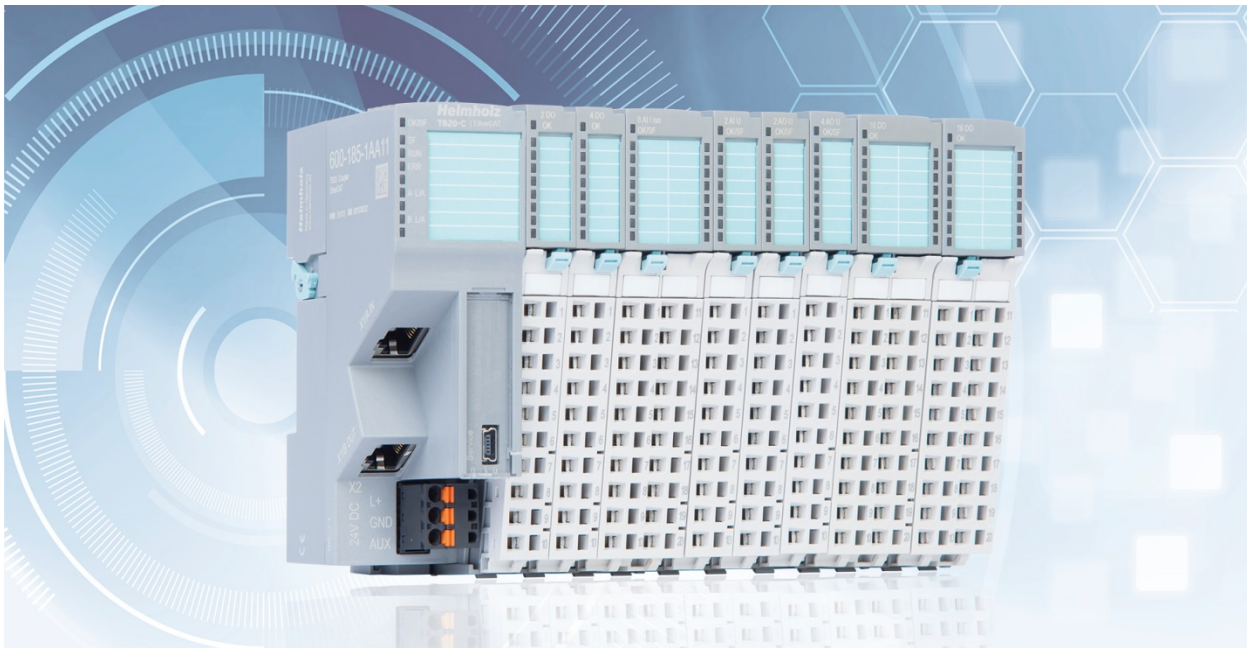


TB20 – EtherCAT® Buskoppler

Handbuch

Ausgabe 02 | 03.07.2019

Handbuch-Bestellnummer: 960-185-1AA11/de

Helmholz GmbH & Co. KG | Hannberger Weg 2 | 91091 Großenseebach | Germany
Phone +49 9135 7380-0 | Fax +49 9135 7380-110 | info@helmholz.de | www.helmholz.de

Hinweise

Die jeweils aktuellste Version des Handbuchs finden Sie im Internet unter www.helmholz.de.

Wir freuen uns über Verbesserungsvorschläge und Anregungen.

Inhalt

1 Allgemeines

Diese Betriebsanleitung gilt ausschließlich für Geräte, Baugruppen, Software und Leistungen der Helmholz GmbH & Co. KG.

1.1 Zielgruppe des Handbuchs

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist. Zur Installation, Inbetriebnahme und zum Betrieb der Komponenten ist die Beachtung der Hinweise und Erklärungen dieser Betriebsanleitung unbedingt notwendig.

Projektierungs-, Ausführungs- und Bedienungsfehler können den ordnungsgemäßen Betrieb der TB20-Geräte beeinträchtigen und Personen-, Sach- oder Umweltschäden zur Folge haben. Es darf nur ausreichend qualifiziertes Fachpersonal die TB20-Geräte bedienen!

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

1.2 Sicherheitshinweise

Die Sicherheitshinweise müssen beachtet werden, um Personen und Lebewesen, materielle Güter und die Umwelt vor Schäden zu bewahren. Die Sicherheitshinweise zeigen mögliche Gefahren auf und geben Hinweise, wie Gefahrensituationen vermieden werden können.

1.3 Hinweiszeichen und Signalwörter im Handbuch

GEFAHR

Wenn der Gefahrenhinweis nicht beachtet wird besteht die unmittelbare Gefahr für Gesundheit und Leben von Personen durch elektrische Spannung.

WARNUNG

Wenn der Gefahrenhinweis nicht beachtet wird besteht die wahrscheinliche Gefahr für Gesundheit und Leben von Personen.

VORSICHT

Wenn der Gefahrenhinweis nicht beachtet wird können Personen verletzt oder geschädigt werden.

ACHTUNG

Macht auf Fehlerquellen aufmerksam, die Geräte oder Umwelt schädigen können.

HINWEIS

Gibt einen Hinweis zum besseren Verständnis oder zur Vermeidung von Fehlern.

1.4 Bestimmungsgemäße Verwendung

Das TB20 I/O-System ist ein offenes, modular aufgebautes, dezentrales Peripheriesystem für die Montage auf einer 35 mm-Hutschiene.

Die Kommunikation mit einer übergeordneten Steuerung erfolgt über ein Bussystem/Netzwerk über einen TB20 Buskoppler. An einen Buskoppler können bis zu 64 Module aus dem TB20-Sortiment angereiht werden. Die Buskoppler unterstützen Hot-Plug für den Tausch von Modulen im laufenden Betrieb.

Die gesamten Komponenten werden mit einer werkseitigen Hard- und Software-Konfiguration ausgeliefert. Die Hard- und Software-Konfiguration auf die Anwendungsbedingungen muss durch den Anwender erfolgen. Änderungen der Hard-, oder Software-Konfiguration, die über die dokumentierten Möglichkeiten hinausgehen, sind unzulässig und bewirken den Haftungsausschluss der Helmholz GmbH & Co. KG.

Die TB20-Geräte dürfen nicht als alleiniges Mittel zur Abwendung gefährlicher Zustände an Maschinen und Anlagen eingesetzt werden.

Der einwandfreie und sichere Betrieb der TB20-Geräte setzt sachgemäßen Transport, sachgemäße Lagerung, Aufstellung, Montage, Installation, Inbetriebnahme, Bedienung und Instandhaltung voraus.

Die in den technischen Daten angegebenen Umgebungsbedingungen müssen eingehalten werden.

Die TB20-Systeme besitzen den Schutzgrad IP 20 und müssen zum Schutz vor Umwelteinflüssen in einem elektrischen Betriebsraum oder einem Schaltkasten/Schaltschrank montiert werden. Um unbefugtes Bedienen zu verhindern, müssen die Türen der Schaltkästen/Schaltschränke während des Betriebes geschlossen und ggf. gesichert sein.

GEFAHR

TB20-Geräte können mit Baugruppen bestückt werden, die gefährlich hohe Spannungen führen können. Durch an die TB20-Geräte angeschlossene Spannungen können Gefährdungen bei Arbeiten an den TB20-Geräten entstehen.

1.5 Missbrauch

WARNUNG

Die Folgen einer nicht bestimmungsgemäßen Verwendung können Personenschäden des Benutzers oder Dritter sowie Sachschäden an der Steuerung, am Produkt oder Umweltschäden sein. Setzen Sie TB20-Geräte nur bestimmungsgemäß ein!

1.6 Montage

1.6.1 Zugangsbeschränkung

Die Baugruppen sind offene Betriebsmittel und dürfen nur in elektrischen Betriebsräumen, Schränken oder Gehäusen installiert werden. Der Zugang zu den elektrischen Betriebsräumen, Schränken oder Gehäusen darf nur über Werkzeug oder Schlüssel möglich sein und nur unterwiesenem oder zugelassenem Personal gestattet werden.

1.6.2 Elektrische Installation

Die regional gültigen Sicherheitsbestimmungen sind zu beachten.

GEFAHR

TB20-Geräte können mit Baugruppen bestückt werden, die gefährlich hohe Spannungen führen können. Durch an die TB20-Geräte angeschlossene Spannungen können Gefährdungen bei Arbeiten an den TB20-Geräten entstehen.

1.6.3 Schutz vor elektrostatischen Entladungen

Um Schäden durch elektrostatische Entladungen zu verhindern, sind bei Montage- und Servicearbeiten folgende Sicherheitsmaßnahmen zu befolgen:

- Bauteile und Baugruppen nie direkt auf Kunststoff-Gegenstände (z.B. Styropor, PE-Folie) legen, auch deren Nähe meiden.
- Vor Beginn der Arbeit geeignete Maßnahme wählen, um sich zu entladen.
- Nur mit entladenem Werkzeug arbeiten.
- Bauteile und Baugruppen nicht an Kontakten berühren.

1.6.4 Überstrom-Schutz

Zum Schutz des TB20 und der Zuleitung ist eine Leitungsschutz-Sicherung *8 A träge* erforderlich.

1.6.5 EMV-Schutz

Um die elektromagnetische Verträglichkeit (EMV) in Ihren Schaltschränken in elektrisch rauer Umgebung sicherzustellen, sind bei der Konstruktion und dem Aufbau die bekannten Regeln des EMV-gerechten Aufbaus zu beachten.

1.6.6 Betrieb

Betreiben Sie das TB20 nur in einwandfreiem Zustand. Die zulässigen Einsatzbedingungen und Leistungsgrenzen müssen eingehalten werden. Nachrüstungen, Veränderungen oder Umbauten am Gerät sind grundsätzlich verboten.

Das TB20 ist ein Betriebsmittel zum Einsatz in industriellen Anlagen. Während des Betriebs kann das TB20 gefährliche Spannungen führen. Während des Betriebs müssen alle Abdeckungen am Gerät und der Installation geschlossen sein, um den Berührungsschutz zu gewährleisten.

1.6.7 Haftung

Der Inhalt dieser Bedienungsanleitung unterliegt Änderungen, die durch die ständige Weiterentwicklung der Produkte der Helmholz GmbH & Co. KG entstehen. Für den Fall, dass dieses Handbuch technische Fehler oder Schreibfehler enthält, behalten wir uns das Recht vor, Änderungen jederzeit und ohne Ankündigung durchzuführen. Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte gemacht werden. Über die in der Bedienungsanleitung enthaltenen Anweisungen hinaus sind in jedem Fall die gültigen nationalen und internationalen Normen und Vorschriften zu beachten.

1.6.8 Haftungsausschluss

Die Helmholz GmbH & Co. KG haftet nicht bei Schäden, wenn diese durch nicht bestimmungs- oder sachgemäße Benutzung oder Anwendung der Produkte verursacht wurden.

Die Helmholz GmbH & Co. KG übernimmt keine Haftung für eventuell in der Bedienungsanleitung enthaltene Druckfehler oder sonstige Ungenauigkeiten, es sei denn, es sind gravierende Fehler, die der Helmholz GmbH & Co. KG nachweislich bereits bekannt sind.

Über die in der Bedienungsanleitung enthaltenen Anweisungen hinaus sind in jedem Fall die gültigen nationalen und internationalen Normen und Vorschriften zu beachten.

Die Helmholz GmbH & Co. KG haftet nicht bei Schäden, die durch Software, die auf Geräten des Anwenders aktiv ist und über die Fernwartungsverbindung weitere Geräte oder Prozesse beeinträchtigt, schädigt oder infiziert und unerwünschten Datentransfer auslöst oder ermöglicht.

1.6.9 Gewährleistung

Melden Sie Mängel sofort nach Feststellung des Fehlers beim Hersteller an.

Die Gewährleistung erlischt bei:

- Missachtung dieser Betriebsanleitung
- nicht bestimmungsgemäßer Verwendung des Geräts
- unsachgemäßem Arbeiten an und mit dem Gerät
- Bedienungsfehlern
- eigenmächtigen Veränderungen am Gerät

Es gelten die bei Vertragsabschluss unter „Allgemeine Geschäftsbedingungen der Firma Helmholz GmbH & Co. KG“ getroffenen Vereinbarungen.

2 Systemübersicht

2.1 Allgemeines

Das TB20 I/O-System ist ein offenes, modular aufgebautes, dezentrales Peripheriesystem für die Montage auf einer 35mm Hutschiene.

Es besteht aus folgenden Komponenten:

- Buskoppler
- Peripheriemodule
- Einspeise-/Trennmodule
- Powermodule.

Aus diesen Komponenten können Sie ein speziell auf Ihre Bedürfnisse zugeschnittenes Automatisierungssystem aufbauen. Dabei lassen sich bis zu 64 Module an einen Buskoppler anreihen. Alle Komponenten gehören zur Schutzklasse IP20.

2.2 Die Komponenten des TB20 I/O-Systems

2.2.1 Der Buskoppler

Der Buskoppler enthält ein Bus-Interface und ein Powermodul. Das Bus-Interface stellt die Verbindung zum übergeordneten Bus-System her und dient somit zum Austausch der E/A-Signale mit der CPU des Automatisierungssystems.

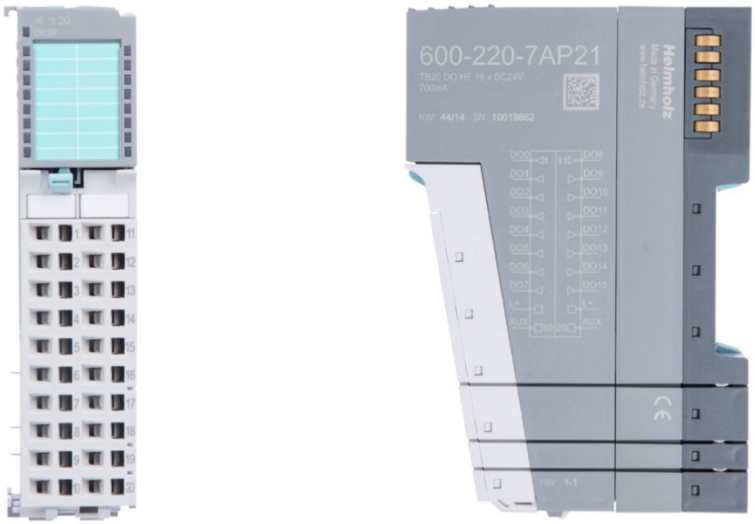
Das Powermodul übernimmt die Stromversorgung sowohl der Koppler-Elektronik als auch der angereihten Peripheriemodule.

2.2.2 Peripheriemodule

Bei den Peripheriemodulen handelt es sich um Elektronik-Komponenten, an die die Peripheriegeräte (Sensoren, Aktoren) angeschlossen werden. Es gibt daher verschiedene Peripheriemodule je nach Aufgabe und Funktion.

Beispiel: Peripheriemodul mit 10-poligem Frontstecker

Beispiel: Peripheriemodul mit 20-poligem Frontstecker



2.2.3 Einspeise-/Trennmodul

Der Buskoppler liefert die Versorgungsspannung für den Kommunikationsbus (5 V, oben) und die Versorgungsspannung für die externen Signale (24 V, unten). Diese Spannungen werden über die Basismodule von Modul zu Modul geleitet.

Mit Einspeise-/Trennmodulen lässt sich die Spannungsversorgung für die externen Signale in einzelne Versorgungsabschnitte aufteilen, die separat gespeist werden. Die Signale des Kommunikationsbusses und die Versorgungsspannung für den Kommunikationsbus sind davon nicht betroffen und werden weiter durchgeleitet.

HINWEIS

Einspeise-/Trennmodule haben zur besseren Unterscheidbarkeit eine hellere Gehäusefarbe.

2.2.4 Powermodul

Der Buskoppler liefert die Versorgungsspannung für den Kommunikationsbus (5 V, oben) und die Versorgungsspannung für die externen Signale (24 V, unten). Diese Spannungen werden über die Basismodule von Modul zu Modul geleitet.

Mit Powermodulen lässt sich die Spannungsversorgung sowohl für die externen Signale als auch für den Kommunikationsbus in einzelne Versorgungsabschnitte aufteilen, die separat gespeist werden.

Powermodule liefern die gesamte Stromversorgung für die nachfolgend angereihten Peripheriemodule, gegebenenfalls bis zum nächsten Power- oder Einspeise-/Trennmodul. Ein Powermodul ist immer dann erforderlich, wenn die Stromversorgung durch den Koppler allein nicht ausreicht, wenn also beispielsweise viele Module mit hohem Strombedarf eingesetzt werden. Wann ein Powermodul benötigt wird lässt sich mit Hilfe der Konfigurationssoftware „TB20-ToolBox“ ermitteln.

HINWEIS

Powermodule haben zur besseren Unterscheidbarkeit eine hellere Gehäusefarbe.

2.2.5 Abschlusselement

Das Abschlusselement schützt die Kontakte des letzten Basismoduls vor unbeabsichtigten Berührungen und deckt diese rechts außen ab.

2.2.6 Aufbau eines Moduls

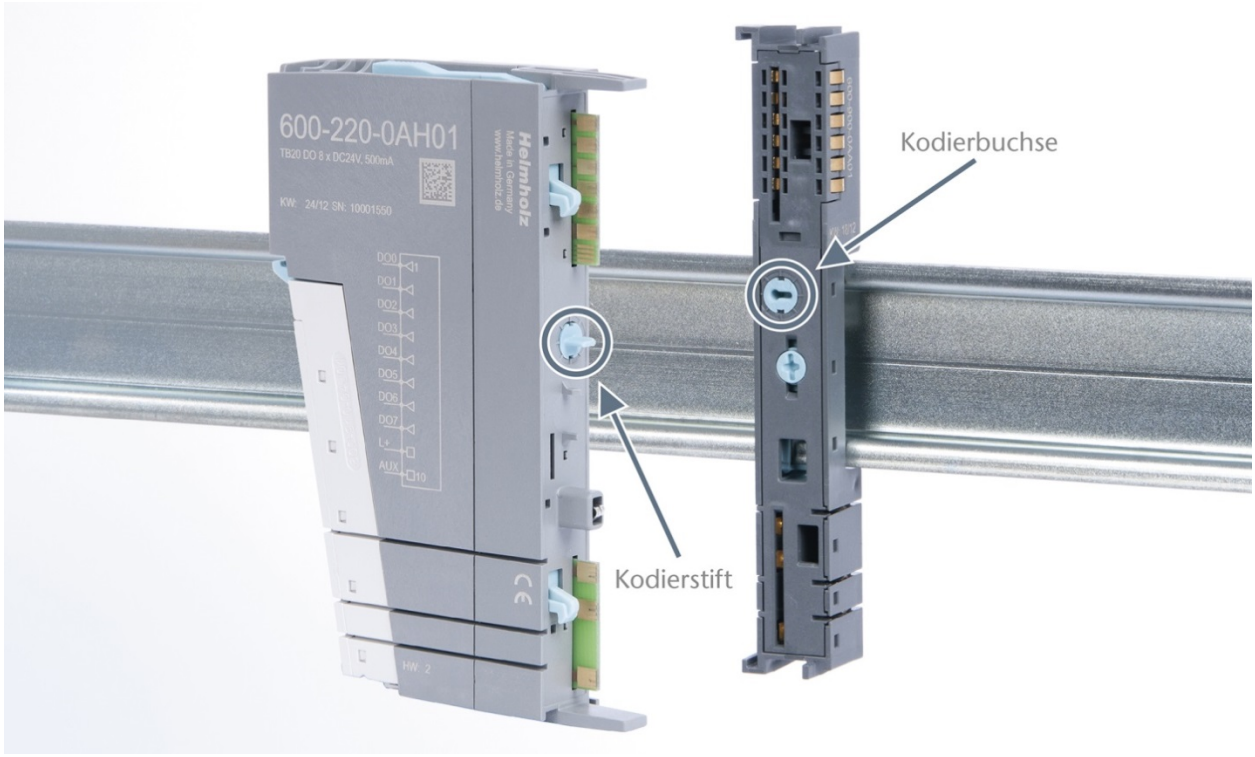
Jedes Modul besteht aus drei Teilen:

- dem Basismodul,
- dem Elektronikmodul und dem
- Frontstecker.

2.2.7 Modulkodierung

Elektronikmodul und Basismodul besitzen Kodierelemente, durch die bei Wartung und Reparatur verhindert werden soll, dass Ersatz-Elektronikmodule versehentlich auf eine falsche Position gesteckt werden.

Bei den Kodierelementen handelt es sich um einen Kodierstift am Elektronikmodul und eine Kodierbuchse am Basismodul (siehe Abbildung).



Kodierstift und Kodierbuchse können jeweils acht verschiedene Positionen einnehmen. Jede der acht Positionen kann einer bestimmten Modulart (Digital In, Digital Out, Analog In, Analog Out, Power usw.) zugeordnet werden. Nur wenn die Position von Kodierstift und Kodierbuchse gleich sind, lässt sich das Elektronikmodul auf das Basismodul stecken. Bei unterschiedlichen Stellungen blockiert das Elektronikmodul mechanisch.

3 Montage und Demontage

GEFAHR

TB20-Module können lebensgefährliche Spannung führen.

Vor Beginn jeglicher Arbeiten an den Komponenten des Systems TB20 sind alle Komponenten und die zuführenden Leitungen spannungsfrei zu schalten! Bei Arbeiten unter Spannung besteht Lebensgefahr durch Stromschlag!

ACHTUNG

Die Montage ist gemäß VDE 0100/IEC 364 durchzuführen bzw. nach geltenden nationalen Normen durchzuführen. Das TB20 I/O-System besitzt den Schutzgrad IP20. Wird ein höherer Schutzgrad benötigt, muss der Einbau in ein Gehäuse oder einen Schaltschrank erfolgen. Um einen sicheren Betrieb zu gewährleisten darf die Umgebungstemperatur nicht mehr als 60°C betragen!

3.1 Einbaulage

Das TB20 I/O-System kann in beliebiger Lage eingebaut werden. Eine optimale Durchlüftung und somit auch die maximale Umgebungstemperatur lassen sich nur im horizontalen Aufbau erreichen.

3.2 Mindestabstand

Es wird empfohlen, bei der Montage von Koppler und Modulen die aufgeführten Mindestabstände einzuhalten. Durch die Einhaltung der Mindestabstände

- ist das Montieren bzw. Demontieren der Module möglich, ohne andere Anlagenteile demontieren zu müssen,
- ist genügend Raum vorhanden um alle Anschlüsse und Kontaktierungsmöglichkeiten mit handelsüblichem Zubehör zu verbinden,
- ist Platz für evtl. nötige Kabelführungen vorhanden.

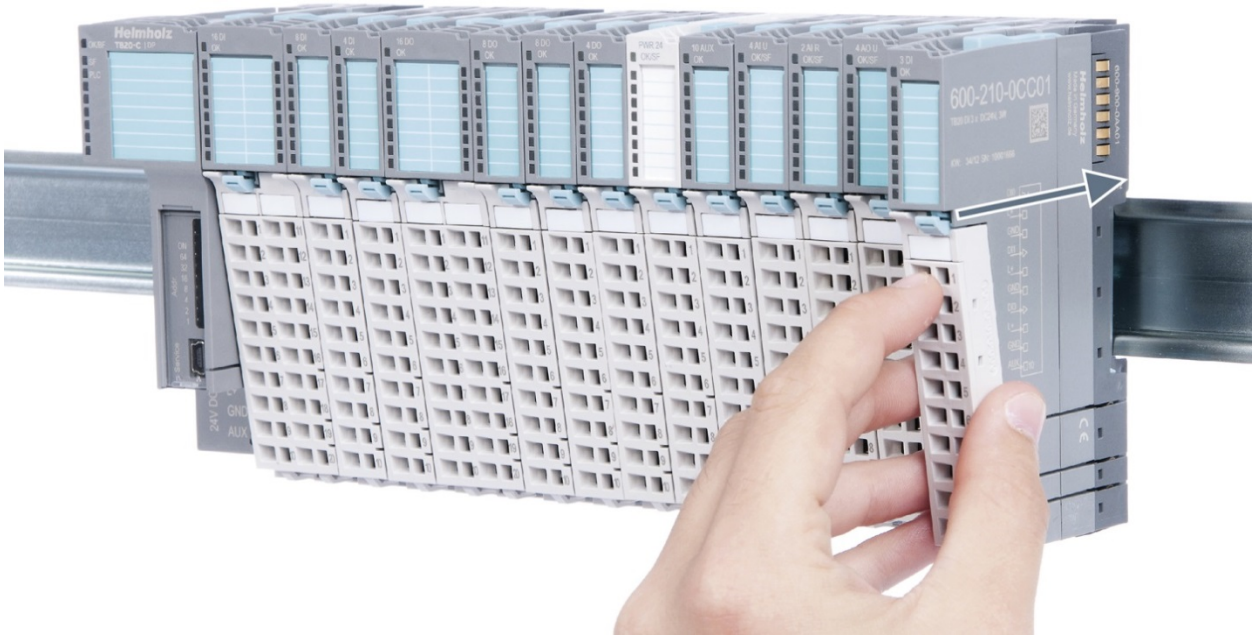
Die Mindestabstände für die Montage der TB20-Komponenten betragen oben und unten je 30 mm und an den Seiten je 10 mm.

3.3 Montage und Demontage von Peripheriemodulen

3.3.1 Montage

Montage eines kompletten Peripheriemoduls:

1. Setzen Sie das zusammengebaute Modul gerade von vorne an die Hutschiene an. Achten Sie darauf, dass das Modul in die obere und untere Führungsschiene des vorhergehenden Moduls eingreift.
2. Drücken Sie dann den oberen Teil des Moduls so weit zur Hutschiene hin, bis die im Inneren befindliche Hutschienenkralle mit einem leisen Klicken auf der Hutschiene einrastet.



Montage der einzelnen Teile eines Peripheriemoduls nacheinander:

1. Setzen Sie das Basismodul schräg von unten an die Hutschiene an.
2. Drücken Sie dann den oberen Teil des Basismoduls so weit zur Hutschiene hin, bis das Modul parallel zur Hutschiene ist und die im Inneren befindliche Hutschienenkralle mit einem leisen Klicken einrastet.
3. Setzen Sie das passend kodierte Elektronikmodul (siehe Abschnitt *2.2.7)* gerade von oben auf das Basismodul und drücken Sie es dann sanft zum Basismodul, bis beide Module ganz aufeinanderliegen und die Modulkralle mit leisem Klicken einrastet.
4. Setzen Sie abschließend den Frontstecker schräg von unten auf das Elektronikmodul und drücken Sie den Frontstecker dann sanft auf das Elektronikmodul, bis die Frontsteckerkralle mit leisem Klicken einrastet.

3.3.2 Demontage

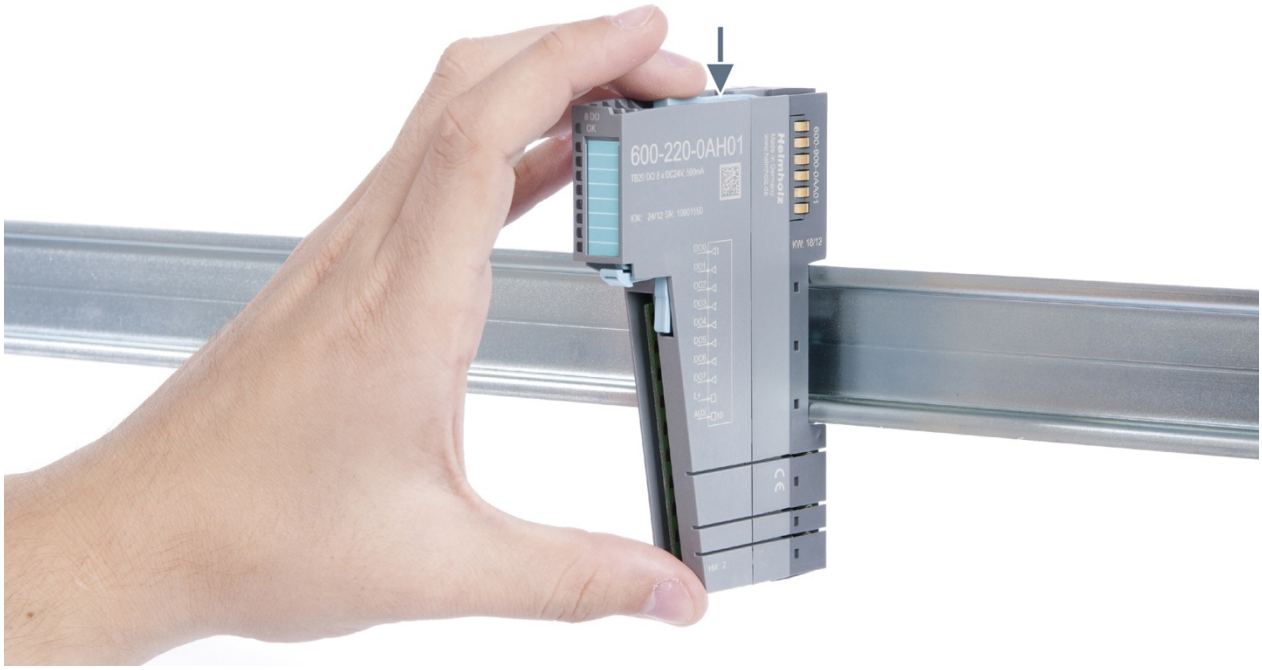
Die Demontage eines Peripheriemoduls geschieht in vier Schritten:

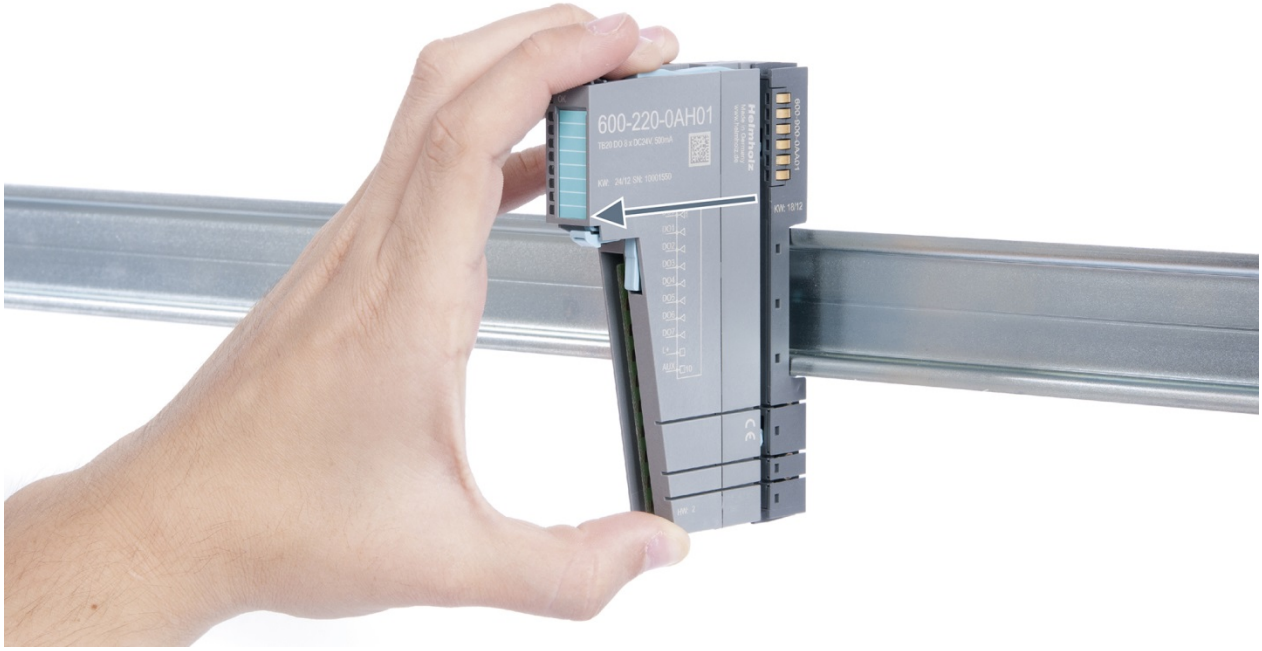
Schritt 1: Frontstecker abziehen

Zum Abziehen des Frontsteckers drücken Sie von unten nach oben auf die Nase oberhalb des Frontsteckers. Der Frontstecker wird herausgedrückt und kann abgezogen werden.

Schritt 2: Elektronikmodul abziehen

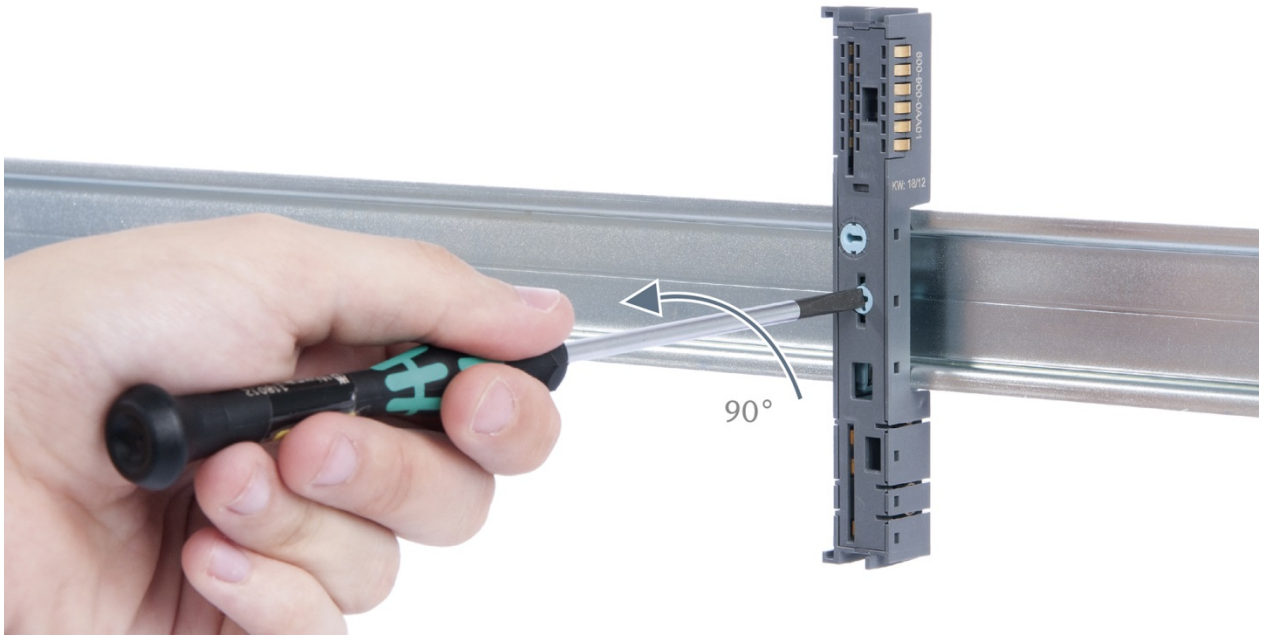
Drücken Sie dazu mit dem Mittelfinger von oben auf den Hebel und ziehen Sie dann bei gedrücktem Hebel das Elektronikmodul mit Daumen und Mittelfinger ab.





Schritt 3: Basismodul entriegeln

Entriegeln Sie das Basismodul mit einem Schraubendreher. Zum Lösen der Entriegelung den Schraubenzieher 90° nach links drehen.



Schritt 4: Basismodul abziehen

Ziehen Sie das Basismodul nach vorne ab.

3.4 Wechsel des Elektronikmoduls

Das Elektronikmodul eines Peripheriemoduls lässt sich in vier Schritten austauschen.

Sollte ein Austausch des Elektronikmoduls im laufenden Betrieb vorgenommen werden, beachten Sie bitte auch die technischen Rahmenbedingungen des eingesetzten Buskopplers.

GEFAHR

TB20 Module können lebensgefährliche Spannung führen.

Vor Beginn jeglicher Arbeiten an den Komponenten des Systems TB20 sind alle Komponenten und die zuführenden Leitungen spannungsfrei zu schalten! Bei Arbeiten unter Spannung besteht Lebensgefahr durch Stromschlag!

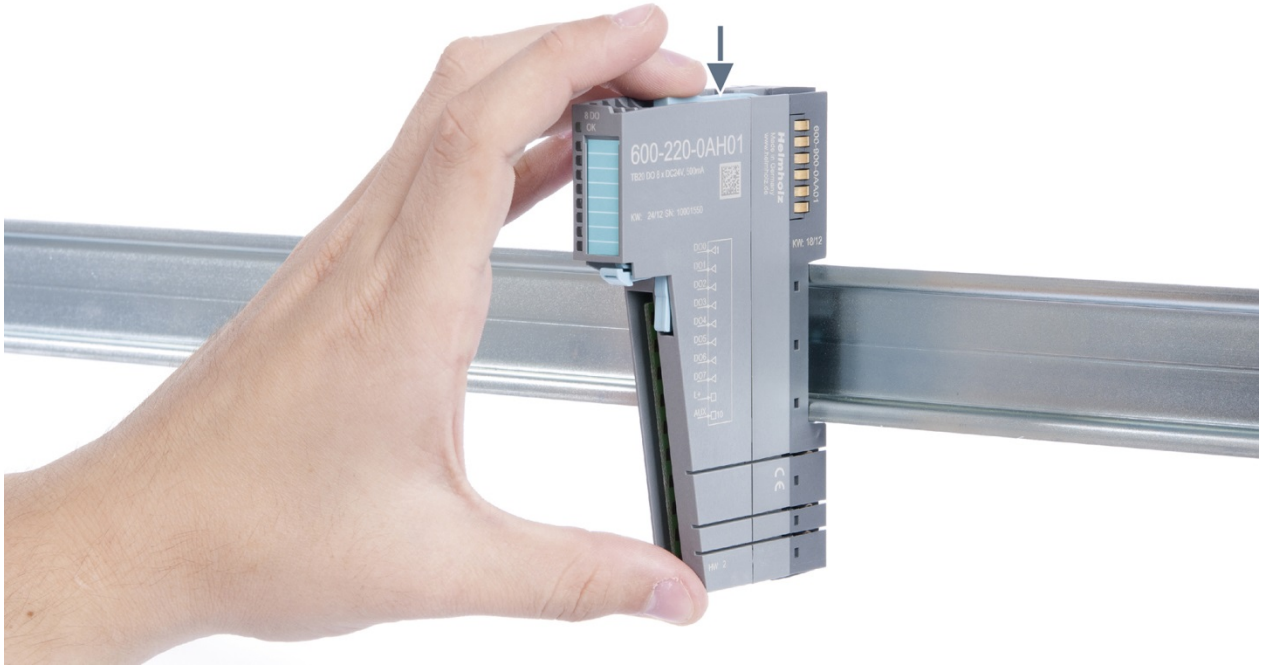
Beachten Sie den Stromlaufplan der Anlage und schalten Sie gefährliche Spannungen vor Beginn der Arbeiten ab!

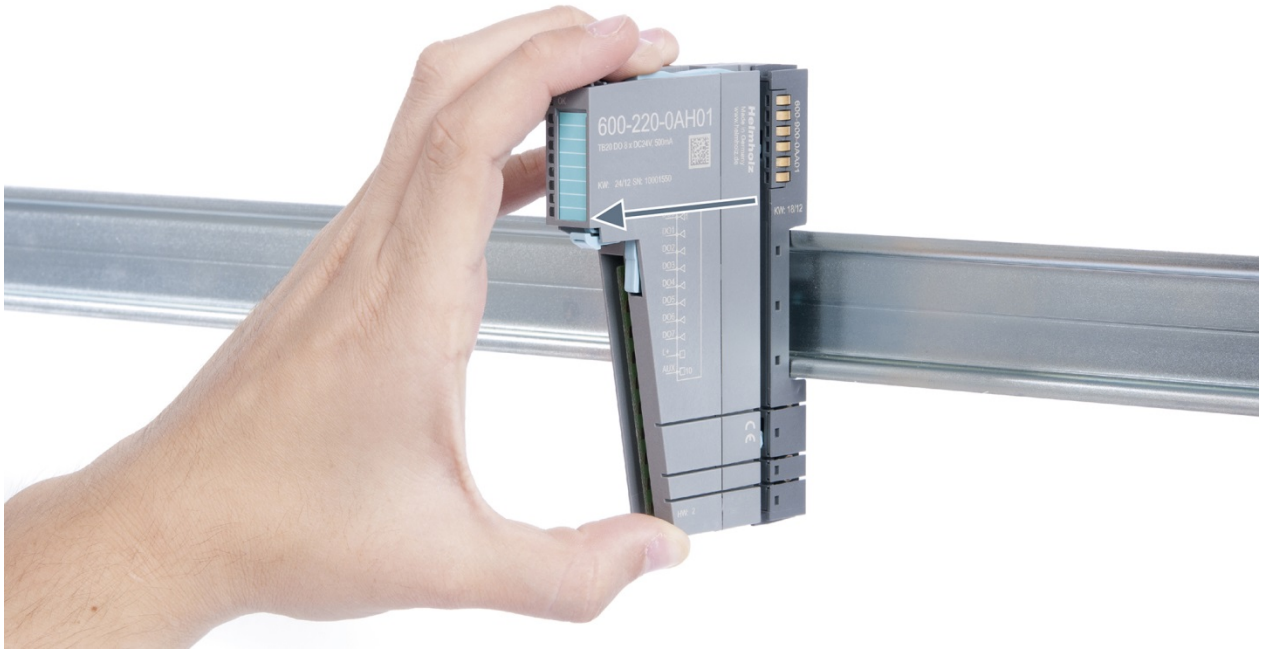
Schritt 1: Frontstecker abziehen

Zum Abziehen des Frontsteckers drücken Sie von unten nach oben auf die Nase oberhalb des Frontsteckers (siehe nachfolgende Abbildung). Der Frontstecker wird herausgedrückt und kann abgezogen werden.

Schritt 2: Elektronikmodul abziehen

Zum Abziehen des Elektronikmoduls drücken Sie mit dem Zeigefinger von oben auf den Hebel und ziehen Sie dann bei gedrücktem Hebel das Elektronikmodul mit Daumen und Mittelfinger ab (siehe nachfolgende Abbildung).





Schritt 3: Neues Elektronikmodul aufstecken

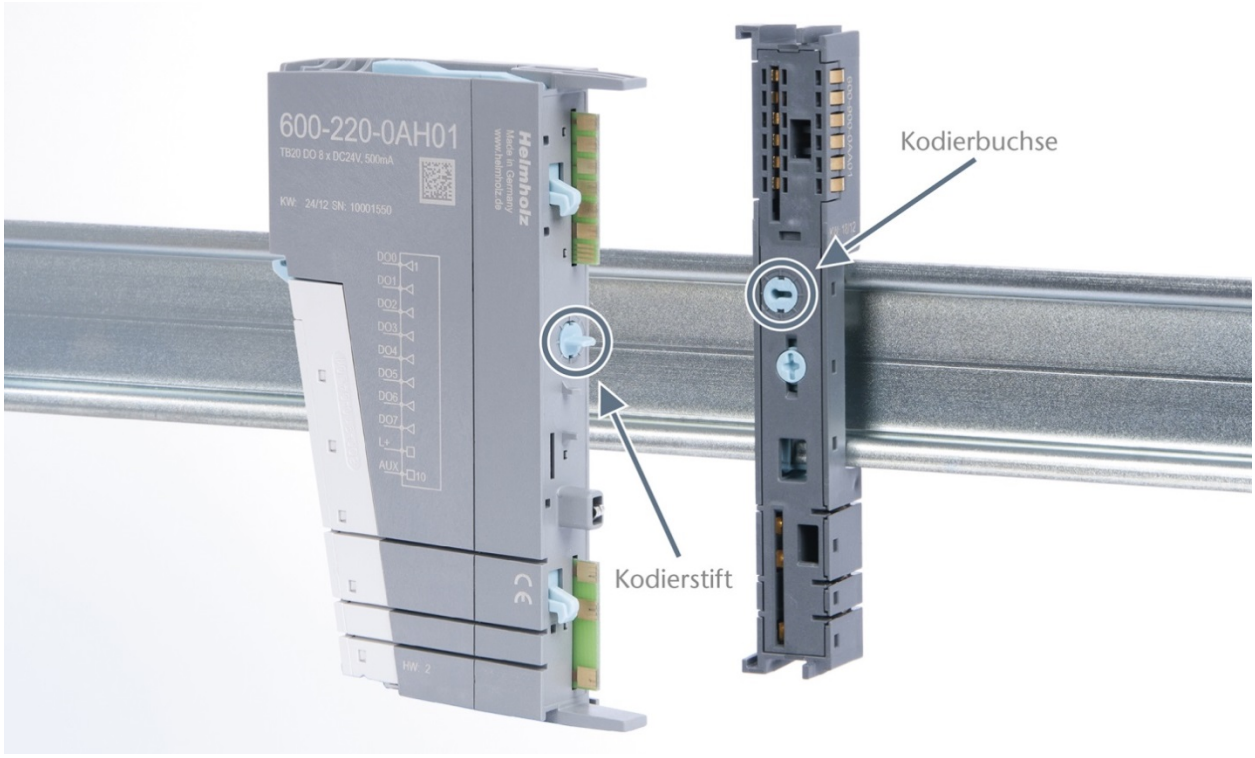
ACHTUNG

Das Elektronikmodul muss in einer durchgehenden Bewegung auf das Basismodul aufgeschnappt werden. Wird das Elektronikmodul nicht gerade und fest auf das Basismodul aufgeschnappt, kann es zu Busstörungen kommen.

ACHTUNG

Lässt sich das Elektronikmodul nicht auf das Basismodul aufstecken, prüfen Sie, ob die Kodierelemente von Elektronikmodul und Basismodul (siehe Abbildung unten) zusammenpassen. Wenn das Elektronikmodul nicht auf das Basismodul passt, dann verwenden Sie möglicherweise ein falsches Elektronikmodul.

Näheres zu den Kodierelementen siehe Abschnitt 2.2.7.

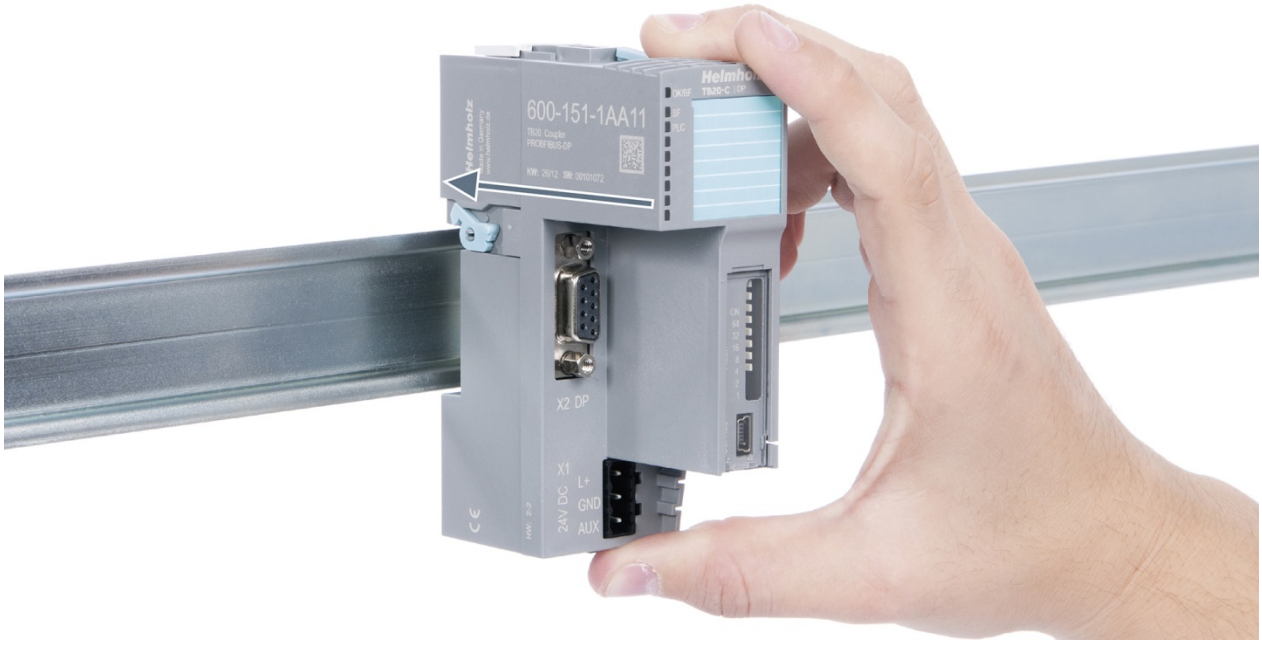


Schritt 4: Frontstecker aufstecken

3.5 Montage und Demontage des Kopplers

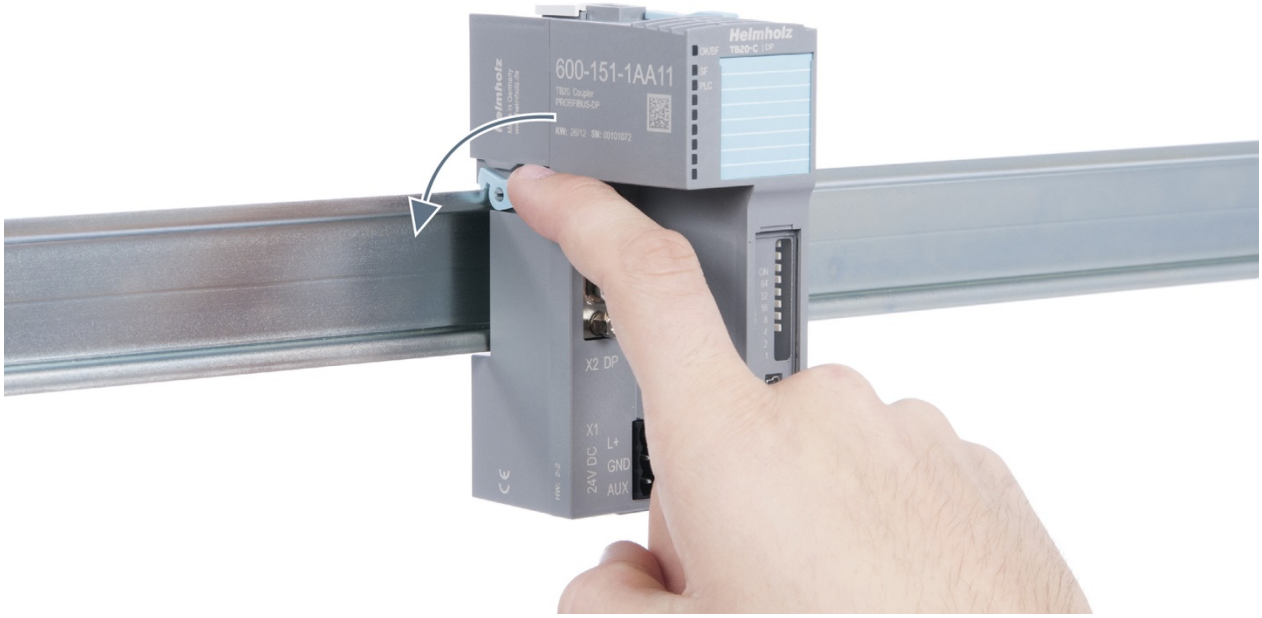
3.5.1 Montage

Setzen Sie den Koppler zusammen mit dem aufgesetzten Basismodul gerade von vorne an die Hutschiene an. Drücken Sie dann den Koppler soweit zur Hutschiene hin, bis die Hutschienenkralle des Basismoduls mit einem leisen Klicken einrastet.



Schritt 2: Koppler an der Hutschiene verriegeln

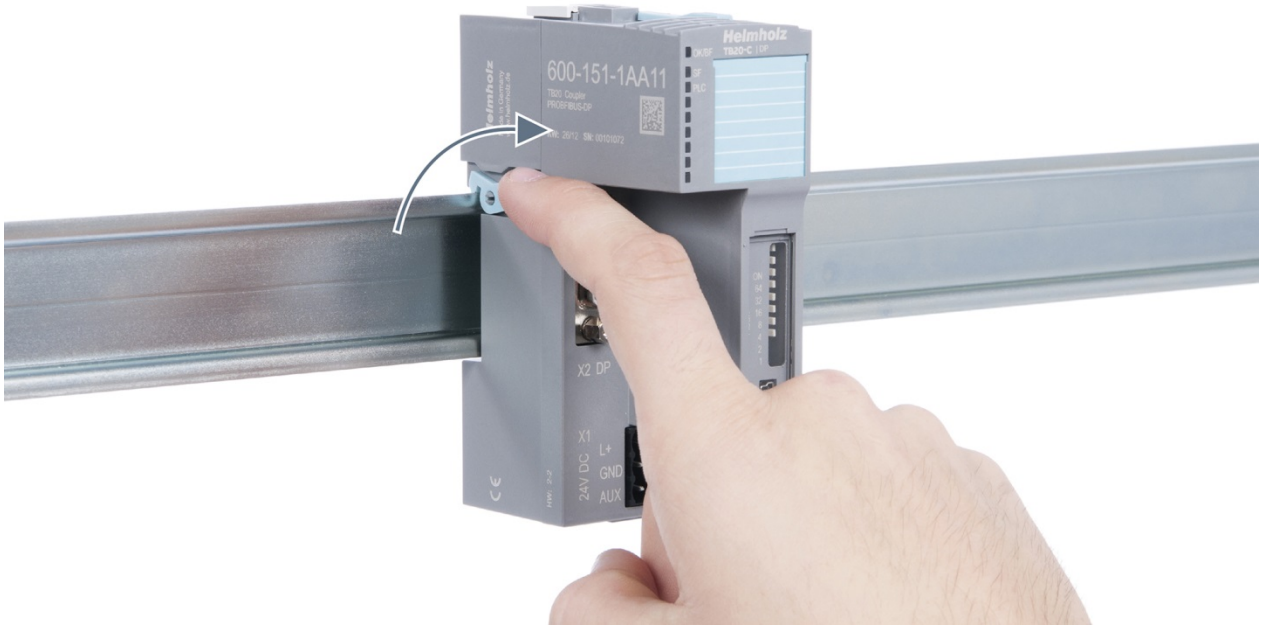
Verriegeln Sie den Koppler an der Hutschiene mit dem Verriegelungshebel auf der linken Seite des Kopplers.



3.5.2 Demontage

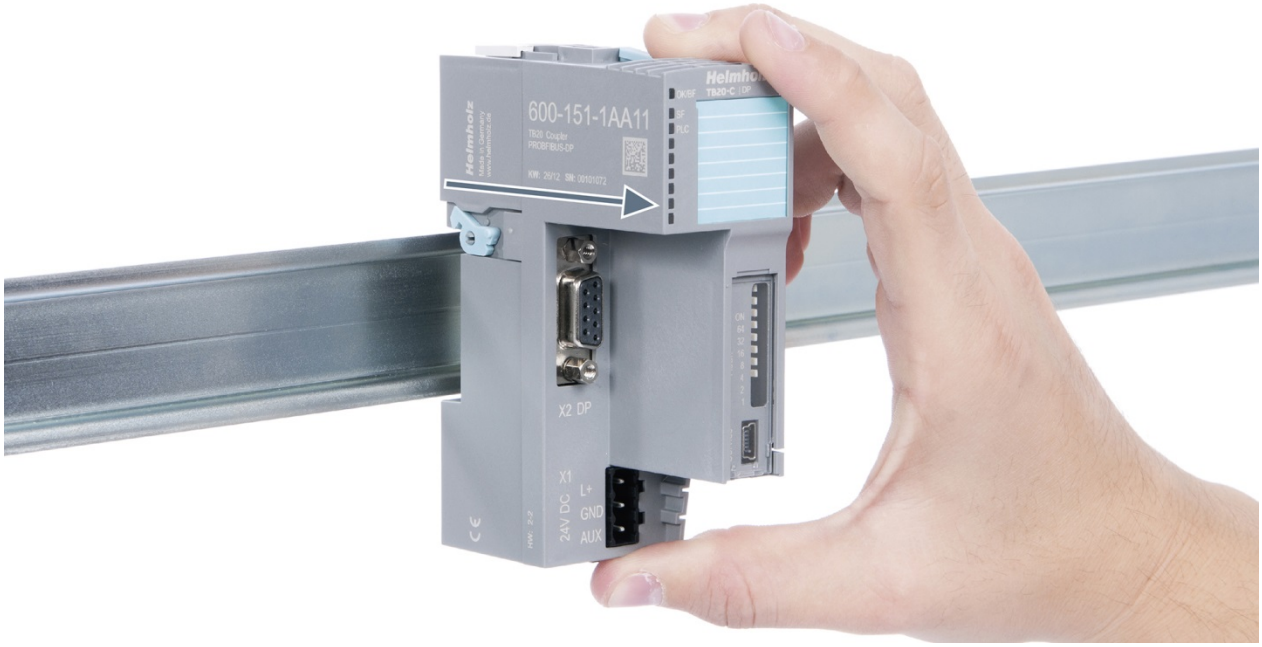
Schritt 1: Verriegelung lösen

Lösen Sie den Verriegelungshebel auf der linken Seite des Kopplers.



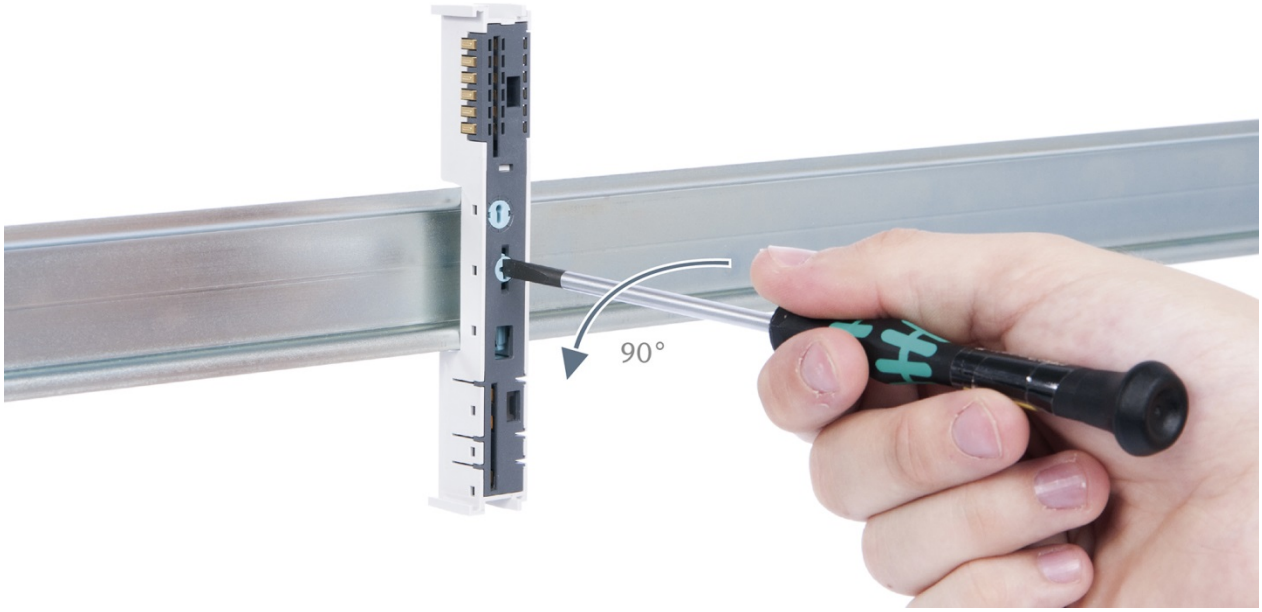
Schritt 2: Koppler abziehen

Drücken Sie mit dem Zeigefinger von oben auf den Hebel und ziehen Sie bei gedrücktem Hebel den Koppler mit Daumen und Mittelfinger ab.



Schritt 3: Basismodul entriegeln

Entriegeln Sie das Basismodul mit einem Schraubendreher.



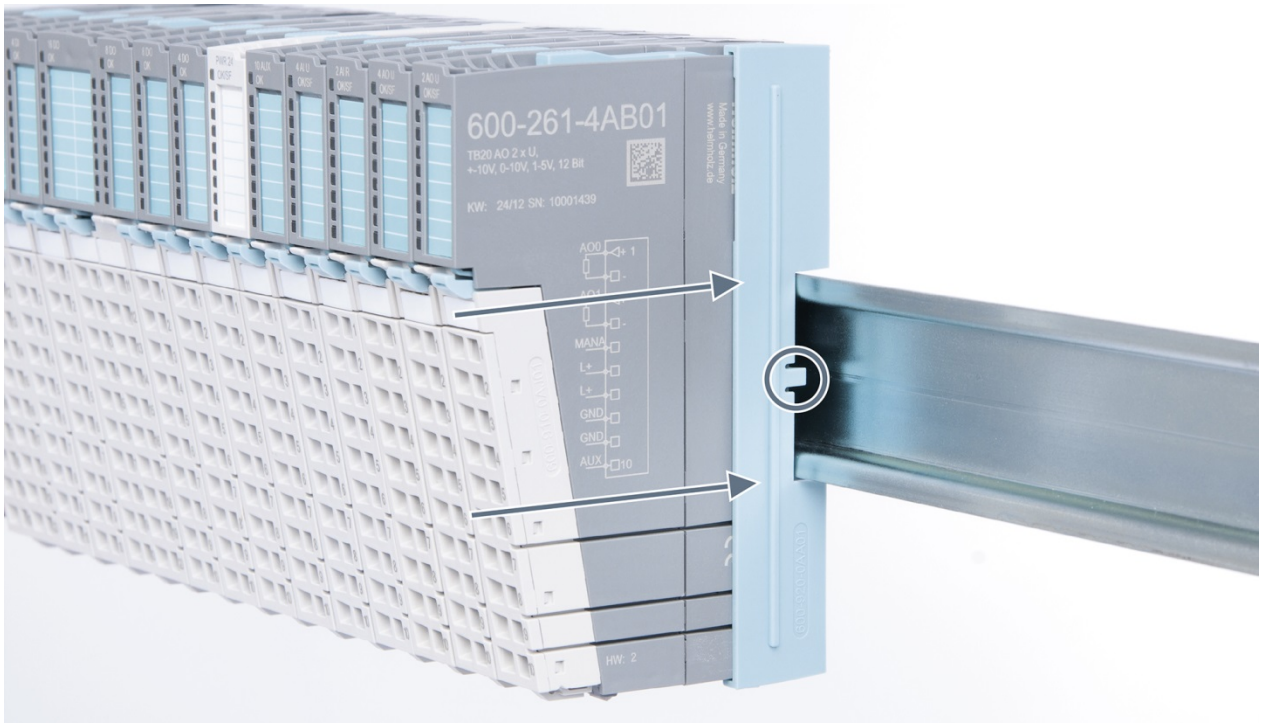
Schritt 4: Basismodul abziehen

Ziehen Sie das Basismodul nach vorne ab.

3.6 Montage und Demontage des Abschlusselements

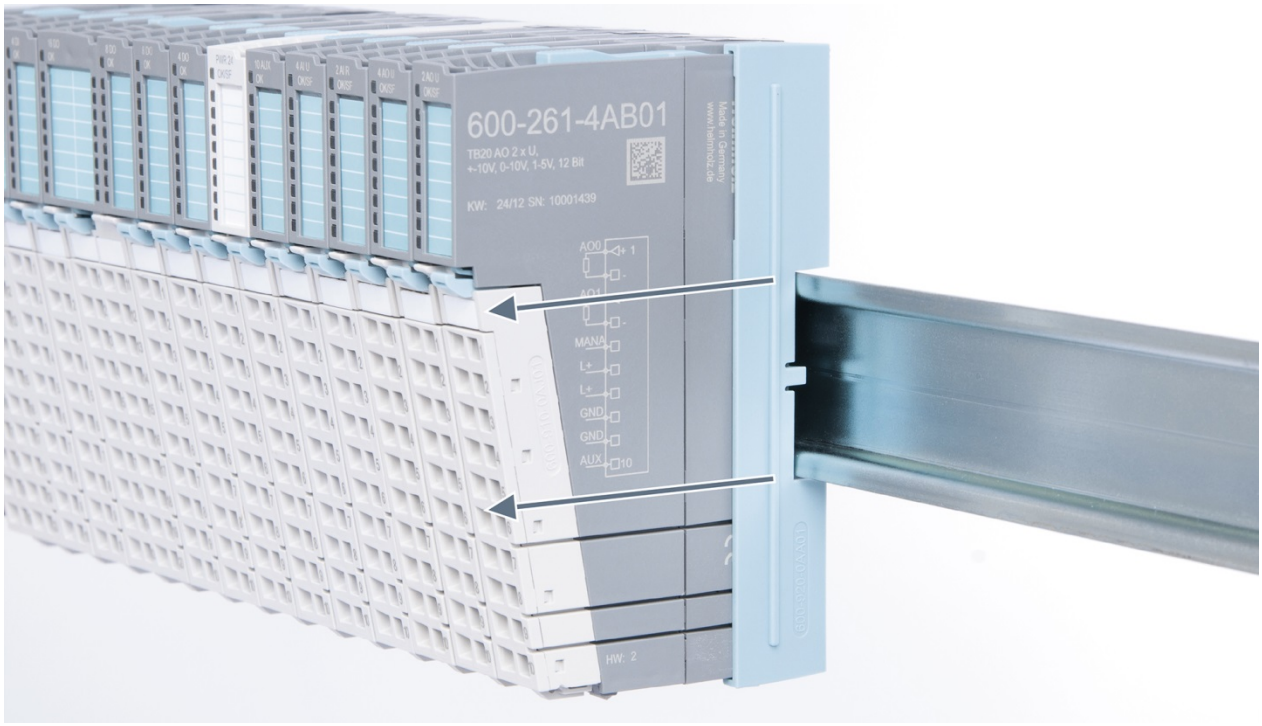
3.6.1 Montage

Schieben Sie das Abschlusselement von oben am letzten Modul über das Gehäuse nach unten, bis es die Kontakte des Basismoduls überdeckt und die Rastnase einrastet.



3.6.2 Demontage

Ziehen Sie das Abschlusselement nach oben am Modul entlang vom Modul ab.



4 Aufbau und Verdrahtung

4.1 EMV/Sicherheit/Schirmung

Das TB20 I/O-System erfüllt die EU-Richtlinie 2004/108/EG („elektromagnetische Verträglichkeit“).

Eine wirksame Schutzmaßnahme gegen störende elektromagnetische Beeinflussungen ist das Abschirmen elektrischer Leitungen und Baugruppen.

ACHTUNG

Achten Sie beim Aufbau der Anlage und bei der Verlegung der notwendigen Leitungen darauf, alle Normen, Vorschriften und Regeln bezüglich der Abschirmung genau einzuhalten.

Arbeiten Sie fachgerecht! Fehler in der Abschirmung können zu Funktionsstörungen bis hin zum Ausfall der Anlage führen.

Um die elektromagnetische Verträglichkeit (EMV) in Ihren Schaltschränken in elektrisch rauer Umgebung sicherzustellen, sind bei der Konstruktion und dem Aufbau folgende EMV-Regeln zu beachten:

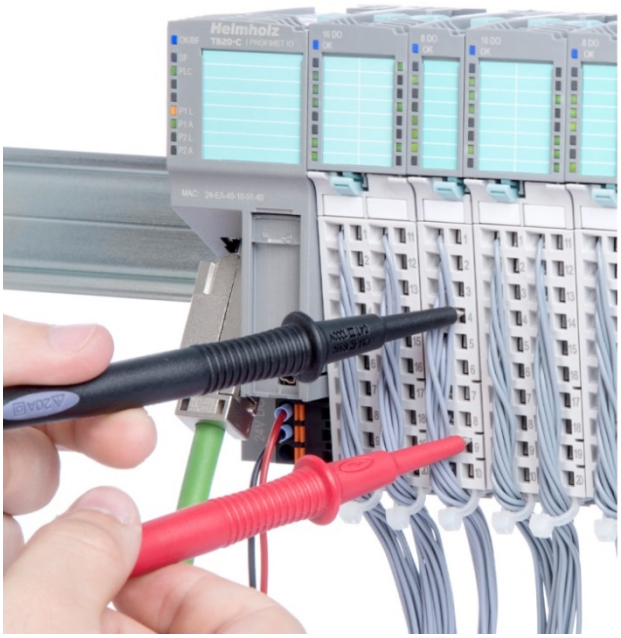
- Alle metallischen Teile des Schaltschrankes sind großflächig und gut leitend miteinander zu verbinden (nicht Lack auf Lack!). Falls nötig Kontakt- oder Kratzscheiben verwenden.
- Die Schranktür ist über die Massebänder (oben, mittig, unten) möglichst kurz mit dem Schrank zu verbinden.
- Signalleitungen und Leistungskabel sind räumlich getrennt mit einem Mindestabstand von 20 cm voneinander zu verlegen, um Koppelstrecken zu vermeiden.
- Signalleitungen möglichst nur von einer Ebene in den Schrank führen.
- Ungeschirmte Leitungen des gleichen Stromkreises (Hin- und Rückleiter) sind möglichst zu verdrillen.
- Schütze, Relais und Magnetventile im Schrank, gegebenenfalls in Nachbarschränken, sind mit Löschkombinationen zu beschalten, z. B. mit RC-Gliedern, Varistoren, Dioden.
- Verdrahtungen nicht frei im Schrank verlegen, sondern möglichst dicht am Schrankgehäuse bzw. an Montageblechen führen. Dies gilt auch für Reservekabel. Diese müssen mindestens an einem Ende auf Erde liegen, besser an beiden Enden (zusätzliche Schirmwirkung).
- Unnötige Leitungslängen sind zu vermeiden. Koppelkapazitäten und -induktivitäten werden dadurch klein gehalten.
- Analoge Signalleitungen und Datenleitungen müssen geschirmt werden.

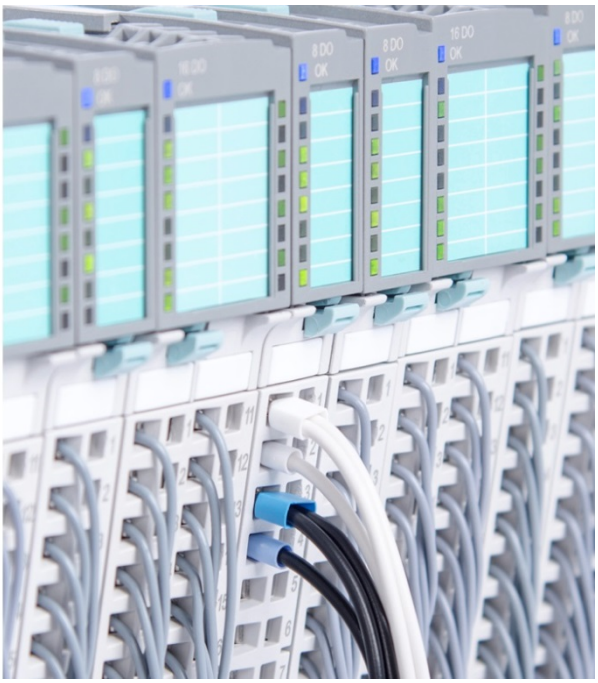
4.2 Frontstecker

Die Federzugklemmen der Frontstecker sind für einen Kabeldurchschnitt bis 1,5 mm² (AWG 16-22) mit und ohne Aderendhülsen geeignet.

Der Anschluss von beispielsweise 2 x 0,75 mm² Leitungen in einer Federzugklemme ist ebenfalls möglich, solange der maximale Leitungsquerschnitt von 1,5 mm² pro Klemme nicht überschritten wird.

Die Kabel können an der Unterseite des Frontsteckers mit einem Kabelbinder befestigt werden.





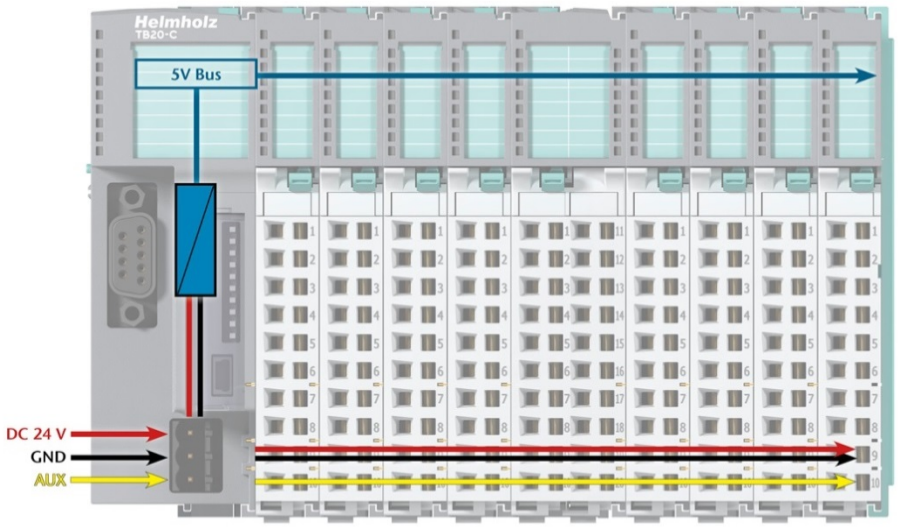
4.3 Verdrahten des Kopplers

Im Buskoppler ist ein Netzteil integriert. Das Netzteil versorgt die angereihten Peripheriemodule. Das Netzteil selbst wird über den dreipoligen Anschluss auf der Vorderseite (24 V, GND, AUX) versorgt.

Über den 24 V-Anschluss werden zwei Busse versorgt:

- Der Leistungs-Bus für die Leistungsversorgung der I/O-Ebene (DC 24 V, GND, AUX).
- Der Kommunikations-Bus für die Leistungsversorgung der Elektronik in den Peripheriemodulen.

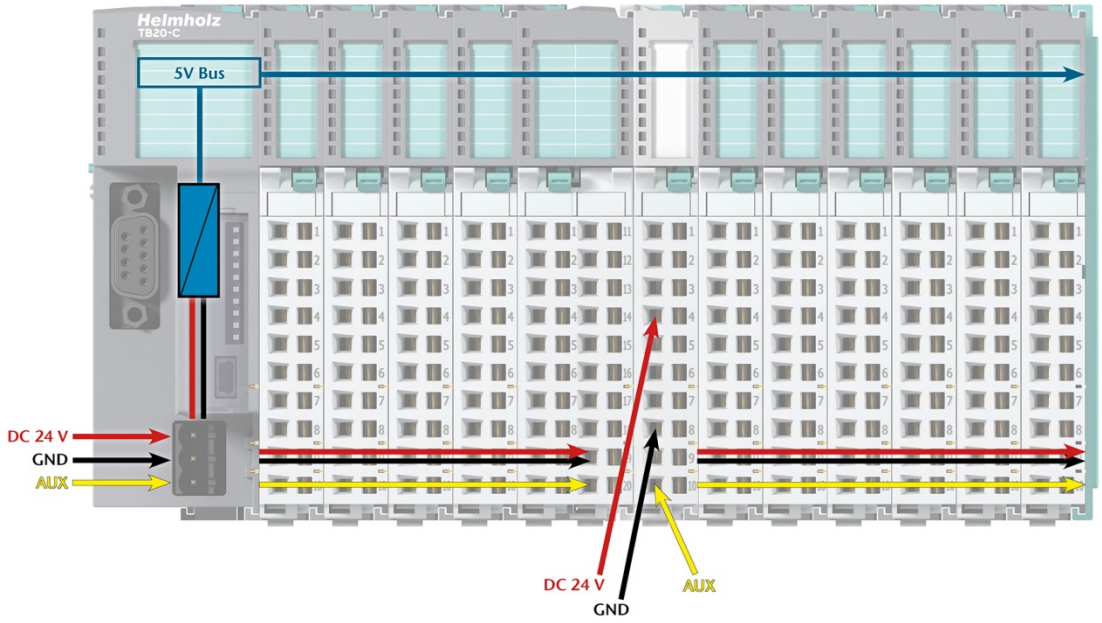
Über den AUX-Anschluss kann ein weiteres Spannungs-Potenzial angeschlossen und benutzt werden. Jedes Peripheriemodul besitzt einen AUX-Anschluss im Frontstecker auf dem untersten Pin (Pin 10 bzw. Pin 20).



Die Schirmung/Erdung des Kopplers und der Module erfolgt über den Schirmkontakt zur Hutschiene. Die Hutschiene muss geerdet sein. Die Oberfläche der Hutschiene muss sauber und elektrisch gut leitend sein.

4.4 Verwendung von Einspeise-/Trennmodulen

Mit Einspeise-/Trennmodulen lässt sich die Spannungsversorgung für die externen Signale (24 V, GND, AUX) in einzelne Versorgungsabschnitte aufteilen, die separat gespeist werden.

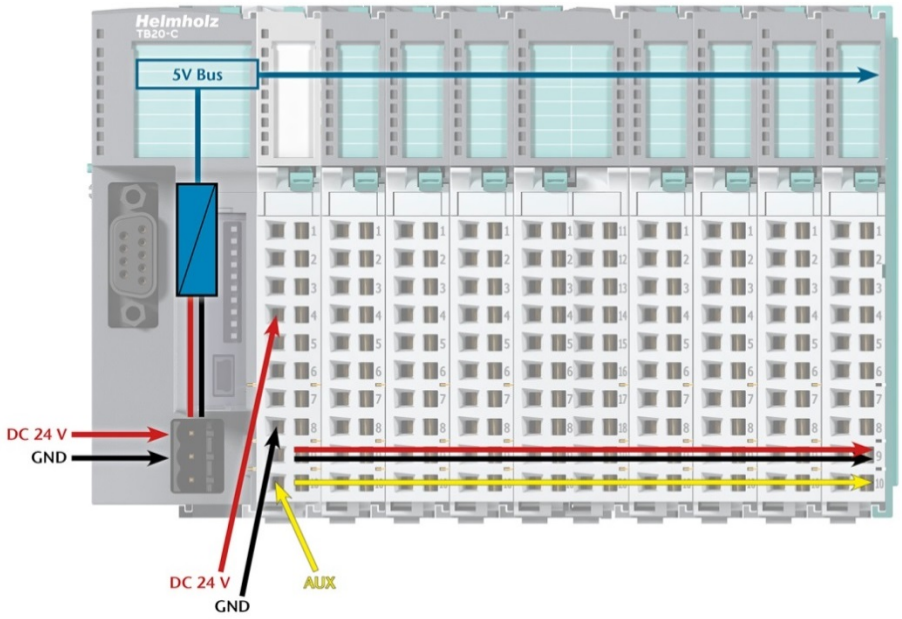


Das Einspeise-/Trennmodul für 24 V Signale hat die Bestellnummer 600-710-0AA01.

Sein Elektronikmodul und sein Basismodul sind hellgrau wie der Fronstecker. Dadurch hebt sich das Einspeise-/Trennmodul im System optisch ab und der Versorgungsabschnitt ist leicht erkennbar.

4.5 Getrennte Spannungsversorgung für Koppler und E/A-Ebene

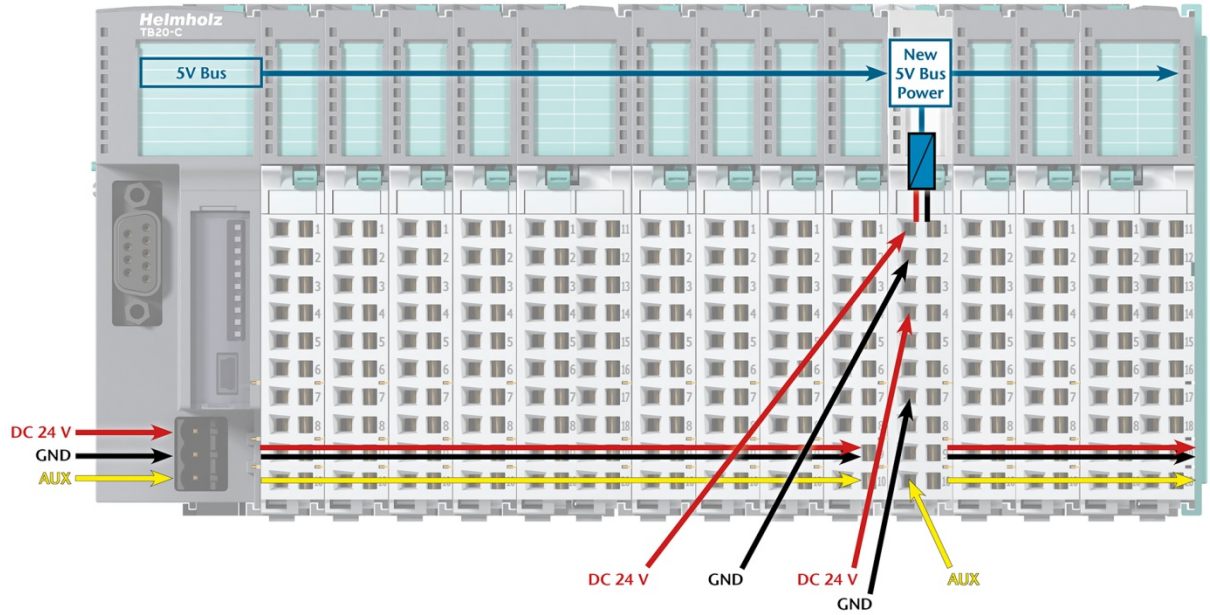
Wenn die Spannungsversorgung des Kopplers getrennt von der Spannungsversorgung der E/A-Ebene geschaltet werden soll, dann kann das Einspeise-/Trennmodul direkt hinter dem Koppler eingesetzt werden.



4.6 Verwendung von Powermodulen

Powermodule liefern die gesamte Stromversorgung für die angereihten Peripheriemodule, gegebenenfalls bis zum nächsten Power- oder Einspeise-/Trennmodul. Der Einsatz von Powermodulen ist immer dann erforderlich, wenn die Stromversorgung durch den Koppler allein nicht ausreicht, wenn also viele Module am Bus angeschlossen sind. Zur Berechnung der Stromaufnahme kann die Parametrier- und Diagnosesoftware *TB20-ToolBox* verwendet werden.

An den Anschlüssen auf der Vorderseite werden DC 24 V, GND und AUX eingespeist, wohingegen die Versorgung der angereihten Module über das Bussystem der Basismodule läuft.



Das Powermodul hat die Bestellnummer 600-700-0AA01. Das Elektronikmodul des Powermoduls ist hellgrau wie der Frontstecker. Das Basismodul des Powermoduls ist hellgrau mit dunklem Oberteil.

4.7 Elektronisches Typenschild

Im elektronischen Typenschild der TB20-Module sind alle wichtigen Informationen des Moduls gespeichert, wie z.B. Modulkennung, Modultyp, Bestellnummer, eindeutige Seriennummer, Hardware-Stand, Firmware-Stand und interner Funktionsumfang.

Diese Informationen können u.a. mit der Konfigurations- und Diagnosesoftware *TB20-ToolBox* ausgelesen werden. Mit dem elektronischen Typenschild können Konfigurationsfehler bei der Inbetriebnahme vermieden und die Wartung erleichtert werden.

4.8 Absicherung

Die Stromversorgung des TB20-Kopplers oder der Power- und Einspeisemodule ist extern mit einer Sicherung maximal 8 A (träge), entsprechend dem benötigten Maximalstrom, abzusichern.

5 Eigenschaften des TB20 EtherCAT® Buskopplers

Der TB20 EtherCAT® Buskoppler hat folgende Eigenschaften:

- EtherCAT-Slave mit Modular Device Profile (MDP)
- 2 EtherCAT-Ports zum Anschluss an das EtherCAT-Netzwerk
- CoE-Objektverzeichnis mit Unterstützung von SDO-Info und Complete-Access
- bis zu 64 TB20-Module
- maximal 1024 Byte Prozesseingangs- und 1024 Byte Prozessausgangsdaten
- automatische Prozessdatenkonfiguration
- Modultausch im Betrieb möglich (Hot-Swap)
- zyklusaktuelle Moduldiagnose
- projektierbar in jedem EtherCAT-Konfigurations-Tool durch bereitgestellte ESI-Datei
- komfortable Parametrierungs- und Diagnosemöglichkeiten über die kostenfreie PC-Software TB20-Toolbox (Anschluss via USB)
- Modulkonfiguration speicherbar
- (Um-)Parametrierung der Module über SDO-Kommunikation möglich
- Zustandsinformationen/Diagnose über 6 LEDs
- Spannungsversorgung DC 24 V
- Netzteil zur Versorgung der Peripheriemodule integriert (2,5 A)
- Einspeisung der E/A-Spannungsversorgung (DC 24 V)

Die Distributed-Clocks-Funktionalität von EtherCAT® wird zurzeit nicht unterstützt.

6 Einführung zu EtherCAT®

EtherCAT® ist die Abkürzung für *Ethernet for Control Automation Technology* und steht für eine Ethernet-basierte, echtzeitfähige Feldbus-Technologie. Das EtherCAT-Protokoll ist offengelegt und in IEC 61158 standardisiert.

Nachfolgend werden die wesentlichen Grundlagen zu EtherCAT® beschrieben, die für ein Verständnis der Funktionsweise des TB20 EtherCAT® Buskopplers hilfreich sind. Für detailliertere Informationen sei auf die Internetseite www.ethercat.org der *EtherCAT Technology Group* verwiesen.

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

6.1 Netzwerk und Topologie

Ein EtherCAT-Netzwerk besteht aus einem Master und bis zu 65535 Slaves. Der Master ist in der Regel Bestandteil einer Speicherprogrammierbaren Steuerung (SPS) und verfügt über einen bzw. bei Kabelredundanz über zwei EtherCAT-Ports zum Anschluss der Slaves. Bei den Slaves handelt es sich um Feldgeräte, also z.B. Sensoren und Aktoren. Auch beim TB20 EtherCAT® Buskoppler handelt es sich um einen EtherCAT-Slave.

Die meisten Slaves verfügen über zwei EtherCAT-Ports: Einen mit *IN* bezeichneten EtherCAT-Port zum Anschluss aus Richtung des Masters und einen mit *OUT* bezeichneten EtherCAT-Port zum Anschluss nachfolgender Slaves. Standardmäßig ergibt sich so eine Linien-Topologie aus in Serie verbundener Slaves. Durch den Einsatz von Slaves mit mehr als zwei EtherCAT-Ports oder entsprechender Port-Erweiterungen lassen sich auch Baum- oder Stern-Topologien realisieren. Switches oder andere aktive Infrasturkturkomponenten werden hierfür nicht benötigt.

EtherCAT unterstützt außerdem das Anschließen und Entfernen einzelner Slaves oder ganzer Netzwerksegmente im laufenden Betrieb, was als 'Hot Connect' bezeichnet wird. Die betreffenden Slaves müssen dazu mit Hilfe des EtherCAT-Konfigurationstools sogenannten Hot-Connect-Gruppen zugeordnet werden.

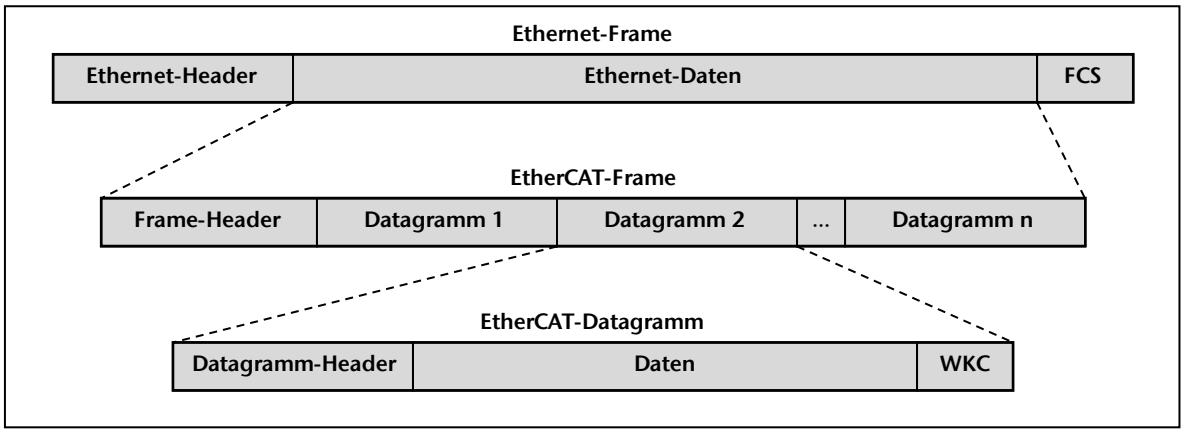
6.2 Kommunikationsprinzip

Die Kommunikation bei EtherCAT basiert auf Ethernet. Zum Datenaustausch werden Standard-Ethernet-Frames verwendet, die allerdings nur vom Master ausgesendet und von den Slaves auf eine besondere Art und Weise weitergeleitet und verarbeitet werden:

Ein vom Master ausgesendetes Ethernet-Frame wird von den Slaves so weitergeleitet, dass es alle zu einem Netzwerk zusammengeschlossenen Slaves durchläuft und abschließend wieder zum Master zurückkehrt. Dabei entnimmt sich jeder Slave bereits während des Frame-Durchlaufs die an ihn adressierten Daten und fügt seine vom Master angeforderten Daten ein. Dadurch ist es dem Master z.B. möglich, einen E/A-Datenaustausch mit einer Vielzahl von Slaves mit der Aussendung eines einzigen Ethernet-Frames durchzuführen. Realisiert wird diese spezielle Art der Frame-Verarbeitung und -Weiterleitung durch den EtherCAT-Slave-Controller (ESC), der in allen Slaves enthalten ist.

6.3 Aufbau eines Ethernet-Frames

Der Aufbau eines zur EtherCAT-Kommunikation verwendeten Ethernet-Frames ist in der nachfolgenden Abbildung vereinfacht dargestellt.



Im *Ethernet-Header* sind standardmäßig die MAC-Adressen von Sender und Empfänger sowie der EtherType des Ethernet-Frames enthalten. Für die Slave-Adressierung (↗ 6.4) spielen diese MAC-Adressen allerdings keine Rolle. Durch den für EtherCAT reservierten EtherType 0x88A4 wird angezeigt, dass in den *Ethernet-Daten* ein EtherCAT-Frame enthalten ist.

Ein *EtherCAT-Frame* besteht aus einem Frame-Header mit Längen- und Protokollinformationen sowie ein oder mehreren *EtherCAT-Datagrammen*. Diese Datagramme werden vom Master für den Datenaustausch mit den Slaves in das EtherCAT-Frame eingefügt. Der *Datagramm-Header* legt jeweils fest, welcher bzw. welche Slaves durch das Datagramm adressiert werden und auf welche Slave-Daten wie zugegriffen werden soll. Der Zugriff kann schreibend, lesend oder schreibend & lesend erfolgen.

Je nach Zugriffsart wird der für einen adressierten Slave relevante Datenanteil innerhalb des Datagramms übernommen und/oder überschrieben. Bei erfolgreicher Durchführung des Zugriffs wird der am Ende des Datagramms befindliche *Working-Counter* (WKC) durch den adressierten Slave erhöht. Nach Rückkehr des Ethernet-Frames an den Master, kann dieser anhand des WKC den Verarbeitungszustand der Datagramme überwachen.

Am Ende eines jeden Ethernet-Frames befindet sich die *Frame Check Sequence* (FCS). Sie dient zur Erkennung von Übertragungsfehlern und wird sowohl vom Master als auch von den Slaves ausgewertet. Nimmt ein Slave Änderungen an den im Ethernet-Frame enthaltenen EtherCAT-Datagrammen vor, dann wertet er die FCS nicht nur aus, sondern passt sie auch entsprechend an.

6.4 Slave-Adressierung

Bei der Adressierung der Slaves durch die EtherCAT-Datagramme wird zwischen verschiedenen Arten der Geräteadressierung und der logischen Adressierung unterschieden.

6.4.1 Geräteadressierung

Bei der *Geräteadressierung* kann ein Slave anhand seiner Position im Netzwerk, anhand seiner Stations-Adresse oder durch einen Broadcast adressiert werden.

Die *positionsabhängige Geräteadressierung* findet mit Hilfe der sogenannten Auto-Increment-Address statt. Diese enthält die Position des Slaves in Form eines negativen Wertes und wird beim Frame-Durchlauf von jedem Slave automatisch inkrementiert. Der Slave, bei dem die Adresse den Wert 0 hat, ist durch das entsprechende Datagramm adressiert. Verwendet wird diese Art der Adressierung nur im Anlauf und danach sporadisch zur Detektion neu angeschlossener Slaves.

Die *Geräteadressierung über eine Stations-Adresse* findet unter Verwendung der Configured-Station-Address oder des Configured-Station-Alias statt. Die *Configured-Station-Address* wird vom Master vergeben und beim Anlauf im Slave konfiguriert. Alternativ dazu kann der *Configured-Station-Alias* zur Adressierung verwendet werden. Dabei handelt es sich um eine vom Anwender im Slave gespeicherte Adresse, die auch zur eindeutigen Identifizierung dieses Slaves verwendet werden kann. Die Geräteadressierung über eine Stations-Adresse wird vorwiegend verwendet, wenn der Master individuell auf die ESC-Register eines Slaves zuzugreifen will.

Kapitel 7.5.1 liefert weitere Informationen zum Configured-Station-Alias in Bezug auf den TB20 EtherCAT® Buskoppler.

Bei einem als *Broadcast* versendeten Datagramm sind alle Slaves adressiert. Dies wird zum Beispiel verwendet, um eine Initialisierung aller Slaves durchzuführen oder um zu überprüfen, ob sich alle Slaves im erwarteten EtherCAT-State befinden.

6.4.2 Logische Adressierung

Die logische Adressierung wird von den EtherCAT-Datagrammen bei der Prozessdatenkommunikation verwendet. Dazu werden die Prozessdaten sämtlicher Slaves in einen gemeinsamen logischen Prozessdatenspeicher abgebildet. Die Adressierung der Prozessdaten innerhalb dieses logischen Speichers erfolgt über 32 Bit breite logische Adressen.

Durch Verwendung der logischen Adressierung ist es möglich, mit nur einem Datagramm die Prozessdaten mehrerer Slaves zu adressieren, sofern sie in einem gemeinsamen Bereich des logischen Prozessdatenspeichers liegen. Für die Zusammenstellung der Slave-Prozessdaten innerhalb des logischen Prozessdatenspeichers ist das EtherCAT-Konfigurationstool zuständig. Es sorgt im Normalfall automatisch dafür, dass eine effiziente logische Prozessdaten-Adressierung möglich ist.

In den Slaves stellt der EtherCAT-Slave-Controller (ESC) einen Speicherbereich zur Verfügung, in dem die Prozessdaten des Slaves zum Austausch mit dem Master abgelegt werden. Die Adressierung innerhalb dieses Speichers findet über lokale Adressen statt.

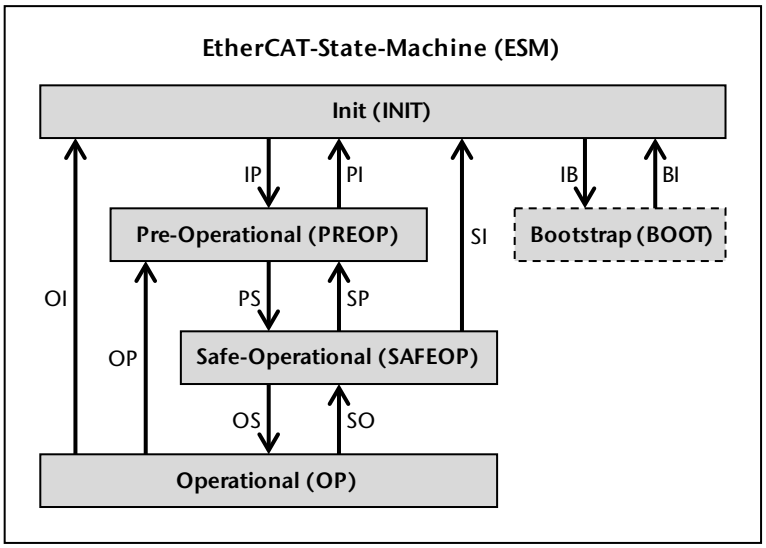
Für die Zuordnung von logischen zu lokalen Adressen beim Zugriff auf die Prozessdaten sind die *Fieldbus-Memory-Management-Units* (FMMUs) zuständig. Diese sind Bestandteil der ESCs und werden beim Anlauf automatisch vom Master konfiguriert.

6.5 EtherCAT-State-Machine (ESM)

Die EtherCAT-State-Machine (ESM) definiert das EtherCAT-spezifische Verhalten der Slaves anhand bestimmter Zustände (EtherCAT-States) und Zustandsübergänge. Dies dient zur Koordination des Zusammenspiels zwischen Master und Slave.

6.5.1 Die EtherCAT-States und deren Übergänge

In der nachfolgenden Abbildung sind die verschiedenen EtherCAT-States der ESM dargestellt. Neben der vollständigen Bezeichnung ist in Klammern zusätzlich eine Kurzbezeichnung angegeben, die in diesem Handbuch vorrangig verwendet wird. Die Pfeile kennzeichnen die möglichen Übergänge zwischen den EtherCAT-States. Sie sind mit Kürzeln versehen, die sich jeweils aus dem Anfangsbuchstaben des Ausgangs- und des Zielzustands zusammensetzen. Diese Kürzel werden von einigen EtherCAT-Konfiguratoren zur Bezeichnung der Zustandsübergänge verwendet.



INIT-State

Der INIT-State ist der Ausgangszustand der ESM. In diesem EtherCAT-State führt der Master eine Grundinitialisierung des Slaves durch, zu der u.a. die Konfiguration der azyklischen Mailbox-Kommunikation gehört.

PREOP-State

Ab dem PREOP-State kann via Mailbox-Kommunikation auf das Objektverzeichnis zugegriffen werden. Über die darin enthaltenen Objekte lassen sich anwendungsspezifische Einstellungen vornehmen und Informationen auslesen. Bevor in den SAFEOP-State gewechselt wird, wird der Slave vom Master für die Prozessdaten-Kommunikation konfiguriert.

SAFEOP-State

Ab dem SAFEOP-State beginnt der zyklische Austausch von Prozesseingangs- und Prozessausgangsdaten mit dem Master. Der Slave liefert ab jetzt aktuelle Eingangsdaten, behält seine Ausgänge aber im sicheren Zustand. Das heißt, die vom Master übermittelten Ausgangsdaten werden noch nicht übernommen.

OP-State

Mit dem OP-State ist der Zielzustand für den zyklischen E/A-Datenbetrieb des Slaves erreicht. Der Slave liefert aktuelle Eingangsdaten und nun werden auch die vom Master übermittelten Ausgangsdaten vom Slave übernommen und dessen Ausgänge entsprechend geschaltet.

BOOT-State

Der BOOT-State ist ein optionaler EtherCAT-State, der zur Übertragung einer neuen Firmware in den Slave vorgesehen ist. Vom TB20 EtherCAT® Buskoppler wird der BOOT-State nicht unterstützt, da eine Aktualisierung der Firmware via USB mit Hilfe der TB20-ToolBox (↗ 7.3.10) erfolgt.

Gesteuert wird die EtherCAT-State-Machine eines Slaves im Normalfall durch den Master, aber auch der Slave selbst kann aufgrund eines Problems einen Zustandswechsel zu einem niedrigeren EtherCAT-State auslösen. Ist Letzteres der Fall oder konnte der Slave aus bestimmten Gründen nicht in den vom Master angeforderten EtherCAT-State wechseln, dann befindet sich die EtherCAT-State-Machine in einem Fehlerzustand. Dieser wird für gewöhnlich dadurch angezeigt, dass der Bezeichnung des aktuellen EtherCAT-States das englische Wort 'ERROR' voran- oder nachgestellt wird (z.B. PREOP ERROR). Nach Auftreten eines Fehlers kann der Master erst dann wieder den Wechsel in einen neuen EtherCAT-State anfordern, wenn er den Fehler bestätigt hat. Ein Wechsel in den INIT-State ist hingegen immer möglich.

Diese allgemeine Einführung zu den EtherCAT-States und den Zustandsübergängen der EtherCAT-State-Machine wird durch das Kapitel 8 ergänzt, in dem detailliert das Verhalten des TB20 EtherCAT® Buskopplers beschrieben wird.

6.5.2 Die ESM-Register im EtherCAT-Slave-Controller

Für die Steuerung der EtherCAT-State-Machine durch den Master sowie für Rückmeldungen zu deren Zustand werden im EtherCAT-Slave-Controller die folgenden Register verwendet.

ESM-Register	ESC-Adresse	Kurzbeschreibung
AL-Control-Register (↗ 6.5.2.1)	0x0120/0x0121	enthält den vom Master angeforderten EtherCAT-State
AL-Status-Register (↗ 6.5.2.2)	0x0130/0x0131	enthält den aktuellen EtherCAT-State der ESM des Slaves
AL-Status-Code-Register (↗ 6.5.2.3)	0x0134/0x0135	enthält den aktuellen Fehlercode der ESM des Slaves

Diese Register werden hier ausnahmsweise genauer beschrieben, da sie wesentliche Aspekte zur Funktionsweise der EtherCAT-State-Machine wiederspiegeln.

6.5.2.1 Das AL-Control-Register

Das AL-Control-Register wird vom Master beschrieben, um die EtherCAT-State-Machine des Slaves zu steuern. Es enthält dazu den angeforderten EtherCAT-State sowie die Möglichkeit, einen im AL-Status-Register angezeigten Fehlerzustand zu bestätigen und damit abzulöschen. Die Anforderung eines bestimmten EtherCAT-States über das AL-Control-Register wird als *AL-Control-Request* bezeichnet.

Vom Slave kann das AL-Control-Register nur gelesen werden. Durch einen AL-Control-Request wird im Slave vom aktuellen EtherCAT-State ausgehend ein Übergang zum angeforderten EtherCAT-State ausgelöst. In Abhängigkeit von den Randbedingungen wird dieser Übergang entweder erfolgreich oder nicht erfolgreich abgeschlossen, was über das AL-Status-Register und das AL-Status-Code-Register zurückgemeldet wird.

Das 16 Bit umfassende AL-Control-Register ist wie folgt aufgebaut:

Bit	Beschreibung	Bedeutung der Werte
0 - 3	Vom Master angeforderter EtherCAT-State	1 = INIT 2 = PREOP 3 = BOOT 4 = SAFEOP 8 = OP Alle anderen Werte sind ungültig und führen dazu, dass das Fehler-Flag im AL-Status-Register gesetzt wird (↗ 8.2.22).
4	Flag zum Bestätigen/Ablöschen eines im AL-Status-Register angezeigten Fehlers	0 = Fehler wird nicht bestätigt 1 = Fehler wird bestätigt
5	Flag zum Anfordern einer Geräteidentifizierung über das AL-Status-Code-Register (Diese als 'Explicit Device Identification' bezeichnete Geräteidentifizierung wird vom TB20 EtherCAT® Buskoppler nicht unterstützt.)	0 = keine Anfrage zur Geräteidentifizierung 1 = Anfrage zur Geräteidentifizierung
6 - 15	Reserviert	sollte vom Master immer mit dem Wert 0 beschrieben werden

Die EtherCAT-State-Machine eines Slaves lässt sich zur Konfiguration/Inbetriebnahme für gewöhnlich auch aus dem verwendeten EtherCAT-Konfigurator heraus steuern. Diese Steuermöglichkeit ist nach Auswahl des Slaves meist unter der Bezeichnung 'Online' oder 'State Machine' zu finden. Dabei wird (indirekt über den Master) das AL-Control-Register im EtherCAT-Slave-Controller beschrieben.

Beim TB20 EtherCAT® Buskoppler lässt sich der aktuell über das AL-Control-Register angeforderte EtherCAT-State auch in der TB20-ToolBox anzeigen, indem während der Echtzeit-Diagnose (↗ 7.3.7) erst der Buskoppler und dann der Tab 'Info' ausgewählt werden.

6.5.2.2 Das AL-Status-Register

Das AL-Status-Register informiert über den aktuellen Zustand der EtherCAT-State-Machine des Slaves. Neben dem aktuellen EtherCAT-State enthält das Register auch ein Fehler-Flag. Dieses zeigt immer dann einen Fehler an, wenn ein Wechsel in den vom Master angeforderten EtherCAT-State nicht möglich war oder wenn der Slave aufgrund eines Problems selbst den Wechsel in einen niedrigeren EtherCAT-State ausgelöst hat. Das nachfolgend beschriebene AL-Status-Code-Register enthält dann einen Fehlercode, der einen Hinweis auf die Fehlerursache gibt.

Das Error-Flag wird bei Auftreten eines Fehlers im Zusammenhang mit der EtherCAT-State-Machine gesetzt, wobei dann im AL-Status-Code-Register ein Fehlercode als Hinweis auf die Fehlerursache enthalten ist. Das Fehler-Flag lässt sich nur durch einen AL-Control-Request mit gesetztem Fehlerbestätigungs-Flag bzw. durch Anforderung von INIT vom Master aus löschen.

Das 16 Bit umfassende AL-Status-Register ist wie folgt aufgebaut:

Bit	Beschreibung	Bedeutung der Werte
0 - 3	Aktueller EtherCAT-State des Slaves	1 = INIT 2 = PREOP 4 = SAFEOP 8 = OP
4	Fehler-Flag	0 = kein Fehler aufgetreten (AL-Status-Code = 0x0000) 1 = Fehler aufgetreten (AL-Status-Code > 0x0000)
5	Geräteidentifizierung über das AL-Status-Code-Register (Diese als 'Explicit Device Identification' bezeichnete Geräteidentifizierung wird vom TB20 EtherCAT® Buskoppler nicht unterstützt.)	0 = keine Geräteidentifizierung 1 = AL-Status-Code-Register enthält Informationen zur Geräteidentifizierung
6 - 15	Reserviert	enthält immer den Wert 0

Im EtherCAT-Konfigurationstool kann nach Auswahl des Slaves meist unter der Bezeichnung 'Online' oder 'State Machine' der aktuelle EtherCAT-State angezeigt. Dazu wird (indirekt über den Master) das AL-Status-Register ausgelesen.

Für den TB20 EtherCAT® Buskoppler lässt sich der aktuelle EtherCAT-State auch in der TB20-ToolBox anzeigen, indem während der Echtzeit-Diagnose (↗ 7.3.7) erst der Buskoppler und dann der Tab 'Info' ausgewählt werden. Ist das Error-Flag im AL-Status-Register gesetzt, wird zusätzlich 'ERROR' angezeigt.

6.5.2.3 AL-Status-Code-Register

Das AL-Status-Code-Register enthält den 16 Bit umfassenden *AL-Status-Code*, der nach Auftreten eines Fehlers einen Hinweis auf die Fehlerursache liefern soll. Der Master liest daher dieses Register immer dann, wenn im zuvor beschriebenen AL-Status-Register das Fehler-Flag gesetzt ist. Liegt aktuell kein Fehler vor, dann enthält das Register den Wert 0x0000.

Im EtherCAT-Konfigurationstool lässt sich nach Auswahl des Slaves meist unter der Bezeichnung 'Online' oder 'State Machine' dessen aktueller AL-Status-Code anzeigen. Dazu wird (indirekt über den Master) das AL-Status-Code-Register ausgelesen.

Für den TB20 EtherCAT® Buskoppler lässt sich der aktuelle AL-Status-Code auch in der TB20-ToolBox anzeigen, indem während der Echtzeit-Diagnose (↗ 7.3.7) erst der Buskoppler und dann der Tab 'Info' ausgewählt werden. Neben dem AL-Status-Code wird hier auch eine kurze Beschreibung des Fehlers angezeigt. Eine Auflistung aller vom Buskoppler im Fehlerfall verwendeten AL-Status-Codes ist in Kapitel 8.3 zu finden.

6.6 Konfiguration

Die Konfiguration des Masters und der angeschlossenen Slaves findet mit Hilfe einer Software oder Softwarekomponente statt, die hier als EtherCAT-Konfigurationstool bezeichnet wird. Auf dem Markt sind eine ganze Reihe unterschiedlicher EtherCAT-Konfigurationstools verfügbar, die abhängig vom verwendeten Master bzw. der eingesetzten Speicherprogrammierbaren Steuerung (SPS) zur Konfiguration in Frage kommen. Deshalb versucht dieses Handbuch möglichst allgemeingültige Informationen zur Konfiguration des TB20 EtherCAT® Kopplers in einem solchen Tool zu geben.

Bei EtherCAT lassen sich Slaves unterschiedlicher Hersteller zu einem Netzwerk zusammenschließen und an einem Master verwenden. Die hierfür notwendigen Informationen zur herstellerunabhängigen Konfiguration der einzelnen Slaves werden über Gerätebeschreibungsdateien (↗ 6.6.1) zur Verfügung gestellt, die EtherCAT-spezifisch als ESI-Dateien bezeichnet werden. Abhängig vom verwendeten EtherCAT-Konfigurationstool kann auch eine Konfiguration ohne Gerätebeschreibungsdatei möglich sein (↗ 6.6.2).

Die über das EtherCAT-Konfigurationstool vorgenommene Konfiguration des Masters *und* der Slaves wird abschließend im Master hinterlegt. Dies geschieht entweder implizit durch das Konfigurationstool oder explizit in Form einer XML-basierten ENI-Datei (ENI = EtherCAT Network Information). Beim Anlauf führt der Master alle bei den einzelnen Slaves erforderlichen Einstellungen auf der Grundlage der hinterlegten Konfiguration durch.

6.6.1 ESI-Dateien zur Gerätebeschreibung

Bei den ESI-Dateien (ESI = EtherCAT Slave Information) handelt es sich um XML-basierte Gerätebeschreibungsdateien, die von den jeweiligen Herstellern für die angebotenen Slaves zur Verfügung gestellt werden. Sie enthalten alle relevanten Informationen, die für eine komfortable Konfiguration der Slaves innerhalb eines EtherCAT-Konfigurationstools erforderlich sind. Die Verbindung zwischen einem Slave und dessen Gerätebeschreibung findet über die eindeutige Kombination aus Vendor-ID (Herstellerkennung) und Product-Code sowie Product-Revision statt.

Beim TB20 EtherCAT® Buskoppler handelt es sich um einen modularen EtherCAT-Slave mit CoE-Objektverzeichnis (↗ Kapitel 9). Daher sind in dessen ESI-Datei auch Beschreibungen aller verfügbaren Module sowie eine Offline-Variante des Objektverzeichnisses mit enthalten.

Um dem EtherCAT-Konfigurationstool die benötigten ESI-Dateien zur Verfügung zu stellen, müssen diese installiert oder importiert werden. Die hierzu notwendige Vorgehensweise entnehmen Sie bitte der Dokumentation des von Ihnen verwendeten Tools.

HINWEIS

Die aktuelle Version der ESI-Datei für den TB20 EtherCAT® Buskoppler kann über den Download-Bereich der Helmholz-Webseite www.helmholz.de heruntergeladen werden.

6.6.2 Konfiguration ohne ESI-Datei

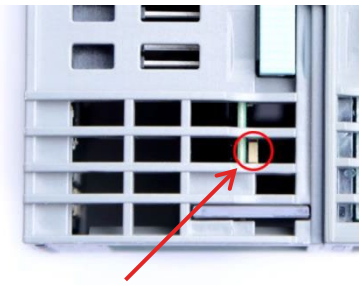
Für den Fall, dass keine ESI-Datei zur Verfügung steht, kann vom Slave selbst eine Online-Gerätebeschreibung abgerufen werden. Diese ist allerdings auf die wesentlichen, für eine grundlegende Konfiguration notwendigen Informationen reduziert. Zum Beispiel werden durch die Online-Gerätebeschreibung keine Informationen zu Modulen oder zu Objekten des CoE-Objektverzeichnisses bereitgestellt. Das macht eine wunschgemäße Konfiguration mit Modulen zwar nicht unbedingt unmöglich, aber deutlich schwieriger und unkomfortabler.

Der Abruf und die ersatzweise Verwendung der Online-Gerätebeschreibung eines Slaves werden nicht von allen EtherCAT-Konfigurationstools unterstützt. In diesem Fall ist eine Slave-Konfiguration ohne ESI-Datei nicht möglich.

Beim TB20 EtherCAT® Buskoppler lässt sich ein Teil der Einschränkungen wieder ausgleichen, indem die Auswahl und Parametrierung der Module mit Hilfe der TB20-ToolBox durchgeführt wird (↗ 7.5.6). Auch ein Zugriff auf das Objektverzeichnis des Buskopplers ist möglich, wenn das verwendete EtherCAT-Konfigurationstool das Abrufen der im Buskoppler vorhandenen Objekte über SDO-Info unterstützt. Dennoch empfehlen wir, die Konfiguration des TB20 EtherCAT® Buskopplers in einem EtherCAT-Konfigurationstool grundsätzlich unter Verwendung der ESI-Datei durchzuführen.

7 Inbetriebnahme und Verwendung

7.1 Rücksetzen auf Werkseinstellung



Der TB20 EtherCAT® Buskoppler kann über einen verborgenen Taster auf Werkseinstellung zurückgesetzt werden. Der Taster ist vom oben durch die Lüftungsöffnungen des Buskoppler-Gehäuses zu erreichen.

Zum Rücksetzen auf Werkseinstellung muss der Taster beim Einschalten der Spannungsversorgung solange gedrückt gehalten werden, bis innerhalb weniger Sekunden die oberen drei LEDs aufleuchten. Danach kann der Taster wieder losgelassen werden und der Buskoppler startet mit Werkseinstellung neu.

Nach dem Rücksetzen des Buskopplers auf Werkseinstellung

- hat der Configured-Station-Alias (↗ 7.5.1) den Wert 0 und
- sind die Parameter des Buskopplers (↗ 7.5.2) auf Standardwerte gesetzt,
- ist keine *gespeicherte Modulkonfiguration* (↗ 7.4.4.3) mehr vorhanden.

Alternativ ist ein Rücksetzen auf Werkseinstellungen auch über die TB20-ToolBox möglich (↗ 7.3.9).

7.2 Anschluss an ein EtherCAT-Netzwerk

Der TB20 EtherCAT® Buskoppler verfügt zum Anschluss an ein EtherCAT-Netzwerk (↗ 6.1) über zwei RJ45-Ports, die auf der linken Gehäuseseite mit 'X1/A IN' und 'X1/B OUT' beschriftet sind. Der obere Port 'X1/A IN' ist für den Anschluss aus Richtung des Masters vorgesehen. Zum Anschluss weiterer, nachfolgender Slaves kann der untere Port 'X1/B OUT' verwendet werden.

ACHTUNG

Achten Sie darauf, dass Sie die beiden Ports genau wie beschrieben zum Anschluss des Buskopplers an ein EtherCAT-Netzwerk verwenden. Ein vertauschter Anschluss hat eine fehlerhafte Bestimmung der Position des Buskopplers innerhalb des EtherCAT-Netzwerkes zur Folge. Ist außerdem Port 'X1/A IN' offen, kann das unter Umständen zum Totalausfall der EtherCAT-Kommunikation führen.

Die physikalische Kommunikation über die beiden Ports findet nach dem Ethernet-Standard 100BASE-TX statt. Als Verbindungsleitungen sind für diesen Fall Industrial-Ethernet-Patch-Kabel (mindestens CAT5, kein Crossover) mit einer maximalen Länge von 100 Metern vorgeschrieben.

Der aktuelle Verbindungszustand der beiden EtherCAT-Ports kann anhand der mit 'A L/A' und 'B L/A' beschrifteten LEDs (↗ 7.7.1) abgelesen werden.

7.3 Die TB20-ToolBox

Die TB20-ToolBox ist eine kostenfreie PC-Software zur Konfiguration, Inbetriebnahme und Diagnose von TB20-Systemen. Ihr Einsatz ist nicht zwingend erforderlich, um den TB20 EtherCAT® Buskoppler wunschgemäß zu konfigurieren und zu betreiben. Die TB20-ToolBox ermöglicht aber zum Beispiel eine komfortable Parametrierung des Buskopplers und der Module und kann hilfreiche Informationen zum aktuellen Systemzustand liefern.

Zur Kommunikation zwischen TB20-ToolBox und Buskoppler ist eine USB-Verbindung notwendig. Hierfür wird ein maximal 5 Meter langes Standard-USB2.0-Kabel benötigt, das zum Buskoppler hin über einen Mini-B-Stecker verfügt.

7.3.1 Download und Installation der TB20-ToolBox

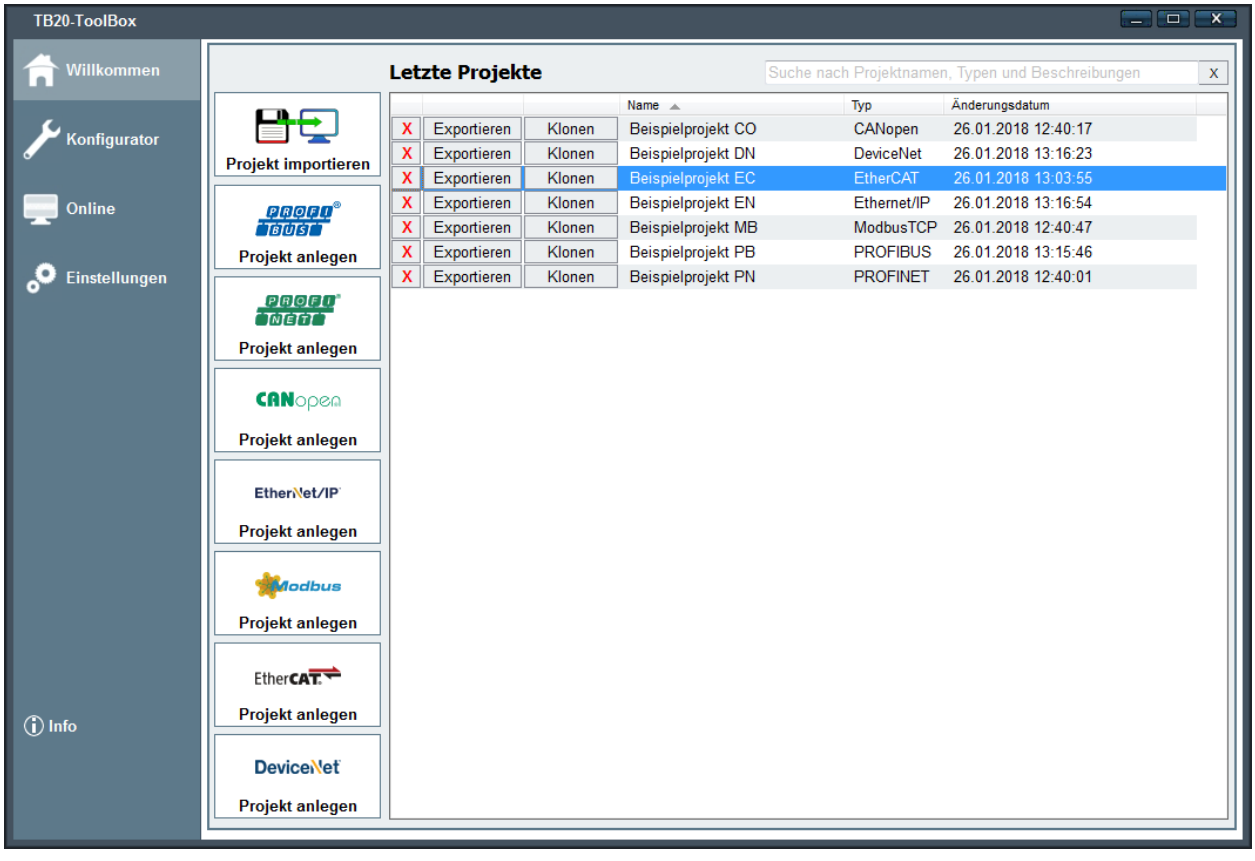
Die Software kann im Download-Bereich der Webseite www.helmholz.de kostenfrei heruntergeladen werden. Sie ist zur Verwendung auf PC-Systemen mit Windows 7 oder Windows 10 bestimmt.

Zur Installation starten Sie die heruntergeladene Setup-Datei und folgen den Anweisungen. Ist bereits eine ältere Version der TB20-ToolBox installiert, werden Sie automatisch zur Deinstallation dieser Version aufgefordert. Sämtliche bereits vorhandenen Projektdateien bleiben bei der Deinstallation erhalten.

Der für die USB-Kommunikation mit dem TB20 EtherCAT® Buskoppler benötigte Treiber wird bei der Installation der TB20-ToolBox mit auf die Festplatte kopiert. Sie finden den USB-Treiber zusammen mit Hinweisen zur Installation in der Ansicht 'Einstellungen' der gestarteten TB20-ToolBox.

7.3.2 Programmstart und Projektverwaltung

Nach dem Start der TB20-ToolBox befinden Sie sich in der Ansicht 'Willkommen'. Hier können Sie Ihre letzten gespeicherten Projekte öffnen und verwalten sowie neue Projekte anlegen und importieren.

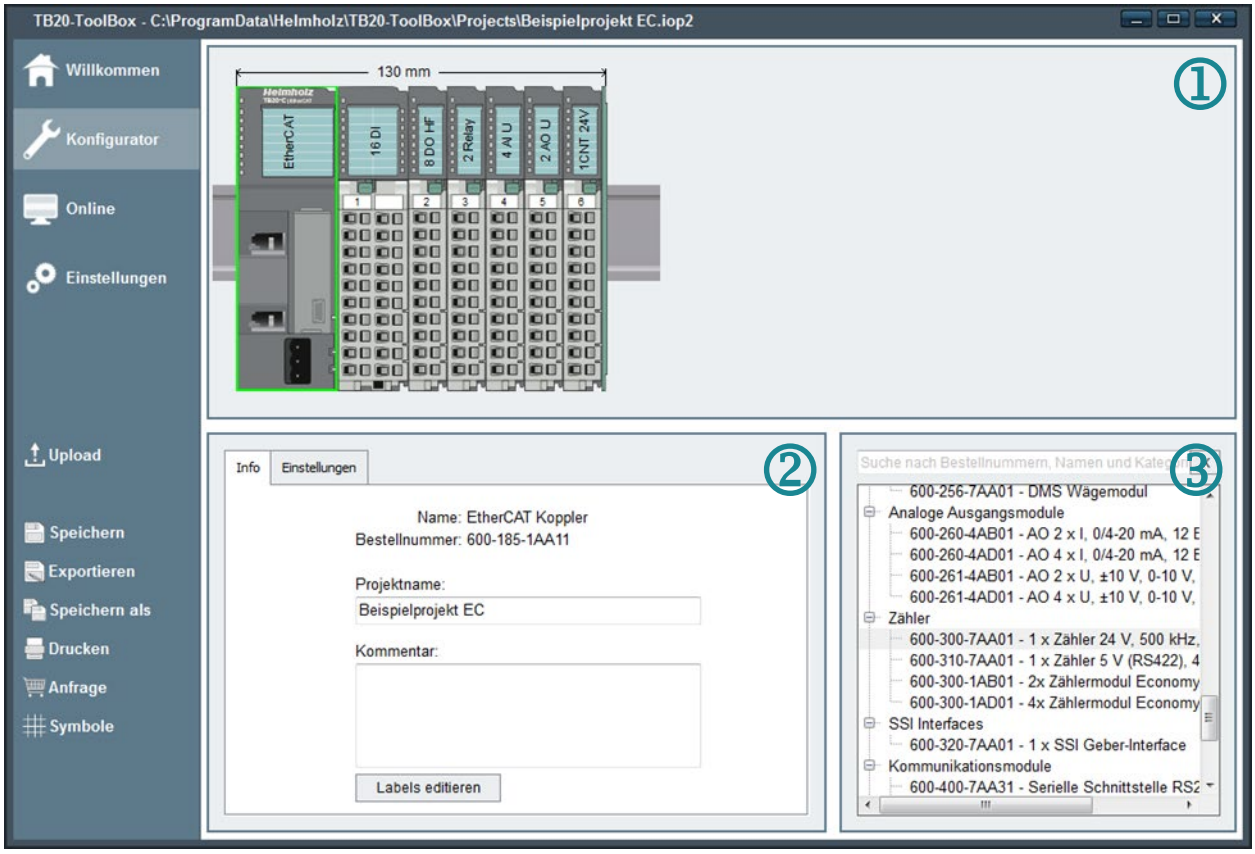


7.3.3 Anlegen eines neuen EtherCAT-Projekts

Wenn Sie sich dazu entschlossen haben, den TB20 EtherCAT® Buskopplers und dessen Module über die TB20-ToolBox zu parametrieren (↗ 7.5.6), müssen Sie zunächst ein neues EtherCAT-Projekt anlegen. Dazu klicken Sie in der Ansicht 'Willkommen' auf die Schaltfläche 'EtherCAT Projekt anlegen'. Im anschließend erscheinenden Dialog werden Sie zur Eingabe eines Namens und einer optionalen Beschreibung für das neue Projekt aufgefordert. Danach befinden Sie sich automatisch in der nachfolgend beschriebenen Konfigurator-Ansicht der TB20-ToolBox.

7.3.4 Parametrierung des Buskopplers und der Module

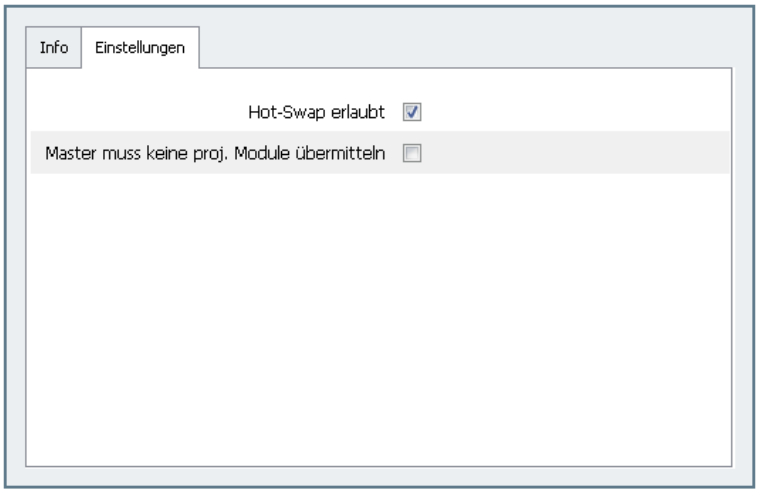
Die Parametrierung des Buskopplers und seiner Module erfolgt in der Konfigurator-Ansicht der TB20-ToolBox. Dazu müssen Sie zuvor ein EtherCAT-Projekt geöffnet oder neu angelegt haben.



Im oberen Layout-Bereich (1) ist der konfigurierte Aufbau des TB20-Systems grafisch dargestellt. Mit Hilfe der Maus können Sie dort Module verschieben, kopieren, einfügen und löschen. Über den rechts unten befindlichen Modulkatalog (3) können Sie dem Aufbau weitere Module per Doppelklick oder Drag-and-Drop hinzufügen.

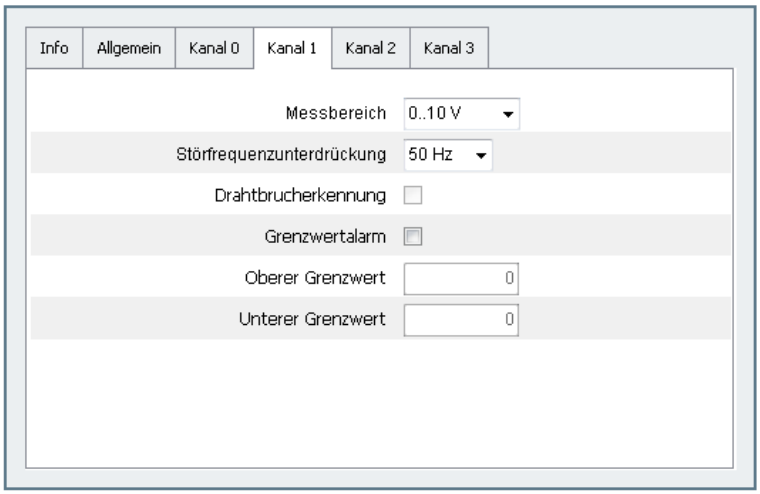
Links neben dem Modulkatalog befindet sich ein Info & Parameter-Bereich (2) mit Buskoppler- oder Modul-spezifischem Inhalt, je nachdem, was im obigen Systemaufbau aktuell ausgewählt und durch eine grüne Umrandung kenntlich gemacht ist. Neben der Anzeige von Informationen findet in diesem Bereich die Parametrierung des Buskopplers und der Module statt.

Die nachfolgende Abbildung zeigt die beiden Parameter-Einstellungen für den TB20 EtherCAT® Buskopplers, deren Bedeutung in Kapitel 7.5.2 beschrieben ist.



Bei den Modulen können Sie die Parameter-Einstellungen komfortabel über Checkboxen, Drop-down-Listen und Eingabefelder mit Wertebereichsprüfung vornehmen. Die Bedeutung der einzelnen Parameter kann in den jeweiligen TB20-Modulhandbüchern nachgeschlagen werden. Diese können Sie im Download-Bereich der Helmholz-Webseite www.helmholz.de herunterladen.

Die nächste Abbildung zeigt beispielhaft die Parameter-Einstellungen von Kanal 1 für das analoge Modul '4 AI U Iso' (600-252-7BD01).



Nachdem Sie die Parametrierung des Buskopplers und der Module abgeschlossen haben, sollten Sie die Einstellungen im aktuellen Projekt speichern. Für eine externe Sicherung oder zur Weitergabe des Projektes können Sie die Exportieren-Funktion verwenden.

7.3.5 Upload der Parametrierung in den Buskoppler

Über die Schaltfläche 'Upload' können Sie die erstellte Parametrierung in einen über USB angeschlossenen TB20 EtherCAT® Buskoppler hochladen, wo sie stromausfallsicher gespeichert wird. Nach einem automatischen Neustart ist der Buskoppler mit dieser gespeicherten Parametrierung vorkonfiguriert.

ACHTUNG

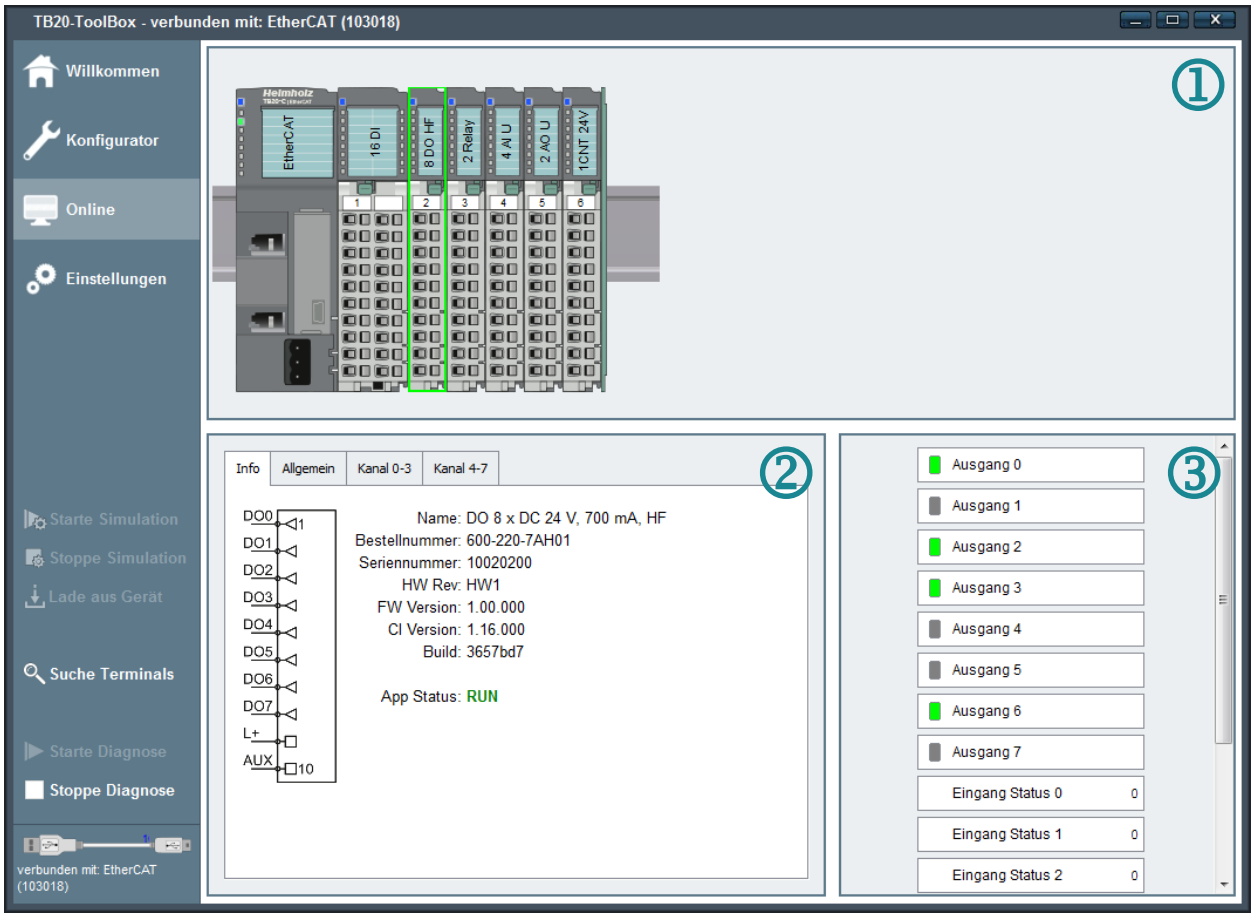
Bitte beachten Sie, dass der Buskoppler zum Hochladen der Parametrierung in den INIT-State zurückfällt. Ein zyklischer E/A-Datenaustausch mit dem Master wird abgebrochen und ist erst nach dem automatischen Neustart des Buskopplers wieder möglich.

Die von Ihnen zusammengestellten Module und deren Parameterdaten werden nach dem Hochladen der Parametrierung als *gespeicherte Modulkonfiguration* (↗ 7.4.4.3) bezeichnet. Der Buskoppler erwartet beim Anlauf mit *gespeicherter Modulkonfiguration*, dass die *gespeicherten Module* gesteckt sind und verwendet für diese automatisch die *gespeicherten Parameterdaten*.

Sie haben auch die Möglichkeit, eine Parametrierung ohne hinzugefügte und parametrierte Module in den Buskoppler hochzuladen. In diesem Fall werden die eingestellten Buskoppler-Parameter übernommen, eine *gespeicherte Modulkonfiguration* ist danach aber nicht (mehr) im Buskoppler vorhanden. Die Parametrierung der Module muss dann mit Hilfe des EtherCAT-Konfigurationstools durchgeführt werden.

7.3.6 Die Online-Ansicht

Klicken Sie auf die links befindliche Schaltfläche 'Online', um in die Online-Ansicht der TB20-ToolBox zu wechseln. Hier können Sie eine Echtzeit-Diagnose (↗ 7.3.7) des TB20-Systems durchführen oder zur Inbetriebnahme in den Simulationsbetrieb (↗ 7.3.8) wechseln. Voraussetzung hierfür ist jeweils, dass der betreffende Buskoppler über USB mit dem PC verbunden ist. Außerdem können Sie aus der Online-Ansicht heraus, den angeschlossenen Buskoppler auf Werkseinstellung zurücksetzen (↗ 7.3.9) oder ein Update der Buskoppler-Firmware (↗ 7.3.10) durchführen.



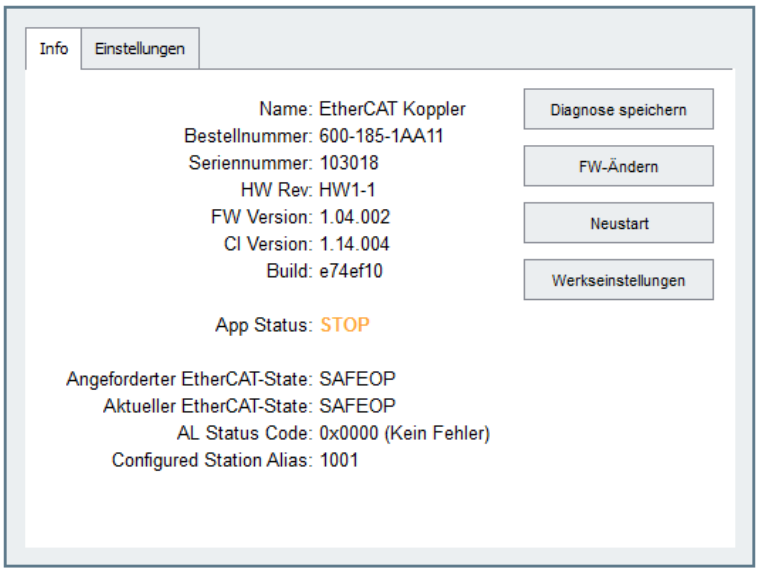
Die Online-Ansicht ist ähnlich zur Konfigurator-Ansicht in drei Bereiche unterteilt. Im oberen Layout-Bereich (1) wird der angeschlossene Buskoppler mit seinen aktuell gesteckten Modulen grafisch dargestellt. Durch Anklicken mit der Maus können Sie dort den Buskoppler oder ein Modul auswählen, was dann durch eine grüne Umrandung kenntlich gemacht wird. Zu dieser Auswahl werden im links unten befindlichen Info & Parameter-Bereich (2) entsprechende Informationen angezeigt. Rechts daneben befindet sich der E/A-Datenbereich (3), in dem abhängig vom aktuellen Zustand die Ein- und Ausgangsdaten eines ausgewählten Moduls angezeigt werden.

7.3.7 Echtzeit-Diagnose

Zur Echtzeit-Diagnose eines über USB angeschlossenen TB20 EtherCAT® Kopplers und seiner Module müssen Sie in die zuvor beschriebene Online-Ansicht wechseln. Im Layout-Bereich sehen Sie den aktuellen Aufbau des angeschlossenen TB20-Systems. Auch der aktuelle Zustand der Status-LEDs von Buskoppler und Modulen wird in dieser Ansicht wiedergegeben.

Im Info & Parameter-Bereich können Sie sich, je nach Auswahl, den aktuellen Zustand des Buskopplers oder eines Moduls sowie die aktuell verwendeten Parameter-Einstellungen anzeigen lassen. Auch wird hier eine eventuell vorliegende Diagnose-Meldung angezeigt. Das Vorliegen einer Diagnose-Meldung können Sie am Buskoppler und den Modulen anhand einer rot leuchtenden oder rot blinkenden OK/BF- bzw. OK/SF-LED erkennen.

Wie in der nachfolgenden Abbildung ersichtlich, werden im Info-Bereich für den Buskoppler auch EtherCAT-spezifische Informationen angezeigt. Dazu zählen der aktuelle und der angeforderte EtherCAT-State (↗ 6.5.1), der AL-Status-Code (↗ 6.5.2.3) mit kurzer Beschreibung sowie der Configured-Station-Alias (↗ 7.5.1).



Befindet sich der Buskoppler im SAFEOP- oder OP-State, das heißt, im zyklischen E/A-Datenaustausch mit dem Master, dann werden im E/A-Datenbereich die aktuellen Ein- und Ausgangsdaten des ausgewählten Moduls angezeigt.

7.3.8 Simulationsbetrieb

Mit dem Simulationsbetrieb können Sie die Steuerung der Ausgangsdaten der Module aus der TB20-ToolBox heraus übernehmen. Dadurch haben Sie unabhängig von einem EtherCAT-Master bzw. einer SPS die Möglichkeit, das TB20-System in Betrieb zu nehmen und dessen korrekte Funktion und Verkabelung zu überprüfen.

Zum Starten des Simulationsbetriebs müssen Sie sich in der Online-Ansicht (↗ 7.3.6) befinden, den über USB angeschlossenen Buskoppler ausgewählt haben und links auf die Schaltfläche 'Simulation

starten' klicken. Voraussetzung für einen erfolgreichen Wechsel in den Simulationsbetrieb ist, dass am Buskoppler mindestens ein Modul steckt und keine Modullücken vorhanden sind.

ACHTUNG

Bitte beachten Sie, dass der TB20 EtherCAT® Buskoppler beim Wechsel in den Simulationsbetrieb EtherCAT-seitig in den INIT-State zurückfällt. Ein zyklischer E/A-Datenaustausch mit dem EtherCAT-Master wird abgebrochen und ist erst nach Beendigung des Simulationsbetriebs wieder möglich.

Wird der Simulationsbetrieb im INIT-, PREOP- oder SAFEOP-State des Buskopplers gestartet, dann werden die momentan gesteckten Module mit ihren aktuellen Parameter-Einstellungen in den App-State RUN geschaltet. Beim Start aus dem OP-State heraus befinden sich die Module bereits im App-State RUN. In diesem Fall werden die zuletzt vom Master gesendeten Ausgangsdaten der Module lückenlos weiter verwendet.

Während des Simulationsbetriebs blinkt die OK/BF-LED des Buskopplers schnell blau und als 'App Status' wird im Info-Bereich des Buskopplers deutlich sichtbar 'SIMULATION' angezeigt. Neben den aus der Echtzeit-Diagnose (↗ 7.3.7) bekannten Möglichkeiten haben Sie jetzt zusätzlich die Möglichkeit, die Ausgangsdaten der Module zu steuern und deren Parameter-Einstellungen zu verändern. In beiden Fällen müssen Sie zur Übernahme der vorgenommenen Änderungen die Schaltfläche 'Aktivieren' drücken.

Zum Beenden des Simulationsbetriebs klicken Sie auf die Schaltfläche 'Stoppe Simulation'. Der Simulationsbetrieb wird außerdem automatisch beendet, wenn die USB-Verbindung zum Buskoppler unterbrochen wird oder wenn mindestens ein bzw. bei erlaubtem Hot-Swap (↗ 7.6) mindestens zwei Module gezogen wurden.

Nach Beenden des Simulationsbetriebs werden die gesteckten Module in Abhängigkeit von der dann *aktiven Modulkonfiguration* (↗ 7.4.4.1) in den App-State IDLE oder STOP geschaltet. Der Buskoppler befindet sich nach wie vor im INIT-State, kann nun aber wieder vom Master aus gesteuert werden.

7.3.9 Rücksetzen auf Werkseinstellung

Alternativ zu dem in Kapitel 7.1 beschriebenen Rücksetzen auf Werkseinstellungen über einen Taster, können Sie diese Funktion auch mit Hilfe der TB20-ToolBox durchführen. Schließen Sie dazu den TB20 EtherCAT® Buskoppler über USB an und wechseln Sie in die Online-Ansicht (↗ 7.3.6). Nach Auswahl des Buskopplers im Layout-Bereich steht im Info-Bereich die Schaltfläche 'Werkseinstellung' zur Verfügung, über die Sie das Rücksetzen auf Werkseinstellung auslösen können.

ACHTUNG

Bitte beachten Sie, dass der TB20 EtherCAT® Buskoppler beim Rücksetzen auf Werkseinstellung in den INIT-State zurückfällt. Ein zyklischer E/A-Datenaustausch mit dem Master wird abgebrochen und ist erst nach einem automatischen Neustart des Buskopplers wieder möglich.

7.3.10 Firmware-Update des Buskopplers

Die für ein Update der Buskoppler-Firmware erforderlichen Firmware-Dateien sind standardmäßig im Installationsumfang der TB20-ToolBox enthalten. Damit Ihnen für ein Firmware-Update die neueste verfügbare Firmware-Version zur Auswahl steht, sollten Sie zuvor die neueste Version der TB20-ToolBox herunterladen und installieren (↗ 7.3.1).

Die aktuell auf einem Buskoppler befindliche Version der Firmware können Sie sich anzeigen lassen, indem Sie den Buskoppler über USB anschließen, in die Online-Ansicht (↗ 7.3.6) der TB20-ToolBox wechseln und im oberen Layout-Bereich den Buskoppler auswählen. Die Versionsangabe wird Ihnen dann im Info-Bereich bei 'FW Version' angezeigt. Befindet sich neben dieser Versionsangabe ein oranges Symbol mit Ausrufezeichen, dann steht eine neuere Firmware-Version zur Aktualisierung des Buskopplers bereit.

Zur Durchführung eines Firmware-Updates klicken Sie im zuvor genannten Info-Bereich auf die Schaltfläche 'FW-Update'. Ist der TB20-ToolBox keine neuere Firmware-Version bekannt, dann heißt diese Schaltfläche stattdessen 'FW-Ändern'. Nach Klick auf die Schaltfläche erscheint ein Dialog zur Auswahl der gewünschten Firmware-Version. Die neueste verfügbare Firmware-Version steht an erster Stelle und ist standardmäßig vorausgewählt. Zum Bestätigen der Auswahl und Starten des Firmware-Updates klicken Sie abschließend auf 'OK'.

Während des Firmware-Updates flackert die OK/BF-LED des Buskopplers rot und in der TB20-ToolBox wird der Update-Verlauf durch einen Fortschrittsbalken angezeigt. Das Firmware-Update endet mit einem Neustart des Buskopplers. Kurz danach wird im Info-Bereich die Version der aktualisierten Buskoppler-Firmware angezeigt.

ACHTUNG

Bitte beachten Sie, dass der TB20 EtherCAT® Buskoppler beim Starten des Firmware-Updates in den INIT-State zurückfällt. Ein zyklischer E/A-Datenaustausch mit dem Master wird abgebrochen und ist erst nach Abschluss des Firmware-Updates und einem automatischen Neustart des Buskopplers wieder möglich.

7.4 Die Module

In diesem Kapitel sind wichtige Informationen zum Einsatz der Module am TB20 EtherCAT® Buskoppler enthalten.

7.4.1 Modulbezeichnungen und ihre Bedeutung

Im Zusammenhang mit dem TB20 EtherCAT® Buskoppler wird zwischen *detektierten, gespeicherten, projektierten* und *konfigurierten Modulen* unterschieden. Diese Modulbezeichnungen werden im Handbuch kursiv gesetzt, um explizit auf deren besondere Bedeutung aufmerksam zu machen.

7.4.1.1 Detektierte bzw. gesteckte Module

Als *detektierte Module* werden diejenigen Module bezeichnet, die vom Buskoppler am Rückwandbus erkannt (detektiert) wurden. In der Regel sollte es sich dabei also um die am Rückwandbus gesteckten Module handeln, weshalb die Bezeichnung *gesteckte Module* gleichbedeutend verwendet wird.

Die *detektierten Module* werden vom Buskoppler über das Objekt 0xF050 (Detected Module Ident List, ↗ 9.6.14) bereitgestellt. Ist allerdings eine *gespeicherte Modulkonfiguration* (↗ 7.4.4.3) vorhanden, dann ist dieses Objekt standardmäßig mit den *gespeicherten Modulen* überlagert.

7.4.1.2 Gespeicherte Module

Als *gespeicherte Module* werden diejenigen Module bezeichnet, die Bestandteil einer *gespeicherten Modulkonfiguration* (↗ 7.4.4.3) sind. Ist eine *gespeicherte Modulkonfiguration* im Buskoppler vorhanden, dann wird das Objekt 0xF050 (Detected Module Ident List, ↗ 9.6.9) standardmäßig mit den *gespeicherten Modulen* überlagert. Ansonsten sind in diesem Objekt die *detektierten Module* enthalten.

7.4.1.3 Projektierte Module

Als *projektierte Module* werden die im EtherCAT-Konfigurationstool für den Betrieb des Buskopplers ausgewählten Module bezeichnet. Der Begriff *'projektiert'* wurde gewählt, um diese Module von den im Buskoppler *konfigurierten Modulen* eindeutiger unterscheidbar zu machen. Im Normalfall findet vor dem Wechsel des Buskopplers nach SAFEOP ein automatischer Abgleich der *konfigurierten* mit den *projektierten Modulen* statt. Dazu wird vom Master das Objekt 0xF030 (Configured Module Ident List, ↗ 9.6.13) des Buskopplers mit den *projektierten Modulen* beschrieben.

7.4.1.4 Konfigurierte Module

Als *konfigurierte Module* werden die im Objekt 0xF030 (Configured Module Ident List, ↗ 9.6.13) enthaltenen Module bezeichnet. Die *konfigurierten Module* legen fest, mit welchen gesteckten Modulen der Buskoppler in SAFEOP und OP betrieben werden soll. Außerdem legen Sie bereits in PREOP fest, für welche Module entsprechende Modul-Objekte (↗ 9.7) in das Objektverzeichnis des Buskopplers eingeblendet werden. Über die eingeblendeten Parameterdaten-Objekte ist so eine Parametrierung der momentan *konfigurierten Module* möglich. Die *konfigurierten Module* und die zugehörigen Parameterdaten werden zusammengefasst als *aktuelle Modulkonfiguration* (↗ 7.4.4.2) des Buskopplers bezeichnet.

Beim Übergang von INIT nach PREOP wird das oben genannte Objekt 0xF030 bei Vorhandensein einer *gespeicherten Modulkonfiguration* mit den *gespeicherten Modulen*, ansonsten mit den *detektierten Modulen* initialisiert. Das heißt, im PREOP-State werden anfänglich entweder die *gespeicherten* oder die *detektierten Module* als *konfigurierte Module* verwendet. Im Normalfall wird das Objekt 0xF030 aber noch während des PREOP-States vom Master mit den *projektierten Modulen* beschrieben.

Ob die *konfigurierten Module* momentan auf den *detektierten*, den *gespeicherten* oder den *projektierten Modulen* beruhen, kann aus den Subindizes 2 bis 4 des Objektes 0x2100 (Module Configuration Status, ↗ 9.6.7) ausgelesen werden.

7.4.2 Modul-Ident-Werte

Zur Unterscheidung der verschiedenartigen Module, die am TB20 EtherCAT® Buskoppler zum Einsatz kommen können, werden bei EtherCAT eindeutige Modulkennungen verwendet, die als Modul-Ident-Werte (Module Ident Numbers) bezeichnet werden. Die Modul-Ident-Werte werden zum Beispiel im Zusammenhang mit den Objekten 0xF030 (Configured Module Ident List, ↗ 9.6.13) und 0xF050 (Detected Module Ident List, ↗ 9.6.14) des Buskopplers verwendet.

Die Modul-Ident-Werte umfassen 32 Bit und setzen sich wie folgt dargestellt zusammen.

Bit 31 - 16	Bit 15-8	Bit 7-0
TB20-Modulkennung	Betriebsart-ID	reserviert (immer 0)

Die 16 Bit breite *TB20-Modulkennung* ist eine für TB20-Module eindeutig vergebene Kennung. Sie ist in den entsprechenden Modulhandbüchern jeweils bei den 'Technischen Daten' aufgeführt.

Die 8 Bit breite *Betriebsart-ID* sorgt bei Modulen mit unterschiedlichen Betriebsarten dafür, dass jeder Betriebsart ein eigener Modul-Ident-Wert zugeordnet ist. Das ist erforderlich, weil solche Module je nach Betriebsart meist unterschiedliche Parameter- sowie Eingangs- und Ausgangsdaten bereitstellen und daher aus EtherCAT-Sicht unterschiedliche Module darstellen.

Zur Auswahl der Betriebsart ist bei den betreffenden Modulen im ersten Byte der Parameterdaten der Betriebsart-Parameter enthalten. Der Wert, der hier zur Auswahl einer bestimmten Betriebsart eingestellt werden muss, entspricht dem Wert, der als Betriebsart-ID im jeweiligen Modul-Ident-Wert verwendet wird. Bei Modulen, die keine unterschiedlichen Betriebsarten unterstützen, ist als Betriebsart-ID der Wert 0 im Modul-Ident-Wert enthalten.

7.4.3 Parameterdaten und Parametrierung

Bei einem Großteil der TB20-Module ist es möglich, das Verhalten und die Funktion über eine Parametrierung an den jeweiligen Anwendungsfall anzupassen. Sämtliche Einstellungen bezüglich einer solchen Parametrierung sind in den Parameterdaten des jeweiligen Moduls enthalten.

Einige parametrierbare Module unterstützen verschiedene Betriebsarten. Für jede Betriebsart sind in der Regel unterschiedliche Parameter definiert, sodass sie sich auch im Aufbau und Inhalt der Parameterdaten unterscheiden. Einheitlich ist nur, dass immer im ersten Byte der Parameterdaten der Betriebsart-Parameter enthalten ist. Die verschiedenen Betriebsarten eines Moduls stellen aus Sicht von EtherCAT unterschiedliche Module mit unterschiedlichen Modul-Ident-Werten (↗ 7.4.2) dar.

Die Parameterdaten der *konfigurierten Module* sind in Form von Parameterdaten-Objekten (↗ 9.7.3) im Objektverzeichnis des Buskopplers enthalten. Beim Anlauf des Buskopplers können diese Objekte vom Master mit den zuvor im EtherCAT-Konfigurationstool (↗ 6.6) parametrierten Modul-Parameterdaten beschrieben werden. Auch ein Umparametrieren der Module im laufenden Betrieb ist möglich, indem dann aus dem SPS-Programm heraus auf die Parameterdaten-Objekte schreibend zugegriffen wird.

Zur Durchführung der Modulparametrierung über die Parameterdaten-Objekte ist es in der Regel erforderlich, sich im entsprechenden Modulhandbuch detailliert über die Codierung und den Aufbau der Parameterdaten zu informieren.

Eine alternative Parametrierung der Module ohne das bei den Parameterdaten-Objekten erforderliche Detailwissen lässt sich mithilfe der TB20-ToolBox durchführen (↗ 7.5.6). Zu diesem Zweck wird der in der TB20-ToolBox enthaltene Konfigurator (↗ 7.3.4) verwendet. Hier können nach Auswahl der gewünschten Module die Einstellungen für die einzelnen Modulparameter komfortabel über Checkboxen, Dropdown-Listen und Eingabefelder mit Wertbereichsprüfung vorgenommen werden. Abschließend werden die ausgewählten Module zusammen mit den resultierenden Parameterdaten an den Buskoppler übermittelt und dort als *gespeicherte Modulkonfiguration* (↗ 7.4.4.3) hinterlegt. Eine gesonderte Parametrierung der Module im EtherCAT-Konfigurationstool ist dann nicht mehr erforderlich. Allerdings wird vorausgesetzt, dass die dort *projektierten* und im Buskoppler *konfigurierten Module* mit den *gespeicherten Modulen* übereinstimmen. Ist dies nicht der Fall, kann der Buskoppler nicht in den EtherCAT-State SAFEOP wechseln.

7.4.4 Modulkonfigurationen des Buskopplers

Eine bestimmte Zusammenstellung von Modulen und die Parameterdaten dieser Module werden zusammengefasst als *Modulkonfiguration* bezeichnet. Im Buskoppler wird zwischen einer *aktiven*, einer *aktuellen* und einer *gespeicherten Modulkonfiguration* unterschieden.

7.4.4.1 Aktive Modulkonfiguration

Durch die *aktive Modulkonfiguration* werden der App-State und die Parameterdaten der am Rückwandbus gesteckten Module festgelegt:

- Gesteckte Module, die der *aktiven Modulkonfiguration* entsprechen, befinden sich je nach aktuellem EtherCAT-State des Buskopplers im App-State STOP oder RUN. Parametrierbare Module werden außerdem mit den in der *aktiven Modulkonfiguration* enthaltenen Parameterdaten versorgt.

- Gesteckte Module, die *nicht* der *aktiven Modulkonfiguration* entsprechen, befinden sich unabhängig vom aktuellen EtherCAT-State des Buskopplers im App-State IDLE. Parametrierbare Module sind auf ihre Default-Parameter zurückgesetzt.

Bei einem beliebigen State-Übergang nach INIT wird eine vorhandene *gespeicherte Modulkonfiguration* als *aktive Modulkonfiguration* übernommen. Ist keine *gespeicherte Modulkonfiguration* vorhanden, dann enthält die *aktive Modulkonfiguration* keine Module. In diesem Fall befinden sich dann alle gesteckten Module im App-State IDLE.

Beim Übergang von PREOP nach SAFEOP wird die *aktuelle Modulkonfiguration*, sofern gültig, als *aktive Modulkonfiguration* übernommen. Diese Übernahme findet unabhängig davon statt, ob der Übergang nach SAFEOP letztendlich erfolgreich durchgeführt werden konnte oder nicht.

Andere State-Übergänge haben keine Auswirkung auf die *aktive Modulkonfiguration*. Allerdings werden Änderungen an den Parameterdaten-Objekten, die in SAFEOP oder OP durchgeführt wurden, nicht nur in die aktuelle, sondern auch in die *aktive Modulkonfiguration* übernommen und somit direkt an das entsprechende Modul übermittelt. Erfolgt der Zugriff ohne Complete-Access müssen die Änderungen aus Konsistenzgründen abschließend noch bestätigt werden. Zur Bestätigung muss der Subindex 0 des geänderten Parameterdaten-Objekts mit dem bereits in Subindex 0 enthaltenen Wert beschrieben werden. Wird das Parameterdaten-Objekt via Complete-Access verändert, muss diese Bestätigung nicht erfolgen.

7.4.4.2 Aktuelle Modulkonfiguration

Die *aktuelle Modulkonfiguration* wird durch den Inhalt der nachfolgend beschriebenen Objekte im Objektverzeichnis des Buskopplers wiedergespiegelt:

Objekt 0xF030 (Configured Module Ident List, ↗ 9.6.13)

Dieses Objekt enthält die Liste der aktuell *konfigurierten Module*. Beim Übergang von INIT nach PREOP wird es bei Vorhandensein einer *gespeicherten Modulkonfiguration* mit den *gespeicherten*, ansonsten mit den *detektierten Modulen* initialisiert. Im letzteren Fall werden Änderungen an den *detektierten Modulen* solange in dem Objekt nachgeführt, wie sich der Buskoppler in PREOP befindet und das Objekt nicht vom Master überschrieben wurde.

Für gewöhnlich wird das Objekt 0xF030 vom Master in Verbindung mit den State-Übergängen INIT→PREOP oder PREOP→SAFEOP überschrieben. Auf diesem Wege werden die vom Master erwarteten *projektierten Module* an den Buskoppler übermittelt und als *konfigurierte Module* übernommen.

In SAFEOP und OP können über dieses Objekt keine Änderungen mehr an den *konfigurierten* und nun im Betrieb befindlichen *Modulen* vorgenommen werden. Schreibzugriffe auf das Objekt sind daher in diesen EtherCAT-States nicht mehr erlaubt.

Parameterdaten-Objekte der konfigurierten, parametrierbaren Module (↗ 9.7.3)

Im Buskoppler online verfügbar sind jeweils nur die Parameterdaten-Objekte von solchen Modulen, die im Objekt 0xF030 aktuell konfiguriert sind. Im PREOP-State muss das *konfigurierte Modul* außerdem gesteckt sein oder mit einem *gespeicherten Modul* übereinstimmen.

Die Initialwerte der Parameterdaten-Objekte im PREOP-State hängen davon ab, ob das zugehörige *konfigurierte Modul* mit der *gespeicherten* oder der *aktiven Modulkonfiguration* übereinstimmt. Stimmt das Modul mit der *gespeicherten Modulkonfiguration* überein, dann wird das zugehörige Parameterdaten-Objekt mit den *gespeicherten Parameterdaten* initialisiert. Stimmt das Modul mit der *aktiven*, aber nicht mit der *gespeicherten Modulkonfiguration* überein, dann wird das zugehörige Parameterdaten-Objekt mit den *aktiven Parameterdaten* initialisiert. Gibt es keine Überstimmung des Moduls zur *gespeicherten* oder zur *aktiven Modulkonfiguration*, dann wird das zugehörige Parameterdaten-Objekt mit den aus dem Modul ausgelesenen Default-Parametern initialisiert.

Handelt es sich um ein Modul mit unterschiedlichen Betriebsarten, muss es bezüglich der *gespeicherten* oder der *aktiven Modulkonfiguration* außerdem eine Übereinstimmung mit der Betriebsart geben, damit die entsprechenden Parameterdaten als Initialwerte übernommen werden. Ist diese Übereinstimmung nicht gegeben und die konfigurierte Betriebsart stimmt auch nicht mit der Default-Betriebsart des Moduls überein, dann wird das zugehörige Parameterdaten-Objekt bis auf den enthaltenen Betriebsart-Parameter mit Nullen initialisiert.

Die online verfügbaren Parameterdaten-Objekte können vom Master oder vom EtherCAT-Konfigurationstool aus in PREOP, SAFEOP und OP beschrieben werden, um die Parameterdaten der *konfigurierten Module* zu verändern. In SAFEOP und OP durchgeführte Änderungen werden nicht nur in die *aktuelle*, sondern auch in die *aktive Modulkonfiguration* übernommen und somit direkt an das entsprechende Modul übermittelt. Erfolgt der Zugriff ohne Complete-Access müssen die Änderungen aus Konsistenzgründen abschließend noch bestätigt werden. Zur Bestätigung muss der Subindex 0 des geänderten Parameterdaten-Objekts mit dem bereits in Subindex 0 enthaltenen Wert beschrieben werden. Wird das Parameterdaten-Objekt via Complete-Access verändert, muss diese Bestätigung nicht erfolgen.

Informationen zum Zustand der *aktuellen Modulkonfiguration* können über das Objekt 0x2100 (Module Configuration Status, ↗ 9.6.7) abgerufen werden. Anhand der Subindizes 2 bis 4 ist zum Beispiel erkennbar, ob die in Objekt 0xF030 enthaltene Liste der *konfigurierten Module* auf den *detektierten*, den *gespeicherten* oder den vom Master übermittelten *projektierten Modulen* beruht. Und Subindex 5 zeigt in diesem Zusammenhang an, ob diese Liste gültig ist, weil sie mindestens ein Modul und keine Modullücken enthält.

In PREOP kann über den Objekteintrag 0x2110:04 (Command: Store current configuration, ↗ 9.6.8) die *aktuelle Modulkonfiguration* stromausfallsicher im Buskoppler gespeichert werden. Sie steht dann danach als *gespeicherte Modulkonfiguration* zur Verfügung.

Wird ein Übergang von PREOP nach SAFEOP angestoßen und die im Objekt 0xF030 enthaltene Liste der *konfigurierten Module* ist gültig, dann wird die *aktuelle Modulkonfiguration* als *aktive Modulkonfiguration* übernommen. Das heißt, alle dann mit der aktualisierten *aktiven Modulkonfiguration* übereinstimmenden Module werden in den App-State STOP geschaltet. Parametrierbare Module haben zuvor außerdem die *gespeicherten Modulparameter* erhalten. Alle anderen Module werden mit Default-Parametrierung in den App-State IDLE geschaltet.

7.4.4.3 Gespeicherte Modulkonfiguration

Als *gespeicherte Modulkonfiguration* wird eine stromausfallsicher im Buskoppler hinterlegte Modulkonfiguration bezeichnet. Analog dazu werden die darin enthaltenen Module und Parameterdaten als *gespeicherte Module* (↗ 7.4.1.2) und *gespeicherte Modulparameter* bezeichnet.

Wurde die Parametrierung des Buskopplers und der Module über die TB20-ToolBox durchgeführt (↗ 7.5.6), dann werden bei einem Upload der Parametrierung (↗ 7.3.5), die darin enthaltenen Module und deren Parameterdaten als *gespeicherte Modulkonfiguration* im Buskoppler hinterlegt. In dem Fall, dass keine Module parametriert wurden, ist nach einem Upload der Parametrierung keine *gespeicherte Modulkonfiguration* (mehr) im Buskoppler vorhanden.

Wird eine Online-Konfiguration der Module über das EtherCAT-Konfigurationstool durchgeführt, dann sind die vorgenommenen Einstellungen vorerst nur temporär in der *aktuellen Modulkonfiguration* des Buskopplers enthalten. Über den Objekteintrag 0x2110:04 (Command: Store current configuration, ↗ 9.6.8) kann zum Abschluss der Online-Konfiguration die *aktuelle Modulkonfiguration* als *gespeicherte Modulkonfiguration* im Buskoppler abgelegt werden.

Ob eine *gespeicherte Modulkonfiguration* im Buskoppler vorhanden ist, kann über den Objekteintrag 0x2100:01 (Stored configuration present) ausgelesen werden. Das Objekt 0x2100 (Module Configuration Status, ↗ 9.6.7) enthält in den anderen Subindizes noch weitere Statusinformationen bezüglich einer *gespeicherten Modulkonfiguration*.

Verhalten des Buskopplers

Ist eine *gespeicherte Modulkonfiguration* vorhanden, dann hat das folgenden Einfluss auf das Verhalten des Buskopplers und den App-State (↗ 7.4.9) der Module:

- Bei einem beliebigen State-Übergang nach INIT wird die *gespeicherte Modulkonfiguration* als *aktive Modulkonfiguration* übernommen. Das heißt, die übereinstimmend zur *gespeicherten Modulkonfiguration* gesteckten Module werden in den App-State STOP geschaltet. Parametrierbare Module haben zuvor außerdem die *gespeicherten Modulparameter* erhalten. Alle anderen Module werden mit Default-Parametrierung in den App-State IDLE geschaltet.
- Beim State-Übergang von INIT nach PREOP wird die *gespeicherte Modulkonfiguration* als *aktuelle Modulkonfiguration* übernommen. Das heißt, das Objekt 0xF030 (Configured Module Ident List, ↗ 9.6.13) wird mit den *gespeicherten Modulen* vorkonfiguriert. Außerdem werden die jeweiligen Parameterdaten-Objekte der Module (↗ 9.7.3) mit den *gespeicherten Modulparametern* initialisiert.
- Ein Übergang von PREOP nach SAFEOP ist nur möglich, wenn die *gespeicherten Module* mit den *detektierten Modulen* und, sofern vom Master übermittelt, auch mit den *projektierten Modulen* übereinstimmen. Dadurch soll verhindert werden, dass die *gespeicherten Modulparameter* unbeabsichtigt gar nicht oder nur teilweise zur Anwendung kommen.

Löschen einer gespeicherten Modulkonfiguration

Eine im Buskoppler vorhandene *gespeicherte Modulkonfiguration* kann auf drei unterschiedlichen Wegen gelöscht werden:

- Befindet sich der Buskoppler in PREOP, dann kann ein Löschen der *gespeicherten Modulkonfiguration* über den Objekteintrag 0x2110:05 (Command: Erase stored configuration, ↗ 9.6.8) erfolgen.
- Aus der TB20-ToolBox heraus kann ein Löschen der *gespeicherten Modulkonfiguration* dadurch erreicht werden, dass in den Buskoppler eine Parametrierung hochgeladen wird, die keine Module enthält (↗ 7.3.5).
- Auch durch ein Rücksetzen des Buskopplers auf Werkseinstellung (↗ 7.1) wird eine *gespeicherte Modulkonfiguration* gelöscht. Allerdings werden dann auch die Buskoppler-Parameter und der Configured-Station-Alias auf ihre Standardwerte zurückgesetzt.

7.4.5 Ein- und Ausgangsdaten

Um einen zyklusaktuellen Zugriff auf die Ein- und Ausgangsdaten der *konfigurierten Module* zu ermöglichen, sind diese immer automatisch in den Prozesseingangs- bzw. Prozessausgangsdaten des Buskopplers enthalten (↗ 7.5.4). Zusätzlich werden sie auch in Form von Eingangsdaten-Objekten (↗ 9.7.4) und Ausgangsdaten-Objekten (9.7.5) in das Objektverzeichnis des Buskopplers abgebildet. Für einen Zugriff auf die Ein- und Ausgangsdaten der Module sind diese Objekte irrelevant. Sie werden aber als Referenzobjekte für die PDO-Mapping-Objekte (↗ 9.8.1) benötigt, die in Verbindung mit den PDO-Assignment-Objekten (↗ 9.8.4) eindeutig festlegen, wie die Prozessdaten des Buskopplers zusammengesetzt sind.

Über welche Ein- und Ausgangsdaten ein Modul verfügt und wie diese aufgebaut sind, ist im jeweiligen Modulhandbuch beschrieben.

7.4.6 Modulstatus

Zu jedem *konfigurierten Modul* wird vom Buskoppler ein Modulstatus bereitgestellt, der wesentliche Informationen zum aktuellen Zustand des Moduls enthält. Um eine zyklusaktuelle Auswertung in der SPS zu ermöglichen, sind die Modulstatus der *konfigurierten Module* daher mit in den Prozesseingangsdaten (↗ 7.5.4) des Buskopplers enthalten. Außerdem ist der Modulstatus eines jeden *konfigurierten Moduls* in Form von Modulstatus-Objekten (↗ 9.7.6) in das Objektverzeichnis eingeblendet.

Der Modulstatus umfasst ein Byte und zeigt in den Bits 0 und 1 an, ob das betreffende Modul *vorhanden* (present) und *betriebsbereit* (operational) ist.

Bit 7 - 2	Bit 1	Bit 0
Reserviert immer 0	**'Operational'** 0 = *nicht betriebsbereit* 1 = *betriebsbereit*	**'Present'** 0 = *nicht vorhanden* 1 = *vorhanden*

Zu Bit 0 (Present):

Ein *konfiguriertes Modul* wird dann als *vorhanden* angezeigt, wenn im entsprechenden Steckplatz ein übereinstimmendes Modul gesteckt ist. Ist dort kein oder ein falsches Modul gesteckt, dann wird das *konfigurierte Modul* als *nicht vorhanden* angezeigt.

Ist ein Modultausch im laufenden Betrieb (Hot-Swap, ↗ 7.5.2.1) erlaubt, dann sollte das SPS-Programm eine Auswertung dieses Bits für alle Module vornehmen, um auf das Ziehen eines Moduls entsprechend reagieren zu können.

Zu Bit 1 (Operational):

Damit ein *konfiguriertes Modul* als *betriebsbereit* angezeigt wird, müssen folgende Bedingungen erfüllt sein:

- der Buskoppler muss sich im EtherCAT-State SAFEOP oder OP befinden
- das Modul muss laut Bit 0 *vorhanden* sein
- bei diagnosefähigen Modulen darf keine modulweite Diagnosemeldung vorliegen (↗ 7.4.8)

Ist eine dieser Bedingungen nicht erfüllt, dann wird das Modul als *nicht betriebsbereit* angezeigt.

Vorausgesetzt der Buskoppler befindet sich in SAFEOP oder OP und das Modul ist *vorhanden*, dann bedeutet ein nicht betriebsbereiter Zustand, dass eine modulweite Diagnosemeldung im Modul vorliegt. In diesem Fall kann aus dem Diagnosestatus (↗ 7.4.8) des Moduls die Ursache für die Diagnosemeldung ausgelesen werden.

7.4.7 Kanalstatus

Zu jedem diagnosefähigen *konfigurierten Modul* mit Eingangs- und/oder Ausgangskanälen wird vom Buskoppler ein Kanalstatus bereitgestellt, der wichtige Zustandsinformationen zu diesen Kanälen enthält. Um eine zyklusaktuelle Auswertung in der SPS zu ermöglichen, sind die Kanalstatus der Module daher mit in den Prozesseingangsdaten (↗ 7.5.4) des Buskopplers enthalten. Außerdem ist der Kanalstatus eines jeden *konfigurierten Moduls* in Form von Kanalstatus-Objekten (↗ 9.7.7) in das Objektverzeichnis eingeblendet.

Im Kanalstatus eines Moduls ist für jeden Kanal ein Bit enthalten, welches den Zustand des Kanals mit 1 = *OK* und 0 = *nicht OK* anzeigt. Diese Zustandsbits sind mit aufsteigender Kanalnummer beginnend bei Bit 0 im Kanalstatus angeordnet. Je nach Anzahl der Kanäle umfasst der Kanalstatus 1 bis maximal 4 Bytes. Nicht verwendete Bits innerhalb des Kanalstatus sind mit Nullen belegt. Nachfolgend ist der Aufbau des Kanalstatus beispielhaft für ein 4-kanaliges Modul dargestellt.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
nicht verwendet 0	**nicht verwendet** 0	**nicht verwendet** 0	**nicht verwendet** 0	**'Channel 3 OK'** 0 = nicht OK 1 = OK	**'Channel 2 OK'** 0 = nicht OK 1 = OK	**'Channel 1 OK'** 0 = nicht OK 1 = OK	**'Channel 0 OK'** 0 = nicht OK 1 = OK

Der Zustand eines Kanals wird dann als *OK* angezeigt, wenn folgende Bedingungen erfüllt sind:

- der Buskoppler befindet sich in den EtherCAT-States SAFEOP oder OP
- das Modul ist laut Modulstatus (↗ 7.4.6) *vorhanden* und *betriebsbereit*
- für den Kanal liegt keine Diagnosemeldung vor

Ist eine dieser Bedingungen nicht erfüllt, dann wird der Kanalzustand als *nicht OK* angezeigt.

Ist der Zustand eines Kanals *nicht OK*, obwohl das Modul laut Modulstatus *vorhanden* und *betriebsbereit* ist, dann liegt eine Diagnosemeldung für diesen Kanal vor. In diesem Fall kann aus dem Diagnosestatus (↗ 7.4.8) des Moduls die Ursache für die kanalbezogene Diagnosemeldung ausgelesen werden.

In der SPS sollte in Verbindung mit den Ein- und Ausgangsdaten eines diagnosefähigen Moduls immer auch der Kanalstatus ausgewertet werden. Er kann dabei als Indikator für die Gültigkeit von Eingangsdaten und die Wirksamkeit von Ausgangsdaten dienen.

7.4.8 Diagnosestatus

Zu jedem diagnosefähigen *konfigurierten Modul* wird vom Buskoppler ein Diagnosestatus bereitgestellt. Er enthält Informationen zu den im Modul anliegenden Diagnosemeldungen. Der Zugriff auf den Diagnosestatus muss über das Objektverzeichnis des Buskopplers erfolgen, wo er für die betreffenden Module in Form von Diagnosestatus-Objekten (↗ 9.7.8) enthalten ist.

Der Diagnosestatus eines Moduls setzt sich aus einer modulbezogenen Diagnose-ID sowie für jeden Kanal aus einer weiteren kanalbezogenen Diagnose-ID zusammen. Die einzelnen Diagnose-IDs belegen jeweils genau ein Byte. Nachfolgend ist der Aufbau des Diagnosestatus beispielhaft für ein 4-kanaliges, diagnosefähiges Modul dargestellt.

Byte	Inhalt des Diagnosestatus
0	modulbezogene Diagnose-ID
1	Diagnose-ID von Kanal 0
2	Diagnose-ID von Kanal 1
3	Diagnose-ID von Kanal 2
4	Diagnose-ID von Kanal 3

Den im Diagnosestatus enthaltenen Werten für die Diagnose-IDs sind bestimmte Diagnosemeldungen zugeordnet. Die nachfolgende Tabelle zeigt diese Zuordnung für eine Auswahl gängiger Diagnosemeldungen an.

Diagnose-ID	Diagnosemeldung
0	kein Fehler; es liegt keine Diagnosemeldung vor
2	Unterspannung
3	Überspannung
4	Überlast
5	Übertemperatur
6	Leitungsbruch / Drahtbruch
7	Messbereichsüberschreitung (Überlauf)
8	Messbereichsunterschreitung (Unterlauf)
17	Externe Hilfsspannung fehlt (L+)
18	Parametrierfehler (allgemein)
20	Parametrierfehler (maximale Parameterdatenlänge überschritten)

Welche Diagnosemeldungen von einem bestimmten Modul konkret unterstützt werden, können Sie dem jeweiligen Modulhandbuch entnehmen.

Das Auslesen und Auswerten des Diagnosestatus ist immer dann sinnvoll, wenn bei Vorliegen einer Diagnosemeldung genauer bestimmt werden soll, um welche Diagnosemeldung es sich handelt. Ob in einem Modul momentan eine Diagnosemeldung vorliegt, lässt sich zyklusaktuell anhand des Modulstatus (↗ 7.4.6) und des Kanalstatus (↗ 7.4.7) ermitteln.

Liegt eine modulbezogene Diagnosemeldung vor, dann ist das Modul als Ganzes, also inklusive aller Kanäle, nicht mehr oder nur noch eingeschränkt funktionsfähig. Aus diesem Grund ist dann auch in den kanalbezogenen Diagnose-IDs jeweils der Wert der modulbezogenen Diagnose-ID enthalten.

7.4.9 Betriebszustand App-State

Der Betriebszustand der Module wird als App-State bezeichnet. Dabei wird zwischen den vier App-States INIT, IDLE, STOP und RUN unterschieden. In den meisten Fällen lässt sich anhand der OK(/SF)-LED eines Moduls abgelesen, in welchem App-State sich das Modul aktuell befindet (↗ 7.7.2). Ansonsten kann der aktuelle App-State der Module auch in der TB20-ToolBox angezeigt werden, indem die Echtzeit-Diagnose (↗ 7.3.7) verwendet wird.

App-State INIT

Der App-State INIT ist ein kurzzeitiger, temporärer Zustand der nach dem Stecken des Moduls bzw. dem Einschalten der Versorgungsspannung auftritt. Er kann auch vom Buskoppler ausgelöst werden, um das Modul gezielt zurückzusetzen.

Nachdem das Modul eine interne Initialisierung abgeschlossen hat, wechselt es automatisch in den App-State IDLE. Bei parametrierbaren Modulen gehört zur internen Initialisierung auch das Laden der Default-Parameter.

App-State IDLE

Der App-State IDLE ist der Grundzustand eines Moduls. Es findet kein Austausch von E/A-Daten mit dem Buskoppler statt und die physikalischen Ausgänge des Moduls befinden sich in einem sicheren Zustand.

App-State STOP

Im App-State STOP kann ein Austausch von E/A-Daten mit dem Buskoppler stattfinden. Das Modul stellt dabei dem Buskoppler aktuelle Eingangsdaten zur Verfügung. Vom Buskoppler an das Modul übermittelte Ausgangsdaten werden allerdings ignoriert. Die physikalischen Ausgänge des Moduls befinden sich in einem sicheren oder gegebenenfalls speziell parametrierten Zustand.

App-State RUN

Im App-State RUN findet ein zyklischer Austausch von E/A-Daten mit dem Buskoppler statt. Das Modul stellt dabei dem Buskoppler aktuelle Eingangsdaten zur Verfügung. Vom Buskoppler an das Modul übermittelte Ausgangsdaten werden übernommen und bestimmen den Zustand der physikalischen Ausgänge des Moduls.

Abhängigkeit zur aktiven Modulkonfiguration und zum EtherCAT-State

Der App-State eines am Rückwandbus gesteckten Moduls ist zunächst davon abhängig, ob es mit der *aktiven Modulkonfiguration* (↗ 7.4.4.1) des Buskopplers übereinstimmt oder nicht. Gibt es keine Übereinstimmung, dann befindet sich das Modul im App-State IDLE. Gibt es eine Übereinstimmung, dann ist der App-State des Moduls zusätzlich vom aktuellen EtherCAT-State des Buskopplers abhängig: In den EtherCAT-States INIT, PREOP und SAFEOP befindet sich ein solches Modul im App-State STOP. Im EtherCAT-State OP, befindet es sich dann im App-State RUN.

7.5 Konfiguration des Buskopplers

Um den TB20 EtherCAT® Buskoppler an einem EtherCAT-Master betreiben zu können, ist zuvor eine entsprechende Konfiguration des Buskopplers innerhalb eines EtherCAT-Konfigurationstools erforderlich. Diese Konfiguration lässt sich in die folgenden Bereiche unterteilen:

- Configured-Station-Alias (↗ 7.5.1)
- Parameter des Buskopplers (↗ 7.5.2)
- Modulkonfiguration (↗ 7.5.3)
- Prozessdaten-Konfiguration (↗ 7.5.4)
- Prozessausgangsdaten-Watchdog (↗ 7.5.5)

Über die TB20-ToolBox kann eine Vorkonfiguration des Buskoppler (↗ 7.5.6) hinsichtlich seiner Parameter und der Modulkonfiguration erfolgen. Die vorkonfigurierten Einstellungen werden im Buskoppler gespeichert (↗ 7.3.5) und müssen dann nicht erneut im EtherCAT-Konfigurationstool vorgenommen werden.

7.5.1 Configured-Station-Alias

Der TB20 EtherCAT® Buskoppler unterstützt zu seiner eindeutigen Identifizierung innerhalb eines EtherCAT-Netzwerkes die Verwendung des Configured-Station-Alias. Dabei handelt es sich um einen 16-Bit-Wert, der im Buskoppler gespeichert ist und über das ESC-Register an Adresse 0x0012 vom Master ausgelesen werden kann.

Eine eindeutige Identifizierbarkeit wird vom Master benötigt, wenn der Buskoppler erstes oder einziges Mitglied einer Hot-Connect-Gruppe (↗ 6.1) ist, die im laufenden Betrieb an möglicherweise unterschiedlichen Stellen des EtherCAT-Netzwerkes an- und abgekoppelt werden kann. Ansonsten kann die Identifizierung optional dazu verwendet werden, um einen unbeabsichtigten Tausch zweier gleichartiger Slaves beim Anlauf erkennbar zu machen.

Standardmäßig ist der Configured-Station-Alias im Buskoppler auf den Wert 0 gesetzt und muss für eine Verwendung zur Identifizierung zuvor geändert werden. Zur Durchführung dieser Änderung wird das EtherCAT-Konfigurationstool verwendet. Dort lässt sich ein neuer Wert für den Configured-Station-Alias einstellen und anschließend im Buskoppler abspeichern. Der neue Wert sollte dabei so gewählt werden, dass er innerhalb des EtherCAT-Netzwerkes nur von diesem Buskoppler verwendet wird. Damit ein Abspeichern des Wertes aus dem Konfigurationstool heraus möglich ist, muss der Buskoppler über EtherCAT angeschlossen und erreichbar sein.

HINWEIS

Im Anschluss an das Abspeichern muss der Buskoppler neu gestartet werden, weil erst dann der neu gespeicherte Wert für den Configured-Station-Alias in das entsprechende ESC-Register übernommen wird.

Wie diese Konfiguration der Configured-Station-Alias bei dem von Ihnen verwendeten EtherCAT-Konfigurationstool im Detail erfolgen muss, entnehmen Sie bitte der entsprechenden Dokumentation.

HINWEIS

Beim Austausch eines vom Master eindeutig identifizierbaren Buskopplers muss darauf geachtet werden, dass im Ersatzgerät derselbe Wert für den Configured-Station-Alias vorab abgespeichert wurde.

Der aktuelle Wert des Configured-Station-Alias im Buskoppler lässt sich mithilfe der TB20-ToolBox anzeigen. Unter Verwendung der Echtzeit-Diagnose (↗ 7.3.7) wird er nach Auswahl des Buskopplers im Info-Bereich angezeigt.

7.5.2 Parameter des Buskopplers

Der Buskoppler verfügt über zwei Parameter, mit denen das Verhalten des Buskopplers im Anlauf und im laufenden Betrieb beeinflusst werden kann. Beide Parameter lassen sich über das Objektverzeichnis des Buskopplers oder alternativ mithilfe der TB20-ToolBox konfigurieren.

Bei Parametrierung über das Objektverzeichnis muss auf das Objekt 0x3000 (Coupler Parameters, ↗ 9.6.11) zugegriffen werden. Online durchgeführte Änderungen an den darin enthaltenen Parametern werden übernommen und stromausfallsicher gespeichert. Auch ist auf diese Weise eine Änderung der Parameter im laufenden Betrieb, also in den States SAFEOP und OP möglich.

Bei Verwendung der TB20-ToolBox erfolgt die Parametrierung des Buskopplers über ein dort angelegtes EtherCAT-Projekt (↗ 7.3.4). Mit dem Hochladen der Projekt-Konfiguration in den Buskoppler werden auch die eingestellten Parameter im Buskoppler gespeichert. Eine Änderung der Parameter im laufenden Betrieb ist auf diesem Weg nicht möglich.

7.5.2.1 Modultausch im laufenden Betrieb (Hot-Swap)

Mit dem hier beschriebenen Parameter lässt sich einstellen, ob ein Modultausch im laufenden Betrieb (Hot-Swap, ↗ 7.6) erlaubt ist oder nicht. In der Standardeinstellung ist ein solcher Modultausch nicht erlaubt und führt in SAFEOP und OP dazu, dass der Buskoppler nach PREOP-ERROR wechselt.

Im Objektverzeichnis ist der Parameter über den Objekteintrag 0x3000:01 (Hot-swapping allowed, ↗ 9.6.11) einstellbar. In der TB20-ToolBox hat der Parameter die Bezeichnung 'Hot-Swap erlaubt' und ist in der Konfigurator-Ansicht nach Auswahl des Buskopplers unter 'Einstellungen' zu finden.

7.5.2.2 Übermittlung projektierter Module im Anlauf

Über den hier beschriebenen Parameter kann eingestellt werden, ob ein erfolgreicher Anlauf des Buskopplers auch dann möglich sein soll, wenn der Master keine Liste mit *projektierten Modulen* übermittelt. Im Objektverzeichnis ist dieser Parameter über den Objekteintrag 0x3000:02 (Master need not provide projected modules, ↗ 9.6.11) einstellbar. In der TB20-ToolBox hat der Parameter die Bezeichnung 'Master muss keine proj. Module übermitteln' und ist in der Konfigurator-Ansicht nach Auswahl des Buskopplers unter 'Einstellungen' zu finden.

Bei den nachfolgenden Erläuterungen zu diesem Parameter wird zwischen *projektierten*, *detektierten* und *gespeicherten Modulen* unterschieden. Um welche Module es sich dabei jeweils handelt, ist in Kapitel ↗ 7.4.1 beschrieben.

Ein sinnvoller Prozessdaten-Austausch zwischen Master und Buskoppler kann nur dann stattfinden, wenn die vom Master erwarteten *projektierten Module* mit den am Rückwandbus *gesteckten detektierten Modulen* übereinstimmen. Der Buskoppler überprüft daher standardmäßig beim State-Übergang PREOP→SAFEOP (↗ 8.2.8), ob diese Übereinstimmung besteht, und lässt nur dann einen Wechsel in den SAFEOP-State inklusive Start der zyklischen Prozessdaten-Kommunikation zu.

Ist eine *gespeicherte Modulkonfiguration* vorhanden, dann muss es zusätzlich eine Übereinstimmung zwischen *projektierten* und *gespeicherten Modulen* geben. Dadurch soll verhindert werden, dass die in der *gespeicherten Modulkonfiguration* enthaltenen Modulparameterdaten unbeabsichtigt gar nicht oder nur teilweise zur Anwendung kommen.

Eine Durchführung dieser Modulprüfungen setzt voraus, dass der Master zuvor die *projektierten Module* an den Buskoppler übermittelt hat. In der Standardeinstellung des hier beschriebenen Parameters kann der Buskoppler ohne diese Übermittlung und die anschließend erfolgreiche Überprüfung der Module nicht in den SAFEOP-State wechseln. Das heißt, bei Verwendung der Standardeinstellung wird durch den Buskoppler selbst sichergestellt, dass ein zyklischer Prozessdaten-Austausch mit dem Master nur dann stattfindet, wenn *projektierte* und *detektierte* sowie ggf. *gespeicherte Module* miteinander übereinstimmen.

In der ESI-Gerätebeschreibungsdatei des Buskopplers ist als Vorgabe enthalten, dass die *projektierten Module* in Verbindung mit dem State-Übergang INIT→PREOP übermittelt werden sollen. Aufgrund dieser Vorgabe fügt das EtherCAT-Konfigurationstool automatisch einen Eintrag in die CoE-Initialisierungsliste ein. Dieser Eintrag kann abhängig vom verwendeten Tool mit 'Download Slot Configuration' oder ähnlich bezeichnet sein. Konkret definiert der Eintrag einen bei INIT→PREOP vom Master durchzuführenden Schreibzugriff auf das Objekt 0xF030 (Configured Module Ident List, ↗ 9.6.13) des Buskopplers. Mit dem Schreibzugriff auf dieses Objekt übermittelt der Master die benötigten *projektierten Module* an den Buskoppler.

Einige EtherCAT-Konfigurationstools konfigurieren den Master abweichend so, dass der Schreibzugriff auf das Objekt 0xF030 nicht bei INIT→PREOP, sondern erst bei PREOP→SAFEOP stattfindet. Das ist unproblematisch, weil die *projektierten Module* dann trotzdem rechtzeitig für eine Überprüfung zur Verfügung stehen.

Möglicherweise unterstützt das von Ihnen verwendete EtherCAT-Konfigurationstool oder der Master keine Übermittlung der *projektierten Module* auf die soeben beschriebene automatische Weise. Oder eine derartige Übermittlung ist nicht möglich, weil Sie die Konfiguration des Buskopplers ohne ESI-Datei durchführen. Nur in diesen Fällen ist es dann erforderlich, die Einstellung des Parameters so zu ändern, dass auch ohne Übermittlung der *projektierten Module* ein Wechsel des Buskopplers nach SAFEOP und ein Prozessdaten-Austausch mit dem Master möglich ist.

Werden bei geänderter Parameter-Einstellung tatsächlich keine *projektierten Module* vom Master an den Buskoppler übermittelt, dann entfallen die Überprüfungen mit den *projektierten Modulen* beim State-Übergang PREOP→SAFEOP, ohne dass dieser abgebrochen wird. Ob eine Übereinstimmung zwischen den *projektierten* und den *detektierten Modulen* besteht, kann vom Buskoppler nur noch indirekt und stark eingeschränkt bei der Konfigurationsüberprüfung der Sync-Manager 2 und 3 festgestellt werden. Stimmen hier die vom Master konfigurierten Größen der Prozesseingangs- und Prozessausgangsdaten nicht mit den vom Buskoppler erwarteten Größen überein, liegt das in der Regel daran, dass auf beiden Seiten von unterschiedlichen Modulkonstellationen ausgegangen wird. In diesem Fall wechselt der Buskoppler mit dem AL-Status-Code 0x001D (Ungültige Prozessausgangsdaten-Konfiguration für Sync-Manager 2) oder 0x001E (Ungültige Prozesseingangsdaten-Konfiguration für Sync-Manager 3) nach PREOP-ERROR.

Umgekehrt kann aufgrund übereinstimmender Prozessdatengrößen nicht verlässlich darauf geschlossen werden, dass die von Master und Buskoppler erwarteten Module übereinstimmen, weil dies auch bei unterschiedlichen Modulkonstellationen der Fall sein kann. Vor allem Unterschiede bei der Reihenfolge der Module lassen sich auf diese Weise nicht erkennen, da dadurch zwar die Anordnung der Prozessdaten, nicht aber deren Größe beeinflusst wird. Deshalb liegt es bei dieser Parameter-Einstellung in der Verantwortung des SPS-Programmierers auf geeignete Weise sicherzustellen, dass die vom Master und Buskoppler erwarteten Module übereinstimmen. Dies kann zum Beispiel aus dem SPS-Programm heraus erfolgen, in dem die im Objekt 0xF050 (Detected Module Ident List, ↗ 9.6.14) enthaltenen *detektierten Module* ausgelesen und mit den erwarten Modulen verglichen werden.

VORSICHT

Wird vom Master keine Liste mit *projektierten Modulen* an den Buskoppler übermittelt, dann kann der Buskoppler beim State-Übergang PREOP→SAFEOP selbst nicht sicherstellen, dass die vom Master erwarteten Module mit den im Buskoppler *konfigurierten* bzw. *detektierten Modulen* übereinstimmen. Deshalb muss dann auf andere Weise sichergestellt werden, dass ein Wechsel des Buskopplers nach SAFEOP und OP nur bei Überstimmung aller Module erfolgt!
Erfolgt der Wechsel nach SAFEOP und OP ohne diese Übereinstimmung, dann führt das zwangsläufig zu einer falschen Zuordnung von Eingangs- und Ausgangsdaten, wovon dann in der Regel auch übereinstimmende Module betroffen sind. Die falsche Zuordnung von Eingangs- und Ausgangsdaten hat ein ungewolltes und größtenteils unvorhersehbares Systemverhalten zur Folge, wodurch Beschädigungen an Geräten hervorgerufen werden können!

7.5.3 Modulkonfiguration

Die Konfiguration der Module des Buskopplers umfasst eine Auswahl der gewünschten Module und deren Parametrierung. Diese Konfiguration wird bei EtherCAT® üblicherweise über ein EtherCAT-Konfigurationstool (↗ 6.6) durchgeführt. Alternativ kann auch die TB20-ToolBox zur Konfiguration der Module verwendet werden (↗ 7.5.6). Vor allem die Parametrierung der Module ist über die TB20-ToolBox komfortabler möglich und erfordert außerdem kein Hintergrundwissen über den Aufbau der Parameterdaten. Eine Umparametrierung der Module im laufenden Betrieb ist auch aus dem SPS-Programm heraus möglich.

7.5.4 Prozessdaten-Konfiguration

7.5.4.1 Allgemeines zu den Prozessdaten des Buskopplers

Bei den Prozessdaten handelt es sich um Daten, die für einen direkten Zugriff aus dem SPS-Programm zur Verfügung stehen und dafür in den EtherCAT-States SAFEOP und OP zyklisch zwischen Master und Buskoppler ausgetauscht werden.

Je nach Übertragungsrichtung wird zwischen den Prozessausgangsdaten und den Prozesseingangsdaten des Buskopplers unterschieden.

Die *Prozessausgangsdaten* werden zyklisch vom Master an den Buskoppler übermittelt. Sie enthalten die Ausgangsdaten der *konfigurierten Module* und können vom SPS-Programm aus beschrieben werden.

Die Prozesseingangsdaten werden zyklisch vom Buskoppler an den Master übermittelt. Sie enthalten neben den Eingangsdaten der *konfigurierten Module* auch jeweils einen Modulstatus (↗ 7.4.6) und bei diagnosefähigen Modulen zusätzlich einen Kanalstatus (↗ 7.4.7). Auf die Prozesseingangsdaten kann aus dem SPS-Programm heraus lesend zugegriffen werden.

7.5.4.2 Zusammensetzung und Inhalt der Prozessdaten

Die Zusammensetzung und der Inhalt der Prozessdaten hängen direkt von den *konfigurierten Modulen* ab und erfolgt automatisch nach folgendem festen Schema:

- Die Prozessausgangsdaten setzen sich mit aufsteigender Steckplatzposition aus den aneinandergereihten Ausgangsdaten der *konfigurierten Module* zusammen. Module ohne Ausgangsdaten belegen in den Prozessausgangsdaten keinen Platz. In dem Fall, dass nur Module ohne Ausgangsdaten konfiguriert wurden, verfügt der Buskoppler über keine Prozessausgangsdaten.

- Die Prozesseingangsdaten setzen sich mit aufsteigender Steckplatzposition aus den aneinandergereihten Eingangsdaten sowie dem Modul- und Kanalstatus der *konfigurierten Module* zusammen. Bezogen auf ein Modul werden zuerst die ggf. vorhandenen Eingangsdaten, dann der Modulstatus und danach der ggf. vorhandene Kanalstatus aneinandergereiht.
- In den Ein- bzw. Ausgangsdaten der Module enthaltene ganzzahlige Zahlenwerte (Integer), die 2 oder 4 Bytes belegen, sind in den Prozessdaten im Little-Endian-Format enthalten. Das heißt, die einzelnen Bytes dieser Zahlenwerte werden beginnend mit dem niederwertigsten hin zum höherwertigsten Byte angeordnet.

Aus EtherCAT-Sicht setzen sich die Prozessdaten der Slaves aus sogenannten Prozessdatenobjekten (PDOs) zusammen. Der Inhalt und die Anordnung dieser PDOs wird über PDO-Mapping- und PDO-Assignment-Objekte definiert. Weitere Details hierzu sind im Kapitel 9.8 zu finden. Dort ist auch die Zusammenstellung der Prozessdaten für eine beispielhafte Modulauswahl dargestellt (↗ 9.8.4.3).

7.5.5 Prozessausgangsdaten-Watchdog

Mit dem Prozessausgangsdaten-Watchdog kann im EtherCAT-State OP die zyklische Prozessdaten-Kommunikation zwischen Master und Buskoppler hinsichtlich der Prozessausgangsdaten überwacht werden. Im Watchdog läuft zu diesem Zweck ein Timer, der mit jedem Empfang eines gültigen Prozessausgangsdaten-Datagramm neu gestartet wird. Überschreitet dieser Timer eine für den Watchdog konfigurierbare Überwachungszeit, dann führt der Buskoppler einen State-Übergang von OP nach SAFEOP-ERROR (↗ 8.2.20) durch. Auf diese Weise wird sichergestellt, dass der Buskoppler bei einem Ausfall des Masters oder bei Kommunikations- und Verkabelungsproblemen automatisch in den sicheren Zustand wechselt.

Für den Fall, dass der Buskoppler über keine Prozessausgangsdaten verfügt, weil alle *konfigurierten Module* nur Eingangsdaten bereitstellen, kann der Buskoppler keinen solchen automatischen Wechsel in den SAFEOP-State durchführen. Das ist allerdings unproblematisch, weil sich der Buskoppler wegen fehlender Modulausgänge auch im OP-State in einem sicheren Zustand befindet.

Standardmäßig ist der Prozessausgangsdaten-Watchdog des TB20 EtherCAT® Buskopplers mit einer Überwachungszeit von 100 ms aktiviert. Sollte diese Standardeinstellung für Ihren Anwendungsfall nicht passen, können Sie mithilfe des EtherCAT-Konfigurationstool auch kürzere oder längere Überwachungszeiten konfigurieren. Eine Deaktivierung des Watchdogs ist nicht zu empfehlen, aber für gewöhnlich dadurch möglich, dass für die Überwachungszeit der Wert 0 konfiguriert wird.

VORSICHT

Wurde der Prozessausgangsdaten-Watchdog deaktiviert, dann erfolgt bei Unterbrechung der Kommunikation mit dem Master kein automatischer Wechsel in den sicheren Zustand. Das kann zu einem ungewollten Verhalten des Systems und zu Beschädigung von Geräten führen.

In den verschiedenen EtherCAT-Konfigurationstools wird der Prozessausgangsdaten-Watchdog oftmals verallgemeinert als Prozessdaten-Watchdog oder auch als Sync-Manager-Watchdog bzw. SM-Watchdog bezeichnet. Wie die Konfiguration des Watchdogs im Detail erfolgen muss, entnehmen Sie bitte der Dokumentation des verwendeten EtherCAT-Konfigurationstools.

7.5.6 Vorkonfiguration über die TB20-ToolBox

Bei der Vorkonfiguration des Buskopplers über die TB20-ToolBox werden bestimmte Einstellungen für den späteren Betrieb des Buskopplers und der Module in der TB20-ToolBox vorgenommen und ab-

schließend im Buskoppler gespeichert. Dazu gehören die Parameter des Buskopplers sowie die Auswahl und Parametrierung der gewünschten Module. Bei der späteren Konfiguration des Buskopplers im EtherCAT-Konfigurationstool müssen diese Einstellungen dann nicht mehr erneut vorgenommen werden. Die vorkonfigurierten Module lassen sich im Konfigurationstool für gewöhnlich automatisch dem Buskoppler hinzufügen, wenn dort ein Geräte- bzw. Modulscan durchgeführt wird.

Der Hauptvorteil der Vorkonfiguration über die TB20-ToolBox liegt bei der Parametrierung der Module. Diese kann in der TB20-ToolBox deutlich komfortabler durchgeführt werden und erfordert außerdem kein spezielles Hintergrundwissen zum jeweiligen Aufbau der Parameterdaten.

Nachteilig ist die Vorkonfiguration gegenüber einer Offline-Konfiguration im EtherCAT-Konfigurationstool, wenn ein Austausch des Buskopplers erforderlich wird. Denn dann muss zuerst auch das Austauschgerät entsprechend vorkonfiguriert werden, wozu die TB20-ToolBox und das passende EtherCAT-Projekt benötigt werden.

Wie die TB20-ToolBox zur Vorkonfiguration des Buskopplers und seiner Module verwendet werden kann, wurde bereits in Kapitel 7.3.4 beschrieben.

7.6 Modultausch im laufenden Betrieb (Hot-Swap)

Der TB20 EtherCAT® Buskoppler erlaubt das als *Hot-Swap* bezeichnete Tauschen eines Moduls im laufenden Betrieb. Das heißt, Sie haben die Möglichkeit, ein eventuell defektes Modul gegen ein Modul gleichen Typs auszutauschen, ohne den laufenden Betrieb dafür unterbrechen zu müssen. Der Buskoppler und die restlichen Module laufen während des Tausches normal weiter.

Ein Modultausch im laufenden Betrieb muss aber auch von der SPS-Seite aus unterstützt werden. Hier sind in der Regel zusätzliche programmtechnische Vorkehrungen notwendig, um das zeitweise Fehlen eines Moduls und die damit zusammenhängenden Auswirkungen korrekt behandeln zu können. Aus diesem Grund ist der TB20 EtherCAT® Buskoppler standardmäßig so konfiguriert, dass der Tausch eines Moduls im laufenden Betrieb nicht erlaubt ist. Wird dennoch ein Modul im laufenden Betrieb gezogen, fällt der Buskoppler in den EtherCAT-State PREOP-ERROR zurück.

Durch Aktivierung des Buskoppler-Parameters 'Hot-Swap erlaubt' (bzw. 'Hot-swap allowed'), kann der Modultausch im laufenden Betrieb erlaubt werden. Diese Parameter-Einstellung lässt sich entweder über die TB20-ToolBox (↗ 7.3.4) oder über das Objektverzeichnis des Buskopplers (↗ 9.6.11) vornehmen.

ACHTUNG

Führen Sie das Ziehen und Stecken eines Moduls jeweils zügig durch, um dabei auftretende Störungen am Rückwandbus zeitlich kurz zu halten. Ansonsten kann auch bei erlaubtem Modultausch nicht ausgeschlossen werden, dass der Buskoppler den zyklischen E/A-Datenaustausch verlässt und nach PREOP-ERROR zurückfällt.
Ziehen Sie außerdem nie mehr als ein Modul gleichzeitig, da der Buskoppler auch in diesem Fall den zyklischen E/A-Datenaustausch verlässt und nach PREOP-ERROR zurückfällt.

7.6.1 Verhalten des Buskopplers beim Modultausch

Das nachfolgend beschriebene Verhalten des Buskopplers setzt voraus, dass der Modultausch im laufenden Betrieb durch den entsprechenden Buskoppler-Parameter (↗ 7.5.2.1) erlaubt wurde. 'Im laufenden Betrieb' bedeutet, dass sich der Buskoppler in den EtherCAT-States SAFEOP oder OP befindet und ein zyklischer E/A-Datenaustausch mit dem Master stattfindet.

Wird ein Modul gezogen, dann setzt der Buskoppler im zugehörigen Modulstatus (↗ 7.4.6) das Flag 'Present' und das Flag 'Operational' auf den Wert 0, um anzuzeigen, dass das Modul *nicht vorhanden* und *nicht betriebsbereit* ist. Da der Modulstatus eines jeden Moduls in den Prozesseingangsdaten des Buskopplers enthalten ist, stehen dem SPS-Programm diese Informationen zyklusaktuell zur Verfügung. Außerdem werden bei einem gezogenen Modul die Eingangsdaten, der Kanalstatus und der Diagnosestatus genullt, sofern jeweils vorhanden. Am Buskoppler wird das Fehlen eines Moduls durch eine gelb blinkende SF-LED angezeigt.

Wird das gezogene Modul durch das Stecken eines Moduls gleichen Typs ersetzt, dann setzt der Buskoppler im zugehörigen Modulstatus das Flag 'Present' wieder auf den Wert 1. Zuvor hat der Buskoppler das neu gesteckte Modul in den App-State STOP oder RUN geschaltet, je nachdem, ob er sich im EtherCAT-State SAFEOP oder OP befindet. Handelt es sich um ein parametrierbares Modul, dann hat der Buskoppler auch bereits die aktuell eingestellten Parameterdaten an das Modul übermittelt. Sofern keine modulweite Diagnose-Meldung im neu gesteckten Modul vorliegt, wird auch das Flag 'Operational' wieder auf den Wert 1 gesetzt. Außerdem werden jetzt wieder aktuelle Werte für die Eingangsdaten, den Kanalstatus und den Diagnosestatus des Moduls bereitgestellt, sofern jeweils vorhanden. Wenn nicht an anderer Stelle ein falsches Modul steckt, ist die SF-LED des Buskopplers nun wieder erloschen.

Wird anstelle des gezogenen Moduls ein falsches Modul gesteckt, dann wird das Modul zur SPS hin weiterhin als *nicht vorhanden* gemeldet. Der Buskoppler belässt das falsch gesteckte Modul im App-State IDLE und schaltet seine gelbe SF-LED dauerhaft ein.

7.7 Diagnose über die Buskoppler- und Modul-LEDs

7.7.1 Die LEDs des TB20 EtherCAT® Buskopplers

Die *blau/rote OK/BF-LED* zeigt den allgemeinen Zustand des Buskopplers an:

Zustand der OK/BF-LED	Erläuterung
ist aus	Der Buskoppler verfügt über keine ausreichende Spannungsversorgung oder es liegt gegebenenfalls ein Hardware-Defekt vor. → Überprüfen Sie die Spannungsversorgung des Buskopplers (↗ 4.3). Besteht das Problem weiterhin, kontaktieren Sie bitte den Support.
blinkt langsam blau	Der Buskoppler befindet sich in der Anlaufphase (INIT oder PREOP) ohne Vorliegen einer Diagnosemeldung.
leuchtet stetig blau	Der Buskoppler befindet sich im zyklischen Betrieb (SAFEOP oder OP) ohne Vorliegen einer Diagnosemeldung.
blinkt schnell blau	Der Buskoppler befindet sich im Simulationsbetrieb, ausgelöst über die TB20-ToolBox (↗ 7.3.8).
blinkt langsam rot	Es gibt keine Verbindung zu einem Master oder der Master kommuniziert nicht (mehr) mit dem Buskoppler. → Überprüfen Sie die Verbindung zum Master. Überprüfen Sie außerdem den aktuellen Zustand und die Konfiguration des Masters.
leuchtet stetig rot	Bei mindestens einem Modul liegt eine Diagnosemeldung vor. → In der TB20-ToolBox haben Sie über die Echtzeit-Diagnose (↗ 7.3.7) die Möglichkeit, sich die Diagnosemeldung anzeigen zu lassen.
blinkt schnell rot	Der Buskoppler befindet sich im Firmware-Update-Modus. → Führen Sie das Firmware-Update über die TB20-ToolBox zu Ende oder starten Sie erneut ein Firmware-Update (↗ 7.3.10).

Die *gelbe SF-LED* zeigt einen Fehler am Rückwandbus an:

Zustand der SF-LED	Erläuterung
ist aus	Es wurde kein Fehler am Rückwandbus detektiert.
blinkt langsam	*In INIT/PREOP:* Es sind keine Module gesteckt oder es fehlt mindestens ein *konfiguriertes Modul*. → Überprüfen Sie Ihren Modulaufbau auf Übereinstimmung mit der *aktiven Modulkonfiguration* (↗ 7.4.4.1). *In SAFEOP/OP:* Es wurde ein Modul im laufenden Betrieb (Hot-Swap, ↗ 7.6) gezogen. → Ersetzen Sie das gezogene Modul durch ein Modul gleichen Typs.
leuchtet stetig	Mindestens ein Modul stimmt nicht mit der *aktiven Modulkonfiguration* überein. → Falsch gesteckte Module verbleiben im App-State IDLE, sodass Sie sie leicht an der schnell blau blinkenden OK- bzw. OK/SF-LED erkennen können.

Die *grüne RUN-LED* zeigt den aktuellen EtherCAT-State des Buskopplers an:

Zustand der RUN-LED	Aktueller EtherCAT-State
ist aus	Init (INIT, ↗ 8.1.1)
blinkt	Pre-Operational (PREOP, ↗ 8.1.3)
blitzt periodisch auf	Safe-Operational (SAFEOP, ↗ 8.1.4)
leuchtet stetig	Operational (OP, ↗ 8.1.5)

In Kombination mit einem über die rote ERR-LED angezeigten Fehlerzustand werden die EtherCAT-States mit INIT-ERROR, PREOP-ERROR und SAFEOP-ERROR (↗ 8.1.6) bezeichnet.

Die *rote ERR-LED* zeigt einen Fehlerzustand der EtherCAT-State-Machine des Buskopplers an:

Zustand der ERR-LED	Erläuterungen
ist aus	Es liegt kein Fehlerzustand vor.
blinkt	Ein Wechsel in den vom Master angeforderten EtherCAT-State war nicht möglich, weil z.B. die Sync-Manager-Einstellungen ungültig sind oder die *konfigurierten* nicht mit den *gesteckten Modulen* übereinstimmen. → Werten Sie den AL-Status-Code (↗ 8.3) aus, um einen Hinweis auf die Fehlerursache zu erhalten und überprüfen Sie dahingehend die Konfiguration und den Modulaufbau.
blitzt periodisch auf	Der Buskoppler selbst hat einen Wechsel zu einem niedrigeren EtherCAT-State ausgelöst. Mögliche Ursachen hierfür sind u.a. das Starten des Simulationsbetriebs über die TB20-ToolBox oder das Ziehen von Modulen. → Werten Sie den AL-Status-Code (↗ 8.3) aus, um einen Hinweis auf die Ursache für den autonom durchgeführten State-Übergang zu erhalten.
blitzt 2x periodisch auf	Der Prozessausgangsdaten-Watchdog (↗ 7.5.5) ist abgelaufen und der Buskoppler ist selbständig von OP nach SAFEOP-ERROR gewechselt. Im AL-Status-Code ist in diesem Fall der Wert 0x001B enthalten. → Überprüfen Sie die Verkabelung und den Zustand des Masters. Überprüfen sie außerdem Ihre Zykluszeit-Einstellung für die Prozessdatenkommunikation im Hinblick auf das konfigurierte Watchdog-Timeout.

Die *grüne 'A L/A'-LED* und die *grüne 'B L/A'-LED* zeigen den Verbindungs- und Aktivitätszustand der beiden EtherCAT-Ports 'X1/A IN' und 'X1/B OUT' (↗ 7.2) an:

Zustand der L/A-LED	Erläuterungen zum betreffenden EtherCAT-Port
ist aus	Es ist keine Ethernet-Verbindung hergestellt.
leuchtet stetig	Über die hergestellte Ethernet-Verbindung läuft momentan keine Kommunikation.
blinkt	Über die hergestellte Ethernet-Verbindung wird gerade kommuniziert.

7.7.2 Die LEDs der Module

Die jeweils oberste LED zeigt bei allen Modulen den aktuellen Modulzustand an. Bei Modulen ohne Diagnosefähigkeit ist sie mit 'OK' beschriftet und leuchtet blau. Bei diagnosefähigen Modulen heißt sie 'OK/SF' und kann durch rotes Leuchten zusätzlich das Vorliegen einer Diagnosemeldung anzeigen.

Zustand der OK(/SF)-LED	Erläuterungen
ist aus	Das Modul verfügt über keine ausreichende Spannungsversorgung oder es liegt gegebenenfalls ein Hardware-Defekt vor. → Überprüfen Sie die korrekte Montage (↗ Kapitel 3) und Spannungsversorgung (↗ 4.3, 4.6) des TB20-Systems. Besteht das Problem weiterhin, ersetzen Sie das Modul wenn möglich durch ein Austauschmodul und kontaktieren Sie ggf. den Support.
blinkt schnell blau	Das Modul befindet sich im App-State IDLE (↗ 7.4.9). Bei diagnosefähigen Modulen liegt kein Parametrierfehler vor. Verfügt der Buskoppler über eine *aktive Modulkonfiguration* (↗ 7.4.4.1), dann verbleiben nur falsch gesteckte Module im App-State IDLE. → Sorgen Sie in diesem Fall für einen Abgleich zwischen *konfigurierten* und *gesteckten Modulen*.
blinkt langsam blau	Das Modul befindet sich im App-State STOP (↗ 7.4.9). Bei diagnosefähigen Modulen liegen kein Parametrierfehler und keine anderweitige Diagnosemeldung vor.
leuchtet stetig blau	Das Modul befindet sich im App-State RUN (↗ 7.4.9).Bei diagnosefähigen Modulen liegt kein Parametrierfehler und keine anderweitige Diagnosemeldung vor.
blinkt langsam rot	Im Modul liegt eine Diagnosemeldung aufgrund eines Parametrierfehlers vor. Das Modul kann sich dabei aktuell im App-State IDLE, STOP oder RUN befinden[1]. Der Diagnosestatus (↗ 7.4.8) des Moduls gibt Auskunft darüber, um welche Art von Parametrierfehler es sich handelt und ob sich dieser auf das gesamte Modul oder einzelne Kanäle bezieht. → Überprüfen und korrigieren Sie die Parameter-Einstellungen für das Modul. Konsultieren Sie hierzu ggf. das entsprechende Modulhandbuch.
leuchtet stetig rot	Im Modul liegt eine Diagnosemeldung vor, da mindestens ein modul- oder kanalbezogenes Problem diagnostiziert wurde. Das Modul kann sich dabei aktuell im App-State STOP oder RUN befinden[1]. Der Diagnosestatus (↗ 7.4.8) des Moduls gibt Auskunft darüber, um welche modul- oder kanalbezogenen Probleme es sich handelt. → Werten Sie den Diagnosestatus aus und versuchen Sie ggf. das Problem unter Zuhilfenahme des entsprechenden Modulhandbuchs zu beheben.

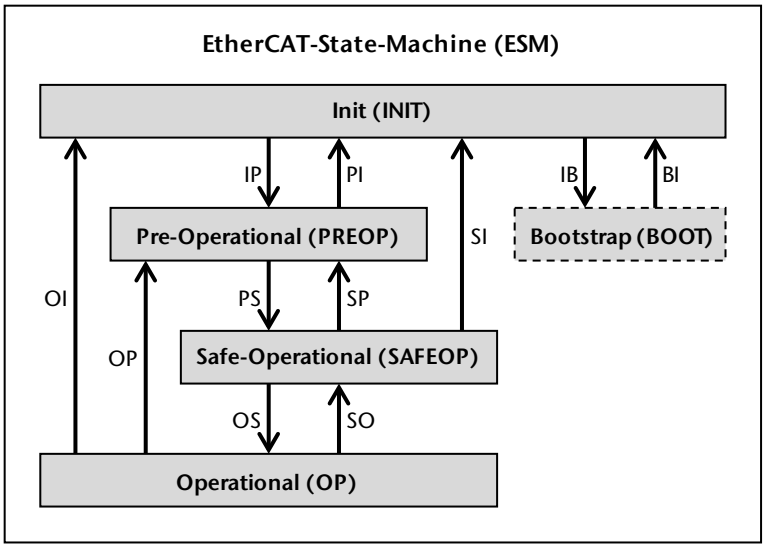
[1])Leuchtet die OK/SF-LED aufgrund einer Diagnosemeldung rot, ist ein Ablesen des aktuellen App-States anhand der LED im Moment nicht möglich. In diesem Fall können Sie die Echtzeit-Diagnose (↗ 7.3.7) der TB20-ToolBox verwenden, um sich bei Unklarheit den App-State anzeigen zu lassen.

Alle weiteren vorhandenen LEDs haben modul- und anwendungsspezifische Bedeutungen, über die Sie sich im jeweiligen Modulhandbuch informieren können.

8 Detailinformationen zur EtherCAT-State-Machine

Mit Kapitel 6.5 wurde bereits eine allgemeine Einführung zur EtherCAT-State-Machine gegeben. In diesem Kapitel wird nun konkret das Verhalten des TB20 EtherCAT® Buskopplers im Zusammenhang mit den verschiedenen EtherCAT-States und den möglichen Übergängen beschrieben. Bei dieser Beschreibung wird vorausgesetzt, dass die Unterschiede zwischen *detektierten, gespeicherten, projektierten* und *konfigurierten Modulen* (↗ 7.4.1) sowie zwischen *aktiver, aktueller* und *gespeicherter Modulkonfiguration* (↗ 7.4.4) bekannt sind.

Zur besseren Übersicht ist hier noch einmal die aus dem Einführungskapitel bekannte Abbildung zur EtherCAT-State-Machine dargestellt.



8.1 Die EtherCAT-States

8.1.1 Init (INIT)

Der INIT-State ist der Ausgangszustand nach Einschalten des Stroms. In diesem EtherCAT-State sind keine Mailbox- und keine Prozessdatenkommunikation möglich. Der INIT-State ist an einer ausgeschalteten RUN-LED erkennbar.

Der Zustand der Module ist davon abhängig, ob im Buskoppler eine *gespeicherte Modulkonfiguration* hinterlegt ist oder nicht:

Zustand der Module ohne gespeicherter Modulkonfiguration

Ohne *gespeicherte Modulkonfiguration* enthält die *aktive Modulkonfiguration* im INIT-State keine Module. Alle gesteckten Module befinden sich daher im App-State IDLE mit schnell blau blinkender OK(/SF)-LED.

Zustand der Module mit gespeicherter Modulkonfiguration

Ist eine *gespeicherte Modulkonfiguration* vorhanden, dann wird diese als *aktive Modulkonfiguration* übernommen. Je nachdem, ob ein gestecktes Modul mit der *aktiven Modulkonfiguration* übereinstimmt oder nicht, handelt es sich um ein korrekt gestecktes oder falsch gestecktes Modul.

Korrekt gesteckte Module befinden sich im App-State STOP. Parametrierbare Module haben außerdem die Modulparameter aus der *gespeicherten Modulkonfiguration* erhalten. Die OK(/SF)-LED der korrekt gesteckten Module blinkt langsam blau. Bei Vorliegen einer Diagnosemeldung blinkt oder leuchtet die OK/SF-LED abweichend rot (↗ 7.7.1).

Falsch gesteckte Module hingegeben verbleiben im APP-State IDLE mit schnell blau blinkender OK(/SF)-LED.

Anhand der OK(/SF)-LEDs der Module lässt sich so bereits im INIT-State optisch erkennen, welche Module korrekt gesteckt sind und welche nicht, selbst wenn kein Master vorhanden ist.

8.1.2 Bootstrap (BOOT)

Der BOOT-State ist ein optionaler EtherCAT-State zur Aktualisierung der Firmware eines Slaves. Er wird vom TB20 EtherCAT® Buskoppler nicht unterstützt, da Firmware-Updates mit Hilfe der TB20-ToolBox (↗ 7.3.10) durchgeführt werden.

8.1.3 Pre-Operational (PREOP)

Im PREOP-State ist Mailbox-, aber keine Prozessdaten-Kommunikation möglich. Über die Mailbox-Kommunikation kann nun auf das Objektverzeichnis des Buskopplers zugegriffen werden. Dieser Zugriff ist unter anderem erforderlich, um die Liste der *projektierten Module* an den Buskoppler übertragen und um Einstellungen an den Buskoppler- und Modul-Parametern vornehmen zu können. Der PREOP-State wird durch eine grün blinkende RUN-LED angezeigt.

Der Zustand der Module hängt von der *aktiven Modulkonfiguration* ab, die aus dem vorherigen EtherCAT-State übernommen wurde.

Alle laut *aktiver Modulkonfiguration* korrekt gesteckten Module befinden sich im App-State STOP. Parametrierbare Module haben die in der *aktiven Modulkonfiguration* hinterlegten Modulparameter erhalten. Die OK(/SF)-LED der korrekt gesteckten Module blinkt langsam blau. Bei Vorliegen einer Diagnosemeldung blinkt oder leuchtet die OK/SF-LED abweichend rot (↗ 7.7.1).

Alle falsch gesteckten Module befinden sich im App-State IDLE. Sie sind an einer schnell blau blinkenden OK(/SF)-LED erkennbar.

8.1.4 Safe-Operational (SAFEOP)

Ab dem SAFEOP-State ist neben der Mailbox-Kommunikation nun auch Prozessdaten-Kommunikation mit dem Master möglich. Die grüne RUN-LED des Buskopplers zeigt diesen EtherCAT-State durch ein periodisches Aufblitzen an.

Bei der Prozessdaten-Kommunikation findet ein zyklischer Austausch der Prozessausgangs- und Prozesseingangsdaten des Buskopplers mit dem Master statt.

Die vom Master an den Buskoppler gesendeten *Prozessausgangsdaten* enthalten die Ausgangsdaten für die *konfigurierten Module*. Da sich die korrekt gesteckten Module bei SAFEOP im App-State STOP befinden, werden die bereitgestellten Ausgangsdaten allerdings gar nicht übernommen. Die physikalischen Ausgänge der Module verbleiben stattdessen in einem sicheren Zustand, der abhängig vom Modul auch parametrierbar sein kann.

Die vom Buskoppler an den Master gesendeten *Prozesseingangsdaten* enthalten die aktuellen Eingangsdaten der *konfigurierten Module*. In den Prozesseingangsdaten sind außerdem für jedes Modul ein Modulstatus (↗ 7.4.6) und ggf. auch ein Kanalstatus (↗ 7.4.7) mit enthalten.

Auf das Objektverzeichnis des Buskopplers kann weiterhin via Mailbox-Kommunikation zugegriffen werden. So lassen sich über das Objektverzeichnis bei Bedarf z.B. der Diagnosestatus eines Moduls auslesen oder die Parameter-Einstellungen eines Moduls verändern.

Im Normalfall entsprechen im SAFEOP-State alle gesteckten Module der *aktiven Modulkonfiguration*. Sie haben ihre Parameterdaten erhalten und befinden sich im App-State STOP. Die OK(/SF)-LED dieser Module blinkt langsam blau. Bei Vorliegen einer Diagnosemeldung blinkt oder leuchtet die OK/SF-LED abweichend rot (↗ 7.7.1).

Lediglich Module, die im Rahmen eines Modultauschs (↗ 7.6) falsch gesteckt wurden, befinden sich im App-State IDLE. Die OK(/SF)-LED dieser falsch gesteckten Module blinkt schnell blau.

8.1.5 Operational (OP)

Im OP-State wird die im SAFEOP-State gestartete zyklische Prozessdaten-Kommunikation mit dem Master fortgeführt. Der entscheidende Unterschied ist aber, dass die korrekt gesteckten Module sich nun im App-State RUN befinden und die vom Master übermittelten Ausgangsdaten übernehmen. Ein Zugriff auf das Objektverzeichnis ist auch im OP-State weiterhin über Mailbox-Kommunikation möglich. Die RUN-LED des Buskopplers leuchtet dauerhaft grün.

Im Normalfall entsprechen im OP-State alle gesteckten Module der *aktiven Modulkonfiguration* und befinden sich im App-State RUN. Die OK(/SF)-LED dieser Module leuchtet stetig blau. Bei Vorliegen einer Diagnosemeldung blinkt oder leuchtet die OK/SF-LED abweichend rot (↗ 7.7.1).

Lediglich Module, die im Rahmen eines Modultauschs (↗ 7.6) falsch gesteckt wurden, befinden sich im App-State IDLE. Die OK(/SF)-LED dieser falsch gesteckten Module blinkt schnell blau.

8.1.6 INIT-ERROR, PREOP-ERROR und SAFEOP-ERROR

Der Zusatz 'ERROR' bei den EtherCAT-States INIT, PREOP und SAFEOP soll verdeutlichen, dass sich die EtherCAT-State-Machine des Buskopplers momentan in einem Fehlerzustand befindet. Optisch lässt sich dieser Fehlerzustand anhand der roten ERR-LED (↗ 7.7.1) erkennen, die abhängig von der Fehlerursache ein bestimmtes Leuchtverhalten zeigt. Zum Master hin wird der Fehlerzustand über ein gesetztes Fehler-Flag im AL-Status-Register (↗ 6.5.2.2) angezeigt. Der im AL-Status-Code-Register (↗ 6.5.2.3) enthaltene Fehler-Code informiert über die Fehlerursache. Eine Liste dieser AL-Status-Codes ist in Kapitel 8.3 zu finden.

Ein Fehlerzustand der EtherCAT-State-Machine wird zum einen immer dann hervorgerufen, wenn der Wechsel in einen vom Master angeforderten, höheren EtherCAT-State nicht möglich war. Ursachen hierfür können z.B. ungültige Sync-Manager-Einstellungen oder Abweichungen zwischen k*onfigurierten* und *gesteckten Modulen* gewesen sein. Der Buskoppler verbleibt in diesem Fall im aktuellen EtherCAT-State und lässt die ERR-LED rot blinken.

Zum anderen kann es aufgrund eines Problems erforderlich sein, dass der Buskoppler selbst einen Wechsel zu einem niedrigeren EtherCAT-State hin durchführt. Zum Beispiel wird der Buskoppler automatisch von OP nach SAFEOP-ERROR in den sicheren Betriebszustand wechseln, wenn der Prozessdaten-Watchdog abgelaufen ist. In diesem Fehlerfall blitzt die rote ERR-LED des Buskopplers periodisch jeweils zweimal kurz hintereinander auf.

Anderen Ursachen für einen automatischen Wechsel in einen niedrigeren EtherCAT-State können das Ziehen von Modulen oder der Start des Simulationsbetriebs über die TB20-ToolBox sein. In diesen Fällen blitzt die ERR-LED des Buskopplers periodisch rot auf.

Um den Fehlerzustand wieder verlassen und ggf. in einen anderen EtherCAT-State wechseln zu können, ist ein entsprechender AL-Control-Request vom Master erforderlich. Darin muss ein bestimmtes Flag zum Bestätigen und Ablöschen des Fehlers gesetzt sein, da der AL-Control-Request ansonsten ignoriert wird. Abweichend davon lässt sich ein Wechsel in den INIT-State immer erfolgreich anfordern.

8.2 Die EtherCAT-State-Übergänge

Nachfolgend sind alle möglichen State-Übergänge der EtherCAT-State-Machine des Buskopplers beschrieben. Die Bedeutung des Zusatzes 'ERROR' bei den State-Bezeichnungen wurde zuvor in Kapitel 8.1.6 erläutert. Mit AL-Status und AL-Status-Code ist jeweils abgekürzt der Inhalt des AL-Status-Registers (↗ 6.5.2.2) bzw. des AL-Status-Code-Registers (↗ 6.5.2.3) gemeint.

8.2.1 Reset→INIT

Nach dem Einschalten der Spannungsversorgung wird aus einem Reset-Zustand heraus automatisch in den INIT-State gewechselt. Ist eine *gespeicherte Modulkonfiguration* im Buskoppler vorhanden, wird diese als *aktive Modulkonfiguration* übernommen. Ansonsten enthält die *aktive Modulkonfiguration* keine Module und keine Parameterdaten.

Abhängig von der *aktiven Modulkonfiguration* werden übereinstimmend gesteckte Module in den App-State STOP geschaltet. Handelt es sich um parametrierbare Module, werden diese außerdem mit den aus der *gespeicherten Modulkonfiguration* stammenden Parameterdaten versorgt. Alle anderen Module verbleiben im App-State IDLE.

Der State-Übergang wird mit *AL-Status = INIT* und *AL-Status-Code = 0x0000* (kein Fehler) abgeschlossen.

8.2.2 INIT-ERROR→INIT

Bei diesem State-Übergang wird lediglich der aktuelle Fehlerzustand durch einen entsprechenden AL-Control-Request vom Master abgelöscht. Der State-Übergang endet mit *AL-Status = INIT* und *AL-Status-Code = 0x0000* (kein Fehler).

8.2.3 INIT(-ERROR)→BOOT

Dieser State-Übergang wird durch einen entsprechenden AL-Control-Request vom Master ausgelöst. Da der Buskoppler den optionalen BOOT-State nicht unterstützt, wird der State-Übergang mit *AL-Status = INIT-ERROR* und *AL-Status-Code = 0x0013* (EtherCAT-State Bootstrap wird nicht unterstützt) abgebrochen.

8.2.4 INIT(-ERROR)→PREOP

Dieser State-Übergang wird durch einen entsprechenden AL-Control-Request vom Master ausgelöst. Der Buskoppler darf sich dabei weder im Simulationsbetrieb (↗ 7.3.8) noch beim Empfang einer neuen Konfiguration (↗ 7.3.5) über die TB20-ToolBox befinden. Ansonsten wird der State-Übergang mit *AL-Status = INIT-ERROR* und *AL-Status-Code = 0x8000* (kein EtherCAT-Betrieb möglich) abgebrochen.

Ist die Vorbedingung erfüllt, dann wird die Konfiguration der Sync-Manager 0 und 1 überprüft, die für die Mailbox-Kommunikation benötigt werden. Die Konfiguration der beiden Sync-Manager hat der Master zuvor über die entsprechenden ESC-Register vorgenommen.

Bei gültiger Sync-Manager-Konfiguration wird die Mailbox-Kommunikation aktiviert. Über diese ist dann ab PREOP ein Zugriff auf das Objektverzeichnis (↗ Kapitel 9) des Buskopplers möglich. Der State-Übergang wird erfolgreich mit *AL-Status = PREOP* und *AL-Status-Code = 0x0000* (kein Fehler) beendet.

Ist die Sync-Manager-Konfiguration ungültig, dann wird der State-Übergang mit *AL-Status = INIT-ERROR* und *AL-Status-Code = 0x0016* (ungültige Mailbox-Konfiguration der Sync-Manager 0 & 1) abgebrochen.

8.2.5 PREOP(-ERROR)→INIT

Dieser State-Übergang wird durch einen entsprechenden AL-Control-Request vom Master ausgelöst. Dabei wird zum einen die Mailbox-Kommunikation wieder deaktiviert. Zum anderen wird die *aktive Modulkonfiguration* zurückgesetzt oder, wenn vorhanden, mit der *gespeicherten Modulkonfiguration* überschrieben. Eine zurückgesetzte *aktive Modulkonfiguration* enthält keine Module und keine Parameterdaten.

Abhängig von der nun *aktiven Modulkonfiguration* werden übereinstimmend gesteckte Module in den App-State STOP geschaltet. Handelt es sich um parametrierbare Module, werden diese außerdem mit

den aus der *gespeicherten Modulkonfiguration* stammenden Parameterdaten versorgt. Alle anderen Module werden in den App-State IDLE geschaltet.

Der State-Übergang wird mit *AL-Status = INIT* und *AL-Status-Code = 0x0000* (kein Fehler) beendet.

8.2.6 PREOP(-ERROR)→INIT-ERROR

Dieser State-Übergang wird vom Buskoppler ausgelöst, wenn über die TB20-ToolBox entweder der Upload einer Konfiguration (↗ 7.3.5) oder der Simulationsbetrieb (↗ 7.3.8) gestartet wird. Die Mailbox-Kommunikation wird deaktiviert. Danach wird der State-Übergang mit *AL-Status = INIT-ERROR* und *AL-Status-Code = 0x8000* (kein EtherCAT-Betrieb möglich) beendet.

8.2.7 PREOP-ERROR→PREOP

Bei diesem State-Übergang wird lediglich der aktuelle Fehlerzustand durch einen entsprechenden AL-Control-Request vom Master abgelöscht. Der State-Übergang endet mit *AL-Status = PREOP* und *AL-Status-Code = 0x0000* (kein Fehler).

8.2.8 PREOP(-ERROR)→SAFEOP

Dieser State-Übergang wird durch einen entsprechenden AL-Control-Request vom Master ausgelöst.

Ist die *aktuelle Modulkonfiguration* gültig, weil Sie mindestens ein *konfiguriertes Modul* und keine Modullücken enthält, dann wird sie als *aktive Modulkonfiguration* übernommen. Infolgedessen werden die nun übereinstimmend gesteckten Module in den App-State STOP geschaltet und erhalten, wenn parametrierbar, die aktuell konfigurierten Parameterdaten. Alle anderen Module werden in den App-State IDLE geschaltet.

Ist die *aktuelle Modulkonfiguration* ungültig, dann hat sie keinen Einfluss auf die *aktive Modulkonfiguration* und den aktuellen App-State der Module.

Ab SAFEOP befindet sich der Buskoppler in einer zyklischen Prozessdaten-Kommunikation mit dem Master. Damit diese Kommunikation erwartungsgemäß funktioniert, findet zuvor eine ganze Reihe an Überprüfungen statt, die nachfolgend genauer beschrieben sind. Ein Wechsel nach SAFEOP ist nur dann möglich, wenn jede der nachfolgend beschriebenen Überprüfungen erfolgreich durchgeführt werden konnte.

Überprüfung der gesteckten/detektierten Module:

Im Normalfall werden alle am Rückwandbus gesteckten Module vom Buskoppler erkannt (detektiert) und im Objekt 0xF050 (Detected Module Ident List, ↗ 9.6.9) verwaltet. Anhand dieser Liste wird überprüft, ob mindestens ein Modul am Rückwandbus gesteckt ist und dass keine Modullücken existieren. Schlägt diese Überprüfung fehlt, dann wird der State-Übergang mit *AL-Status = PREOP-ERROR* und *AL-Status-Code = 0x8100* (Keine Module oder Modullücke am Rückwandbus detektiert) abgebrochen.

Überprüfungen bei Vorhandensein einer gespeicherten Konfiguration:

Ist eine *gespeicherte Modulkonfiguration* im Buskoppler hinterlegt, dann wird überprüft, ob die darin enthaltenen Module mit den *projektierten Modulen* übereinstimmen. Wurden vom Master keine *projektierten Module* übermittelt, findet diese Überprüfung stattdessen mit den *detektierten Modulen* statt.

Gibt es keine Übereinstimmung zwischen den *gespeicherten* und den *projektierten* bzw. *detektierten Modulen*, dann wird der State-Übergang mit *AL-Status = PREOP-ERROR* und *AL-Status-Code = 0x8122* (Keine Übereinstimmung zwischen *projektierten* und *gespeicherten Modulen*) bzw. *AL-Status-Code = 0x8110* (Keine Übereinstimmung zwischen *gespeicherten* und *detektierten Modulen*) abgebrochen.

Überprüfungen zu den projektierten Modulen:

Sofern nicht explizit über einen bestimmten Buskoppler-Parameter (↗ 7.5.2.2) deaktiviert, muss der Master durch Beschreiben des Objektes 0xF030 (Configured Module Ident List, ↗ 9.6.13) die Liste der *projektierten Module* an den Buskoppler übermitteln. Ist dies nicht geschehen, dann wird der State-Übergang mit *AL-Status = PREOP-ERROR* und *AL-Status-Code = 0x8123* (Master hat Liste projektierter Module nicht übermittelt) abgebrochen.

Hat der Master eine Liste mit *projektierten Modulen* an den Buskoppler übermittelt, dann wird unabhängig vom zuvor erwähnten Buskoppler-Parameter überprüft, ob diese Liste gültig ist und ob sie mit den *detektierten, also gesteckten Modulen* übereinstimmt.

Ist die übermittelte Liste der *projektierten Module* ungültig, weil sie entweder leer ist oder Modullücken enthält, dann wird der State-Übergang mit *AL-Status = PREOP-ERROR* und *AL-Status-Code = 0x8120* (Vom Master übermittelte Liste *projektierter Module* ist ungültig) abgebrochen.

Gibt es keine Übereinstimmung zwischen den übermittelten *projektierten* und den *detektierten Modulen*, dann wird der State-Übergang mit *AL-Status = PREOP-ERROR* und *AL-Status-Code = 0x8121* (Keine Übereinstimmung zwischen *projektierten* und *detektierten Modulen*) abgebrochen.

Überprüfung der Konfiguration von Sync-Manager 2 und 3:

Die Sync-Manager 2 und 3 werden für die zyklische Prozessdaten-Kommunikation zwischen Buskoppler und Master benötigt. Sie sind konkret für den konsistenten Austausch der Prozessausgangsdaten (Sync-Manager 2) und der Prozesseingangsdaten (Sync-Manager 3) über den EtherCAT-Slave-Controller zuständig.

Die Konfiguration der beiden Sync-Manager hat der Master zuvor über entsprechende Register des EtherCAT-Slave-Controllers vorgenommen. Für eine Überprüfung von besonderer Relevanz sind die vom Master eingestellten Größen der Prozessausgangs- und Prozesseingangsdaten, die mit den vom Buskoppler erwarteten Prozessdatengrößen übereinstimmen müssen. Ist diese Übereinstimmung nicht gegeben, liegt das in der Regel daran, dass Master und Buskoppler von unterschiedlichen Modulkonfigurationen ausgehen. In diesem Fall wird dann der State-Übergang mit *AL-Status = PREOP-ERROR* und *AL-Status-Code = 0x001D* (Ungültige Prozessausgangsdaten-Konfiguration für Sync-Manager 2) bzw. *AL-Status-Code = 0x001E* (Ungültige Prozesseingangsdaten-Konfiguration für Sync-Manager 3) abgebrochen.

Alle anderen erforderlichen Einstellungen an den Sync-Managern 2 und 3 sollte der Master projektunabhängig anhand der Vorgaben aus der ESI-Datei des Buskopplers durchgeführt haben. Daher ist es eher unwahrscheinlich, dass sie die Ursache für eine ungültige Sync-Manager-Konfiguration und den Abbruch des State-Übergangs sind.

Erfolgreicher Abschluss der Überprüfungen:

Nachdem alle Überprüfungen erfolgreich abgeschlossen wurden, bereitet sich der Buskoppler auf den Start der zyklischen Prozessdaten-Kommunikation vor. Dazu zählt auch, dass der Buskoppler mit den am Rückwandbus befindlichen Modulen, einen lokalen zyklischen Austausch von Moduleingangs- und Modulausgangsdaten startet.

Der State-Übergang wird abschließend mit *AL-Status = SAFEOP* und *AL-Status-Code = 0x0000* (kein Fehler) erfolgreich beendet.

8.2.9 SAFEOP(-ERROR)→INIT

Dieser State-Übergang wird durch einen entsprechenden AL-Control-Request vom Master ausgelöst. Dabei werden zum einen die Prozessdaten- und die Mailbox-Kommunikation deaktiviert. Zum anderen wird die *aktive Modulkonfiguration* zurückgesetzt oder, wenn vorhanden, mit der *gespeicherten Modulkonfiguration* überschrieben. Eine zurückgesetzte *aktive Modulkonfiguration* enthält keine Module und keine Parameterdaten.

Abhängig von der nun *aktiven Modulkonfiguration* werden übereinstimmend gesteckte Module in den App-State STOP geschaltet. Handelt es sich um parametrierbare Module, werden diese außerdem mit den aus der *gespeicherten Modulkonfiguration* stammenden Parameterdaten versorgt. Alle anderen Module werden in den App-State IDLE geschaltet.

Der State-Übergang wird mit *AL-Status = INIT* und *AL-Status-Code = 0x0000* (kein Fehler) beendet.

8.2.10 SAFEOP(-ERROR)→INIT-ERROR

Dieser State-Übergang wird vom Buskoppler ausgelöst, wenn über die TB20-ToolBox entweder der Upload einer Konfiguration (↗ 7.3.5) oder der Simulationsbetrieb (↗ 7.3.8) gestartet wird. Die Prozessdaten- und die Mailbox-Kommunikation werden deaktiviert. Danach wird der State-Übergang mit *AL-Status = INIT-ERROR* und *AL-Status-Code = 0x8000* (kein EtherCAT-Betrieb möglich) beendet.

8.2.11 SAFEOP(-ERROR)→PREOP

Dieser State-Übergang wird durch einen entsprechenden AL-Control-Request vom Master ausgelöst. Die Prozessdaten-Kommunikation mit dem Master wird deaktiviert und der lokale zyklische Austausch von Ein- und Ausgangsdaten zwischen Buskoppler und Modulen wird beendet. Danach wird der State-Übergang mit *AL-Status = PREOP* und *AL-Status-Code = 0x0000* (kein Fehler) beendet.

8.2.12 SAFEOP(-ERROR)→PREOP-ERROR

Dieser State-Übergang wird durch den Buskoppler wegen eines Kommunikationsproblems am Rückwandbus oder wegen eines unerlaubterweise gezogenen Moduls ausgelöst. Die Prozessdaten-Kommunikation mit dem Master wird deaktiviert und der lokale zyklische Austausch von Ein- und Ausgangsdaten zwischen Buskoppler und Modulen wird beendet. Danach wird der State-Übergang mit *AL-Status = PREOP-ERROR* und *AL-Status-Code = 0x8200* (Kommunikationsproblem am Rückwandbus) oder *AL-Status-Code = 0x8201* (Unerlaubtes Ziehen eines Moduls) beendet.

8.2.13 SAFEOP-ERROR→SAFEOP

Bei diesem State-Übergang wird lediglich der aktuelle Fehlerzustand durch einen entsprechenden AL-Control-Request vom Master abgelöscht. Der State-Übergang endet mit *AL-Status = SAFEOP* und *AL-Status-Code = 0x0000* (kein Fehler).

8.2.14 SAFEOP(-ERROR)→OP

Dieser State-Übergang wird durch einen entsprechenden AL-Control-Request vom Master ausgelöst. Alle korrekt gesteckten Module werden in den App-State RUN geschaltet, sodass die vom Master bereitgestellten Ausgangsdaten nun in den Modulen zum Einsatz kommen. Der State-Übergang wird mit *AL-Status = OP* und *AL-Status-Code = 0x0000* (kein Fehler) beendet.

8.2.15 OP→INIT

Dieser State-Übergang wird durch einen entsprechenden AL-Control-Request vom Master ausgelöst. Die Prozessdaten-Kommunikation mit dem Master wird deaktiviert und der lokale zyklische Austausch von Ein- und Ausgangsdaten zwischen Buskoppler und Modulen wird beendet. Außerdem wird auch die Mailbox-Kommunikation beendet, sodass kein Zugriff mehr auf das Objektverzeichnis des Buskopplers möglich ist.

Weiterhin wird die *aktive Modulkonfiguration* zurückgesetzt oder, wenn vorhanden, mit der *gespeicherten Modulkonfiguration* überschrieben. Eine zurückgesetzte *aktive Modulkonfiguration* enthält keine Module und keine Parameterdaten.

Abhängig von der nun *aktiven Modulkonfiguration* werden übereinstimmend gesteckte Module in den App-State STOP geschaltet. Handelt es sich um parametrierbare Module, werden diese außerdem mit den aus der *gespeicherten Modulkonfiguration* stammenden Parameterdaten versorgt. Alle anderen Module werden in den App-State IDLE geschaltet.

Der State-Übergang wird mit *AL-Status = INIT* und *AL-Status-Code = 0x0000* (kein Fehler) beendet.

8.2.16 OP→INIT-ERROR

Dieser State-Übergang wird vom Buskoppler ausgelöst, wenn über die TB20-ToolBox entweder der Upload einer Konfiguration (↗ 7.3.5) oder der Simulationsbetrieb (↗ 7.3.8) gestartet wird. Die Prozessdaten- und die Mailbox-Kommunikation werden deaktiviert. Danach wird der State-Übergang mit *AL-Status = INIT-ERROR* und *AL-Status-Code = 0x8000* (kein EtherCAT-Betrieb möglich) beendet.

8.2.17 OP→PREOP

Dieser State-Übergang wird durch einen entsprechenden AL-Control-Request vom Master ausgelöst. Die Prozessdaten-Kommunikation mit dem Master wird deaktiviert und der lokale zyklische Austausch von Ein- und Ausgangsdaten zwischen Buskoppler und Modulen wird beendet. Außerdem werden alle korrekt gesteckten Module in den App-State STOP zurückgeschaltet. Danach wird der State-Übergang mit *AL-Status = PREOP* und *AL-Status-Code = 0x0000* (kein Fehler) beendet.

8.2.18 OP→PREOP-ERROR

Dieser State-Übergang wird durch den Buskoppler wegen eines Kommunikationsproblems am Rückwandbus oder wegen eines unerlaubterweise gezogenen Moduls ausgelöst. Die Prozessdaten-Kommunikation mit dem Master wird deaktiviert und der lokale zyklische Austausch von Ein- und Ausgangsdaten zwischen Buskoppler und Modulen wird beendet. Außerdem werden alle korrekt gesteckten Module in den App-State STOP zurückgeschaltet. Danach wird der State-Übergang mit *AL-Status = PREOP-ERROR* und *AL-Status-Code = 0x8200* (Kommunikationsproblem am Rückwandbus) oder *AL-Status-Code = 0x8201* (Unerlaubtes Ziehen eines Moduls) beendet.

8.2.19 OP→SAFEOP

Dieser State-Übergang wird durch einen entsprechenden AL-Control-Request vom Master ausgelöst. Alle korrekt gesteckten Module werden in den App-State STOP zurückgeschaltet. Danach wird der State-Übergang mit *AL-Status = SAFEOP* und *AL-Status-Code = 0x0000* (kein Fehler) beendet.

8.2.20 OP→SAFEOP-ERROR

Dieser State-Übergang wird durch den Buskoppler ausgelöst, weil der Prozessdaten-Watchdog abgelaufen ist. Alle korrekt gesteckten Module werden in den App-State STOP zurückgeschaltet. Danach wird der State-Übergang mit *AL-Status = SAFEOP-ERROR* und *AL-Status-Code = 0x001B* (Prozessdaten-Watchdog abgelaufen) beendet.

8.2.21 Ungültiger EtherCAT-State-Übergang

Ein ungültiger State-Übergang wird ausgelöst, wenn im AL-Control-Request vom Master (↗ 6.5.2.1) ein EtherCAT-State angefordert wird, in den aus dem aktuellen EtherCAT-State heraus nicht gewechselt werden kann. Beispiele hierfür sind INIT→SAFEOP, INIT→OP und PREOP→OP.

Wurde der State-Übergang aus OP heraus angefordert, dann wechselt der Buskoppler nach SAFEOP-ERROR. Die korrekt gesteckten Module werden dabei in den App-State STOP zurückgeschaltet. Der geänderte State-Übergang wird *mit AL-Status = SAFEOP-ERROR* und *AL-Status-Code = 0x0011* (Ungültiger EtherCAT-State-Übergang) beendet.

Erfolgte der State-Übergang aus einem anderen EtherCAT-State heraus, dann wird wie gewohnt der aktuelle EtherCAT-State beibehalten und im AL-Status das Fehler-Flag gesetzt. Der State-Übergang wird mit *AL-Status-Code = 0x0011* (Ungültiger EtherCAT-State-Übergang) abgebrochen.

8.2.22 Übergang zu unbekanntem EtherCAT-State

Ein Übergang zu einem unbekannten EtherCAT-State wird ausgelöst, wenn der AL-Control-Request vom Master (↗ 6.5.2.1) einen Wert enthält, der keinem bekannten EtherCAT-State zugeordnet werden konnte.

Wurde der State-Übergang aus OP heraus angefordert, dann wechselt der Buskoppler nach SAFEOP-ERROR. Die korrekt gesteckten Module werden dabei in den App-State STOP zurückgeschaltet. Der geänderte State-Übergang wird *mit AL-Status = SAFEOP-ERROR* und *AL-Status-Code = 0x0012* (Unbekannter EtherCAT-State) beendet.

Erfolgte der State-Übergang aus einem anderen EtherCAT-State heraus, dann wird wie gewohnt der aktuelle EtherCAT-State beibehalten und im AL-Status das Fehler-Flag gesetzt. Der State-Übergang wird mit *AL-Status-Code = 0x0012* (Unbekannter EtherCAT-State) abgebrochen.

8.3 Liste der AL-Status-Codes

Nachfolgend sind sämtliche AL-Status-Codes, die bei der EtherCAT-State-Machine des TB20 EtherCAT® Buskopplers auftreten können, aufgelistet und kurz beschrieben. Ab 0x8000 handelt es sich um hersteller- bzw. produktspezifische AL-Status-Codes.

AL-Status-Code	Beschreibung
0x0000	Kein Fehler
0x0001	Unspezifischer Fehler Ein nicht genauer bestimmbarer Fehler ist aufgetreten.
0x0011	Ungültiger EtherCAT-State-Übergang Ein Wechsel in den vom Master angeforderten EtherCAT-State ist aus dem aktuellen EtherCAT-State heraus nicht möglich (↗ 8.2.21).
0x0012	Unbekannter EtherCAT-State Der vom Master angeforderte EtherCAT-State ist nicht bekannt (↗ 8.2.22).
0x0013	EtherCAT-State Bootstrap wird nicht unterstützt Der optionale EtherCAT-State Bootstrap (BOOT) wird nicht unterstützt (↗ 8.1.2).
0x0016	Ungültige Mailbox-Konfiguration (Sync Manager 0 & 1) Ein Übergang von INIT nach PREOP war nicht möglich, weil die vom Master vorgenommene Mailbox-Kommunikation von Sync-Manager 0 und 1 ungültig ist (↗ 8.2.4).
0x001B	Prozessdaten-Watchdog abgelaufen Der Buskoppler hat automatisch einen Wechsel von OP nach SAFEOP-ERROR durchgeführt, weil der Prozessdaten-Watchdog abgelaufen ist (↗ 8.2.20).
0x001D	Ungültige Prozessausgangsdaten-Konfiguration (Sync-Manager 2) Ein Übergang von PREOP nach SAFEOP war nicht möglich, weil die vom Master vorgenommene Prozessausgangsdaten-Konfiguration von Sync-Manager 2 ungültig ist (↗ 8.2.8).
0x001E	Ungültige Prozesseingangsdaten-Konfiguration (Sync-Manager 3) Ein Übergang von PREOP nach SAFEOP war nicht möglich, weil die vom Master vorgenommene Prozesseingangsdaten-Konfiguration von Sync-Manager 3 ungültig ist (↗ 8.2.8).
0x001F	Ungültige Konfiguration des Prozessdaten-Watchdogs Ein Übergang von PREOP nach SAFEOP war nicht möglich, weil die vom Master vorgenommene Konfiguration des Prozessdaten-Watchdogs ungültig ist.
0x0028	Angeforderter Sync-Mode wird nicht unterstützt Ein Übergang von PREOP nach SAFEOP war nicht möglich, weil die vom Master vorgenommene Konfiguration des Sync-Modes ungültig ist. Der TB20 EtherCAT® Buskoppler unterstützt nur einen Prozessdaten-Austausch im Free-Run-Mode.

0x0029	Ungültige Buffer-Konfiguration für Sync-Manager 2 und 3 Ein Übergang von PREOP nach SAFEOP war nicht möglich, weil die vom Master vorgenommene Buffer-Konfiguration für Sync-Manager 2 und/oder 3 ungültig ist. Im vom Buskoppler unterstützten Free-Run-Mode müssen die beiden Sync-Manager im 3-Buffer-Mode konfiguriert sein.
0x8000	EtherCAT-Betrieb nicht möglich Der Buskoppler hat automatisch in den INIT-ERROR-State gewechselt bzw. lässt ein Verlassen des INIT-States nicht zu, da momentan kein EtherCAT-Betrieb möglich ist. Das ist immer dann der Fall, wenn sich der Buskoppler im Simulationsbetrieb befindet oder eine neue Konfiguration über die TB20-ToolBox hochgeladen wird.
0x8100	Keine Module oder Modullücke am Rückwandbus detektiert Ein Übergang von PREOP nach SAFEOP war nicht möglich, weil am Rückwandbus keine Module oder eine Modullücke detektiert wurden (↗ 8.2.8).
0x8110	Keine Übereinstimmung zwischen gespeicherten und detektierten Modulen Ein Übergang von PREOP nach SAFEOP war nicht möglich, weil die Module einer vorhandenen *gespeicherten Modulkonfiguration* nicht mit den am Rückwandbus *detektierten/gesteckten Modulen* übereinstimmen (↗ 8.2.8).
0x8120	Vom Master übermittelte Liste projektierter Module ist ungültig Ein Übergang von PREOP nach SAFEOP war nicht möglich, weil die vom Master übermittelte Liste der *projektierten Module* entweder leer ist oder Modullücken enthält (↗ 8.2.8).
0x8121	Keine Übereinstimmung zwischen projektierten und detektierten Modulen Ein Übergang von PREOP nach SAFEOP war nicht möglich, weil die vom Master übermittelte Liste der *projektierten Module* nicht mit den am Rückwandbus *detektierten/gesteckten Modulen* übereinstimmt (↗ 8.2.8).
0x8122	Keine Übereinstimmung zwischen projektierten und gespeicherten Modulen Ein Übergang von PREOP nach SAFEOP war nicht möglich, weil die vom Master übermittelte Liste der *projektierten Module* nicht mit den Modulen einer vorhandenen *gespeicherten Modulkonfiguration* übereinstimmt (↗ 8.2.8).
0x8123	Master hat Liste projektierter Module nicht übermittelt Ein Übergang von PREOP nach SAFEOP war nicht möglich, weil der Master keine Liste mit *projektierten Modulen* übermittelt hat und dies laut Buskoppler-Parameter (↗ 7.5.2.1) aber erforderlich ist (↗ 8.2.8).
0x8200	Kommunikationsproblem am Rückwandbus Der Buskoppler ist von SAFEOP/OP nach PREOP-ERROR zurückgefallen, weil am Rückwandbus ein Kommunikationsproblem mit den Modulen aufgetreten ist.
0x8201	Unerlaubtes Ziehen eines Moduls Der Buskoppler ist von SAFEOP/OP nach PREOP-ERROR zurückgefallen, weil am Rückwandbus unerlaubterweise ein Modul gezogen wurde. Wenn über die Buskoppler-Parametrierung Hot-Swap (↗ 7.5.2.1) erlaubt ist, dann darf maximal ein, ansonsten kein Modul gezogen werden.

9 Detailinformationen zum Objektverzeichnis

9.1 Allgemeines

Im Objektverzeichnis des TB20 EtherCAT® Buskopplers werden Objekte zur der Identifikation, Konfiguration, Parametrierung und Diagnose des Buskopplers und seiner Module bereitgestellt. Das Objektverzeichnis und der Zugriff darauf basieren auf dem CANopen-Protokoll, welches ursprünglich für CAN-Geräte entwickelt wurde (CAN = Controller Area Network). Die Adaption für EtherCAT heißt daher "CANopen over EtherCAT" (CoE).

Jedem im Objektverzeichnis enthaltenen Objekt ist zur Adressierung ein eindeutiger Index zugewiesen. Der Index ist ein 16-Bit-Wert, der für gewöhnlich in vierstelliger, hexadezimaler Schreibweise angegeben wird (z.B. 0xF030). Jedes Objekt hat eine Bezeichnung, einen Datentyp sowie bestimmte Flags. Durch die Flags werden u.a. die Zugriffsmöglichkeiten auf das Objekt definiert, das heißt, ob das Objekt gelesen und/oder beschrieben werden kann und in welchen EtherCAT-States (PREOP, SAFEOP, OP) dies jeweils erlaubt ist.

Nachfolgend wird zwischen Buskoppler- und Modul-Objekten unterschieden, je nachdem, ob sich die darin enthaltenen Informationen auf den Buskoppler selbst oder auf die Module beziehen.

9.2 Modular-Device-Profile (MDP)

Beim TB20 EtherCAT® Buskoppler handelt es sich um einen modularen EtherCAT-Slave, weshalb in Bezug auf das Objektverzeichnis das Modular-Device -Profile (MDP) verwendet wird. Für das MDP ist die Profilkennung 5001 spezifiziert, die in diesem Fall im Objekt 0x1000 (Device Type, ↗ 9.6.1) angegeben ist.

Das Modular-Device-Profile legt u.a. fest:

- wie der profilabhängige Bereich des Objektverzeichnisses aufgebaut ist
- welche Informationen bestimmte Objekte enthalten und ob diese zwingend oder optional zur Verfügung gestellt werden müssen
- auf welche Weise die Modul-Objekte (↗ 9.7) der *konfigurierten Module* in das Objektverzeichnis eingeblendet werden

9.3 Bereiche des Objektverzeichnisses

Die nachfolgende Tabelle gibt einen Überblick zu den verschiedenen Bereichen des Objektverzeichnisses und wie diese beim TB20 EtherCAT Buskoppler verwendet werden.

Index-Bereich	Beschreibung	Verwendung beim Buskoppler
0x0000 - 0x0FFF	**Bereich für Datentyp-Definitionen**	nicht in Verwendung
0x1000 - 0x1FFF	**Profilunabhängiger Bereich**	enthält profilunabhängige Buskoppler-Objekte
0x1XXX	Diverse Informationen	u.a. Identifikations- und Kenndaten des Buskopplers
0x1400 - 0x15FF	RxPDO-Parametrierung	nicht in Verwendung
0x1600 - 0x17FF	RxPDO-Konfiguration	Abbildung der Modulausgangsdaten auf RxPDOs
0x1800 - 0x19FF	TxPDO-Parametrierung	nicht in Verwendung
0x1A00 - 0x1BFF	TxPDO-Konfiguration	Abbildung der Moduleingangsdaten sowie des Modul- und Kanalstatus auf TxPDOs
0x1C10 - 0x1C2F	PDO-Zuordnung	Zuordnung der Rx- und TxPDOs zu den Sync-Managern
0x2000 - 0x5FFF	**Herstellerspezifischer Bereich**	enthält proprietäre Buskoppler- und Modul-Objekte
0x2100 - 0x2130	Modulkonfiguration	Status- und Steuerobjekte zur Modulkonfiguration
0x3000 - 0x3400	Parameterdaten	Parametrierung des Buskopplers und der Module

Index-Bereich	Beschreibung	Verwendung beim Buskoppler
0x6000 - 0xEFFF	**MDP-spezifischer Modulbereich**	enthält MDP-spezifische Buskoppler- und Modul-Objekte
0x6000 - 0x6FFF	Eingangsdaten-Bereich	Definition der Eingangsdaten der Module
0x7000 - 0x7FFF	Ausgangsdaten-Bereich	Definition der Ausgangsdaten der Module
0x8000 - 0x8FFF	Konfigurations-Bereich	nicht in Verwendung
0x9000 - 0x9FFF	Informations-Bereich	nicht in Verwendung
0xA000 - 0xAFFF	Diagnose-Bereich	Diagnose-Informationen zu den Module
0xB000 - 0xBFFF	Dienste-Transfer-Bereich	nicht in Verwendung
0xF000 - 0xFFFF	MDP-spezifischer Buskoppler-Bereich	enthält MDP-spezifische Buskoppler-Objekte - MDP-Kenndaten des Buskopplers - Liste der vom Buskoppler *detektierten Module* - Liste der im Buskoppler *konfigurierten Module*

9.4 Datentypen der Objekte

Jedes Objekt repräsentiert seine Daten in einem bestimmten Datentyp. Dabei handelt es sich entweder um einen sogenannten Basisdatentyp (Base Data Type) oder um die strukturierten Datentypen ARRAY und RECORD.

9.4.1 Basisdatentypen

Die nachfolgende Tabelle listet die vom Buskoppler verwendeten Basisdatentypen auf.

Basisdatentyp	Bitgröße	Wertebereich	Beschreibung
BOOL	1	0 (FALSE) und 1 (TRUE)	ein Bit umfassender, boolescher Datentyp (Boolean)
BYTE	8	0 bis 255	für Bitfelder verwendeter, vorzeichenloser, ganzzahliger Datentyp (Byte)
SINT	8	-128 bis 127	vorzeichenbehafteter, ganzzahliger Datentyp (Short Integer)
USINT	8	0 bis 255	vorzeichenloser, ganzzahliger Datentyp (Unsigned Short Integer)
INT	16	-32768 bis 32767	vorzeichenbehafteter, ganzzahliger Datentyp (Integer)
UINT	16	0 bis 65535	vorzeichenloser, ganzzahliger Datentyp (Unsigned Integer)
DINT	32	-2^{31} bis 2^{31} -1	vorzeichenbehafteter, ganzzahliger Datentyp (Double Integer)
UDINT	32	0 bis 2^{32} -1	vorzeichenloser, ganzzahliger Datentyp (Unsigned Double Integer)
STRING(n)	8·n	8 Bit pro Zeichen	Zeichenkette mit n Zeichen à 8 Bits (ohne Nullterminierung)

9.4.2 Strukturierte Datentypen ARRAY und RECORD

Bei Objekten vom Datentyp ARRAY oder RECORD sind die enthaltenen Daten auf mehrere Objekteinträge (Object Entries) aufgeteilt. Die Adressierung der einzelnen Objekteinträge findet unter Angabe eines Subindizes statt. Beim Subindex handelt es sich um einen 8-Bit-Wert im Bereich 0 bis 255, sodass maximal 256 unterschiedliche Objekteinträge adressiert werden können. Der Begriff Subindex wird oft auch als Synonym für den damit adressierten Objekteintrag verwendet (z.B. 'Subindex 3 enthält den Wert X' anstelle von 'der Objekteintrag mit Subindex 3 enthält den Wert X').

Als Referenz auf einen bestimmten Objekteintrag wird folgende Notation aus Objektindex und Subindex verwendet: <hexadezimaler Objektindex>:<dezimaler Subindex>

z.B. 0xF030:12 (Subindex 12 von Objekt 0xF030)

Subindex 0

Subindex 0 ist ein spezieller Subindex, der bei ARRAY- und RECORD-Objekten die Anzahl der nachfolgenden, verfügbaren Subindizes (0 bis maximal 255) enthält. Der Subindex 0 hat immer den Basisdatentyp USINT (8 Bits) und trägt die Bezeichnung "SubIndex 000". Durch entsprechende Zugriffs-Flags wird für den Subindex 0 angegeben, ob dieser nur gelesen oder auch beschrieben werden kann und in welchen EtherCAT-States dies jeweils möglich ist.

Subindex 1 bis 255 eines ARRAY-Objekts

Bei einem ARRAY-Objekt sind für alle Subindizes 1 bis maximal 255 der gleichen Basisdatentyp und die gleichen Zugriffseinstellungen festgelegt. Außerdem wird für all diese Subindizes eine einheitliche Bezeichnung nach dem Schema "SubIndex *nnn*" verwendet, wobei *nnn* für den dreistelligen dezimalen Subindex steht.

Als Beispiele für ARRAY-Objekte seien hier die Objekte 0xF030 (Configured Module Ident List, ↗ 9.6.13) und 0xF050 (Detected Module Ident List, ↗ 9.6.14) aufgeführt.

Subindizes 1 bis 255 eines RECORD-Objekts

Im Unterschied zu den ARRAY-Objekten sind bei den RECORD-Objekten der Basisdatentyp und die Zugriffseinstellungen für jeden der Subindizes 1 bis maximal 255 individuell festgelegt. Auch können für jeden der Subindizes individuelle Bezeichnungen verwendet werden.

Als Beispiele für RECORD-Objekte seien hier das Objekt 0x2100 (Module Configuration Status, ↗ 9.6.7) und die Parameterdaten-Objekte der Module (↗ 9.7.3) genannt.

9.5 Zugriff auf das Objektverzeichnis

Der Zugriff auf das Objektverzeichnis erfolgt vom Master aus über die folgenden CoE-Dienste:

- SDO-Upload zum Lesen von Objektdaten
- SDO-Download zum Schreiben von Objektdaten
- SDO-Info zum Auslesen der Index-Liste der online verfügbaren Objekte sowie weiterer Informationen zu den einzelnen Objekten (u.a. Bezeichnung, Datentyp, Flags)

SDO ist die Abkürzung für "Service Data Object" und bezeichnet die Daten, die beim Zugriff auf das Objektverzeichnis über die CoE-Dienste ausgetauscht werden. Der CoE-Dienst verwendet zur Durchführung der SDO-Kommunikation den Mailbox-Dienst, der ab dem EtherCAT-State PREOP zur Verfügung steht. Im EtherCAT-State INIT ist deshalb noch kein Zugriff auf das Objektverzeichnis möglich.

Auf welche Daten des Objektverzeichnisses lesend oder schreibend zugegriffen werden soll, wird durch den Index und bei ARRAY- und RECORD-Objekten zusätzlich durch den Subindex angegeben. Die Übertragung der SDO-Daten findet byte-orientiert und im Little-Endian-Format statt:

- Auch beim Einzelzugriff auf Subindizes vom Datentyp BOOL, deren Daten jeweils nur ein Bit umfassen, wird ein ganzes Byte übertragen. Die relevante Information ist in diesem Fall im niederwertigsten Bit (Bit 0) des übertragenen Bytes enthalten.
- Beim Zugriff auf Subindizes vom Datentyp INT, DINT, UINT und UDINT werden die betreffenden Zahlenwerte im Little-Endian-Format, also beginnend mit dem niederwertigsten Byte übertragen.

Der Zugriff auf ARRAY- und RECORD-Objekte kann entweder Subindex-weise oder via Complete-Access erfolgen.

9.5.1 Zugriff via Complete-Access

Beim Zugriff via Complete-Access kann auf alle Subindizes eines ARRAY- oder RECORD-Objektes mit nur einem SDO-Auftrag lesend oder schreibend zugegriffen werden. Dies stellt neben einem Geschwindigkeitsvorteil vor allem eine Erleichterung für entsprechende Objektzugriffe aus einem SPS-Programm heraus dar, wo sonst bei Bedarf auf alle Subindizes einzeln nacheinander zugegriffen werden müsste.

Der Zugriff via Complete-Access wird nicht von allen, aber von den meisten ARRAY- und RECORD-Objekten des Buskopplers unterstützt. Hierzu ist jeweils eine entsprechende Angabe bei der Beschreibung der einzelnen Objekte vorhanden.

Als Subindex können bei Complete-Access nur die folgenden Werte verwendet werden:

- 0 = Zugriff beginnt bei Subindex 0 (Subindex 0 ist also mit eingeschlossen)
- 1 = Zugriff beginnt bei Subindex 1 (Subindex 0 ist also nicht mit eingeschlossen)

Wird ein anderer Subindex als 0 oder 1 verwendet, dann wird der SDO-Auftrag mit dem SDO-Abort-Code 0x06010000 (nicht unterstützte Zugriffsweise) abgebrochen.

Die zu lesenden oder zu schreibenden Objektdaten setzen sich bei einem Zugriff via Complete-Access aus den einzelnen Daten der Subindizes zusammen. Das bei der Beschreibung der ARRAY- und RECORD-Objekte für jeden Subindex angegebene Bitoffset zeigt die Bitposition der jeweiligen Subindex-Daten innerhalb der Objektdaten an. Die Angabe des Bitoffsets bezieht sich dabei auf die Objektdaten bei Complete-Access inklusive Subindex 0. Die Daten von Subindex 1 beginnen in diesem Fall immer bei Bitoffset 16, da hinter den 8 Bit umfassenden Daten von Subindex 0 standardmäßig ein zusätzliches Füllbyte eingefügt ist. Bei Complete-Access exklusive Subindex 0 beginnen die Objektdaten mit den Daten von Subindex 1. Die Bitoffset-Angaben der Subindizes müssen dann um den Wert 16 reduziert werden, um die jeweilige Bitposition innerhalb der Objektdaten zu erhalten.

Wird zum Beispiel auf ein Parameterdaten-Objekt via Complete-Access zugegriffen, dann entsprechen die ab Subindex 1 enthaltenen Objektdaten genau dem byte-weisen Aufbau der im jeweiligen Modulhandbuch beschriebenen Parameterdaten.

9.5.2 Zugriff aus einem EtherCAT-Konfigurationstool heraus

Für gewöhnlich besteht in den EtherCAT-Konfigurationstools die Möglichkeit, auf das Objektverzeichnis des Buskopplers zuzugreifen. Dabei kann meist zwischen einer Offline- und einer Online-Variante des Objektverzeichnisses unterschieden werden.

9.5.2.1 Offline-Objektverzeichnis

Das Offline-Objektverzeichnis wird vom EtherCAT-Konfigurationstool aus den in der ESI-Datei (↗ 6.6.1) enthaltenen Informationen generiert. Das heißt, zur Anzeige des Offline-Objektverzeichnisses muss die ESI-Datei des Buskopplers installiert sein. Sofern keine Online-Werte im Offline-Objektverzeichnis angezeigt werden sollen, muss der Buskoppler nicht über EtherCAT erreichbar sein.

Sämtliche Modul-Objekte (↗ 9.7), wie z.B. die Parameterdaten-Objekte der Module, werden vom EtherCAT-Konfigurationstool in Abhängigkeit von den aktuell *projektierten Modulen* dynamisch in das Offline-Objektverzeichnis eingeblendet.

Beim Offline-Objektverzeichnis ist es in der Regel möglich, zwischen der Anzeige von Offline- und Online-Werten für die enthaltenen Objekte umzuschalten:

Bei der Anzeige von *Offline-Werten* können nur für solche Objekte und Objekteinträge tatsächlich Werte angezeigt werden, für die entsprechende Default-Werte in der ESI-Datei hinterlegt wurden. Das ist momentan nur bei den Parameterdaten-Objekten der Module der Fall, weil es vor allem für eine Offline-Konfiguration des Buskopplers von Vorteil ist. Bei allen anderen Objekten können daher keine Offline-Werte angezeigt werden.

Auf die Anzeige von *Online-Werten* kann nur dann umgeschaltet werden, wenn der entsprechende Buskoppler über EtherCAT erreichbar ist und sich mindestens im EtherCAT-State PREOP befindet. Bei Modul-Objekten funktioniert die Anzeige von Online-Werten außerdem nur dann verlässlich, wenn die im EtherCAT-Konfigurationstool *projektierten Module* mit den im Buskoppler *konfigurierten* und *detektierten Modulen* übereinstimmen, weil nur dann die ins Offline-Objektverzeichnis eingeblendeten Modul-Objekte auch tatsächlich vom Buskoppler online bereitgestellt werden.

Bei der Anzeige von Online-Werten im Offline-Objektverzeichnis ist es in der Regel auch möglich, Objekte bzw. Objekteinträge zu beschreiben, sofern dies die jeweiligen Zugriffseinstellungen des Objektes im aktuellen EtherCAT-State des Buskopplers erlauben.

9.5.2.2 Online-Objektverzeichnis

Für die Anzeige des Online-Objektverzeichnisses ist es erforderlich, dass der entsprechende Buskoppler über EtherCAT erreichbar ist und sich mindestens im EtherCAT-State PREOP befindet. Alle zur Darstellung des Online-Objektverzeichnis benötigten Informationen werden über den CoE-Dienst 'SDO Info' aus dem Buskoppler ausgelesen. Im Normalfall werden zu den Objekten automatisch auch die jeweiligen Online-Werte vom Buskoppler abgerufen und angezeigt.

Im Online-Objektverzeichnis basieren die enthaltenen Modul-Objekte auf den im Buskoppler *konfigurierten Modulen*. Damit der Buskoppler die Modul-Objekte eines bestimmten *konfigurierten Moduls* auch tatsächlich online bereitstellen kann, muss das entsprechende Modul am Rückwandbus gesteckt sein. Das ist erforderlich, weil der Buskoppler selbst über kein Detailwissen zu den einzelnen Modulen verfügt und sämtliche benötigten Informationen aus den Modulen auslesen muss. Abweichend dazu können Parameterdaten-Objekte auch dann bereitgestellt werden, wenn im Buskoppler eine *gespeicherte Modulkonfiguration* vorliegt und das *konfigurierte Modul* mit dem entsprechenden *gespeicherten Modul* übereinstimmt.

9.5.3 Zugriff aus einem SPS-Programm heraus

Der Zugriff aus einem SPS-Programm heraus findet mit Hilfe bestimmter Funktionsbausteine statt, über die lesend und schreibend auf das Objektverzeichnis des Buskopplers zugegriffen werden kann. Voraussetzung dafür ist, dass der Buskoppler über EtherCAT an den Master angeschlossen ist und sich mindestens im EtherCAT-State-PREOP befindet. Die benötigten Funktionsbausteine werden durch den Hersteller der verwendeten Speicherprogrammierbaren Steuerung (SPS) bzw. des EtherCAT-Masters meist in Form einer Funktionsbibliothek zur Verfügung gestellt.

In den Namen der Funktionsbausteine sind für gewöhnlich die Begriffe 'CoE' (CANopen over EtherCAT) und/oder 'SDO' (Service Data Object) in Verbindung mit 'Read' für Lesen und 'Write' für Schreiben enthalten. Bei ARRAY- oder RECORD-Objekt kann im Normalfall angegeben werden, ob sich der Zugriff nur auf einen Subindex oder via Complete-Access auf das gesamte Objekt beziehen soll.

Beim Aufruf der Funktionsbausteine findet dann im Hintergrund eine entsprechende SDO-Kommunikation zwischen Master und Buskoppler statt, für deren Abwicklung in der Regel mehrere Task-Zyklen benötigt werden. Bei fehlgeschlagenen Zugriffen wird ein entsprechender SDO-Abort-Code (↗ 9.5.4) als Fehlercode zurückgemeldet, der einen Hinweis auf die Fehlerursache gibt.

Für konkrete Details zu den Funktionsbausteinen und deren Verwendung sei auf die jeweilige Dokumentation vom Anbieter der Funktionsbausteine verwiesen.

Relevante Anwendungsfälle für diesen Zugriffsweg auf das Objektverzeichnis sind:

- das Auslesen des detaillierteren Diagnosestatus-Objektes (↗ 9.7.8) eines Moduls, nachdem über den zyklusaktuell bereitgestellten Modul- oder Kanalstatus ein Problem angezeigt wurde
- die Umparametrierung von Modulen im laufenden Betrieb über die entsprechenden Parameterdaten-Objekte (↗ 9.7.3)

9.5.4 SDO-Abort-Codes

Vom TB20 EtherCAT® Buskoppler werden bei fehlgeschlagenen SDO-Aufträgen die nachfolgend aufgelisteten SDO-Abort-Codes verwendet.

SDO-Abort-Code	Beschreibung
0x05030000	neue SDO-Anfrage ohne geändertes Toggle-Bit
0x05040001	SDO-Anfrage mit unbekanntem 'Command Specifier'
0x06010000	nicht unterstützte Zugriffsweise
0x06010002	unerlaubter Schreibzugriff auf ein nur lesbares Objekt/einen nur lesbaren Objekteintrag
0x06010003	Änderung des Objekteintrags nicht möglich, da Subindex 0 nicht zuvor auf den Wert 0 gesetzt wurde
0x06010004	Zugriff via Complete-Access wird bei diesem Objekt nicht unterstützt
0x06020000	betreffendes Objekt existiert nicht
0x06040043	Objekteintrag kann nicht mit übermittelten Wert beschrieben werden
0x06070010	Länge der Daten passt nicht zum Objekt/Objekteintrag
0x06090011	betreffender Objekteintrag existiert nicht
0x06090030	zu schreibender Wert ist zu groß für den Objekteintrag
0x08000000	allgemeiner, unspezifischer Fehler
0x08000022	Zugriff im aktuellen EtherCAT-State nicht möglich

9.6 Die Buskoppler-Objekte

Nachfolgend sind die statisch im Objektverzeichnis vorhandenen Buskoppler-Objekte beschrieben. Sie dienen zur Identifikation, Konfiguration und Parametrierung des Buskopplers. Die Buskoppler-Objekte zur PDO-Konfiguration werden gesondert in Kapitel 9.8 beschrieben.

9.6.1 Objekt 0x1000 - Device Type

Index 0x1000	Device Type
Kurzbeschreibung	Typenkennung des Gerätes
Datentyp (Bitgröße)	UDINT (32)
Zugriff	nur lesend
Online-Wert	0x00001389 (5001) = Gerät verwendet das Modular Device Profile (MDP, ↗ 9.2)

9.6.2 Objekt 0x1008 - Device Name

Index 0x1008	Device Name
Kurzbeschreibung	Name des Gerätes
Datentyp (Bitgröße)	STRING (variabel)
Zugriff	nur lesend
Online-Wert	"TB20 Coupler EtherCAT"

9.6.3 Objekt 0x1009 - Hardware Version

Index 0x1009	Hardware Version
Kurzbeschreibung	Hardware-Version des Gerätes
Datentyp (Bitgröße)	STRING (variabel)
Zugriff	nur lesend
Online-Wert	Beispiel: "HW1-1"

9.6.4 Objekt 0x100A - Software Version

Index 0x100A	Software Version
Kurzbeschreibung	Software-Version des Gerätes
Datentyp (Bitgröße)	STRING (variabel)
Zugriff	nur lesend
Online-Wert	Beispiel: "1.04.002"

9.6.5 Objekt 0x1018 - Identity

Index 0x1018	Identity
Kurzbeschreibung	Kenndaten des Gerätes
Datentyp (Bitgröße)	RECORD (144)
Zugriff via Complete-Access	möglich
Subindex 0	**SubIndex 000**
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	4
Subindex 1	**Vendor ID**
Kurzbeschreibung	EtherCAT-spezifische Kennung des Geräteherstellers
Datentyp (Bitgröße/Bitoffset)	UDINT (32 / 16)
Zugriff	nur lesend
Online-Wert	0x00000223 (547) = Kennung der Helmholz GmbH & Co. KG
Subindex 2	**Product code**
Kurzbeschreibung	Produktkennung des Gerätes
Datentyp (Bitgröße/Bitoffset)	UDINT (32 / 48)
Zugriff	nur lesend
Online-Wert	0x00000009 (9) = Kennung des TB20 EtherCAT® Buskopplers
Subindex 3	**Revision**
Kurzbeschreibung	Revisionskennung des Gerätes
Datentyp (Bitgröße/Bitoffset)	UDINT (32 / 80)
Zugriff	nur lesend
Online-Wert	z.B. 0x00000002 (2)
Subindex 4	**Serial number**
Kurzbeschreibung	Seriennummer des Gerätes
Datentyp (Bitgröße/Bitoffset)	UDINT (32 / 112)
Zugriff	nur lesend
Online-Wert	z.B. 0x0001E240 (123456)

9.6.6 Objekt 0x1C00 - Sync Manager Communication Types

Index 0x1C00	Sync Manager Communication Types
Kurzbeschreibung	Kommunikationstypen der verfügbaren Sync-Manager
Datentyp (Bitgröße)	ARRAY OF USINT (48)
Zugriff via Complete-Access	möglich

Index 0x1C00	Sync Manager Communication Types
Subindex 0	**SubIndex 000**
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes = Anzahl der verfügbaren Sync-Manager
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	4
Subindex 1	**SubIndex 001**
Kurzbeschreibung	Kommunikationstyp von Sync-Manager 0
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 16)
Zugriff	nur lesend
Online-Wert	1 = Mailbox-Kommunikation (Master → Slave)
Subindex 2	**SubIndex 002**
Kurzbeschreibung	Kommunikationstyp von Sync-Manager 1
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 24)
Zugriff	nur lesend
Online-Wert	2 = Mailbox-Kommunikation (Slave → Master)
Subindex 3	**SubIndex 003**
Kurzbeschreibung	Kommunikationstyp von Sync-Manager 2
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 32)
Zugriff	nur lesend
Online-Wert	3 = Prozessausgangsdaten-Kommunikation (Master → Slave)
Subindex 4	**SubIndex 004**
Kurzbeschreibung	Kommunikationstyp von Sync-Manager 3
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 40)
Zugriff	nur lesend
Online-Wert	4 = Prozesseingangsdaten-Kommunikation (Slave → Master)

9.6.7 Objekt 0x2100 - Module Configuration Status

Index 0x2100	Module Configuration Status
Kurzbeschreibung	Aktueller Status der Modulkonfiguration
Datentyp (Bitgröße)	RECORD (24)
Zugriff via Complete-Access	möglich
Weitere Details	Die Subindizes 2 bis 7 stehen in direktem Zusammenhang mit den Subindizes 1 bis 6 von Objekt 0x2130 (Configured Module List Status).
Subindex 0	**SubIndex 000**
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	8
Subindex 1	**Stored configuration present**
Kurzbeschreibung	zeigt an, ob eine *gespeicherte Modulkonfiguration* vorhanden ist
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 16)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nicht vorhanden; 1 (TRUE) = vorhanden

Index 0x2100	Module Configuration Status
Subindex 2	**Module list: based on detected modules**
Kurzbeschreibung	zeigt an, ob die Liste der *konfigurierten Module* auf den *detektierten Modulen* basiert
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 17)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein, 1 (TRUE) = ja
Weitere Details	Die *konfigurierten* basieren immer dann auf den *detektierten Modulen*, wenn keine *gespeicherte Modulkonfiguration* vorhanden ist und der Master (noch) keine Liste mit *projektierten Modulen* übermittelt hat.
Subindex 3	**Module list: based on stored modules**
Kurzbeschreibung	zeigt an, ob die Liste der *konfigurierten Module* auf den *gespeicherten Modulen* aus einer vorhandenen *gespeicherten Modulkonfiguration* basiert
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 18)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Die *konfigurierten* basieren immer dann auf den *gespeicherten Modulen*, wenn eine *gespeicherte Modulkonfiguration* vorhanden ist und der Master (noch) keine Liste mit *projektierten Modulen* übermittelt hat.
Subindex 4	**Module list: based on projected modules**
Kurzbeschreibung	zeigt an, ob die Liste der *konfigurierten Module* auf den vom Master übermittelten *projektierten Modulen* basiert
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 19)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Die *konfigurierten* basieren immer dann auf den *projektierten Modulen*, wenn der Master die Liste mit den *projektierten Modulen* übermittelt hat.
Subindex 5	**Module list: valid (not empty, no gaps)**
Kurzbeschreibung	zeigt an, ob die Liste der *konfigurierten Module* gültig ist (nicht leer und ohne Lücke)
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 20)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Der State-Übergang PREOP→SAFEOP ist nur mit gültiger Modulliste möglich.
Subindex 6	**Module list: matches detected modules**
Kurzbeschreibung	zeigt an, ob die *konfigurierten* mit den *detektierten Modulen* übereinstimmen
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 21)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Der State-Übergang PREOP→SAFEOP ist nur möglich, wenn die *konfigurierten* mit den *detektierten Modulen* übereinstimmen
Subindex 7	**Module list: matches stored modules**
Kurzbeschreibung	zeigt an, ob die *konfigurierten* mit den Modulen aus einer vorhandenen *gespeicherten Modulkonfiguration* übereinstimmen
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 22)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Wenn eine *gespeicherte Modulkonfiguration* vorhanden ist, dann ist ein State-Übergang PREOP→SAFEOP nur möglich, wenn die *konfigurierten* mit den *gespeicherten Modulen* übereinstimmen.

Index 0x2100	Module Configuration Status
Subindex 8	**Module parameters: match stored parameters**
Kurzbeschreibung	zeigt an, ob die aktuellen Parameter der *konfigurierten Module* mit denen in der vorhandenen *gespeicherten Modulkonfiguration* übereinstimmen
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 23)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Wenn keine *gespeicherte Modulkonfiguration* vorhanden ist oder die *konfigurierten* nicht mit den *gespeicherten Modulen* übereinstimmen, dann ist der Wert immer 0 (FALSE). Über den Objekteintrag 0x2110:03 (Command: Reload stored parameters) können in PREOP die gespeicherten Modulparameter wiederhergestellt werden.

9.6.8 Objekt 0x2110 - Module Configuration Control

Index 0x2110	Module Configuration Control
Kurzbeschreibung	Steuerobjekt der Modulkonfiguration
Datentyp (Bitgröße)	RECORD (24)
Zugriff via Complete-Access	nicht möglich
Weitere Details	Bei den Subindizes 1 bis 5 findet nur dann eine Aktion statt, wenn sie mit dem Wert 1 (TRUE) beschrieben werden. Ein Beschreiben mit dem Wert 0 (FALSE) hat also keinen Effekt. Ein schreibender Zugriff auf die Subindizes 3 bis 5 ist nur in PREOP erlaubt, ansonsten wird der SDO-Abort-Code 0x08000022 (Zugriff im aktuellen State nicht möglich) zurückgemeldet. Ein lesender Zugriff auf die Subindizes 3 bis 5 liefert immer den Wert 0 (FALSE), der keine weitere Bedeutung hat.
Subindex 0	**SubIndex 000**
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	5
Subindex 1	**Setting: Module scan provides detected modules**
Kurzbeschreibung	Einstellung, dass für einen vom EtherCAT-Konfigurationstool durchgeführten Modul- bzw. Gerätescan die *detektierten Module* bereitgestellt werden
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 16)
Zugriff	lesend und schreibend
Online-Wert	0 (FALSE) = deaktiviert (Standardwert, wenn eine *gespeicherte Modulkonfiguration* vorhanden ist) 1 (TRUE) = aktiviert (Standardwert, wenn keine *gespeicherte Modulkonfiguration* vorhanden ist)
Weitere Details	Konkret wird durch diese Einstellung festgelegt, dass im Objekt 0xF050 (Detected Module Ident List) die *detektierten Module* enthalten sind. Durch ein Beschreiben mit 1 (TRUE) kann die Einstellung aktiviert werden, wobei die Einstellung von Subindex 2 dann automatisch deaktiviert wird.

Index 0x2110	Module Configuration Control
Subindex 2	**Setting: Module scan provides stored modules**
Kurzbeschreibung	Einstellung, dass für einen vom EtherCAT-Konfigurationstool durchgeführten Modul- bzw. Gerätescan die *gespeicherten Module* bereitgestellt werden
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 17)
Zugriff	lesend und schreibend
Online-Wert	0 (FALSE) = deaktiviert (Standardwert, wenn keine *gespeicherte Modulkonfiguration* vorhanden ist) 1 (TRUE) = aktiviert (Standardwert, wenn eine *gespeicherte Modulkonfiguration* vorhanden ist)
Weitere Details	Konkret wird durch diese Einstellung festgelegt, dass im Objekt 0xF050 (Detected Module Ident List) die *gespeicherten Module* enthalten sind. Durch ein Beschreiben mit 1 (TRUE) bei Vorhandensein einer *gespeicherten Modulkonfiguration* kann die Einstellung aktiviert werden, wobei die Einstellung von Subindex 1 dann automatisch deaktiviert wird. Ein Beschreiben mit 1 (TRUE) ohne Vorhandsein einer *gespeicherten Modulkonfiguration* führt zum Abbruch mit SDO-Abort-Code 0x08000000 (General Error).
Subindex 3	**Command: Reload stored parameters**
Kurzbeschreibung	Kommando zum Wiederherstellen der Modulparameter in PREOP
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 18)
Zugriff	lesend (immer) und schreibend (nur in PREOP)
Online-Wert	0 (FALSE) = keine Bedeutung
Weitere Details	Durch ein Beschreiben mit dem Wert 1 (TRUE) bei Vorhandensein einer *gespeicherten Modulkonfiguration* wird das Kommando ausgelöst. Durch das Kommando werden die *gespeicherten Parameterdaten* für diejenigen *konfigurierten Module* wiederhergestellt, die mit den *gespeicherten Modulen* übereinstimmen. Bei Modulen, die nicht mit den *gespeicherten Modulen* übereinstimmen, werden keine Parameteränderungen vorgenommen. Ein Beschreiben mit 1 (TRUE) ohne Vorhandsein einer *gespeicherten Modulkonfiguration* führt zum Abbruch mit SDO-Abort-Code 0x08000000 (General Error).
Subindex 4	**Command: Store current configuration**
Kurzbeschreibung	Kommando zum Speichern der *aktuellen Modulkonfiguration* in PREOP
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 19)
Zugriff	lesend (immer) und schreibend (nur in PREOP)
Online-Wert	0 (FALSE) = keine Bedeutung
Weitere Details	Das Kommando wird durch Beschreiben mit dem Wert 1 (TRUE) ausgelöst. Als Vorbedingung muss die Liste der *konfigurierten Module* gültig sein und jedes *konfigurierte Modul* muss entweder mit dem *gesteckten* oder dem *gespeicherten Modul* des gleichen Steckplatzes übereinstimmen. Wenn diese Vorbedingungen nicht erfüllt sind, wird der SDO-Abort-Code 0x08000000 (General Error) zurückgemeldet. Mit dem Kommando wird die *aktuelle Modulkonfiguration* als *gespeicherte Modulkonfiguration* stromausfallsicher im Buskoppler hinterlegt. Eine zuvor vorhandene *gespeicherte Modulkonfiguration* wird dadurch überschrieben.
Subindex 5	**Command: Erase stored configuration**
Kurzbeschreibung	Kommando zum Löschen der *gespeicherten Modulkonfiguration*
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 20)
Zugriff	lesend (immer) und schreibend (nur in PREOP)
Online-Wert	0 (FALSE) = keine Bedeutung
Weitere Details	Das Kommando wird durch Beschreiben mit dem Wert 1 (TRUE) ausgelöst. Danach ist dann keine *gespeicherte Modulkonfiguration* mehr vorhanden.

9.6.9 Objekt 0x2120 - Detected Module List (0xF050) Status

Index 0x2120	Detected Module List (0xF050) Status
Kurzbeschreibung	Status zur Liste der *detektierten Module* im Objekt 0xF050 (↗ 9.6.14)
Datentyp (Bitgröße)	RECORD (24)
Zugriff via Complete-Access	möglich
Subindex 0	**SubIndex 000**
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	4
Subindex 1	**Valid (not empty, no gaps)**
Kurzbeschreibung	zeigt an, ob die Liste der *detektierten Module* gültig, also nicht leer und ohne Lücke ist
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 16)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Ein State-Übergang von PREOP nach SAFEOP ist nur dann möglich, wenn die Liste der *detektierten Module* gültig ist.
Subindex 2	**Matches configured modules**
Kurzbeschreibung	zeigt an, ob die *detektierten* mit den *konfigurierten Modulen* übereinstimmen
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 17)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Ein State-Übergang von PREOP nach SAFEOP ist nur dann möglich, wenn die *detektierten* mit den *konfigurierten Modulen* übereinstimmen.
Subindex 3	**Matches stored modules**
Kurzbeschreibung	zeigt an, ob die *detektierten Module* mit den Modulen aus einer vorhandenen *gespeicherten Modulkonfiguration* übereinstimmen
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 18)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Sofern eine *gespeicherte Modulkonfiguration* vorhanden ist, ist ein State-Übergang von PREOP nach SAFEOP nur dann möglich, wenn die *detektierten* mit den *gespeicherten Modulen* übereinstimmen.
Subindex 4	**Overlaid with stored modules**
Kurzbeschreibung	zeigt an, ob das Objekt 0xF050 (Detected Module Ident List) anstelle der *detektierten*, die Module aus einer vorhandenen *gespeicherten Modulkonfiguration* auflistet
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 19)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Ob die *detektierten* oder die *gespeicherten Module* im Objekt 0xF050 aufgelistet werden, hängt davon ab, ob aktuell die Einstellung in 0x2110:01 oder in 0x2110:02 aktiviert ist (↗ 9.6.8). Standardmäßig werden bei Vorhandensein einer *gespeicherten Modulkonfiguration* die *gespeicherten Module* aufgelistet, ansonsten die *detektierten*. Unabhängig davon beziehen sich die Angaben in den Subindizes 1 bis 3 immer auf die *detektierten Module*!

9.6.10 Objekt 0x2130 - Configured Module List (0xF030) Status

Index 0x2130	Configured Module List (0xF030) Status
Kurzbeschreibung	Status zur Liste der *konfigurierten Modulen* im Objekt 0xF030 (↗ 9.6.13)
Datentyp (Bitgröße)	RECORD (24)
Zugriff via Complete-Access	möglich
Weitere Details	Die Subindizes 1 bis 6 stehen in direktem Zusammenhang mit den Subindizes 2 bis 7 von Objekt 0x2100 (Module Configuration Status, ↗ 9.6.7).
Subindex 0	**SubIndex 000**
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	6
Subindex 1	**Preconfigured with detected modules**
Kurzbeschreibung	zeigt an, ob die Liste der *konfigurierten Module* im Objekt 0xF030 mit den *detektierten Modulen* vorkonfiguriert ist
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 16)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Objekt 0xF030 ist immer dann mit den *detektierten Modulen* vorkonfiguriert, wenn keine *gespeicherte Modulkonfiguration* vorhanden ist und der Master seit dem letzten State-Übergang INIT→PREOP (noch) keine Liste mit *projektierten Modulen* übermittelt hat. Solange das der Fall ist, werden in PREOP auftretende Änderungen an den *detektierten Modulen* automatisch nachgeführt.
Subindex 2	**Preconfigured with stored modules**
Kurzbeschreibung	zeigt an, ob die Liste der *konfigurierten Module* mit den Modulen aus einer vorhandenen *gespeicherten Modulkonfiguration* vorkonfiguriert ist
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 17)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Objekt 0xF030 ist immer dann mit den *gespeicherten Modulen* vorkonfiguriert, wenn eine *gespeicherte Modulkonfiguration* vorhanden ist und der Master seit dem letzten State-Übergang INIT→PREOP (noch) keine Liste mit *projektierten Modulen* übermittelt hat.
Subindex 3	**Written by master with projected modules**
Kurzbeschreibung	zeigt an, dass die Liste der *konfigurierten Module* vom Master mit den *projektierten Modulen* beschrieben wurde
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 18)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Standardmäßig ist der Buskoppler so parametriert (↗ 7.5.2.2), dass ein State-Übergang von PREOP nach SAFEOP nur dann möglich ist, wenn der Master das Objekt 0xF030 zuvor mit den *projektierten Modulen* beschrieben hat.
Subindex 4	**Valid (not empty, no gaps)**
Kurzbeschreibung	zeigt an, ob die Liste der *konfigurierten Module* gültig ist (nicht leer und ohne Lücke)
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 19)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Ein State-Übergang von PREOP nach SAFEOP ist nur mit gültiger Modulliste möglich.

Index 0x2130	Configured Module List (0xF030) Status
Subindex 5	**Matches detected modules**
Kurzbeschreibung	zeigt an, ob die *konfigurierten* mit den *detektierten Modulen* übereinstimmen
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 20)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Ein State-Übergang von PREOP nach SAFEOP ist nur möglich, wenn die *konfigurierten* mit den *detektierten Modulen* übereinstimmen.
Subindex 6	**Matches stored modules**
Kurzbeschreibung	zeigt an, ob die *konfigurierten Module* mit den Modulen aus einer vorhandenen *gespeicherten Modulkonfiguration* übereinstimmen
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 21)
Zugriff	nur lesend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	Wenn eine *gespeicherte Modulkonfiguration* vorhanden ist, dann ist ein State-Übergang von PREOP nach SAFEOP nur möglich, wenn die *konfigurierten* mit den *gespeicherten Modulen* übereinstimmen.

9.6.11 Objekt 0x3000 - Coupler Parameters

Index 0x3000	Coupler Parameters
Kurzbeschreibung	Parameter des Buskopplers
Datentyp (Bitgröße)	RECORD (24)
Zugriff via Complete-Access	möglich
Weitere Details	Die beiden Parameter des Buskopplers sind in Kapitel 7.5.2 beschrieben. Einstellungen an den Parametern lassen sich in PREOP, SAFEOP und OP durchführen und werden automatisch stromausfallsicher gespeichert.
Subindex 0	**SubIndex 000**
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	2
Subindex 1	**Hot-swapping allowed**
Kurzbeschreibung	legt fest, ob im laufenden Betrieb (SAFEOP/OP) das Tauschen eines Moduls (Hot-Swap, ↗ 7.6) erlaubt ist
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 16)
Zugriff	lesend und schreibend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	siehe Kapitel 7.5.2.1
Subindex 2	**Master need not provide projected modules**
Kurzbeschreibung	legt fest, ob ein State-Übergang von PREOP nach SAFEOP auch ohne Übermittlung projektierter Module vom Master möglich sein soll
Datentyp (Bitgröße/Bitoffset)	BOOL (2 / 16)
Zugriff	lesend und schreibend
Online-Wert	0 (FALSE) = nein; 1 (TRUE) = ja
Weitere Details	siehe Kapitel 7.5.2.2

9.6.12 Objekt 0xF000 - Modular Device Profile

Index 0xF000	Modular Device Profile
Kurzbeschreibung	Kenndaten zum verwendeten Modular Device Profile (MDP)
Datentyp (Bitgröße)	RECORD (48)
Zugriff via Complete-Access	möglich
Subindex 0	**SubIndex 000**
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	2
Subindex 1	**Module index distance**
Kurzbeschreibung	Index-Abstand zwischen den Modulobjekten zweier aufeinanderfolgender Slots
Datentyp (Bitgröße/Bitoffset)	UINT (16 / 16)
Zugriff	nur lesend
Online-Wert	0x0010 (16)
Weitere Details	Der Index-Abstand wird bei folgenden Modulobjekten angewendet, wobei *MN* als Platzhalter für die zweitstellige hexadezimale Modulnummer steht: • Objekte zu den Eingangsdaten (0x6*MN*0) und Ausgangsdaten (0x7*MN*0) • Objekte zum Modulstatus (0xA*MN*0), Kanalstatus (0xA*MN*1) und Diagnose-status (0xA*MN*2)
Subindex 2	**Maximum number of modules**
Kurzbeschreibung	Maximale Anzahl der Module
Datentyp (Bitgröße/Bitoffset)	UINT (16 / 32)
Zugriff	nur lesend
Online-Wert	64 = an den Buskoppler können bis zu 64 Module gesteckt werden

9.6.13 Objekt 0xF030 - Configured Module Ident List

Index 0xF030	Configured Module Ident List
Kurzbeschreibung	Liste der *konfigurierten Module*
Datentyp (Bitgröße)	ARRAY OF UDINT (2064)
Zugriff via Complete-Access	möglich
Weitere Details	Das Objekt enthält in den Subindizes 1 bis maximal 64 die Modul-Ident-Werte (↗ 7.4.2) der *konfigurierten Module*. Die Subindizes 1 bis 64 entsprechen dabei den Steckplätzen bzw. Modulnummern 1 bis 64. Der Modul-Ident-Wert 0x00000000 steht für 'kein Modul konfiguriert' und entspricht somit einer Modullücke. Die Liste der *konfigurierten Module* ist nur dann gültig, wenn mindestens ein Modul konfiguriert und keine Modullücke vorhanden ist. Das Objekt ist bei Vorhandensein einer *gespeicherten Modulkonfiguration* mit den *gespeicherten*, ansonsten mit den *detektierten Modulen* vorkonfiguriert. Es kann vom Master mit den *projektierten Modulen* beschrieben werden. Das Objekt 0x2130 (Configured Module Ident List Status, ↗ 9.6.10) enthält wesentliche Status-Informationen zu diesem Objekt.
Subindex 0	**SubIndex 000**
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes = Anzahl der konfigurierten Steckplätze/ Module
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	lesend (immer) und schreibend (nur PREOP)
Online-Wert	z.B. 10 = Liste umfasst 10 *konfigurierte Module* für die Steckplätze 1 bis 10 in den Subindizes 1 bis 10
Weitere Details	es können maximal 64 Module konfiguriert werden

Index 0xF030	Configured Module Ident List
Subindex 1	**SubIndex 001**
Kurzbeschreibung	Modul-Ident-Wert des Moduls für den ersten Steckplatz
Datentyp (Bitgröße/Bitoffset)	UDINT (32 / 16)
Zugriff	lesend (immer) und schreibend (nur PREOP)
Online-Wert	z.B. 0x07080000 (8 DI Modul)
Subindex 2	**SubIndex 002**
Kurzbeschreibung	Modul-Ident-Wert des Moduls für den zweiten Steckplatz
Datentyp (Bitgröße/Bitoffset)	UDINT (32 / 48)
Zugriff	lesend (immer) und schreibend (nur PREOP)
Online-Wert	z.B. 0x0AF00000 (8 DO Modul)
...	...

9.6.14 Objekt 0xF050 - Detected Module Ident List

Index 0xF050	Detected Module Ident List
Kurzbeschreibung	Liste der *detektierten*, also der *gesteckten Module*
Datentyp (Bitgröße)	ARRAY OF UDINT (2064)
Zugriff via Complete-Access	möglich
Weitere Details	Das Objekt enthält in den Subindizes 1 bis maximal 64 die Modul-Ident-Werte (↗ 7.4.2) der *detektierten Module*. Ist allerdings eine *gespeicherte Modulkonfiguration* im Buskoppler vorhanden, dann wird die Liste standardmäßig mit den *gespeicherten Modulen* überlagert, was durch den Objekteintrag 0x2120:04 (Overlaid with stored modules,↗ 9.6.9) angezeigt wird. Die im Objekt 0xF050 enthaltenen Module können vom EtherCAT-Konfigurationstool im Rahmen eines Modul- bzw. Geräte-Scans ausgelesen und als *projektierte Module* übernommen werden. Mithilfe der Subindizes 1 und 2 des Objektes 0x2110 (Module Configuration Control, ↗ 9.6.8) kann manuell festgelegt werden, ob das Objekt 0xF050 die Modul-Ident-Werte der *detektierten* oder der *gespeicherten Module* enthalten soll. Die Liste der *detektierten Module* ist nur dann gültig, wenn mindestens ein Modul enthalten und keine Modullücke vorhanden ist. Eine Modullücke wird durch den Modul-Ident-Wert 0x00000000 repräsentiert. Die Subindizes 1 bis 64 entsprechen den Steckplätzen bzw. Modulnummern 1 bis 64. Dieser Zusammenhang gilt möglicherweise nicht mehr für Steckplätze, die sich hinter eine Modullücke befinden, da die tatsächliche Größe einer Modullücke nicht immer detektiert werden kann. Das Objekt 0x2120 (Detected Module Ident List Status, ↗ 9.6.9) enthält wesentliche Status-Informationen zu diesem Objekt.
Subindex 0	**SubIndex 000**
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes = (Anzahl der *detektierten Module* + Anzahl detektierter Modullücken) bzw. Anzahl der *gespeicherten Module*
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	z.B. 10 = 10 Module detektiert
Weitere Details	Die Liste kann maximal 64 Module umfassen.
Subindex 1	**SubIndex 001**
Kurzbeschreibung	Modul-Ident-Wert des *detektierten* bzw. *gespeicherten Moduls* im ersten Steckplatz
Datentyp (Bitgröße/Bitoffset)	UDINT (32 / 16)
Zugriff	nur lesend
Online-Wert	z.B. 0x07080000 (8 DI Modul)

Index 0xF050	Detected Module Ident List
Subindex 2	**SubIndex 002**
Kurzbeschreibung	Modul-Ident-Wert des *detektierten* bzw. *gespeicherten Moduls* im zweiten Steckplatz
Datentyp (Bitgröße/Bitoffset)	UDINT (32 / 48)
Zugriff	nur lesend
Online-Wert	z.B. 0x0AF00000 (8 DO Modul)
...	...

9.7 Die Modul-Objekte

Die Modul-Objekte werden dynamisch in das Objektverzeichnis eingeblendet, je nachdem, welche Module *projektiert* bzw. *konfiguriert* sind. Welche Modul-Objekte für ein bestimmtes Modul eingeblendet werden, ist von dessen Funktionsumfang abhängig. In der nachfolgenden Tabelle sind die unterschiedlichen Modul-Objekte mit einer kurzen Beschreibung aufgelistet.

Modul-Objekt	Beschreibung
Parameterdaten-Objekt (↗ 9.7.3)	Dieses Objekt enthält die Parameterdaten eines Moduls und ist daher nur für parametrierbare Module vorhanden. Zum Parametrieren bzw. Umparametrieren eines Moduls lässt sich dieses Objekt in PREOP, SAFEOP und OP beschreiben.
Eingangsdaten-Objekt (↗ 9.7.4)	Dieses Objekt enthält die Eingangsdaten eines Moduls und ist daher nur für Module mit Eingangsdaten vorhanden. Für einen Zugriff auf die Eingangsdaten wird es nicht benötigt, da diese auch zyklusaktuell über die Prozesseingangsdaten des Buskopplers bereitgestellt werden.
Ausgangsdaten-Objekt (↗ 9.7.5)	Dieses Objekt enthält die Ausgangsdaten eines Moduls und ist daher nur für Module mit Ausgangsdaten vorhanden. Ein schreibender Zugriff auf die Ausgangsdaten über dieses Objekt ist nicht erlaubt und auch nicht erforderlich, da diese über die Prozessausgangsdaten des Buskopplers bereitgestellt werden.
Modulstatus-Objekt (↗ 9.7.6)	Dieses Objekt enthält den Modulstatus und ist für alle Module vorhanden. Für einen Zugriff auf den Modulstatus wird es nicht benötigt, da dieser auch zyklusaktuell über die Prozesseingangsdaten des Buskopplers bereitgestellt wird.
Kanalstatus-Objekt (↗ 9.7.7)	Dieses Objekt enthält den Kanalstatus eines Moduls und ist daher nur für diagnosefähige Module mit E/A-Daten vorhanden. Für einen Zugriff auf den Kanalstatus wird es nicht benötigt, da dieser auch zyklusaktuell über die Prozesseingangsdaten des Buskopplers bereitgestellt wird.
Diagnosestatus-Objekt (↗ 9.7.8)	Dieses Objekt enthält den Diagnosestatus eines Moduls und ist daher nur für diagnosefähige Module vorhanden. Der Zugriff auf den Diagnosestatus eines Moduls muss über dieses Objekt erfolgen.

9.7.1 Index der Modul-Objekte

Die nachfolgende Tabelle listet auf, in welche Bereiche des Objektverzeichnisses die unterschiedlichen Modul-Objekte eingeblendet werden und wie sich der Index eines solchen Objektes für ein bestimmtes Modul berechnen lässt.

Modul-Objekt	Bereich	Indexberechnung
Parameterdaten-Objekt	0x3010 – 0x3400	0x3000 + 0x10 · <Modulnummer>
Eingangsdaten-Objekt	0x6010 – 0x6400	0x6000 + 0x10 · <Modulnummer>
Ausgangsdaten-Objekt	0x7010 – 0x7400	0x7000 + 0x10 · <Modulnummer>
Modulstatus-Objekt	0xA010 – 0xA400	0xA000 + 0x10 · <Modulnummer>
Kanalstatus-Objekt	0xA011 – 0xA401	0xA001 + 0x10 · <Modulnummer>
Diagnosestatus-Objekt	0xA012 – 0xA402	0xA002 + 0x10 · <Modulnummer>

Die zur Indexberechnung verwendete Modulnummer entspricht der Steckplatznummer und liegt im Bereich von 1 bis 64. Der Abstand zwischen gleichartigen Modul-Objekten direkt aufeinanderfolgender Module wird als 'Module Index Distance' bezeichnet und ist im Objekt 0xF000 (Modular Device

Profile, ↗ 9.6.12) angegeben. Beim TB20 EtherCAT® Buskoppler wird ein fester Index-Abstand von 0x10 (=16) verwendet.

9.7.2 Modul-Objekte und deren Indizes für eine beispielhafte Modulauswahl

Die nachfolgende Tabelle zeigt für eine beispielhafte Modulauswahl, welche Modul-Objekte zur Verfügung stehen und über welche Indizes sie adressiert werden können.

Modulnummer	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Modul (Bestellnummer)	**PS 24V** (600-700-0AA01)	**8 DI** (600-210-0AH01)	**4 DO** (600-220-0AD01)	**8 DI/8 DO** (600-230-0AP21)	**4 DO HF** (600-220-7AD01)	**PWR 24V** (600-710-0AA01)	**2 AI U** (600-252-4AB01)	**4 AO U** (600-261-4AD01)
Parameterdaten	0x3010	-	-	-	0x3050	-	0x3070	0x3080
Eingangsdaten	-	0x6020	-	0x6040	0x6050	-	0x6070	-
Ausgangsdaten	-	-	0x7030	0x7040	0x7050	-	-	0x7080
Modulstatus	0xA010	0xA020	0xA030	0xA040	0xA050	0xA060	0xA070	0xA080
Kanalstatus	-	-	-	-	0xA051	-	0xA071	0xA081
Diagnosestatus	0xA012	-	-	-	0xA052	-	0xA072	0xA082

9.7.3 Die Parameterdaten-Objekte

In das Objektverzeichnis des Buskopplers wird für jedes *konfigurierte Modul*, das parametrierbar ist, ein entsprechendes Parameterdaten-Objekt eingeblendet. Voraussetzung dafür ist, dass des jeweilige Modul am Rückwandbus steckt oder mit dem *gespeicherten Modul* einer vorhandenen *gespeicherten Modulkonfiguration* übereinstimmt. Nur in diesen beiden Fällen verfügt der Buskoppler über die erforderlichen Informationen zu den Parameterdaten.

Die Parameterdaten-Objekte haben den Datentyp RECORD. Die einzelnen Objekteinträge innerhalb eines solchen Objektes werden dynamisch anhand des Aufbaus der Parameterdaten generiert. Der Aufbau der Parameterdaten kann im entsprechenden Modulhandbuch nachgeschlagen werden.

Subindex 0 eines Parameterdaten-Objektes enthält die Anzahl *n* der nachfolgenden Subindizes. Auf die Subindizes 1 bis *n* sind die einzelnen Parameterdaten-Elemente wie folgt abgebildet:

- Die Abbildung findet geordnet nach aufsteigender Position innerhalb der Parameterdaten statt.
- Parameterdaten-Elemente, die 1, 2 oder 4 Bytes umfassen, werden jeweils auf einen eigenen Subindex abgebildet. Der Datentyp (↗ 9.4) des Subindizes richtet sich dabei nach Bitgröße (8, 16 oder 32) und Art (mit oder ohne Vorzeichen) des enthaltenen Parameterdaten-Elements.
- Parameterdaten-Elemente, die nur einzelne Bits umfassen und in einem gemeinsamen Parameterdaten-Byte enthalten sind, werden zusammengefasst auf einen Subindex vom Datentyp BYTE abgebildet.
- Beim Zugriff via Complete-Access entsprechen die gelesenen bzw. geschriebenen Objektdaten ab Subindex 1 genau den im jeweiligen Modulhandbuch beschriebenen Parameterdaten.

Auf Parameterdaten-Objekte kann in den EtherCAT-States PREOP, SAFEOP und OP lesend und schreibend zugegriffen werden. Im EtherCAT-State PREOP vorgenommene Änderungen an den Parameterdaten-Objekten werden vorerst nur in die *aktuelle Modulkonfiguration* übernommen. Die Übermittlung der geänderten Parameterdaten an die jeweiligen Module findet erst beim State-Übergang von PREOP nach SAFEOP statt, wo die *aktuelle* als *aktive Modulkonfiguration* übernommen wird.

Änderungen an Parameterdaten-Objekten, die in den EtherCAT-States SAFEOP- und OP mittels Complete-Access durchgeführt werden, werden sofort in die aktive Modulkonfiguration übernommen und direkt an die jeweiligen Module übermittelt. Werden die Änderungen nicht via Complete-

Access sondern Subindex-weise vorgenommen, dann muss abschließend noch der Subindex 0 mit dem bereits enthaltenen Wert beschrieben werden, damit sich die Änderungen auswirken.

9.7.3.1 Allgemeiner Aufbau des Parameterdaten-Objekts

Parameterdaten-Objekt	
Index-Bereich	0x3010 - 0x3400 (herstellerspezifisch)
Index-Berechnung	0x3000 + 0x10 · <Modulnummer>
Offline-Bezeichnung	"Parameters (<Kurzkennung> - <Bestellnummer>)"
Online-Bezeichnung	"Module <Modulnummer> - Parameters"
Datentyp	RECORD
Bitgröße	abhängig von der Anzahl der verfügbaren Subindizes und deren Datentypen
Zugriff via Complete-Access	möglich
Subindex 0	
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Offline/Online-Bezeichnung	"SubIndex 000"
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	lesend und schreibend
Offline/Online-Wert	abhängig vom Aufbau der Parameterdaten des Moduls
Weitere Details	Durch das Beschreiben von Subindex 0 mit dem bereits enthaltenen Wert kann in SAFEOP und OP die konsistente Übertragung zuvor geänderter Parameterdaten an das betreffende Modul angestoßen werden. Wird versucht, Subindex 0 mit einem anderen Wert zu beschreiben, dann wird der Zugriff mit SDO-Abort-Code 0x06040043 abgelehnt.
Subindex <*s*>	
Kurzbeschreibung	Daten von Parameterdaten-Element <*e*> mit *e* = *s* - 1 (Nummerierung beginnt mit 0 in Subindex 1)
Offline-Bezeichnung	Es wird eine parameterspezifische Bezeichnung verwendet, die aus der Modulbeschreibung in der ESI-Datei stammt. Handelt es sich um ein zusammengefasstes Parameterdaten-Element vom Datentyp BYTE, dann wird abweichend die generische Bezeichnung "Parameter byte [<Byte Offset>] verwendet. <Byte Offset> steht als Platzhalter für das Byte-Offset, also die Position innerhalb der Parameterdaten.
Online-Bezeichnung	"Parameter <*e*>"
Datentyp	abhängig von Bitgröße und Art des Parameterdaten-Elements
Bitoffset	beginnt bei Subindex 1 mit 16 und erhöht sich mit jedem weiteren Subindex um die Bitgröße des vorangegangenen Datentyps
Zugriff	lesend und schreibend
Offline-Wert	Default-Wert laut Modulbeschreibung in der ESI-Datei
Online-Wert	aktueller Wert des Parameters
…	

9.7.3.2 Konkreter Aufbau anhand eines Beispiels

Als Beispiel wird das Modul 'AO 2x I, 0/4-20 mA, 12 Bit' (600-260-4AB01) verwendet. Laut "Handbuch TB20 Baugruppen" sind die Parameterdaten wie folgt aufgebaut:

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Diagnose-alarm	-	-	1	-	-	Werte Darstellung	
1	Drahtbruch-erkennung Kanal 0	Verhalten bei Stopp Kanal 0			Kanal 0 Ausgabebereich			
2	Ersatzwert Kanal 0							
3								
4	Drahtbruch-erkennung Kanal 1	Verhalten bei Stopp Kanal 1			Kanal 1 Ausgabebereich			
5	Ersatzwert Kanal 1							
6								

Davon abgeleitet ergibt sich folgender Aufbau des Parameterdaten-Objektes für das Modul:

Index 0x3050	
Kurzbeschreibung	Parameterdaten-Objekt von Module 5
Offline-Bezeichnung	"Parameters (2 AO I - 600-260-4AB01)"
Online-Bezeichnung	"Module 5 - Parameters"
Datentyp (Bitgröße)	RECORD (72)
Subindex 0	
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Offline-/Online-Bezeichnung	"SubIndex 000"
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Subindex 1	
Kurzbeschreibung	entspricht den in Byte 0 zusammengefassten Modulparametern
Offline-Bezeichnung	"Parameter byte [0]"
Online-Bezeichnung	"Parameter 0"
Datentyp (Bitgröße/Bitoffset)	BYTE (8 / 16)
Subindex 2	
Kurzbeschreibung	entspricht den in Byte 1 zusammengefassten Modulparametern
Offline-Bezeichnung	"Parameter byte [1]"
Online-Bezeichnung	"Parameter 1"
Datentyp (Bitgröße/Bitoffset)	BYTE (8 / 24)
Subindex 3	
Kurzbeschreibung	entspricht dem in Byte 2 und 3 enthaltenen "Ersatzwert Kanal 0"
Offline-Bezeichnung	"Channel 0: Substitute value"
Online-Bezeichnung	"Parameter 2"
Datentyp (Bitgröße/Bitoffset)	INT (16 / 32)
Subindex 4	
Kurzbeschreibung	entspricht dem in Byte 4 zusammengefassten Modulparametern
Offline-Bezeichnung	"Parameter byte [4]"
Online-Bezeichnung	"Parameter 3"
Datentyp (Bitgröße/Bitoffset)	BYTE (8 / 48)

Index 0x3050	
Subindex 5	
Kurzbeschreibung	entspricht dem in Byte 5 und 6 enthaltenen "Ersatzwert Kanal 1"
Offline-Bezeichnung	"Channel 1: Substitute value"
Online-Bezeichnung	"Parameter 4"
Datentyp (Bitgröße/Bitoffset)	INT (16 / 56)

9.7.4 Die Eingangsdaten-Objekte der Module

In das Objektverzeichnis des Buskopplers wird für jedes *konfigurierte Modul*, das über Eingangsdaten verfügt, ein entsprechendes Eingangsdaten-Objekt eingeblendet. Voraussetzung dafür ist, dass das jeweilige Modul am Rückwandbus steckt, da der Buskoppler nur dann die erforderlichen Informationen zu den Eingangsdaten vom Modul abrufen kann.

Die Eingangsdaten-Objekte haben den Datentyp RECORD. Die einzelnen Objekteinträge innerhalb eines solchen Objektes werden dynamisch anhand des Aufbaus der Eingangsdaten generiert. Der Aufbau der Eingangsdaten kann im entsprechenden Modulhandbuch nachgeschlagen werden.

Subindex 0 eines Eingangsdaten-Objektes enthält die Anzahl *n* der nachfolgenden Subindizes. Auf die Subindizes 1 bis *n* sind die einzelnen Eingangsdaten-Elemente wie folgt abgebildet:

- Die Abbildung findet geordnet nach aufsteigender Position innerhalb der Eingangsdaten statt.
- Eingangsdaten-Elemente, die 1, 2 oder 4 Bytes umfassen, werden jeweils auf einen eigenen Subindex abgebildet. Der Datentyp (↗ 9.4) des Subindizes richtet sich dabei nach Bitgröße (8, 16 oder 32) und Art (mit oder ohne Vorzeichen) des enthaltenen Eingangsdaten-Elements.
- Eingangsdaten-Elemente, die nur einzelne Bits umfassen und in einem gemeinsamen Eingangsdaten-Byte enthalten sind, werden zusammengefasst auf einen Subindex vom Datentyp BYTE abgebildet.
- Bei digitalen Modulen werden die Eingangsdaten-Elemente, welche die Zustände der digitalen Eingangskanäle enthalten, bitgenau aufgelöst und jeweils auf einen eigenen Subindex vom Datentyp BOOL abgebildet. Die Abbildung findet dabei geordnet mit aufsteigender Byte- und Bit-Position innerhalb der Eingangsdaten statt.

Auf Eingangsdaten-Objekte kann in PREOP, SAFEOP und OP lesend zugegriffen werden. Aktuelle Eingangsdaten werden von den Modulen aber nur in SAFEOP und OP zur Verfügung gestellt. In PREOP werden für die enthaltenen Eingangsdaten Nullwerte zurückgeliefert.

Im Normalfall gibt es keinen Grund, über diese Objekte auf die Eingangsdaten der Module zuzugreifen, da in SAFEOP und OP die Eingangsdaten der Module auch über die zyklusaktuellen Prozesseingangsdaten des Buskopplers zur Verfügung gestellt werden (↗ 7.4.5). Die Eingangsdaten-Objekte werden vielmehr im Rahmen der PDO-Konfiguration als Referenzobjekte für die Abbildung der Eingangsdaten auf TxPDOs (↗ 9.8.3) benötigt.

9.7.4.1 Allgemeiner Aufbau des Objekts

Eingangsdaten-Objekt	
Index-Bereich	0x6010 - 0x6400 (MDP-spezifisch)
Index-Berechnung	0x6000 + 0x10 · <Modulnummer>
Offline-Bezeichnung	"Inputs (<Kurzkennung> - <Bestellnummer>)"
Online-Bezeichnung	"Module <Modulnummer> - Inputs"
Datentyp	RECORD
Bitgröße	• ist abhängig von der Anzahl und vom Datentyp der verfügbaren Subindizes • wird auf ein Vielfaches von 8 aufgerundet
Zugriff via Complete-Access	nicht möglich
Subindex 0	
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Offline/Online-Bezeichnung	"SubIndex 000"
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	abhängig vom Aufbau der Eingangsdaten des Moduls
…	
Subindex <*s*>	
Kurzbeschreibung	Daten von Eingangsdaten-Element <*e*> mit *e* = *s* - 1 (Nummerierung beginnt mit 0 in Subindex 1)
Offline-Bezeichnung	Es wird eine Datenelement-spezifische Bezeichnung laut Modulbeschreibung in der ESI-Datei verwendet. Handelt es sich um ein zusammengefasstes Eingangsdaten-Element vom Datentyp BYTE, dann wird abweichend eine der folgenden generischen Bezeichnungen verwendet: • "Input byte [<Byte-Offset>]" (enthält allgemein Eingangsdaten) • "Status byte [<Byte-Offset>]" (enthält Status-Informationen) • "Reserved byte [<Byte-Offset>" (reserviert, ohne Funktion) <Byte Offset> steht als Platzhalter für das Byte-Offset, also die Position innerhalb der Eingangsdaten.
Online-Bezeichnung	"Input <*e*>"
Datentyp	abhängig von Bitgröße und Art des Eingangsdaten-Elements
Bitoffset	beginnt bei Subindex 1 mit 16 und erhöht sich mit jedem weiteren Subindex um die Bitgröße des vorangegangenen Datentyps
Zugriff	nur lesend
Online-Wert	in PREOP: immer 0 in SAFEOP und OP: aktueller Wert aus dem Modul
…	

9.7.4.2 Konkreter Aufbau des Objekts anhand zweier Beispiele

Beispiel 1 basierend auf dem digitalen Eingangsmodul 'DI 4 x DC 24 V' (600-210-0AD01)

Die Eingangsdaten des Moduls sind wie folgt aufgebaut:

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	-	-	-	-	DI 3 (Eingangs-zustand von Kanal 3)	DI 2 (Eingangs-zustand von Kanal 2)	DI 1 (Eingangs-zustand von Kanal 1)	DI 0 (Eingangs-zustand von Kanal 0)

Der Aufbau des entsprechenden Eingangsdaten-Objekts für das Modul verdeutlicht die bitgenaue Auflösung digitaler Eingangsdaten:

Index 0x60B0	
Kurzbeschreibung	Eingangsdaten-Objekt von Modul 11
Offline-Bezeichnung	"Inputs (4 DI - 600-210-0AD01)"
Online-Bezeichnung	"Module 11 - Inputs"
Datentyp (Bitgröße)	RECORD (24)
Subindex 0	
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Offline-/Online-Bezeichnung	"SubIndex 000"
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Subindex 1	
Kurzbeschreibung	entspricht dem Eingangsdaten-Element 'DI 0' in Bit 0 von Eingangsdaten-Byte 0
Offline-Bezeichnung	"Input 0"
Online-Bezeichnung	"Input 0"
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 16)
Subindex 2	
Kurzbeschreibung	entspricht dem Eingangsdaten-Element 'DI 1' in Bit 1 von Eingangsdaten-Byte 0
Offline-Bezeichnung	"Input 1"
Online-Bezeichnung	"Input 1"
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 17)
Subindex 3	
Kurzbeschreibung	entspricht dem Eingangsdaten-Element 'DI 2' in Bit 2 von Eingangsdaten-Byte 0
Offline-Bezeichnung	"Input 2"
Online-Bezeichnung	"Input 2"
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 18)
Subindex 4	
Kurzbeschreibung	entspricht dem Eingangsdaten-Element 'DI 3' in Bit 3 von Eingangsdaten-Byte 0
Offline-Bezeichnung	"Input 3"
Online-Bezeichnung	"Input 3"
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 19)

Beispiel 2 basierend auf dem 5V-Zählermodul (600-310-7AA01) in der Betriebsart "Zählbetrieb"

Aufbau der Eingangsdaten für den Zählbetrieb laut Handbuch des Zählermoduls:

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 - 3	Zählwert							
4	Kurzschluss Geber-versorgung	-	Parametrier-fehler	-	-	Rücksetzen Statusbits läuft	Fehler bei Lade-funktion	Lade-funktion läuft
5	Zählrichtung rückwärts	Zählrichtung vorwärts	-	-	-	-	-	Status SW-Tor
6	Nulldurch-gang	Untere Zählgrenze	Obere Zählgrenze	Status Ver-gleicher2	Status Ver-gleicher1	-	-	Status Synchroni-sation
7	-	-	-	-	-	-	-	-

Nachfolgend ist der Aufbau des entsprechenden Eingangsdaten-Objekts für das Modul dargestellt. Bei den Subindizes 2 bis 4 handelt es sich jeweils um zusammengefasste Eingangsdaten-Elemente vom Datentyp BYTE. Darin sind die in Byte 4 bis 6 enthaltenen Eingangsdaten-Bits zusammengefasst und in gleicher Anordnung abgebildet. Subindex 5 trägt die Offline-Bezeichnung "Reserved byte [5]", da das darin abgebildete Eingangsdaten-Byte aktuell ohne Funktion ist.

Index 0x6140	
Kurzbeschreibung	Eingangsdaten-Objekt von Modul 20
Offline-Bezeichnung	"Inputs (1CNT 5V - 600-310-7AA01)"
Online-Bezeichnung	"Module 20 - Inputs"
Datentyp (Bitgröße)	RECORD (80)
Subindex 0	
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Offline-/Online-Bezeichnung	"SubIndex 000"
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Subindex 1	
Kurzbeschreibung	entspricht dem Eingangsdaten-Element 'Zählwert' in den Eingangsdaten-Bytes 0 bis 3
Offline-Bezeichnung	"Counter value"
Online-Bezeichnung	"Input 0"
Datentyp (Bitgröße/Bitoffset)	BOOL (32 / 16)
Subindex 2	
Kurzbeschreibung	entspricht den zusammengefassten Eingangsdaten-Elementen in Eingangsdaten-Byte 4
Offline-Bezeichnung	"Input byte [4]"
Online-Bezeichnung	"Input 1"
Datentyp (Bitgröße/Bitoffset)	BOOL (8 / 48)
Subindex 3	
Kurzbeschreibung	entspricht den zusammengefassten Eingangsdaten-Elementen in Eingangsdaten-Byte 5
Offline-Bezeichnung	"Input byte [5]"
Online-Bezeichnung	"Input 2"
Datentyp (Bitgröße/Bitoffset)	BOOL (8 / 56)
Subindex 4	
Kurzbeschreibung	entspricht den zusammengefassten Eingangsdaten-Elementen in Eingangsdaten-Byte 6
Offline-Bezeichnung	"Input byte [6]"
Online-Bezeichnung	"Input 3"
Datentyp (Bitgröße/Bitoffset)	BOOL (8 / 64)
Subindex 5	
Kurzbeschreibung	entspricht dem reservierten Eingangsdaten-Byte 7
Offline-Bezeichnung	"Reserved byte [7]"
Online-Bezeichnung	"Input 4"
Datentyp (Bitgröße/Bitoffset)	BOOL (8 / 72)

9.7.5 Die Ausgangsdaten-Objekte

In das Objektverzeichnis des Buskopplers wird für jedes *konfigurierte Modul*, das über Ausgangsdaten verfügt, ein entsprechendes Ausgangsdaten-Objekt eingeblendet. Voraussetzung dafür ist, dass das jeweilige Modul am Rückwandbus steckt, da der Buskoppler nur dann die erforderlichen Informationen zu den Ausgangsdaten vom Modul abrufen kann.

Die Ausgangsdaten-Objekte haben den Datentyp RECORD. Die einzelnen Objekteinträge innerhalb eines solchen Objektes werden dynamisch anhand des Aufbaus der Ausgangsdaten generiert. Der Aufbau der Ausgangsdaten kann im entsprechenden Modulhandbuch nachgeschlagen werden.

Subindex 0 eines Ausgangsdaten-Objektes enthält die Anzahl *n* der nachfolgenden Subindizes. Auf die Subindizes 1 bis *n* sind die einzelnen Ausgangsdaten-Elemente wie folgt abgebildet:

- Die Abbildung findet geordnet nach aufsteigender Position innerhalb der Ausgangsdaten statt.
- Ausgangsdaten-Elemente, die 1, 2 oder 4 Bytes umfassen, werden jeweils auf einen eigenen Subindex abgebildet. Der Datentyp (↗ 9.4) des Subindizes richtet sich dabei nach Bitgröße (8, 16 oder 32) und Art (mit oder ohne Vorzeichen) des enthaltenen Ausgangsdaten-Elements.
- Ausgangsdaten-Elemente, die nur einzelne Bits umfassen und in einem gemeinsamen Ausgangsdaten-Byte enthalten sind, werden zusammengefasst auf einen Subindex vom Datentyp BYTE abgebildet.
- Bei digitalen Modulen werden die Ausgangsdaten-Elemente, welche die Zustände der digitalen Ausgangskanäle enthalten, bitgenau aufgelöst und jeweils auf einen eigenen Subindex vom Datentyp BOOL abgebildet. Die Abbildung findet dabei geordnet mit aufsteigender Byte- und Bit-Position innerhalb der Ausgangsdaten statt.

Auf die Ausgangsdaten-Objekte kann in PREOP, SAFEOP und OP nur lesend zugegriffen werden. Ein schreibender Zugriff ist nicht möglich und auch nicht erforderlich, da das eigentliche Setzen der Ausgangsdaten der Module in SAFEOP und OP über die zyklusaktuellen Prozessausgangsdaten des Buskopplers stattfindet. Die Ausgangsdaten-Objekte werden stattdessen im Rahmen der PDO-Konfiguration als Referenzobjekte für die Abbildung der Ausgangsdaten auf RxPDOs (↗ 9.8.2) benötigt.

9.7.5.1 Allgemeiner Aufbau des Objekts

Ausgangsdaten-Objekt	
Index-Bereich	0x7010 - 0x7400 (MDP-spezifisch)
Index-Berechnung	0x7000 + 0x10 · <Modulnummer>
Offline-Bezeichnung	"Outputs (<Kurzkennung> - <Bestellnummer>)"
Online-Bezeichnung	"Module <Modulnummer> - Outputs"
Datentyp	RECORD
Bitgröße	• ist abhängig von der Anzahl und vom Datentyp der verfügbaren Subindizes • wird immer auf ein Vielfaches von 8 aufgerundet
Zugriff via Complete-Access	nicht möglich
Subindex 0	
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Offline/Online-Bezeichnung	"SubIndex 000"
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	abhängig vom Aufbau der Ausgangsdaten des Moduls
…	
Subindex <*s*>	
Kurzbeschreibung	Daten von Eingangsdaten-Element <*e*> mit *e* = *s* - 1 (Nummerierung beginnt mit 0 in Subindex 1)
Offline-Bezeichnung	Es wird eine Datenelement-spezifische Bezeichnung laut Modulbeschreibung in der ESI-Datei verwendet. Handelt es sich um ein zusammengefasstes Ausgangsdaten-Element vom Datentyp BYTE, dann wird abweichend eine der folgenden generischen Bezeichnungen verwendet: • "Output byte [<Byte-Offset>]" (enthält allgemein Ausgangsdaten) • "Reserved byte [<Byte-Offset>" (reserviert, ohne Funktion) <Byte Offset> steht als Platzhalter für das Byte-Offset, also die Position innerhalb der Ausgangsdaten.
Online-Bezeichnung	"Output <*e*>"
Datentyp	abhängig von Bitgröße und Art des Ausgangsdaten-Elements
Bitoffset	beginnt bei Subindex 1 mit 16 und erhöht sich mit jedem weiteren Subindex um die Bitgröße des vorangegangenen Datentyps

Zugriff	nur lesend
Online-Wert	in PREOP: immer 0 in SAFEOP / OP: über die Prozessausgangsdaten des Buskopplers gesetzte Werte Die hier enthaltenen Ausgangsdaten werden nur in OP zyklusaktuell an das jeweilige Modul übermittelt. In SAFEOP findet diese Übermittlung nicht statt und die Ausgänge des Moduls befinden sich im sicheren Zustand. Das heißt, in SAFEOP muss es keine Übereinstimmung zwischen den hier angezeigten Ausgangsdaten und dem sicheren Zustand der Ausgänge am Modul geben.
…	

Die Abbildung der Modul-Ausgangsdaten auf die Ausgangsdaten-Objekte findet analog zur Abbildung der Modul-Eingangsdaten auf die Eingangsdaten-Objekte statt. Deshalb sei an dieser Stelle auf die zwei konkreten Beispiele zum Aufbau der Eingangsdaten-Objekte in Kapitel 9.7.4.2 verwiesen.

9.7.6 Die Modulstatus-Objekte

In das Objektverzeichnis des Buskopplers wird für jedes *konfigurierte Modul* ein Modulstatus-Objekt eingeblendet, das den in Kapitel 7.4.6 beschriebenen Modulstatus enthält.

Das Modulstatus-Objekt hat den Datentyp RECORD. Es ist für alle Module gleich aufgebaut und enthält in den Subindizes 1 und 2 die beiden Informationen, ob das Modul *vorhanden* (present) und *betriebsbereit* (operational) ist. Der Subindex 0 mit der Anzahl der nachfolgenden Subindizes enthält folglich den Wert 2.

Auf Modulstatus-Objekte kann in PREOP, SAFEOP und OP lesend zugegriffen werden. Allerdings ist ein solcher Zugriff nicht sinnvoll, da der Modulstatus auch in den zyklisch an den Master übermittelten Prozesseingangsdaten des Buskopplers enthalten ist. Die Modulstatus-Objekte werden vielmehr im Rahmen der PDO-Konfiguration als Referenzobjekte für die Abbildung des Modulstatus auf TxPDOs (↗ 9.8.3) benötigt.

9.7.6.1 Aufbau des Modulstatus-Objekts

Modulstatus-Objekt	
Index-Bereich	0xA010 - 0xA400 (MDP-spezifisch)
Index-Berechnung	0xA000 + 0x10 · <Modulnummer>
Offline-Bezeichnung	"Module Status"
Online-Bezeichnung	"Module <Modulnummer> - Module Status"
Datentyp	RECORD
Bitgröße	24
Zugriff via Complete-Access	möglich
Subindex 0	
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes
Offline/Online-Bezeichnung	"SubIndex 000"
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	2
Subindex 1	
Kurzbeschreibung	zeigt an, ob das Modul *vorhanden* ist
Offline/Online-Bezeichnung	"Present"
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 16)
Zugriff	nur lesend
Online-Wert	0 (FALSE): Modul ist nicht *vorhanden* oder falsches Modul gesteckt 1 (TRUE): Modul ist *vorhanden*

Modulstatus-Objekt	
Subindex 2	
Kurzbeschreibung	zeigt an, ob das Modul *betriebsbereit* ist
Offline/Online-Bezeichnung	"Operational"
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 16)
Zugriff	nur lesend
Online-Wert	0 (FALSE): Modul ist *nicht betriebsbereit* 1 (TRUE): Modul ist *betriebsbereit*

9.7.7 Die Kanalstatus-Objekte

In das Objektverzeichnis des Buskopplers wird für jedes *konfigurierte Modul*, das diagnosefähig ist und über E/A-Daten verfügt, ein Kanalstatus-Objekt eingeblendet, das den in Kapitel 7.4.7 beschriebenen Kanalstatus enthält. Voraussetzung dafür ist, dass das jeweilige Modul am Rückwandbus steckt, da der Buskoppler nur dann die erforderlichen Informationen vom Modul abrufen kann.

Die Kanalstatus-Objekte haben den Datentyp RECORD. Für jeden vorhandenen Kanal des diagnosefähigen Moduls ist in diesem Objekt ein entsprechender Objekteintrag vom Datentyp BOOL enthalten. Die Objekteinträge zeigen den Zustand der jeweiligen Kanäle, der entweder "nicht OK" (0) oder "OK" (1) sein kann. Die in Subindex 0 enthaltene Anzahl der nachfolgenden Subindizes entspricht folglich der Anzahl der Kanäle des jeweiligen Moduls.

Auf Kanalstatus-Objekte kann in PREOP, SAFEOP und OP lesend zugegriffen werden. Allerdings ist ein solcher Zugriff nicht sinnvoll, da der Kanalstatus der diagnosefähigen *konfigurierten Module* auch in den zyklisch an den Master übermittelten Prozesseingangsdaten des Buskopplers enthalten ist. Die Kanalstatus-Objekte werden vielmehr im Rahmen der PDO-Konfiguration als Referenzobjekte für die Abbildung des Kanalstatus auf TxPDOs (↗ 9.8.3) benötigt.

9.7.7.1 Allgemeiner Aufbau des Objekts

Kanalstatus-Objekt	
Index-Bereich	0x7011 - 0x7401 (MDP-spezifisch)
Index-Berechnung	0x7001 + 0x10 · <Modulnummer>
Offline-Bezeichnung	"Channel Status"
Online-Bezeichnung	"Module <Modulnummer> - Channel Status"
Datentyp	RECORD
Bitgröße	• ist abhängig von der Anzahl der Kanäle des Moduls • wird auf ein Vielfaches von 8 aufgerundet
Zugriff via Complete-Access	möglich
Subindex 0	
Kurzbeschreibung	Anzahl *n* der nachfolgenden Subindizes
Offline/Online-Bezeichnung	"SubIndex 000"
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	entspricht der Anzahl der Kanäle des Moduls
...	

Kanalstatus-Objekt	
Subindex <*s*>	
Kurzbeschreibung	Status des Kanals <c> mit c = *s* - 1 (Nummerierung beginnt mit 0 in Subindex 1)
Offline-/Online-Bezeichnung	"Channel <*c*> OK"
Datentyp (Bitgröße/Bitoffset)	BOOL (1 / 16 + *c*)
Zugriff	nur lesend
Online-Wert	0 (FALSE): Kanal ist nicht 'OK' 1 (TRUE): Kanal ist 'OK' Ein Kanal wird dann als "OK" angezeigt, wenn folgende Bedingungen erfüllt sind: • der Buskoppler befindet sich in SAFEOP oder OP • das Modul ist laut Modulstatus *vorhanden* und *betriebsbereit* • für den Kanal liegt keine Diagnosemeldung im Modul vor
…	

9.7.8 Die Diagnosestatus-Objekte

In das Objektverzeichnis des Buskopplers wird für jedes *konfigurierte Modul*, das diagnosefähig ist, ein Diagnosestatus-Objekt eingeblendet, das den in Kapitel 7.4.8 beschriebenen Diagnosestatus enthält. Voraussetzung dafür ist, dass das jeweilige Modul am Rückwandbus steckt, da der Buskoppler nur dann die erforderlichen Informationen vom Modul abrufen kann.

Die Diagnosestatus-Objekte haben den Datentyp RECORD. Jede der im Diagnosestatus enthaltenen Diagnose-IDs wird in diesem Objekt auf einen entsprechenden Objekteintrag vom Datentyp USINT abgebildet. Diese Abbildung beginnt mit aufsteigender Position innerhalb des Diagnosestatus bei Subindex 1. Die in Subindex 0 enthaltene Anzahl der nachfolgenden Subindizes entspricht der Anzahl der im Diagnosestatus enthaltenen Diagnose-IDs.

Auf Diagnosestatus-Objekte kann in PREOP, SAFEOP und OP lesend zugegriffen werden. Im Gegensatz zum Modul- und Kanalstatus ist der Diagnosestatus nicht in den zyklisch an den Master übermittelten Prozesseingangsdaten des Buskopplers enthalten. Das bedeutet, der Diagnosestatus eines Moduls kann nur durch einen Zugriff auf das jeweilige Diagnosestatus-Objekt ermittelt werden. Dieser Zugriff ist immer dann sinnvoll, wenn durch den Modul- oder Kanalstatus des jeweiligen Moduls das Vorliegen einer Diagnosemeldung angezeigt wird und dann näher bestimmt werden soll, um welche Diagnosemeldung(en) es sich dabei handelt.

9.7.8.1 Allgemeiner Aufbau des Objekts

Diagnosestatus-Objekt	
Index-Bereich	0x7012 - 0x7402 (MDP-spezifisch)
Index-Berechnung	0x7002 + 0x10 · <Modulnummer>
Offline-Bezeichnung	"Diagnostic Status"
Online-Bezeichnung	"Module <Modulnummer> - Diagnostic Status"
Datentyp	RECORD
Bitgröße	16 + 8 · *n*
Zugriff via Complete-Access	möglich
Subindex 0	
Kurzbeschreibung	Anzahl *n* der nachfolgenden Subindizes
Offline/Online-Bezeichnung	"SubIndex 000"
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	entspricht der Anzahl der Kanäle des Moduls + 1

Diagnosestatus-Objekt	
Subindex 1	
Kurzbeschreibung	enthält die modulbezogene Diagnose-ID
Offline-/Online-Bezeichnung	"Module"
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 16)
Zugriff	nur lesend
Online-Wert	0: es liegt keine modulbezogene Diagnosemeldung vor 1-255: es liegt eine bestimmte modulbezogene Diagnosemeldung vor (↗ 7.4.8)
…	
Subindex <*s*>	
Kurzbeschreibung	Diagnose-ID für Kanal <*c*> mit *c* = *s* - 2 (Nummerierung beginnt mit 0 in Subindex 2)
Offline-/Online-Bezeichnung	"Channel <c>"
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 24 + 8 · c)
Zugriff	nur lesend
Online-Wert	0: es liegt keine kanalbezogene Moduldiagnose vor 1-255: es liegt eine bestimmte kanalbezogene Diagnosemeldung vor (↗ 7.4.8)
…	

9.8 Die Objekte zur PDO-Konfiguration des Buskopplers

Der zyklische Austausch von Prozessdaten zwischen dem Master und dem Buskoppler ist in Form von Prozessdatenobjekten (Process Data Objects, PDOs) organisiert, wobei je nach Übertragungsrichtung zwischen RxPDOs und TxPDOs unterschieden wird:

RxPDOs werden für die Übertragung von Prozessausgangsdaten vom Master an den Buskoppler verwendet. Aus Sicht des Buskopplers werden RxPDOs also empfangen und haben daher den Präfix Rx = Reception = Empfang. Als Prozessausgangsdaten des Buskopplers wird der Inhalt der Ausgangsdaten-Objekte aller *konfigurierten Module* in RxPDOs abgebildet.

TxPDOs werden für die Übertragung von Prozesseingangsdaten vom Buskoppler an den Master verwendet. Aus Sicht des Buskopplers werden TxPDOs also gesendet und haben daher den Präfix Tx = Transmission = Aussendung. Als Prozesseingangsdaten des Buskopplers wird der Inhalt der Eingangsdaten-Objekte aller *konfigurierten Module* in TxPDOs abgebildet. Zusätzlich wird auch der Inhalt der Modulstatus- und Kanalstatus-Objekte dieser Module in TxPDOs abgebildet.

Für die Abbildung des Inhalts bestimmter Modul-Objekte in Rx- und TxPDOs sind die nachfolgend beschriebenen PDO-Mapping-Objekte des Buskopplers zuständig. Im Anschluss daran werden die beiden PDO-Assignment-Objekte 0x1C12 und 0x1C13 des Buskopplers beschrieben (↗ 9.8.4). Sie sorgen dafür, dass die Rx- und TxPDOs den Sync-Managern 2 und 3 zugeordnet werden, die für den zyklischen Austausch der Prozessdaten mit dem Master zuständig sind.

9.8.1 Die PDO-Mapping-Objekte

Durch die PDO-Mapping-Objekte wird festgelegt, welche Daten aus den im Objektverzeichnis vorhandenen Objekten in PDOs abgebildet werden. Dazu enthalten die als RECORD definierten PDO-Mapping-Objekte in den Subindizes 1 bis n Referenzdaten, die auf die entsprechenden Objekte bzw. Objekteinträge verweisen. Der Aufbau eines PDOs ergibt sich aus der lückenlosen Aneinanderreihung der mit aufsteigendem Subindex enthaltenen, referenzierten Objektdaten.

Die in den Subindizes vom Datentyp UDINT enthaltenen Referenzdaten umfassen 32 Bit und sind wie folgt aufgebaut:

Bits	Bedeutung
31 – 24	Index des referenzierten Objektes
23 – 16	
15 – 8	Subindex bei Referenz auf ein RECORD- oder ARRAY-Objekt, ansonsten 0
7 – 0	Bitgröße der referenzierten Daten

Bezüglich der Referenzdaten wird in diesem Handbuch folgende Notation verwendet:

<hexadezimaler Index>:<dezimaler Subindex>:<dezimale Bitgröße>

Beispiel: 0x6040:03:16

Referenziert wird ein 16 Bit umfassendes Datenelement in Subindex 3 des Objektes 0x6040. Es handelt sich dabei um das Eingangsdaten-Objekt von Modul 4.

Über einen speziellen Referenzdaten-Eintrag lassen sich auch sogenannte Padding-Bits (Füllbits) in ein PDO einfügen. Diese werden dazu verwendet, ein PDO bei Bedarf bis zu einer Byte-Grenze aufzufüllen. Ein Padding-Eintrag verweist immer auf den Subindex 0 von Objekt 0x0000 und enthält als Bitgröße die Anzahl der einzufügenden Padding-Bits. Wertmäßig werden Padding-Bits im PDO mit Nullen belegt.

Beispiel: 0x0000:00:4

Dieser Referenzeintrag umfasst 4 Padding-Bits.

9.8.2 Die RxPDO-Mapping-Objekte

Für jedes im Objektverzeichnis vorhandene Ausgangsdaten-Objekt existiert ein RxPDO-Mapping-Objekt. Das RxPDO-Mapping-Objekt sorgt dafür, dass die im Ausgangsdaten-Objekt enthaltenen Daten in ein Ausgangsdaten-RxPDO abgebildet werden.

Der Index eines RxPDO-Mapping-Objekts hängt von der Nummer des Moduls ab, auf dessen Ausgangsdaten-Objekt verwiesen wird, und berechnet sich wie folgt:

Index des RxPDO-Mapping-Objekts = 0x1600 + <Modulnummer>

Auch der Index des Ausgangsdaten-Objekts, auf das verwiesen wird, hängt von dieser Modulnummer ab:

Index des Ausgangsdaten-Objekts = 0x7000 + 0x10 · <Modulnummer>

Daraus ergibt sich folgender Zusammenhang:

Für ein Modul mit der Modulnummer 0x*MN* werden die im Ausgangsdaten-Objekt 0x7*MN*0 enthaltenen Daten durch das RxPDO-Mapping-Objekt 0x16*MN* in ein Ausgangsdaten-RxPDO abgebildet.

In den Ausgangsdaten-RxPDOs werden immer alle im Ausgangsdaten-Objekt enthaltenen Datenelemente in gleicher Anordnung abgebildet. Das bedeutet, dass im RxPDO-Mapping-Objekt für jeden im Ausgangsdaten-Objekt vorhandenen Subindex ein entsprechender Referenzdaten-Eintrag existiert, der auf diesen Subindex verweist.

Bei Bedarf ist ein zusätzlicher Padding-Eintrag vorhanden, mit dem durch Anfügen einer bestimmte Anzahl an Padding-Bits dafür gesorgt wird, dass die Größe des RxPDOs dem Vielfachen eines Bytes entspricht. Dadurch wird sichergestellt, dass die im Prozessausgangsabbild des Buskopplers lückenlos aneinandergereihten RxPDOs immer auf Byte-Grenzen ausgerichtet sind.

Die RxPDO-Mapping-Objekte und deren Inhalt werden automatisch vom Buskoppler anhand der für die *konfigurierten Module* verfügbaren Ausgangsdaten-Objekte generiert. Sie sind nur im Online-

Objektverzeichnis des Buskopplers sichtbar und erlauben einen lesenden, aber keinen schreibenden Zugriff. Das heißt, die RxPDO-Konfiguration des Buskopplers kann nicht manuell verändert werden.

9.8.2.1 Allgemeiner Aufbau eines RxPDO-Mapping-Objekts

RxPDO-Mapping-Objekt für Ausgangsdaten	
Index-Bereich	0x1601 - 0x1640
Index-Berechnung	0x1600 + <Modulnummer> = 0x16*MN* (*MN* steht für die zweistellige, hexadezimale Modulnummer)
Online-Bezeichnung	"RxPDO (Module <Modulnummer> - Outputs)"
Kurzbeschreibung	Abbildung der Daten des Ausgangsdaten-Objekts 0x7*MN*0 (= Referenzobjekt) in ein Ausgangsdaten-RxPDO
Datentyp	RECORD
Bitgröße	16 + 32 · *n*
Zugriff via Complete-Access	möglich
Subindex 0	
Kurzbeschreibung	Anzahl *n* der nachfolgenden Subindizes
Online-Bezeichnung	"SubIndex 000"
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	• entspricht dem Wert von Subindex 0 des Referenzobjekts und damit der Anzahl der einzelnen, in diesem Objekt enthaltenen Datenelemente • erhöht sich um 1 für den Fall, dass ein Padding-Eintrag existiert
…	
Subindex <*s*>	
Kurzbeschreibung	Referenzdaten-Eintrag <*e*> mit *e* = *s* - 1, der auf den Subindex <*s*> im Referenzobjekt verweist (Nummerierung beginnt mit 0 in Subindex 1)
Online-Bezeichnung	"Mapping entry <*e*>"
Datentyp (Bitgröße/Bitoffset)	UDINT (32 / 16 + 32 · *e*)
Zugriff	nur lesend
Online-Wert	0x7*MN*0:<*s*>:<Bitgröße des Subindex im Referenzobjekt>
…	

9.8.2.2 Beispiel für den Inhalt des RxPDO-Objekts

In diesem Beispiel wird das analoge Eingangsmodul 'AI 4 x TC, Iso., 16 Bit' (600-254-4AD02) als Modul 3 verwendet. Das Modul verfügt über Ausgangsdaten, in denen die Werte für die externe Temperaturkompensation der 4 Kanäle enthalten sind.

Die Abbildung der in Objekt 0x7030 enthaltenen Ausgangsdaten in ein RxPDO erfolgt im RxPDO-Mapping-Objekt 0x1603, wie nachfolgend dargestellt:

Ausgangdaten-Objekt 0x7030				**RxPDO-Mapping-Objekt 0x1603**		
SI	**Datentyp (Bitgröße)**	**Beschreibung**		**SI**	**Online-Wert**	**Beschreibung**
0	USINT (8)	Subindex 0 (= 4)	→	**0**	4	Subindex 0
1	INT (16)	Ext. Temp.-Kompensation Kanal 0	→	**1**	0x7030:01:16	Verweis auf 0x7030:01 mit 16 Bits
2	INT (16)	Ext. Temp.-Kompensation Kanal 1	→	**2**	0x7030:02:16	Verweis auf 0x7030:02 mit 16 Bits
3	INT (16)	Ext. Temp.-Kompensation Kanal 2	→	**3**	0x7030:03:16	Verweis auf 0x7030:03 mit 16 Bits
4	INT (16)	Ext. Temp.-Kompensation Kanal 3	→	**4**	0x7030:04:16	Verweis auf 0x7030:04 mit 16 Bits

In Kapitel 9.8.3.2 ist für das beispielhaft gewählte Modul eine ähnliche Darstellung zum Inhalt der TxPDO-Mapping-Objekte bezüglich der Eingangsdaten sowie des Modul- und Kanalstatus enthalten.

9.8.3 Die TxPDO-Mapping-Objekte

Für jedes der folgenden, im Objektverzeichnis vorhandenen Modul-Objekte, existiert ein TxPDO-Mapping-Objekt, das die enthaltenen Daten auf ein entsprechendes TxPDO abbildet:

Eingangsdaten-Objekt → Eingangsdaten-TxPDO

Modulstatus-Objekt → Modulstatus-TxPDO

Kanalstatus-Objekt → Kanalstatus-TxPDO

Sowohl der Index des TxPDO-Mapping-Objektes als auch der Index des referenzierten Modul-Objekts hängen von der jeweiligen Nummer (1 - 64) des betreffenden Moduls ab:

TxPDO	Index des TxPDO-Mapping-Objekt	Index des Referenzobjekts
Eingangsdaten-TxPDO	0x1A00 + 0x04 · <Modulnummer>	0x6000 + 0x10 · <Modulnummer>
Modulstatus-TxPDO	0x1A01 + 0x04 · <Modulnummer>	0xA000 + 0x10 · <Modulnummer>
Kanalstatus-TxPDO	0x1A02 + 0x04 · <Modulnummer>	0xA001 + 0x10 · <Modulnummer>

In die verschiedenen TxPDOs werden immer alle im jeweiligen Referenzobjekt enthaltenen Datenelemente in gleicher Anordnung abgebildet. Das bedeutet, dass in den TxPDO-Mapping-Objekten für jeden im jeweiligen Referenzobjekt vorhandenen Subindex ein entsprechender Referenzdaten-Eintrag existiert, der auf diesen Subindex verweist.

Bei Bedarf ist ein zusätzlicher Padding-Eintrag vorhanden, mit dem durch Anfügen einer bestimmten Anzahl an Padding-Bits dafür gesorgt wird, dass die Größe des TxPDOs dem Vielfachen eines Bytes entspricht. Dadurch wird sichergestellt, dass die im Prozessausgangsabbild des Buskopplers lückenlos aneinandergereihten TxPDOs immer auf Byte-Grenzen ausgerichtet sind.

Die TxPDO-Mapping-Objekte und deren Inhalt werden automatisch vom Buskoppler anhand der für die *konfigurierten Module* verfügbaren Eingangsdaten-, Modulstatus- und Kanalstatus-Objekte generiert. Sie sind nur im Online-Objektverzeichnis des Buskopplers sichtbar und erlauben einen lesenden, aber keinen schreibenden Zugriff. Das heißt, die TxPDO-Konfiguration des Buskopplers kann nicht manuell verändert werden.

9.8.3.1 Allgemeiner Aufbau des TxPDO-Objekts

TxPDO-Mapping-Objekt für Eingangsdaten, Modulstatus oder Kanalstatus	
Index-Bereich	bezüglich Eingangsdaten: 0x1A04 - 0x1B00 bezüglich Modulstatus: 0x1A05 - 0x1B01 bezüglich Kanalstatus: 0x1A06 - 0x1B02
Index-Berechnung	bezüglich Eingangsdaten: 0x1A00 + 0x04 · <Modulnummer> bezüglich Modulstatus: 0x1A01 + 0x04 · <Modulnummer> bezüglich Kanalstatus: 0x1A02 + 0x04 · <Modulnummer>
Online-Bezeichnung	bezüglich Eingangsdaten: "TxPDO (Module <Modulnummer> - Inputs)" bezüglich Modulstatus: "TxPDO (Module <Modulnummer> - Module Status)" bezüglich Kanalstatus: "TxPDO (Module <Modulnummer> - Channel Status)"
Kurzbeschreibung	Abbildung der Daten des jeweiligen Referenzobjekts auf ein TxPDO
Datentyp	RECORD
Bitgröße	16 + 32 · *n*
Zugriff via Complete-Access	möglich

TxPDO-Mapping-Objekt für Eingangsdaten, Modulstatus oder Kanalstatus	
Subindex 0	
Kurzbeschreibung	Anzahl *n* der nachfolgenden Subindizes
Online-Bezeichnung	"SubIndex 000"
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	• entspricht dem Wert von Subindex 0 des Referenzobjekts und damit der Anzahl der einzelnen, in diesem Objekt enthaltenen Datenelemente • erhöht sich um 1 für den Fall, dass ein Padding-Eintrag existiert
…	
Subindex <*s*>	
Kurzbeschreibung	Referenzdaten-Eintrag <*e*> mit *e* = *s* - 1, der auf den Subindex <*s*> im Referenzobjekt verweist (Nummerierung beginnt mit 0 in Subindex 1)
Online-Bezeichnung	"Mapping entry <*e*>"
Datentyp (Bitgröße/Bitoffset)	UDINT (32 / 16 + 32 · *e*)
Zugriff	nur lesend
Online-Wert	bzgl. Eingangsdaten: 0x6*MN*0:<s>:<Bitgröße des Subindex im Referenzobjekt> bzgl. Modulstatus: 0xA*MN*0:<s>:<Bitgröße des Subindex im Referenzobjekt> bzgl. Kanalstatus: 0xA*MN*1:<s>:<Bitgröße des Subindex im Referenzobjekt> (*MN* steht für die zweistellige, hexadezimale Modulnummer)
…	

9.8.3.2 Beispiel für den Inhalt der TxPDO-Mapping-Objekte

In diesem Beispiel wird das analoge Eingangsmodul 'AI 4 x TC, Iso., 16 Bit' (600-254-4AD02) als Modul 3 verwendet. Das Modul verfügt über Eingangsdaten, in denen die Analogwerte für die 4 Kanäle enthalten sind.

Vom Buskoppler werden für dieses Modul das Eingangsdaten-Objekt 0x6030, das Modulstatus-Objekt 0xA030 und aufgrund der Diagnosefähigkeit das Kanalstatus-Objekt 0xA031 zur Verfügung gestellt. Nachfolgend ist dargestellt, wie der Inhalt dieser Modulobjekte über entsprechende TxPDO-Mapping-Objekte auf TxPDOs abgebildet werden.

Eingangsdaten-Objekt 0x6030				TxPDO-Mapping-Objekt 0x1A0C		
SI	**Datentyp (Bitgröße)**	**Beschreibung**		**SI**	**Online-Wert**	**Beschreibung**
0	USINT (8)	Subindex 0 (= 4)	→	**0**	4	Subindex 0
1	INT (16)	Analogwert Kanal 0	→	**1**	0x6030:01:16	Verweis auf 0x6030:01 mit 16 Bits
2	INT (16)	Analogwert Kanal 1	→	**2**	0x6030:02:16	Verweis auf 0x6030:02 mit 16 Bits
3	INT (16)	Analogwert Kanal 2	→	**3**	0x6030:03:16	Verweis auf 0x6030:03 mit 16 Bits
4	INT (16)	Analogwert Kanal 3	→	**4**	0x6030:04:16	Verweis auf 0x6030:04 mit 16 Bits

Modulstatus-Objekt 0xA030				TxPDO-Mapping-Objekt 0x1A0D		
SI	**Datentyp (Bitgröße)**	**Beschreibung**		**SI**	**Online-Wert**	**Beschreibung**
0	USINT (8)	Subindex 0 (= 4)	→	**0**	2 + 1	Subindex 0
1	BOOL (1)	Present	→	**1**	0xA030:01:01	Verweis auf 0xA030:01 mit 1 Bit
2	BOOL (1)	Operational	→	**2**	0xA030:02:01	Verweis auf 0xA030:02 mit 1 Bit
				3	0x0000:00:06	Padding-Eintrag mit 6 Bits

Kanalstatus-Objekt 0xA031				TxPDO-Mapping-Objekt 0x1A0E		
SI	Datentyp (Bitgröße)	Beschreibung		SI	Online-Wert	Beschreibung
0	USINT (8)	Subindex 0 (= 4)	→	0	4 + 1	Subindex 0
1	BOOL (1)	Channel 0 OK	→	1	0xA031:01:01	Verweis auf 0xA031:01 mit 1 Bit
2	BOOL (1)	Channel 1 OK	→	2	0xA031:02:01	Verweis auf 0xA031:02 mit 1 Bit
3	BOOL (1)	Channel 2 OK	→	3	0xA031:03:01	Verweis auf 0xA031:03 mit 1 Bit
4	BOOL (1)	Channel 3 OK	→	4	0xA031:04:01	Verweis auf 0xA031:04 mit 1 Bit
				5	0x0000:00:04	Padding-Eintrag mit 4 Bits

In den TxPDO-Mapping-Objekten bezüglich des Modul- und des Kanalstatus ist jeweils ein zusätzlicher Padding-Eintrag vorhanden. Dieser sorgt dafür, dass sich die Größe der TxPDOs auf 8 Bits, also 1 Byte erhöht.

In Kapitel 9.8.2.2 ist für das beispielhaft gewählte Modul eine ähnliche Darstellung zum Inhalt des RxPDO-Mapping-Objektes bezüglich der Ausgangsdaten enthalten.

9.8.4 Die PDO-Assignment-Objekte

Die PDO-Assignment-Objekte sind für die Zuweisung der verfügbaren PDOs zu den Sync-Managern 2 und 3 und damit für die Zusammenstellung der Prozessdaten des Buskopplers zuständig.

Das RECORD-Objekt 0x1C12 (Sync Manager 2 RxPDO Assignment) enthält in seinen Subindizes, welche RxPDOs dem für die Prozessausgangsdaten-Kommunikation zuständigen Sync-Manager 2 zugewiesen werden und somit in den Prozessausgangsdaten des Buskopplers enthalten sind.

Das RECORD-Objekt 0x1C13 (Sync Manager 3 TxPDO Assignment) enthält in seinen Subindizes, welche TxPDOs dem für die Prozesseingangsdaten-Kommunikation zuständigen Sync-Manager 3 zugewiesen werden und somit in den Prozesseingangsdaten des Buskopplers enthalten sind.

Als Referenz auf ein bestimmtes PDO wird in den Subindizes der PDO-Assignment-Objekte jeweils der Index des zugehörigen PDO-Mapping-Objekts verwendet.

Der Inhalt der beiden PDO-Assignment-Objekte wird automatisch generiert und zwar so, dass sämtliche verfügbaren Rx- und TxPDOs in den Prozessausgangs- bzw. Prozesseingangsdaten des Buskopplers enthalten sind.

Die Reihenfolge der PDOs innerhalb der Prozessdaten ist durch die Reihenfolge der Module vorgegeben. Bezüglich der verschiedenen TxPDOs, die für ein einzelnes Modul in den Prozesseingangsdaten des Buskopplers enthalten sein können, gilt folgende Reihenfolge:

- zuerst Eingangsdaten-TxPDO des Moduls, sofern vorhanden
- dann Modulstatus-TxPDO des Moduls
- zuletzt Kanalstatus-TxPDO des Moduls, sofern vorhanden

Die automatisch durchgeführte PDO-Zuordnung kann vom Anwender nicht nachträglich verändert werden. Auf die PDO-Assignment-Objekte kann daher nur lesend zugegriffen werden.

9.8.4.1 Allgemeiner Aufbau des Objekts 0x1C12

Index 0x1C12	Sync Manager 2 RxPDO Assignment
Kurzbeschreibung	Zuordnung der RxPDOs zum Sync-Manager 2
Datentyp (Bitgröße)	ARRAY OF UINT (1040)
Weitere Details	Dem Sync-Manager 2 werden alle verfügbaren RxPDOs mit den Ausgangsdaten der *konfigurierten Module* zugeordnet. Diese Zuordnung findet automatisch und mit aufsteigender Modulnummer statt.
Zugriff via Complete-Access	möglich

Index 0x1C12	Sync Manager 2 RxPDO Assignment
Subindex 0	SubIndex 000
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes = Anzahl zugeordneter RxPDOs
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	z.B. 8
Subindex 1	SubIndex 001
Kurzbeschreibung	Index des 1. zugeordneten RxPDOs
Datentyp (Bitgröße/Bitoffset)	UINT (16 / 16)
Zugriff	nur lesend
Online-Wert	z.B. 0x1601 = RxPDO mit Ausgangsdaten von Modul 1
Subindex 2	SubIndex 002
Kurzbeschreibung	Index des 2. zugeordneten RxPDOs
Datentyp (Bitgröße/Bitoffset)	UINT (16 / 32)
Zugriff	nur lesend
Online-Wert	z.B. 0x1604 = RxPDO mit Ausgangsdaten von Modul 4
…	…
Subindex 8	SubIndex 008
Kurzbeschreibung	Index des 8. zugeordneten RxPDOs
Datentyp (Bitgröße/Bitoffset)	UINT (16 / 128)
Zugriff	nur lesend
Online-Wert	z.B. 0x160C = RxPDO mit Ausgangsdaten von Modul 12

9.8.4.2 Allgemeiner Aufbau des Objekts 0x1C13

Index 0x1C13	Sync Manager 3 TxPDO Assignment
Kurzbeschreibung	Zuordnung der TxPDOs zum Sync-Manager 3
Datentyp (Bitgröße)	ARRAY OF UINT (3088)
Zugriff via Complete-Access	möglich
Weitere Details	Dem Sync-Manager 3 werden alle verfügbaren TxPDOs der *konfigurierten Module* automatisch zugeordnet. Diese Zuordnung findet mit aufsteigender Modulnummer und bezogen auf einzelnes Modul wie folgt statt: • TxPDO mit den Eingangsdaten (sofern vorhanden) • TxPDO mit dem Modulstatus (immer vorhanden) • TxPDO mit dem Kanalstatus (sofern vorhanden)
Subindex 0	**SubIndex 000**
Kurzbeschreibung	Anzahl der nachfolgenden Subindizes = Anzahl zugeordneter TxPDOs
Datentyp (Bitgröße/Bitoffset)	USINT (8 / 0)
Zugriff	nur lesend
Online-Wert	z.B. 22
Subindex 1	**SubIndex 001**
Kurzbeschreibung	Index des 1. zugeordneten TxPDOs
Datentyp (Bitgröße/Bitoffset)	UINT (16 / 16)
Zugriff	nur lesend
Online-Wert	z.B. 0x1A05 = TxPDO mit Modulstatus von Modul 1

Index 0x1C13	Sync Manager 3 TxPDO Assignment
Subindex 2	**SubIndex 002**
Kurzbeschreibung	Index des 2. zugeordneten TxPDOs
Datentyp (Bitgröße/Bitoffset)	UINT (16 / 32)
Zugriff	nur lesend
Online-Wert	z.B. 0x1A08 = TxPDO mit Eingangsdaten von Modul 2
Subindex 3	**SubIndex 003**
Kurzbeschreibung	Index des 3. zugeordneten TxPDOs
Datentyp (Bitgröße/Bitoffset)	UINT (16 / 48)
Zugriff	nur lesend
Online-Wert	z.B. 0x1A09 = TxPDO mit Modulstatus von Modul 2
Subindex 4	**SubIndex 004**
Kurzbeschreibung	Index des 4. zugeordneten TxPDOs
Datentyp (Bitgröße/Bitoffset)	UINT (16 / 64)
Zugriff	nur lesend
Online-Wert	z.B. 0x1A0C = TxPDO mit Eingangsdaten von Modul 3
Subindex 5	**SubIndex 005**
Kurzbeschreibung	Index des 5. zugeordneten TxPDOs
Datentyp (Bitgröße/Bitoffset)	UINT (16 / 80)
Zugriff	nur lesend
Online-Wert	z.B. 0x1A0D = TxPDO mit Modulstatus von Modul 3
Subindex 6	**SubIndex 006**
Kurzbeschreibung	Index des 6. zugeordneten TxPDOs
Datentyp (Bitgröße/Bitoffset)	UINT (16 / 96)
Zugriff	nur lesend
Online-Wert	z.B. 0x1A0E = TxPDO mit Kanalstatus von Modul 3
…	…

9.8.4.3 Beispiel für eine PDO-Zuweisung und resultierende Prozessdaten

#	Modul	Ausgangsdaten	Eingangsdaten	Modulstatus	Kanalstatus
1	**16 DI** 600-210-0AP21	-	0x6010 → 0x1A04 (2 Byte)	0xA010 → 0x1A05 (1 Byte)	-
2	**8 DO** 600-220-0AH01	0x7020 → 0x1602 (1 Byte)	-	0xA020 → 0x1A09 (1 Byte)	-
3	**1 CNT 24V** 600-300-7AA01	0x7030 → 0x1603 (8 Byte)	0x6040 → 0x1A0C (8 Byte)	0xA030 → 0x1A0D (1 Byte)	0xA031 → 0x1A0E (1 Byte)
4	**PWR 24V** 600-710-0AA01	-	-	0xA040 → 0x1A11 (1 Byte)	-
5	**2 AO U** 600-261-4AB01	0x7050 → 0x1605 (4 Byte)	-	0xA050 → 0x1A15 (1 Byte)	0xA051 → 0x1A16 (1 Byte)
6	**8 AI U** 600-252-4AD01	-	0x6060 → 0x1A18 (16 Byte)	0xA060 → 0x1A19 (1 Byte)	0xA061 → 0x1A1A (1 Byte)

Objekt 0x1C12 (Sync Manager 2 RxPDO Assignment)				Ausgangsprozessdaten des Buskopplers		
SI	**Wert**	**Beschreibung**		**Byte**	**Beschreibung**	**Größe**
0	3	Anzahl nachfolgender Subindizes				
1	0x1602	Ausgangsdaten-RxPDO von Modul 2	→	**0**	Ausgangsdaten von Modul 2	1 Byte
2	0x1603	Ausgangsdaten-RxPDO von Modul 3	→	**1 - 8**	Ausgangsdaten von Modul 3	8 Byte
3	0x1605	Ausgangsdaten-RxPDO von Modul 5	→	**9 - 12**	Ausgangsdaten von Modul 5	4 Byte

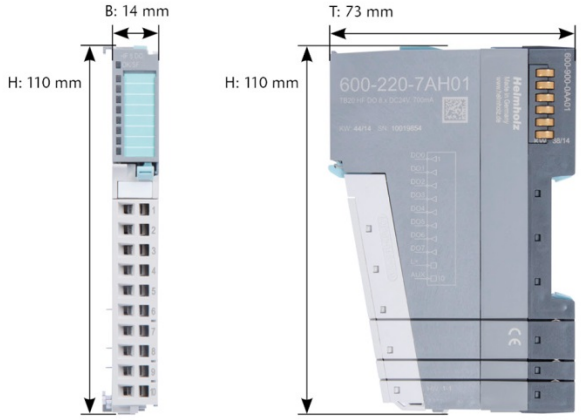
Objekt 0x1C13 (Sync Manager 3 TxPDO Assignment)				Eingangssprozessdaten des Buskopplers		
SI	**Wert**	**Beschreibung**		**Byte**	**Beschreibung**	**Größe**
0	10	Anzahl nachfolgender Subindizes				
1	0x1A04	Eingangsdaten-TxPDO von Modul 1	→	**0 - 1**	Eingangsdaten von Modul 1	2 Byte
2	0x1A05	Modulstatus-TxPDO von Modul 1	→	**2**	Modulstatus von Modul 1	1 Byte
3	0x1A09	Modulstatus-TxPDO von Modul 2	→	**3**	Modulstatus von Modul 2	1 Byte
4	0x1A0C	Eingangsdaten-TxPDO von Modul 3	→	**4 - 11**	Eingangsdaten von Modul 3	8 Byte
5	0x1A0D	Modulstatus-TxPDO von Modul 3	→	**12**	Modulstatus von Modul 3	1 Byte
6	0x1A0E	Kanalstatus-TxPDO von Modul 3	→	**13**	Kanalstatus von Modul 3	1 Byte
7	0x1A11	Modulstatus-TxPDO von Modul 4	→	**14**	Modulstatus von Modul 4	1 Byte
8	0x1A15	Modulstatus-TxPDO von Modul 5	→	**15**	Modulstatus von Modul 5	1 Byte
9	0x1A16	Kanalstatus-TxPDO von Modul 5	→	**16**	Kanalstatus von Modul 5	1 Byte
10	0x1A18	Eingangsdaten-TxPDO von Modul 6	→	**17 - 32**	Eingangsdaten von Modul 6	16 Byte
11	0x1A19	Modulstatus-TxPDO von Modul 6	→	**33**	Modulstatus von Modul 6	1 Byte
12	0x1A1A	Kanalstatus-TxPDO von Modul 6	→	**34**	Kanalstatus von Modul 6	1 Byte

10 Technische Daten

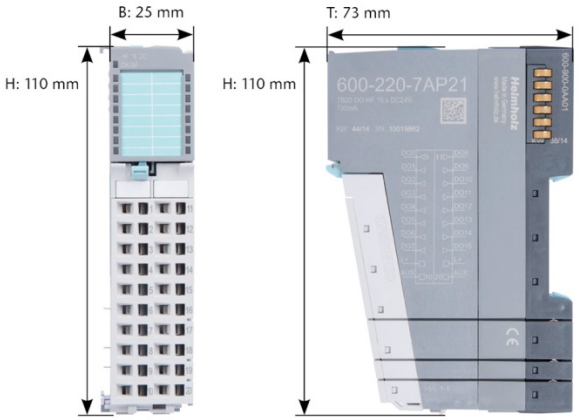
Bestellnummer	**600-185-1AA11**
Artikelbezeichnung	TB20 EtherCAT® Buskoppler
EtherCAT Schnittstelle	
Anschluss	2x RJ45
Protokoll	EtherCAT nach IEC-Standard 61158
Physik	Ethernet, 100BASE-TX
Übertragungsrate	100 Mbit/s Vollduplex
E/A-Abbild-Größe	jeweils max. 1024 Bytes
Features	automatische Prozessdatenkonfiguration CANopen over EtherCAT (CoE) Objektverzeichnis nach Modular Device Profile (MDP) SDO-Info, Complete-Access
Diagnosefunktionen	zyklusaktueller Modul- und Kanalstatus Diagnosestatus im Objektverzeichnis
USB-Port	
Protokoll	USB 1.1 Device, Full Speed
Anschluss	Mini-USB
Isolationsspannung	1,5 kV
Potentialtrennung	ja
Anreihbare Module	max. 64, alle Varianten
Spannungsversorgung	DC 24 V; 18 … 28 V DC
Stromaufnahme ohne Module	75 mA
Verlustleistung	max. 8 W
Stromversorgung für Module	DC 5 V; max. 2,5 A
Elektrisch vom Rückwandbus getrennt	nein
Abmessungen (H x B x T)	110 mm x 35 mm x 73 mm
Gewicht	115 g
Einbaulage	beliebig
Zertifizierung	CE, UL
Störfestigkeit	DIN EN 61000-6-2 „EMV-Störfestigkeit“
Störaussendung	DIN EN 61000-6-4 „EMV-Störaussendung“
Vibration und Schockfestigkeit	DIN EN 60068-2-8:2008 „Schwingung“ DIN EN 60068-27:2010 „Schock“
Schutzgrad	IP20
Relative Luftfeuchte	95% ohne Kondensation
Zulässige Umgebungstemperatur	0 … 60 °C; für UL-Anwendungen: 0 … 50 °C
Transport- und Lagertemperatur	-20 … 80 °C
Verschmutzungsgrad	2

11 Abmessungen

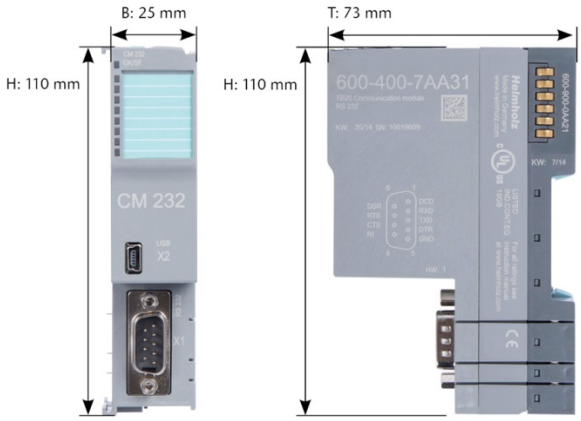
Einfachbreites Modul



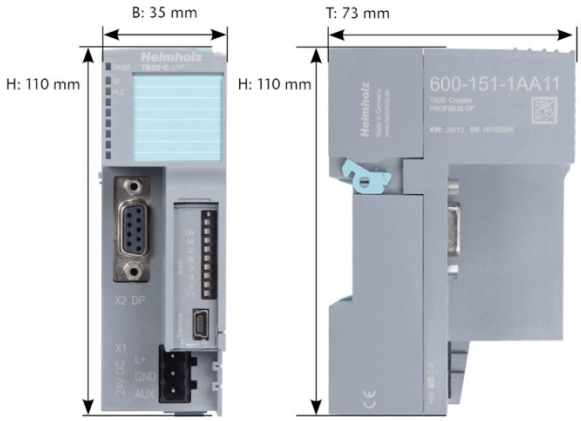
Doppelbreites Modul

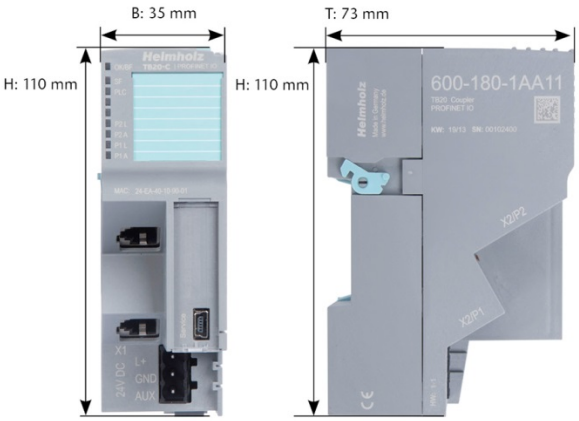


Kommunikations-Modul



Buskoppler





12 Ersatzteile

12.1 Basismodule

12.1.1 Standard Basismodul 14er Breite

Das Standard Basismodul 14er Breite ist im 5er Pack unter der Bestellnummer 600-900-9AA01 erhältlich.

12.1.2 Basismodul 25er Breite

Das Standard Basismodul 25er Breite ist im 5er Pack unter der Bestellnummer 600-900-9AA21 erhältlich.

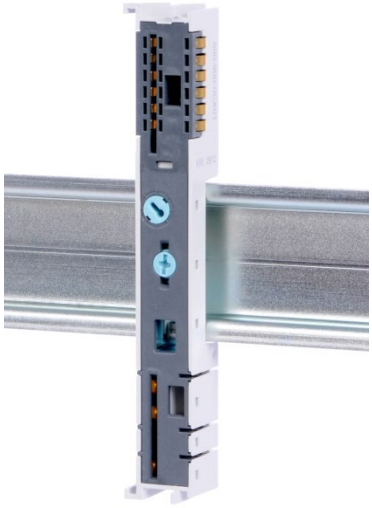
12.1.3 Einspeise-/Trenn Basismodul

Das Einspeise-/Trenn Basismodul ist im 5er Pack unter der Bestellnummer 600-900-9BA01 erhältlich.

12.1.4 Power Basismodul

Das Power Basismodul ist im 5er Pack unter der Bestellnummer 600-900-9CA01 erhältlich.

Es ist sowohl für das Powermodul (600-700-0AA01) geeignet, als auch für alle Buskoppler.



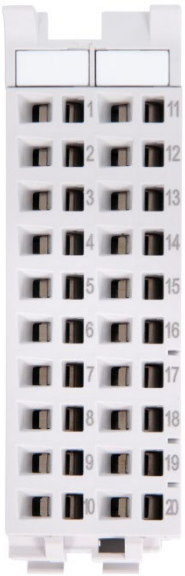
12.2 Frontstecker

12.2.1 Frontstecker 10-polig

Der Frontstecker 10-polig ist im 5er Pack unter der Bestellnummer 600-910-9AJ01 erhältlich.

12.2.2 Frontstecker 20-polig

Der Frontstecker 20-polig ist im 5er Pack unter der Bestellnummer 600-910-9AT21 erhältlich.



12.3 Elektronikmodule

Elektronikmodule können unter der Bestellnummer des Produkts als Ersatzteil bezogen werden. Es wird immer das gesamte Produkt inkl. Basismodul und Frontstecker geliefert.

12.4 Abschlusselement

Das Abschlusselement ist im 5er Pack unter der Bestellnummer 600-920-9AA01 erhältlich.