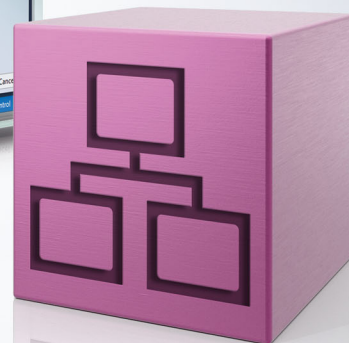
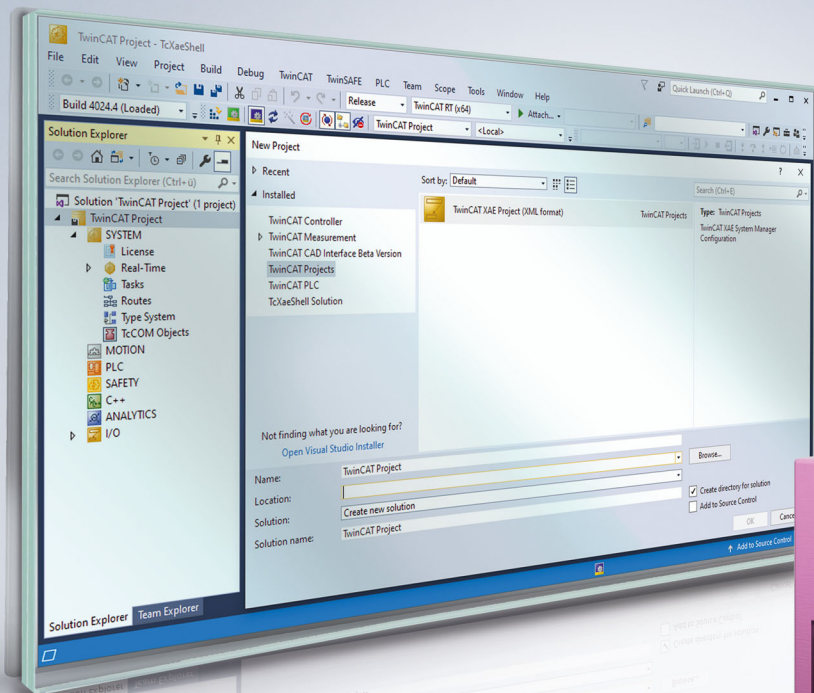


BECKHOFF New Automation Technology

Handbuch | DE

TF6600

TwinCAT 3 | RFID Reader Communication



Inhaltsverzeichnis

1	Vorwort.....	5
1.1	Hinweise zur Dokumentation	5
1.2	Zu Ihrer Sicherheit.....	6
1.3	Hinweise zur Informationssicherheit	7
2	Übersicht.....	8
3	Installation	9
3.1	Systemvoraussetzungen	9
3.2	Installation	9
3.3	Lizenzierung	12
4	Technische Einführung	15
4.1	RFID-Reader-Hardware	15
4.2	RFID-Reader-Anbindung	19
4.3	RFID-Befehlssatz	21
5	Konfiguration.....	26
5.1	RFID-Reader-Einstellungen und -Handhabung	26
5.1.1	Balluff	26
5.1.2	Baltech	27
5.1.3	Deister electronic	29
5.1.4	Leuze electronic	30
5.1.5	Pepperl+Fuchs	32
6	SPS API	35
6.1	Funktionsbaustein	35
6.1.1	FB_RFIDReader	35
6.1.2	Handhabung.....	39
6.1.3	Konfiguration	40
6.1.4	Low-Level-Kommunikation.....	41
6.2	Datentypen	43
6.2.1	Strukturen.....	43
6.2.2	Enumerationen.....	65
6.2.3	T_RFID_TranspSRN.....	68
6.3	Globale Konstanten.....	68
6.3.1	Global_Version.....	68
7	Beispiele	69
7.1	Tutorial	69
7.1.1	Glossar.....	70
7.1.2	Installation/Bibliotheken	70
7.1.3	Serielle Anbindung	70
7.1.4	Bausteindeklaration.....	71
7.1.5	Bausteinverwendung.....	71
7.1.6	Test	72
7.2	Beispiel 1.....	73
7.3	Beispiel 2.....	74
7.4	Beispiel 3.....	75

8 Anhang	77
8.1 RFID-Fehlercodes.....	77
8.2 Support und Service.....	79

1 Vorwort

1.1 Hinweise zur Dokumentation

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

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

Das Fachpersonal ist verpflichtet, stets die aktuell gültige Dokumentation zu verwenden.

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

Disclaimer

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

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

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

Marken

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

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

Patente

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

EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702

mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.



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

Copyright

© Beckhoff Automation GmbH & Co. KG, Deutschland.

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

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

1.2 Zu Ihrer Sicherheit

Sicherheitsbestimmungen

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

Haftungsausschluss

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

Qualifikation des Personals

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

Signalwörter

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

Warnungen vor Personenschäden

GEFAHR

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

WARNUNG

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

VORSICHT

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

Warnung vor Umwelt- oder Sachschäden

HINWEIS

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

Information zum Umgang mit dem Produkt



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

1.3 Hinweise zur Informationssicherheit

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

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

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

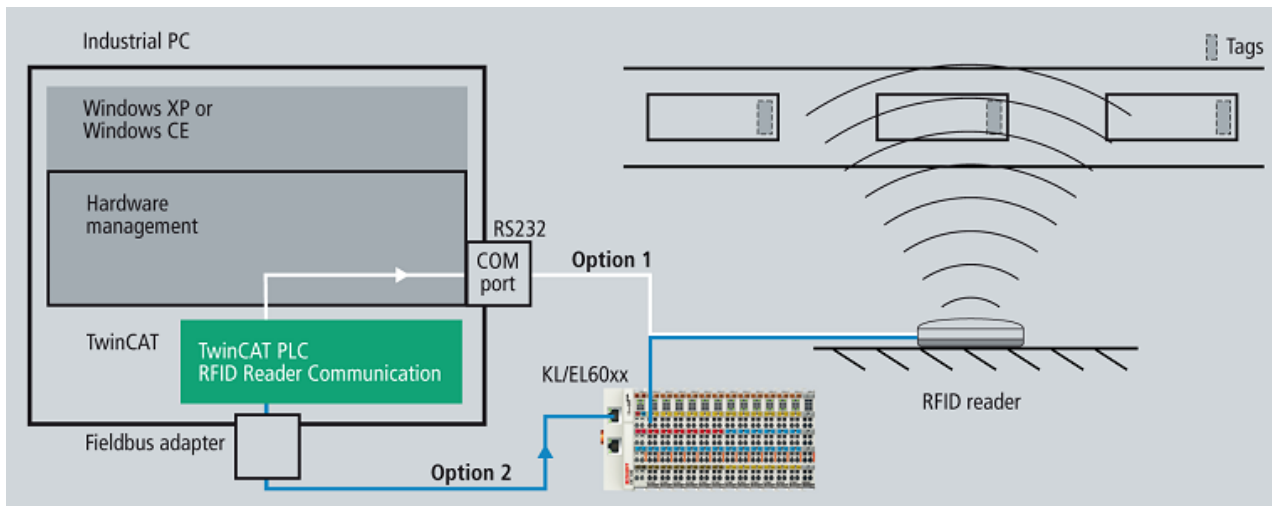
Um stets über Hinweise zur Informationssicherheit zu Produkten von Beckhoff informiert zu sein, abonnieren Sie den RSS Feed unter <https://www.beckhoff.de/secinfo>.

2 Übersicht

Die Bibliothek TC3 RFID Reader Communication ermöglicht die Kommunikation zu RFID Readern aus dem SPS-Programm heraus. Als RFID Reader werden dabei sowohl reine Lesegeräte als auch Schreib-/Lesegeräte bezeichnet.

Mit der TwinCAT-RFID-Bibliothek können umfangreiche Applikationen, die unterschiedliche Funktionen der RFID Reader nutzen, leicht verwirklicht werden. Da kein herstellerspezifisches Schnittstellenprotokoll detailliert recherchiert und umgesetzt werden muss, ist der Implementierungsaufwand sehr gering. Der Frameaufbau, die Telegrammzusammensetzung, die Befehlsbezeichnung, die Telegrammerkennung und einige weitere Protokolleigenarten werden automatisch durch die Bibliothek ausgeführt.

Das nachfolgende Bild zeigt die schematische Darstellung einer RFID-Reader-Anwendung.



Die Handhabung der Bibliothek ist für alle unterstützten RFID-Reader-Modelle gleich. Bei einem Herstellerwechsel müssen nur kleine Änderungen in der Applikation vorgenommen werden.

Eine Übersicht der unterstützten RFID-Reader-Modelle finden Sie im Abschnitt [Technische Einführung > RFID-Reader-Hardware](#) [► 15]. Zu den unterstützten Modellen gehören u. a. Geräte der Hersteller Balluff, Baltech, Deister electronic, Leuze electronic und Pepperl+Fuchs.

Für Beckhoff Multitouch-Control-Panel ist optional der Compact RFID Reader verfügbar. Im Gegensatz zu den anderen RFID-Reader-Modellen wird dieser nicht mithilfe der TF6600 RFID Reader Communication aus TwinCAT heraus angesprochen, sondern allein mithilfe der TF6340 Serial Communication. Eine Produktbeschreibung des Compact RFID Reader sowie ein SPS-Beispiel, das die Kommunikation mit diesem Gerät zeigt, finden Sie im Online Information System im Abschnitt [Industrie-PC > Compact RFID Reader](#).

3 Installation

3.1 Systemvoraussetzungen

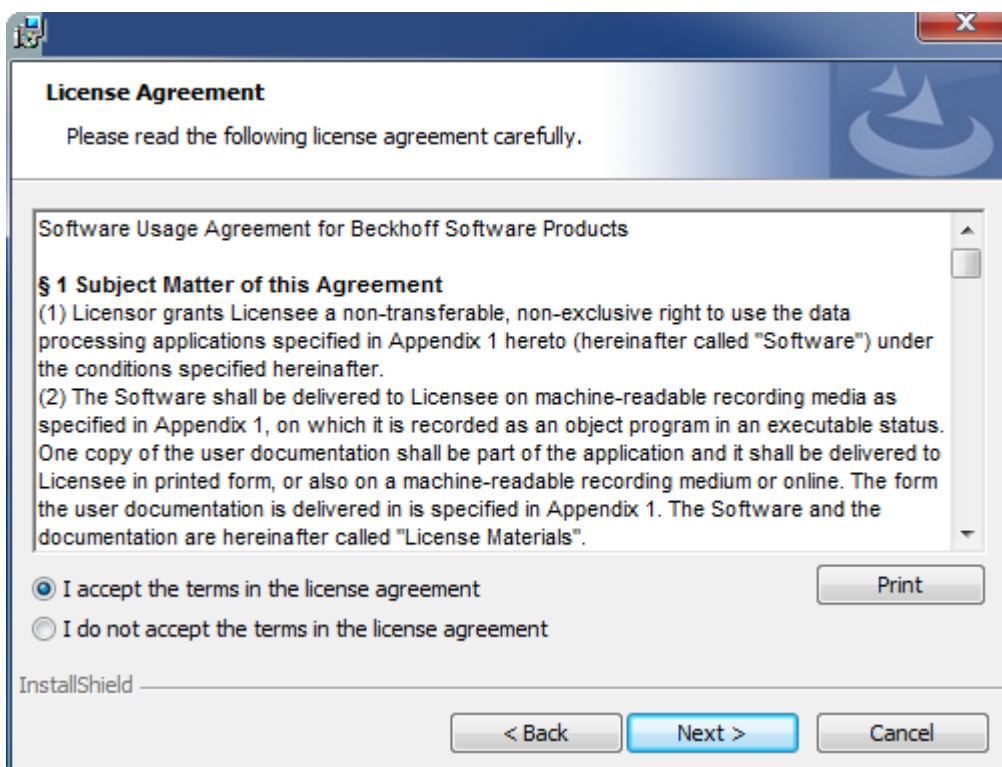
Technische Daten	Beschreibung
Betriebssystem	WinXP, WES, Win7, WES7, Win10
Zielpattform	PC oder CX (x86, x64, ARM): WinXP, WES, Win7, WES7, WEC7, Win10
Minimale Plattform Level	P 20 P 30 bei Anbindung über USB / VirtualComPort
Minimale TwinCAT-Version	TwinCAT 3.1.4013 oder höher
Erforderliches TwinCAT-Setup-Level	TwinCAT XAE TC3 PLC
Erforderliche TwinCAT-Lizenz	TF6600 TC3 RFID Reader Communication
Einzubindende TwinCAT-SPS-Bibliothek	Tc2_RFID Tc2_SerialCom

Je nach RFID-Reader-Modell benötigen Sie zur grundlegenden Konfiguration einmalig ein herstellereigenes Tool (siehe [RFID-Reader-Einstellungen und -Handhabung](#) [▶ 26]). Beachten Sie auch dessen Systemvoraussetzungen. Die Voreinstellung kann ebenso von einem anderen PC aus vorgenommen werden. Es bietet sich auch die Nutzung von proprietären Testtools für den Aufbau an.

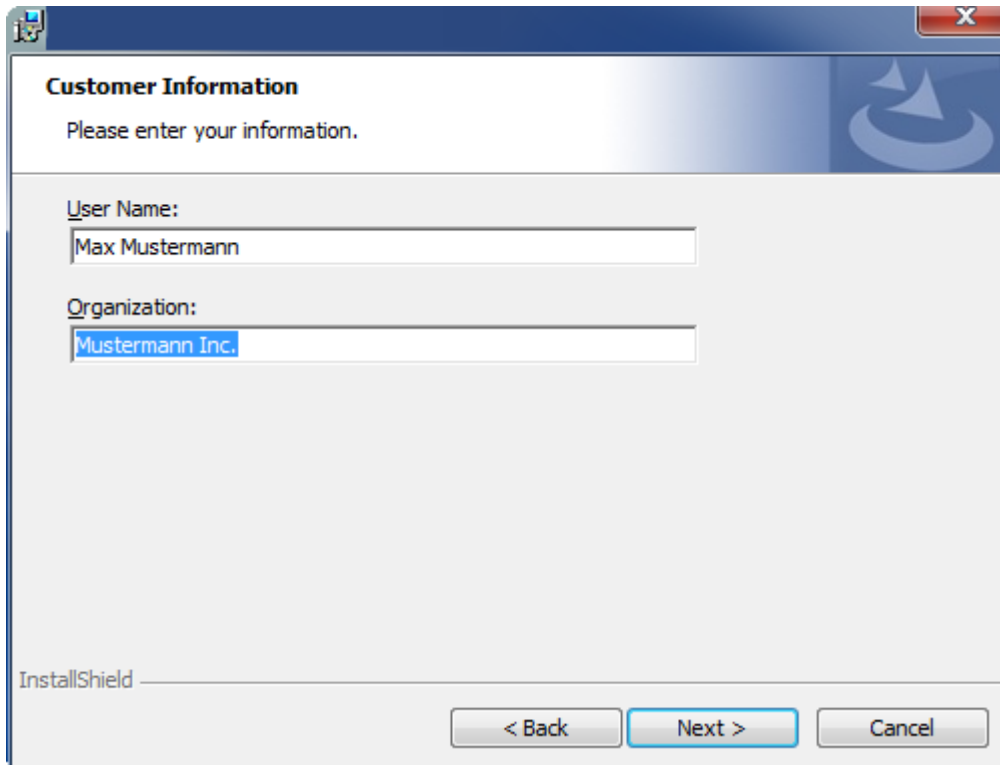
3.2 Installation

Nachfolgend wird beschrieben, wie die TwinCAT 3 Function für Windows-basierte Betriebssysteme installiert wird.

- ✓ Die Setup-Datei der TwinCAT 3 Function wurde von der Beckhoff-Homepage heruntergeladen.
- 1. Führen Sie die Setup-Datei als Administrator aus. Wählen Sie dazu im Kontextmenü der Datei den Befehl **Als Administrator ausführen**.
 - ⇒ Der Installationsdialog öffnet sich.
- 2. Akzeptieren Sie die Endbenutzerbedingungen und klicken Sie auf **Next**.



3. Geben Sie Ihre Benutzerdaten ein.



Customer Information
Please enter your information.

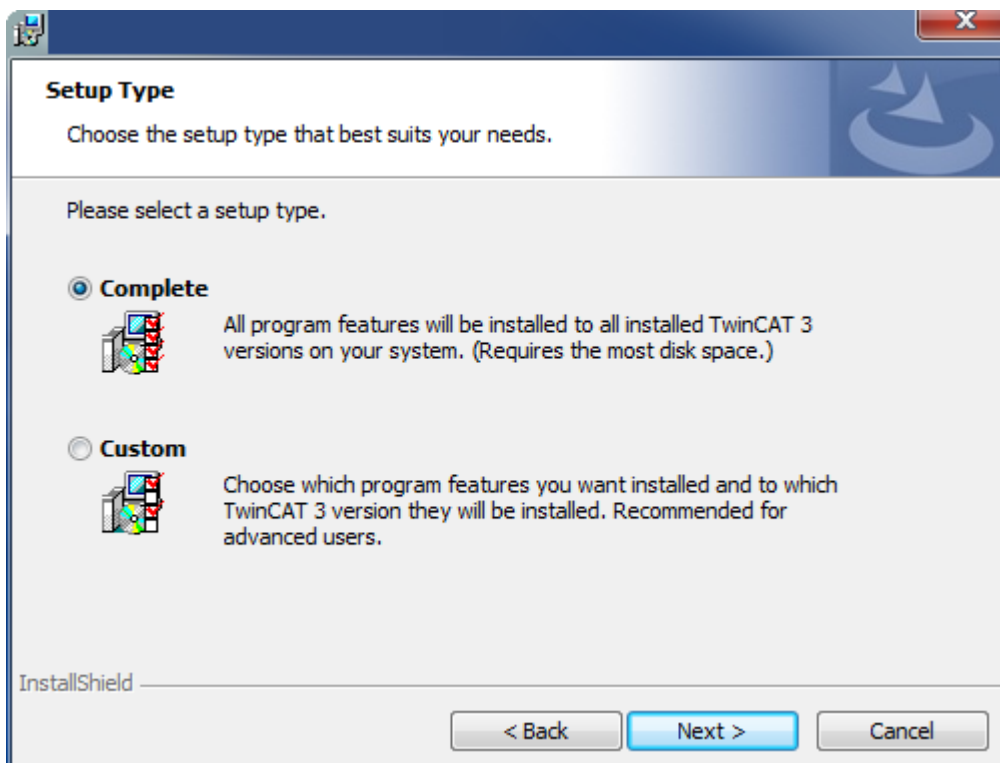
User Name:
Max Mustermann

Organization:
Mustermann Inc.

InstallShield

< Back Next > Cancel

4. Wenn Sie die TwinCAT 3 Function vollständig installieren möchten, wählen Sie **Complete** als Installationstyp. Wenn Sie die Komponenten der TwinCAT 3 Function separat installieren möchten, wählen Sie **Custom**.



Setup Type
Choose the setup type that best suits your needs.

Please select a setup type.

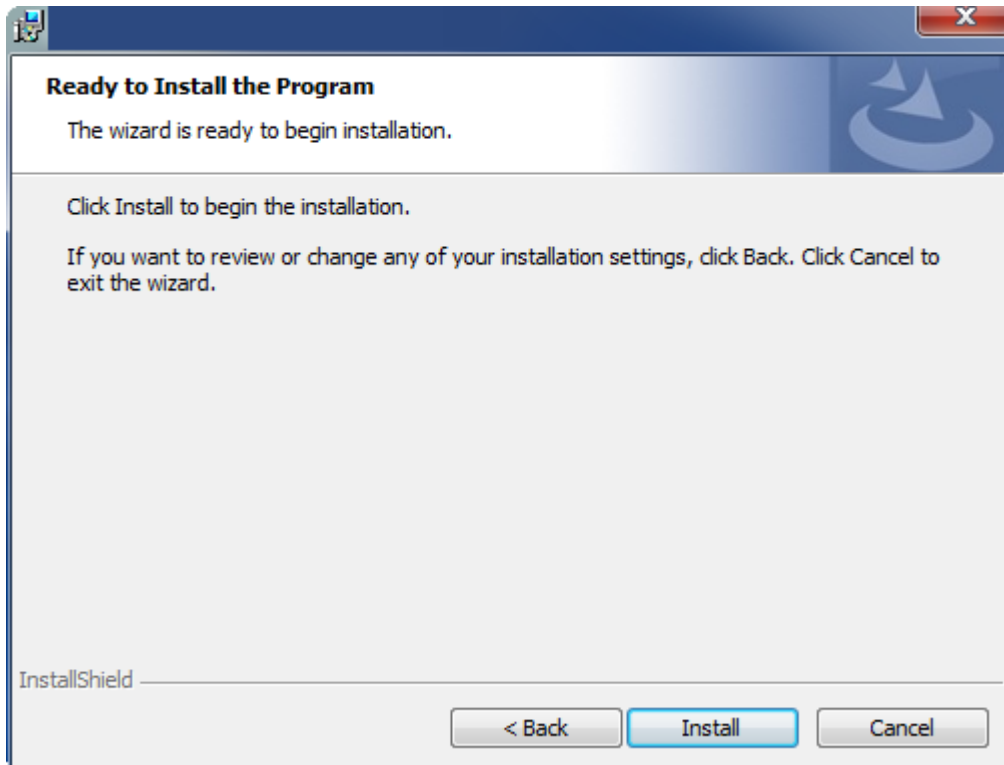
Complete
All program features will be installed to all installed TwinCAT 3 versions on your system. (Requires the most disk space.)

Custom
Choose which program features you want installed and to which TwinCAT 3 version they will be installed. Recommended for advanced users.

InstallShield

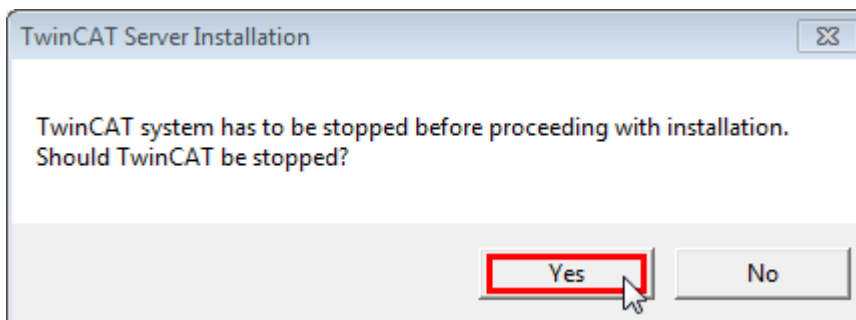
< Back Next > Cancel

5. Wählen Sie **Next** und anschließend **Install**, um die Installation zu beginnen.

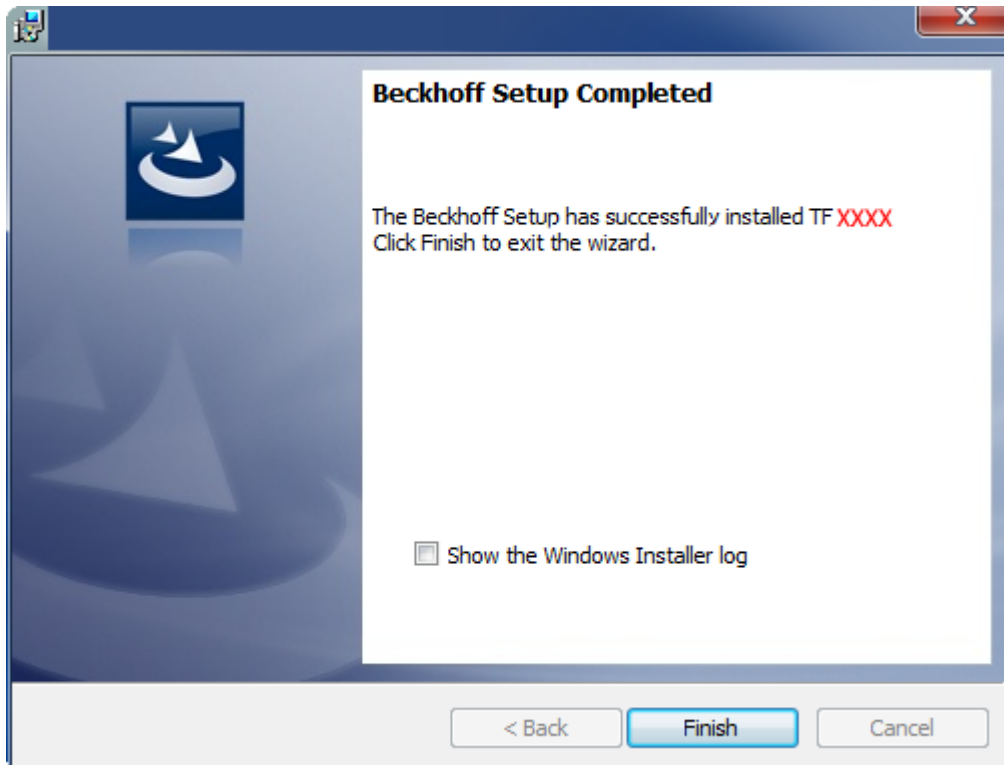


⇒ Ein Dialog weist Sie darauf hin, dass das TwinCAT-System für die weitere Installation gestoppt werden muss.

6. Bestätigen Sie den Dialog mit **Yes**.



7. Wählen Sie **Finish**, um das Setup zu beenden.



⇒ Die TwinCAT 3 Function wurde erfolgreich installiert und kann lizenziert werden (siehe [Lizenzierung \[► 12\]](#)).

3.3 Lizenzierung

Die TwinCAT 3 Function ist als Vollversion oder als 7-Tage-Testversion freischaltbar. Beide Lizenztypen sind über die TwinCAT-3-Entwicklungsumgebung (XAE) aktivierbar.

Lizenzierung der Vollversion einer TwinCAT 3 Function

Die Beschreibung der Lizenzierung einer Vollversion finden Sie im Beckhoff Information System in der Dokumentation „[TwinCAT 3 Lizenzierung](#)“.

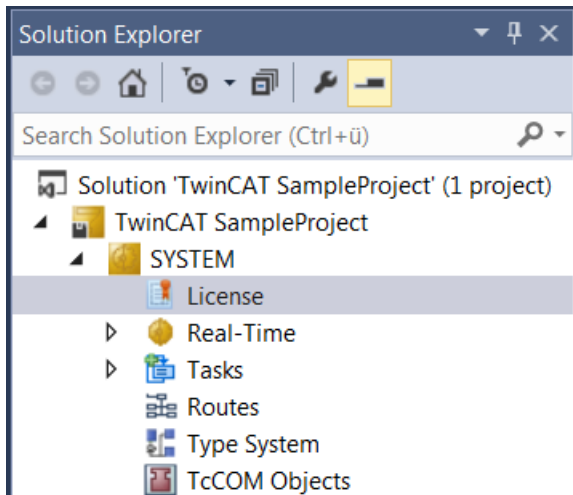
Lizenzierung der 7-Tage-Testversion einer TwinCAT 3 Function



Eine 7-Tage-Testversion kann nicht für einen [TwinCAT-3-Lizenz-Dongle](#) freigeschaltet werden.

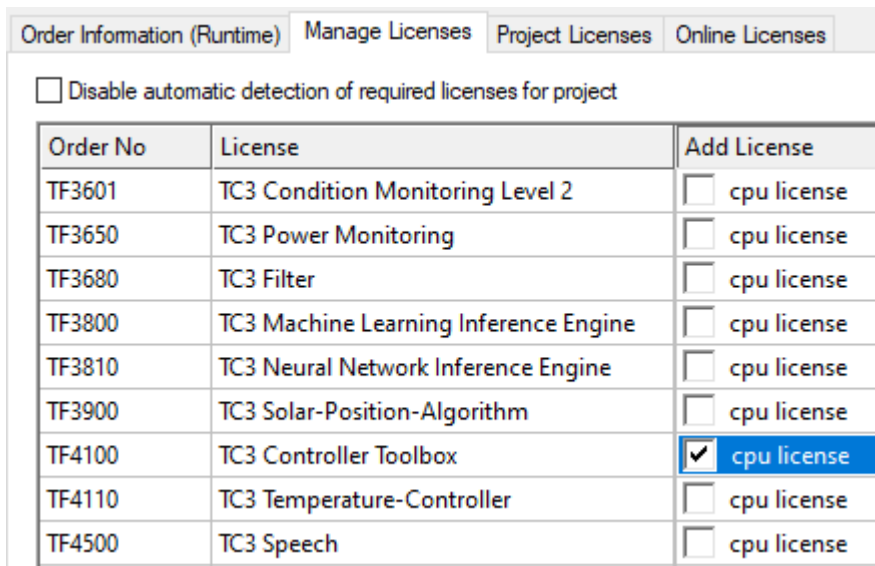
1. Starten Sie die TwinCAT-3-Entwicklungsumgebung (XAE).
2. Öffnen Sie ein bestehendes TwinCAT-3-Projekt oder legen Sie ein neues Projekt an.
3. Wenn Sie die Lizenz für ein Remote-Gerät aktivieren wollen, stellen Sie das gewünschte Zielsystem ein. Wählen Sie dazu in der Symbolleiste in der Drop-down-Liste **Choose Target System** das Zielsystem aus.
 - ⇒ Die Lizenzierungseinstellungen beziehen sich immer auf das eingestellte Zielsystem. Mit der Aktivierung des Projekts auf dem Zielsystem werden automatisch auch die zugehörigen TwinCAT-3-Lizenzen auf dieses System kopiert.

4. Klicken Sie im **Solution Explorer** im Teilbaum **SYSTEM** doppelt auf **License**.



⇒ Der TwinCAT-3-Lizenzmanager öffnet sich.

5. Öffnen Sie die Registerkarte **Manage Licenses**. Aktivieren Sie in der Spalte **Add License** das Auswahlkästchen für die Lizenz, die Sie Ihrem Projekt hinzufügen möchten (z. B. „TF4100 TC3 Controller Toolbox“).



6. Öffnen Sie die Registerkarte **Order Information (Runtime)**.

⇒ In der tabellarischen Übersicht der Lizenzen wird die zuvor ausgewählte Lizenz mit dem Status „missing“ angezeigt.

7. Klicken Sie auf **7 Days Trial License...**, um die 7-Tage-Testlizenz zu aktivieren.

The screenshot shows the 'License Management' window with the following sections:

- Order Information (Runtime)**: Includes tabs for 'Manage Licenses', 'Project Licenses', and 'Online Licenses'. Below are fields for 'License Device' (Target (Hardware Id)), 'System Id' (2DB25408-B4CD-81DF-5488-6A3D9B49EF19), and 'Platform' (other (91)).
- License Request**: Includes a 'Provider' dropdown (Beckhoff Automation), 'License Id', 'Customer Id', and a 'Comment' field. A 'Generate File...' button is also present.
- License Activation**: This section is highlighted with a red box and contains two buttons: '7 Days Trial License...' and 'License Response File...'.

⇒ Es öffnet sich ein Dialog, der Sie auffordert, den im Dialog angezeigten Sicherheitscode einzugeben.

The 'Enter Security Code' dialog box contains the following elements:

- Title: Enter Security Code
- Text: Please type the following 5 characters:
- Code display: Kg8T4
- Input field: A text box with a red border, currently empty.
- Buttons: 'OK' (highlighted with a red box) and 'Cancel'.

8. Geben Sie den Code genauso ein, wie er angezeigt wird, und bestätigen Sie ihn.

9. Bestätigen Sie den nachfolgenden Dialog, der Sie auf die erfolgreiche Aktivierung hinweist.

⇒ In der tabellarischen Übersicht der Lizenzen gibt der Lizenzstatus nun das Ablaufdatum der Lizenz an.

10. Starten Sie das TwinCAT-System neu.

⇒ Die 7-Tage-Testversion ist freigeschaltet.

4 Technische Einführung

4.1 RFID-Reader-Hardware

Allgemeine Hinweise



- Montagehinweise sowie Informationen zur Transponder-Reader-Handhabung und Lesegeschwindigkeiten etc. entnehmen Sie bitte den herstellereigenen Produkthandbüchern.
- Teilweise wird von den RFID Readern ein externer Trigger oder ein Schaltausgang angeboten. Dieser muss für die Funktionalität der TwinCAT-RFID-Bibliothek nicht verwendet werden.
- Die TwinCAT-RFID-Bibliothek bildet nicht den kompletten Leistungsumfang der herstellereigenen RFID-Kommunikationsprotokolle ab. Weitere Informationen dazu finden Sie in der Beschreibung des Befehlssatzes des Bibliotheksbausteins (siehe [RFID-Befehlssatz \[► 21\]](#)). Ergänzend kann auf die integrierte Möglichkeit zurückgegriffen werden, Rohdaten zu senden und zu empfangen (siehe Befehl `eRFC_Send_RawData`).

RFID-Reader-Modelle

Die TwinCAT-RFID-Bibliothek unterstützt unterschiedliche RFID-Reader-Modelle.

Die nachfolgende Tabelle gibt an, welche RFID-Reader-Modelle welcher Hersteller unterstützt werden. Bei den Abbildungen handelt es sich um Symbolfotos von RFID-Leser-Modellen. Diese können somit von den angegebenen unterstützten Modellen abweichen. Außerdem wird nicht jedes unterstützte Modell als Foto abgebildet. Veraltete Firmware-Versionen seitens der Reader werden teilweise nicht unterstützt.

RFID-Reader-Hersteller	RFID-Reader-Modell	Symbolfoto
Balluff	BIS M-400-007 (RS232) BIS M-401-007 (RS232) BIS L-6000-007 (2 read heads) (RS232) BIS L-6020-007 (2 read heads) (RS232)	 <p data-bbox="1098 808 1390 835">Bildnachweis: BALLUFF</p>
Baltech	ID-engine SD-M1415-ANT1F (RS232 or USB) ID-engine SD-LP-ANT1F (RS232 or USB) ID-engine PAD M1415 (USB) ID-engine ZM-L2M-U2-A1 (RS232)	
Deister electronic	RDL90 (deBus) (RS232, RS485) UDL 500 (deBus) (RS485) PRM5M/2V (deBus) (RS232, RS485)	

RFID-Reader-Hersteller	RFID-Reader-Modell	Symbolfoto
Leuze electronic	RFM12 (SL200) (RS232) RFM32 (SL200) (RS232) RFM32ex (SL200) (RS232)	
Pepperl+Fuchs	IDENTControl Compact (2 read heads) [IC-KP2-2HRX-2V1] (RS232) IDENTControl (4 read heads) [IC-KP-R2-V1] (RS232)	

Die nachfolgende Tabelle gibt an, welche RFID-Reader-Modelle laut Herstellerbeschreibung und -protokoll kompatibel sind. Die Kompatibilität der gelisteten Modelle sowie sonstiger Modelle ist jedoch nicht von Beckhoff bestätigt. Die Geräte werden nicht offiziell unterstützt. Vor Verwendung wird eine Kontaktaufnahme zu Beckhoff Automation empfohlen.

RFID-Reader-Hersteller	Reader-Modelle
Balluff	BIS M-6000
Baltech	ID-engine series (BRP)
Deister electronic	RDL30; RDL150; RDL160; UDL 50; UDL 100; UDL 120; UDL 150; UDL 160; PRM5
Leuze electronic	RFM62 (SL200)
Pepperl+Fuchs	IDENTControl Compact (1 read head)

Eine Unterstützung weiterer Beckhoff Automation nicht bekannter Modelle der oben genannten Hersteller ist möglicherweise implizit gegeben. Laut Deister electronic ist in weiteren Modellen dasselbe Protokoll (deBus) implementiert. Eine Verwendung dieser Modelle ist ggf. nur mit beschränktem Funktionsumfang möglich.

Des Weiteren bieten manche Hersteller eigene Software an, um ihre Geräte für Beckhoff TwinCAT-Systeme zugänglich zu machen.

● TwinCAT-SPS-Bibliothek „Serielle Kommunikation“

I Weitere RFID Reader werden mit der TwinCAT-SPS-Bibliothek „Serielle Kommunikation“ unterstützt. Mit dieser Bibliothek ist es möglich, Datenbytes mit einem beliebigen seriellen Gerät auszutauschen.

Diese Alternative zur TwinCAT-RFID-Bibliothek kann bei read-only RFID Readern sinnvoll sein. So können nicht unterstützte Geräte dennoch mit TwinCAT an einer Beckhoff-Steuerung verwendet werden. Wenn allein die Seriennummer des Transponders erforderlich ist und diese autark vom RFID-Gerät gesendet wird, ist der Aufwand einer Auswertung der empfangenen Bytes überschaubar.

i Verwendung vom Beckhoff Compact RFID Reader (iDTRONIC)

Der Beckhoff Compact RFID Reader ist eine Möglichkeit in einer Tastererweiterung für Beckhoff Multitouch-Control-Panels ein RFID Gerät des Herstellers iDTRONIC zu integrieren. Entgegen der Integration von Baltech Geräten in Beckhoff Panels wird hierbei nicht die SPS-Bibliothek Tc2_RFID (TF6600) sondern allein die SPS-Bibliothek Tc2_SerialCom (TF6340) verwendet.

Transpondertypen

Eine vollständige Liste aller unterstützten Transpondertypen entnehmen Sie bitte dem Handbuch des jeweiligen RFID Readers. Klären Sie bei Bedarf mit dem Hersteller der RFID Reader oder Transponder, welcher Transpondertyp für die Applikation sinnvoll ist.

Die TwinCAT-RFID-Bibliothek arbeitet mit den Daten, die aus der seriellen Schnittstelle bezogen werden. Das serielle Übertragungsprotokoll des Herstellers ist demnach entscheidend für eine Unterstützung durch die Bibliothek. Die verwendete Funkfrequenz ist beispielsweise irrelevant.

Die nachfolgende Tabelle gibt exemplarisch an, welche Transpondertypen bei den jeweiligen RFID-Reader-Modellen laut Hersteller unterstützt werden. Diese Liste ist nicht vollständig. Vollständige und weitergehende Informationen erhalten Sie beim Hersteller des RFID-Reader-Modells. Beachten Sie, dass manche RFID Reader nur Transponder mit bestimmten Herstellerkennungen akzeptieren. Auf diese Beschränkung kann leider kein Einfluss genommen werden.

RFID-Reader-Modell	RFID-Transpondertypen
Balluff M-401	[13.56 MHz] Fujitsu MB89R118; I-Code SLI; Infineon My-D SRF55(1024 Bytes); Mifare Classic (752 Bytes); TI TagIT HFI (256 Bytes), ...
Balluff L-6000	[125KHz]
Baltech ID-engine SD ANT1F (M1415, LP)	[13.56 MHz] Infineon My-D, Legic Prime, Mifare Classic, ...
Deister electronic RDL90	[13.56 MHz] I-Code SLI; Infineon My-D SRF55(1024 Bytes), ...
Deister electronic UDL 500	[868 MHz] EPCclass1gen2 (12 Bytes), ...
Deister electronic PRM5	[13.56 MHz] Mifare Classic (752 Bytes), ...
Leuze electronic RFM12, RFM32, RFM32ex	[13.56 MHz] I-Code SLI; Infineon My-D SRF55(1024 Bytes); TI TagIT HFI (256 Bytes), ...
Pepperl+Fuchs IDENTControl Compact	[125 KHz; 250 KHz; 13.56 MHz; 2.45 GHz (depends on read head)] I-Code SLI; Fujitsu MB89R118; TI TagIT HFI; Infineon My-D SRF55, ...

Diese Transpondertypen sind zudem in weiteren Speichergrößen erhältlich. Eine Kompatibilität ist hardwareabhängig und wird nicht garantiert. Ein Test wird empfohlen.

Teilweise sind spezielle werksseitige Programmierungen der Transponder möglich. Diese haben keinen Einfluss auf das Protokoll und müssen demnach applikationsabhängig mit Rücksprache zum Hersteller entschieden werden.

Spezifische Transponderparameter, die in der TwinCAT-RFID-Bibliothek verwendet werden, können vom Nutzer bei Verwendung eines speziellen Transponders angepasst werden (siehe [ST_RFID_AccessData](#) [► 52]).

Transponder werden bis zu einer maximalen Größe von 64 Kilobytes seitens der TwinCAT-RFID-Bibliothek unterstützt.

Frequency	Transponder Types	HF standards	range	metallic influence	fluid influence	data rate	radio interaction	hardware positioning	temperature influence
LF 125-135 KHz	...	ISO 11784/5, ISO 14223, ISO 18000-2	< 2m	++	+	-	-	++	++
HF 13,56 MHz	Fujitsu MB89R118, I-Code SLI, Infineon My-D, Legic, Mifare, TI TagIT HFI, ...	ISO 14443, ISO 15693, ISO 18000-3	< 1m	+	+	+	+	++	+
UHF 865-868 MHz (EU), 902-928 MHz (USA)	EPCclass1gen2, ..	ISO 18000-6, EPC-Gen2	< 10m	-	-	+	+	+	+
MW 2,45 GHz	...		< 12m	-	-	++	++	-	-

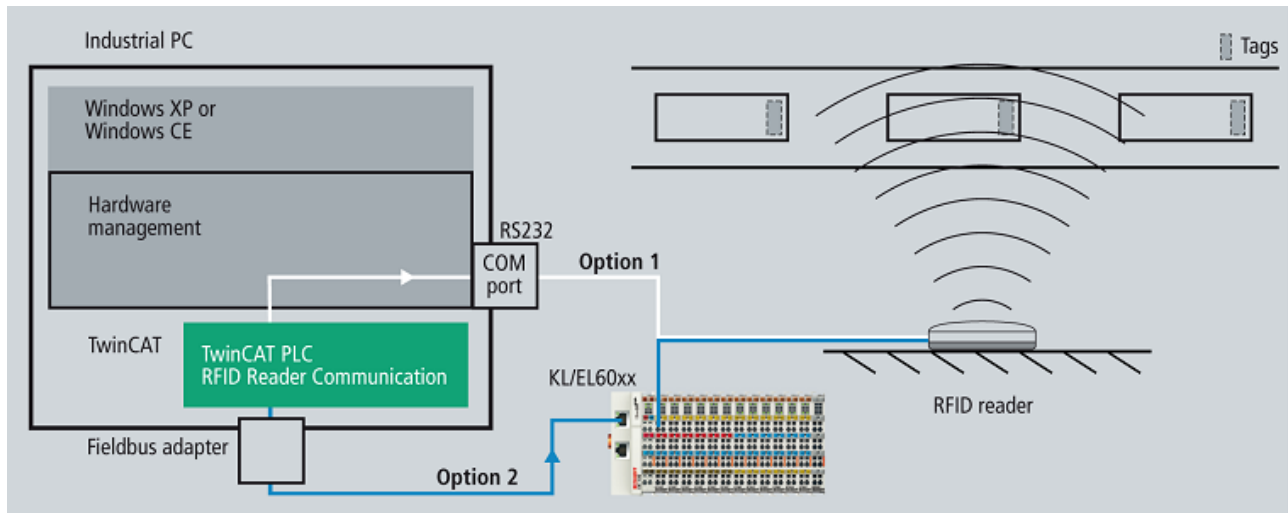
[++ very good; + good; - bad]

4.2 RFID-Reader-Anbindung

Alle mittels dieser SPS-Bibliothek unterstützten RFID Reader werden über serielle Kommunikationsschnittstellen mit der Steuerung verbunden (RS 232, RS 422, RS 485 und virtuelle serielle COM Ports).

Dazu können folgende Beckhoff Produkte genutzt werden:

- Serielle EtherCAT-Klemmen: EL6001, EL6002, EL6021, ...
- Serielle K-Bus-Klemmen: KL6001/KL6031, KL6021, ...
- COM Port eines beliebigen IPC und Embedded-PC mit TwinCAT-System



Je RFID Reader muss eine separate Verbindung zu einer separaten Klemme erfolgen. Eine Unterstützung mehrerer RFID Reader an einem RS485-Netz ist mit der TwinCAT-RFID-Bibliothek vorerst nicht gegeben.

Einrichten der seriellen Kommunikation in TwinCAT 3 XAE

Der serielle Datenaustausch wird mit den Bausteinen der TwinCAT-SPS-Bibliothek Tc2_SerialCom eingerichtet.

Legen Sie einen Sendebuffer sowie einen Empfangsbuffer vom Typ „ComBuffer“ an. Dies kann global geschehen, muss aber nicht zwangsläufig. Legen Sie außerdem zwei Datenstrukturen an, wie Sie im TwinCAT System Manager zur seriellen Kommunikation verwendet werden.

Wenn der COM Port verwendet wird, sieht dies wie folgt aus:

```
gPcComRxBuffer      : ComBuffer;
gPcComTxBuffer      : ComBuffer;
PcComInData         AT %I* : PcComInData;
PcComOutData        AT %Q* : PcComOutData;
```

Neben PcComInData/PcComOutData sind bei Verwendung einer seriellen Klemme EL6inData22B/EL6outData22B sowie KL6inData5B/ KL6outData5B und andere Datentypen möglich.

Verlinken Sie die Strukturen im TwinCAT System Manager mit den Kanälen der seriellen Schnittstelle. Bei Verwendung des ComPorts müssen Sie hierzu im TwinCAT System Manager am IO-Gerät zusätzlich die Option **SyncMode** aktivieren. Die SPS-Variablen müssen im TwinCAT System Manager der richtigen (schnellen) Task zugeordnet und von dort passend verlinkt sein.

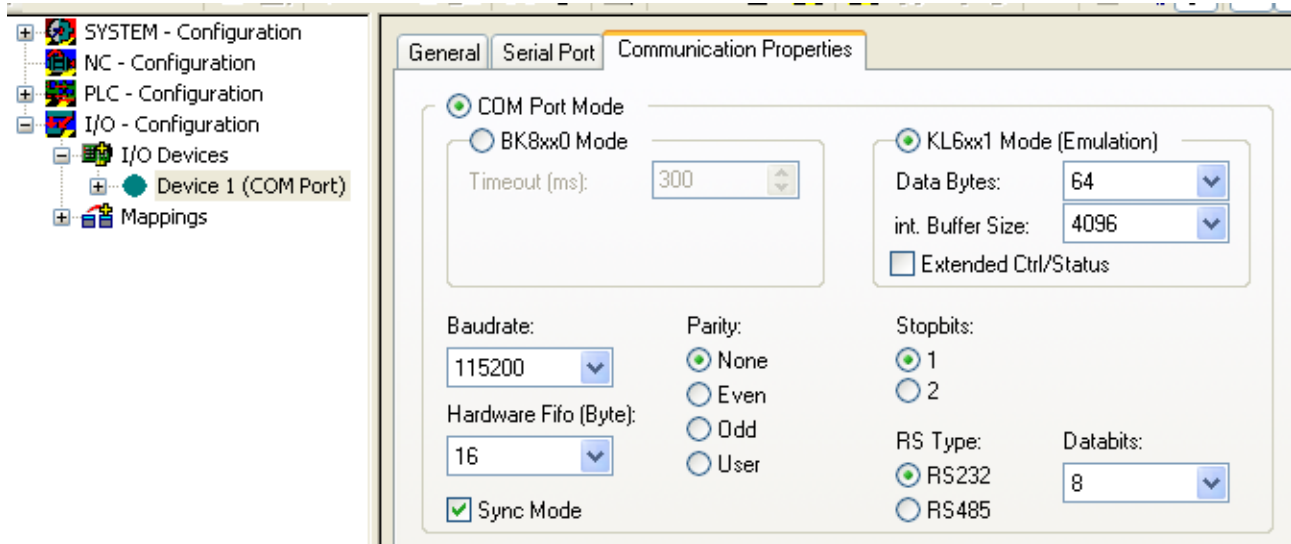
Legen Sie zur seriellen Kommunikation eine Instanz des SerialLineControl an. Diese muss in einer schnellen Task (<= 1 ms) zyklisch aufgerufen werden. Die nötige Taskzykluszeit ist abhängig von der Anwendung, der Datenmenge, der Baudrate und der Schnittstelle. Je nach Anwendung und Schnittstelle ist es oft sinnvoll, dies in einer zusätzlichen Task auszuführen, die schneller ist als die Task der Applikation.

Beispiel 1: Beim Anschluss eines RFID-Gerätes an einen COM Port und einer Baudrate von 115200 Baud ist eine Zykluszeit von 1 ms notwendig.

Beispiel 2: Beim Anschluss eines RFID-Gerätes an eine EL6001 und einer Baudrate von 9600 Baud ist eine Zykluszeit von maximal 6 ms notwendig.

Weitere Informationen sowie Erläuterungen zur Verwendung virtueller COM Ports finden Sie in der Dokumentation der SPS-Bibliothek „Serielle Kommunikation“.

Exemplarische Darstellung der COM-Port-Einstellungen im TwinCAT System Manager:



Der Aufruf des SerialLineControl ist im Folgenden exemplarisch dargestellt.

Aufruf als StructuredText im Falle der COM-Port-Verwendung:

```
LineControl (
  Mode      := SERIALLINEMODE_PC_COM_PORT,
  pComIn    := ADR (PcComInData),
  pComOut   := ADR (PcComOutData),
  SizeComIn := SIZEOF (PcComInData),
  TxBuffer  := gPcComTxBuffer,
  RxBuffer  := gPcComRxBuffer
);
```

Aufruf als StructuredText im Falle der Verwendung einer EtherCAT-Klemme:

```
LineControl (
  Mode      := SERIALLINEMODE_EL6_22B,
  pComIn    := ADR (EL6ComInData),
  pComOut   := ADR (EL6ComOutData),
  SizeComIn := SIZEOF (EL6ComInData),
  TxBuffer  := gEL6ComTxBuffer,
  RxBuffer  := gEL6ComRxBuffer
);
```

Aufruf als StructuredText im Falle der Verwendung einer K-Bus-Klemme:

```
KL6Config3 (
  Execute      := bConfig3,
  Mode         := SERIALLINEMODE_KL6_5B_STANDARD,
  Baudrate     := 9600,
  NoDatabits   := 8,
  Parity       := 0,
  Stopbits     := 1,
  Handshake    := RS485_FULLDUPLEX,
  ContinuousMode := FALSE,
  pComIn       := ADR (KlComInData3),
  pComOut      := ADR (KlComOutData3),
  SizeComIn    := SIZEOF (KlComInData3),
  Busy => bConfig3Act,
  Done => bConfig3Done,
  Error => bConfig3Error
);
IF NOT KL6Config3.Busy THEN
  bConfig3 := FALSE;
```

```
LineControl3(  
  Mode      := SERIALLINEMODE_KL6_5B_STANDARD,  
  pComIn    := ADR(KlComInData3),  
  pComOut   := ADR(KlComOutData3),  
  SizeComIn := SIZEOF(KlComInData3),  
  TxBuffer  := gKlComTxBuffer3,  
  RxBuffer  := gKlComRxBuffer3  
);  
END_IF
```

4.3 RFID-Befehlssatz

Die nachfolgende Matrix listet den zur Verfügung stehenden RFID-Befehlssatz auf.

Wegen der grundsätzlichen Unterschiede der verschiedenen RFID-Reader-Modelle können nicht alle Befehle bei jedem Modell zur Verfügung gestellt werden.

Die Komplexität mancher proprietärer Protokolle macht es erforderlich, dass nicht jeder Befehl oder jede detaillierte Parametriermöglichkeit auch über die TwinCAT-SPS-Bibliothek gegeben sein kann. In Einzelfällen kann deshalb auf die weniger komfortable Kommunikationsmöglichkeit mittels dem angebotenen LowLevel Interface zurückgegriffen werden. Informationen dazu finden Sie in der Beschreibung des Befehls [SendRawData \[► 25\]](#) und im Abschnitt [Low-Level-Kommunikation \[► 41\]](#).

Informationen zu den Eigenschaften und Eigenarten der proprietären Protokolle können zu jedem Modell vom Hersteller bezogen werden und werden meist mit dem Gerät mitgeliefert. Diese Protokoll-Spezifikationen sollten zumindest beim Anwender vorhanden sein, um Detailfragen recherchieren zu können und Eigenarten des RFID Readers nachzulesen. Auf die Eigenarten der einzelnen RFID Reader wird bereits, soweit möglich, an den speziellen Stellen innerhalb dieser Dokumentation hingewiesen. Allerdings bleibt der Hersteller der RFID-Geräte weiterhin selbst in der Verantwortung seine Geräte zu beschreiben und deren Verhalten und Eigenschaften zu gewährleisten. Eine detaillierte Beschreibung jedes Befehls und des speziellen Verhaltens des RFID Readers ist in den proprietären Protokoll-Spezifikationen gegeben. Welcher herstellerproprietäre Befehl dem hier gelisteten Befehl entspricht, wird im Folgenden jeweils *kursiv* angegeben. Details können Sie ebenso der Ausgangsstruktur [ST RFID RawData \[► 45\]](#) des Funktionsbausteins [FB RFIDReader \[► 35\]](#) entnehmen.

Command	Balluff BIS M-40x BIS L-60x0	Baltech IDE SD ANT1F	Deister electronic RDL90	Deister electronic UDL 500	Deister electronic PRM5M/2V	Leuze electronic RFM12; RFM32; RFM32ex	Pepperl+Fuchs IDENTControl Compact
GetReaderVersion [▶ 22]		x	x	x	x	x	x
GetConfig [▶ 22]			x	x		x	x
SetConfig [▶ 23]		x	x	x		x	
GetInventory [▶ 23]	x	x	x			x	x
Polling [▶ 23]			x	x	x		
TriggerOn [▶ 23]			x	x		x	
TriggerOff [▶ 23]			x	x		x	
AbortCommand [▶ 24]			x			x	x
ResetReader [▶ 24]	x	x	x	x	x	x	x
ReadBlock [▶ 24]	x	x	x	x		x	x
WriteBlock [▶ 24]	x	x	x	x		x	x
OutputOn [▶ 25]			x	x		x	
OutputOff [▶ 25]			x	x		x	
FieldOn [▶ 25]		x	x	x		x	
FieldOff [▶ 25]		x	x	x		x	
SendRawData [▶ 25]	x	x	x	x	x	x	x
ChangeDCType [▶ 25]							x

Diese Liste ist analog zur Enumeration [E_RFID_Command](#) [\[▶ 65\]](#) in der TwinCAT-RFID-Bibliothek. Eine erfolgreiche Bearbeitung des angefragten Befehls durch den RFID Reader erkennen Sie an den Statusausgängen des Funktionsbausteins sowie an der jeweiligen Response. Eine Liste möglicher Responses finden Sie in der Beschreibung der Enumeration [E_RFID_Response](#) [\[▶ 65\]](#).

Im Folgenden werden die Befehle im Einzelnen erläutert:

GetReaderVersion [16#01]

Mit diesem Befehl können Informationen zum RFID Reader abgefragt werden. Je nach Verfügbarkeit wird die Modellbezeichnung, die Hard- und Softwareversion des Readers etc. am Bausteinausgang in der Struktur [ST_RFID_ReaderInfo](#) [\[▶ 43\]](#) ausgegeben.

Entsprechung im proprietären Protokoll:

Deister: 0x02

Leuze: 'V'

Pepperl+Fuchs: 'VE'

Baltech: System GetInfo

GetConfig [16#02]

Mit diesem Befehl wird die aktuelle Konfiguration des RFID Readers abgefragt. Alle relevanten empfangenen Parameter werden in der Beschreibung der Struktur [ST_RFID_Config](#) [\[▶ 52\]](#) erläutert. Weitere Informationen sind im Abschnitt [Konfiguration](#) [\[▶ 40\]](#) zusammengefasst.

Entsprechung im proprietären Protokoll:

Deister: 0x09

Leuze: 'G'

Pepperl+Fuchs: 'GS'

SetConfig [16#03]

Parametrierte Konfigurationseinstellungen können auf den RFID Reader übertragen werden. Nähere Informationen zur möglichen Konfiguration des RFID Readers finden Sie in der Beschreibung der Struktur [ST_RFID_ConfigIn](#) [► 50].

Es wird empfohlen, nach einem Konfigurationsbefehl erneut die aktuelle Konfiguration des Readers mittels des Befehls [GetConfig](#) [► 22] abzufragen. Weitere Informationen sind im Abschnitt [Konfiguration](#) [► 40] zusammengefasst.

Entsprechung im proprietären Protokoll:

Baltech: System CfgWriteTLVBlock

Deister: 0x09

Leuze: 'C'

GetInventory [16#04]

Mit diesem Befehl werden der Typ und die Seriennummer eines aktuell im Lesefeld befindlichen Transponders abgefragt. Wenn kein Transponder gefunden wird, folgt eine entsprechende Response.

Pepperl+Fuchs: Mit dem Parameter `iHeadNumber` in der Struktur [ST_RFID_Control](#) [► 45] wird festgelegt, für welchen Lesekopf der Befehl auszuführen ist.

Entsprechung im proprietären Protokoll:

Balluff: 'U'

Deister: 0x82

Leuze: 'I'

Pepperl+Fuchs: 'SF' & 'EF'

Baltech: VHLSelect + VHLGetSnr

Polling [16#05]

Mit diesem Befehl werden Informationen aus dem Stack des RFID Readers abgefragt. Dabei kann es sich beispielsweise um die Seriennummer des letzten Transponders handeln. Beachten Sie, dass unterschiedliche RFID Reader verschieden große Stacks besitzen und teils nur eine Nachricht gespeichert wird.

Präsenzerkennung: Wenn dies nicht über einen Konfigurationsparameter eingestellt werden kann, ist es nötig, den Reader mittels zyklischem Polling-Befehl lesebereit zu halten, sodass ein Transponder in Reichweite automatisch detektiert wird.

Entsprechung im proprietären Protokoll:

Deister: 0x0B

TriggerOn [16#06]

Wenn der Trigger Mode aktiv ist, kann mit diesem Befehl ein Software Trigger anstatt eines Hardware Triggers ausgelöst werden. Das darauffolgende Antworttelegramm wird von dem Funktionsbaustein der TwinCAT-RFID-Bibliothek empfangen. Eine Zuweisung von gelesenen Transponderdaten ist in dem Fall nicht gegeben. Die empfangenen Rohdaten können zur weiteren Verarbeitung dem Bausteininterface entnommen werden.

Entsprechung im proprietären Protokoll:

Deister: 0x85

Leuze: '+'

TriggerOff [16#07]

Siehe [TriggerOn](#) [► 23].

Entsprechung im proprietären Protokoll:

Deister: 0x85

Leuze: '-'

AbortCommand [16#08]

Wenn ein Befehl seitens des RFID Readers in Bearbeitung ist, wird er mit diesem Befehl abgebrochen.

Pepperl+Fuchs: Mit dem Parameter *iHeadNumber* in der Struktur [ST_RFID_Control \[► 45\]](#) wird festgelegt für welchen Lesekopf der Befehl auszuführen ist.

Entsprechung im proprietären Protokoll:

Leuze: 'H'

Pepperl+Fuchs: 'QU'

ResetReader [16#09]

Dieser Befehl veranlasst den RFID Reader, ein Reset durchzuführen.

Entsprechung im proprietären Protokoll:

Balluff: 'Q'

Deister: 0x01

Leuze: 'R'

Pepperl+Fuchs: 'RS'

Baltech: System Reset

ReadBlock [16#0A]

Mit diesem Befehl wird eine bestimmte Anzahl von Datenbytes in Form von Blöcken definierter Größe aus dem Speicher des Transponders gelesen.

Für diesen Befehl ist die Übergabe der Eingangsstruktur [ST_RFID_AccessData \[► 52\]](#) nötig.

Bevor Daten vom Transponder gelesen werden, ist es üblich den Transponder zu erkennen und auszuwählen (siehe Befehl [GetInventory \[► 23\]](#)).

Pepperl+Fuchs: Mit dem Parameter *iHeadNumber* in der Struktur [ST_RFID_Control \[► 45\]](#) wird festgelegt, für welchen Lesekopf der Befehl auszuführen ist.

Entsprechung im proprietären Protokoll:

Balluff: 'L'

Deister: 0x83

Leuze: 'N'

Pepperl+Fuchs: 'SR' & 'ER'

Baltech: VHLRead

WriteBlock [16#0B]

Mit diesem Befehl wird eine bestimmte Anzahl von Datenbytes in Form von Blöcken definierter Größe in den Speicher des Transponders geschrieben.

Für diesen Befehl ist die Übergabe der Eingangsstruktur [ST_RFID_AccessData \[► 52\]](#) nötig.

Bevor Daten auf einen Transponder geschrieben werden, ist es üblich den Transponder zu erkennen und auszuwählen (siehe Befehl [GetInventory \[► 23\]](#)).

Pepperl+Fuchs: Mit dem Parameter *iHeadNumber* in der Struktur [ST_RFID_Control \[► 45\]](#) wird festgelegt, für welchen Lesekopf der Befehl auszuführen ist.

Entsprechung im proprietären Protokoll:

Balluff: 'P'

Deister: 0x84

Leuze: 'W'

Pepperl+Fuchs: 'SW' & 'EW'

Baltech: VHLWrite

OutputOn [16#0C]

Der Befehl setzt den Schaltausgang des RFID Readers permanent auf TRUE. Dies ist nur möglich, wenn der Schaltausgang nicht per Konfiguration automatisch angesprochen wird.

Entsprechung im proprietären Protokoll:

Deister: 0x0F

Leuze: 'A0FF'

OutputOff [16#0D]

Der Befehl setzt den Schaltausgang des RFID Readers permanent auf FALSE. Dies ist nur möglich, wenn der Schaltausgang nicht per Konfiguration automatisch angesprochen wird.

Entsprechung im proprietären Protokoll:

Deister: 0x0F

Leuze: 'A000'

FieldOn [16#0E]

Mit diesem Befehl kann das RFID-Feld angeschaltet werden.

Entsprechung im proprietären Protokoll:

Deister: 0x81

Leuze: 'F1'

Baltech: System HFRreset

FieldOff

Mit diesem Befehl kann das RFID-Feld ausgeschaltet werden. Je nach RFID-Reader-Modell wird das Feld bei Trigger o.a. automatisch wieder aktiv.

Entsprechung im proprietären Protokoll:

Deister: 0x81

Leuze: 'F2'

Baltech: System HFRreset

SendRawData [16#10]

Mit diesem Befehl kann der RFID-Funktionsbaustein als Low-Level-Schnittstelle genutzt werden. Die zu übermittelnden Daten werden in der [Control-Struktur \[▶ 45\]](#) als Pointer übergeben. Bibliotheksintern wird ein Telegramm zusammengesetzt und versendet. Es können auf diese Art und Weise beliebige Daten an den RFID Reader gesendet werden. Die daraufhin empfangenen Daten stehen am Ausgang des Funktionsbausteins in der [Rohdatenstruktur \[▶ 45\]](#) als adressiertes Datenfeld zur Verfügung. Weitere Informationen zum Ablauf finden Sie im Abschnitt [Low-Level-Kommunikation \[▶ 41\]](#).

Bei Nutzung des Befehls [SendRawData \[▶ 25\]](#) kann eine Auswertung der empfangenen Antwort nicht garantiert werden.

Beispiel: Wird ein Lesebefehl manuell als Bytefolge mittels des Befehls [SendRawData](#) versandt, so werden empfangene Transponderdaten nicht auf eine in [ST_RFID_AccessData \[▶ 52\]](#) angegebene Adresse geschrieben. Eine Auswertung/Speicherung der Daten sollte demnach auch manuell mithilfe der immer angegebenen [Rohdatenstruktur \[▶ 45\]](#) geschehen.

ChangeDCType [16#11]

Mit diesem Befehl kann der Transpondertyp am Lesekopf eingestellt werden. Dazu wird mittels [iDCType](#) in [ST_RFID_Control \[▶ 45\]](#) der Typ angegeben.

Pepperl+Fuchs: Mit dem Parameter [iHeadNumber](#) in der Struktur [ST_RFID_Control \[▶ 45\]](#) wird festgelegt, für welchen Lesekopf der Befehl auszuführen ist.

Entsprechung im proprietären Protokoll:

Pepperl+Fuchs: 'CT'

5 Konfiguration

5.1 RFID-Reader-Einstellungen und -Handhabung

In den nachfolgenden Abschnitten werden die einzelnen RFID-Reader-Modelle gemäß den Gerätehersteller beschrieben. Für jedes Gerät werden die nötigen Einstellungen sowie die Handhabung erläutert.

5.1.1 Balluff

RFID-Reader-Einstellungen

Für eine reibungslose Kommunikation zwischen Steuerung und RFID Reader müssen manche Einstellungen vor Systemstart vorgenommen werden. Hierzu zählt beispielsweise die Baudrate der seriellen Kommunikation. Um diese Einstellungen auf den RFID Reader zu übertragen, kann ein proprietäres Tool des RFID-Reader-Herstellers nötig sein.

Für alle unterstützten RFID-Reader-Modelle haben sich die folgenden Standardeinstellungen der Datenübertragung bewährt:

Einstellung	Wert
Baudrate (RS232 und RS485)	9600 Baud
Parity Bit	none
Datenbits	8
Stopbits	1

Bei Bedarf können je nach Hardware auch andere Parameter eingestellt oder die Werkseinstellungen des RFID Readers verwendet werden. Diese müssen dann auch in der softwareseitigen Reader-Anbindung übernommen werden (siehe [RFID-Reader-Anbindung](#) [► 19]).

Mittels der proprietären Tools müssen vor Systemstart folgende spezielle Einstellungen parametrierbar werden:

Einstellung	Wert
Parameter der Datenübertragung (s.o.)	Einstellung in Analogie zu den im SPS-Programm gewählten Werten
Protokolltyp - Telegramm Endekennung	LF CR
Datenträgertyp	All Types (oder Einstellung je nach Bedarf)
CT-Daten sofort senden	deaktiviert (oder aktiviert - es erfolgt jedoch keine Auswertung)
Dynamik-Betrieb	deaktiviert
Einschaltmeldung senden	deaktiviert (oder aktiviert - es erfolgt jedoch keine Auswertung)
CRC16 Datenprüfung	deaktiviert
Typ und serial number bei CT pres.	deaktiviert (oder aktiviert, je nach Bedarf)

Wenn **Typ und serial number bei CT pres.** aktiviert ist, so sendet der RFID Reader automatisch den Transponder-Typ und dessen Seriennummer, sobald ein Transponder erkannt wurde. Wenn ein Befehl unverzüglich nach der Detektion eines Transponders und Erhalt dieser eingestellten Meldung abgesendet wird, so kann eine korrekte Zuordnung der Art der folgenden Response und eine zugehörige Auswertung nicht garantiert werden. Es wird empfohlen, vorhandene Transponder manuell mit dem Befehl [GetInventory](#) [► 23] abzufragen. Andernfalls sollte zumindest eine kurze Wartezeit bis zum Absenden des Befehles eingehalten werden und der Aufbau einem Testzyklus unterzogen werden.

Wenn der RFID Reader so eingestellt ist, dass automatisch Telegramme vom Reader zur Steuerung gesendet werden (beispielsweise bei Detektion eines Transponders durch **Typ und serial number bei CT pres.**), muss Folgendes beachtet werden:

Die Endekennung (LF CR) wird in dem Fall als Suffix zur Erkennung von Telegrammen genutzt. Sobald diese 2 Bytes im Datenstrom erkannt werden, werden vorherige Daten zu einem Telegramm zusammengefasst. Ggf. führt dies zu einem Fehler und fehlender Auswertung des Telegramms. Sollte der

Fall auftreten können, dass die Endekennung in automatisch gesendeten Telegrammen innerhalb der Daten vorhanden ist, so muss anstatt der automatischen Übertragung eine Datenabfrage mittels Befehlsaufruf gewählt werden. Durch diese Maßnahme werden die Telegramme sicher erkannt.

RFID-Reader-Handhabung

Die Funktionsbausteine der Bibliothek unterstützen die Kommunikation von Balluff Readern zu Transponder mit 4-8 Bytes Seriennummer.

Bei Verwendung von Balluff RFID Readern wird die Seriennummer bei 13,56 Mhz Transpondern im Gesamten byteweise vom Bibliotheksbaustein gedreht. Dies geschieht, weil die ausgelesene Seriennummer eines Transponders andernfalls nicht mit der an einem anderen Reader ausgelesenen Seriennummer übereinstimmen würde. So lassen sich Geräte verschiedener Hersteller gemeinsam in einem Verbund betreiben.

Bei Verwendung eines Balluff BIS-L60x0:

- Die Variable `iDCType = 0` muss gesetzt werden (siehe Eingangsstruktur [stCtrl](#) [► 45]).
- Beim Aufruf des Befehls [GetInventory](#) [► 23] werden Informationen von beiden Leseköpfen über die serielle Schnittstelle zurückgeliefert. Ausgewertet und am Ausgang `stTransplInfo` ausgegeben wird jedoch nur die Information von dem per `stCtrl.iHeadNumber` ausgewählten Lesekopfes.
- Wenn **Typ und serial number bei CT pres.** aktiviert ist, so sendet der RFID Reader automatisch den Transponder-Typ und dessen Seriennummer, sobald ein Transponder erkannt wurde. Dies betrifft per default nur den ersten Lesekopf. Ein Umschalten auf den zweiten Lesekopf wird hierbei von der Bibliothek nicht direkt unterstützt. Des Weiteren kann die Nummer des Lesekopfes, an dem der Tag erkannt wurde, nicht zugewiesen werden (`iHeadNumber = 0`).

Hier sei darauf hingewiesen, dass nicht alle Eigenarten jedes unterstützten RFID-Reader-Modells hier genannt werden können. Für detaillierte Informationen wird auf die herstellereigenen Dokumentationen hingewiesen.

5.1.2 Baltech

Wenn ein unterstütztes Baltech RFID-Stand-Alone-Gerät verwendet wird, kann die TwinCAT-RFID-Bibliothek als Schnittstelle genutzt werden. Eine Alternative ist die Verwendung mit bestimmten Beckhoff Control-Panels oder Panel-PCs. In diesen Geräten kann als Option ein RFID Reader integriert werden. In diesem Fall wird ein SDK mitgeliefert, in dem sich die proprietären Dokumentationen befinden. Der Funktionsumfang der TwinCAT-Bibliothek ist in beiden Fällen derselbe.



RFID-Reader-Einstellungen

Für eine reibungslose Kommunikation zwischen Steuerung und RFID Reader müssen manche Einstellungen vor Systemstart vorgenommen werden. Hierzu zählt beispielsweise die Baudrate der seriellen Kommunikation. Um diese Einstellungen auf den RFID Reader zu übertragen, kann das proprietäre Tool „Baltech id-engine explorer“ des RFID-Reader-Herstellers verwendet werden. Ebenso kann mit dem Tool ein Funktionstest gemacht werden, um festzustellen, ob das RFID-Gerät erkannt und die Transponderkarten erkannt werden.

Die folgenden Standardeinstellungen der Datenübertragung haben sich bewährt:

Einstellung	Wert
Baudrate	115200 Baud
Parity Bit	none
Datenbits	8
Stopbit	1

Dies entspricht der Werkseinstellung der unterstützten Baltech RFID-Geräte. Bei Bedarf lassen sich auch andere Parameter einstellen. Diese müssen dann auch in der softwareseitigen Reader-Anbindung übernommen werden (siehe [RFID-Reader-Anbindung](#) [► 19]).

Die Baudrate der Lesegeräte kann mit dem Tool „Baltech id-engine explorer“ geändert werden (siehe Baltech Dokumentation: IdEngineExplorer.pdf).

Das Tool „Baltech id-engine explorer“ ist nur unter Windows XP lauffähig. Es ist nicht für Windows CE verfügbar. Somit ist die Baudrate unter Windows CE nicht konfigurierbar.

In der SPS wird eine schnelle Task benötigt, um die ankommenden Daten zu verarbeiten. Beim Anschluss des RFID-Gerätes an einen COM Port und einer Baudrate von 115200 Baud ist eine Zykluszeit von 1 ms notwendig (siehe [RFID-Reader-Anbindung](#) [► 19]).

Um die Baudrate aus der SPS zu konfigurieren, kann folgende Bytefolge [0x 1C 00 09 06 00 01 03 00 02 xx xx - wobei xx xx für die Baudrate steht, so beispielsweise 9600 Baud: 96 Einheiten a 100Baud → 0x 00 60] als Rohdatenblock übertragen werden. Näheres wird im Abschnitt [Low-Level-Kommunikation](#) [► 41] erläutert. Dies ist auch unter Windows CE möglich, sofern eine Übertragung mit der derzeitig eingestellten Baudrate möglich ist.

Verwendung vom virtuellen seriellen COM Port (USB)

Ist das Gerät mittels USB verbunden, so muss der passende Usb-To-Virtual-Com-Port-Treiber installiert sein. Handelt es sich um einen Beckhoff Panel-PC, so ist der Treiber bereits vorinstalliert. Ebenso beinhaltet das SDK des RFID-Gerätes den Treiber. Der virtuelle COM Port wird im Windows Gerätemanager angezeigt.

Die Kommunikation zum Treiber erfolgt über die Beckhoff TwinCAT Serielle Kommunikation. Jedoch wird kein entsprechendes Gerät im TwinCAT System Manager angelegt und auf eine dortige Verknüpfung verzichtet. Weitere Informationen finden Sie in der Dokumentation der SPS-Bibliothek „Serielle Kommunikation“.

RFID-Reader-Handhabung

Die Bibliothek unterstützt die üblichen Standardeinstellungen der Baltech Kommunikation. Als „Operational Mode“ wird der Mode „Host Operation“ unterstützt. Andere Modi werden nicht unterstützt. Der Zugriff erfolgt intern über das BRP (Baltech Reader Protocol) im „Communication Mode“ „Normal Mode“. Wenn Rohdaten über die Low-Level-Kommunikationsmöglichkeit gesendet werden, achten Sie darauf, dass obige Einstellungen korrekt innerhalb des Frames angegeben werden.

Hier sei darauf hingewiesen, dass nicht alle Eigenarten jedes unterstützten RFID-Reader-Modells hier genannt werden können. Deshalb wird für detaillierte Informationen auf die herstellereigenen Dokumentationen hingewiesen.

Konfiguration

Wenn mit verschlüsselten Transponderkarten gearbeitet wird, so ist es nötig, dass derselbe Schlüssel auch im RFID-Gerät bekannt ist. Der Baltech RFID Reader wird einmal konfiguriert. Daraufhin muss der Schlüssel nicht mehr angegeben werden. Ebenso kann der Schlüssel aus Sicherheitsgründen nicht aus der Gerätekonfiguration herausgelesen werden. Passend zur Verschlüsselung einer Transponderkarte wird in der Gerätekonfiguration eine VHL-Datei abgespeichert. Es können mehrere solcher VHL-Dateien abgespeichert werden, um auf verschiedene Karten Zugriff ohne Umkonfiguration zu erhalten.

Es gibt drei Möglichkeiten, eine solche VHL-Datei in die Konfiguration des RFID-Gerätes zu übertragen:

Konfigurationsart	Beschreibung
Konfigurationskarte	Eine Konfiguration kann über eine Konfigurationskarte übertragen werden. Dies ist die bevorzugte Variante.
Tool „Baltech id-engine explorer“	Mit dem Tool ist die Übertragung einer Konfiguration in den Speicher des Baltech RFID-Gerätes möglich (siehe Baltech Dokumentation: IdEngineExplorer.pdf). Die spezifische Konfiguration kann in einfachen Fällen direkt in dem Tool erstellt werden. Alternativ bietet die Firma Baltech Unterstützung bei der Erstellung an und kann eine Datei mit der Konfiguration zusenden. Das Tool „Baltech id-engine explorer“ ist auf allen Windows Betriebssystemen ab Windows XP lauffähig. Es ist nicht für Windows CE verfügbar.
aus der SPS	Für Mifare-Classic-Karten kann die Übertragung einer VHL-Datei-Konfiguration im SPS-Programmcode programmiert werden. Der Befehl SetConfig [▶ 23] überträgt die Konfiguration, die am Eingang in ST_RFID_ConfigIn [▶ 50] angegeben ist. Die Struktur einer Mifare-Karte und die möglichen Einstellungen zur Schlüsselvergabe sind auf der Detailseite von ST_RFID_CfgStruct_BaltechMifVHLFile [▶ 53] erläutert. (Ausführliche Informationen zu Mifare-Karten finden Sie in dem Baltech Dokument Mifare.pdf im Baltech SDK)

Transponder

Passende Transponderkarten für Baltech RFID-Geräte können von verschiedenen Herstellern bezogen werden. Sollen die Karten mit Verschlüsselung genutzt werden, so bietet die Firma Baltech den Vertrieb von bereits vorkonfigurierten Karten an.

Herstellerkontakt

<http://www.baltech.de>

5.1.3 Deister electronic

RFID-Reader-Einstellungen

Für eine reibungslose Kommunikation zwischen Steuerung und RFID Reader müssen manche Einstellungen vor Systemstart vorgenommen werden. Hierzu zählt beispielsweise die Baudrate der seriellen Kommunikation. Um diese Einstellungen auf den RFID Reader zu übertragen, kann ein proprietäres Tool des RFID-Reader-Herstellers nötig sein.

Für alle unterstützten RFID-Reader-Modelle haben sich die folgenden Standardeinstellungen der Datenübertragung bewährt:

Einstellung	Wert
Baudrate (RS232 und RS485)	9600 Baud
Parity Bit	none
Datenbits	8
Stopbit	1

Bei Bedarf können je nach Hardware auch andere Parameter eingestellt oder die Werkseinstellungen des RFID Readers verwendet werden. Diese müssen dann auch in der softwareseitigen Reader-Anbindung übernommen werden (siehe [RFID-Reader-Anbindung \[▶ 19\]](#)).

Mittels der proprietären Tools müssen gegebenenfalls vor Systemstart folgende spezielle Einstellungen parametrisiert werden:

Einstellung	Wert
Parameter der Datenübertragung (s.o.)	Einstellung in Analogie zu den im SPS-Programm gewählten Werten

RFID-Reader-Handhabung

Hier sei erneut auf die Funktionsweise „Polling“ hingewiesen, die einen mehrfachen Aufruf des Befehls sinnvoll macht, wenn auf aktuelle Transponderinformationen Wert gelegt wird (siehe [Befehlsbeschreibung \[► 21\]](#)).

Hinzu kommt die Eigenart, dass bei den proxEntry-Modellen ein Polling-Befehl anliegen muss, um die Verbindung zum Transponder aufzubauen. Bei den UDL-Modellen sieht die Konfiguration wiederum einen automatischen Verbindungsaufbau zu detektierten Transpondern vor, sodass kein Polling-Befehl zwangsweise nötig ist.

In der RFID-Reader-Konfiguration muss die dem Tag entsprechende Blockgröße konfiguriert sein.

Die Deister RDL-Geräte unterstützen 4 Bytes oder 8 Bytes Blockgröße.

Beispiel: Wenn für den Transponder eine Blockgröße von 8 Byte angegeben ist, muss der Reader mit dem Parameter `iBlocksize := 8` konfiguriert sein und der Lese- bzw. Schreibzugriff über die Struktur `ST_RFID_AccessData [► 52]` muss mit 8 Byte Blockgröße geschehen.

Deister RDL: Mit einem Schreibbefehl können maximal 36 Bytes Daten am Stück geschrieben werden. Sollen mehr Daten auf den Transponder geschrieben werden, müssen diese auf mehrere Befehle aufgeteilt werden.

Hier sei darauf hingewiesen, dass nicht alle Eigenarten jedes unterstützten RFID-Reader-Modells hier genannt werden können. Deshalb wird für detaillierte Informationen auf die herstellereigenen Dokumentationen hingewiesen.

Konfiguration

Wird eine neue Konfiguration auf das RFID-Gerät geschrieben (Befehl `SetConfig [► 23]`) muss Folgendes beachtet werden:

Deister RDL-Geräte: Nicht jede Kombination von Konfigurationsparametern ist zulässig (siehe `ST_RFID_CfgStruct_DeisterRDL [► 56]`). Eine Missachtung der erforderlichen Abhängigkeiten führt zu einem Fehler (`eRFERR_InvalidCfg`):

Konfigurationsparameter	erforderliche Abhängigkeiten
<code>eReadMode = eRFRD_ContinuousRead</code>	<code>eTriggerMode = eRFTR_ImmediateRead</code> <code>eWriteMode = eRFWR_ImmediateWrite</code>
<code>eWriteMode = eRFWR_WriteToNextTag</code>	<code>eTriggerMode = eRFTR_ReadWithTrigger</code>
<code>bMultiTranspMode = TRUE</code>	<code>bSerialNumberMode = TRUE</code> <code>eWriteMode = eRFWR_ImmediateWrite</code> <code>eReadMode = eRFRD_SingleShot</code>

Wird die Konfiguration als Register übertragen, bestehen diese Abhängigkeiten ebenso und das RFID-Gerät wird bei Unzulässigkeit einen Fehlercode zurückliefern.

5.1.4 Leuze electronic

RFID-Reader-Einstellungen

Für eine reibungslose Kommunikation zwischen Steuerung und RFID Reader müssen manche Einstellungen vor Systemstart vorgenommen werden. Hierzu zählt beispielsweise die Baudrate der seriellen Kommunikation. Um diese Einstellungen auf den RFID Reader zu übertragen, kann ein proprietäres Tool des RFID-Reader-Herstellers nötig sein.

Für alle unterstützten RFID-Reader-Modelle haben sich diese Standardeinstellungen der Datenübertragung bewährt:

Einstellung	Wert
Baudrate (RS232 und RS485)	9600 Baud
Parity Bit	none
Datenbits	8
Stopbit	1

Bei Bedarf können je nach Hardware auch andere Parameter eingestellt oder die Werkseinstellungen des RFID Readers verwendet werden. Diese müssen dann auch in der softwareseitigen Reader-Anbindung übernommen werden (siehe [RFID-Reader-Anbindung](#) [▶ 19]).

Mittels der proprietären Tools müssen gegebenenfalls vor Systemstart folgende spezielle Einstellungen parametrisiert werden.

Einstellung	Wert
Parameter der Datenübertragung (s. o.)	Einstellung in Analogie zu den im SPS-Programm gewählten Werten

Sollte der RFID Reader mittels eines Triggers angesteuert werden, so wird das darauffolgende Antworttelegramm vom Funktionsbaustein der TwinCAT-RFID-Bibliothek empfangen. Eine Zuweisung von gelesenen Transponderdaten ist in dem Fall nicht gegeben. Die empfangenen Rohdaten können zur weiteren Verarbeitung dem Bausteininterface entnommen werden.

RFID-Reader-Handhabung

In der RFID-Reader-Konfiguration muss die dem Tag entsprechende Blockgröße konfiguriert sein.

Die Leuze-Geräte unterstützen 4 Bytes oder 8 Bytes Blockgröße.

Beispiel: Falls für den Transponder eine Blockgröße von 8 Byte angegeben ist, muss der Reader mit dem Parameter `iBlocksize := 8` konfiguriert sein und der Lese- bzw. Schreibzugriff über die Struktur [ST_RFID_AccessData](#) [▶ 52] muss mit 8 Byte Blockgröße geschehen.

Mit einem Schreibbefehl können maximal 36 Bytes Daten am Stück geschrieben werden. Sollen mehr Daten auf den Transponder geschrieben werden, müssen diese auf mehrere Befehle aufgeteilt werden.

Hier sei darauf hingewiesen, dass nicht alle Eigenarten jedes unterstützten RFID-Reader-Modells hier genannt werden können. Deshalb wird für detaillierte Informationen auf die herstellereigenen Dokumentationen hingewiesen.

Konfiguration

Wird eine neue Konfiguration auf das RFID-Gerät geschrieben (Befehl [SetConfig](#) [▶ 23]) muss Folgendes beachtet werden:

Nicht jede Kombination von Konfigurationsparametern (Struktur [ST_RFID_CfgStruct_LeuzeRFM](#) [▶ 61]) ist zulässig. Eine Missachtung der erforderlichen Abhängigkeiten führt zu einem Fehler (`eRFERR_InvalidCfg`):

Konfigurationsparameter	erforderliche Abhängigkeiten
<code>eReadMode = eRFRD_ContinuousRead</code>	<code>eTriggerMode = eRFTR_ImmediateRead</code> <code>eWriteMode = eRFWR_ImmediateWrite</code>
<code>eWriteMode = eRFWR_WriteToNextTag</code>	<code>eTriggerMode = eRFTR_ReadWithTrigger</code>
<code>bMultiTranspMode = TRUE</code>	<code>bSerialNumberMode = TRUE</code> <code>eWriteMode = eRFWR_ImmediateWrite</code> <code>eReadMode = eRFRD_SingleShot</code>

Wird die Konfiguration als Register übertragen, bestehen diese Abhängigkeiten ebenso und das RFID-Gerät wird bei Unzulässigkeit einen Fehlercode zurückliefern.

5.1.5 Pepperl+Fuchs

RFID-Reader-Einstellungen

Für eine reibungslose Kommunikation zwischen Steuerung und RFID Reader müssen manche Einstellungen vor Systemstart vorgenommen werden. Hierzu zählt beispielsweise die Baudrate der seriellen Kommunikation. Um diese Einstellungen auf den RFID Reader zu übertragen, kann ein proprietäres Tool des RFID-Reader-Herstellers nötig sein.

Für alle unterstützten RFID-Reader-Modelle haben sich folgende Standardeinstellungen der Datenübertragung bewährt:

Einstellung	Wert
Baudrate (RS232 und RS485)	38400 Baud
Parity Bit	none
Datenbits	8
Stopbit	1

Bei Bedarf können je nach Hardware auch andere Parameter eingestellt werden. Diese müssen dann auch in der softwareseitigen Reader-Anbindung übernommen werden (siehe [RFID-Reader-Anbindung](#) [► 19]).

RFID-Reader-Handhabung

Bei Systemstart müssen die Modellinformationen (Befehl [GetReaderVersion](#) [► 22]) und die aktuelle Reader-Konfiguration (Befehl [GetConfig](#) [► 22]) ausgewertet werden.

Der empfangene Status des Gerätes wird über den Ausgang `iErrCodeRcv` des Funktionsbausteins `FB_RFIDReader` angezeigt und im Fehlerfall durch `bError = TRUE` und `iErrorId = eRFERR_ErrorRcv` signalisiert. Die Leseköpfe besitzen ebenfalls eigene Status. Diese können mit der über [GetConfig](#) gelesenen [Konfigurationsstruktur](#) [► 63] geprüft werden.

Bei einem Neustart sollten die eingestellten Transpondertypen überprüft werden. Wenn die über [GetConfig](#) gelesene Konfigurationsstruktur nicht die richtigen Transpondertypen für jeden Lesekopf anzeigt, so können diese mit dem Befehl [ChangeDCType](#) [► 25] korrigiert werden. Es wird empfohlen, anstatt dem Default-Wert (99) den für den Transponder spezifizierten Wert einzustellen. Dadurch erkennt der Schreib-/Lesekopf zudem den Datenträger schneller.

Beim Lese- sowie Schreibzugriff auf den Datenspeicher eines Transponders muss für alle Pepperl+Fuchs RFID-Geräte eine Blockgröße passend zum Transponder verwendet werden (siehe [ST_RFID_AccessData](#) [► 52]).

Blockgrößen möglicher Transponder:

4 Byte (IQC21, IPC03, IQC22, IQC24)
 8 Byte (IQC33)
 16 Byte (IQC40, IQC41, IQC42 und IQC43)
 32 Byte (IQC37)

Bis Version 3.3.3.0 der Bibliothek wurde nur die Verwendung von 4 Byte Blockgröße unterstützt.

Hier sei darauf hingewiesen, dass nicht alle Eigenarten jedes unterstützten RFID-Reader-Modells hier genannt werden können. Deshalb wird für detaillierte Informationen auf die herstellereigenen Dokumentationen hingewiesen.

Buffered Command - Gepufferter Befehl

Mit der Eingangsvariablen `bBufferedCmd` in [ST_RFID_Control](#) [► 45] können Befehle abgesetzt werden, die für eine spätere dauerhafte Ausführung gepuffert werden. Dies ist mit den Befehlen `eRFC_GetInventory`, `eRFC_ReadBlock` und `eRFC_WriteBlock` möglich. Ein gepufferter Befehl kann mit dem Befehl `eRFC_AbortCommand` beendet werden.

i Gepufferter Befehl

Ist an einem Lesekopf ein gepufferter Befehl aktiv, darf der Trigger Mode nicht für diesen Kanal aktiviert werden bzw. aktiv sein. Ebenso darf kein Rohdatenbefehl abgesetzt werden, welcher diesen Kanal betrifft.

Trigger Mode

Es wird empfohlen, keinen Trigger bzw. Sensor kanal zu verwenden. Der Trigger Mode sollte also für alle Kanäle deaktiviert sein. Per Werkseinstellung des RFID-Gerätes ist der Trigger auf allen Kanälen deaktiviert.

Alternativ kann beispielsweise der Befehl `GetInventory` zyklisch oder `GetInventory` als gepufferter Befehl (`bBufferedCmd` in [ST RFID Control \[▶ 45\]](#)) aufgerufen werden.

Ist ein Trigger als Sensor kanal an der RFID-Einheit nötig, so gibt es mit der TwinCAT-Bibliothek folgende Möglichkeit:

Der Trigger liefert eine Meldung, ob er gerade ausgelöst oder ob der Triggerbereich verlassen wird. Diese Messages werden empfangen und als `eResponse = eRFR_CmdConfirmation` oder `eRFR_NoTransponder` angezeigt. In der Applikation kann darauf reagiert werden und der gewünschte Befehl ausgelöst werden.

Um einen Kanal dementsprechend als Sensor kanal/Trigger zu konfigurieren, muss der zugehörige Identkanal = 0 sein. Der notwendige Rohdatenbefehl ist im nächsten Absatz erläutert.

i Triggereinstellung

Die Triggereinstellung darf nicht vom herstellereigenen Tool aus vorgenommen werden. Andernfalls sind die daraufhin eintreffenden Meldungen des Sensor kanals nicht von dem Baustein der TwinCAT-RFID-Bibliothek lesbar.

Die korrekte Einstellung des Trigger Mode sollte mit dem Befehl [GetConfig \[▶ 22\]](#) und der Auswertung der gelesenen Konfigurationsstruktur überprüft werden. So kann die Einstellung falls nötig bei Programmstart nachgeholt werden.

Einstellungen per Rohdatenbefehle absetzen

Details können Sie dem Abschnitt [Low-Level-Kommunikation \[▶ 41\]](#) sowie der Beschreibung der Strukturen [ST RFID Control \[▶ 45\]](#) und [ST RFID RawData \[▶ 45\]](#) entnehmen.

Baudrate:

Um die Baudrate des RFID-Gerätes auf 9600 Baud einzustellen, müssen folgende Rohdaten gesendet werden:

ASCII	hex
CI0,9600	43 49 30 2C 39 36 30 30

Nach dem Ändern der Baudrate ist ein Reset des RFID-Gerätes nötig.

Triggermode:

Um an Kanal 3 einen Triggersensor zu deaktivieren, sodass der Kanal als Lesekopf genutzt werden kann, müssen folgende Rohdaten gesendet werden:

ASCII	hex
TM300	54 4D 33 30 30

Um an Kanal 2 einen Sensor als Trigger zu konfigurieren, müssen folgende Rohdaten gesendet werden:

ASCII	hex
TM201	54 4D 32 30 31

Antwort: `eResponse = eRFR_CmdConfirmation`, sobald der Sensor ausgelöst wird.

Antwort: `eResponse = eRFR_NoTransponder`, sobald der Sensor verlassen wird.

Am Ausgang stTransplInfo.iHeadNumber ist der Sensorkanal angegeben, von dem die Antwort gesendet wurde.

Um an Kanal 4 einen Sensor als invertierten Trigger zu konfigurieren, müssen folgende Rohdaten gesendet werden:

ASCII	hex
TM402	54 4D 34 30 32

Antwort: eResponse = eRFR_NoTransponder, sobald der Sensor ausgelöst wird.

Antwort: eResponse = eRFR_CmdConfirmation, sobald der Sensor verlassen wird.

Am Ausgang stTransplInfo.iHeadNumber ist der Sensorkanal angegeben von dem die Antwort gesendet wurde.

Nachdem eine solche Einstellung vorgenommen wurde, muss die Gerätekonfiguration mit dem Befehl [Get Config](#) ([▶ 22](#)) erneut gelesen werden. Es empfiehlt sich, die vorgenommenen Einstellungen mittels Auswertung der gelesenen Konfigurationsstruktur zu überprüfen.

6 SPS API

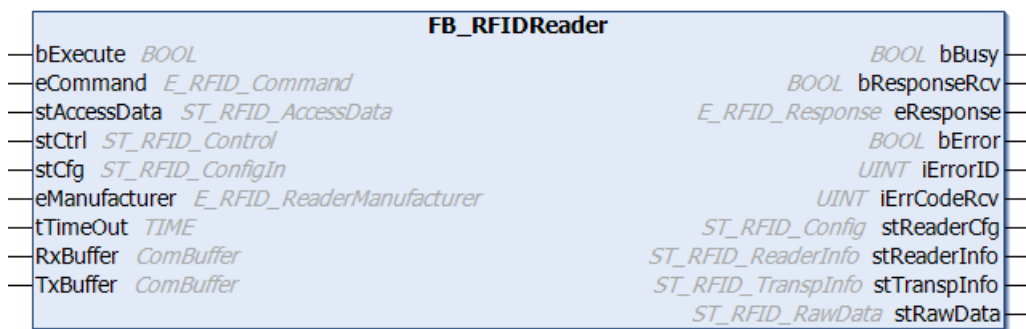
6.1 Funktionsbaustein

6.1.1 FB_RFIDReader

Die TwinCAT-RFID-Bibliothek besteht lediglich aus einem Funktionsbaustein.

In diesem Abschnitt werden für einen schnellen Einstieg in die Handhabung der Bibliothek die Schnittstellenvariablen des Funktionsbausteins erläutert. Beachten Sie auch das [Tutorial \[▶ 69\]](#) und die [Beispiele \[▶ 69\]](#).

Die einheitliche Handhabung für alle RFID-Reader-Modelle und die damit verbundenen aufbereiteten Schnittstellendeklarationen sind besonders anwenderfreundlich. Allerdings sei darauf hingewiesen, dass der Funktionsbaustein der TwinCAT-RFID-Bibliothek aufgrund der Unterschiede einiger RFID-Reader-Modelle einen geringfügigen Overheat besitzt. Diese unabdingbare Eigenschaft wird jedoch stark durch die Vorteile überwogen, welche die verfügbare Flexibilität bietet.



Syntax

```

FUNCTION_BLOCK FB_RFIDREADER
VAR_INPUT
    bExecute      : BOOL;
    eCommand      : E_RFID_Command;
    stAccessData  : ST_RFID_AccessData;
    stCtrl        : ST_RFID_Control;
    stCfg         : ST_RFID_ConfigIn;
    eManufacturer : E_RFID_ReaderManufacturer;
    tTimeout      : TIME := T#5s;
END_VAR
VAR_IN_OUT
    RxBuffer      : ComBuffer;
    TxBuffer      : ComBuffer;
END_VAR
VAR_OUTPUT
    bBusy         : BOOL;
    bResponseRcv  : BOOL;
    eResponse     : E_RFID_Response;

    bError        : BOOL;
    iErrorID      : UINT;      (* general RFID error *)
    iErrCodeRcv   : UINT;      (* error received by reader *)

    stReaderCfg   : ST_RFID_Config;
    stReaderInfo  : ST_RFID_ReaderInfo;
    stTranspInfo  : ST_RFID_TranspInfo;
    stRawData     : ST_RFID_RawData;
END_VAR
    
```

Eingänge

Name	Typ	Beschreibung
bExecute	BOOL	Um Nachrichten des RFID Readers zu empfangen, wird der Funktionsbaustein mit FALSE an diesem Eingang aufgerufen. Der Funktionsbaustein reagiert auf eine positive Flanke von bExecute, indem der ausgewählte Befehl eCommand ausgeführt bzw. beim RFID Reader angefragt wird.
eCommand	E_RFID_Command [▶ 65]	Der Eingang eCommand bietet eine Auswahl an Befehlen, wie beispielsweise das Lesen oder Schreiben eines Transponders, in Form einer Enumeration an. Ein Befehl wird ausgeführt, indem der Eingang bExecute gesetzt wird.
stAccessData	ST_RFID_AccessData [▶ 52]	Wenn ein Schreib- oder Lesebefehl ausgeführt werden soll, müssen mit dieser Eingangsstruktur Parameter übergeben werden.
stCtrl	ST_RFID_Control [▶ 45]	Mit stCtrl können unterschiedliche Kontrollparameter am Eingang übergeben werden. Dazu gehört unter anderem auch die Möglichkeit, Verzögerungszeiten anzugeben.
stCfg	ST_RFID_ConfigIn [▶ 50]	Ein RFID Reader besitzt eine interne Konfiguration. Diese lässt sich bei einigen Geräten auslesen und ändern. An dem Eingang stCfg werden Konfigurationsparameter übergeben, die auf den RFID Reader übertragen werden sollen. Siehe auch: Konfiguration [▶ 40]
eManufacturer	E_RFID_ReaderManufacturer [▶ 67]	An diesem Eingang wird der Hersteller des verwendeten RFID-Reader-Modells angegeben.
tTimeOut	TIME	Gibt eine maximale Zeitdauer für die Ausführung des Funktionsbausteins an. Der Default-Wert ist 5 Sekunden.

Es gilt die Bedingung $tTimeOut > tPreSendDelay + tPostSendDelay$. Andernfalls wird ein Fehler am Ausgang ausgegeben. Siehe Details zu den Verzögerungszeiten in [ST_RFID_Control](#) [▶ 45].

/ Ein-/Ausgänge

Name	Typ	Beschreibung
RxBuffer	ComBuffer	Es wird der Empfangspuffer angegeben, welcher als Eingangsvariable deklariert und im TwinCAT System Manager mit der seriellen Klemme verlinkt wurde. Siehe dazu die Beschreibung der seriellen Anbindung eines RFID Readers im Abschnitt RFID-Reader-Anbindung [▶ 19]
TxBuffer	ComBuffer	Es wird der Ausgangspuffer angegeben, welcher als Ausgangsvariable deklariert und im TwinCAT System Manager mit der seriellen Klemme verlinkt wurde. Siehe dazu die Beschreibung der seriellen Anbindung eines RFID Readers im Abschnitt RFID-Reader-Anbindung [▶ 19]

 **Ausgänge**

Name	Typ	Beschreibung
bBusy	BOOL	<p>Der Ausgang bBusy wird bei einem gültigen Kommandoaufruf für mindestens einen Takt TRUE. Der Funktionsbaustein darf erst wieder mit bExecute = TRUE für einen erneuten Befehl aufgerufen werden, wenn bBusy zu FALSE gewechselt ist und der Funktionsbaustein somit nicht mehr im aktiven Sendezustand ist. Somit können, wenn bBusy = FALSE erkannt wird, wiederum alle weiteren Ausgangsvariablen bResponseRcv, eResponse, bError, iErrCodeRcv, ... und stRawData ausgewertet werden.</p> <p>Wenn zu einem getätigten Kommandoaufruf eine Response erwartet wird, so bleibt der Funktionsbaustein so lange bBusy = TRUE, bis ein Telegramm empfangen wird oder das Timeout tTimeOut erreicht wird.</p> <p>Wenn dem Funktionsbaustein Verzögerungszeiten tPreSendDelay und/oder tPostSendDelay mitgegeben wurden, ist bBusy mindestens so lange TRUE wie die Summe dieser Zeiten.</p> <p>Siehe auch: ST_RFID_Control [▶ 45].</p>
bResponseRcv	BOOL	<p>Sobald eine Response vom RFID Reader bei der Steuerung eingetroffen ist, wird dieses Flag für mindestens einen Zyklus gesetzt.</p> <p>Mit steigender Flanke zu bResponseRcv = TRUE wird allgemein das Eintreffen eines Telegramms signalisiert. Somit können, wenn dieses Flag erkannt wird, wiederum auch unerwartete Telegramme und die dazugehörigen Ausgangsvariablen eResponse, bError, iErrCodeRcv, ... und stRawData ausgewertet werden.</p> <p>Wenn zu einem getätigten Kommandoaufruf eine Response erwartet wird, so bleibt der Funktionsbaustein so lange bBusy = TRUE, bis ein Telegramm empfangen wird. Je nach Kommandoaufruf kann mehr als eine Response eintreffen, bevor die Aktion abgeschlossen und bBusy = FALSE ist.</p> <p>Je nach Konfigurationseinstellung der Verzögerungszeiten in ST_RFID_Control [▶ 45] kann bResponseRcv bereits TRUE werden, bevor bBusy wieder auf FALSE wechselt.</p>
eResponse	E_RFID_Response [▶ 65]	<p>Sobald bResponseRcv TRUE anzeigt, gibt diese Enumeration die Art der empfangenen Nachricht an. Je nach Art kann beispielsweise die entsprechende Auswertung folgen.</p>
bError	BOOL	<p>Der Ausgang bError wird TRUE, sobald ein Fehler auftritt. Dabei kann es sich um fehlerhafte Eingangsparameter, um Übertragungsfehler, um Fehler seitens des RFID Readers oder um ein Timeout handeln.</p> <p>Welche Art von Fehler aufgetreten ist, wird mit der Ausgangsvariablen iErrorID angezeigt. Details zur Fehlerdarstellung sind im Abschnitt Fehlercodes [▶ 77] angegeben.</p>
iErrorID	UNIT	<p>Wenn ein Fehler auftritt, wird am Ausgang iErrorID die Art des Fehlers angezeigt. Details zu den möglichen Fehler-IDs sind im Abschnitt Fehlercodes [▶ 77] angegeben.</p>
iErrCodeRcv	UINT	<p>Der am Ausgang iErrCodeRcv angegebene Fehlercode entspricht dem vom RFID Reader an die Steuerung gesendetem Fehlercode. Details zur Fehlerdarstellung sind im Abschnitt Fehlercodes [▶ 77] angegeben.</p>

Name	Typ	Beschreibung
stReaderCfg	ST_RFID_Config [▶ 52]	Ein RFID Reader besitzt eine interne Konfiguration. Diese lässt sich bei einigen Geräten auslesen und ändern. Am Ausgang stReaderCfg werden diese ausgelesenen Konfigurationsparameter zur Verfügung gestellt.
stReaderInfo	ST_RFID_ReaderInfo [▶ 43]	Jeder RFID Reader besitzt eigene Kenndaten wie Bezeichnung, Hardwareversion etc. Diese Werte, die unter anderem über den Befehl GetReaderVersion [▶ 22] abgefragt werden können, sind in der Ausgangsstruktur stReaderInfo angegeben.
stTransplInfo	ST_RFID_TransplInfo [▶ 43]	Die Struktur stTransplInfo enthält Informationen zu dem zuletzt gelesenen Transponder. Hier wird unter anderem die Seriennummer des Transponders ausgegeben.
stRawData	ST_RFID_RawData [▶ 45]	Die Ausgangsstruktur stRawData gibt die gesendeten sowie die empfangenen Rohdaten aus.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.1.2 Handhabung

RFID-Bibliothek-Handhabung

Wenn Sie die Bibliotheksdatei Tc2_RFID eingebunden haben, erhalten Sie Zugriff auf alle Funktionen. Die Bibliothek stellt zur Kommunikation mit einem RFID Reader einen Funktionsbaustein zur Verfügung.

Es kann der generelle Funktionsbaustein FB_RFIDReader genutzt werden, der für alle RFID-Reader-Modelle verwendbar ist oder einer der herstellerspezifischen Funktionsbausteine. Diese bieten den identischen Funktionsumfang, annähernd das gleiche Interface, das gleiche Handling und sind zudem code- und performanceoptimiert.

Der zur RFID-Reader-Kommunikation zur Verfügung gestellte Funktionsbaustein bietet High-Level-Kommunikation mit High-Level Interface. Ein Befehlssatz stellt unterschiedlichste Kommandos zur Verfügung (siehe [RFID-Befehlssatz](#) [[▶ 21](#)]). Zusätzlich ermöglicht die integrierte Low-Level-Kommunikation das Senden und Empfangen von Rohdaten (siehe [Low-Level-Kommunikation](#) [[▶ 41](#)]).

Die Tc2_RFID-Bibliothek stellt an die RFID Reader die Erwartung, dass eine auf einen Befehl folgende Response unmittelbar nach dem Befehl erfolgt und der Dialog nicht durch ein anderes Telegramm unterbrochen wird. Andernfalls ist eine Auswertung ggf. nicht möglich.

Allgemeine Handhabung des Funktionsbausteins

Je nach RFID-Reader-Modell kann das Gerät ohne vorherige Aufforderung ein Telegramm zur Steuerung senden. Zum Empfang reicht ein zyklischer Aufruf des RFID-Funktionsbausteins mit bExecute = FALSE.

Alle möglichen aktiven Zugriffe auf das RFID-Gerät sind im Befehlssatz gelistet (siehe [RFID-Befehlssatz](#) [[▶ 21](#)]). Allen Befehlen ist folgende Vorgehensweise gleich:

Der Funktionsbaustein wird mit einer positiven Flanke am Eingang bExecute aufgerufen. Danach liefert zyklisches Aufrufen des Funktionsbausteins (bExecute = FALSE) das Ergebnis der Abfrage am Ausgang, sobald die Bearbeitung der Abfrage abgeschlossen ist (bBusy = FALSE). Der Funktionsbaustein muss so lange aufgerufen werden (bExecute = FALSE), bis die interne Bearbeitung abgeschlossen ist (bBusy = FALSE). Währenddessen sind alle Eingänge des Funktionsbausteins unverändert zu belassen.

Alle empfangenen Nachrichten werden zusätzlich als Rohdaten in nicht aufbereiteter Form komplett am Ausgang zur Verfügung gestellt.

Weitere Handhabungshinweise finden Sie in der Beschreibung der Ein- und Ausgangsvariablen des Funktionsbausteins sowie in dem Tutorial und den Beispielen dieser Dokumentation.

Initialisierung eines über die TwinCAT-Bibliothek integrierten RFID Readers

Bei Systemstart sind folgende Aktionen zur Initialisierung eines über die TwinCAT-Bibliothek integrierten RFID Readers notwendig:

Soweit laut Befehlssatz verfügbar müssen die Modellinformationen (Befehl [GetReaderVersion](#) [[▶ 22](#)]) und die aktuelle Reader-Konfiguration (Befehl [GetConfig](#) [[▶ 22](#)]) ausgewertet werden. Weil eine erfolgreiche Kommunikation mit dem RFID Reader von diesen Daten abhängig ist, muss sichergestellt werden, dass immer die aktuellen Werte vorliegen und bei Bedarf abgefragt werden.

RFID-Reader-Handhabung

Im Abschnitt [RFID-Reader-Einstellungen und -Handhabung](#) [[▶ 26](#)] werden Eigenarten der unterstützten RFID-Reader-Modelle genannt. Die dort aufgeführten Hinweise sind den speziellen RFID-Reader-Herstellern zugeordnet.

6.1.3 Konfiguration

Alle unterstützten RFID Reader lassen sich mit demselben Befehl konfigurieren. Dieser muss gemäß dem Befehlssatz für das spezielle Modell verfügbar sein (siehe [RFID-Befehlssatz](#) [[▶ 21](#)]).

Zu jedem Programmstart sollte neben der Reader-Version auch die aktuelle Konfiguration des Readers angefordert werden.

Weil die RFID Reader unterschiedlicher Hersteller nie identische Konfigurationsmöglichkeiten besitzen, bietet die TwinCAT-RFID-Bibliothek neben der Eingangs-Konfigurationsstruktur jeweils eine Unterstruktur pro Hersteller mit den spezifischen Parametern (ST_RFID_CfgStruct_DeisterUDL, ST_RFID_CfgStruct_LeuzeRFM, ...). Die dort gelisteten Parameter sind vom Nutzer im Rahmen der gültigen Wertebereiche beliebig zu parametrieren. Die Bedeutung der Parameter ist entweder der Strukturdeklaration oder den proprietären Spezifikationen zu entnehmen.

Konfiguration lesen

Um die aktuelle RFID-Reader-Konfiguration auszulesen, wird der Befehl [GetConfig](#) [[▶ 22](#)] aus dem Befehlssatz verwendet. Daraufhin können bei erfolgreicher Abfrage die Konfigurationsdaten am Ausgang des Funktionsbausteins entnommen werden. Sie liegen dort in der Struktur [ST_RFID_Config](#) [[▶ 52](#)] als Konfigurationsstruktur sowie auch als Konfigurationsregister vor.

Konfiguration ändern

Um eine RFID-Reader-Konfiguration zu schreiben, wird der Befehl [SetConfig](#) [[▶ 23](#)] aus dem Befehlssatz verwendet. Nach einem Befehl [SetConfig](#) muss die aktuelle Konfiguration einmal mit dem Befehl [GetConfig](#) [[▶ 22](#)] ausgelesen werden.

Wenn der Nutzer weitergehende spezielle Konfigurationsparameter über ein externes Tool einstellt und diese beibehalten will, sollte das Flag für „Default Values“ `bUseCfgDefault` in der Struktur [ST_RFID_ConfigIn](#) [[▶ 50](#)] deaktiviert werden.

i Unzulässige Kombination von Konfigurationsparametern

Teilweise sind bestimmte Kombinationen von Konfigurationsparametern unzulässig. Welche Parameterwerte sich bei welcher Kombination ausschließen, können Sie den proprietären Protokollspezifikationen der RFID-Reader-Hersteller entnehmen.

Bei fehlerhafter Eingabe der Parameter wird entweder bereits vor Konfigurationsanfrage ein Fehler generiert oder der RFID Reader signalisiert durch seine Response, dass die Konfigurationsdaten nicht übernommen werden konnten.

Konfigurationsdaten

Jede Konfiguration kann als Register (Byte Array) oder als Struktur gesehen werden. Dabei handelt es sich nicht um die Parametrierung der TwinCAT-RFID-Bibliothek sondern um die proprietäre Konfiguration des RFID Readers. So gibt es in der TwinCAT-RFID-Bibliothek verschiedene Konfigurationsstrukturen, welche die Rohdaten der Konfigurationsregister unterschiedlicher RFID Reader aufarbeiten. Am Ausgang des Funktionsbausteins der Bibliothek werden, wenn verfügbar, beide Varianten in [ST_RFID_Config \[▶ 52\]](#) zur Verfügung gestellt. Dies geschieht über Pointer.

Baltech

Die Konfigurationsdaten werden für Baltech RFID Reader als Struktur verwendet.

- [ST_RFID_CfgStruct_BaltechMifVHLFile \[▶ 53\]](#)

Die Struktur ist für das Schreiben mit dem Befehl `eRFC_SetConfig` geeignet (siehe [RFID-Befehlssatz \[▶ 21\]](#)).

Balluff

Es wird keine Möglichkeit der Konfiguration angeboten.

Deister

Die Konfigurationsdaten können für Deister RFID Reader sowohl als Struktur als auch als Register verwendet werden.

Wenn ein Register (Byte Array) verwendet wird, muss dieses immer die Größe der vollständigen Konfigurationsdaten besitzen. Bei den unterstützten Deister RDL-Geräten ist dies 88 Byte und bei den UDL-Geräten 117 Byte.

- [ST_RFID_CfgStruct_DeisterRDL \[▶ 56\]](#)
- [ST_RFID_CfgStruct_DeisterUDL \[▶ 58\]](#)

Die Strukturen sind für das Schreiben mit `eRFC_SetConfig` sowie das Lesen mit `eRFC_GetConfig` geeignet (siehe [RFID-Befehlssatz \[▶ 21\]](#)).

Leuze

Die Konfigurationsdaten können für Leuze RFID Reader sowohl als Struktur als auch als Register verwendet werden.

Wenn ein Register (Byte Array) verwendet wird, muss dieses immer die Größe der vollständigen Konfigurationsdaten besitzen. Bei den unterstützten Leuze-Geräten ist dies 88 Byte.

- [ST_RFID_CfgStruct_LeuzeRFM \[▶ 61\]](#)

Die Struktur ist für das Schreiben mit `eRFC_SetConfig` sowie das Lesen mit `eRFC_GetConfig` geeignet (siehe [RFID-Befehlssatz \[▶ 21\]](#)).

Pepperl+Fuchs

Die Konfigurationsdaten werden für Pepperl+Fuchs RFID Reader als Struktur verwendet.

- [ST_RFID_CfgStruct_PepperlFuchsIDENT \[▶ 63\]](#)

Die Struktur ist für das Lesen mit `eRFC_GetConfig` geeignet (siehe [RFID-Befehlssatz \[▶ 21\]](#)).

6.1.4 Low-Level-Kommunikation

Die TwinCAT-RFID-Bibliothek bietet neben dem High-Level-Befehlssatz auch die Möglichkeit der Low-Level-Kommunikation. Dies ist implizit gelöst. Es wird derselbe Funktionsbaustein verwendet. Es können beliebige Telegramme bis zu einer maximalen Größe von 1024 Bytes empfangen und bis zu einer Größe von 300 Bytes versendet werden.

Ein gesamtes Telegramm setzt sich dabei wie folgt zusammen:

| Prefix | Adressierung | **Rohdaten** | CRC | Suffix |

Je nach proprietärer Protokollspezifikation können einzelne Bestandteile fehlen. Generell ist die Zusammensetzung aber gleich.

Senden

Zum Senden wird der Befehl `eRFC_SendRawData` aus dem Befehlssatz verwendet (siehe [RFID-Befehlssatz](#) [► 21]). Die zu sendenden Rohdaten werden in der Eingangsstruktur [ST_RFID_Control](#) [► 45] angegeben.

Um ein Low-Level-Telegramm abzuschicken, werden nur die Rohdaten angegeben. Die anderen Bestandteile des Telegramms werden automatisch von der TwinCAT-RFID-Bibliothek ergänzt. Ebenso werden Prüfdaten wie CRC intern erzeugt und eingefügt.

Wenn das Protokoll eine Umkodierung von bestimmten Bytes innerhalb der Rohdaten verlangt, wird dies ebenfalls automatisch von der TwinCAT-RFID-Bibliothek vorgenommen.

Wenn es sich um eine RS485-Schnittstelle handelt, so muss die Adressierung gesondert angegeben werden. Sie darf nicht mit in den angegebenen Rohdaten enthalten sein. Per Default wird die Adressierung automatisch von der Bibliothek übernommen. Sie kann allerdings über Eingangsvariablen in [ST_RFID_Control](#) [► 45] parametrisiert werden.

Die zuletzt gesendeten Rohdaten können jederzeit am Ausgang des Funktionsbausteins mithilfe der Struktur [ST_RFID_RawData](#) [► 45] eingesehen werden. Dies ist unabhängig vom verwendeten Befehl.

Empfangen

Die zuletzt empfangenen Rohdaten können jederzeit am Ausgang des Funktionsbausteins mithilfe der Struktur [ST_RFID_RawData](#) [► 45] eingesehen werden. Die zugehörige Adressierung wird in der Struktur [ST_RFID_ReaderInfo](#) [► 43] ausgegeben.

Bei Nutzung des Befehls [SendRawData](#) [► 25] kann eine direkte Auswertung der empfangenen Antwort nicht garantiert werden.

Beispiel: Wird ein Lesebefehl manuell als Bytefolge mittels des Befehls `SendRawData` versandt, so werden empfangene Transponderdaten nicht auf eine in [ST_RFID_AccessData](#) [► 52] angegebene Adresse geschrieben. Eine Auswertung/Speicherung der Daten sollte demnach auch manuell mithilfe der immer angegebenen Rohdatenstruktur [ST_RFID_RawData](#) [► 45] geschehen. Bei den angegebenen Rohdaten handelt es sich um das empfangene Telegramm, allerdings ohne Prefix, Suffix, Checksum, CRC oder Shift-Sequence-Kodierung. Wenn das empfangene Telegramm nicht regulär von dem Funktionsbaustein ausgewertet wurde, wird dies zudem über einen Fehler signalisiert.

Um die empfangenen Daten nutzen zu können, müssen diese beispielsweise auf ein Bytearray kopiert werden.

Beispiel einer Zuweisung von empfangenen Daten mithilfe der Funktion `MEMCPY()`:

```
fbRFIDReader      : FB_RFIDReader;
arrReceivedData  : ARRAY [0..511] OF BYTE;

MEMSET( ADR(arrReceivedData), 0, SIZEOF(arrReceivedData) );
MEMCPY( ADR(arrReceivedData),
        fbRFIDReader.stRawData.pReceivedRsp,
        MIN(fbRFIDReader.stRawData.iReceivedRspLen, SIZEOF(arrReceivedData)) );
```

Balluff RFID Reader: Die Endekennung (LF CR) wird als Suffix zur Erkennung von Telegrammen genutzt. Sobald diese 2 Bytes im Datenstrom erkannt werden, werden vorherige Daten zu einem Telegramm zusammengefasst.

6.2 Datentypen

6.2.1 Strukturen

6.2.1.1 ST_RFID_TranspInfo

Die Struktur ST_RFID_TranspInfo gibt den Typ und die Seriennummer des zuletzt detektierten RFID Transponders an. Mit dem Befehl [GetInventory](#) [[23](#)] werden diese Informationen abgefragt und aktualisiert.

```

TYPE ST_RFID_TranspInfo :
STRUCT
  sSerialNumber : T_RFID_TranspSRN; (* serial number shown as hex coded string(ascii) *)
  eType         : E_RFID_TranspType;
  iHeadNumber  : USINT; (* read head where the last transponder was detected *)
  iDCType      : USINT;
(* data carrier type: the received transponder type code (see device specific type list) *)
END_STRUCT
END_TYPE
    
```

Alle Hersteller

Name	Beschreibung
sSerialNumber	Die Seriennummer des Transponders (häufig 8 Byte) wird im String sSerialNumber in hexadezimaler Darstellung angegeben. Der Datentyp ist T_RFID_TranspSRN [68]. Bei Verwendung von Balluff RFID Readern wird die Seriennummer bei 13,56 Mhz Transpondern im Gesamten byteweise vom Bibliotheksbaustein gedreht. Dies geschieht, weil die ausgelesene Seriennummer eines Transponders andernfalls nicht mit der an einem anderen Reader ausgelesenen Seriennummer übereinstimmen würde. So lassen sich Geräte verschiedener Hersteller gemeinsam in einem Verbund betreiben.
eType	Der Typ des Transponders wird als Enumerationswert der Enumeration E_RFID_TranspType [67] angegeben.
iDCType	Der Typ des Transponders wird als Zahlencode angegeben. Siehe hierzu die proprietäre Geräte-spezifische Transpondertypenliste.

Pepperl+Fuchs

Name	Beschreibung
iHeadNumber	Wenn ein RFID Reader mit mehreren Leseköpfen angeschlossen ist, so gibt iHeadNumber an, von welchem Kopf der RFID Transponder detektiert wurde.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.1.2 ST_RFID_ReaderInfo

Nach dem Befehl [GetReaderVersion](#) [[22](#)] aus dem Befehlssatz werden die empfangenen Daten in dieser Ausgangsstruktur aufbereitet.

Nicht jede Variable wird dabei von jedem RFID-Reader-Modell in Form der Versionsinformation bedient. So gibt ein Reader-Modell beispielsweise das Produktionsdatum bekannt, während ein anderes RFID-Reader-Modell die Hardwareversion übermittelt. Nähere Informationen zu den Angaben finden Sie in den herstellereigenen Protokollspezifikationen und -handbüchern.

```

TYPE ST_RFID_ReaderInfo :
STRUCT
  dDate      : DATE;
    
```

```

eType      : E_RFID_ReaderType;
eGroup     : E_RFID_ReaderGroup;
eManufacturer : E_RFID_ReaderManufacturer;
iReserved  : UINT;
sSWVersion : STRING(31);
sHWVersion : STRING(31);
sCode      : STRING(39);
sSerialNumber : STRING(39);

iSrcAddrRcv : UDINT;          (* RS485 address *)
iDstAddrRcv : UDINT;          (* RS485 address *)
END_STRUCT
END_TYPE

```

Alle Hersteller

Name	Beschreibung
eType	An diesem Ausgang wird der RFID Reader Typ als Enumeration angegeben.
eGroup	An diesem Ausgang wird die RFID Reader Gruppe/Serie angegeben. Durch diese Gruppenzuordnung wird die bibliotheksinterne Verarbeitung aller Telegramme festgelegt.
eManufacturer	An diesem Ausgang wird der Hersteller des angeschlossenen RFID Readers angegeben. Die Enumeration gibt folgende Auswahl: <pre>TYPE E_RFID_ReaderManufacturer : (eRFMR_Unknown, eRFMR_Balluff, eRFMR_Deister, eRFMR_Leuze, eRFMR_PepperlFuchs, eRFMR_Baltech); END_TYPE</pre>
iSrcAddrRcv	Im Falle der RS485-Schnittstelle wird hier die empfangene Quelladresse angegeben.
iDstAddrRcv	Im Falle der RS485-Schnittstelle wird hier die empfangene Zieladresse angegeben.

Baltech

Name	Beschreibung
dDate	An diesem Ausgang wird das Produktionsdatum des RFID Readers angegeben. Das Datum 01.01.1970 bedeutet, dass kein Produktionsdatum übermittelt wurde.
sSWVersion	Gibt die Softwareversion als Text an.
sCode	Der spezielle Typ des RFID Readers wird als Zahlencode übermittelt. Dieser ist am Ausgang sCode als String ausgegeben.
sSerialNumber	Die Seriennummer des RFID Readers wird am Ausgang sSerialNumber als String in hexadezimaler Darstellung ausgegeben. Diese ist nicht zu verwechseln mit der Transponderseriennummer.

Deister

Name	Beschreibung
sSWVersion	Gibt die Softwareversion als Text an.
sHWVersion	Gibt die Hardwareversion als Text an.
sCode	Der spezielle Typ des RFID Readers wird als Zahlencode übermittelt. Dieser ist am Ausgang sCode als String ausgegeben.
sSerialNumber	Die Seriennummer des RFID Readers wird am Ausgang sSerialNumber als String in hexadezimaler Darstellung ausgegeben. Diese ist nicht zu verwechseln mit der Transponderseriennummer.

Leuze

Name	Beschreibung
dDate	An diesem Ausgang wird das Produktionsdatum des RFID Readers angegeben. Das Datum 01.01.1970 bedeutet, dass kein Produktionsdatum übermittelt wurde.
sCode	Der spezielle Typ des RFID Readers wird als Zahlencode übermittelt. Dieser ist am Ausgang sCode als String ausgegeben.

Pepperl+Fuchs:

Name	Beschreibung
dDate	An diesem Ausgang wird das Produktionsdatum des RFID Readers angegeben. Das Datum 01.01.1970 bedeutet, dass kein Produktionsdatum übermittelt wurde.
sSWVersion	Gibt die Softwareversion als Text an.
sCode	Der spezielle Typ des RFID Readers wird als Zahlencode übermittelt. Dieser ist am Ausgang sCode als String ausgegeben.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.1.3 ST_RFID_RawData

Diese Struktur gibt die gesendeten und empfangenen Daten als Rohdaten aus. Dabei handelt es sich immer um das komplette Telegramm, allerdings ohne Prefix, Suffix, Checksum, CRC oder Shift-Sequence-Kodierung. Die Low-Level-Kommunikation wird im Abschnitt [Low-Level-Kommunikation \[▶ 41\]](#) detailliert beschrieben.

Über die angegebenen Pointer können die Bytefolgen eingesehen werden. Zur Auswertung kann die Funktion MEMCPY() genutzt werden.

```

TYPE ST_RFID_RawData :
STRUCT
    pReceivedRsp :POINTER TO BYTE;
    pSentCommand :POINTER TO BYTE;

    iReceivedRspLen :UINT;
    iSentCommandLen :UINT;
END_STRUCT
END_TYPE
    
```

Name	Beschreibung
pReceivedRsp	Ein empfangenes Telegramm ist als Bytefolge abgelegt und der Pointer pReceivedRsp zeigt auf diese Bytefolge.
pSentCommand	Ein gesendetes Telegramm ist als Bytefolge abgelegt und der Pointer pSentCommand zeigt auf diese Bytefolge.
iReceivedRspLen	Gibt die Länge der abgelegten Bytefolge in Bytes an.
iSentCommandLen	Gibt die Länge der abgelegten Bytefolge in Bytes an.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.1.4 ST_RFID_Control

Die Eingangsstruktur stCtrl beinhaltet Variablen, mit denen sich das Verhalten des Funktionsbausteins parametrieren lässt.

Die zwei Variablen `tPreSendDelay` und `tPostSendDelay` bieten optional die Möglichkeit Verzögerungszeiten zu parametrieren.

```

TYPE ST_RFID_Control :
STRUCT
  sSelTranspSRN      :T_RFID_TranspSRN; (* serial number shown as hex coded string(ascii) *)

  tPreSendDelay      :TIME;           (* condition: tTimeOut > tPreSendDelay + tPostSendDelay *)
  tPostSendDelay     :TIME;           (* condition: tTimeOut > tPreSendDelay + tPostSendDelay *)

  iSrcAddrSnd        :UDINT;          (* RS485 address *)
  iDstAddrSnd        :UDINT;          (* RS485 address *)
  bAddrAutoMode      :BOOL := TRUE;  (* if AutoMode is activated the communication addresses are handled automatically and set addresses are not used. *)

  bLogging           :BOOL;

  iHeadNumber        :USINT := 1;    (* if multiple read heads are installed at the reader unit, one can be selected *)
  iDCType            :USINT := 1;

  pRawDataCommand    :POINTER TO BYTE; (* data input for low level communication *)
  iRawDataCommandLen :UINT;

  bBufferedCmd       :BOOL;

  iVHLFileID         :USINT := 16#FF; (* selection of VHL file; default 0xFF (ad hoc VHL file of vhlSetup) *)

  iCardTypeMask      :UINT := 16#FFFF; (* select which card type should be detected via GetInventory; default 0xFFFF (all types) *)
  bReselect          :BOOL := TRUE;   (* with Reselect every GetInventory gets the first item of the reader's detected card list *)
  bAcceptConfCard    :BOOL := TRUE;   (* a read command also accept configuration cards to configure the RFID Reader *)
END_STRUCT
END_TYPE

```

● Mehrere RFID Reader an einem RS485-Netz

I Eine Unterstützung mehrerer RFID Reader an einem RS485-Netz ist mit der TwinCAT-RFID-Bibliothek vorerst nicht gegeben. Es muss je RFID Reader eine separate Verbindung zu einer separaten Klemme erfolgen. In diesem stand-alone Betrieb ist eine individuelle Adressierung mit `iSrcAddrSnd` und `iDstAddrSnd` nicht nötig. Die Adressierung kann demnach automatisch von der TwinCAT-RFID-Bibliothek übernommen werden, wozu die Eingangsvariable `bAddrAutoMode` auf `TRUE` belassen werden kann.

Alle Hersteller

Name	Beschreibung																								
tPreSendDelay	Vor dem Senden eines Kommandos an den RFID Reader wartet der Funktionsbaustein die mit tPreSendDelay angegebene Zeit ab.																								
tPostSendDelay	Nach dem Senden eines Kommandos an den RFID Reader wartet der Funktionsbaustein mindestens die mit tPostSendDelay angegebene Zeit ab. Wenn bis dahin die erwartete Response noch nicht eingetroffen ist, wartet der Funktionsbaustein weiter auf die Response bis maximal die angegebene Timeout-Zeit tTimeOut abgelaufen ist.																								
iSrcAddrSnd	Quelladresse im Falle der RS485-Kommunikation. Diese Adresse wird verwendet, wenn bAddrAutoMode nicht gesetzt ist.																								
iDstAddrSnd	Zieladresse im Falle der RS485-Kommunikation. Diese Adresse wird verwendet, wenn bAddrAutoMode nicht gesetzt ist.																								
bAddrAutoMode	Auswahlmöglichkeit im Falle der RS485-Kommunikation. Wenn bAddrAutoMode aktiviert ist [default: TRUE], werden die RS485-Adressen automatisch zugewiesen. Die oben angegebenen Adressen werden nicht verwendet. Wenn bAddrAutoMode deaktiviert (FALSE) ist, werden oben angegebene Adressen verwendet.																								
bLogging	Mit bLogging lässt sich eine zusätzliche Ausgabe aktivieren. Mithilfe von Log View Messages kann so die serielle Kommunikation nachvollzogen werden. Diese Messages werden im Windows Event Viewer sowie im TwinCAT System Manager sichtbar. Dies ist sinnvoll für Test- und Analysezwecke. Das Format der Ausgabe ist nicht festgelegt, um Erweiterungen zu ermöglichen. <table border="1"> <thead> <tr> <th>Server (Port)</th> <th>Timestamp</th> <th>Message</th> </tr> </thead> <tbody> <tr> <td>TCPLC.PlcAuxTask (801)</td> <td>27.10.2010 11:29:12 50 ms</td> <td>TcPclLibrary RFID (Baltech IDE LP) Rx data (3 bytes): 00007F</td> </tr> <tr> <td>TCPLC.PlcAuxTask (801)</td> <td>27.10.2010 11:29:12 46 ms</td> <td>TcPclLibrary RFID (Baltech IDE LP) Rx data (16 bytes): 1C01020D0048616C6C6F2057656C7421</td> </tr> <tr> <td>TCPLC.PlcAuxTask (801)</td> <td>27.10.2010 11:29:12 2 ms</td> <td>TcPclLibrary RFID (Baltech IDE LP) Tx data (11 bytes): 1C01020500FF000A0000E2</td> </tr> <tr> <td>TCPLC.PlcAuxTask (801)</td> <td>27.10.2010 11:29:11 57 ms</td> <td>TcPclLibrary RFID (Baltech IDE LP) Rx data (6 bytes): 1C010300001E</td> </tr> <tr> <td>TCPLC.PlcAuxTask (801)</td> <td>27.10.2010 11:29:10 993 ms</td> <td>TcPclLibrary RFID (Baltech IDE LP) Tx data (23 bytes): 1C01031100FF000A000C48616C6C6F2057656C7421009B</td> </tr> <tr> <td>TCPLC.PlcAuxTask (801)</td> <td>27.10.2010 11:28:54 610 ms</td> <td>TcPclLibrary RFID (Baltech IDE LP) Rx data (10 bytes): 1C0101040057F4E98BD9</td> </tr> <tr> <td>TCPLC.PlcAuxTask (801)</td> <td>27.10.2010 11:28:54 602 ms</td> <td>TcPclLibrary RFID (Baltech IDE LP) Tx data (6 bytes): 1C010100001C</td> </tr> </tbody> </table> <p>Beim ersten Aufruf des Funktionsbausteins mit bLogging = TRUE, wird die Ausgabe „Logging initialized“. erzeugt. In diesem ersten Zyklus werden keine Kommunikationsdaten überwacht.</p>	Server (Port)	Timestamp	Message	TCPLC.PlcAuxTask (801)	27.10.2010 11:29:12 50 ms	TcPclLibrary RFID (Baltech IDE LP) Rx data (3 bytes): 00007F	TCPLC.PlcAuxTask (801)	27.10.2010 11:29:12 46 ms	TcPclLibrary RFID (Baltech IDE LP) Rx data (16 bytes): 1C01020D0048616C6C6F2057656C7421	TCPLC.PlcAuxTask (801)	27.10.2010 11:29:12 2 ms	TcPclLibrary RFID (Baltech IDE LP) Tx data (11 bytes): 1C01020500FF000A0000E2	TCPLC.PlcAuxTask (801)	27.10.2010 11:29:11 57 ms	TcPclLibrary RFID (Baltech IDE LP) Rx data (6 bytes): 1C010300001E	TCPLC.PlcAuxTask (801)	27.10.2010 11:29:10 993 ms	TcPclLibrary RFID (Baltech IDE LP) Tx data (23 bytes): 1C01031100FF000A000C48616C6C6F2057656C7421009B	TCPLC.PlcAuxTask (801)	27.10.2010 11:28:54 610 ms	TcPclLibrary RFID (Baltech IDE LP) Rx data (10 bytes): 1C0101040057F4E98BD9	TCPLC.PlcAuxTask (801)	27.10.2010 11:28:54 602 ms	TcPclLibrary RFID (Baltech IDE LP) Tx data (6 bytes): 1C010100001C
Server (Port)	Timestamp	Message																							
TCPLC.PlcAuxTask (801)	27.10.2010 11:29:12 50 ms	TcPclLibrary RFID (Baltech IDE LP) Rx data (3 bytes): 00007F																							
TCPLC.PlcAuxTask (801)	27.10.2010 11:29:12 46 ms	TcPclLibrary RFID (Baltech IDE LP) Rx data (16 bytes): 1C01020D0048616C6C6F2057656C7421																							
TCPLC.PlcAuxTask (801)	27.10.2010 11:29:12 2 ms	TcPclLibrary RFID (Baltech IDE LP) Tx data (11 bytes): 1C01020500FF000A0000E2																							
TCPLC.PlcAuxTask (801)	27.10.2010 11:29:11 57 ms	TcPclLibrary RFID (Baltech IDE LP) Rx data (6 bytes): 1C010300001E																							
TCPLC.PlcAuxTask (801)	27.10.2010 11:29:10 993 ms	TcPclLibrary RFID (Baltech IDE LP) Tx data (23 bytes): 1C01031100FF000A000C48616C6C6F2057656C7421009B																							
TCPLC.PlcAuxTask (801)	27.10.2010 11:28:54 610 ms	TcPclLibrary RFID (Baltech IDE LP) Rx data (10 bytes): 1C0101040057F4E98BD9																							
TCPLC.PlcAuxTask (801)	27.10.2010 11:28:54 602 ms	TcPclLibrary RFID (Baltech IDE LP) Tx data (6 bytes): 1C010100001C																							
pRawDataCommand	Der reguläre Funktionsbaustein der TwinCAT-RFID-Bibliothek kann ebenso zur Low-Level-Kommunikation verwendet werden. In dem Fall können Rohdaten direkt an den RFID Reader gesendet werden. Die zu sendenden Rohdaten (z. B. als Bytearray) müssen in dieser Eingangsvariablen angegeben werden. Dabei handelt es sich um einen Pointer auf die zu sendenden Rohdaten. Auf diese Adresse wird von dem Bibliotheksbaustein nur lesend zugegriffen. Weitere Informationen zum Ablauf der Low-Level-Kommunikation sind im Abschnitt Low-Level-Kommunikation [► 41] zusammengefasst. Am Ausgang können die gesendeten sowie die empfangenen Rohdaten jederzeit mittels der Struktur ST RFID RawData [► 45] eingesehen werden.																								
iRawDataCommandLen	Die Eingangsvariable iRawDataCommandLen gibt die Länge in Bytes der mittels dem Pointer pRawDataCommand angegebenen Rohdaten an.																								

Balluff

Name	Beschreibung
iHeadNumber	Wenn ein RFID-Lesegerät mit mehreren Leseköpfen angesprochen wird, so wird in der Eingangsvariablen iHeadNumber eine Wahl des Lesekopfes angegeben.
iDCType	Es kann bei Verwendung von Balluff Readern über iDCType die Blockgröße des Chipspeichers (0→64bytes, 1→32bytes) angegeben werden.

Baltech

Name	Beschreibung																					
iVHLFileID	Schreib- und Lesebefehle werden mit der hier ausgewählten VHL-Datei ausgeführt. Diese muss in der Konfiguration des RFID-Gerätes hinterlegt sein. Wenn iVHLFileID den Wert 0xFF [default] hat, so wird der RFID Reader keine normale VHL-Datei aus seiner Konfiguration nehmen, sondern eine einfache Ad-Hoc-Datei, welche gewählt werden kann, um nicht konfigurierte/unverschlüsselte Blankokarten zu verwenden.																					
iCardTypeMask	Mithilfe der iCardTypeMask lassen sich bestimmte Transponderkartenfamilien einstellen. Nur die gesetzten Kartentypen werden mit dem Befehl GetInventory erkannt. Alle anderen Kartentypen werden herausgefiltert. Sollen alle Kartentypen erkannt werden, muss dieser Wert nicht verändert werden [default: 0xFFFF]. Für jede akzeptierte Kartenfamilie wird in der iCardTypeMask das jeweilige Bit gesetzt. <table border="1" data-bbox="475 656 1426 931"> <thead> <tr> <th>Card Type</th> <th>Card Family</th> <th>CardTypeMask</th> </tr> </thead> <tbody> <tr> <td>Mifare / ISO 14443-A</td> <td>1</td> <td>0x0001 (Bit 1)</td> </tr> <tr> <td>LEGIC</td> <td>2</td> <td>0x0002 (Bit 2)</td> </tr> <tr> <td>ISO 15693</td> <td>3</td> <td>0x0004 (Bit 3)</td> </tr> <tr> <td>ISO 14443-B</td> <td>4</td> <td>0x0008 (Bit 4)</td> </tr> <tr> <td>PICOPASS via ISO 14443-2-B</td> <td>5</td> <td>0x0010 (Bit 5)</td> </tr> <tr> <td>PICOPASS via ISO 15693</td> <td>6</td> <td>0x0020 (Bit 6)</td> </tr> </tbody> </table>	Card Type	Card Family	CardTypeMask	Mifare / ISO 14443-A	1	0x0001 (Bit 1)	LEGIC	2	0x0002 (Bit 2)	ISO 15693	3	0x0004 (Bit 3)	ISO 14443-B	4	0x0008 (Bit 4)	PICOPASS via ISO 14443-2-B	5	0x0010 (Bit 5)	PICOPASS via ISO 15693	6	0x0020 (Bit 6)
Card Type	Card Family	CardTypeMask																				
Mifare / ISO 14443-A	1	0x0001 (Bit 1)																				
LEGIC	2	0x0002 (Bit 2)																				
ISO 15693	3	0x0004 (Bit 3)																				
ISO 14443-B	4	0x0008 (Bit 4)																				
PICOPASS via ISO 14443-2-B	5	0x0010 (Bit 5)																				
PICOPASS via ISO 15693	6	0x0020 (Bit 6)																				
bReselect	Wenn bReselect gesetzt ist [default: TRUE], wird bei jeder Abfrage mit dem Befehl GetInventory [► 23] ein Transponder am Ausgang angegeben, sofern sich ein Transponder im HF-Feld befindet. Wenn sichergestellt ist, dass sich jeweils nur eine Karte vor dem RFID Reader befindet, sollte diese Einstellung beibehalten werden. Wenn sich mehr als eine Transponderkarte im HF-Feld befindet, existiert im RFID-Gerät eine Liste erkannter Karten. Um alle Transponder mit dem Befehl GetInventory anwählen zu können, sollte bReselect ausgeschaltet [FALSE] werden. Der Befehl GetInventory geht diese Liste dann durch mehrfaches Aufrufen einem nach dem anderen durch. Nachdem die letzte Karte angewählt wurde, folgt die Rückmeldung eRFR_NoTransponder. Erst wenn eine Karte aus den HF-Feld herausgenommen und erneut hineingeführt wird, gibt der Befehl GetInventory den Transponder wieder samt Seriennummer am Ausgang an.																					
bAcceptConfCard	Wenn bAcceptConfCard gesetzt ist [default: TRUE], werden mit dem Befehl GetInventory auch Konfigurationskarten erkannt. Es wird automatisch versucht, die darauf vorhandene Konfiguration auf den RFID Reader zu übertragen. Um eine Beeinflussung durch fremde Konfigurationskarten zu verhindern, kann dieser Parameter deaktiviert werden.																					

Leuze

Name	Beschreibung
sSelTranspSRN	An der Eingangsvariablen sSelTranspSRN kann die Seriennummer des Transponders, auf den der Befehl (z. B. Read/Write) angewendet werden soll, als String angegeben werden. Der Datentyp ist <code>T_RFID_TranspSRN</code> [► 68]. Dies ist in den meisten Fällen nicht notwendig. Wenn eine bestimmte Reader-Konfiguration dies notwendig macht, sind Details hierzu bei der diesbezüglichen Beschreibung in der proprietären Spezifikation zu finden.

Pepperl+Fuchs

Name	Beschreibung																																																				
iHeadNumber	Wenn ein RFID-Lesegerät mit mehreren Leseköpfen angesprochen wird, so wird in der Eingangsvariablen iHeadNumber eine Wahl des Lesekopfes angegeben. iHeadNumber wird auch als IdentChannel bezeichnet.																																																				
iDCType	<p>Mit dieser Variablen und dem Befehl ChangeDCType [► 25] kann bei Pepperl+Fuchs Readern der Transpondertyp am Lesekopf eingestellt werden. Mögliche Werte dafür sind im proprietären Protokoll beim Befehl ChangeTag gelistet. (Sie stimmen nicht mit den Werten der Enumeration E_RFID_TranspType überein.)</p> <p>Auszug unterstützter Transpondertypen:</p> <table border="1" data-bbox="480 584 1425 1077"> <thead> <tr> <th>Data Carrier Type [dec]</th> <th>Description P+F</th> <th>Chip Type</th> <th>Frequency</th> </tr> </thead> <tbody> <tr> <td>02</td> <td>IPC02</td> <td>Unique, EM4102</td> <td>125 KHz</td> </tr> <tr> <td>03</td> <td>IPC03</td> <td>EM4450</td> <td>125 KHz</td> </tr> <tr> <td>14</td> <td>IPC14</td> <td>T5557 (Atmel)</td> <td>125 KHz</td> </tr> <tr> <td>20</td> <td></td> <td>all ISO 15693 complaint data carriers</td> <td>13.56 MHz</td> </tr> <tr> <td>21</td> <td>IQC21</td> <td>I-Code SLI (NXP)</td> <td>13.56 MHz</td> </tr> <tr> <td>22</td> <td>IQC22</td> <td>Tag-it HF-I Plus (TI)</td> <td>13.56 MHz</td> </tr> <tr> <td>23</td> <td>IQC23</td> <td>my-D (infineon) SRF55V02P</td> <td>13.56 MHz</td> </tr> <tr> <td>24</td> <td>IQC24</td> <td>my-D (infineon) SRF55V10P</td> <td>13.56 MHz</td> </tr> <tr> <td>33</td> <td>IQC33</td> <td>Fujitsu FRAM MB89R118</td> <td>13.56 MHz</td> </tr> <tr> <td>35</td> <td>IQC35</td> <td>I-Code SLI-S (NXP)</td> <td>13.56 MHz</td> </tr> <tr> <td>...</td> <td>...</td> <td>...</td> <td>...</td> </tr> <tr> <td>99</td> <td></td> <td>default type of the connected read head</td> <td></td> </tr> </tbody> </table>	Data Carrier Type [dec]	Description P+F	Chip Type	Frequency	02	IPC02	Unique, EM4102	125 KHz	03	IPC03	EM4450	125 KHz	14	IPC14	T5557 (Atmel)	125 KHz	20		all ISO 15693 complaint data carriers	13.56 MHz	21	IQC21	I-Code SLI (NXP)	13.56 MHz	22	IQC22	Tag-it HF-I Plus (TI)	13.56 MHz	23	IQC23	my-D (infineon) SRF55V02P	13.56 MHz	24	IQC24	my-D (infineon) SRF55V10P	13.56 MHz	33	IQC33	Fujitsu FRAM MB89R118	13.56 MHz	35	IQC35	I-Code SLI-S (NXP)	13.56 MHz	99		default type of the connected read head	
Data Carrier Type [dec]	Description P+F	Chip Type	Frequency																																																		
02	IPC02	Unique, EM4102	125 KHz																																																		
03	IPC03	EM4450	125 KHz																																																		
14	IPC14	T5557 (Atmel)	125 KHz																																																		
20		all ISO 15693 complaint data carriers	13.56 MHz																																																		
21	IQC21	I-Code SLI (NXP)	13.56 MHz																																																		
22	IQC22	Tag-it HF-I Plus (TI)	13.56 MHz																																																		
23	IQC23	my-D (infineon) SRF55V02P	13.56 MHz																																																		
24	IQC24	my-D (infineon) SRF55V10P	13.56 MHz																																																		
33	IQC33	Fujitsu FRAM MB89R118	13.56 MHz																																																		
35	IQC35	I-Code SLI-S (NXP)	13.56 MHz																																																		
...																																																		
99		default type of the connected read head																																																			
bBufferedCmd	<p>Die meisten Befehle werden umgehend nach dem Aufruf einmal bearbeitet. Je nach RFID Reader können Befehle dauerhaft anstehen. Dies ermöglicht unter anderem einen Lesevorgang, sobald der RFID Transponder in das Lesefeld kommt, ohne einen extra Befehl dazu abzuschicken. Entweder ist dies in der RFID-Reader-Konfiguration konfigurierbar (Balluff, Deister, Leuze) oder mit dieser Eingangsvariablen auswählbar (Pepperl+Fuchs).</p> <p>Ist an einem Lesekopf ein solch gepufferter Befehl aktiv, darf der Trigger Mode nicht für diesen Kanal aktiviert werden bzw. aktiv sein. Ebenso darf kein Rohdatenbefehl abgesetzt werden, welcher diesen Kanal betrifft.</p>																																																				

Verzögerungszeiten

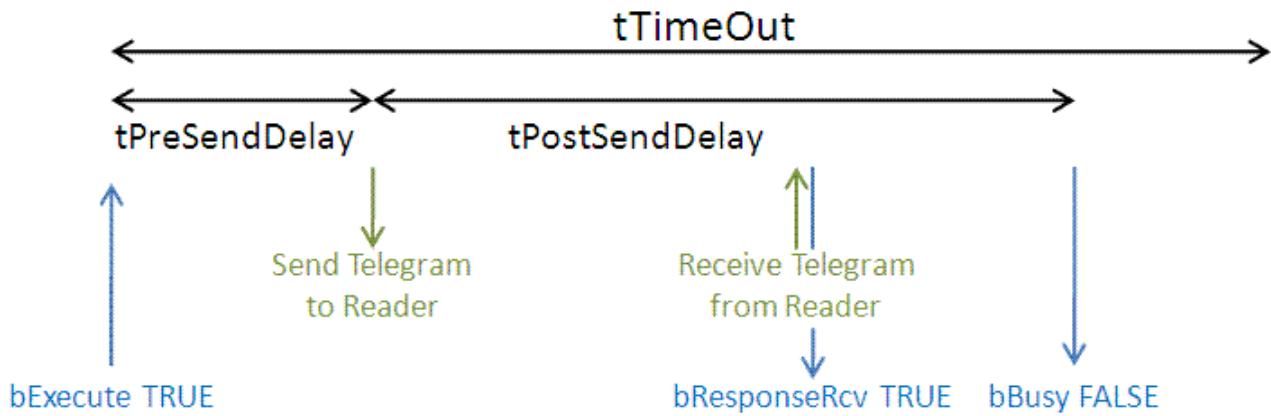
Die zwei Variablen tPreSendDelay und tPostSendDelay bieten optional die Möglichkeit Verzögerungszeiten zu parametrieren. Beide Variablen stellen sicher, dass zwischen zwei Anfragen an den RFID Reader eine Verzögerung abgewartet wird.

Wird die Verzögerungszeit als tPreSendDelay angegeben, so ist eine Verzögerung zwischen dem letzten Antworttelegramm und dem nächsten Anfragetelegramm sichergestellt. Soll das Anfragetelegramm möglichst direkt abgeschickt werden, kann tPostSendDelay verwendet werden.

Es gilt die Bedingung $tTimeOut > tPreSendDelay + tPostSendDelay$. Andernfalls wird ein Fehler am Ausgang ausgegeben.

Im proprietären Protokoll der Balluff RFID Reader ist eine Mindestverzögerungszeit zwischen zwei Befehlen von 300 ms spezifiziert.

Im proprietären Protokoll der Leuze electronic RFID Reader ist eine Mindestverzögerungszeit zwischen Empfang und Befehl von 150 ms spezifiziert.



Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.1.5 ST_RFID_ConfigIn

Am Eingang des RFID-Funktionsbausteins gibt diese Struktur die Möglichkeit eine beliebige Konfiguration auf den RFID Reader zu übertragen.

Die zuletzt gelesene RFID-Reader-Konfiguration ist am Ausgang mit der Struktur ST_RFID_Config [▶ 52] angegeben. In deren Beschreibung finden sich auch ergänzende Informationen zu den Konfigurationen.

Konfigurationsdaten können in Form einer spezifischen Konfigurationsstruktur (ST_RFID_CfgStruct_DeisterUDL, ST_RFID_CfgStruct_LeuzeRFM, ...) oder auch in Form eines Konfigurationsregisters (Byte Array) vorliegen. Mit der Variable **bUseCfgReg** lässt sich diese Auswahl treffen.

```
(* defines the configuration input parameters.
The data can be set via Config structure or Config register.
Different RFID Reader in different ReaderGroups can differ in their configuration data. *)
TYPE ST_RFID_ConfigIn :
STRUCT
  pCfg          : POINTER TO BYTE; (* pointer to config structure or register *)
  iCfgSize      : UINT := 0;      (* size in bytes of the structure or register *)

  bUseCfgReg    : BOOL := FALSE; (* Set Config via Register instead of CfgStructure *)
  bUseCfgDefault : BOOL := TRUE; (* Set Config using default parameters beside CfgStructure *)

  (* An additional option to demand/
set a specific config parameter without transmission of the whole config register.
Not possible at all reader models.
Set a desired value before calling GetConfig/
SetConfig or keep the default for full register request. *)
  iRegIdx      : UINT := 0;
  iRegGroup    : USINT := 0;      (* 0:full register; 1:reg.00-0F; 2:single register *)

  bReserved    : BOOL;
END_STRUCT
END_TYPE
```

Baltech

Name	Beschreibung
pCfg	Dieser Pointer muss die Speicheradresse der zu schreibenden Konfiguration beinhalten. Dabei handelt es sich um die Konfigurationsstruktur ST_RFID_CfgStruct_BaltechMifVHLFile.
iCfgSize	Diese Eingangsvariable gibt die Länge in Bytes der über den Pointer angegebenen Konfigurationsdaten an.

Deister

Name	Beschreibung
pCfg	Dieser Pointer muss die Speicheradresse der zu schreibenden Konfiguration beinhalten. Dabei kann es sich um eine Konfigurationsstruktur sowie auch ein Konfigurationsregister handeln.
iCfgSize	Diese Eingangsvariable gibt die Länge in Bytes der über den Pointer angegebenen Konfigurationsdaten an.
bUseCfgReg	Wird die Eingangsvariable bUseCfgReg gesetzt (TRUE), so kann über den Pointer pCfg ein Konfigurationsregister (Byte Array) adressiert werden anstatt einer Konfigurationsstruktur. Per Default wird eine spezifische Konfigurationsstruktur angegeben.
bUseCfgDefault	Dieser Parameter ist nur relevant, wenn die Konfigurationsdaten in Form einer spezifischen Konfigurationsstruktur anliegen. Eine Konfigurationsstruktur ist keine namentliche Gesamtdarstellung des Konfigurationsregisters. Die Struktur beinhaltet nur die wichtigsten Konfigurationsparameter. Wird die Eingangsvariable bUseCfgDefault gesetzt (TRUE), so werden für die nicht angegebenen Konfigurationsparameter Standardwerte verwendet. Andernfalls wird der Wert dieser Konfigurationsparameter nicht verändert, weil die zuletzt gelesenen Werte wiederverwendet werden.

Leuze

Name	Beschreibung
pCfg	Dieser Pointer muss die Speicheradresse der zu schreibenden Konfiguration beinhalten. Dabei kann es sich um eine Konfigurationsstruktur sowie auch ein Konfigurationsregister handeln.
iCfgSize	Diese Eingangsvariable gibt die Länge in Bytes der über den Pointer angegebenen Konfigurationsdaten an.
bUseCfgReg	Wird die Eingangsvariable bUseCfgReg gesetzt (TRUE), so kann über den Pointer pCfg ein Konfigurationsregister (Byte Array) adressiert werden anstatt einer Konfigurationsstruktur. Per Default wird eine spezifische Konfigurationsstruktur angegeben.
bUseCfgDefault	Dieser Parameter ist nur relevant, wenn die Konfigurationsdaten in Form einer spezifischen Konfigurationsstruktur anliegen. Eine Konfigurationsstruktur ist keine namentliche Gesamtdarstellung des Konfigurationsregisters. Die Struktur beinhaltet nur die wichtigsten Konfigurationsparameter. Wird die Eingangsvariable bUseCfgDefault gesetzt (TRUE), so werden für die nicht angegebenen Konfigurationsparameter Standardwerte verwendet. Andernfalls wird der Wert dieser Konfigurationsparameter nicht verändert, weil die zuletzt gelesenen Werte wiederverwendet werden.
iRegIdx	Wird an iRegIdx ein spezieller Index angegeben, so wird alleinig dieser Index des Konfigurationsregisters geändert/gelesen. Dazu muss iRegGroup zudem als „SingleRegister“ angegeben sein. Diese Konfigurationsvariante ist nur für Leuze electronic RFID Reader verfügbar.
iRegGroup	Drei Werte stehen zur Verfügung: 0, um das gesamte Konfigurationsregister zu ändern/lesen; 1, um die Indizes 16#00-16#0F des Registers zu ändern/lesen; 2, um einen einzelnen Index des Registers zu ändern/lesen, wozu dieser mit iRegIdx angegeben werden muss. Diese Konfigurationsvariante ist nur für Leuze electronic RFID Reader verfügbar.

Weitere Informationen zur RFID-Reader-Konfiguration sind im Abschnitt [Konfiguration](#) [▶ 40] zusammengefasst.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.1.6 ST_RFID_Config

Die Struktur gibt die zuletzt gelesene RFID-Reader-Konfiguration an. Dabei handelt es sich nicht um die Parametrierung der TwinCAT-RFID-Bibliothek, sondern um die proprietäre Konfiguration des RFID Readers. Diese kann mit dem Befehl `eRFC_GetConfig` abgefragt werden (siehe [RFID-Befehlssatz](#) | 211).

Jede Konfiguration kann als Register (Byte Array) oder als Struktur gesehen werden. So gibt es in der TwinCAT-RFID-Bibliothek verschiedene Konfigurationsstrukturen (`ST_RFID_CfgStruct_DeisterUDL`, `ST_RFID_CfgStruct_LeuzeRFM`, ...), welche die Rohdaten der Konfigurationsregister unterschiedlicher RFID Reader aufarbeiten. Am Ausgang des Funktionsbausteins der Bibliothek werden beide Varianten zur Verfügung gestellt. Dies geschieht über Pointer. Zur weiteren Auswertung kann die Funktion `MEMCPY()` genutzt werden mit der angegebenen Datenlänge in Bytes.

```
(* defines the configuration as structure and register.
Different RFID Reader in different ReaderGroups can differ in their configuration data. *)
TYPE ST_RFID_Config :
STRUCT
  pCfgStruct      : POINTER TO BYTE;    (* pointer to config structure *)
  pCfgReg         : POINTER TO BYTE;    (* pointer to config register *)
  iCfgStructSize  : UINT := 0;          (* size in bytes of the structure *)
  iCfgRegSize     : UINT := 0;          (* size in bytes of the register *)
END_STRUCT
END_TYPE
```

Name	Beschreibung
<code>pCfgStruct</code>	Dieser Pointer gibt die Speicheradresse der spezifischen Konfigurationsstruktur an.
<code>pCfgReg</code>	Dieser Pointer gibt die Speicheradresse des spezifischen Konfigurationsregisters an.
<code>iCfgStructSize</code>	Diese Ausgangsvariable gibt die Länge in Bytes der spezifischen Konfigurationsstruktur an.
<code>iCfgRegSize</code>	Diese Ausgangsvariable gibt die Länge in Bytes des spezifischen Konfigurationsregisters an. Wenn <code>iCfgRegSize = 0</code> ist, sind die Konfigurationsdaten nicht als Register (Byte Array) verfügbar.

Weitere Informationen zur RFID-Reader-Konfiguration sind im Abschnitt [Konfiguration](#) | 401 zusammengefasst.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.1.7 ST_RFID_AccessData

Wenn ein Lese- oder Schreibbefehl ausgeführt werden soll, ist es notwendig die Eingangsstruktur `stAccessData` anzugeben.

Mit dieser Struktur wird angegeben, wie viele und welche Daten gelesen und wo diese abgespeichert werden sollen bzw. wie viele und welche Daten geschrieben werden sollen.

```
TYPE ST_RFID_AccessData :
STRUCT
  (* access specific parameters *)
  pData      : POINTER TO BYTE; (* pointer to write data or free space for read data *)
  iDataSize  : UINT;            (* length of data buffer in Bytes *)
  iStartBlock : UINT;           (* attend that the UserDataStartBlock which is not obligatory 0 is added
automatically. *)
  iBlockCount : UINT;           (* condition: Blockcount*Blocksize=Datasize *)
  iBlockSize  : UINT := 1;     (* in Bytes *)

  iUserDataStartBlock : UINT := 0; (* depending on the transponder type its user data memory start
s with block index 0 or higher *)
  (* The upper parameter iStartBlock depends on the iUserDataStartBlock. The used StartBlock is iS
tartBlock+iUserDataStartBlock. *)
  (* Different RFID Readers can differ in their interpretation of the first block. *)
  iReserved : UINT;
END_STRUCT
END_TYPE
```

Name	Beschreibung
pData	Der Pointer pData zeigt auf die zu schreibenden Daten bzw. auf den freien Speicherplatz für die zu lesenden Daten.
iDataSize	Gibt die Größe der zu schreibenden/lesenden Daten in Bytes an.
iStartBlock	Gibt den ersten Blockindex an, ab dem gelesen bzw. auf den Speicher des Transponders geschrieben werden soll. Verschiedene RFID Reader Modelle interpretieren diesen Index allerdings teilweise unterschiedlich. Beispiel: Auf einen Lesebefehl mit Startindex 0 gibt Reader A den ersten Block Nutzdaten zurück und Reader B die Seriennummer.
iBlockCount	Gibt die Anzahl von Blöcken an, die gelesen bzw. geschrieben werden soll.
iBlockSize	Mit der Variablen iBlockSize kann die Blockgröße der Nutzdaten (in Bytes) angegeben werden. Je nach Transponder und RFID-Reader-Modell sind hier nur bestimmte Einstellungen möglich (gebräuchlich sind z.B. 8,4 oder 1 Byte). Dies sollte vorab aus den Transponderinformationen in Erfahrung gebracht und getestet werden. Ebenso ist die Variable iBlockSize in Übereinstimmung mit der Einstellung in der RFID-Reader-Konfiguration zu wählen. Andernfalls ist es teilweise möglich, dass ein Zugriff auf den Transponder oder die Auswertung der empfangenen Daten nicht erfolgen kann.
iUserDataStartBlock	Mit der Variablen iUserDataStartBlock kann optional der Startblock (als Index von Blöcken) der Nutzdaten auf dem Transponder angegeben werden. Die Blockgröße (iBlockSize) ist zu beachten. Je nach Transponder können dessen erste Blöcke für Systemdaten wie der Seriennummer reserviert sein. Der Bereich der Nutzdaten kann dementsprechend bei Index 0 beginnen oder auch bei einem höheren Wert. Falls dies der Fall ist kann die Variable iUserDataStartBlock genutzt werden, um diesen zusätzlichen Parameter anzugeben und den eigentlichen Index iStartBlock gleich zu belassen. Intern werden beide Werte zusammen addiert.

i Zugriff unterschiedlicher RFID Reader auf denselben Transponder

Wenn unterschiedliche RFID Reader auf denselben Transponder zugreifen sollen, so muss der Speicherzugriff auf dem Transponder vorher getestet werden. Es ist möglich, dass ein Reader-Modell die Datenblöcke in verdrehter Bytereihenfolge auf dem Transponder ablegt im Vergleich zu einem anderen Reader-Modell. Oder von einem Reader-Modell wird der gesamte Speicherbereich in umgekehrter Reihenfolge gesehen als es ein anderes Reader-Modell tut. Ebenso kann die lesbare Speichergröße des Transponders zwischen verschiedenen Reader-Modellen leicht variieren. Dies ist zusätzlich abhängig vom Transpondertypen. Die TwinCAT-RFID-Bibliothek nimmt darauf keinen Einfluss. Der Anwender muss obige Eingangsparameter dementsprechend wählen.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.1.8 Konfigurationsdaten

6.2.1.8.1 ST_RFID_CfgStruct_BaltechMifVHLFile

Die Struktur ist für das Schreiben mit dem Befehl eRFC_SetConfig geeignet (siehe [RFID-Befehlssatz](#) [▶ 21]).

Dabei handelt es sich nicht um die Parametrierung der TwinCAT-RFID-Bibliothek, sondern um die proprietäre Konfiguration des RFID Readers.

```

TYPE ST_RFID_CfgStruct_BaltechMifVHLFile :
STRUCT
  iVHLFile      : USINT := 1;          (* nr. of VHL file to configure *)
  iNrOfKeys     : USINT(1..8) := 1;
  iNrOfSectors  : USINT(1..56) := 16;  (* default: 16 sectors -
> 1024 bytes mifare card with 752 bytes user data *)
  iRC500EEPoffset : USINT := 16#FF;
  arrKeyList    : ARRAY [0..7] OF T_RFID_MifareKey; (* up to 8 keys, 6 byte each *)
  arrSectorList : ARRAY [0..55] OF BYTE          (* up to 56 sectors accessible *)

```


Name	Beschreibung
iVHLFile	Mit iVHLFile wird die Nummer der zu konfigurierenden VHL-Datei angegeben. In der Konfiguration des RFID-Gerätes können mehrere VHL-Dateien nebeneinander existieren.
iNrOfKeys	Mit iNrOfKeys wird die erforderliche Anzahl an Schlüsseln vorgegeben. Es können 1 bis 8 Schlüssel definiert werden.
iNrOfSectors	Mit iNrOfSectors wird die Anzahl an Sektoren angegeben, die für Nutzdaten verwendet werden sollen. Eine 1-KB-Mifare-Karte besitzt 16 Sektoren [default: 16]. Beispiel: Sollen nur die Sektoren 4-6 verwendet werden, so wird iNrOfSectors = 3 angegeben.
iRC500EEPOffset	Dieser Parameter betrifft die interne Übertragung der Schlüssel innerhalb der Hardware des RFID-Gerätes. Die Standardeinstellung [default: 16#FF] garantiert erhöhte Sicherheit. Eine Änderung wird nicht empfohlen.
arrKeyList	In dem Array arrKeyList werden alle Schlüssel hinterlegt. Ein Schlüssel ist vom Typ T_RFID_MifareKey und besteht aus 6 Byte. Beispiel: Sollen zwei Schlüssel verwendet werden, so werden bei arrKeyList[0] und arrKeyList[1] die jeweiligen Schlüssel hinterlegt. <pre>TYPE T_RFID_MifareKey : ARRAY[0..5] OF BYTE; END_TYPE</pre>
arrSectorList	In dem Array arrSectorList werden alle Sektoren hinterlegt, die für Nutzdaten verwendet werden sollen. Beispiel: Sollen nur die Sektoren 4-6 verwendet werden, so wird arrSectorList[0]=4, arrSectorList[1]=5, arrSectorList[2]=6 angegeben. Das Array ist bereits mit durchlaufender Nummerierung initialisiert. In den meisten Fällen muss deshalb keine Änderung vorgenommen werden.
arrRdKeyAssign	In dem Array arrRdKeyAssign wird der Schlüsselindex für jeden verwendeten Sektor hinterlegt. Beispiel: Bei einer 1-KB-Mifare-Karte sollen alle Sektoren verwendet werden (iNrOfSectors = 16). Zwei Schlüssel werden verwendet (iNrOfKeys = 2) Die erste Hälfte der Sektoren auf dieser Karte ist mit dem ersten Schlüssel (arrKeyList[0]) zu lesen und die zweite Hälfte mit dem zweiten Schlüssel (arrKeyList[1]). In dem Array arrRdKeyAssign müssen demnach die Indizes stCfg : ST_RFID_CfgStruct_BaltechMifVHLFile := (arrRdKeyAssign:=0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1); hinterlegt werden. Soll für alle Sektoren ein Schlüssel verwendet werden, handelt es sich für alle Sektoren um den Schlüsselindex 0. Das Array ist bereits mit 0 initialisiert. Deshalb muss in diesem Fall keine Änderung vorgenommen werden.
arrWrKeyAssign	In dem Array arrWrKeyAssign wird der Schlüsselindex für jeden verwendeten Sektor hinterlegt. Beispiel: Bei einer 1-KB-Mifare-Karte sollen alle Sektoren verwendet werden (iNrOfSectors = 16). Zwei Schlüssel werden verwendet (iNrOfKeys = 2) Die erste Hälfte der Sektoren auf dieser Karte ist mit dem ersten Schlüssel (arrKeyList[0]) zu beschreiben und die zweite Hälfte mit dem zweiten Schlüssel (arrKeyList[1]). In dem Array arrWrKeyAssign müssen demnach die Indizes stCfg:ST_RFID_CfgStruct_BaltechMifVHLFile:=(arrWrKeyAssign:=0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1); hinterlegt werden. Soll für alle Sektoren ein Schlüssel verwendet werden, handelt es sich für alle Sektoren um den Schlüsselindex 0. Das Array ist bereits mit 0 initialisiert. Deshalb muss in diesem Fall keine Änderung vorgenommen werden.
bMAD_Mode	Wenn MAD (Mifare Application Directory) mit AIDs (Application Identifiers) anstatt der Sektorzuweisung verwendet werden soll, so muss bMAD_Mode gesetzt (TRUE) werden. Als Standard wird jedoch die Sektorzuweisung verwendet [default: FALSE].
iMAD_AID	Der Eingang wird nur benötigt, wenn MAD (Mifare Application Directory) verwendet wird (bMAD_Mode = TRUE). Am Konfigurationseingang iMAD_AID wird die MAD AID (Application Identifier) für die VHL-Datei angegeben.

Weitere Informationen zum Ablauf der RFID-Reader-Konfiguration sind im Abschnitt [Konfiguration \[► 40\]](#) zusammengefasst.

Detaillierte Informationen zu den Themen VHL-Datei und Konfiguration von Baltech RFID-Geräten finden Sie zudem in den herstellereigenen Dokumentationen [Mifare.pdf](#) und [ConfigurationValues.pdf](#) im Baltech SDK.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.1.8.2 ST_RFID_CfgStruct_DeisterRDL

Die Struktur ist für das Schreiben mit `eRFC_SetConfig` sowie das Lesen mit `eRFC_GetConfig` geeignet (siehe [RFID-Befehlssatz \[► 21\]](#)).

Dabei handelt es sich nicht um die Parametrierung der TwinCAT-RFID-Bibliothek, sondern um die proprietäre Konfiguration des RFID Readers.

```

TYPE ST_RFID_CfgStruct_DeisterRDL :
STRUCT
  eOpMode      : E_RFID_OpMode := eRFOP_ReadData;
  eTriggerMode : E_RFID_TriggerMode := eRFTR_ImmediateRead;
  eReadMode    : E_RFID_ReadMode := eRFRD_SingleShot;
  eWriteMode   : E_RFID_WriteMode := eRFWR_ImmediateWrite;

  eNetworkMode : E_RFID_NetworkMode := eRFNM_StandAlone;
  bAFI : BOOL := FALSE; (* not implemented; ready for future extention *)
  iAFI : BYTE; (* not implemented; ready for future extention *)

  bSerialNumberMode : BOOL := FALSE;
  bMultiTranspMode : BOOL := FALSE;
  bOutputAutomatic : BOOL := TRUE;
  iBlockSize       : USINT := 8;

  tOutputPulseTime : TIME := T#300ms;

  eTranspType : E_RFID_TranspType := eRFTT_TagItHfi;

  iCountBlocksRead : USINT := 1;
  iCountBlocksWrite : USINT := 1;

  iStartBlockRead : UINT := 16#4000;
  iStartBlockWrite : UINT := 5;
  arrWriteData : ARRAY [0..71] OF BYTE;
END_STRUCT
END_TYPE

```


Name	Beschreibung
eOpMode	<p>Die Betriebsart legt fest, welche Funktion durch einen Triggerimpuls ausgelöst wird. Der Befehl eRFC_TriggerOn oder ein Impuls am optionalen Triggereingang löst die hier eingestellte Aktion aus.</p> <p>Wenn eRFOP_WriteData eingestellt ist, wird ein Schreibzugriff ausgeführt.</p> <p>Wenn eRFOP_ReadData eingestellt ist, wird ein Lesezugriff ausgeführt [default].</p> <p>Das darauffolgende Antworttelegramm wird vom Funktionsbaustein der RFID-Bibliothek empfangen. Eine Zuweisung von gelesenen Transponderdaten ist in dem Fall nicht gegeben. Die empfangenen Rohdaten können zur weiteren Verarbeitung dem Bausteininterface entnommen werden.</p> <pre data-bbox="432 517 767 629"> TYPE E_RFID_OpMode : (eRFOP_WriteData, eRFOP_ReadData, eRFOP_ReadSerialNumber);END_TYPE </pre>
eTriggerMode	<p>Wenn eRFTR_ImmediateRead eingestellt ist, ist das Gerät immer lesebereit. Durch diese Einstellung gilt die Triggerbedingung immer als erfüllt [default].</p> <p>Wenn eRFTR_ReadWithTrigger eingestellt ist, liest das Gerät nur bei Triggerbedingung. Dazu kann der Befehl eRFC_TriggerOn genutzt werden (siehe RFID-Befehlssatz [▶ 21])</p> <p>Das darauffolgende Antworttelegramm wird vom Funktionsbaustein der TwinCAT-RFID-Bibliothek empfangen. Eine Zuweisung von gelesenen Transponderdaten ist in dem Fall nicht gegeben. Die empfangenen Rohdaten können zur weiteren Verarbeitung dem Bausteininterface entnommen werden.</p> <pre data-bbox="432 958 767 1048"> TYPE E_RFID_TriggerMode : (eRFTR_ImmediateRead, eRFTR_ReadWithTrigger);END_TYPE </pre>
eReadMode	<p>Falls eRFRD_ContinuousRead eingestellt ist, liest das Gerät dauerhaft und gibt ebenfalls dauerhaft gelesene Daten aus.</p> <p>Falls eRFRD_SingleShot eingestellt ist, liest das Gerät genau einmal [default].</p> <pre data-bbox="432 1167 767 1256"> TYPE E_RFID_ReadMode : (eRFRD_ContinuousRead, eRFRD_SingleShot);END_TYPE </pre>
eWriteMode	<p>Wenn eRFWR_ImmediateWrite eingestellt ist, muss der Transponder im Feld sein, um einen Schreib- oder Lesebefehl korrekt auszuführen [default].</p> <p>Wenn eRFWR_WriteToNextTag eingestellt ist, werden die Daten eines Schreibbefehles in den nächst folgenden Transponder geschrieben. („Vorspannen“)</p> <pre data-bbox="432 1413 767 1503"> TYPE E_RFID_WriteMode : (eRFWR_ImmediateWrite, eRFWR_WriteToNextTag);END_TYPE </pre>
eNetworkMode	<p>Wenn eRFNM_Network eingestellt ist, können mehrere Geräte in einem RS485-Netzwerk eingebunden sein.</p> <p>Wenn eRFNM_StandAlone eingestellt ist, befindet sich das Gerät im stand-alone Betrieb [default].</p> <p>Der Betrieb von mehreren Geräten innerhalb eines RS485-Netzwerkes wird von der Bibliothek nicht unterstützt.</p> <pre data-bbox="432 1738 767 1827"> TYPE E_RFID_NetworkMode : (eRFNM_Network, eRFNM_StandAlone);END_TYPE </pre>
bSerialNumberMode	<p>Wenn bSerialNumberMode TRUE ist, wird die Seriennummer bei Schreib- und Lesebefehlen mit übertragen.</p> <p>Im Standardfall entspricht dies der zuletzt mit dem Befehl GetInventory [▶ 23] detektierten Transponderseriennummer. Andernfalls wird die Transponderseriennummer durch Angabe in ST RFID Control [▶ 45] festgelegt.</p>
bMultiTranspMode	<p>Wenn bMultiTranspMode TRUE ist, so ist Antikollision aktiv, wenn mehrere Transponder im Feld sind.</p>

Name	Beschreibung
bOutputAutomatic	Wenn bOutputAutomatic TRUE ist, wird der Schaltausgang automatisch geschaltet.
iBlockSize	Die Blockgröße kann als 4 Byte oder 8 Byte eingestellt werden. Sie muss mit der zum Lesen und Schreiben in ST_RFID_AccessData [► 52] verwendeten Blockgröße überein stimmen.
tOutputPulseTime	Mit tOutputPulseTime wird die Aktionszeit des Ausganges konfiguriert. Die Impulsdauer des optionalen Ausgangssignales kann zwischen 30 ms und 9000 ms eingestellt werden.
eTranspType	Wenn das RFID-Gerät nur Transponder eines Typs erkennen soll, kann dieser mit eTranspType eingestellt werden. Mit dem Wert 16#FE wird keine Einschränkung vorgenommen. Folgende Werte sind möglich (E_RFID_TranspType [► 67]): eRFTT_ICode eRFTT_STmLRI512 eRFTT_Taglt eRFTT_ICodeSli eRFTT_InfineonSRF55 eRFTT_Inside eRFTT_TagltHfi
iCountBlocksRead	Mit iCountBlocksRead wird die Anzahl der automatisch zu lesenden Blöcke konfiguriert. Das Produkt mit iBlockSize ergibt die Anzahl an Bytes. Die maximale Anzahl an Blöcken beträgt je nach Blockgröße und anderen Einstellungen zwischen 4 und 9 Blöcke.
iCountBlocksWrite	Mit iCountBlocksWrite wird die Anzahl der automatisch zu schreibenden Blöcke konfiguriert. Das Produkt mit iBlockSize ergibt die Anzahl an Bytes. Die maximale Anzahl an Blöcken beträgt je nach Blockgröße und anderen Einstellungen zwischen 4 und 9 Blöcke.
iStartBlockRead	Mit iStartBlockRead wird die Startadresse für das automatische Lesen konfiguriert.
iStartBlockWrite	Mit iStartBlockWrite wird die Startadresse für das automatische Schreiben konfiguriert.
arrWriteData	Es können maximal 72 Bytes als Schreibdaten angegeben werden.

Es gibt Kombinationen von Werten, welche unzulässig sind. Die bestehenden Abhängigkeiten sind in der proprietären Spezifikation des Herstellers dargelegt. Wenn versucht wird, eine unzulässige Konfiguration zu schreiben, tritt der Fehler eRFERR_InvalidCfg ein oder es wird ein Fehlercode vom RFID-Gerät empfangen.

Weitere Informationen zum Ablauf der RFID-Reader-Konfiguration sind im Abschnitt [Konfiguration](#) [► 40] zusammengefasst.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.1.8.3 ST_RFID_CfgStruct_DeisterUDL

Die Struktur ist für das Schreiben mit eRFC_SetConfig sowie das Lesen mit eRFC_GetConfig geeignet (siehe [RFID-Befehlssatz](#) [► 21]).

Dabei handelt es sich nicht um die Parametrierung der TwinCAT-RFID-Bibliothek, sondern um die proprietäre Konfiguration des RFID Readers.

```

TYPE ST_RFID_CfgStruct_DeisterUDL :
STRUCT
  ePollingMode : E_RFID_PollingMode := eRFPO_PollingMode; (* CMD: 0x0A OR   Byte 32, Bit 5 *)
  eTriggerMode : E_RFID_TriggerMode := eRFTR_ImmediateRead; (* Byte 15, Bit 1 *)
  eOpMode      : E_RFID_OpMode := eRFOP_ReadSerialNumber; (* Byte 15, Bit 6,7 *)
  eTranspType  : E_RFID_TranspType := eRFTT_EPC1Gen2; (* Byte 33 *)

  tOutputPulseTime : TIME := T#300ms; (* Byte 38 and 39 *)
  bOutputLevel     : BOOL;           (* TRUE = high; FALSE = low *)

```

```
iReserved : USINT;  
  
iCountBlocksRead : USINT := 1;      (* Byte 41 *)  
iCountBlocksWrite : USINT := 1;     (* Byte 43 *)  
  
iStartBlockRead   : UINT := 0;      (* Byte 40 *)  
iStartBlockWrite  : UINT := 0;      (* Byte 42 *)  
arrWriteData      : ARRAY [0..31] OF BYTE; (* Byte 44 - 75 *)  
END_STRUCT  
END_TYPE
```

Es muss ggf. der Unterschied zwischen Polling und Trigger beachtet werden. Hinzu kommt in diesem Kontext, dass neben dem PollingMode dennoch der TriggerMode vorhanden sein kann.

Name	Beschreibung
ePollingMode	<p>Wenn eRFPO_PollingMode eingestellt ist, sendet das RFID-Gerät nur auf Anfrage Daten [default].</p> <p>Wenn eRFPO_ReportMode eingestellt ist, darf das RFID-Gerät jederzeit von sich aus Daten übertragen.</p> <pre data-bbox="411 338 746 432">TYPE E_RFID_PollingMode : (eRFPO_ReportMode, eRFPO_PollingMode);END_TYPE</pre>
eTriggerMode	<p>Wenn eRFTR_ImmediateRead eingestellt ist, ist das Gerät immer lesebereit. Durch diese Einstellung gilt die Triggerbedingung immer als erfüllt [default].</p> <p>Wenn eRFTR_ReadWithTrigger eingestellt ist, liest das Gerät nur bei Triggerbedingung. Dazu kann der Befehl eRFC_TriggerOn genutzt werden (siehe RFID-Befehlssatz [▶ 21]).</p> <p>Das darauffolgende Antworttelegramm wird vom Funktionsbaustein der TwinCAT-RFID-Bibliothek empfangen. Eine Zuweisung von gelesenen Transponderdaten ist in dem Fall nicht gegeben. Die empfangenen Rohdaten können zur weiteren Verarbeitung dem Bausteininterface entnommen werden.</p> <pre data-bbox="411 757 759 851">TYPE E_RFID_TriggerMode : (eRFTR_ImmediateRead, eRFTR_ReadWithTrigger);END_TYPE</pre>
eOpMode	<p>Diese Betriebsmodi sind nicht mit jedem Transpondertyp möglich.</p> <p>Wenn eRFOP_WriteData eingestellt ist, wird ein Schreibzugriff ausgeführt sobald ein Transponder erkannt wird.</p> <p>Wenn eRFOP_ReadData eingestellt ist, wird ein Lesezugriff ausgeführt sobald ein Transponder erkannt wird.</p> <p>Wenn eRFOP_ReadSerialNumber eingestellt ist, wird keine Aktion ausgeführt. Der Befehl Polling liefert die Seriennummer [default].</p> <p>Das darauffolgende Antworttelegramm wird vom Funktionsbaustein der TwinCAT-RFID-Bibliothek empfangen. Eine Zuweisung von gelesenen Transponderdaten ist in dem Fall nicht gegeben. Die empfangenen Rohdaten können zur weiteren Verarbeitung dem Bausteininterface entnommen werden.</p> <pre data-bbox="411 1256 746 1373">TYPE E_RFID_OpMode : (eRFOP_WriteData, eRFOP_ReadData, eRFOP_ReadSerialNumber);END_TYPE</pre>
eTranspType	<p>Soll das RFID-Gerät nur Transponder eines Typs erkennen, kann dieser mit eTranspType eingestellt werden. Mit dem Wert 16#FE wird keine Einschränkung vorgenommen.</p> <p>Folgende Werte sind möglich (E_RFID_TranspType [▶ 67]):</p> <p>eRFTT_EPC1Gen1 eRFTT_EPC1Gen2</p>
tOutputPulseTime	<p>Mit tOutputPulseTime wird die Aktionszeit des Ausganges konfiguriert. Die Impulsdauer des optionalen Ausgangssignals kann zwischen 30 ms und 9000 ms eingestellt werden.</p>
bOutputLevel	<p>Mit bOutputLevel wird die Kontrolle des optionalen digitalen Ausganges beeinflusst. Nach einem erfolgreichen Lesen kann der Ausgang auf HighLevel (bOutputLevel = TRUE) oder Lowlevel (bOutputLevel = FALSE) gesetzt werden.</p>
iCountBlocksRead	<p>Mit iCountBlocksRead wird die Anzahl der automatisch zu lesenden Blöcke konfiguriert. Das Produkt mit iBlockSize ergibt die Anzahl an Bytes. Die maximale Anzahl an Blöcken beträgt je nach Blockgröße und anderen Einstellungen zwischen 4 und 9 Blöcke. Die Blockgröße ist abhängig vom Transpondertyp.</p>
iCountBlocksWrite	<p>Mit iCountBlocksWrite wird die Anzahl der automatisch zu schreibenden Blöcke konfiguriert. Das Produkt mit iBlockSize ergibt die Anzahl an Bytes. Die maximale Anzahl an Blöcken beträgt je nach Blockgröße und anderen Einstellungen zwischen 4 und 9 Blöcke. Die Blockgröße ist abhängig vom Transpondertyp.</p>

Name	Beschreibung
iStartBlockRead	Mit iStartBlockRead wird die Startadresse für das automatische Lesen konfiguriert.
iStartBlockWrite	Mit iStartBlockWrite wird die Startadresse für das automatische Schreiben konfiguriert.
arrWriteData	Es können maximal 32 Bytes als Schreibdaten angegeben werden.

Weitere Informationen zum Ablauf der RFID-Reader-Konfiguration sind im Abschnitt [Konfiguration](#) [► 40] zusammengefasst.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.1.8.4 ST_RFID_CfgStruct_LeuzeRFM

Die Struktur ist für das Schreiben mit eRFC_SetConfig sowie das Lesen mit eRFC_GetConfig geeignet (siehe [RFID-Befehlssatz](#) [► 21]).

Dabei handelt es sich nicht um die Parametrierung der TwinCAT-RFID-Bibliothek, sondern um die proprietäre Konfiguration des RFID Readers.

```

TYPE ST_RFID_CfgStruct_LeuzeRFM :
STRUCT
  eOpMode      : E_RFID_OpMode := eRFOP_ReadData;
  eTriggerMode : E_RFID_TriggerMode := eRFTR_ImmediateRead;
  eReadMode    : E_RFID_ReadMode := eRFRD_SingleShot;
  eWriteMode   : E_RFID_WriteMode := eRFWR_ImmediateWrite;

  eNetworkMode : E_RFID_NetworkMode := eRFNM_Network;
  bAFI : BOOL := FALSE; (* not implemented; ready for future extention *)
  iAFI : BYTE;          (* not implemented; ready for future extention *)

  bSerialNumberMode : BOOL      := FALSE;
  bMultiTranspMode  : BOOL      := FALSE;
  bOutputAutomatic  : BOOL      := TRUE;
  iBlockSize        : USINT     := 8;

  tOutputPulseTime  : TIME      := T#300ms;

  eTranspType       : E_RFID_TranspType := eRFTT_TagItHfi;

  iCountBlocksRead  : USINT     := 1;
  iCountBlocksWrite : USINT     := 1;

  iStartBlockRead   : UINT      := 16#4000;
  iStartBlockWrite  : UINT      := 5;
  arrWriteData      : ARRAY [0..71] OF BYTE;
END_STRUCT
END_TYPE
    
```

Name	Beschreibung
eOpMode	<p>Die Betriebsart legt fest, welche Funktion durch einen Triggerimpuls ausgelöst wird. Der Befehl eRFC_TriggerOn oder ein Impuls am optionalen Triggereingang löst die hier eingestellte Aktion aus.</p> <p>Wenn eRFOP_WriteData eingestellt ist, wird ein Schreibzugriff ausgeführt. Wenn eRFOP_ReadData eingestellt ist, wird ein Lesezugriff ausgeführt [default].</p> <p>Das darauffolgende Antworttelegramm wird vom Funktionsbaustein der TwinCAT-RFID-Bibliothek empfangen. Eine Zuweisung von gelesenen Transponderdaten ist in dem Fall nicht gegeben. Die empfangenen Rohdaten können zur weiteren Verarbeitung dem Bausteininterface entnommen werden.</p> <pre data-bbox="432 517 767 629"> TYPE E_RFID_OpMode : (eRFOP_WriteData, eRFOP_ReadData, eRFOP_ReadSerialNumber);END_TYPE </pre>
eTriggerMode	<p>Wenn eRFTR_ImmediateRead eingestellt ist, ist das Gerät immer lesebereit. Durch diese Einstellung gilt die Triggerbedingung immer als erfüllt [default].</p> <p>Wenn eRFTR_ReadWithTrigger eingestellt ist, liest das Gerät nur bei Triggerbedingung. Dazu kann der Befehl eRFC_TriggerOn genutzt werden (siehe RFID-Befehlssatz ▶ 21).</p> <p>Das darauffolgende Antworttelegramm wird vom Funktionsbaustein der TwinCAT-RFID-Bibliothek empfangen. Eine Zuweisung von gelesenen Transponderdaten ist in dem Fall nicht gegeben. Die empfangenen Rohdaten können zur weiteren Verarbeitung dem Bausteininterface entnommen werden.</p> <pre data-bbox="432 958 767 1048"> TYPE E_RFID_TriggerMode : (eRFTR_ImmediateRead, eRFTR_ReadWithTrigger);END_TYPE </pre>
eReadMode	<p>Wenn eRFRD_ContinuousRead eingestellt ist, liest das Gerät dauerhaft und gibt ebenfalls dauerhaft gelesene Daten aus.</p> <p>Wenn eRFRD_SingleShot eingestellt ist, liest das Gerät genau einmal [default].</p> <pre data-bbox="432 1176 767 1265"> TYPE E_RFID_ReadMode : (eRFRD_ContinuousRead, eRFRD_SingleShot);END_TYPE </pre>
eWriteMode	<p>Wenn eRFWR_ImmediateWrite eingestellt ist, muss der Transponder im Feld sein, um einen Schreib- oder Lesebefehl korrekt auszuführen [default].</p> <p>Wenn eRFWR_WriteToNextTag eingestellt ist, werden die Daten eines Schreibbefehles in den nächst folgenden Transponder geschrieben. („Vorspannen“)</p> <pre data-bbox="432 1429 767 1518"> TYPE E_RFID_WriteMode : (eRFWR_ImmediateWrite, eRFWR_WriteToNextTag);END_TYPE </pre>
eNetworkMode	<p>Wenn eRFNM_Network eingestellt ist, können mehrere Geräte in einem RS485-Netzwerk eingebunden sein [default].</p> <p>Wenn eRFNM_StandAlone eingestellt ist, befindet sich das Gerät im stand-alone Betrieb.</p> <p>Der Betrieb von mehreren Geräten innerhalb eines RS485-Netzwerkes wird von der Bibliothek nicht unterstützt.</p> <pre data-bbox="432 1749 767 1839"> TYPE E_RFID_NetworkMode : (eRFNM_Network, eRFNM_StandAlone);END_TYPE </pre>
bSerialNumberMode	<p>Wenn bSerialNumberMode TRUE ist, wird die Seriennummer bei Schreib- und Lesebefehlen mit übertragen.</p> <p>Im Standardfall entspricht dies der zuletzt mit dem Befehl GetInventory detektierten Transponder Seriennummer. Andernfalls wird die Transponder Seriennummer durch Angabe in ST_RFID_Control ▶ 45 festgelegt.</p>

Name	Beschreibung
bMultiTranspMode	Wenn bMultiTranspMode TRUE ist, so ist Antikollision aktiv, wenn mehrere Transponder im Feld sind.
bOutputAutomatic	Wenn bOutputAutomatic TRUE ist, wird der Schaltausgang automatisch geschaltet.
iBlockSize	Die Blockgröße kann als 4 Byte oder 8 Byte eingestellt werden. Sie muss mit der zum Lesen und Schreiben in ST_RFID_AccessData [► 52] verwendeten Blockgröße übereinstimmen.
tOutputPulseTime	Mit tOutputPulseTime wird die Aktionszeit des Ausganges konfiguriert. Die Impulsdauer des optionalen Ausgangssignals kann zwischen 30 ms und 9000 ms eingestellt werden.
eTranspType	Wenn das RFID Gerät nur Transponder eines Typs erkennen, kann dieser mit eTranspType eingestellt werden. Mit dem Wert 16#FE wird keine Einschränkung vorgenommen. Folgende Werte sind möglich (E_RFID_TranspType [► 67]): eRFTT_ICode eRFTT_STmLRI512 eRFTT_TagIt eRFTT_ICodeSli eRFTT_InfineonSRF55 eRFTT_Inside eRFTT_TagItHfi
iCountBlocksRead	Mit iCountBlocksRead wird die Anzahl der automatisch zu lesenden Blöcke konfiguriert. Das Produkt mit iBlockSize ergibt die Anzahl an Bytes. Die maximale Anzahl an Blöcken beträgt je nach Blockgröße und anderen Einstellungen zwischen 4 und 9 Blöcke.
iCountBlocksWrite	Mit iCountBlocksWrite wird die Anzahl der automatisch zu schreibenden Blöcke konfiguriert. Das Produkt mit iBlockSize ergibt die Anzahl an Bytes. Die maximale Anzahl an Blöcken beträgt je nach Blockgröße und anderen Einstellungen zwischen 4 und 9 Blöcke.
iStartBlockRead	Mit iStartBlockRead wird die Startadresse für das automatische Lesen konfiguriert.
iStartBlockWrite	Mit iStartBlockWrite wird die Startadresse für das automatische Schreiben konfiguriert.
arrWriteData	Es können maximal 72 Bytes als Schreibdaten angegeben werden.

Es gibt Kombinationen von Werten, welche unzulässig sind. Die bestehenden Abhängigkeiten sind in der proprietären Spezifikation des Herstellers dargelegt. Wenn versucht wird, eine unzulässige Konfiguration zu schreiben, tritt der Fehler eRFERR_InvalidCfg ein oder es wird ein Fehlercode vom RFID Gerät empfangen.

Weitere Informationen zum Ablauf der RFID-Reader-Konfiguration sind im Abschnitt [Konfiguration \[► 40\]](#) zusammengefasst.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.1.8.5 ST_RFID_CfgStruct_PepperlFuchsIDENT

Die Struktur ist für das Lesen mit eRFC_GetConfig geeignet (siehe [RFID-Befehlssatz \[► 21\]](#)). Dabei handelt es sich nicht um die Parametrierung der TwinCAT-RFID-Bibliothek, sondern um die proprietäre Konfiguration des RFID Readers.

```

TYPE ST_RFID_CfgStruct_PepperlFuchsIDENT :
STRUCT
    tTimeout          :TIME;
    iBaudrate         :UINT;
    iIdentChannel     :USINT;
    bMultiplexMode    :BOOL;
    arrHeadCfg        :ARRAY [0..3] OF ST_RFID_HeadCfg;
    
```

```

arrTriggerCfg :ARRAY [0..1] OF ST_RFID_TriggerCfg;
END_STRUCT
END_TYPE

```

Das Pepper+Fuchs Gerät Ident Control Compact besteht aus einer Zentraleinheit und 1-4 Schreib-/Leseköpfen. Jedes dieser fünf Elemente erhält einen Identifikationskanal (Ident Channel), mit dem sich Befehle zu einzelnen Elementen zuordnen lassen. Im Standardfall ist die Zentraleinheit mit dem Kanal 0 versehen und die Schreib-/Leseköpfe mit den Kanälen 1-4.

Mit dem Befehl eRFC_GetConfig und den Ausgaben am Ausgang stReaderCfg können die Einstellungen für alle Identifikationskanäle überprüft werden.

Name	Beschreibung
tTimeout	tTimeout gibt die Dauer an, die das RFID-Gerät auf weitere Zeichen eines Telegramms wartet. Hat das Gerät nach dieser Dauer keinen verständlichen Befehl erkannt, folgt eine Fehlermeldung. (Standard ist 0 ms)
iBaudrate	Mit iBaudrate wird die aktuell verwendete Baudrate des RFID-Gerätes angezeigt. Die unterstützten RFID-Geräte verfügen über eine maximale Übertragungsrate von 38400 Baud.
identChannel	Identifikationskanal der Zentraleinheit
bMultiplexMode	Im Multiplex-Modus ist eine gegenseitige Beeinflussung der Schreib-/Leseköpfe minimiert, weil immer nur ein Kopf gleichzeitig aktiv ist.
arrHeadCfg	<p>Es existieren Geräte mit bis zu vier Schreib-/Leseköpfen. Jeder Kopf hat einen Status und einen DataCarrierType. Diese Information ist je Kopf in einer Struktur vom Typ ST_RFID_HeadCfg hinterlegt.</p> <p>Der Status der Zentraleinheit wird in iErrCodeRcv direkt am Ausgang von FB RFIDReader [▶ 35] ausgegeben.</p> <p>Mögliche Werte für iDCType sind in ST_RFID_Control [▶ 45] erläutert.</p> <pre> TYPE ST_RFID_HeadCfg : STRUCT eStatus :E_RFID_ErrCodeRcv_PepperlFuchs; iDCType :USINT; (* not equal to E_RFID_TranspType enumeration *) iReserved :USINT; END_STRUCT END_TYPE </pre>
arrTriggerCfg	<p>Es existieren Geräte mit bis zu vier Schreib-/Leseköpfen. Jeder Kopf besitzt einen identChannel und bTriggerMode. Diese Information ist je Triggersensor in einer Struktur vom Typ ST_RFID_TriggerCfg hinterlegt. identChannel bezeichnet den Schreib-/Lesekopf für den der Triggersensor konfiguriert ist.</p> <p>Wenn iTriggerMode TRUE ist, ist der Trigger Modus ist für einen Triggersensor aktiv.</p> <p>Wenn zudem bInverted TRUE ist, so handelt es sich um ein invertiertes Triggersignal.</p> <p>Weitere Informationen zum Thema Trigger Mode finden Sie im Abschnitt Pepperl+Fuchs [▶ 32]</p> <pre> TYPE ST_RFID_TriggerCfg : STRUCT iIdentChannel :USINT; bTriggerMode :BOOL; bInverted :BOOL; bReserved :BOOL; END_STRUCT END_TYPE </pre>

Weitere Informationen zum Ablauf der RFID-Reader-Konfiguration sind im Abschnitt [Konfiguration \[▶ 40\]](#) zusammengefasst.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.2 Enumerationen

6.2.2.1 E_RFID_Command

```

TYPE E_RFID_Command : (
  eRFC_Unknown      := 0,
  eRFC_GetReaderVersion,
  eRFC_GetConfig,
  eRFC_SetConfig,
  eRFC_GetInventory,
  eRFC_Polling,
  eRFC_TriggerOn,
  eRFC_TriggerOff,
  eRFC_AbortCommand,
  eRFC_ResetReader,
  eRFC_ReadBlock,
  eRFC_WriteBlock,
  eRFC_OutputOn,
  eRFC_OutputOff,
  eRFC_FieldOn,
  eRFC_FieldOff,
  eRFC_SendRawData,
  eRFC_ChangeDCType,
END_TYPE

```

Der Funktionsbaustein [FB_RFIDReader](#) [► 35] der TwinCAT-SPS-RFID-Bibliothek bietet am Eingang `eCommand` die im Codebereich dargestellten Enumerationswerte an. Dabei handelt es sich um eine Auswahl an Befehlen, wie z. B. das Lesen oder Schreiben eines Transponders. Ausführliche Erläuterungen zu den Befehlen finden Sie im Abschnitt [RFID-Befehlssatz](#) [► 21].

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.2.2 E_RFID_Response

```

TYPE E_RFID_Response : (
  eRFR_NoRsp,
  eRFR_Unknown,
  eRFR_Ready,
  eRFR_CmdConfirmation,
  eRFR_CfgChangeExecuted,
  eRFR_WriteCmdSucceeded,
  eRFR_NoTransponder,
  eRFR_Error,
  eRFR_Data_ReaderVersion,
  eRFR_Data_Config,
  eRFR_Data_Inventory,
  eRFR_Data_ReadData,
);
END_TYPE

```

Der Funktionsbaustein [FB_RFIDReader](#) [► 35] der TwinCAT-SPS-RFID-Bibliothek bietet am Ausgang `eResponse` die im Codebereich dargestellten Enumerationswerte an. Diese sind teilweise analog zu den Telegrammantworttypen der herstellerproprietären Protokolle. Welche herstellerproprietäre MessageID der hier gelisteten Response entspricht, wird in der folgenden Beschreibung jeweils *kursiv* angegeben. Aufgrund der Auswertungskomplexität sind nicht alle Entsprechungen aufgeführt. Detaillierten Aufschluss kann bei Bedarf die Rohdatendarstellung [ST_RFID_RawData](#) [► 45] am Ausgang des Funktionsbausteins geben.

Wert	Beschreibung
eRFR_NoRsp	Dieser Wert signalisiert, dass zuletzt keine Antwort eingetroffen ist.
eRFR_Unknown	Dieser Wert gibt an, dass das angekommene Telegramm nicht einem bestimmten Typ zugeordnet werden konnte und somit auch nicht ausgewertet wurde. Meist geht dies einher mit einer Fehlermeldung (bError = TRUE).
eRFR_Ready	Manche RFID-Reader-Modelle signalisieren ihre Betriebsbereitschaft, z. B. nach einem Reset, mit einem extra Telegramm. In diesem Fall nimmt eResponse den Wert eRFR_Ready an. <i>Entsprechung im proprietären Protokoll:</i> <i>Leuze: 'S'</i> <i>Pepperl+Fuchs: Status='2'</i>
eRFR_CmdConfirmation	Mit dieser Antwort wird der gesendete Befehl bestätigt. Dies kann bei vielen Befehlen auftreten. Ausnahmen sind die Befehle „Konfiguration ändern“ und „Daten schreiben“, denn auf diese beiden Befehle folgen spezielle Bestätigungen, welche durch die folgenden zwei Enumerationswerte repräsentiert werden. <i>Entsprechung im proprietären Protokoll:</i> <i>Leuze: 'Q2', 'Q4'</i>
eRFR_CfgChangeExecuted	Wenn die RFID-Reader-Konfiguration geändert wurde, sendet der RFID Reader dieses Telegramm zur Bestätigung der Aktion. <i>Entsprechung im proprietären Protokoll:</i> <i>Leuze: 'Q1'</i>
eRFR_WriteCmdSucceeded	Sobald Daten auf den Transponder geschrieben wurden, sendet der RFID Reader diese Bestätigung. <i>Entsprechung im proprietären Protokoll:</i> <i>Leuze: 'Q5'</i>
eRFR_NoTransponder	Diese Antwort wird gegeben, falls sich kein Transponder im Lesefeld befindet. Dies wird nicht zwangsläufig als Fehler gewertet, so dass der Ausgang bError auch nicht gesetzt wird. <i>Entsprechung im proprietären Protokoll:</i> <i>Leuze: '\$18'</i> <i>Pepperl+Fuchs: Status='5'</i>
eRFR_Error	Wenn ein Telegramm empfangen wurde, welches einen Fehlercode übermittelt hat, so wird eRFR_Error am Ausgang eResponse ausgegeben. Der übermittelte Fehlercode wird in der Ausgangsvariablen iErrCodeRcv angegeben (siehe RFID-Fehlercodes [► 77]). Wenn eResponse den Wert eRFR_Error annimmt, so wird auch ein Fehler mittels bError = TRUE am Ausgang des Funktionsbausteins signalisiert. <i>Entsprechung im proprietären Protokoll:</i> <i>Balluff: <NAK>+Failurenumber</i> <i>Deister: MessageErrorCode=>=16#20</i> <i>Leuze: 'Exx', 'Q0'</i>
eRFR_Data_ReaderVersion	Durch eine Versionsabfrage wird ein RFID Reader dazu aufgefordert, Modellinformationen zu senden. Diese Art von empfangenen Daten wird mit dem Wert eRFR_Data_ReaderVersion am Ausgang eResponse bezeichnet.
eRFR_Data_Config	Eine ausgelesene RFID-Reader-Konfiguration wird mittels dem Enumerationswert eRFR_Data_Config signalisiert. <i>Entsprechung im proprietären Protokoll:</i> <i>Leuze: 'G'</i>
eRFR_Data_Inventory	Dieser Telegrammtyp wird angezeigt, wenn ein Transponder erkannt wurde bzw. die Seriennummer eines Transponders ausgelesen wurde.

Wert	Beschreibung
eRFR_Data_ReadData	Ein Empfang von ausgelesenen Daten aus dem Transponderspeicher wird mit dem Wert eRFR_Data_ReadData signalisiert. <i>Entsprechung im proprietären Protokoll: Deister: MessageID=16#40 or 16#41</i>

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.2.3 E_RFID_ReaderGroup

```
TYPE E_RFID_ReaderGroup : (
    eRFRG_Unknown,
    eRFRG_BalluffBIS,
    eRFRG_DeisterBasic,
    eRFRG_DeisterRDL,
    eRFRG_DeisterUDL,
    eRFRG_DeisterVReader,
    eRFRG_LeuzeRFM,
    eRFRG_PepperlFuchsIDENT,
    eRFRG_BaltechIDE
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.2.4 E_RFID_ReaderManufacturer

```
TYPE E_RFID_ReaderManufacturer : (
    eRFRM_Unknown,
    eRFRM_Balluff,
    eRFRM_Deister,
    eRFRM_Leuze,
    eRFRM_PepperlFuchs,
    eRFRM_Baltech
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.2.5 E_RFID_TranspType

```
TYPE E_RFID_TranspType : (
    eRFTT_NoTag,
    eRFTT_TypeUnknown,
    eRFTT_ATA5590,
    eRFTT_ATA5590UID,
    eRFTT_EM4022_4222,
    eRFTT_EM4135,
    eRFTT_EPC1Gen1,
    eRFTT_EPC1Gen2,
    eRFTT_FujitsuMB89R118,
    eRFTT_ICode,
    eRFTT_ICodeSli,
    eRFTT_InfineonSLE55,
    eRFTT_InfineonSRF55, (* also known as Infineon my-d vicinity *)
    eRFTT_Inside,
    eRFTT_ISO180006TypB,
    eRFTT_LegicPrime,
    eRFTT_LegicAdvant,
```

```

eRFTT_MifareClassic,      (* Philips *)
eRFTT_MifareUltraLight,
eRFTT_MifareDESFire,
eRFTT_STmLRI512,
eRFTT_TagIt,
eRFTT_TagItHfi,          (* TI *)
eRFTT_UCodeEPC119,       (* Philips *)
eRFTT_PICOPASS,          (* INSIDE Contactless *)
);
END_TYPE

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.2.3 T_RFID_TranspSRN

```

(* serial number shown as hex coded string(ascii) *)
TYPE T_RFID_TranspSRN : STRING(iRFID_MAXSRNLENGTH);
END_TYPE

```

Der Datentyp beinhaltet eine RFID-Transponder-Seriennummer. Die Seriennummer des Transponders (häufig 8 Byte) wird als String in hexadezimaler Darstellung angegeben. (iRFID_MAXSRNLENGTH = 51)

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

6.3 Globale Konstanten

6.3.1 Global_Version

Alle Bibliotheken haben eine bestimmte Version. Diese Version ist u. a. im SPS-Bibliotheks-Repository zu sehen. Eine globale Konstante vom Typ ST_LibVersion enthält die Information über die Bibliotheksversion:

```

VAR_GLOBAL CONSTANT
  stLibVersion_Tc2_RFID : ST_LibVersion;
END_VAR

```

Um zu sehen, ob die Version, die Sie haben, auch die Version ist, die Sie brauchen, benutzen Sie die Funktion F_CmpLibVersion (definiert in Tc2_System-Bibliothek).

Alle anderen Möglichkeiten Bibliotheksversionen zu vergleichen, die Sie von TwinCAT 2 kennen, sind veraltet.

7 Beispiele

Die folgenden Beispiele wurden mit unterschiedlichen RFID-Reader-Modellen entwickelt.

Weil grundsätzlich kaum Unterschiede in der Handhabung der RFID Reader mit der TwinCAT-RFID-Bibliothek bestehen, kann auch ein Beispiel, das mit einem anderen Modell entwickelt wurde, zur Einarbeitung herangezogen werden.

Tutorial

Das Tutorial beschreibt, wie ein RFID Reader in Betrieb genommen wird. Dabei wird Schritt für Schritt von der Einbindung der TwinCAT-Bibliothek bis hin zur Präsenzerkennung von RFID Transpondern vorgegangen. (Siehe [Tutorial](#) [|](#) [69](#))

Beispiel 1

Dieses Beispiel kann für unterschiedliche RFID Reader genutzt werden (Balluff, Baltech, Deister, Leuze, Pepper+Fuchs).

Getestet ist das Beispiel mit den Modellen Balluff BIS M 401 und Leuze electronic RFM32. Im Projekt wurde ein RFID Reader an eine einkanalige serielle EL6001 (an einem EK1100) angeschlossen. Es lassen sich ebenso andere serielle Klemmen nutzen.

Das Projekt kann ebenso für zwei RFID Reader genutzt werden. Das Beispielprogramm ist bereits für zwei RFID Reader vorbereitet. Es muss lediglich die zweite Verlinkung im TwinCAT System Manager erfolgen. (Siehe [Beispiel 1](#) [|](#) [73](#))

Beispiel 2

Dieses Beispiel ist mit einem Baltech RFID Reader, welcher in den Beckhoff Control-Panels sowie Panel-PCs optional verbaut ist, entwickelt worden. Das Gerät wird an einen seriellen Com Port oder einen USB Port angeschlossen.

Es kann genutzt werden, um das Gerät komfortabel in Betrieb zu nehmen und zu testen. Das Beispiel verfügt über eine einfache Visualisierung. (Siehe [Beispiel 2](#) [|](#) [74](#))

Beispiel 3

Dieses Beispiel entspricht einer kleinen Applikation. Die Anwendung umfasst das Erkennen, Lesen und Schreiben eines Transponders in einem automatischen Ablauf.

Erstellt wurde das Beispiel mit einem Pepperl+Fuchs RFID Reader. Es kann sowohl das 2-kanalige als auch das 4-kanalige Modell genutzt werden. (Siehe [Beispiel 3](#) [|](#) [75](#))

7.1 Tutorial

Dieses Tutorial zeigt, wie Sie mithilfe der TwinCAT SPS einen RFID Reader in Betrieb nehmen können. Das Tutorial wird mit dem RFID-Reader-Modell Balluff BIS M 401 durchgeführt. Für andere Modelle kann die Vorgehensweise allerdings grundsätzlich übernommen werden.

Folgende Schritte werden dabei durchgeführt:

1. Glossar
2. Installation/Bibliotheken
3. Serielle Anbindung
4. Baustein Deklaration
5. Baustein Verwendung
6. Test

Annahmen:

- Sie haben Ihre RFID-Anwendung detailliert geplant und Ihre Applikation bereits modelliert.

- Sie haben festgestellt, dass ein von der Tc2_RFID-Bibliothek unterstützter Reader grundsätzlich Ihren Anforderungen entspricht und Sie mit den angebotenen Kommandos Ihre Applikation implementieren können.
- Sie entscheiden sich, die TwinCAT-RFID-Bibliothek zu nutzen, um sich die Kommunikation zu Ihrem RFID Reader zu erleichtern.

Download: https://infosys.beckhoff.com/content/1031/tf6600_tc3_rfid/Resources/234213899.zip

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

7.1.1 Glossar

Begriff	Erläuterung
CT	Carrier Type, Datenträgertyp, Transpondertyp
DC	Data Carrier - RFID Transponder (Datenspeicher)
HF	High Frequency
Label	RFID Transponder
LF	Low Frequency
RF	Radio Frequency
RFID	Radio Frequency Identification
RFID Reader	Ein RFID Lesegerät ('Reader') kann sowohl ein rein lesefähiges Gerät sein als auch ein Schreib-und Lesegerät.
Smart Label	RFID Transponder
SRN	Serial Number
Tag	RFID Transponder
UHF	Ultra High Frequency

Siehe: [Installation/Bibliotheken \[► 70\]](#)

7.1.2 Installation/Bibliotheken

1. Starten Sie das TwinCAT XAE.
 2. Legen eine neues TwinCAT-Projekt an (Menü **Datei** > Befehl **Neu**).
 3. Legen Sie ein neues SPS-Projekt an (Kontextmenü des SPS-Objekts im TwinCAT-Projektbaum > Befehl **Neues Element hinzufügen**).
 4. Wählen Sie Ihre Zielplattform PC und CX (x86, x64).
 5. Fügen Sie dem SPS-Projekt die Bibliothek Tc2_RFID und Tc2_SerialCom hinzu (Kontextmenü des Reference-Objekts > Befehl **Bibliothek hinzufügen**)
- ⇒ Alle benötigten SPS-Bausteine für die RFID-Reader-Kommunikation stehen Ihnen zur Verfügung.

Siehe: [Serielle Anbindung \[► 70\]](#)

7.1.3 Serielle Anbindung

In diesem Beispiel wird der RFID Reader über die serielle EtherCAT-Klemme EL 6001 angebunden.

1. Legen Sie global einen Sendepuffer sowie einen Empfangspuffer (gEL6ComTxBuffer, gEL6ComRxBuffer) vom Type „ComBuffer“ an. Legen Sie außerdem zwei Datenstrukturen an, wie sie im TwinCAT System Manager zur seriellen Kommunikation verwendet werden:

```
gEL6ComRxBuffer      :ComBuffer;
gEL6ComTxBuffer      :ComBuffer;
EL6ComInData         AT %I* : EL6ComInData;
EL6ComOutData        AT %Q* : EL6ComOutData;
```

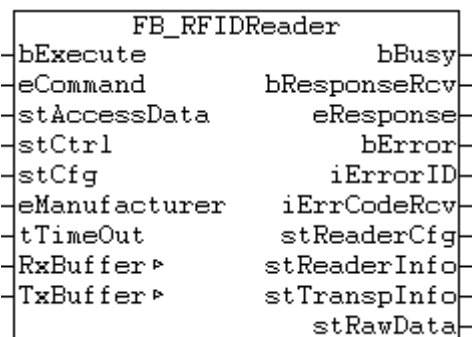
2. Verlinken Sie die Strukturen im TwinCAT System Manager mit den Kanälen des seriellen Ports.
3. Legen Sie zur seriellen Kommunikation eine Instanz des SerialLineControl an. Rufen Sie diese in einer schnellen Task zyklisch auf. (Mode: Geben Sie als Handle die serielle EtherCAT-Klemme mit 22 Byte Nutzdaten an.)

```
LineControl(
  Mode      := SERIALLINEMODE_EL6_22B,
  pComIn    := ADR(EL6ComInData),
  pComOut   := ADR(EL6ComOutData),
  SizeComIn := SIZEOF(EL6ComInData),
  TxBuffer  := gEL6ComTxBuffer,
  RxBuffer  := gEL6ComRxBuffer
);
```

Siehe: [RFID-Reader-Anbindung \[► 19\]](#) und [Bausteindeklaration \[► 71\]](#)

7.1.4 Bausteindeklaration

Der Funktionsbaustein FB_RFIDReader ist das Herzstück der ganzen RFID-Reader-Kommunikation. Nachfolgend wird die Deklaration und Initialisierung des Bausteins beschrieben.



1. Legen Sie eine Instanz des Bausteins FB_RFIDReader an.
2. Übergeben Sie der Instanz den Hersteller Ihres RFID-Modells am Eingang eManufacturer.

```
fbRFIDReader : FB_RFIDReader := (eManufacturer := eRFRM_Balluff);
sTranspSerialNumber : STRING;
```

FB_RFIDReader verfügt über 7 Eingänge (6 bei den spezifischen FBs aufgrund der fehlenden Herstellerangabe), 2 Ein-Ausgänge und 10 Ausgänge. Zum Empfangen von Nachrichten, die seitens des RFID Readers zur Steuerung gesendet werden, ist es ausreichend den Funktionsbaustein zyklisch aufzurufen. Der Eingang bExecute muss dabei FALSE bleiben. Dies wird in diesem Beispiel genutzt, um vorerst eine einfache Präsenzerkennung zu implementieren.

3. Rufen Sie den Baustein wie folgt auf:

```
fbRFIDReader(
  bExecute      := FALSE,

  RxBuffer      := RxBuffer,
  TxBuffer      := TxBuffer,

  bBusy         => ,
  bError        => ,
  iErrorID      => ,
  iErrCodeRcv   =>
);
sTranspSerialNumber := fbRFIDReader.stTranspinfo.sSerialNumber;
```

Nun wird die zuletzt gelesene Seriennummer eines RFID Transponders in Ihrer String-Variablen dargestellt. Zur Fehleranalyse sollten ebenfalls die Ausgänge bError und iErrorID etc. ausgewertet werden.

Siehe: [Bausteinverwendung \[► 71\]](#)

7.1.5 Bausteinverwendung

Eine effektivere Auswertung der empfangenen Daten können Sie mit folgenden Anweisungen erreichen:

Deklarationen:

```

fbRFIDReader      : FB_RFIDReader := (eManufacturer := eFRM_Balluff);
sTranspSerialNumber : STRING;

bBusy      : BOOL;
bError     : BOOL;
iErrorID   : UINT;
iErrCodeRcv : UINT;

stTranspInfo : ST_RFID_TranspInfo;

eErrorID     : E_RFID_ErrID;
eErrCodeRcv : E_RFID_ErrCodeRcv_Balluff;

fbTriggerResponse : R_TRIG;
arrRspRcv         : ARRAY[0..99] OF BYTE;

```

Programmablauf:

```

fbRFIDReader(
    bExecute      := FALSE,

    RxBuffer      := RxBuffer,
    TxBuffer      := TxBuffer,

    bBusy         => bBusy,
    bError        => bError,
    iErrorID      => iErrorID,
    iErrCodeRcv  => iErrCodeRcv
);
(* convert Error Codes *)
eErrorID := UINT_TO_INT(iErrorID);
eErrCodeRcv := UINT_TO_INT(iErrCodeRcv);

fbTriggerResponse(CLK := fbRFIDReader.bResponseRcv);
IF (fbTriggerResponse.Q) THEN
    stTranspInfo := fbRFIDReader.stTranspInfo;
    sTranspSerialNumber := stTranspInfo.sSerialNumber;      (* detected RFID Tag Serial Number *)

    MEMSET(ADR(arrRspRcv), 0, SIZEOF(arrRspRcv));
    MEMCPY(ADR(arrRspRcv), fbRFIDReader.stRawData.pReceivedRsp, MIN(fbRFIDReader.stRawData.iReceivedRspLen, SIZEOF(arrRspRcv)));
END_IF

```

Empfangene Fehlercodes können online als Enumerationswert dargestellt werden, indem die Integer Variablen iErrorID und iErrCodeRcv direkt zugewiesen werden.

Mithilfe eines Triggers werden weitere Daten nur ausgewertet, wenn eine neue Nachricht empfangen wird.

Die String-Variablen sTranspSerialNumber gibt nun immer die Seriennummer des zuletzt detektierten Transponders wieder. Diese ist in diesem Fall ebenso am Funktionsbaustein fbRFIDReader.stTranspInfo.sSerialNumber zu sehen.

Weitere Informationen können je nach Anwendung von den Ausgängen des Funktionsbausteins übernommen werden.

Um eine empfangene Nachricht als komplette Bytefolge anzuzeigen, nutzen Sie beispielsweise die MEMCPY-Funktion und kopieren die Rohdaten in ihr deklariertes Bytearray.

Jede Meldung Ihres RFID Readers wird nun empfangen und in obiger Weise ausgewertet.

Siehe: [Test \[▶ 72\]](#)

7.1.6 Test

Sobald Sie das Programm gemäß der beschriebenen Vorgehensweise erstellt haben, aktivieren Sie die aktuelle Konfiguration mit den verlinkten Variablen im TwinCAT System Manager und setzen Sie TwinCAT in den Run-Modus setzen. Loggen Sie sich auf der Steuerung ein und starten Sie die Applikation.

Wenn Sie einen Transponder vor Ihren RFID Reader bewegen, wird dieser detektiert und die empfangene Nachricht sowie Seriennummer im Programmcode übernommen. Die Werte werden online im Programmcode oder in einer zusätzlichen Visualisierung dargestellt.

i Gerätekonfiguration

Der Balluff RFID Reader muss für diese Funktionalität passend konfiguriert sein. Die Option **Typ und serial number bei CT pres.** muss aktiviert sein. Diese und andere Konfigurationsparameter werden im Abschnitt [Balluff > RFID-Reader-Einstellungen \[► 26\]](#) näher beschreiben. Nicht jeder RFID Reader unterstützt diese Einstellung.

Download: https://infosys.beckhoff.com/content/1031/tf6600_tc3_rfid/Resources/234213899.zip

7.2 Beispiel 1

Dieses Beispiel kann für unterschiedliche RFID Reader genutzt werden (Balluff, Baltech, Deister, Leuze, Pepper+Fuchs).

Getestet ist das Beispiel mit den Modellen Balluff BIS M 401 und Leuze electronic RFM32.

Im Projekt wurde ein RFID Reader an eine einkanalige serielle EL6001 (an einem EK1100) angeschlossen. Es lassen sich ebenso andere serielle Klemmen nutzen. Bei Verwendung von KL-Klemmen muss der Aufruf des Serial Line Control im Programmcode angepasst werden. (Siehe [RFID-Reader-Anbindung \[► 19\]](#))

Das Projekt kann ebenso für zwei RFID Reader genutzt werden. Das Beispielprogramm ist bereits für zwei RFID Reader vorbereitet. Es muss lediglich die zweite Verlinkung im TwinCAT System Manager erfolgen.

Das Beispielprojekt beinhaltet den Aufruf des RFID-Funktionsbausteins mit unterschiedlichen Befehlen. Die wichtigsten Befehle wurden in diesem Beispiel implementiert. Dazu gehört unter anderem das Lesen und Schreiben vom RFID-Transponder-Speicher.

Nach Programmstart muss über die lokale Enumeration eManufacturer der passende RFID Reader Hersteller ausgewählt werden.

Mittels der lokalen Enumeration eCommand kann der Befehlstyp ausgewählt werden. Um den Aufruf zu starten, muss die lokale Variable bExecute einmal auf TRUE gesetzt werden. Daraufhin sind an den Ausgängen des RFID-Funktionsbausteins die Ergebnisse der Abfrage angegeben. Alternativ kann die Bedienung mit der im Beispiel integrierten Visualisierung geschehen:

fbRFID1

Manufacturer: eRFRM_Balluff

<- chosen command: eRFC_ReadBlock ->

Execute to send the command

access data:

start block: 4

block count: 1

block size: 8

data size: 8

data to write:

tag serial number (optional):

last response:

response type: eRFR_Data_Inventory

error id: eRFERR_NoError

read data: vutszyxw

last detected tag:

tag serial number: E00780ACDDEB563D

tag type: eRFTT_TaglHfi

Je nach RFID-Reader-Modell müssen zuerst die Befehle [GetReaderVersion](#) [► 22] und [GetConfig](#) [► 22] (ggf. auch [SetConfig](#) [► 23]) ausgeführt werden, um eine korrekte Kommunikation mit dem RFID Reader zu ermöglichen.

Für weitere Informationen zum Ablauf der RFID-Reader-Kommunikation wird auf den Abschnitt Funktionsbaustein FB_RFIDReader > [Handhabung](#) [► 39] verwiesen.

Download: https://infosys.beckhoff.com/content/1031/tf6600_tc3_rfid/Resources/5252190859.zip

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

7.3 Beispiel 2

Dieses Beispiel ist mit einem Baltech RFID Reader, welcher in den Beckhoff Control Panels sowie Panel-PCs optional verbaut ist, entwickelt worden. Das Gerät wird an einen seriellen Com Port oder den USB Port angeschlossen. Es kann genutzt werden, um das Gerät komfortabel in Betrieb zu nehmen und zu testen.

Wird ein Baltech RFID Reader verwendet, der an eine serielle Beckhoff Klemme anstatt an den Com Port angeschlossen ist, muss die serielle Hintergrundkommunikation im SPS-Code geändert werden und diese im TwinCAT System Manager neu konfiguriert werden. (Siehe [RFID-Reader-Anbindung](#) [► 19])

Das Beispielprojekt beinhaltet den Aufruf des RFID-Funktionsbausteins mit unterschiedlichen Befehlen. Mithilfe der integrierten Visualisierung können die Befehle ausgeführt werden. Implementiert sind die Befehle [GetReaderVersion](#) [► 22], [GetInventory](#) [► 23], [ReadBlock](#) [► 24] und [WriteBlock](#) [► 24]. So kann auch ein RFID Transponder getestet werden und Daten können in Form eines ASCII-Strings auf diesen geschrieben sowie gelesen werden.

Rfid Demo Application for serial connected Rfid Reader in Beckhoff Panels

Inputs / Commands	Outputs					
Init Rfid Reader	Reader Info: <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 80%;">Reader Type: eRFRT_BaltechIDE_LP</td> <td style="width: 20%;">1034</td> </tr> <tr> <td colspan="2">Reader's SW version : 1.08.01</td> </tr> </table>	Reader Type: eRFRT_BaltechIDE_LP	1034	Reader's SW version : 1.08.01		
Reader Type: eRFRT_BaltechIDE_LP	1034					
Reader's SW version : 1.08.01						
Detect Transponder	Transponder Info: <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 100%;">Transp.Type: eRFTT_LegicPrime</td> </tr> <tr> <td>Transp.SerialNbr: 57F4E98B</td> </tr> </table>	Transp.Type: eRFTT_LegicPrime	Transp.SerialNbr: 57F4E98B			
Transp.Type: eRFTT_LegicPrime						
Transp.SerialNbr: 57F4E98B						
Select Data To Read: <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Byteldx: 10</td> <td style="width: 50%;">ByteCount: 13</td> </tr> </table>	Byteldx: 10	ByteCount: 13	Response Type: <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 80%;">eResponse: eRFR_Data_ReadData</td> <td style="width: 20%;">Busy</td> </tr> </table>	eResponse: eRFR_Data_ReadData	Busy	
Byteldx: 10	ByteCount: 13					
eResponse: eRFR_Data_ReadData	Busy					
Read Data	Read Data: <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 100%;">ReadData(hex): 48 61 6C 6C 6F 20 57 65 6C 74 21 00 00</td> </tr> <tr> <td>ReadData(ASCII): Hallo Welt!</td> </tr> </table>	ReadData(hex): 48 61 6C 6C 6F 20 57 65 6C 74 21 00 00	ReadData(ASCII): Hallo Welt!			
ReadData(hex): 48 61 6C 6C 6F 20 57 65 6C 74 21 00 00						
ReadData(ASCII): Hallo Welt!						
Select Data To Write: <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Byteldx: 10</td> <td style="width: 50%;">WriteData(ASCII): Hallo Welt!</td> </tr> </table>	Byteldx: 10	WriteData(ASCII): Hallo Welt!				
Byteldx: 10	WriteData(ASCII): Hallo Welt!					
Write Data						
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Byteldx: 10</td> <td style="width: 50%;">ByteCount: 12</td> </tr> </table>	Byteldx: 10	ByteCount: 12				
Byteldx: 10	ByteCount: 12					
Erase Data (Write 0x00 00 00 ...)	Error Detection: <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 100%;">RfidError: FALSE</td> </tr> <tr> <td>ErrorID: eRFERR_NoError</td> <td style="width: 10%;">0</td> </tr> <tr> <td>ErrCodeRcv: eRFERRBaltech_NoError</td> <td>0</td> </tr> </table>	RfidError: FALSE	ErrorID: eRFERR_NoError	0	ErrCodeRcv: eRFERRBaltech_NoError	0
RfidError: FALSE						
ErrorID: eRFERR_NoError	0					
ErrCodeRcv: eRFERRBaltech_NoError	0					
Activate LogView Output	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 100%;">ComError: FALSE</td> </tr> <tr> <td>ComErrorID: COMERROR_NOERROR</td> </tr> <tr> <td>LastDetectedComErrorID: COMERROR_NOERROR</td> </tr> </table>	ComError: FALSE	ComErrorID: COMERROR_NOERROR	LastDetectedComErrorID: COMERROR_NOERROR		
ComError: FALSE						
ComErrorID: COMERROR_NOERROR						
LastDetectedComErrorID: COMERROR_NOERROR						

Der RFID Reader muss zuerst initialisiert werden, um eine korrekte Kommunikation mit dem RFID Reader zu ermöglichen. Die Schaltfläche **Init Rfid Reader** führt dazu den Befehl GetReaderVersion aus.

Für weitere Informationen zum Ablauf der RFID-Reader-Kommunikation wird auf den Abschnitt Funktionsbaustein FB_RFIDReader > [Handhabung \[▶ 39\]](#) verwiesen.

Durch Aktivierung der LogView-Ausgabe wird im TwinCAT System Manager LoggerView die komplette serielle Übertragung dargestellt.

Download: https://infosys.beckhoff.com/content/1031/tf6600_tc3_rfid/Resources/227370251.zip oder https://infosys.beckhoff.com/content/1031/tf6600_tc3_rfid/Resources/949454859.zip

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID v3.3.6.0

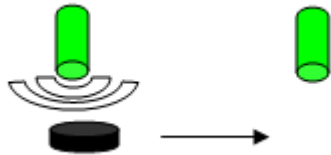
7.4 Beispiel 3

Dieses Beispiel entspricht einer kleinen Applikation. Die Anwendung umfasst das Erkennen, Lesen und Schreiben eines Transponders in einem automatischen Ablauf.

Erstellt wurde das Beispiel mit einem Pepperl+Fuchs RFID Reader. Es kann sowohl das 2-kanalige als auch das 4-kanalige Modell genutzt werden.

Im Beispiel ist das Gerät direkt an den Com Port angeschlossen. Wird ein Pepperl+Fuchs RFID Reader verwendet, der an eine serielle Beckhoff Klemme anstatt an den Com Port angeschlossen ist, muss die serielle Hintergrundkommunikation im SPS Code geändert werden und dies im TwinCAT System Manager neu konfiguriert werden. (Siehe [RFID-Reader-Anbindung](#) [► 19])

Ablauf der implementierten Applikation:



Mit dem RFID-Gerät sind zwei Schreib-/Leseköpfe verbunden. Beide erkennen vollautomatisch im Feld eintreffende Transponder. Nach Erkennung wird ein Speicherblock aus dem Datenspeicher des Transponders ausgelesen. Der sich darin befindliche 4-byte Wert wird vom ersten Lesekopf um eins addiert bzw. vom zweiten Lesekopf um eins subtrahiert. Der neue Wert wird sofort zurück auf den Transponder geschrieben. Dieser Ablauf des Erkennens, Lesens und Schreibens dauert in Summe ca. eine halbe Sekunde. Zwischen zwei solchen Vorgängen am selben Lesekopf müssen mindestens 3 Sekunden liegen, um eine ungewollte Mehrfachausführung zu vermeiden. (Dies kann ebenso mittels Prüfung der Tag-Seriennummer gelöst werden.)

Das Programm beinhaltet im Wesentlichen eine Zustandsmaschine mit 6 Zuständen:

- 0: Initialisierung - Ausführung von GetReaderVersion, GetConfig, etc.
- 1: Tag-Erkennung an Lesekopf 1- buffered GetInventory
- 2: Tag-Erkennung an Lesekopf 2- buffered GetInventory
- 3: Warten auf Tag-Erkennung
- 4: Aktion an Lesekopf 1 - ReadBlock und WriteBlock
- 5: Aktion an Lesekopf 2 - ReadBlock und WriteBlock

Der data carrier type (iUSEDTYPE) sollte auf den jeweils verwendeten Transpondertypen angepasst werden.

Das Beispielprojekt beinhaltet den Aufruf des RFID-Funktionsbausteins mit unterschiedlichen Befehlen.

Für weitere Informationen zum Ablauf der RFID-Reader-Kommunikation wird auf den Abschnitt Funktionsbaustein FB_RFIDReader > [Handhabung](#) [► 39] verwiesen.

Download: https://infosys.beckhoff.com/content/1031/tf6600_tc3_rfid/Resources/5252195339.zip

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TC3.1.4013	PC oder CX (x86, x64)	Tc2_RFID

8 Anhang

8.1 RFID-Fehlercodes

Fehlerausgaben werden an zwei Ausgängen des RFID-SPS-Funktionsbausteins zur Verfügung gestellt. Die beiden Ausgangsvariablen `iErrorID` [► 77] und `iErrCodeRcv` [► 79] werden im Folgenden erläutert.

`iErrorID`

Wenn ein Fehler vorliegt, zeigt die Ausgangsvariable `iErrorID` die Art des Fehlers an. Die nachfolgende Liste zeigt die möglichen Werte.

```
TYPE E_RFID_ErrID :(  
    eRFERR_NoError           := 0,  
  
    (* general errors *)  
    eRFERR_TimeOutElapsed   := 16#4001,  
  
    (* invalid input parameters *)  
    eRFERR_InvalidCommand   := 16#4101,  
    eRFERR_IncompatibleCfg   := 16#4102,  
    eRFERR_InvalidManufacturer := 16#4103,  
    eRFERR_InvalidTimeOutParam := 16#4104,  
    eRFERR_InvalidRawDataParam := 16#4105,  
    eRFERR_InvalidAccessData := 16#4106,  
    eRFERR_InvalidCfg       := 16#4107,  
    eRFERR_InvalidCfgParam  := 16#4108,  
    eRFERR_InvalidCtrlHeadNumber := 16#4109,  
  
    (* error at receive of response *)  
    eRFERR_InvalidResponse := 16#4201,  
    eRFERR_InvalidRspLen   := 16#4202,  
    eRFERR_InvalidBlocksize := 16#4203,  
    eRFERR_ErrorRcv        := 16#4204,  
    eRFERR_ChecksumError   := 16#4205,  
  
    (* internal errors *)  
    eRFERR_UnknownReaderGroup := 16#4401,  
    eRFERR_CreatedTelegramTooBig := 16#4402,  
);  
END_TYPE
```

Wenn `iErrorID` den Wert `eRFERR_ErrorRcv` besitzt, wurde ein Fehlercode von dem RFID Reader empfangen. Der empfangene Fehlercode wird in der Ausgangsvariablen `iErrCodeRcv` angegeben.

Bei einem anderen Wert für `iErrorID` kann zusätzlich vom RFID Reader ein Fehler empfangen worden sein, der auch in der Ausgangsvariablen `iErrCodeRcv` angezeigt wird.

Wert	ID (hex)	ID (dec)	Beschreibung
eRFERR_TimeOutElapsed	0x4001	16385	<p>Dieser Fehler tritt ein, wenn die als Timeout am Eingang (Default = 5 s) angegebene Zeitdauer abgelaufen ist. Eine noch in Bearbeitung befindliche Aktion wird damit abgebrochen.</p> <p>Es kommt auch zu einem Timeout-Fehler, wenn die serielle Hintergrundkommunikation nicht funktioniert. Dies kann beispielsweise der Fall sein, wenn die falsche Baudrate eingestellt ist oder die Taskzykluszeit nicht ausreicht, um die Daten zu verarbeiten.</p> <p>Siehe auch: RFID Reader Anbindung [► 19] und Dokumentation der SPS-Bibliothek Serielle Kommunikation.</p>
eRFERR_InvalidCommand	0x4101	16641	<p>Der Befehl wurde nicht ausgeführt. Der gewählte Befehl kann mit diesem RFID-Reader-Modell nicht ausgeführt werden. Im Befehlssatz [► 21] sind die zur Verfügung stehenden Befehle gelistet.</p> <p>Um sicherzustellen, dass das vorliegende RFID-Reader-Modell dem Funktionsbaustein bekannt ist, muss, wenn möglich, zu allererst der Befehl „GetReaderVersion“ ausgeführt werden.</p>
eRFERR_IncompatibleCfg	0x4102	16642	<p>Der Befehl wurde nicht ausgeführt. Die aktuelle RFID-Reader-Konfiguration (siehe Ausgangsstruktur ST_RFID_Config [► 52]) ist nicht kompatibel mit dem ausgewählten Befehl (und den dazu gehörigen Eingangsparametern).</p> <p>Es muss sichergestellt werden, dass die Konfiguration zuvor einmal gelesen wurde. Andernfalls kann ebenso dieser Fehlercode auftreten.</p>
eRFERR_InvalidManufacturer	0x4103	16643	<p>Am Eingang des generischen Funktionsbausteins FB_RFIDReader muss mit der Variablen eManufacturer der RFID-Hersteller des RFID-Reader-Modells angegeben werden. Der Fehler eRFERR_InvalidManufacturer signalisiert eine ungültige Angabe.</p>
eRFERR_InvalidTimeOutParam	0x4104	16644	<p>Dieser Fehler wird ausgegeben, wenn die Angabe der Eingangsvariablen tTimeout ungültig ist. Es gilt die Bedingung $tTimeout > tPreSendDelay + tPostSendDelay$.</p>
eRFERR_InvalidRawDataParam	0x4105	16645	<p>Wenn Rohdaten gesendet werden sollen, müssen diese am Eingang des Funktionsbausteins angegeben werden. Der Fehler eRFERR_InvalidRawDataParam wird ausgegeben, wenn die Angabe der Eingangsvariablen pRawDataCommand oder iRawDataCommandLen ungültig ist.</p> <p>Siehe auch: LowLevel-Kommunikation [► 41].</p>
eRFERR_InvalidAccessData	0x4106	16646	<p>Der Befehl wurde nicht ausgeführt, weil die am Eingang angegebenen Parameter in der Struktur ST_RFID_AccessData [► 52] ungültig sind.</p>
eRFERR_InvalidCfg	0x4107	16647	<p>Der Befehl „Konfiguration ändern“ wurde nicht ausgeführt, weil die angegebenen Konfigurationsdaten ungültig sind. Die Konfigurationsdaten liegen als Konfigurationsstruktur an (bUseCfgReg = FALSE). Im Zweifelsfall sind die genauen gültigen Wertebereiche der Daten in der proprietären Protokollspezifikation durch den Hersteller gegeben.</p> <p>Wenn die zuletzt gelesene Konfiguration für die in der Konfigurationsstruktur nicht verfügbaren Parameter genutzt wird (bUseCfgDefault = FALSE in ST_RFID_ConfigIn [► 50]), muss sichergestellt werden, dass die Konfiguration zuvor gelesen wurde. Andernfalls kann ebenso dieser Fehlercode auftreten.</p>
eRFERR_InvalidCfgParam	0x4108	16648	<p>Der Befehl „Konfiguration ändern“ wurde nicht ausgeführt, weil die angegebenen Eingangsparameter der Konfigurationsdaten ungültig sind. Die Konfigurationsdaten sollen als Konfigurationsregister oder Konfigurationsstruktur anliegen. Dessen Länge oder ein anderer Parameter in ST_RFID_ConfigIn [► 50] sind ungültig.</p>
eRFERR_InvalidCtrlHeadNumber	0x4109	16649	<p>Wenn ein RFID-Lesegerät mit mehreren Leseköpfen angesprochen wird, wird in der Eingangsstruktur ST_RFID_Control [► 45] eine Wahl des Lesekopfes angegeben. Ist für den angegebenen Wert kein Lesekopf verfügbar, so wird dieser Fehlerwert ausgegeben.</p>

Wert	ID (hex)	ID (dec)	Beschreibung
eRFERR_InvalidResponse	0x4201	16897	Wenn die Bytefolge der empfangenen Response keiner bekannten Nachrichtenart entspricht oder einzelne Bytes nicht die erforderlichen Werte aufweisen, wird dieser Fehler ausgegeben. Die empfangenen Daten können als Rohdatenblock am Ausgang ST_RFID_RawData [▶ 45] analysiert werden.
eRFERR_InvalidRspLen	0x4202	16898	Wenn die Bytefolge theoretisch einer bekannten Nachrichtenart entspricht, die Länge allerdings nicht der Erwartung entspricht, wird diese Fehlermeldung erzeugt. Die empfangenen Daten können als Rohdatenblock am Ausgang ST_RFID_RawData [▶ 45] analysiert werden.
eRFERR_InvalidBlocksize	0x4203	16899	Tritt dieser Fehlerwert ein, so konnten die empfangenen vom Transponder gelesenen Daten nicht ausgewertet werden. Die Anzahl der empfangenen Bytes signalisiert, dass die konfigurierte Blocksize nicht mit der Eingabe beim Befehlsaufruf übereinstimmt.
eRFERR_ErrorRcv	0x4204	16900	Ein Fehlercode wurde mit der empfangenen Nachricht gesendet. Der vom RFID Reader angezeigte Fehler wird ebenso ausgegeben und von der Ausgangsvariablen „iErrCodeRcv“ repräsentiert (Erläuterung am Ende dieses Abschnitts).
eRFERR_ChecksumError	0x4205	16901	Je nach Protokollspezifikation wird eine Checksumme, beispielsweise eine CRC-Prüfsumme, im Telegramm mit gesendet. Wird von der Steuerung ein Telegramm mit fehlerhafter Checksumme empfangen, so wird dieser Fehler ausgegeben.
eRFERR_UnknownReaderGroup	0x4401	17409	Die RFID-Bibliothek teilt die RFID-Reader-Modelle intern in Gruppen ein. Der Fehler „eRFERR_UnknownReaderGroup“ kann auftreten, wenn der RFID Reader noch keiner Gruppe zugeordnet ist. Deshalb muss je nach RFID-Reader-Modell bei Programmstart als erste Abfrage der Befehl „GetReaderVersion“ ausgeführt werden.
eRFERR_CreatedTelegramTooBig	0x4402	17410	Es wurde versucht, ein Telegramm zu senden, das die maximal mögliche Größe von 300 Bytes überschritten hat. Es können nur Telegramme mit bis zu 300 Bytes versendet werden.



In einigen wenigen Fällen werden vom RFID-Gerät mehrere Telegramme unmittelbar hintereinander versendet. Es ist deshalb wichtig, immer den Fehler zu erkennen und zu beheben, der als Erstes eintraf.

iErrCodeRcv

Wenn vom RFID Reader ein Fehlercode mitgeschickt wird, wird dieser in der Ausgangsvariable iErrCodeRcv ausgegeben. Teilweise werden auch Statusmeldungen vom RFID Reader mitgeschickt und an iErrCodeRcv ausgegeben, die nicht zu einem Fehler führen (bError bleibt FALSE und iErrorID zeigt keinen Fehler).

Eine Liste möglicher Werte kann entweder der Datentypdeklarationen (E_RFID_ErrCodeRcv_Balluff, E_RFID_ErrCodeRcv_Deister, E_RFID_ErrCodeRcv_Leuze usw.) der SPS-RFID-Bibliothek über die TwinCAT-XAE-Bibliothekdetails oder direkt der Protokollspezifikation entnommen werden.

Zur weiteren Fehleranalyse wird auf die Logging-Möglichkeit hingewiesen. Hierzu wird der Eingangsparameter bLogging gesetzt. Details finden Sie in der [Parameterbeschreibung in der API \[▶ 45\]](#).

8.2 Support und Service

Beckhoff und seine weltweiten Partnerfirmen bieten einen umfassenden Support und Service, der eine schnelle und kompetente Unterstützung bei allen Fragen zu Beckhoff Produkten und Systemlösungen zur Verfügung stellt.

Downloadfinder

Unser [Downloadfinder](#) beinhaltet alle Dateien, die wir Ihnen zum Herunterladen anbieten. Sie finden dort Applikationsberichte, technische Dokumentationen, technische Zeichnungen, Konfigurationsdateien und vieles mehr.

Die Downloads sind in verschiedenen Formaten erhältlich.

Beckhoff Niederlassungen und Vertretungen

Wenden Sie sich bitte an Ihre Beckhoff Niederlassung oder Ihre Vertretung für den [lokalen Support und Service](#) zu Beckhoff Produkten!

Die Adressen der weltweiten Beckhoff Niederlassungen und Vertretungen entnehmen Sie bitte unserer Internetseite: www.beckhoff.com

Dort finden Sie auch weitere Dokumentationen zu Beckhoff Komponenten.

Beckhoff Support

Der Support bietet Ihnen einen umfangreichen technischen Support, der Sie nicht nur bei dem Einsatz einzelner Beckhoff Produkte, sondern auch bei weiteren umfassenden Dienstleistungen unterstützt:

- Support
- Planung, Programmierung und Inbetriebnahme komplexer Automatisierungssysteme
- umfangreiches Schulungsprogramm für Beckhoff Systemkomponenten

Hotline: +49 5246 963-157
E-Mail: support@beckhoff.com

Beckhoff Service

Das Beckhoff Service-Center unterstützt Sie rund um den After-Sales-Service:

- Vor-Ort-Service
- Reparaturservice
- Ersatzteilservice
- Hotline-Service

Hotline: +49 5246 963-460
E-Mail: service@beckhoff.com

Beckhoff Unternehmenszentrale

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20
33415 Verl
Deutschland

Telefon: +49 5246 963-0
E-Mail: info@beckhoff.com
Internet: www.beckhoff.com

Mehr Informationen:
www.beckhoff.com/tf6600

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

