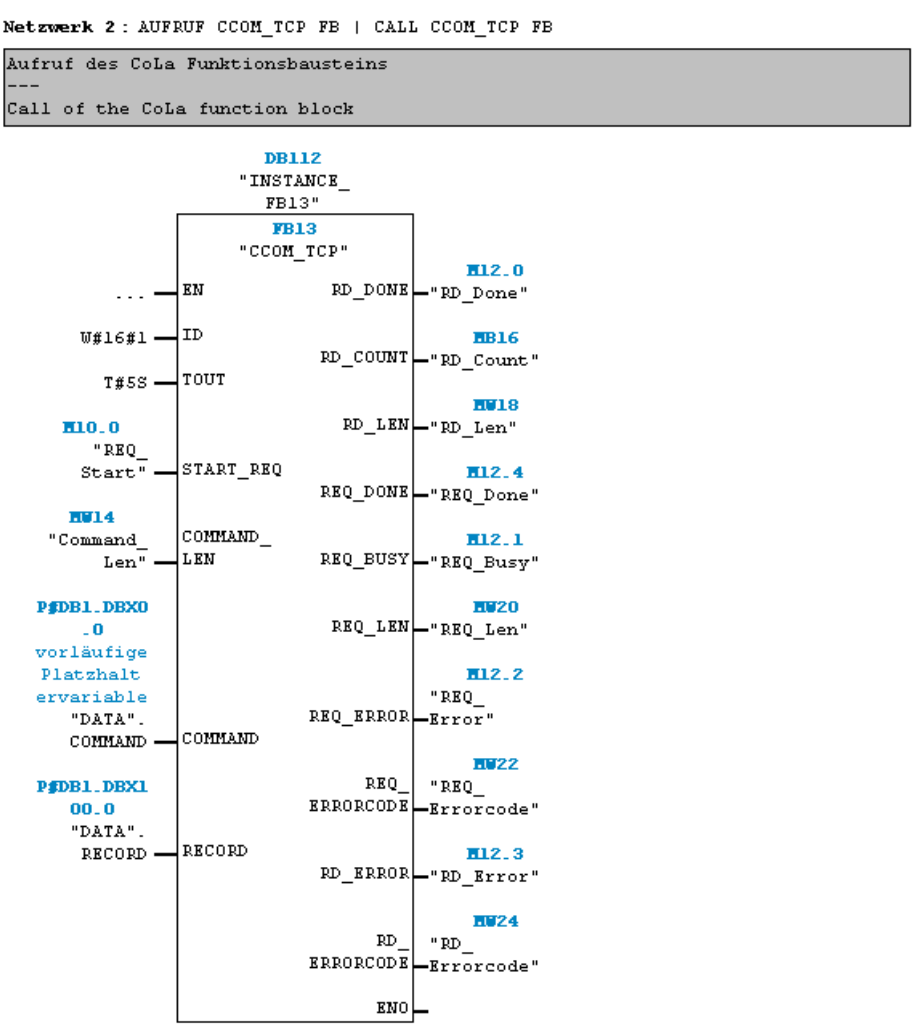

Abbildung 7: Aufruf des CCOM_TCP Kommunikationsbausteins

Variablentabelle zum ausführen eines CoLa Kommandos:

// CCOM TCP Integrated function block				
M 10.0	"REQ_Start"	BOOL	true	
// Reading Result Status				
M 12.0	"RD_Done"	BOOL	false	
MB 16	"RD_Count"	DEC	23	
M 12.3	"RD_Error"	BOOL	false	
MW 24	"RD_Errorcode"	HEX	W#16#0000	
// Requesting Result Status				
M 12.4	"REQ_Done"	BOOL	true	
M 12.1	"REQ_Busy"	BOOL	false	
M 12.2	"REQ_Error"	BOOL	false	
MW 22	"REQ_Errorcode"	HEX	W#16#0000	
// Command				
MW 14	"Command_Len"	DEC	15	15
DB1.DBB 0	"DATA".COMMAND[1]	CHARACTER	'['	'['
DB1.DBB 1	"DATA".COMMAND[2]	CHARACTER	's'	's'
DB1.DBB 2	"DATA".COMMAND[3]	CHARACTER	'M'	'M'
DB1.DBB 3	"DATA".COMMAND[4]	CHARACTER	'N'	'N'
DB1.DBB 4	"DATA".COMMAND[5]	CHARACTER	' '	' '
DB1.DBB 5	"DATA".COMMAND[6]	CHARACTER	'm'	'm'
DB1.DBB 6	"DATA".COMMAND[7]	CHARACTER	'T'	'T'
DB1.DBB 7	"DATA".COMMAND[8]	CHARACTER	'C'	'C'
DB1.DBB 8	"DATA".COMMAND[9]	CHARACTER	'g'	'g'
DB1.DBB 9	"DATA".COMMAND[10]	CHARACTER	'a'	'a'
DB1.DBB 10	"DATA".COMMAND[11]	CHARACTER	't'	't'
DB1.DBB 11	"DATA".COMMAND[12]	CHARACTER	'e'	'e'
DB1.DBB 12	"DATA".COMMAND[13]	CHARACTER	'o'	'o'
DB1.DBB 13	"DATA".COMMAND[14]	CHARACTER	'n'	'n'
DB1.DBB 14	"DATA".COMMAND[15]	CHARACTER	']'	']'
DB1.DBB 15	"DATA".COMMAND[16]	CHARACTER	B#16#00	
DB1.DBB 16	"DATA".COMMAND[17]	CHARACTER	B#16#00	
DB1.DBB 17	"DATA".COMMAND[18]	CHARACTER	B#16#00	
DB1.DBB 18	"DATA".COMMAND[19]	CHARACTER	B#16#00	
DB1.DBB 19	"DATA".COMMAND[20]	CHARACTER	B#16#00	

Das CoLa Kommando (hier: ’[STX]sMN mTCgateon[ETX]’) wird ausgeführt, sobald das Bit „REQ_START“ mit einer positiven Flanke angesteuert wird. Dem Parameter Command_Len wird die Länge des Kommandos übergeben (hier: 15 Zeichen).

Variablentabelle für eingehende Kommandoantworten:

// Record			
MW 18	"RD_Len"	DEC	50
MW 20	"REQ_Len"	DEC	17
DB1.DBB 100	"DATA".RECORD[1]	CHARACTER	'┐'
DB1.DBB 101	"DATA".RECORD[2]	CHARACTER	's'
DB1.DBB 102	"DATA".RECORD[3]	CHARACTER	'A'
DB1.DBB 103	"DATA".RECORD[4]	CHARACTER	'N'
DB1.DBB 104	"DATA".RECORD[5]	CHARACTER	' '
DB1.DBB 105	"DATA".RECORD[6]	CHARACTER	'm'
DB1.DBB 106	"DATA".RECORD[7]	CHARACTER	'T'
DB1.DBB 107	"DATA".RECORD[8]	CHARACTER	'C'
DB1.DBB 108	"DATA".RECORD[9]	CHARACTER	'g'
DB1.DBB 109	"DATA".RECORD[10]	CHARACTER	'a'
DB1.DBB 110	"DATA".RECORD[11]	CHARACTER	't'
DB1.DBB 111	"DATA".RECORD[12]	CHARACTER	'e'
DB1.DBB 112	"DATA".RECORD[13]	CHARACTER	'o'
DB1.DBB 113	"DATA".RECORD[14]	CHARACTER	'n'
DB1.DBB 114	"DATA".RECORD[15]	CHARACTER	' '
DB1.DBB 115	"DATA".RECORD[16]	CHARACTER	'1'
DB1.DBB 116	"DATA".RECORD[17]	CHARACTER	'└'
DB1.DBB 117	"DATA".RECORD[18]	CHARACTER	B#16#00
DB1.DBB 118	"DATA".RECORD[19]	CHARACTER	B#16#00
DB1.DBB 119	"DATA".RECORD[20]	CHARACTER	B#16#00

Die Kommandoantwort (REQ) auf ein gesendete Kommando (hier: ’[STX]sAN mTCgateon 1[ETX]’) steht im Record Bereich zur Verfügung, sobald sich der Wert des Ausgangsbits „REQ_DONE“ von FALSE auf TRUE ändert (positive Flanke). Der Parameter „REQ_LEN“ gibt an, wie viele Bytes empfangen wurden bzw. gültig sind.

Variablentabelle für eingehende Leseergebnisse:

// Record			
MW 18	"RD_Len"	DEC	11
MW 20	"REQ_Len"	DEC	17
DB1.DBB 100	"DATA".RECORD[1]	CHARACTER	'ך'
DB1.DBB 101	"DATA".RECORD[2]	CHARACTER	'1'
DB1.DBB 102	"DATA".RECORD[3]	CHARACTER	'2'
DB1.DBB 103	"DATA".RECORD[4]	CHARACTER	'3'
DB1.DBB 104	"DATA".RECORD[5]	CHARACTER	'4'
DB1.DBB 105	"DATA".RECORD[6]	CHARACTER	'5'
DB1.DBB 106	"DATA".RECORD[7]	CHARACTER	'6'
DB1.DBB 107	"DATA".RECORD[8]	CHARACTER	'7'
DB1.DBB 108	"DATA".RECORD[9]	CHARACTER	'8'
DB1.DBB 109	"DATA".RECORD[10]	CHARACTER	'9'
DB1.DBB 110	"DATA".RECORD[11]	CHARACTER	'ט'
DB1.DBB 111	"DATA".RECORD[12]	CHARACTER	'e'
DB1.DBB 112	"DATA".RECORD[13]	CHARACTER	'o'
DB1.DBB 113	"DATA".RECORD[14]	CHARACTER	'n'
DB1.DBB 114	"DATA".RECORD[15]	CHARACTER	' '
DB1.DBB 115	"DATA".RECORD[16]	CHARACTER	'1'
DB1.DBB 116	"DATA".RECORD[17]	CHARACTER	'ט'
DB1.DBB 117	"DATA".RECORD[18]	CHARACTER	B#16#00
DB1.DBB 118	"DATA".RECORD[19]	CHARACTER	B#16#00

Daten die geräteseitig gesendet werden (RD), werden in den Record geschrieben, sobald der Funktionsbaustein neue Daten empfangen hat. Das Bit „RD_DONE“ zeigt für einen SPS Zyklus den Empfang neuer Daten an (Signalwechsel von FALSE auf TRUE). Sobald neue Daten empfangen wurden, wird der Zähler RD_COUNT inkrementiert. Der Parameter „RD_LEN“ gibt an, wie viele Bytes empfangen wurden bzw. gültig sind.