BECKHOFF New Automation Technology

Dokumentation | DE

EtherCAT

System-Dokumentation

29.03.2022 | Version: 5.6

Inhaltsverzeichnis

1 Vorwort

1.1 Hinweise zur Dokumentation

Zielgruppe

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist.
Zur Installation und Inbetriebnahme der Komponenten ist die Beachtung der Dokumentation und der nachfolgenden Hinweise und Erklärungen unbedingt notwendig.
Das Fachpersonal ist verpflichtet, für jede Installation und Inbetriebnahme die zu dem betreffenden Zeitpunkt veröffentlichte Dokumentation zu verwenden.

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

Disclaimer

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte werden jedoch ständig weiter entwickelt.
Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.
Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

Marken

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® und XPlanar® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH. Die Verwendung anderer in dieser Dokumentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Bezeichnungen führen.

Patente

Die EtherCAT-Technologie ist patentrechtlich geschützt, insbesondere durch folgende Anmeldungen und Patente: EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702 mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.

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

Copyright

© Beckhoff Automation GmbH & Co. KG, Deutschland.
Weitergabe sowie Vervielfältigung dieses Dokuments, Verwertung und Mitteilung seines Inhalts sind verboten, soweit nicht ausdrücklich gestattet.
Zuwiderhandlungen verpflichten zu Schadenersatz. Alle Rechte für den Fall der Patent-, Gebrauchsmuster- oder Geschmacksmustereintragung vorbehalten.

1.2 Sicherheitshinweise

Sicherheitsbestimmungen

Beachten Sie die folgenden Sicherheitshinweise und Erklärungen!
Produktspezifische Sicherheitshinweise finden Sie auf den folgenden Seiten oder in den Bereichen Montage, Verdrahtung, Inbetriebnahme usw.

Haftungsausschluss

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

Qualifikation des Personals

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

Erklärung der Hinweise

In der vorliegenden Dokumentation werden die folgenden Hinweise verwendet.
Diese Hinweise sind aufmerksam zu lesen und unbedingt zu befolgen!

⚠ GEFAHR

Akute Verletzungsgefahr!

Wenn dieser Sicherheitshinweis nicht beachtet wird, besteht unmittelbare Gefahr für Leben und Gesundheit von Personen!

⚠ WARNUNG

Verletzungsgefahr!

Wenn dieser Sicherheitshinweis nicht beachtet wird, besteht Gefahr für Leben und Gesundheit von Personen!

⚠ VORSICHT

Schädigung von Personen!

Wenn dieser Sicherheitshinweis nicht beachtet wird, können Personen geschädigt werden!

HINWEIS

Schädigung von Umwelt/Geräten oder Datenverlust

Wenn dieser Hinweis nicht beachtet wird, können Umweltschäden, Gerätebeschädigungen oder Datenverlust entstehen.

Tipp oder Fingerzeig

Dieses Symbol kennzeichnet Informationen, die zum besseren Verständnis beitragen.

1.3 Ausgabestände der Dokumentation

Version	Kommentar
5.6	• Ergänzung Kapitel „Systemübersicht“ • Strukturupdate
5.5	• Ergänzung Kapitel „Kanäle in Info Data einblenden“ • Strukturupdate
5.4	• Update Kapitel „Einrichtung im TwinCAT Systemmanager“ • Strukturupdate
5.3	• Ergänzung Beispielprogramm • Strukturupdate
5.2	• Kapitel „Distributed Clocks“, “TwinCAT & Zeit“ aktualisiert
5.1	• Kapitel „TwinCAT Quick Start“ ergänzt
5.0	• Migration • Update Struktur
4.4	• Update Kapitel "Interne und externe EtherCAT Synchronisierung"
4.3	• Anpassung CurTaskTime
4.2	• Aktualisierung Ordner-Struktur
4.1	• Ergänzungen Kapitel "Hinweise ESI-Gerätebeschreibung" • Kapitel "EtherCAT AL Status Codes" hinzugefügt
4.0	• Ergänzungen Kapitel "Einrichtung im TwinCAT System Manager"
3.3	• Ergänzung Kapitel "Versionsidentifikation EtherCAT Geräte - online"
3.2	• Ergänzung Kapitel "Hot Connect, Fast Hot Connect"
3.1	• Ergänzung Kapitel "Online Konfigurationserstellung 'Scannen' "
3.0	• Ergänzung Kapitel "EtherCAT Diagnose"
2.9	• Ergänzung Beispielprogramm
2.8	• Ergänzung Kapitel "CoE-Interface - Parameterverwaltung im EtherCAT System"
2.7	• Ergänzung Kapitel "Identifizierung der aktuellen Anschlussstelle"
2.6	• Ergänzung Hinweis Logger-Fenster"
2.5	• Ergänzung Kapitel "Settings EtherCAT PDO"
2.4	• Ergänzung Kapitel "EtherCAT"
2.3	• Ergänzung Kapitel "Kabel-Redundanz"
2.2	• Ergänzung Kapitel "EtherCAT Datenaustausch"
2.1	• Ergänzung Kapitel "DC Unterstützung Buskoppler"
2.0	• UL Hinweis ergänzt • Ergänzung Beispiel
1.9	• Kapitel "Firmware" aktualisiert • Ergänzung Beispiel
1.8	• Kapitel "Kabel-Redundanz" ergänzt
1.7	• CoE Grundlagen ergänzt
1.6	• Korrekturen TwinCAT-Zeiten
1.5	• Ergänzung Beispiele
1.4	• Korrekturen TwinCAT-Zeiten
1.3	• TwinCAT 2.11 Beschreibung hinzugefügt
1.2	• Beschreibung HotConnect hinzugefügt • FastMode ergänzt
1.1	• Beschreibung Master Kabel-Redundanz hinzugefügt
1.0	• Korrekturen, erste Veröffentlichung

2 EtherCAT Grundlagen

2.1 Systemübersicht

2.1.1 Grenzen bisheriger Ethernet-Kommunikationsansätze

Es gibt viele verschiedene Ansätze, mit denen Ethernet echtzeitfähig gemacht werden soll: so wird z. B. das Zugriffsverfahren CSMA/CD durch überlagerte Protokollschichten außer Kraft gesetzt und durch ein Zeitscheibenverfahren oder durch Polling ersetzt; andere Vorschläge sehen spezielle Switches vor, die Ethernet Pakete zeitlich präzise kontrolliert verteilen. Diese Lösungen mögen Datenpakete mehr oder weniger schnell und exakt zu den angeschlossenen Ethernet Knoten transportieren – die Zeiten, die für die Weiterleitung zu den Ausgängen oder Antriebsreglern und für das Einlesen der Eingangsdaten benötigt werden, sind jedoch stark implementierungsabhängig. Speziell bei modularen E/A-Systemen kommt hier in der Regel noch ein Sub-Bus hinzu, der wie der Beckhoff K-Bus zwar synchronisiert und schnell sein kann, jedoch prinzipbedingt stets kleine Verzögerungen zur Kommunikation hinzufügt.

Wenn für jeden Teilnehmer individuelle Ethernet Frames Verwendung finden, so ist zudem die Nutzdatenrate prinzipiell sehr gering: Das kürzeste Ethernet Frame ist 84 Bytes lang (incl. Inter Packet Gap IPG). Wenn z. B. ein Antrieb zyklisch 4 Bytes Istwert und Status sendet und entsprechend 4 Bytes Sollwert und Kontrollwort empfängt, so wird bei 100% Buslast (also unendlich kurzer Antwortzeit des Antriebs) nur eine Nutzdatenrate von 4/84 = 4,7% erreicht. Bei durchschnittlich 10 µs Antwortzeit sinkt die Rate schon auf 1,9%. Diese Limitierungen gelten für alle Echtzeit-Ethernet Ansätze, die an jeden Teilnehmer ein Ethernet Frame senden bzw. erwarten – und zwar unabhängig von den verwendeten Protokollen innerhalb des Ethernet Frames.

2.1.2 Neuer Lösungsansatz

Mit der EtherCAT-Technologie werden diese prinzipiellen Begrenzungen anderer Ethernet-Lösungen überwunden: das Ethernet Paket wird nicht mehr in jeder Anschaltung zunächst empfangen, dann interpretiert und die Prozessdaten weiterkopiert. Die neu entwickelten FMMU (Fieldbus Memory Management Unit) in jeder E/A-Klemme entnimmt die für sie bestimmten Daten, während das Telegramm das Gerät durchläuft. Ebenso werden Eingangsdaten im Durchlauf in das Telegramm eingefügt. Die Telegramme werden dabei nur wenige Nanosekunden verzögert.

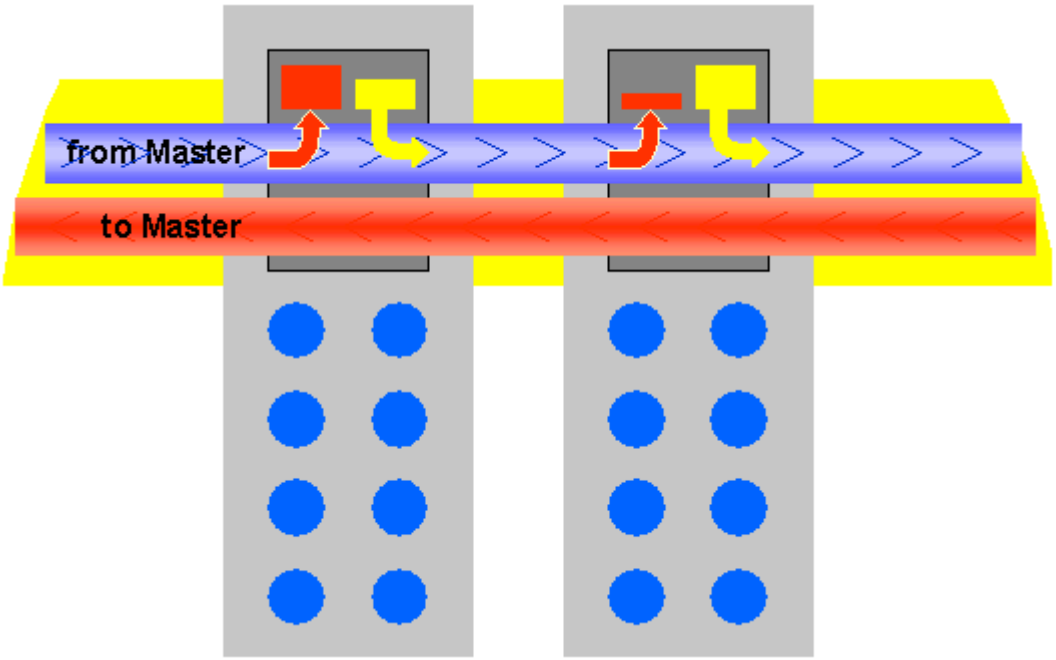


Abb. 1: Telegrammbearbeitung im Durchlauf

Da ein Ethernet-Frame sowohl in Sende- als auch in Empfangsrichtung die Daten vieler Teilnehmer erreicht, steigt die Nutzdatenrate auf ca. 80% an. Dabei werden die Voll-Duplex Eigenschaften von 100BaseTx vollständig ausgenutzt, so dass effektive Datenraten von über 100 Mbit/s (bis zu 80% von 2 x 100 Mbit/s) erreichbar sind.

Ethernet bis in die Klemme

Das Ethernet-Protokoll gemäß IEEE 802.3 bleibt bis in die einzelne Klemme erhalten, der Sub-Bus entfällt. Lediglich die Übertragungsphysik wird im Koppler von Twisted Pair bzw. Lichtleiterphysik auf E-Bus gewandelt, um den Anforderungen der elektronischen Reihenklemme gerecht zu werden.

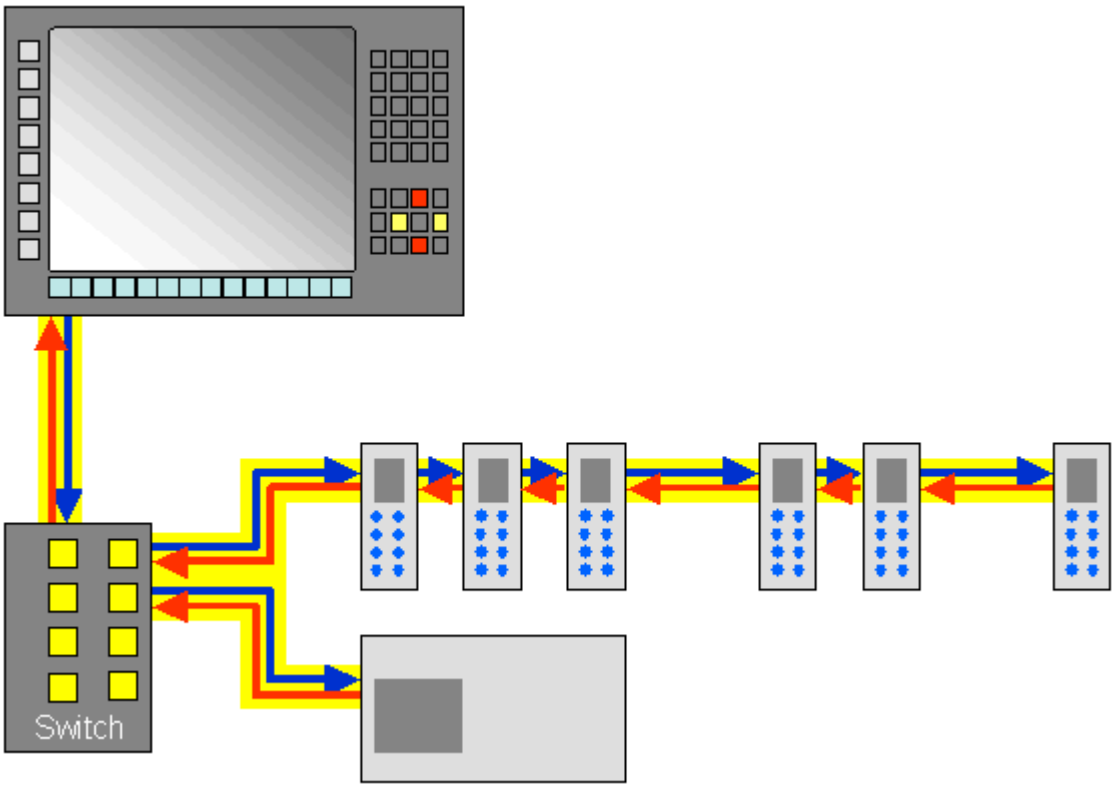


Abb. 2: Voll-Duplex Ethernet im Ring, ein Frame für viele Teilnehmer. Die EtherCAT System-Architektur steigert den Kommunikations-Wirkungsgrad

Die FMMU-Technologie kann steuerungsseitig durch den TwinCAT Y-Treiber für Ethernet ergänzt werden. Dieser bindet sich transparent in das System ein, so dass er als betriebssystemkonformer Netzwerktreiber und zusätzlich als TwinCAT-Feldbuskarte erscheint. Auf der Sendeseite wird über interne Priorisierung und Puffer sichergestellt, dass Ethernet-Frames aus dem Echtzeitsystem immer dann eine freie Sendeleitung vorfinden, wenn sie an der Reihe sind. Die Ethernet-Frames des Betriebssystems werden erst danach in den Lücken verschickt, wenn entsprechend Zeit ist.

Auf der Empfangsseite werden alle empfangenen Ethernet-Frames vom TwinCAT I/O-System überprüft und die echtzeitrelevanten herausgefiltert. Alle anderen Frames werden nach der Überprüfung außerhalb des Echtzeitkontextes an das Betriebssystem übergeben.

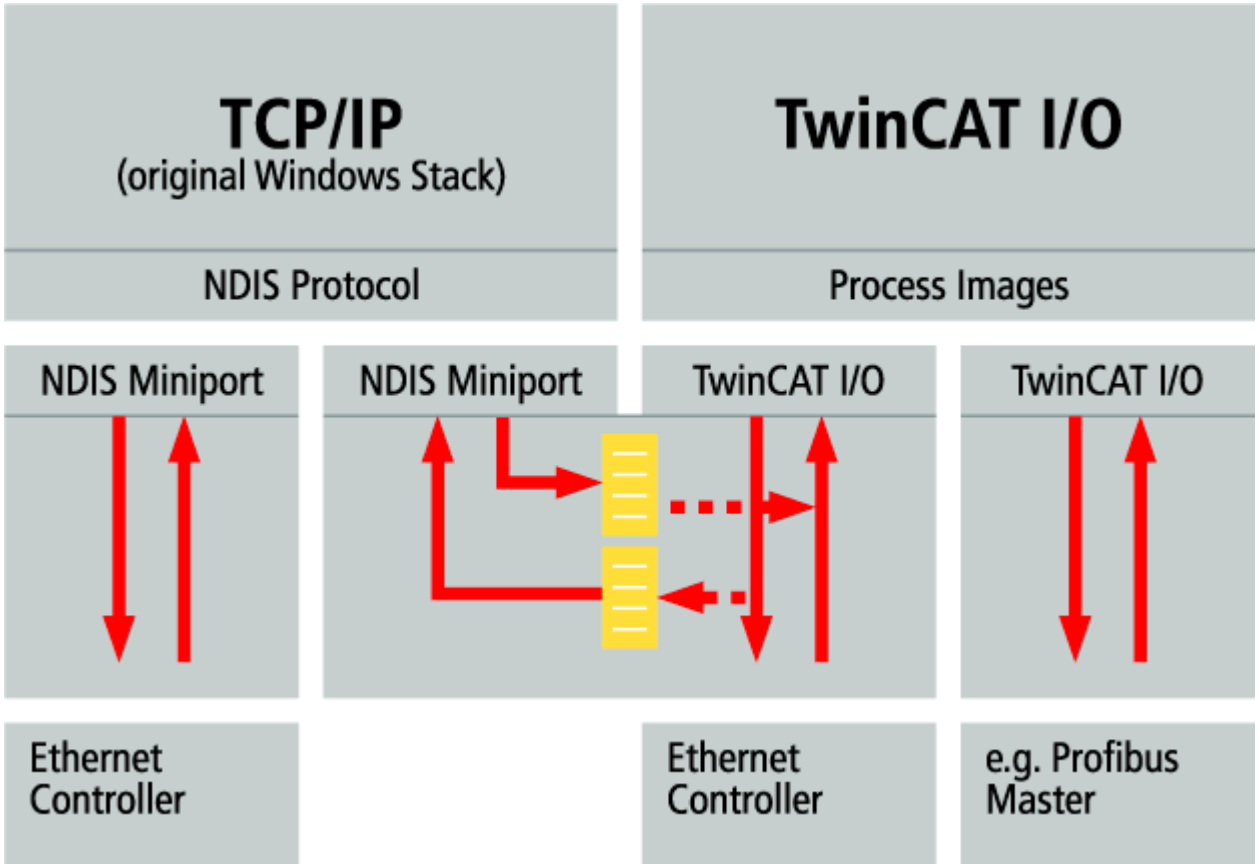


Abb. 3: Betriebssystemkonformer TwinCAT Y-Treiber:

Als Hardware in der Steuerung kommen sehr preiswerte handelsübliche Standard Netzwerk-Interface-Karten (NIC) zum Einsatz. Der Datentransfer zum PC erfolgt per DMA (Direct Memory Access); es wird also keine CPU-Performance für den Netzwerkzugriff abgezweigt.

Da die Ethernet-Funktionalität des Betriebssystems vollständig erhalten bleibt, können alle betriebssystemkonformen Protokolle parallel auf demselben physikalischen Netzwerk betrieben werden. Dies umfasst nicht nur Standard IT-Protokolle wie TCP/IP, HTTP, FTP oder SOAP, sondern auch praktisch alle Industrial-Ethernet-Protokolle wie Modbus TCP, ProfiNet oder Ethernet/IP.

2.1.3 Systemeigenschaften

Protokoll

Das für Prozessdaten optimierte EtherCAT-Protokoll wird dank eines speziellen Ether-Types direkt im Ethernet-Frame transportiert. Es kann aus mehreren Sub-Telegrammen bestehen, die jeweils einen Speicherbereich des bis zu 4 Gigabyte großen logischen Prozessabbildes bedienen. Die datentechnische Reihenfolge ist dabei unabhängig von der physikalischen Reihenfolge der Ethernet-Klemmen im Netz, es kann wahlfrei adressiert werden. Broadcast, Multicast und Querkommunikation zwischen Slaves sind möglich. Die Übertragung direkt im Ethernet-Frame wird stets dann eingesetzt, wenn EtherCAT-Komponenten im gleichen Subnetz wie der Steuerungsrechner betrieben werden.

Der Einsatzbereich von EtherCAT ist jedoch nicht auf ein Subnetz beschränkt: EtherCAT UDP verpackt das EtherCAT Protokoll in UDP/IP-Datagramme. Hiermit kann jede Steuerung mit Ethernet-Protokoll-Stack EtherCAT-Systeme ansprechen. Selbst die Kommunikation über Router hinweg in andere Subnetze ist möglich. Selbstverständlich hängt die Leistungsfähigkeit des Systems in dieser Variante von den Echtzeiteigenschaften der Steuerung und ihrer Ethernet-Protokollimplementierung ab. Die Antwortzeiten des EtherCAT-Netzwerks an sich werden jedoch nur minimal eingeschränkt: lediglich in der ersten Station muss das UDP-Datagramm entpackt werden.

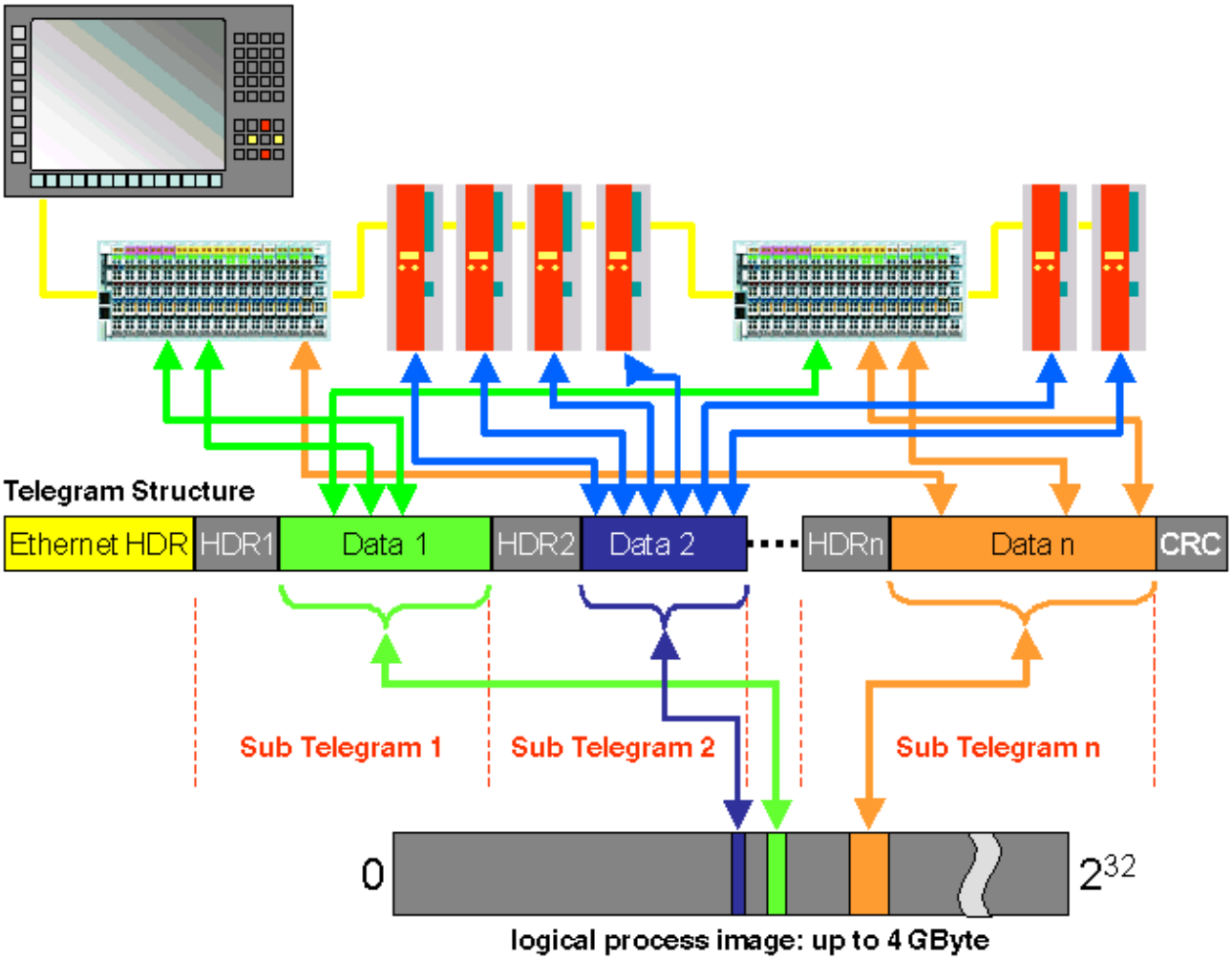


Abb. 4: Protokollstruktur EtherCAT

Protokollstruktur: Die Prozessabbild-Zuordnung ist frei konfigurierbar. Daten werden direkt in der E/A-Klemme an die gewünschte Stelle des Prozessabbilds kopiert: zusätzliches Mapping ist überflüssig. Der zur Verfügung stehende logische Adressraum ist mit 4 Gigabyte sehr groß.

Topologie

Linie, Baum oder Stern: EtherCAT unterstützt nahezu beliebige Topologien. Die von den Feldbussen her bekannte Bus- oder Linienstruktur wird damit auch für Ethernet verfügbar. Besonders praktisch für die Anlagenverdrahtung ist die Kombination aus Linie und Abzweigen bzw. Stichleitungen. Die hierzu benötigten Schnittstellen sind auf den Kopplern vorhanden; zusätzliche Switches werden nicht benötigt. Natürlich kann aber auch die klassische Switch-basierte Ethernet-Sterntopologie eingesetzt werden.

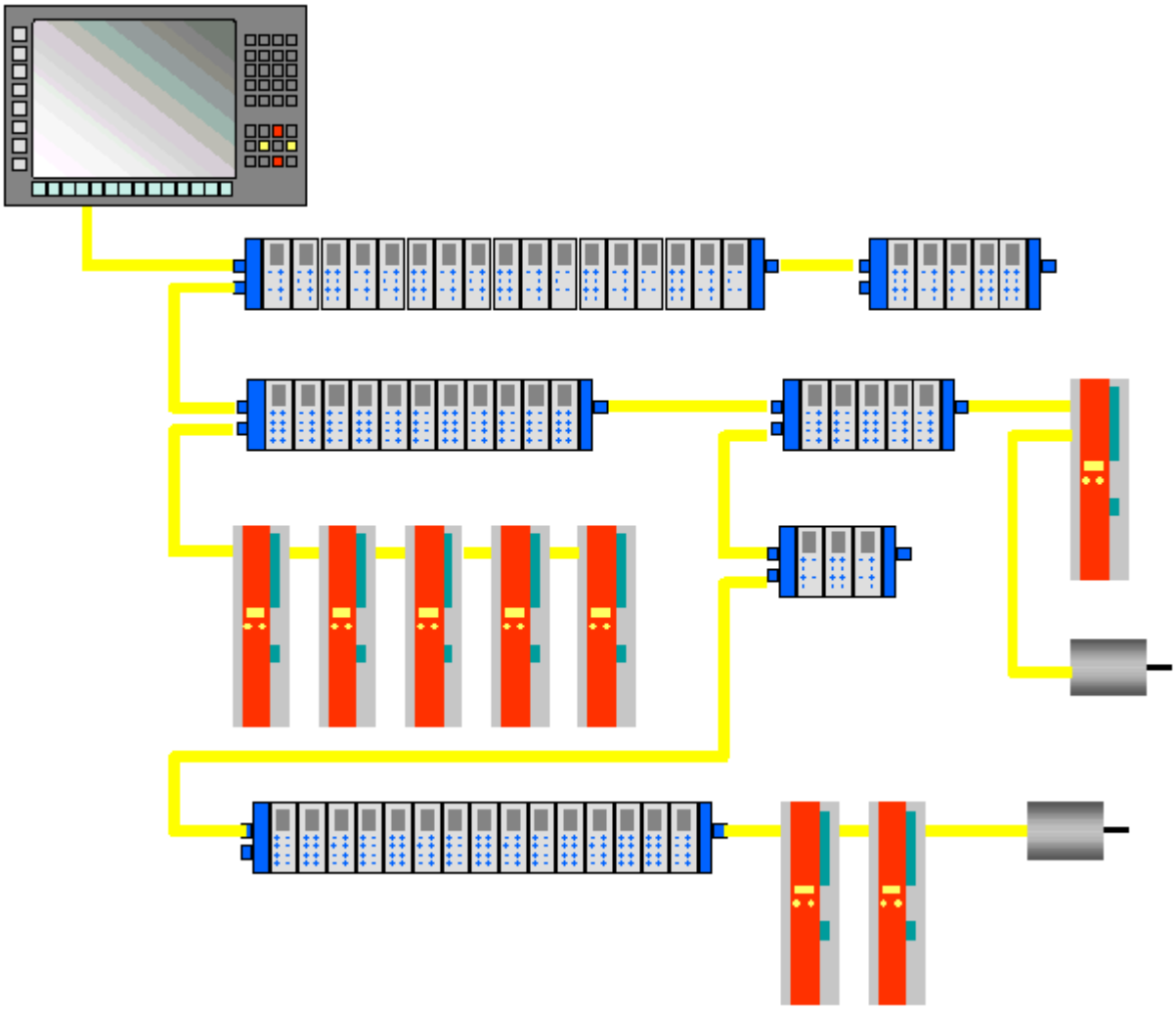


Abb. 5: Beispiel: Topologie EthetCAT

Maximale Flexibilität bei der Verdrahtung:
mit und ohne Switch, Linien- und Baumtopologien frei wähl- und kombinierbar.

Die maximale Flexibilität bei der Verdrahtung wird durch die Auswahl verschiedener Leitungen vervollständigt. Flexible und sehr preiswerte Standard Ethernet-Patch-Kabel übertragen die Signale auf Ethernet-Art (100Base-TX). Die gesamte Bandbreite der Ethernet-Vernetzung – wie verschiedenste Lichtleiter und Kupferkabel – kann in der Kombination mit Switches oder Medienumsetzern zum Einsatz kommen.

Distributed Clocks

Der exakten Synchronisierung kommt immer dann eine besondere Bedeutung zu, wenn räumlich verteilte Prozesse gleichzeitige Aktionen erfordern. Das kann z. B. in Applikationen der Fall sein, wo mehrere Servo-Achsen gleichzeitig koordinierte Bewegungen ausführen.

Der leistungsfähigste Ansatz zur Synchronisierung ist der exakte Abgleich verteilter Uhren – wie im neuen Standard IEEE 1588 beschrieben. Im Gegensatz zur vollsynchronen Kommunikation, deren Synchronisationsqualität bei Kommunikationsstörungen sofort leidet, verfügen verteilte abgeglichene Uhren über ein hohes Maß an Toleranz gegenüber möglichen störungsbedingten Verzögerungen im Kommunikationssystem.

Beim EtherCAT basiert der Datenaustausch vollständig auf einer reinen Hardware-Maschine. Da die Kommunikation eine logische (und dank Vollduplex-Fast-Ethernet auch physikalische) Ringstruktur nutzt, kann die Mutter-Uhr den Laufzeitversatz zu den einzelnen Tochter-Uhren einfach und exakt ermitteln – und umgekehrt. Auf Basis dieses Wertes werden die verteilten Uhren nachgeführt und es steht eine hochgenaue netzwerkweite Zeitbasis zur Verfügung, deren Jitter deutlich unter einer Mikrosekunde beträgt.

Hochauflösende verteilte Uhren dienen aber nicht nur der Synchronisierung, sondern können auch exakte Informationen zum lokalen Zeitpunkt der Datenerfassung liefern. Steuerungen berechnen beispielsweise häufig Geschwindigkeiten aus nacheinander gemessenen Positionen. Speziell bei sehr kurzen Abtastzeiten führt schon ein kleiner zeitlicher Jitter in der Wegerfassung zu großen Geschwindigkeitssprüngen. Konsequenterweise werden mit EtherCAT auch neue, erweiterte Datentypen eingeführt (Timestamp und Oversampling Data Type). Mit dem Messwert wird die lokale Zeit mit einer Auflösung von bis zu 10 ns verknüpft - die große Bandbreite von Ethernet macht das möglich. Damit hängt dann die Genauigkeit einer Geschwindigkeitsberechnung nicht mehr vom Jitter des Kommunikationssystems ab. Sie wird um Größenordnungen besser als diejenige von Messverfahren, die auf jitterfreier Kommunikation basieren.

Performance

Mit EtherCAT werden neue Dimensionen in der Netzwerk-Performance erreicht. Dank FMMU-Chip in der Klemme und DMA-Zugriff auf die Netzwerkkarte des Masters erfolgt die gesamte Protokollbearbeitung in Hardware. Sie ist damit unabhängig von der Laufzeit von Protokoll-Stacks, von CPU-Performance oder Software-Implementierung. Die Update-Zeit für 1000 E/As beträgt nur 30 µs – einschließlich Klemmen-Durchlaufzeit. Mit einem einzigen Ethernet-Frame können bis zu 1486 Bytes Prozessdaten ausgetauscht werden – das entspricht fast 12000 digitalen Ein- und Ausgängen. Für die Übertragung dieser Datenmenge werden dabei nur 300 µs benötigt.

Für die Kommunikation mit 100 Servoachsen werden nur 100 µs benötigt. In dieser Zeit werden alle Achsen mit Sollwerten und Steuerdaten versehen und melden ihre Ist-Position und Status. Mit den Distributed-Clocks können die Achsen dabei mit einer Abweichung von deutlich weniger als einer Mikrosekunde synchronisiert werden.

Die extrem hohe Performance der EtherCAT-Technologie ermöglicht Steuerungs- und Regelungskonzepte, die mit klassischen Feldbussystemen nicht realisierbar waren. So kann beispielsweise nicht nur die Geschwindigkeitsregelung, sondern neu auch die Stromregelung verteilter Antriebe über das Ethernet-System erfolgen. Die enorme Bandbreite erlaubt es, zu jedem Datum z. B. auch Status-Informationen zu übertragen. Mit EtherCAT steht eine Kommunikationstechnologie zur Verfügung, die der überlegenen Rechenleistung moderner Industrie-PCs entspricht. Das Bussystem ist nicht mehr der Flaschenhals im Steuerungskonzept. Verteilte E/As werden schneller erfasst, als dies mit den meisten lokalen E/A-Schnittstellen möglich ist. Das EtherCAT Technologieprinzip ist skalierbar und nicht an die Baudrate von 100 MBaud gebunden – eine Erweiterung auf GBit Ethernet ist möglich.

Diagnose

Die Erfahrungen mit Feldbussystemen zeigen, dass die Verfügbarkeit und Inbetriebnahmezeiten entscheidend von der Diagnosefähigkeit abhängen. Nur eine schnelle und präzise erkannte und eindeutig lokalisierbare Störung kann kurzfristig behoben werden. Deshalb wurde bei der Entwicklung des EtherCAT-Systems besonderer Wert auf vorbildliche Diagnoseeigenschaften gelegt.

Bei der Inbetriebnahme gilt es zu prüfen, ob die Ist-Konfiguration der E/A-Klemmen mit der Soll-Konfiguration übereinstimmt. Auch die Topologie sollte der gespeicherten Konfiguration entsprechen. Durch die eingebaute Topologie-Erkennung bis hinunter zu den einzelnen Klemmen kann nicht nur diese Überprüfung beim Systemstart stattfinden – auch ein automatisches Einlesen des Netzwerkes ist möglich (Konfigurations-Upload).

Bitfehler in der Übertragung werden durch die Auswertung der CRC-Prüfsumme zuverlässig erkannt: das 32 Bit CRC-Polynom weist eine minimale Hamming-Distanz von 4 auf. Neben der Bruchstellenerkennung und -lokalisierung erlauben Protokoll, Übertragungsphysik und Topologie des EtherCAT-Systems eine individuelle Qualitätsüberwachung jeder einzelnen Übertragungsstrecke. Die automatische Auswertung der entsprechenden Fehlerzähler ermöglicht die exakte Lokalisierung kritischer Netzwerkabschnitte. Schleichende oder wechselnde Fehlerquellen wie EMV-Einflüsse, fehlerhafte Steckverbindungen oder Kabelschäden werden erkannt und lokalisiert, auch wenn sie die Selbstheilungsfähigkeit des Netzwerkes noch nicht überfordern.

Integration von Beckhoff Standard-Busklemmen

Neben den neuen Busklemmen mit E-Bus-Anschluss (ELxxxx) lassen sich auch sämtliche Busklemmen aus dem bewährten Standardprogramm mit K-Bus-Anschluss (KLxxxx) über den Buskoppler BK1120 oder BK1250 anschließen. Damit sind Kompatibilität und Durchgängigkeit zum bestehenden Beckhoff Busklemmensystemen gewährleistet. Bestehende Investitionen werden geschützt.

2.1.4 Einsatzfelder

Regelkreise über den Bus

Im Verbund mit schnellen Maschinensteuerungen lässt sich die überragende Performance von EtherCAT natürlich besonders gut darstellen. Es steht ein Bussystem zur Verfügung, das Zugriffsgeschwindigkeiten wie bei lokalen E/As bietet. Mit EtherCAT lässt sich nicht nur der Lageregelkreis, sondern auch der Geschwindigkeitsregelkreis oder gar der Stromregelkreis über den Bus schließen; kostengünstige Antriebsregler sind die Folge. EtherCAT bringt damit die Leistungsfähigkeit der schnellen PC-basierten Steuerungstechnik voll zur Geltung. Die IPCs werden dabei klein und kostengünstig, da auf Steckplätze verzichtet werden kann: Erweiterungskarten werden ebenfalls über EtherCAT angesprochen.

Doch EtherCAT eignet sich nicht nur für extrem schnelle Anwendungen. Da die Protokollbearbeitung komplett in Hardware erfolgt, stellt die Technologie nur geringe Anforderungen an die Master-Anschaltung: ein handelsüblicher Ethernet Controller genügt. Der Master muss lediglich zyklisch oder bei Bedarf EtherCAT-Frames abschicken – Mapping, Adressierung, Knotenüberwachung etc. erfolgt in den Slave-ASICs. Das Eingangs-Prozessabbild kommt vollständig sortiert und damit fertig aufbereitet für die Steuerungsapplikation im Master an. Bei einfachen zyklischen Applikationen bis 1486 Byte Prozessabbildgröße ist lediglich ein einziges Telegramm im Master zu bearbeiten – und das besteht fast ausschließlich aus den Prozessdaten. Größere Prozessabbilder werden auf mehrere Frames verteilt.

EtherCAT dürfte damit diejenige Feldbus-Technologie sein, die für die geringsten Anforderungen im Master sorgt.

EtherCAT statt PCI

Mit der fortschreitenden Verkleinerung der PC-Komponenten wird die Baugröße von Industrie-PCs zunehmend von der Anzahl der benötigten Steckplätze bestimmt. Die Bandbreite von Fast-Ethernet zusammen mit der Datenbreite der EtherCAT Kommunikations-Hardware (FMMU-Chip) ermöglicht hier, neue Wege zu gehen: klassisch im IPC vorgesehene Schnittstellen werden in intelligente Schnittstellenklemmen am EtherCAT ausgelagert. Über einen einzigen Ethernet-Port im PC können dann neben den dezentralen E/As, Achsen und Bediengeräten auch komplexe Systeme wie Feldbus-Master, schnelle serielle Schnittstellen, Gateways und andere Kommunikations-Interfaces angesprochen werden. Selbst weitere Ethernet-Geräte mit beliebigen Protokollvarianten lassen sich über dezentrale Switchport-Klemmen anschließen. Der zentrale IPC wird kleiner und damit kostengünstiger. Eine Ethernet-Schnittstelle genügt zur kompletten Kommunikation mit der Peripherie.

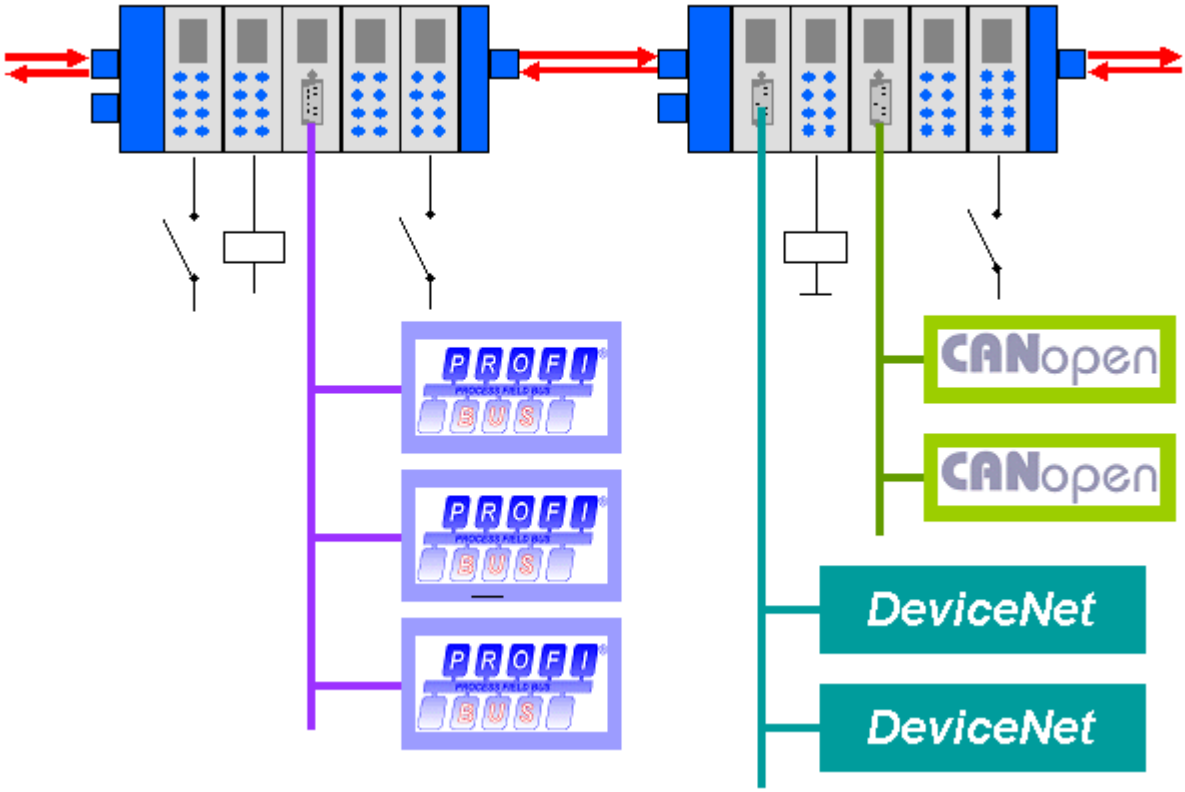


Abb. 6: Dezentralisierung durch intelligente Schnittstellenklemmen am EtherCAT-Fekdbus

Feldbusgeräte (z. B. Profibus, CANopen, DeviceNet, AS-Interface etc.)
werden durch dezentrale Feldbus-Masterklemmen integriert.

2.1.5 EtherCAT ist offen

Die EtherCAT Technologie ist nicht nur vollständig Ethernet-kompatibel, sondern schon vom Konzept her durch besondere Offenheit gekennzeichnet: das Protokoll verträgt sich mit weiteren Ethernet-basierten Diensten und Protokollen auf dem gleichen physikalischen Netz – in der Regel sogar mit nur minimalen Einbußen bei der Performance: im EtherCAT OpenMode direkt gleichzeitig nebeneinander, im EtherCAT DirectMode getunnelt „auf“ den EtherCAT Frame.
Beliebige Ethernet-Geräte können innerhalb des EtherCAT-Strangs via Switchport angeschlossen werden. Geräte mit Feldbusschnittstelle werden über EtherCAT Feldbus-Master-Klemmen integriert. Die Protokollvariante UDP lässt sich auf jedem Socket-Interface implementieren.

EtherCAT Geräte können zusätzlich über einen TCP/IP-Stack verfügen und damit nach außen wie ein normales Ethernet-Gerät auftreten. Der Master fungiert dabei wie ein Switch, der die Frames gemäß der Adressinformation zu den entsprechenden Teilnehmern weiterleitet. Statt Switch-Ports werden lediglich die automatisch im Hochlauf vergebenen EtherCAT-Adressen verwendet.

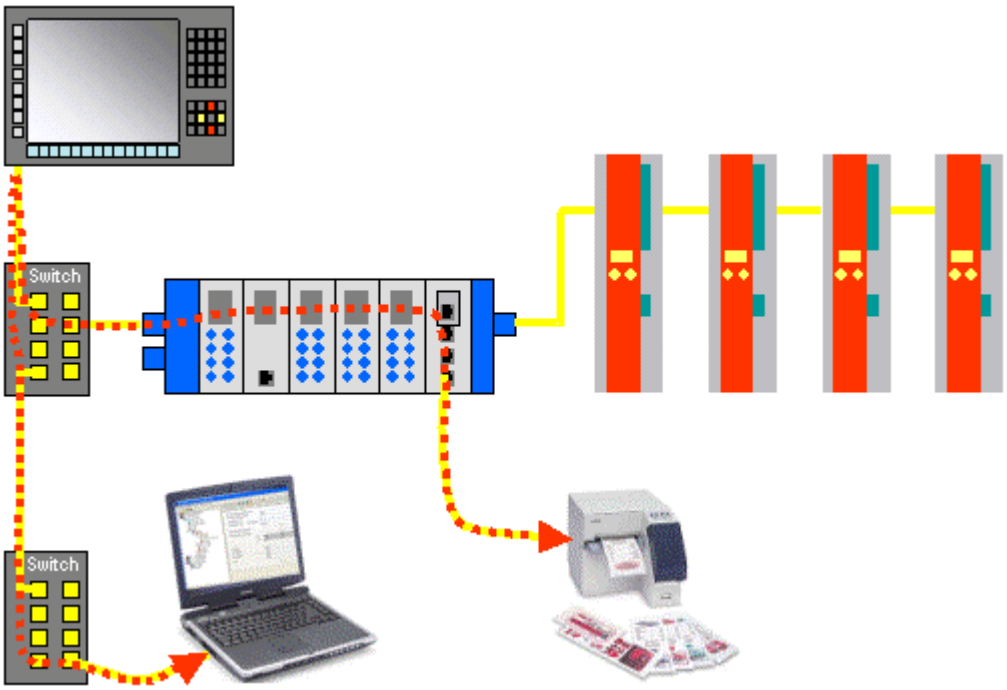


Abb. 7: Switchport-Klemmen tunneln Ethernet-Frames durch das EtherCAT Protokoll.
Hinweis: das Bild zeigt eine Mischung aus EtherCAT OpenMode mit Ethernet/EtherCAT über Switches und EtherCAT DirectMode mit On-the-fly-Verarbeitung in den untergeordneten Teilnehmern.

Entsprechend funktionieren auch die Switchport-Klemmen: beliebige Ethernet-Geräte können angeschlossen werden. Die Ethernet-Frames werden durch das EtherCAT Protokoll getunnelt, wie es im Internet üblich ist (z. B. VPN, PPPoE (DSL) etc.). Das EtherCAT-Netzwerk ist dabei für das Ethernet-Gerät voll transparent, und die Echtzeiteigenschaften des EtherCAT werden nicht beeinträchtigt.

EtherCAT Technology Group

Die EtherCAT Technology Group (ETG, http://www.ethercat.org) ist die internationale Hersteller- und Anwendervereinigung, die sich die Verbreitung, Weiterentwicklung, und Unterstützung von EtherCAT zum Ziel gesetzt hat. Jede Firma ist zur Mitgliedschaft eingeladen.

International genormt

Die EtherCAT Technology Group ist offizieller Normungspartner der IEC (International Electrotechnical Commission). Die EtherCAT-Spezifikation ist als IEC/PAS 62407 von der IEC veröffentlicht und auch über die IEC (http://www.iec.ch) erhältlich. Aktuell erfolgt die Integration in die Feldbusstandards IEC 61158, IEC 61800-7 und ISO 15745.

2.1.6 Zusammenfassung und Ausblick

Überragende Performance, einfachste Verdrahtung und Offenheit für andere Protokolle kennzeichnen EtherCAT. Wo herkömmliche Feldbussysteme an ihre Grenzen kommen, setzt EtherCAT neue Maßstäbe: 1000 E/A s in 30 µs, wahlweise verdrillte Zweidrahtleitung (Twisted Pair) oder Lichtleiter, und dank Ethernet- und Internet-Technologien optimale vertikale Integration. Mit EtherCAT kann die aufwändige Ethernet-Sterntopologie durch eine einfache Linienstruktur ersetzt werden – teure Infrastrukturkomponenten entfallen. Wahlweise kann EtherCAT aber auch klassisch mit Switches verkabelt werden, um andere Ethernet-Geräte zu integrieren. Wo andere Echtzeit-Ethernet-Ansätze spezielle Anschaltungen in der Steuerung erfordern, kommt EtherCAT mit äußerst kostengünstigen normalen Ethernet-Karten (NICs) aus.

Ethernet bis in die Reihenklemme wird durch EtherCAT technisch möglich und wirtschaftlich sinnvoll. Volle Ethernet-Kompatibilität, Internet-Technologien auch in einfachsten Geräten, maximale Nutzung der großen Ethernet-Bandbreite, hervorragende Echtzeiteigenschaften bei niedrigen Kosten sind herausragende Eigenschaften dieses neuen Netzwerkes. Als schneller Antriebs- und E/A-Bus am Industrie-PC oder auch in Kombination mit kleiner Steuerungstechnik wird EtherCAT vielfältige Einsatzmöglichkeiten finden.

2.2 Grundlagen

2.2.1 EtherCAT State Machine

Über die EtherCAT State Machine (ESM) wird der Zustand des EtherCAT-Slaves gesteuert. Je nach Zustand sind unterschiedliche Funktionen im EtherCAT-Slave zugänglich bzw. ausführbar. Insbesondere während des Hochlaufs des Slaves müssen in jedem State spezifische Kommandos vom EtherCAT Master zum Gerät gesendet werden.

Es werden folgende Zustände unterschieden:

- Init
- Pre-Operational
- Safe-Operational und
- Operational
- Boot

Regulärer Zustand eines jeden EtherCAT Slaves nach dem Hochlauf ist der Status OP.

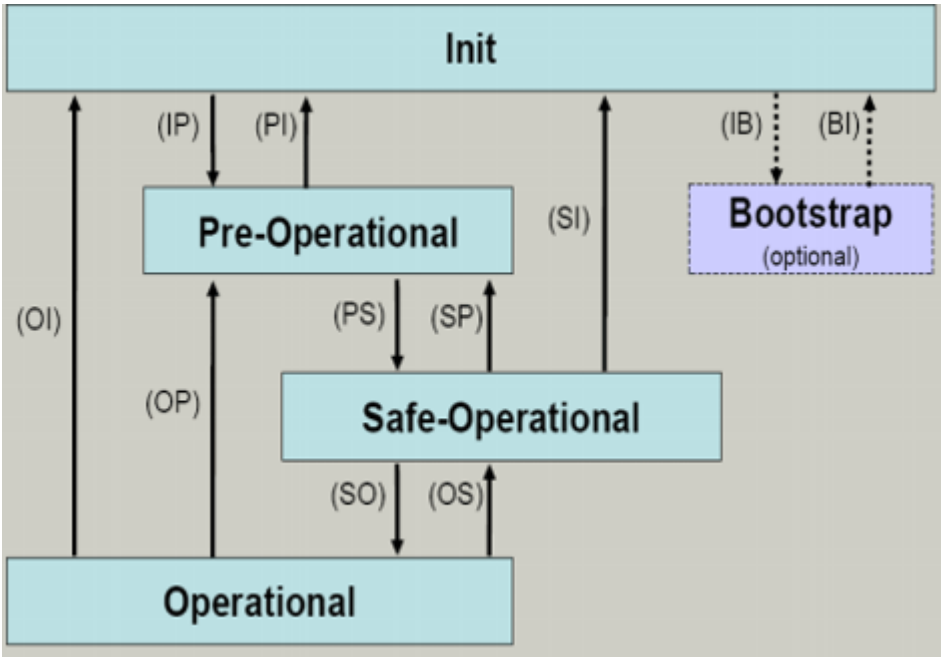


Abb. 8: Zustände der EtherCAT State Machine

Init

Nach dem Einschalten befindet sich der EtherCAT-Slave im Zustand Init. Dort ist weder Mailbox- noch Prozessdatenkommunikation möglich. Der EtherCAT-Master initialisiert die Sync-Manager-Kanäle 0 und 1 für die Mailbox-Kommunikation.

Pre-Operational (Pre-Op)

Beim Übergang von *Init* nach *Pre-Op* prüft der EtherCAT-Slave, ob die Mailbox korrekt initialisiert wurde.

Im Zustand *Pre-Op* ist Mailbox-Kommunikation aber keine Prozessdaten-Kommunikation möglich. Der EtherCAT-Master initialisiert die Sync-Manager-Kanäle für Prozessdaten (ab Sync-Manager-Kanal 2), die FMMU-Kanäle und falls der Slave ein konfigurierbares Mapping unterstützt das PDO-Mapping oder das Sync-Manager-PDO-Assignement. Weiterhin werden in diesem Zustand die Einstellungen für die Prozessdatenübertragung sowie ggf. noch klemmenspezifische Parameter übertragen, die von den Defaulteinstellungen abweichen.

Safe-Operational (Safe-Op)

Beim Übergang von *Pre-Op* nach *Safe-Op* prüft der EtherCAT-Slave, ob die Sync-Manager-Kanäle für die Prozessdatenkommunikation sowie ggf. ob die Einstellungen für die Distributed-Clocks korrekt sind. Bevor er den Zustandswechsel quittiert, kopiert der EtherCAT-Slave aktuelle Inputdaten in die entsprechenden DP-RAM-Bereiche des EtherCAT-Slave-Controllers (ECSC).

Im Zustand *Safe-Op* ist Mailbox- und Prozessdaten-Kommunikation möglich, allerdings hält der Slave seine Ausgänge im sicheren Zustand und gibt sie noch nicht aus. Die Inputdaten werden aber bereits zyklisch aktualisiert.

Ausgänge im SAFEOP

Die standardmäßig aktivierte Watchdogüberwachung bringt die Ausgänge im Modul in Abhängigkeit von den Einstellungen im SAFEOP und OP in einen sicheren Zustand - je nach Gerät und Einstellung z. B. auf AUS. Wird dies durch Deaktivieren der Watchdogüberwachung im Modul unterbunden, können auch im Geräte-Zustand SAFEOP Ausgänge geschaltet werden bzw. gesetzt bleiben.

Operational (Op)

Bevor der EtherCAT-Master den EtherCAT-Slave von *Safe-Op* nach *Op* schaltet, muss er bereits gültige Outputdaten übertragen.

Im Zustand *Op* kopiert der Slave die Ausgangsdaten des Masters auf seine Ausgänge. Es ist Prozessdaten- und Mailbox-Kommunikation möglich.

Boot

Im Zustand *Boot* kann ein Update der Slave-Firmware vorgenommen werden. Der Zustand *Boot* ist nur über den Zustand *Init* zu erreichen.

Im Zustand *Boot* ist Mailbox-Kommunikation über das Protokoll *File-Access over EtherCAT (FoE)* möglich, aber keine andere Mailbox-Kommunikation und keine Prozessdaten-Kommunikation.

2.2.2 CoE Interface

2.2.2.1 CoE-Interface - Parameterverwaltung im EtherCAT System

Hintergrund

In einer Automatisierungsumgebung werden viele unterschiedliche Geräte eingesetzt. Diese Geräte können einzeln oder im Verbund an einem Bussystem zusammen eingesetzt werden. Solche Geräte können Steuerungen, Drehgeber, Servoverstärker, Motore, I/O-Module oder Sensoren u.a. sein.

Je nach Komplexität muss ein solches Gerät für den jeweiligen Bedarfsfall parametrierbar/einstellbar sein. Bei einem einfachen digitalen 24 V-Eingang mit fester Schaltschwelle und -verzögerung ist eine Parametrierung unter Umständen nicht nötig, jedoch wird man bei einem Drehgeber nicht darauf verzichten können (z. B. Anzahl der Striche, absolut oder relativ, Datenformat, usw.)

Darüber hinaus kann es von Interesse sein, im Gerät bei der Produktion oder im Betrieb Daten abzulegen. Der Hersteller könnte Produktionsdaten ablegen wie Gerätename, Seriennummer, Firmwarestand, Abgleichdaten oder Herstellungsdatum, ggf. mit Zugangs- oder Änderungsschutz versehen.
Der Anwender könnte Anwenderabgleichdaten und die einsatzspezifischen Einstellungen im Gerät hinterlegen.

Um hier eine anwenderfreundliche Schnittstelle zur Gerätebedienung zu schaffen, sind von unterschiedlichen Organisationen verschiedene Standards angelegt worden, in denen definiert wurde:

- welche Geräteklassen es gibt (z. B.: Klasse "Drehgeber", "analoges Eingangsmodul")
- über welche Parameter jeder Vertreter einer solchen Klasse verfügt (obligate und optionale Elemente)
- an welcher Stelle und mit welchem Mechanismus diese Parameter zu finden und zu ändern sind.

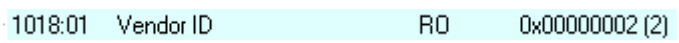
EtherCAT lehnt sich hier an den so genannten CoE-Standard an, Can-Application-protocoll-over-EtherCAT.

Can-Over-EtherCAT

Die CiA-Organisation (CAN in Automation) verfolgt u.a. das Ziel, durch Standardisierung von Gerätebeschreibungen Ordnung und Austauschbarkeit zwischen gleichartigen Geräten herzustellen. Zu diesem Zweck werden so genannte Profile definiert, die die veränderlichen und unveränderlichen Parameter eines Gerätes abschließend beschreiben. Solch ein Parameter umfasst mindestens folgende Eigenschaften:

- Indexnummer - zur eindeutigen Identifizierung aller Parameter. Um zusammengehörige Parameter zu kennzeichnen und zu ordnen, unterteilt sich die Indexnummer in einen Haupt- und Subindex.
  - Hauptindex
  - Subindex, abgesetzt durch einen Doppelpunkt ":"
- Offizieller Name - als verständlicher, selbsterklärender Text
- Angaben zur Veränderbarkeit, z. B. ob er nur gelesen oder auch beschrieben werden kann
- Einen Wert - je nach Parameter kann der Wert ein Text, eine Zahl oder wieder ein anderer Parameterindex sein.

Beispiel: der Parameter "Herstellerkennung" (Vendor ID) habe die Indexnummer 0x4120:01 und den Zahlenwert "2" als Kennzeichnung eines Beckhoff-Gerätes.
Da im maschinellen Umfeld gerne mit hexadezimalen Werten gearbeitet wird, wird der Parameter aus Anwendersicht also dargestellt als



mit der Eigenschaft RO (read-only), denn die Herstellerkennung soll vom Anwender nicht verändert werden.

Solch eine Liste an Parametern, die Gesamtheit des gerätespezifischen CoE-Verzeichnisses, kann sehr umfangreich werden. Die ersten Einträge einer Beckhoff EL3152 analogen Eingangsklemme in der Ansicht des TwinCAT System Manager lauten:

Index:Subindex	Name	read/write possible	value
1000	Device type	RO	0x00001389 (5001)
1008	Device name	RO	EL3152
1009	Hardware version	RO	08
100A	Software version	RO	08
1011:0	Restore default parameters	RO	> 1 <
1011:01	SubIndex 001	RW	0x00000000 (0)
1018:0	Identity	RO	> 4 <
1018:01	Vendor ID	RO	0x00000002 (2)
1018:02	Product code	RO	0x0C503052 (206581842)
1018:03	Revision	RO	0x00110000 (1114112)
1018:04	Serial number	RO	0x00000000 (0)

Abb. 9: CoE-Verzeichnis EL3152

Die Indexnummern werden im Profil festgelegt, sie beginnen bei EtherCAT deshalb bei 0x1000, weil die darunterliegenden Einträge nicht dargestellt werden müssen.

CoE-Verzeichnis - Verfügbarkeit

Ein EtherCAT-Teilnehmer kann, muss aber nicht über ein CoE-Verzeichnis verfügen. Einfache Slaves benötigen kein Parameterverzeichnis bzw. verfügen nicht über den zur Verwaltung nötigen Controller. Andererseits kann auch der EtherCAT Master (wie TwinCAT) als Software-EtherCAT-Gerät ein CoE-Verzeichnis verwalten.

Wenn vorhanden, ist das CoE-Verzeichnis ab dem Status PREOP in Betrieb.

Das Objektverzeichnis ist per SDO-Information-Dienst auslesbar (Service Data Objects).

CoE-Verzeichnis - Lokalisierung im EtherCAT-Slave

Das CoE-Verzeichnis als Parametersystem muss im Gerät in der Firmware (FW) im lokalen Controller verwaltet werden. Dies ist das so genannte *Online-Verzeichnis*, weil es dem Anwender nur zur Verfügung steht, wenn der EtherCAT-Slave unter Betriebsspannung in Betrieb ist und ggf. über EtherCAT-Kommunikation manipuliert werden kann.

Damit auch ohne vorhandenen Slave schon vorab die Parameter eingesehen und verändert werden können, wird üblicherweise eine Default-Kopie des gesamten Verzeichnisses in der Gerätebeschreibungsdatei ESI (XML) hinterlegt. Dies wird *Offline-Verzeichnis* genannt. Änderungen in diesem Verzeichnis wirken sich bei TwinCAT nicht auf den späteren Betrieb des Slave aus. Die xml-Dateien können auf der Beckhoff-Website im Downloadbereich bezogen werden.

Der TwinCAT System Manager 2.11 kann beide Listen anzeigen und kennzeichnet dies:

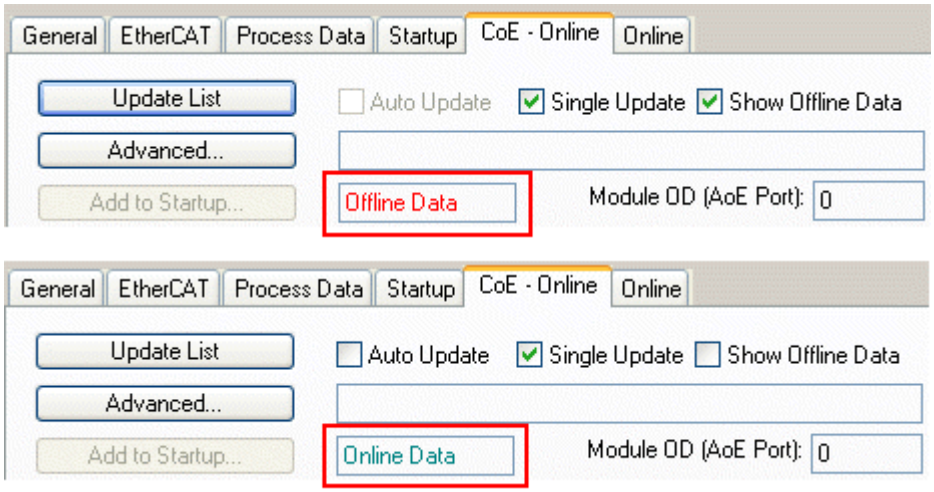


Abb. 10: Online/Offline Anzeige

Im Online Verzeichnis	Im Offline Verzeichnis
wird das reale aktuelle Verzeichnis des Slaves ausgelesen. Dies kann je nach Größe und Zykluszeit einige Sekunden dauern	wird das Offline-Verzeichnis aus der ESI-Datei angezeigt. Änderungen sind hier nicht sinnvoll bzw. möglich.
wird die tatsächliche Identität angezeigt	wird in der Identität der konfigurierte Stand angezeigt
wird der Firmware- und Hardware-Stand des Gerätes laut elektronischer Auskunft angezeigt	wird kein Firmware- oder Hardware-Stand angezeigt, da dies Eigenschaften des realen Gerätes sind
ist ein grünes **Online** im TwinCAT System Manager, Karteireiter *CoE-Online* zu sehen	ist ein rotes **Offline** im TwinCAT System Manager, Karteireiter *CoE-Online* zu sehen

Einteilung

Es sind verschiedene Typen für CoE-Parameter möglich wie String (Text), Integer-Zahlen, Bool'sche Werte oder größere Byte-Felder. Damit lassen sich ganz verschiedene Eigenschaften beschreiben. Beispiele für solche Parameter sind Herstellerkennung, Seriennummer, Prozessdateneinstellungen, Gerätename, Abgleichwerte für analoge Messung oder Passwörter.

Die für den anwendungsorientierten EtherCAT-Feldbusanwender wichtigen Bereiche im Slave-CoE sind

- 0x1000: hier sind feste Identitäts-Informationen zum Gerät hinterlegt wie Name, Hersteller, Seriennummer etc. Außerdem liegen hier Angaben über die aktuellen und verfügbaren Prozessdatenkonstellationen.
- 0x8000: hier sind die für den Betrieb erforderlichen funktionsrelevanten Parameter für alle Kanäle zugänglich wie Filtereinstellung oder Ausgabefrequenz.

Ferner interessant sind die Bereiche

- 0x4000: hier liegen in manchen EtherCAT-Geräten alternativ zum 0x8000-Bereich die Kanalparameter.
- 0x6000: hier liegen die Eingangs-PDO ("Eingang" aus Sicht des EtherCAT-Masters)
- 0x7000: hier liegen die Ausgangs-PDO ("Ausgang" aus Sicht des EtherCAT-Masters)

Kanalweise Ordnung

Das CoE-Verzeichnis ist in EtherCAT Geräten angesiedelt, die meist mehrere funktional gleichwertige Kanäle umfassen. z. B. hat eine 4 kanalige Analogeingangsklemme 0..10 V auch 4 logische Kanäle und damit 4 gleiche Sätze an Parameterdaten für die Kanäle. Um in den Dokumentationen nicht jeden Kanal auflisten zu müssen, wird gerne der Platzhalter "n" für die einzelnen Kanalnummern verwendet.

Im CoE-System sind für die Menge aller Parameter eines Kanals eigentlich immer 16 Indize mit jeweils 255 Subindizes ausreichend. Deshalb ist die kanalweise Ordnung in $16_{dez}/10_{hex}$-Schritten eingerichtet. Am Beispiel des Parameterbereichs 0x8000 sieht man dies deutlich:

- Kanal 0: Parameterbereich 0x8000:00 ... 0x800F:255
- Kanal 1: Parameterbereich 0x8010:00 ... 0x801F:255
- Kanal 2: Parameterbereich 0x8020:00 ... 0x802F:255
- tbc...

Allgemein wird dies geschrieben als 0x80n0.

CoE-Verzeichnis - Wertänderungen

Einige, insbesondere die vorgesehenen Einstellungsparameter des Slaves sind durch den Anwender von der Feldbusseite aus veränderlich und beschreibbar. Dies kann schreibend/lesend geschehen

- über den System Manager (Abb. *Manuelles Einfügen eines StarUp-Eintrages*) durch Anklicken durch den Bediener
  Die Werte werden dann direkt im online verbundenen Slave geändert.
  Dies bietet sich bei der Inbetriebnahme der Anlage/Slaves an. Klicken Sie auf die entsprechende Zeile des zu parametrierenden Indizes und geben sie einen entsprechenden Wert im "SetValue"-Dialog ein.

- aus der Steuerung/PLC über ADS z. B. durch die Bausteine aus der TcEtherCAT.lib Bibliothek
  Dies wird für Änderungen während der Anlagenlaufzeit empfohlen oder wenn kein System Manager bzw. Bedienpersonal zur Verfügung steht.
- während des EtherCAT-Starts durch vordefinierte Befehle, die sog. StartUp-Liste.
  Meist wird die TwinCAT Konfiguration im Vorfeld ohne tatsächlich vorhandene EtherCAT-Slaves erstellt. Dann sollen bereits vor der Inbetriebnahme offline bekannte Eigenschaften wie Filtereinstellungen vorgenommen werden können, um die Inbetriebnahme zu beschleunigen.

CoE-Verzeichnis - StartUp-Liste

StartUp-Liste

Veränderungen im lokalen CoE-Verzeichnis des EtherCAT Slaves gehen im Austauschfall mit dem alten Gerät verloren. Wird im Austauschfall ein neues Gerät mit Werkseinstellungen ab Lager Beckhoff eingesetzt, bringt diese die Standardeinstellungen mit. Es ist deshalb empfehlenswert, alle Veränderungen im CoE-Verzeichnis eines EtherCAT Slave in der Startup List des Slaves zu verankern, die bei jedem Start des EtherCAT Feldbus abgearbeitet wird. So wird auch ein im Austauschfall ein neuer EtherCAT Slave automatisch mit den Vorgaben des Anwenders parametriert.

Wenn EtherCAT Slaves verwendet werden, die lokal CoE-Wert nicht dauerhaft speichern können, ist zwingend die StartUp-Liste zu verwenden.

Für diese Fälle kommt die StartUp-Liste zum Einsatz: die hier vorliegenden, vom Anwender eingetragenen Werte werden bei jedem EtherCAT Statusübergang/Start zum entsprechenden Slave gesendet. Ein StartUp-Eintrag besteht aus

- Zeitpunkt: in welchem Statusübergang wird das Kommando gesendet
  Meist ist PS (PREOP-->SAFEOP) die richtige Wahl, da dann ein EtherCAT Slave in den operativen Input-Betrieb schaltet.
- Index: Subindex
- Daten

Die Reihenfolge der Einträge wird nicht berücksichtigt: alle Einträge, für die ein Statusübergang zutrifft, werden gleichzeitig als asynchrone Kommandos an das EtherCAT System übergeben und dort ausgeführt, sobald es die Buslast zulässt.

Eine Überprüfung, ob ein Eintrag schon gleichlautend im Slave vorliegt, findet nicht statt.

Beispiel

Im Folgenden wird in die StartUp-Liste einer EL3152 die Anwenderskalierung aktiviert. Bereits vorhandene, zum Betrieb benötigte Einträge in der Liste sind grau hinterlegt.

Beim Rechtsklick auf die Fläche erscheint der Dialog:

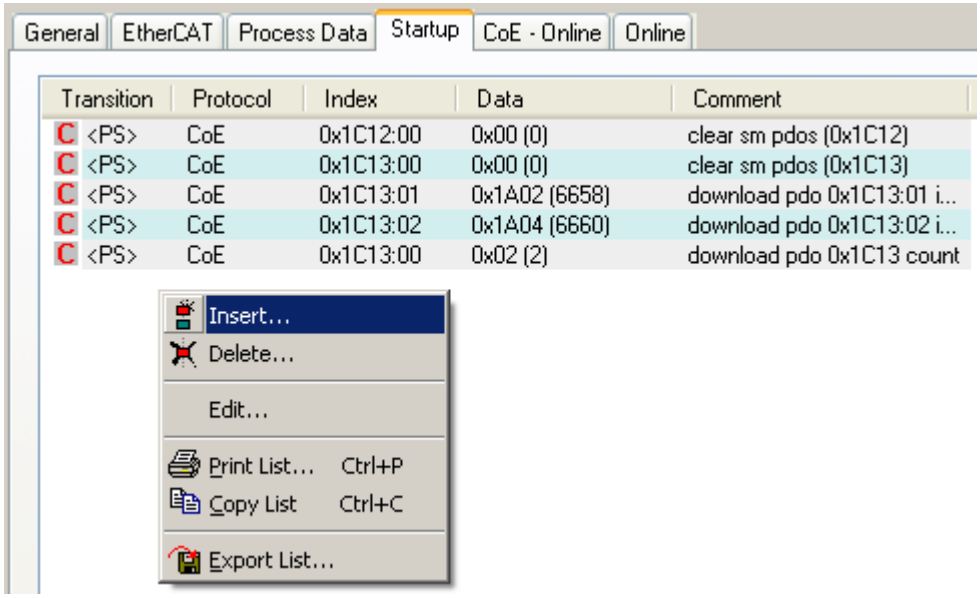


Abb. 11: Manuelles Einfügen eines StartUp-Eintrages

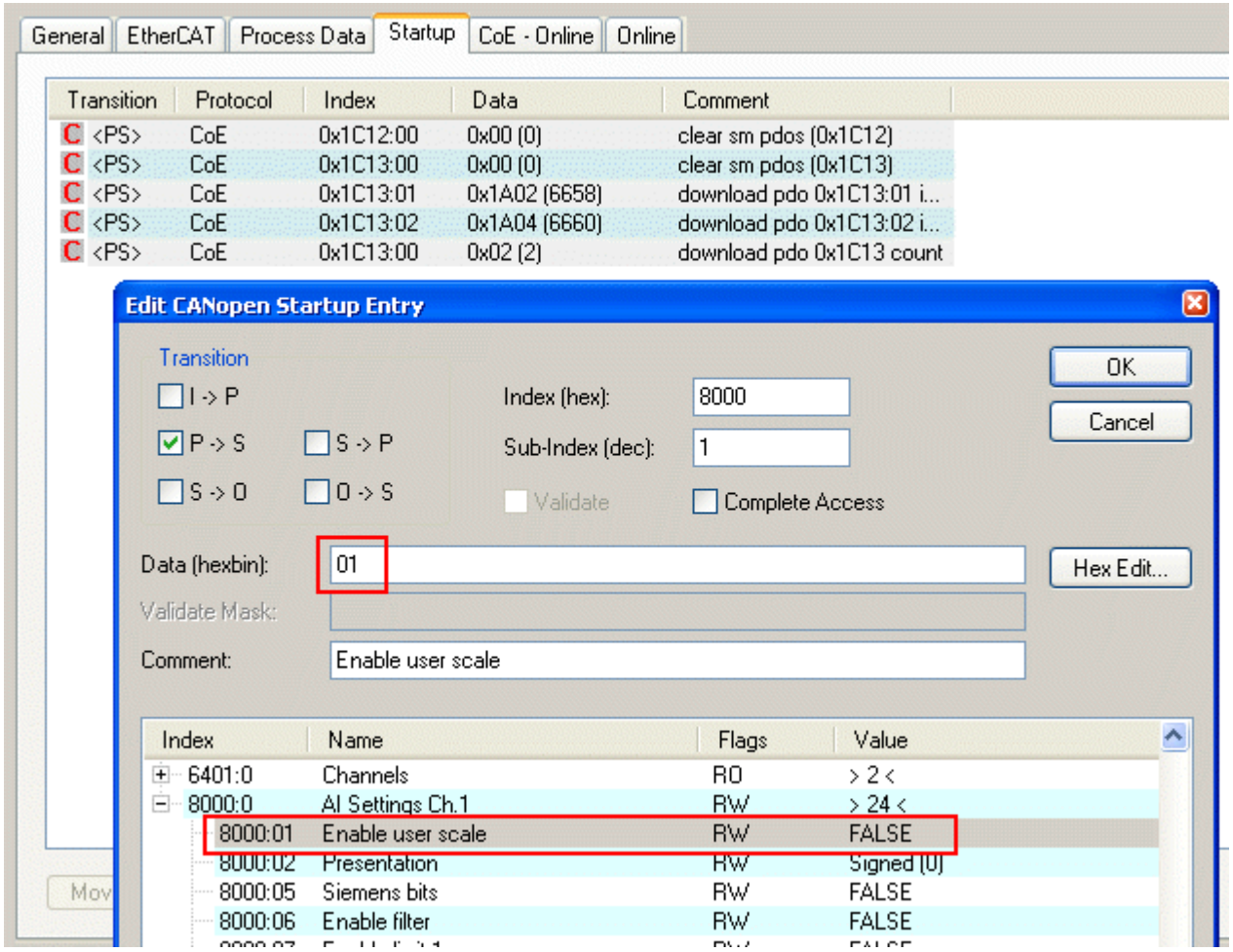


Abb. 12: Editieren

Beim Klick auf den Eintrag 0x8000:01 werden die entsprechenden Werte übernommen, bei *Data* wird 01 als gewünschter Wert eingetragen. Der Eintrag "P->S" kennzeichnet den Zeitpunkt der Ausführung.

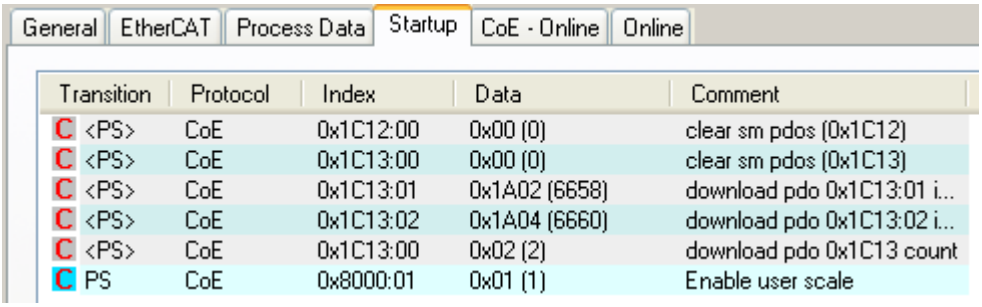


Abb. 13: Geänderte StartUp-Liste

Vom Anwender angelegte Einträge sind hellblau hinterlegt.

CoE-Verzeichnis - Datenerhaltung

Datenerhaltung

Werden online auf dem Slave CoE-Parameter geändert, wird dies in Beckhoff-Geräten ausfallsicher im Gerät (EEPROM) gespeichert. D.h. nach einem Neustart sind die veränderten CoE-Parameter immer noch erhalten.
Andere Hersteller können dies anders handhaben.
Wenn EtherCAT Slaves verwendet werden, die lokal CoE-Wert nicht dauerhaft speichern können, ist zwingend die StartUp-Liste zu verwenden.

Zusammenfassung der Eigenschaften

- Nicht jedes EtherCAT-Gerät muss über ein CoE-Verzeichnis verfügen
- Wenn ein CoE-Verzeichnis vorhanden ist, wird es im Gerät vom Controller verwaltet, zur Abfrage und Beschreibung aufbereitet und gespeichert.
- Zur Ansicht/Abfrage/Änderung kann der EtherCAT-Master verwendet werden, oder eine lokale Bedienoberfläche am Gerät (Tastenfeld, Bildschirm) erlaubt den Zugriff.
- Geänderte Einstellungen werden in Beckhoff Geräten stromausfallsicher gespeichert.
  Wenn das Gerät später getauscht wird, gehen allerdings die vom Serienstand geänderten Einstellungen verloren. Der EtherCAT-Master kann dann über die StartUp-Liste in das neue Gerät die geänderten CoE-Parameter beim Start laden, wenn er entsprechend eingestellt ist.
- Damit bei der Konfigurationsvorbereitung offline ein CoE-Verzeichnis zur Verfügung steht, kann es als Kopie in der Gerätebeschreibung enthalten sein.
- In welchem Umfang das CoE-Verzeichnis unterstützt wird, hängt von den Fähigkeiten des EtherCAT-Masters ab.

2.2.2.2 Beispielprogramm R/W CoE

Programmbeschreibung/ Funktion:

Nach Start dieses Programmbeispiels kann durch das Setzen von TRUE der Variablen *startRead* oder *startWrite* ein Schreib- oder Lesezugriff auf das CoE Verzeichnis eines bestimmten EtherCAT-Teilnehmers erfolgen.

Hinweise:

- ein EtherCAT-Slave an einem EtherCAT Master mit CoE Verzeichnis ist vorhanden; das Beispielprogramm ist initial für die Verwendung einer Klemme EL3751 konfiguriert;
  Aufbau: z.B. IPC/CX + (EK1100) + EL3751 + EL9011
- die AmsNetId des EtherCAT Master ist bekannt (diese ist vor Programmstart in den Code einzutragen und ist i.d.R. unter dem Karteireiter EtherCAT des ETherCAT-Masters zu finden)
- die Adresse des EtherCAT Slave ist ggf. in der Variablendeklaration von *userSlaveAddr* anzupassen (i.d.R. die Klemme, dessen CoE Verzeichnis angesprochen werden soll, z.B. 1002, 1007, etc.)

- je nachdem welcher EtherCAT Slave verwendet wird, sind entsprechend *nIndex* für die CoE-Objekt-ID sowie *nSubIndex* für den CoE-Objekt Sub-Index an den Stellen der Werteübergabe für das Lesen und das Schreiben einzutragen. Die korrekte Datenlänge und Datentyp ist dabei ggf. ebenfalls anzupassen.
- dieses Beispielprogramm führt lediglich Zugriffe auf einen bestimmten Wert durch, der über einen (sub) index eines CoE objektes bestimmt ist. Sollen Zugriffe auf ein komplettes Objekt erfolgen, so sind die entsprechenden Funktionsbausteine *FB_EcCoeSdoReadEx* und *FB_EcCoeSdoWriteEx* zu verwenden; siehe ergänzende Dokumentation zur Bibliothek Tc2_EtherCAT unter:
  https://infosys.beckhoff.com/
  TwinCAT 3 → TE1000 XAE → PLC → Bibliotheken → TwinCAT 3 PLC Lib: Tc2_EtherCAT → CoE Interface

Dieses Beispiel schreibt in das CoE Objekt 0x8000 mit dem Sub-Index 0x16 den Wert 22 und aktiviert damit den Filter 1 mit „IIR Butterw. LP 5th Ord. 1000 Hz“ bzw. liest den internen Temperaturwert der Klemme EL3751 aus dem CoE Objekt 0x9000, Sub-Index 0x01. Die Programmvariable für das Schreiben hat den Wert bei der Variablendeklaration bereits zugewiesen. Zur Kontrolle kann das Schreiben durch Einsehen des CoE-Verzeichnisses und das Lesen kann zur Laufzeit durch Betrachtung der Variable für das Lesen (*int16Buffer*) geprüft werden.

Download: https://infosys.beckhoff.com/content/1031/ethercatsystem/Resources/zip/5299954699.zip

Vorbereitungen zum Starten des Beispielprogramms (tnzip-Datei/TwinCAT 3)

- Nach Klick auf den Download-Button speichern Sie das Zip-Archiv lokal auf ihrer Festplatte und entpacken die *.tnzip-Archivdatei in einem temporären Ordner.

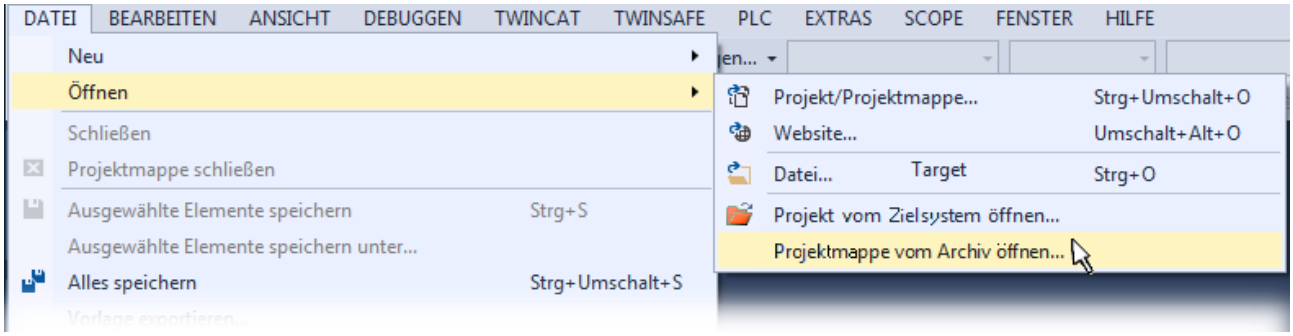


Abb. 14: Öffnen des *. tnzip-Archives

- Wählen Sie die zuvor entpackte .tnzip-Datei (Beispielprogramm) aus.
- Ein weiteres Auswahlfenster öffnet sich: wählen nun Sie das Zielverzeichnis, wo das Projekt gespeichert werden soll.
- Die generelle Vorgehensweise für die Inbetriebnahme der PLC bzw. dem Start des Programms kann u. a. den Klemmen-Dokumentationen oder der EtherCAT-Systemdokumentation entnommen werden.

Deklaration (ST)

```
PROGRAM MAIN
VAR
   fb_coe_write          : FB_EcCoESdoWrite; // Function Block for writing in CoE
   fb_coe_read           : FB_EcCoESdoRead;  // Function Block for reading from CoE
   userNetId             : T_AmsNetId := 'a.b.c.d.4.1'; // Have to be entered
   userSlaveAddr         : UINT := 1002;     // Have to be entered
   startWrite            : BOOL := FALSE;    // Sign for start writing
   startRead             : BOOL := FALSE;    // Sign for start reading
   nState                : BYTE := 0;        // RW-status
   // Example: Read EL3751 PAI Internal Data: Temperature Value
   int16Buffer           : INT;              // Buffer for reading
   // Example: Select EL3751 Filter1: 22 = IIR Butterw. LP 5th Ord. 1000 Hz:
   uint16Buffer          : UINT:=22;         // Buffer for writing
   bTxPDOState AT%I*     : BOOL;             // (PDO for synchronization)
END_VAR
```

Implementierung (ST):

```
CASE nState OF
0:
   IF startWrite THEN
      // Prepare CoE-Access: Write value of CoE object/ sub index:
      fb_coe_write(bExecute := FALSE);
      nState := 1;// Next state for writing
      startWrite := FALSE;
   END_IF
   IF startRead THEN
      // Prepare CoE-Access: Read value of CoE object/ sub index:
      fb_coe_read(bExecute := FALSE);
      nState := 11;// Next state for reading
      startRead := FALSE;
   END_IF
// ============== COE WRITE ================
1:
   // Write entry
   fb_coe_write(
   sNetId:= userNetId,
   nSlaveAddr:= userSlaveAddr,
   nSubIndex:= 16#16,
   nIndex:= 16#8000,
   pSrcBuf:= ADR(uint16Buffer),
   cbBufLen:= 2,
   bExecute:= TRUE,
   tTimeout:= T#1S
   );
   nState := nState + 1; // Next state
2:
   fb_coe_write();       // Execute CoE write until done
   IF fb_coe_write.bError THEN
      nState := 100;     // Error case
   ELSE
      IF NOT fb_coe_write.bBusy THEN
         nState := 0;    // Done
      END_IF
   END_IF
// ============== COE READ =================
11:
   // Read entry
   fb_coe_read(
   sNetId:= userNetId,
   nSlaveAddr:= userSlaveAddr,
   nSubIndex:= 1,
   nIndex:= 16#9000,
   pDstBuf:= ADR(int16Buffer),
   cbBufLen:= 2,
   bExecute:= TRUE,
   tTimeout:= T#1S
   );

   nState := nState + 1; // Next state
12:
   fb_coe_read();        // Execute CoE read until done
   IF fb_coe_read.bError THEN
      nState := 100;     // Error case
   ELSE
      IF NOT fb_coe_read.bBusy THEN
         nState := 0;    // Done
      END_IF
   END_IF
   100:
   ;                     // Error handling..
END_CASE
```

2.2.2.3 CoE-Reset, Wiederherstellen der Default-Werte

Um den Auslieferungszustand der veränderbaren CoE-Objekte bei den ELxxxx-Klemmen wiederherzustellen, kann das CoE-Objekt "Restore default parameters", Subindex 001 (wenn vorhanden) verwendet werden (s. Abb. *Auswahl des PDO „Restore default parameters“*).

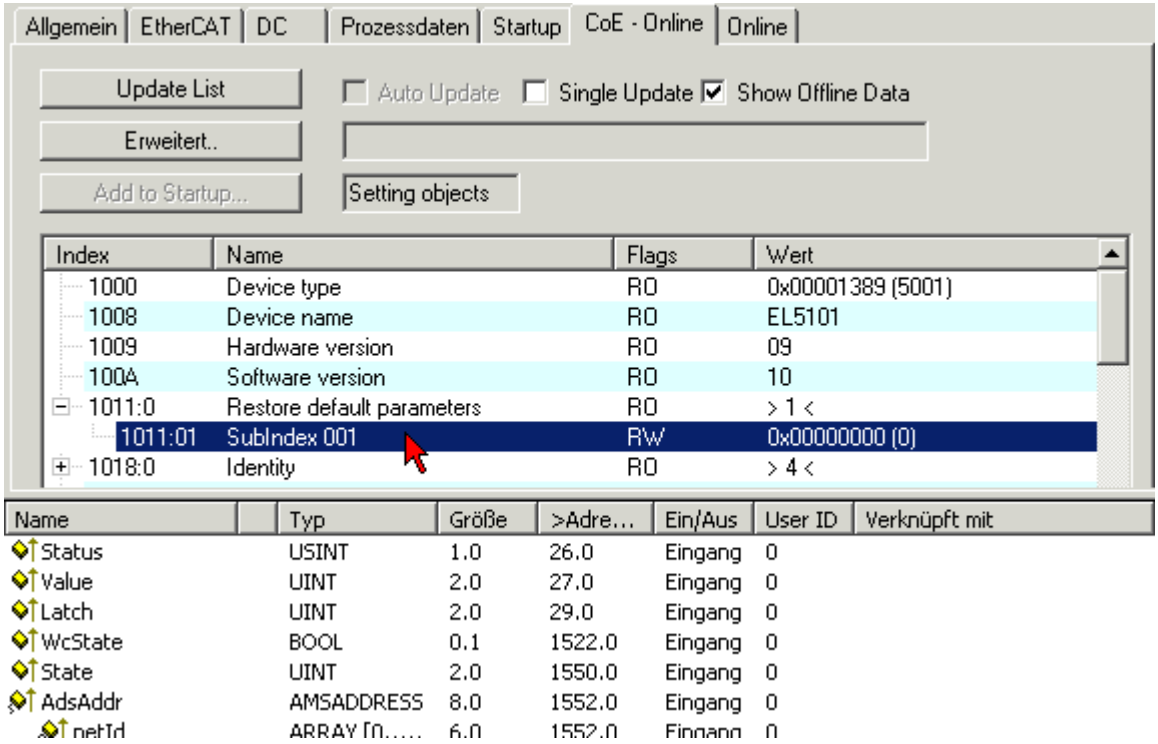


Abb. 15: Auswahl des PDO „Restore default parameters“

Durch Beschreiben diese Indizes mit dem Reset-Word werden im Slave alle veränderbaren Einträge auf die Default-Werte zurückgesetzt.

- Die Werte können nur erfolgreich zurückgesetzt werden, wenn das Reset auf das Online-CoE, d.h. auf dem Slave direkt angewendet wird. Im Offline-CoE können keine Werte verändert werden.
- TwinCAT muss dazu im Zustand RUN oder CONFIG/Freerun befinden, ein fehlerfreier EtherCAT-Verkehr muss stattfinden.
- Zur Kontrolle kann zuvor ein veränderbares Objekt manipuliert werden; es findet keine gesonderte Bestätigung statt.
- Dieser Reset-Vorgang kann auch als erster Eintrag in die StartUp-Liste des Slaves mit aufgenommen werden, z. B. im Statusübergang PREOP->SAFEOP oder, wie in Abb. *CoE-Reset als StartUp-Eintrag*, bei SAFEOP->OP.
  Damit werden etwaige Einstellungen im Slave zurückgesetzt.

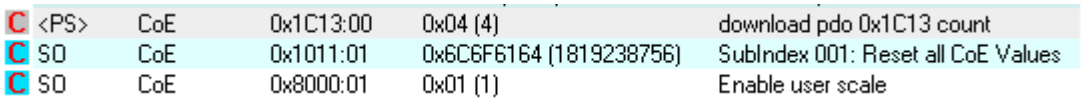


Abb. 16: CoE-Reset als StartUp-Eintrag

- Der EtherCAT Slave muss sich dazu mindestens im Status PREOP befinden
- Je nach Firmware und Slave lautet das Reset-Word
  - **64616F6C "load"**$_{hex}$ (i. d R. bei FW seit ca. 2008) oder
  - 6C6F6164$_{hex}$ "daol" (bei früheren FW-Versionen)
- Eine falsche Eingabe des Restore-Wertes zeigt keine Wirkung.

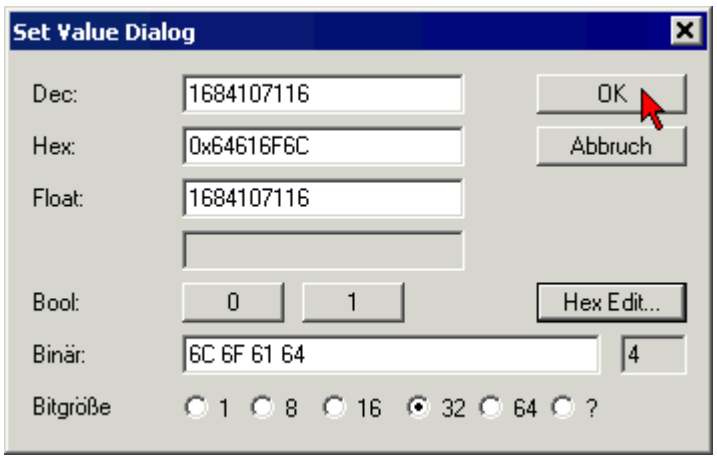


Abb. 17: Eingabe des Restore-Wertes im Set Value Dialog

2.3 System Features

2.3.1 EtherCAT Kabelredundanz

2.3.1.1 Prinzip

Die Beckhoff TwinCAT Kabelredundanz ist für die Kompensation beim Ausfall einer Kommunikations-Kabelstrecke im EtherCAT System konzipiert. Dazu wird eine Ringtopologie verwendet, die standardmäßig in beiden Richtungen betrieben wird. Wird der Ring an einer Stelle unterbrochen, werden dennoch beide Zweige erreicht.

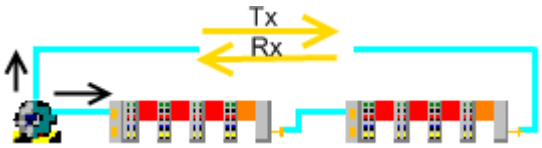


Abb. 18: Kabelredundanz-Prinzip

Am EtherCAT Master IPC wird ein zweiter Netzwerkport für den Ringschluss verwendet. Sowohl zyklische als auch azyklische Frames werden gleichzeitig über beide Ports versendet und durchlaufen das System.

- Vom primären Port aus werden im störungsfreien Betrieb alle EtherCAT Slaves im Vorwärtsdurchlauf erreicht - sie werden also bearbeitet, da der EtherCAT Slave Controller (ESC) nur im Vorwärtsbetrieb durchlaufen wird.
- Vom sekundären Port aus werden im störungsfreien Betrieb alle EtherCAT Slaves im Rückwärtsdurchlauf erreicht - die Daten im "Redundanz" Frame werden also nicht verändert.

Am jeweils anderen Port kommen die ggf. veränderten EtherCAT-Frames an und werden vom EtherCAT Master wieder kombiniert. Tritt der Redundanzfall durch Kabelunterbrechung ein, ist es dann unerheblich ob ein EtherCAT Slave über den Primär- oder den Redundanzport erreicht wird.

Wenn durch den Redundanzfall (beschädigtes Kabel, beschädigter Stecker, elektromagnetische Störung) nicht zufällig beide Ethernet-Frames direkt getroffen werden, erfolgt der Redundanzbetrieb ruckfrei ohne verlorene Daten.

Das Supplement ist 1-Fehler-tolerant, ermöglicht also die unbeeinträchtigte Kommunikation zu den Slaves, wenn an *einer* Stelle eine Kabelunterbrechung vorliegt. Bei Wiederherstellen der Kommunikation wird auch die originäre Kommunikationsrichtung wiederhergestellt. Wird die Kommunikation an mehr als einer Stelle unterbrochen, müssen erst alle Verbindungen wiederhergestellt werden, bevor ein erneuter Fehler auftreten darf.

Auch ein Systemstart unter Redundanzbedingungen ist zulässig.

Prinzipbedingt ist eine geschlossene Ringtopologie am besten für den Kabelredundanzbetrieb geeignet. Je nach Betriebsbedingungen können auch andere als der letzte Anschlusspunkt für die Redundanzverbindung zur Steuerung verwendet werden, s. Abb. *Sonderlösung Kabelredundanz*.

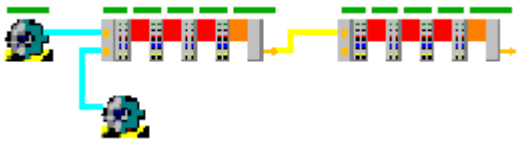


Abb. 19: Sonderlösung Kabelredundanz

Auf Grund des Synchronisierungsmechanismus für Distributed Clocks ist für eine Kombination von Kabelredundanz mit Distributed Clocks Slaves der PortMultiplier CU2508 einzusetzen.

Ausfall eines EtherCAT Slaves

Soll das Supplement „Kabelredundanz“ auch dazu verwendet werden, um bei Ausfall eines EtherCAT-Teilnehmers (z. B. Koppler EK1100 oder Box EPxxxx) die Kommunikation zu allen verbliebenen Teilnehmern weiterhin zu ermöglichen, ist Folgendes zu beachten:

- Es sind für die jeweilige ausfallträchtige IO-Gruppe SyncUnits anzulegen.
- Der Ausfall mehrerer EtherCAT-Slaves wird nicht abgedeckt. In Abhängigkeit von Ort, Anzahl und Reihenfolge der Ausfälle muss das EtherCAT-Feld dann ggf. komplett neu gestartet werden. Es empfiehlt sich, Master und Slave States aus der Applikation heraus zu bedienen und die entsprechenden Automatismen in den Einstellungen System Manager | Gerät EtherCAT | Erweiterte Einstellungen zu deaktivieren.

2.3.1.2 Bedingungen

Der Einsatz der Kabelredundanz unterliegt folgenden Bedingungen:

- Eine Supplement-Lizenz in entsprechendem Umfang wurde auf dem IPC (XP/CE) installiert. Zurzeit stehen Lizenzen für max. 250, 1000 und eine unbegrenzte Anzahl Slaves zur Verfügung.
  Das TwinCAT EtherCAT Redundancy Supplement wird auf der Beckhoff-Webseite für die Windows NT- und CE-Familie zum Download angeboten
- Plattformanhängigkeit/Betriebssystem auf dem Zielgerät
  - Windows NT-Familie (XP/XPe/7): das Supplement wird auf dem Zielgerät installiert. Der Start in den TwinCAT RUN-Modus bei aktivierter Kabelredundanz wird verhindert, wenn lokal keine Lizenz vorhanden ist.
  - Windows CE-Familie (CE, WES): das Supplement muss auf mindestens. einer NT-Plattform installiert werden. Danach liegt unter TwinCAT -> CE -> TwinCAT EtherCAT Redundancy die Installationsdatei ("TwinCAT_EtherCAT_Redundancy_CE.I586_250.cab" - Datei) vor. Diese muss auf das Embedded-Gerät kopiert und dort ausgeführt werden.
- Zur Kombination von Kabelredundanz mit dem HotConnect-Prinzip bitte die entsprechende Dokumentation beachten.
- TwinCAT 2.10 ab build 1313 oder TwinCAT ab Version 2.11 kommt zum Einsatz.
- Vom Anwender werden im System Manager im Bedarfsfall entsprechend der Konfiguration SyncUnits angelegt, s. Kapitel SyncUnits.
- Der Steuerungs-PC verfügt über 2 vollwertige, echtzeitfähige und ungeswitchte Ethernet-Ports. Embedded-Geräte (BXxxxx / CXxxxx) sind in der Regel nicht dazu geeignet.
- Eine Parallelverkabelung von einem EK1122 aus zum Zwecke der Kabelredundanz ist nicht zulässig.
- Das Supplement EtherCAT Kabelredundanz sichert ab
  - den Ausfall von Ethernet-Kabelstrecken (z. B. zwischen Kopplern EK1100), wenn die Spannungsversorgung der Stationen nicht unterbrochen wird.
  - den Ausfall des Kommunikationsteils einer einzelnen Klemme, wenn die Spannungsversorgung zu den nachfolgenden Klemmen nicht unterbrochen wird.

- Das Supplement *Kabelredundanz* ist 1-Fehler-tolerant. Tritt mehr als eine Störung in der Topologie auf, wird die vollständige IO-Kommunikation erst dann wiederhergestellt, wenn alle Fehler beseitigt sind. Unter Umständen ist ein manueller oder automatisierter Neustart der betroffenen Slaves nötig. Dies wird teilweise auch vom System Manager durchgeführt.
- Eine Kombination von Kabelredundanz mit Distributed Clocks ist nur unter Verwendung des Zusatzgerätes CU2508 möglich.

2.3.1.3 Einrichtung - Netzwerkeinstellung

Konfigurationserstellung

Die Erstellung einer Konfiguration durch Scannen in TwinCAT mit Redundanzpfad ist nicht möglich. Zum Scannen oder Anlegen der Slaves trennen sie deshalb die Verbindung zum Redundanzport.

Nach Installation des Redundanz-Supplements erfolgt die Einrichtung unter TwinCAT 2.11 wie folgt:

- Legen sie den primären Port manuell an oder lassen ihn per Scan finden.

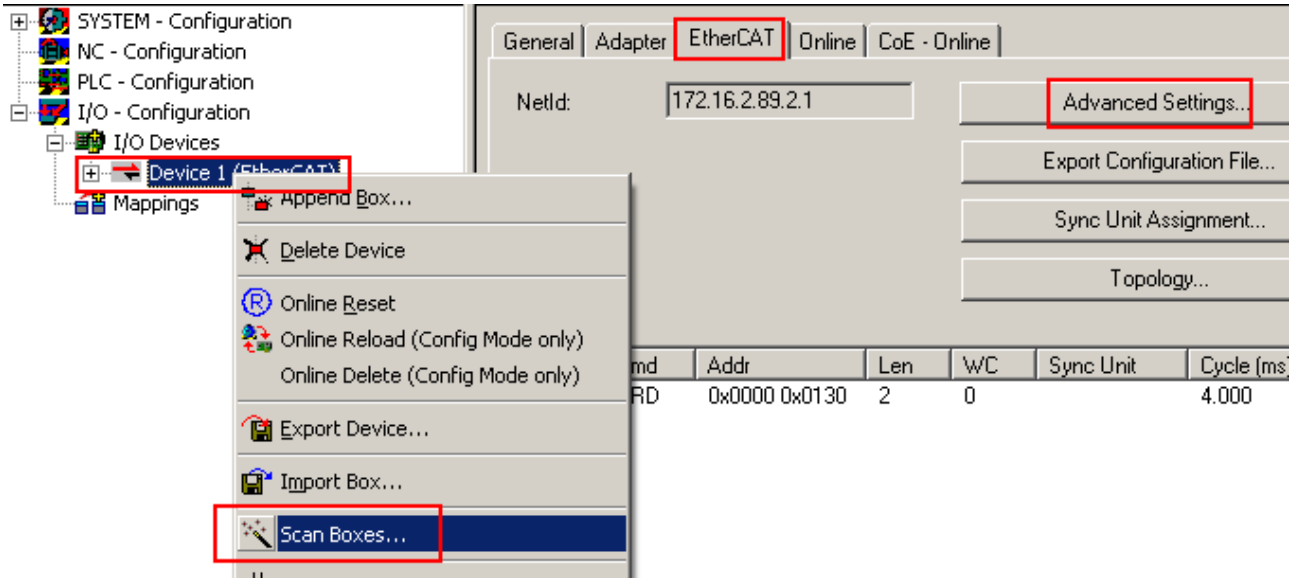


Abb. 20: Device einrichten

- Konfigurieren Sie die Redundanzeigenschaften des EtherCAT-Systems in den erweiterten Einstellungen. Geben Sie dabei den Punkt an, von dem aus die Verbindung zum Redundanzport erstellt wird - zur Auswahl stehen dabei auch Einsprungpunkte, die nicht am Ende einer Linientopologie liegen.

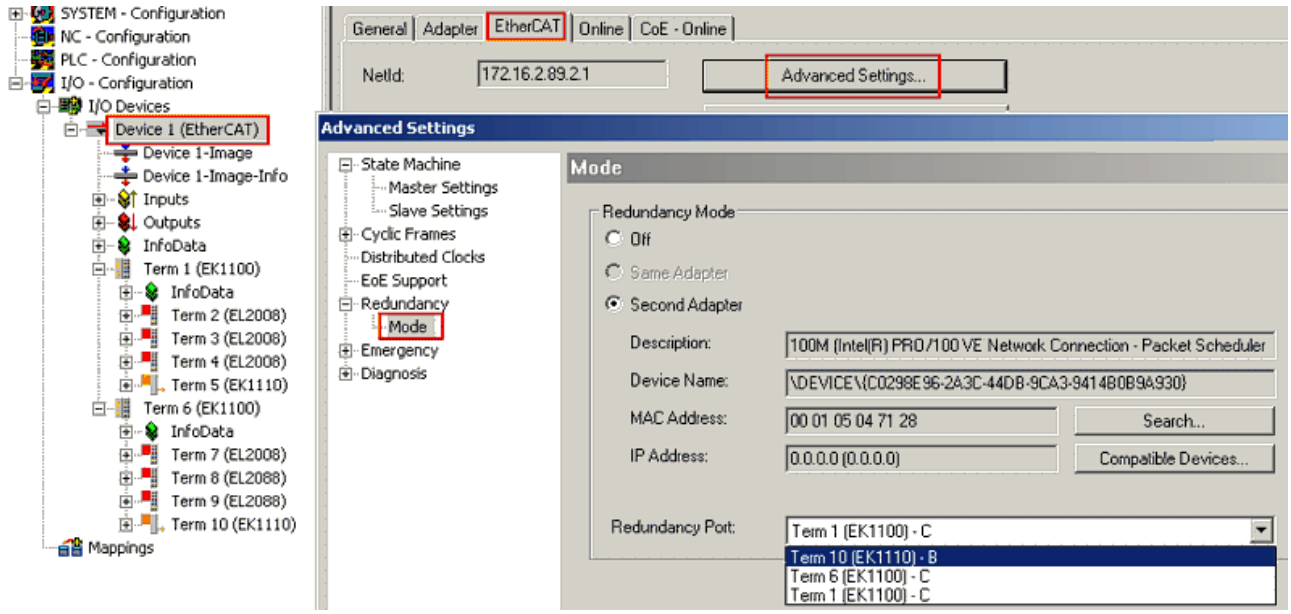


Abb. 21: Redundanz parametrieren

Nach der Aktivierung der Konfiguration ist die Kabelredundanz aktiv.

2.3.1.4 Einrichtung - SyncUnits

SyncUnits müssen nur angelegt werden, wenn mit dem Ausfall eines EtherCAT Slaves gerechnet wird.

Zyklische Daten

Mit der Einrichtung von SyncUnits kann nur die Kommunikation durch die zyklischen Daten beeinflusst werden, die azyklische Kommunikation z. B. per Mailbox wird immer vom System Manager verwaltet.

Standardmäßig versucht der System Manager alle IO-Daten in so wenig wie möglich Ethernet/EtherCAT-Frames zu integrieren. D.h. die maximale Framegröße von 1518 Byte je Ethernet Frame bzw. 15 Datagrammen wird möglichst ausgeschöpft. Dadurch wird eine effiziente und optimierte, den Feldbus wenig belastende Kommunikation erreicht.
Ein Datagramm ist Telegramm mit explizit einem Lese/Schreibauftrag. Ein Datagramm kann z. B. ein Lesebefehl über 2 Byte von einer analogen Eingangsklemme sein oder ein Schreibbefehl von 400 Byte Daten an 10 Servoantriebe.

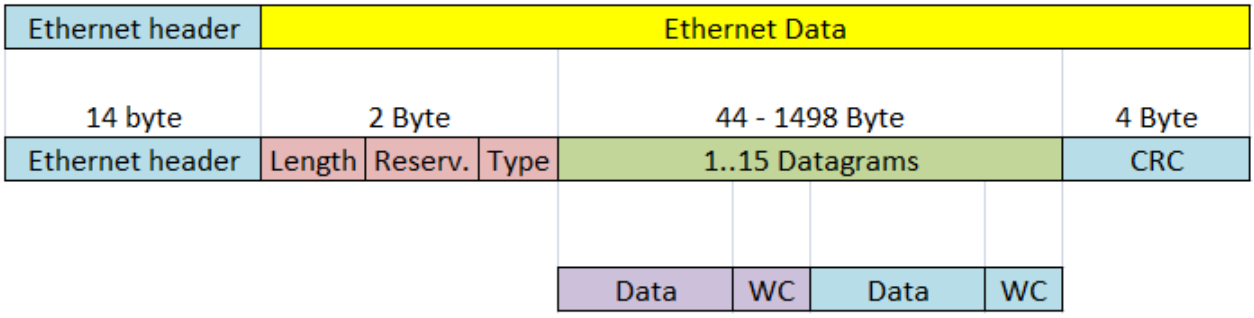


Abb. 22: Aufbau Ethernet Frame mit EtherCAT Protokolldaten

In Abb. *Aufbau Ethernet Frame mit EtherCAT Protokolldaten* wir dies verdeutlicht: Der Ethernet Frame (Zeile 1) besteht aus Header (blau), Daten (gelb) und der Checksumme CRC (blau). Wenn der Ethernet Frame EtherCAT-Protokolldaten trägt, setzen sich diese Daten wiederum zusammen aus einem EtherCAT Header (rot) und den 1 bis 15 Datagrammen (grün). Diese wiederum bestehen jeweils aus Header und Daten und einem WorkingCounter (WC).

Der Working Counter ist dabei für das Kabelredundanzprinzip entscheidend. Der Working Counter ist eine 16 bit Zahl, die mit dem Wert "0" vom EtherCAT Master abgeschickt wird. Jeder EtherCAT Slave, der von diesem Datagramm schreibend oder lesend angesprochen wird, erhöht diese Zahl um 1, 2 oder 3, je nach Art des Zugriffs. Wenn das Datagramm die gesamte Konfiguration durchlaufen hat, kommt der Working Counter also mit einem Wert größer 0 zum Master zurück. Der Master wiederum hat einen Erwartungswert, denn ihm ist ja bekannt, wie viele Slaves dieses Datagramm bearbeitet haben müssen. Wenn der WC des Datagramms nicht mit dem Erwartungswert übereinstimmt, hat einer der angesprochenen Slaves seinen Arbeitsauftrag nicht aufgeführt. Es obliegt dann dem Master im Weiteren durch besondere Maßnahmen herauszufinden, welcher Slave betroffen ist und ob ggf. das Datagramm wiederholt werden soll.

Im TwinCAT System Manager (Reiter EtherCAT des EtherCAT Gerätes) ist der der aktuellen Konfiguration entsprechende aktuelle Frameaufbau der zyklischen Daten einzusehen.

Hier ein Beispiel mit einer kleinen Konfiguration:

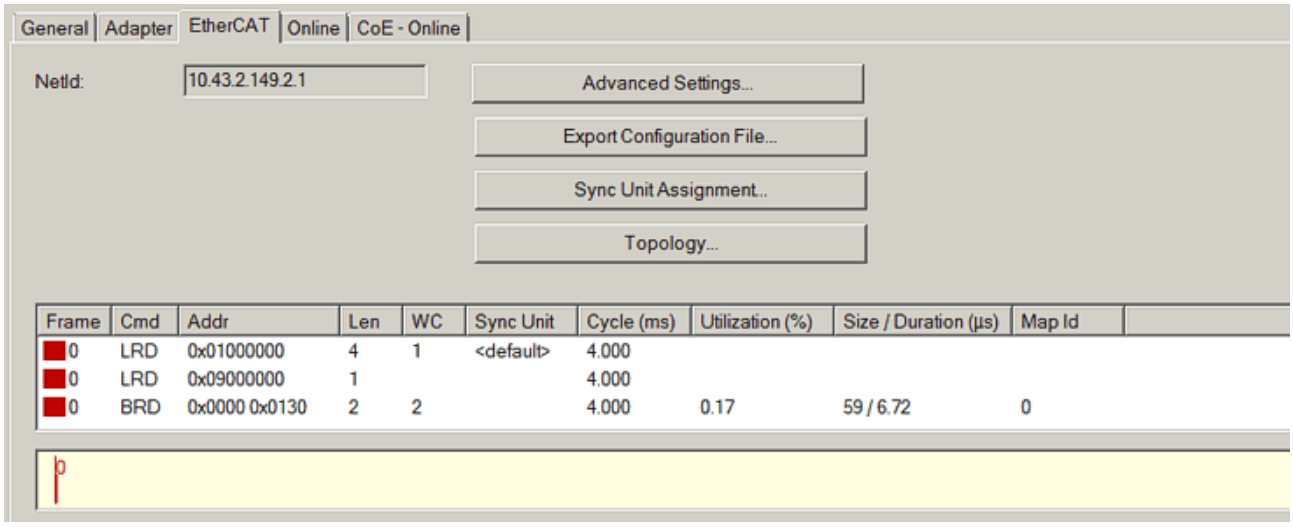


Abb. 23: TwinCAT Darstellung Frameaufbau - kleine Topologie

Für die zyklischen IO-Daten ist hier nur ein Ethernet Frame im Einsatz (rot), der 3 EtherCAT Datagramme trägt, und zwar 2x LRD (**L**ogical**R**ea**d**) und 1x BRD (**B**roadcast**R**ea**d**).
Der Frame ist bei 59 Byte Daten 6.72 µs lang und nutzt damit 0.17% der Zykluszeit von 4 ms.

Das erste Datagramm hat einen erwarteten Working Counter von 1, das BRD-Kommando von 2, siehe Spalte WC.

In einer deutlich größeren Konfiguration wird der Frameaufbau komplexer:

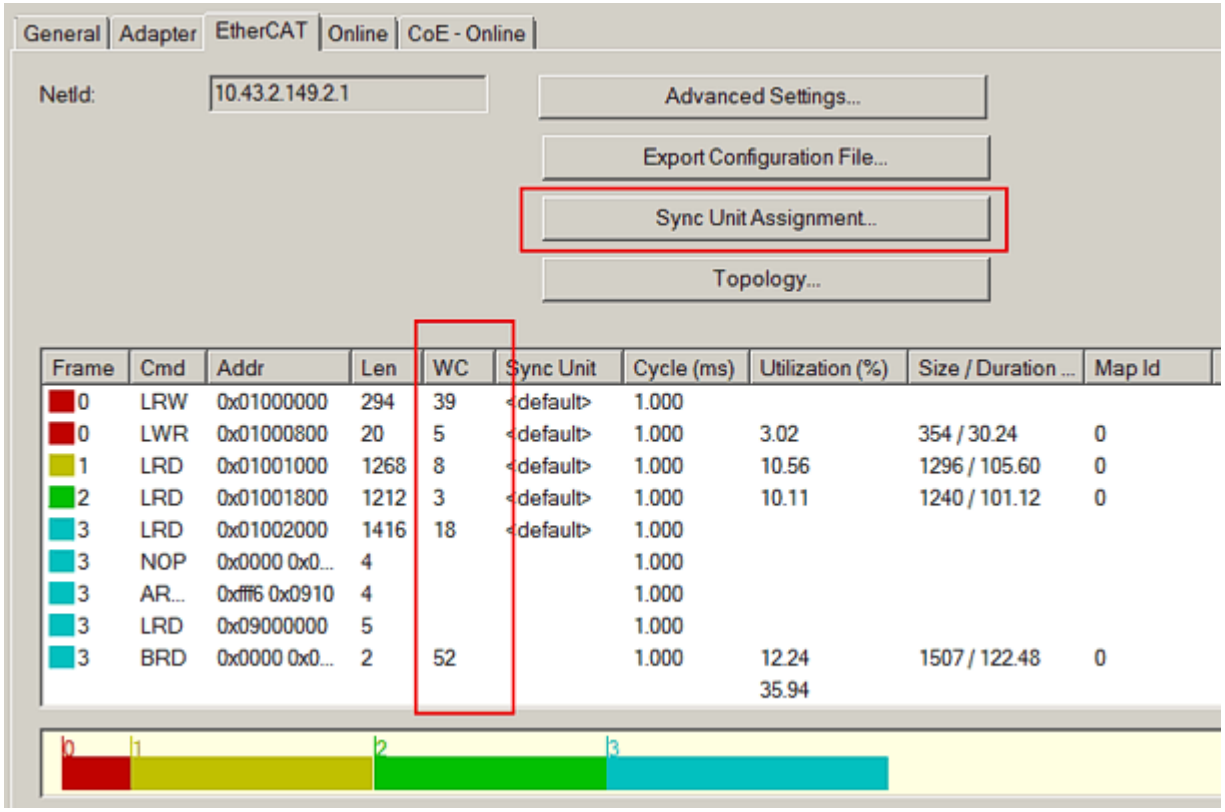


Abb. 24: TwinCAT Darstellung Frameaufbau - größere Topologie

Bei 1 ms Zykluszeit sind nun 3 Ethernet Frames unterwegs, die insgesamt 9 Datagramme tragen, mit jeweils erwarteten Working Countern von 3 bis 52. Diese Konfiguration nutzt im Übrigen bereits 35.94% der zur Verfügung stehenden Bandbreite von 100 MBit FastEthernet bei der Zykluszeit von 1 ms.

Anwendung auf die Kabelredundanz bei Slaveausfall

Wenn ein Datagramm mit einem "falschen" Working Counter zum Master zurückkommt, kann dieser im ersten Moment nicht feststellen, welcher Slave keine Daten lieferte - der Master muss also alle Eingangsdaten in diesem Datagramm für ungültig erklären. D.h. bei allen betroffenen Slaves geht die WC-Anzeige auf 1=*invalidData*.

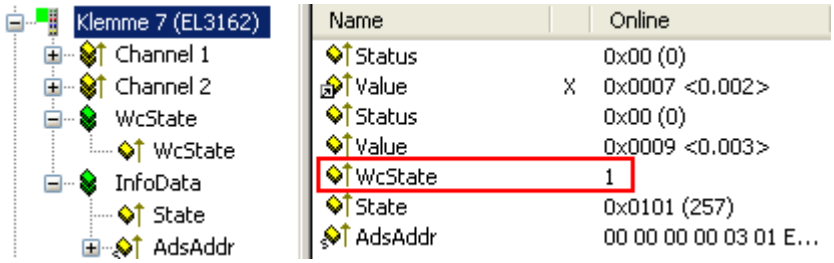


Abb. 25: WC = 1 zeigt invalidData an

Diese WC-Anzeige bildet der System Manager sofort beim Empfang aus den empfangenen Datagrammen für den jeweiligen Slave. In Abb. *WC = 1 zeigt invalidData an* ist dies für eine EL3162 dargestellt, die Eingangsdaten sind eingefroren, *WcState* = 1 und der *State* entsprechend der Bitbedeutungen von 0x0101 "Slave in INIT" und "Slave not present".

Bei Ausfall eines EtherCAT Slave, z. B. einer analogen Eingangsbox EP3174, würden somit auch andere Eingangsdaten verworfen, die durch den effizienten Frameaufbau im selben Datagramm liegen. Als Abhilfe kann anwenderseitig jeder Slave manuell einer sogenannten SyncUnit zugeordnet werden. Im Extremfall könnte jeder Slave eine eigene SyncUnit bekommen, mit den entsprechend nachteiligen Folgen für die Feldbuseffizienz bis hin zu hoher Busauslastung.

Der entsprechende Dialog ist über den Reiter EtherCAT zugänglich, s. Abb. *TwinCAT Darstellung Frameaufbau - größere Topologie*.

Ein Slave wird einer SyncUnit zugeordnet, s. Abb. *Eintragen der SyncUnits bei Default-Frameaufbau*

- durch Anklicken in der Spalte
- Eintragen des gewünschten Namens (beliebige Bezeichnung) bei SyncUnitNames

Wird eine Bezeichnung mehrmals vergeben, werden entsprechend auch die Slaves in einem Datagramm (logisch SyncUnit) betrieben.

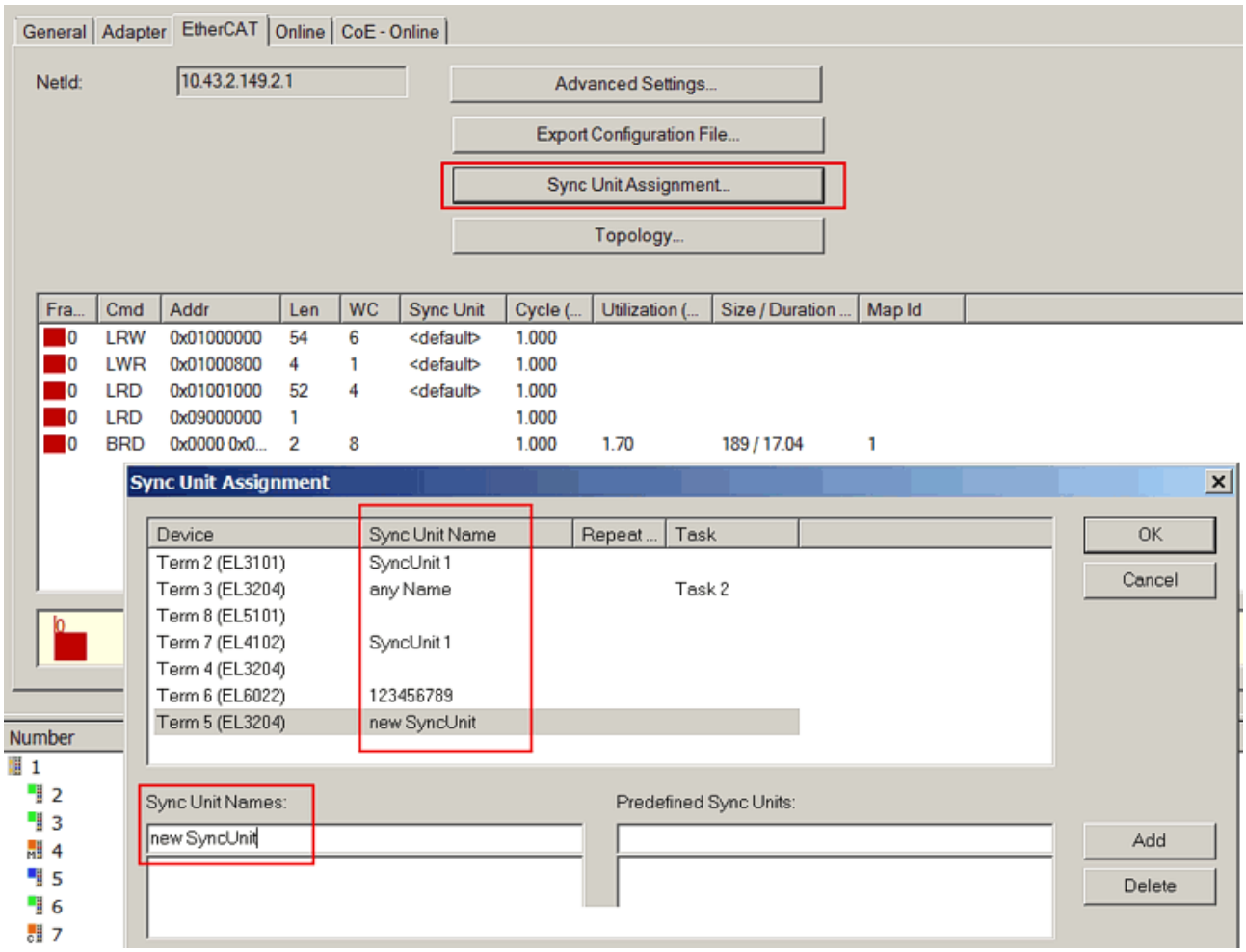


Abb. 26: Eintragen der SyncUnits bei Default-Frameaufbau

Dadurch ergibt sich ein neuer Aufbau der zyklischen Ethernet-Frames nach Abb. *Neuer Frameaufbau mit SyncUnits*:

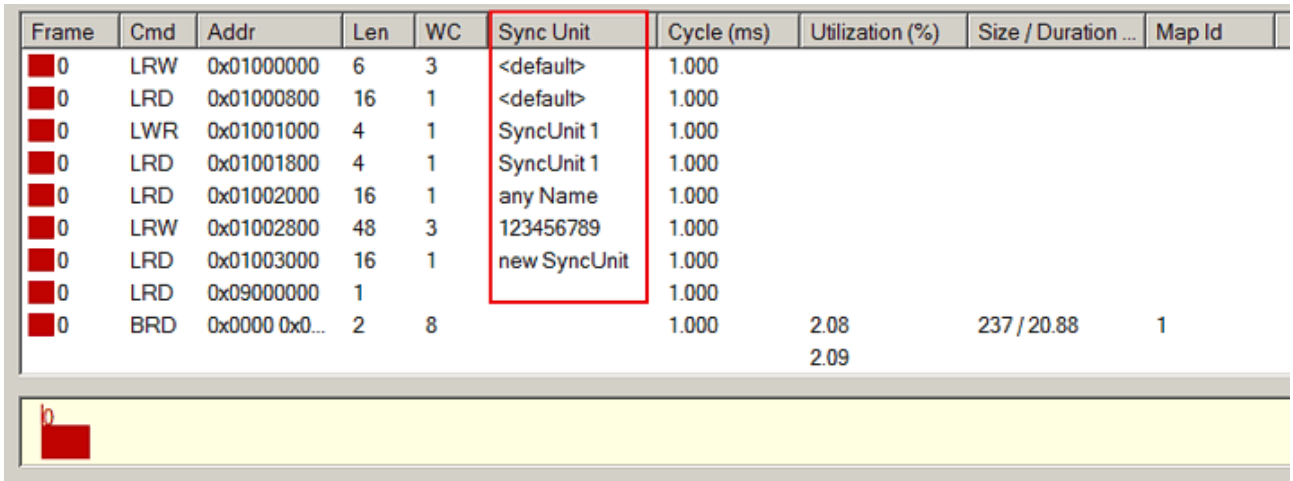


Abb. 27: Neuer Frameaufbau mit SyncUnits

Teilnehmer, die nicht manuell gesetzt werden, werden weiterhin vom System Manager automatisch in Datagrammen zusammengefasst und ggf. mit <default> gekennzeichnet.

Typischerweise werden Baugruppen als eine eigene SyncUnit definiert, die über Ethernet-Kabelverbindung kommunizieren:

- Koppler EK11xx inkl. den anhängende Klemmen EL/ES/EMxxxx
- EP-Boxen
- AX-Antriebe
- ...

Fällt jetzt eine solche Station aus der Kommunikation, sind nur die Daten dieser SyncUnit ungültig, alle anderen Stationen nehmen unbeeinflusst am Datenaustausch teil.

2.3.1.5 Systemverhalten

Das Handling des Redundanzfalls übernimmt TwinCAT im Echtzeitkontext, der Anwender hat keine Eingriffsmöglichkeit. Es stehen allerdings über Variablen im System Manager Information zum aktuellen Status bereit.

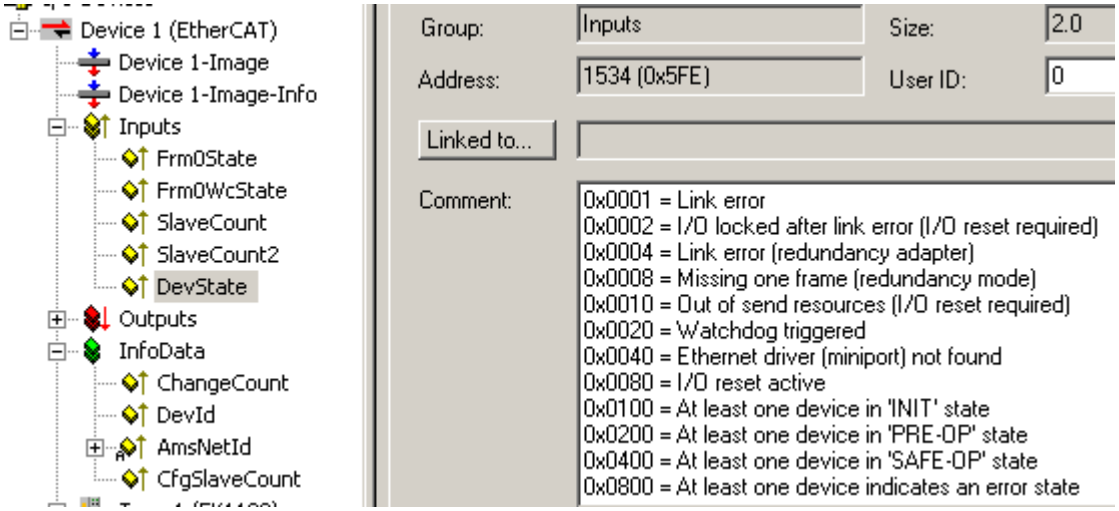


Abb. 28: Verfügbare Informationen

- SlaveCount/SlaveCount2
  Hier wird angezeigt, wie viele EtherCAT Slaves über den Primär- bzw. Redundanzport aktuell erreicht werden. Diese Information ist zyklusaktuell.
- FrmXWcState
  Diese Information ist zyklusaktuell. Es wird empfohlen, auch im Redundanzbetrieb diese Sammelinformation aller Working Counter eines EtherCAT Frames auf Applikationsebene zu überwachen.
- DevState
  Die Bit-Bedeutungen
  - 0x0001: Link Error am primären Port
  - 0x0004: Link Error am Redundanzport
  - 0x0008: Missing one frame
- beziehen sich auf die Kabelredundanz. Sind alle 3 Bits = 0, dann befindet sich die Kabelredundanz im Normalzustand und der nächste Verbindungsfehler kann auftreten.
  Die Link-Status Meldungen können ggf. einige Millisekunden verzögert gemeldet werden und sind deshalb nicht im Echtzeitkontext auswertbar.

In der Topologie Anzeige wird der Redundanzfall entsprechend abgebildet.

- Abb. *Typische Fehlerbilder im Redundanzfall*, oben: Unterbrechung am Redundanzport

- Abb. *Typische Fehlerbilder im Redundanzfall*, Mitte: 2 Unterbrechungen, Kommunikationsausfall zu einem Klemmenblock.
- Abb. *Typische Fehlerbilder im Redundanzfall*, unten: Klemme im 2. Klemmenblock im laufenden Betrieb entfernt, dadurch Spannungsunterbrechung zu nachfolgenden Klemmen und Kommunikationsausfall dorthin

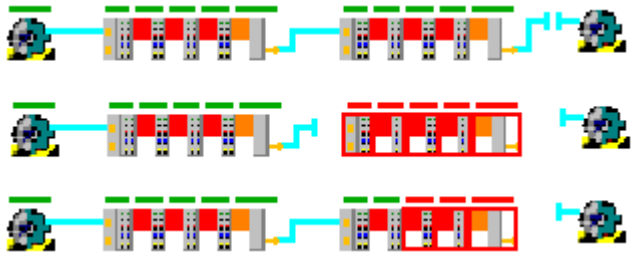


Abb. 29: Typische Fehlerbilder im Redundanzfall

Redundanzfall

Der Redundanzfall ist ein irregulärer Betriebszustand. Es wird empfohlen, durch Auswertung der o.a. Diagnosevariablen das Eintreten frühzeitig zu diagnostizieren und die Ursache umgehend abzustellen.

2.3.2 HotConnect

2.3.2.1 Das Prinzip

Die Beckhoff TwinCAT EtherCAT Hot-Connect Funktionalität erlaubt es, vor dem Start oder auch während des Betriebs der Anlage vorkonfigurierte Abschnitte aus dem Datenverkehr zu nehmen bzw. hinzuzufügen. Dies kann durch Trennen/Verbinden der Kommunikationsstrecke, An/Ausschalten des Teilnehmers oder sonstige Maßnahmen geschehen. Dies wird "flexible Topologie" oder Hot-Connect genannt.

Beispiel:
In einer modularen Fertigungsanlage werden Module mit integrierten EtherCAT-IO während des Betriebs kommunikationstechnisch angekoppelt oder getrennt.

Nach der Verbindungsherstellung zu einer Hot-Connect-Gruppe werden einige Maßnahmen zur Inbetriebnahme durchgeführt, die einige Sekunden Hochlaufzeit bewirken:

- Ethernet Linkaufbau
- Parametrierung der Geräte
- EtherCAT Hochlauf INIT -> OP
- ggf. Distributed Clocks Synchronisierung

Gerade in Anwendungen in denen oft Topologiewechsel vorgenommen werden wie z. B. Werkzeugwechslern ist ein beschleunigter Hochlauf von Vorteil. Dazu sind besondere Fast-Hot-Connect-Komponenten verfügbar, die eine Hochlaufzeit von bis zu < 1 Sekunde gewährleisten. Siehe dazu die Hinweise zum Fast-Hot-Connect [▶ 42].

Das Hot-Connect/Fast-Hot-Connect-Prinzip ist somit eine Erweiterung zur sonst allgemeingültigen Regel, dass die Reihenfolge/Anordnung der EtherCAT-Teilnehmer im Feld genau der angelegten Konfiguration entsprechen muss.

Zur Einrichtung wird keine gesonderte Lizenz sondern nur die dafür konzipierten EtherCAT-Geräte (Koppler und Abzweige) benötigt.

TwinCAT

Der Einsatz der Hot-Connect-Funktionalität in den hier beschriebenen Ausprägungen ist nur mit TwinCAT 2.11 ab build 1539 möglich. Insbesondere unterstützt TwinCAT 2.10 keine Kombination mit Distributed Clocks.

Eigenschaften und Systemverhalten

- Die Angaben in dieser Dokumentation gelten sowohl für XP- als auch CE-basierte TwinCAT 2.11-Systeme.
- Hot-Connect-Gruppe können Gruppen von Slaves (Koppler + Klemmen) oder auch einzelne Slaves (Antriebe, Klemmen, Sensoren, Positionsgeber) sein.
- Eine Hot-Connect-Gruppe kann zwar physikalisch mit dem EtherCAT-Netzwerk verbunden sein und empfängt und versendet die EtherCAT-Frames, sie nimmt aber aus Sicht der Steuerung erst am Prozessdatenverkehr teil, wenn ihr Adresse (s.u.) vom Master erkannt wurde.
- Die Identifizierung und Inbetriebnahme einer Hot-Connect-Gruppe durch den Master kann je nach Position bis zu mehreren Sekunden dauern.
  Beschleunigter Hochlauf ist mit Fast-Hot-Connect-Komponenten zu realisieren.
- Die Überwachung von WcState, Status und Link-Angaben der Slaves wird dringend empfohlen.
- Inbetriebnahme einer Hot-Connect-Gruppe durch den Master
  - erkennt der EtherCAT Master keine gültige Adresse in der Station, wird sie nicht in den Prozessdatenverkehr aufgenommen.
  - nachdem eine gültige Adresse erkannt wurde ("Hineinschalten" mit einem DIP-Switch, Ändern der SSA) wird die Station mit aufgenommen.
  - obere Punkte bleiben unverändert, bis die Station abgeschaltet oder vom Netzwerk getrennt wird.
- SyncUnit: Der System Manager legt automatisch eigene und untereinander getrennte SyncUnits für Hot-Connect-Gruppen an, damit sie eigene WorkingCounter erhalten und ihre Außerbetriebnahme nicht die Prozessdaten anderer Teilnehmer beeinflusst.
- Der erste Teilnehmer/Koppler nach dem Master sollte kein Hot-Connect-konfigurierter Teilnehmer sein, da dies die Linkerkennung am Master verlangsamt.
- Die Kombination mit der Eigenschaft "Kabelredundanz" ist teilweise möglich.
- Distributed-Clocks-fähige Slaves können verwendet werden.
  Bei Inbetriebnahme eines Distributed Clocks Slaves (DC-Slaves) wird dessen lokale Uhr initialisiert und dann fortlaufend mit dem bestehenden Netzwerk synchronisiert.
  Nach Verbindungsaufbau wird die Hot-Connect-Gruppe resynchronisiert, dieser Vorgang kann einige Sekunden dauern. Die betroffenen EtherCAT-Slaves werden solange im Zustand SAFEOP gehalten. Nach erfolgreicher Resynchronisierung werden sie in den Zustand OP geschaltet.

EL-Klemmen als Hot-Connect-Gruppe

Das Stecken/Ziehen von KL/KS/EL/ES-Klemmen unter Spannung ist nicht zulässig. Deshalb ist eine Konfiguration einzelner Klemmen bzw. Klemmenblöcke unterhalb eines Kopplers als variable Hot-Connect-Gruppe nicht sinnvoll. Kleinste Einheit auf Klemmenebene ist ein Koppler (EK/BK) bzw. eine EP-Box.

Topologien

Folgende Topologien wurden überprüft:

Typ 1: Stern

Besonders zweckmäßig für das Hot-Connect-Konzept sind Sterntopologien - die abzeigenden Gruppen werden als Hot-Connect-Gruppe definiert und sind im Betrieb an- und abkoppelbar.

Im Allgemeinen ist jede Topologie möglich, bei der ein EtherCAT-Netz an einer EtherCAT-Port hängt.

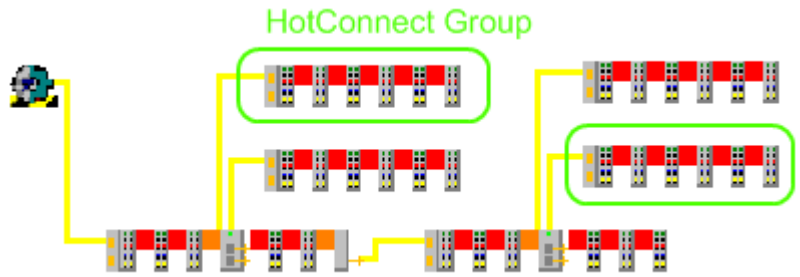


Abb. 30: Sterntopologie

Als Abzweigpunkte bieten sich an: EK110x, EK150x, EK1122, EP1122 und alle Slaves, die mehr als nur einen IN- und OUT-Port besitzen.

Eine Hot-Connect-Gruppe kann an jedem regulär freien Port in der Topologie angeschlossen werden.

Die Kombination von Kabelredundanz (kostenpflichtiges Supplement) im Hauptkreis und abzweigenden Hot-Connect-Gruppen an Stichverbindungen ist vorläufig nicht zulässig. Bei Verwendung von Kabelredundanz ist der Einsatz von DC-Slaves z.Z. noch nicht möglich

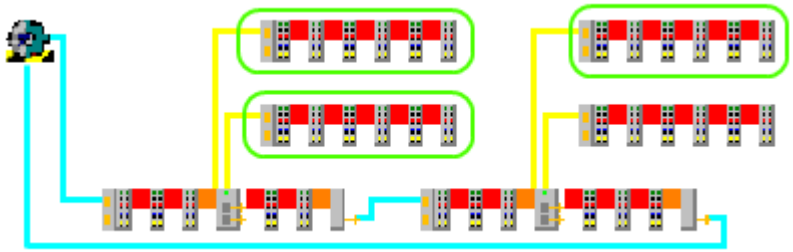


Abb. 31: Sterntopologie mit Kabelredundanz - z.Z. nicht zulässig

Typ 2: Linie

Bei der Verwendung der Linientopologie werden alle Teilnehmer nach einer Trennstelle bei Wiederherstellen der Verbindung neu initialisiert.

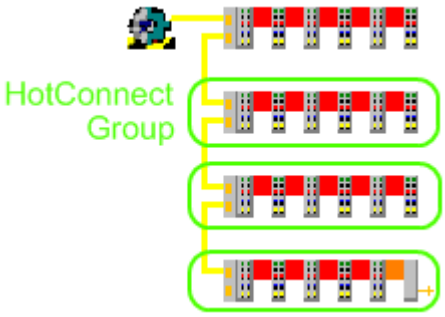


Abb. 32: Linientopologie

Werden in der Linientopologie HC-Gruppen und Nicht-HC-Gruppen gemischt, müssen alle Nicht-HC-Gruppen vor den HC-Gruppen und (wie bei EtherCAT sonst auch üblich) in der richtigen Reihenfolge angeordnet werden.

Hier im Bild ist eine Nicht-HC-Gruppe vor 3 HC-Gruppen angeordnet. Im Extremfall können alle Stationen HC-Gruppen sein.

Die Verwendung von Kabelredundanz (kostenpflichtiges Supplement) mit HC-Gruppen auf dem Redundanzpfad ist nicht möglich.

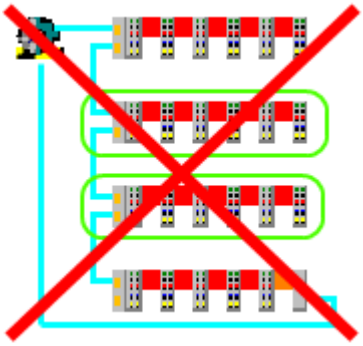


Abb. 33: Linientopologie mit Kabel-Redundanz nicht möglich

Allgemeine Hinweise

- **Stacked Groups**
  Wenn Hot-Connect-Gruppen physikalisch hintereinander betrieben werden ("stacked"), muss zuerst die übergeordnete Gruppe am Verkehr teilnehmen, bevor die unterlagerte in Betrieb genommen werden kann.

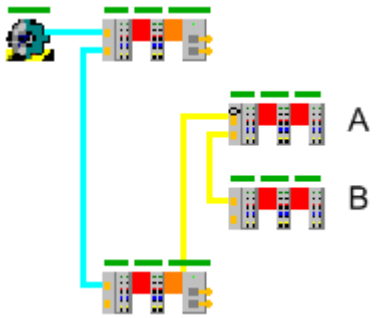


Abb. 34: Stacked groups

Beispiel: die Gruppen A und B mit je eigenen Adressen werden zusammen zugesteckt/angeschaltet. Gruppe B wird (trotz evtl. gültiger Adresse) erst dann in Betrieb genommen, wenn Gruppe A regulär vom Master in Betrieb genommen werden konnte.

- **Regulär freie Ports**
  die Hot-Connect-Gruppen können nur an in der Konfiguration *freien* Ethernet-Ports angeschlossen werden - wird eine Gruppe an einen irregulär freien Port angeschlossen, wird die Gruppe trotz evtl. gültiger Adresse nicht in Betrieb genommen.

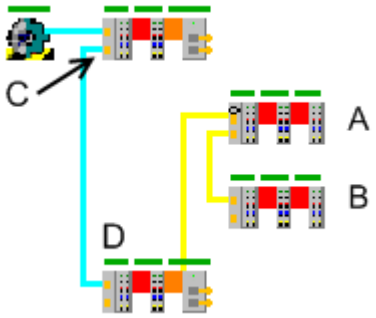


Abb. 35: Freie Ports

Beispiel: der Koppler D (keine Hot-Connect-Gruppe!) wird vom Port C getrennt. Damit ist der Port C irregulär frei. Eine an diesen Port konnektierte Hot-Connect-Gruppe A oder B wird vom Master nicht in Betrieb genommen.

2.3.2.2 Hinweise zur EtherCAT Fast-Hot-Connect Technologie

Mit EtherCAT-Komponenten, die Fast-Hot-Connect unterstützen, ist ein deutlich schnellerer Feldbus-Hochlauf nach Verbindungsherstellung möglich. Die Hochlaufzeit ist im Detail abhängig vom Umfang der Geräte, Topologie und aktivierten Distributed Clocks. Benötigt ein normaler Verbindungs- und Kommunikationsaufbau mehrere Sekunden, ist mit FHC-Komponenten < 1 Sekunde möglich.

Eigenschaften und Systemverhalten

- Fast-Hot-Connect wird ab TwinCAT 2.11R3 build 2221 unterstützt
- Fast-Hot-Connect-Ports sind besonders gekennzeichnet.

Abb. 36: Kennzeichnung FHC-Port am EK1122-0080 bzw. EK1101-0080

- an Fast-Hot-Connect-Ports dürfen keine Standard-EtherCAT-Geräte angeschlossen werden. Dies ist durch applikationsseitige Maßnahmen sicherzustellen, was durch die in derartigen Applikationen i. d. R. maschinell durchgeführten Topologiewechsel einfach umzusetzen ist.

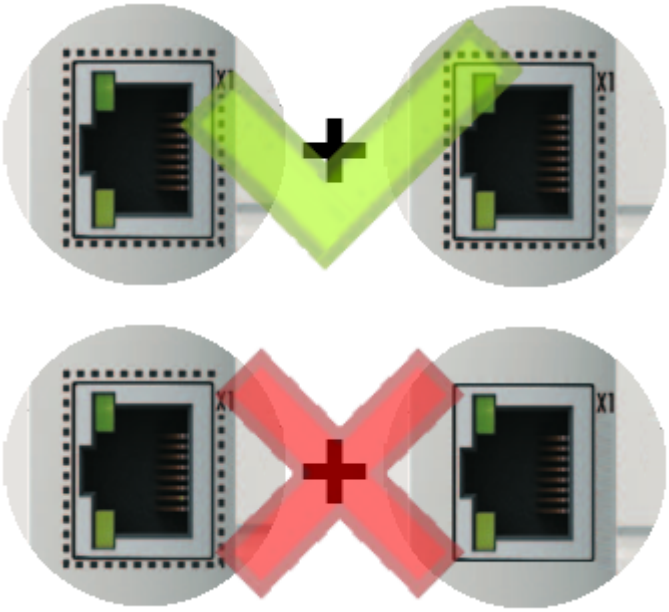


Abb. 37: Empfehlung Kombination Ethernet Ports

- Wurden dennoch entsprechende Ports verbunden, ist ggf. ein PowerReset der beteiligten Geräte (Abzweigklemme und Koppler/Box) erforderlich.
- Es findet bei Fast-Hot-Connect-Geräten ein beschleunigter Ethernet-Verbindungsaufbau gegenüber der normalen FastEthernet-Verbindung statt.
  Wird zusätzlich noch auf den Einsatz von Distributed-Clocks-Funktionen in der gesamten Topologie

verzichtet, entfällt auch die Resynchronisierungszeit der Komponenten. Dann sind Gruppenhochlaufzeiten von < 1 Sekunde möglich, vom Stecken der Ethernet-Verbindung bis zum OP-State.

- im TwinCAT ADS Logger wird eine falsche Port-Zuordnung detektiert

Message
'Term 17 (EK1122-0080)' detected invalid hot connect group at port 3
'Term 21 (EK1122)' detected invalid hot connect group at port 3. Only Fast-Hotconnect slaves are allowed at this port.

Abb. 38: Detektion falsche Portzuordnung TwinCAT-Logger

Konfiguration

Die Konfiguration von Fast-Hot-Connect-Gruppen im TwinCAT System Manager erfolgt genauso wie Hot-Connect-Gruppen unter Angabe der zugehörigen Gruppen-ID.

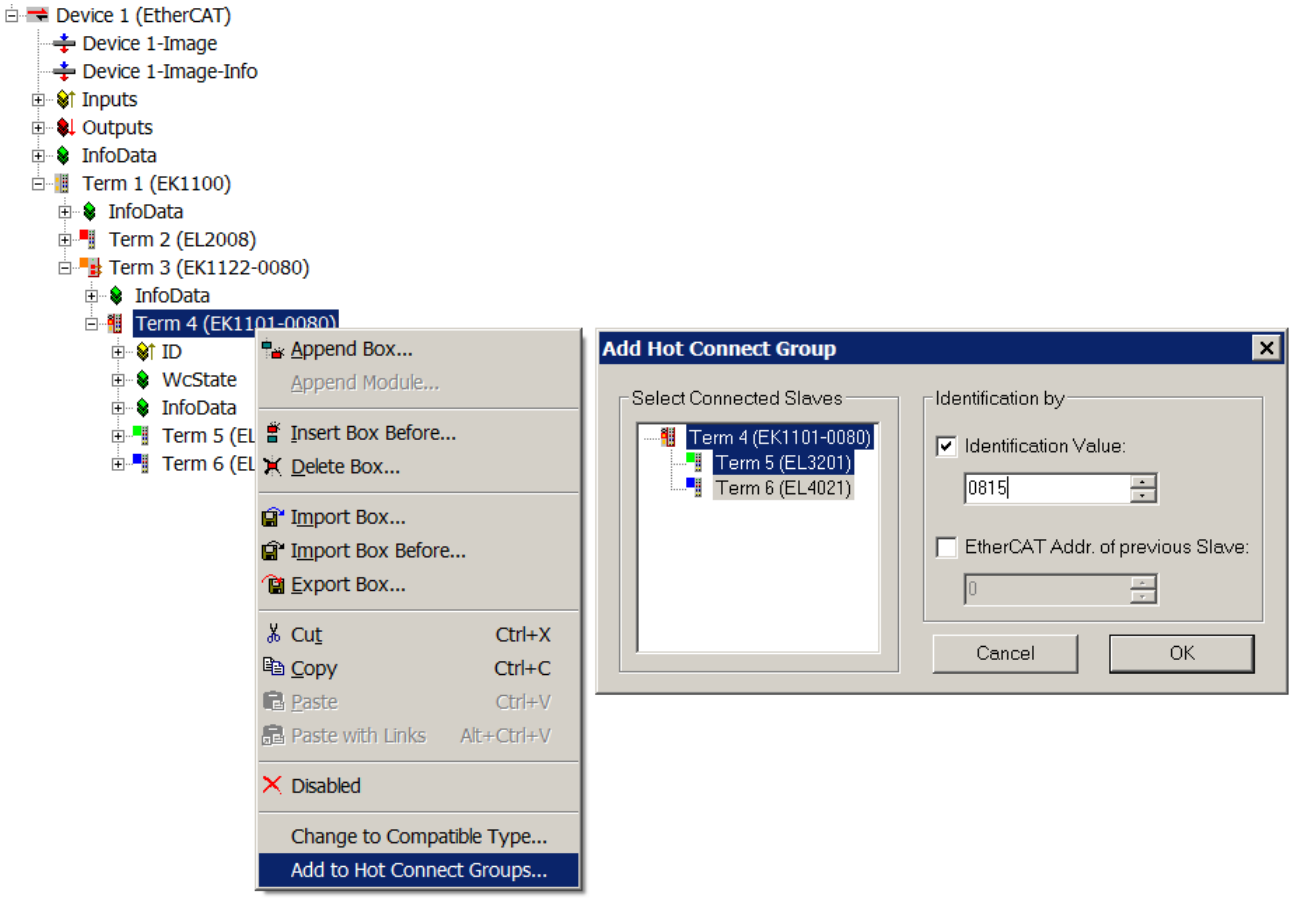


Abb. 39: Konfiguration Fast-HotConnect Gruppe

Im TwinCAT-System Manager sind entsprechende FastHotConnect-Ports rot gekennzeichnet.

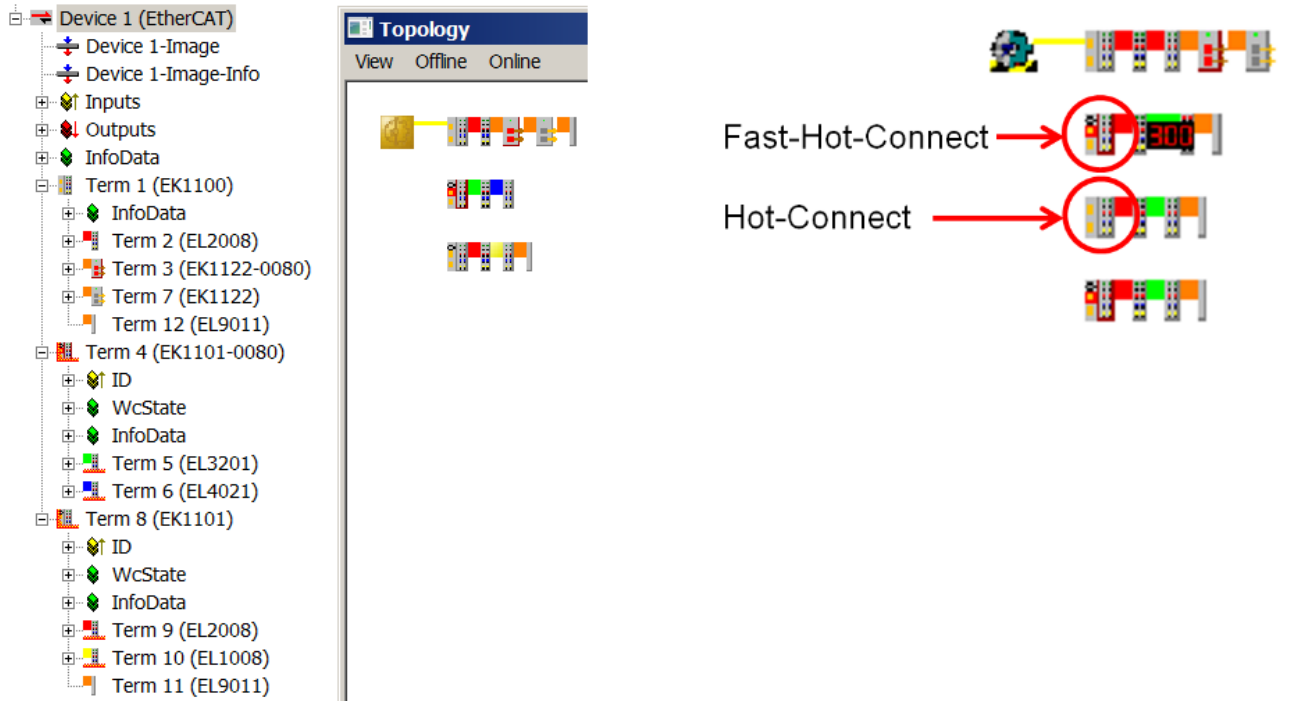


Abb. 40: Kennzeichnung im TwinCAT System Manager

Eine Konfiguration von FHC-Gruppen ist nur möglich, wenn mindestens 1 entsprechender Abzweig z. B. EK1122-0080 vorhanden ist.

Distributed Clocks

Wenn keine Distributed-Clocks-Funktionen genutzt werden, ist dies in den Master-Einstellungen durch ein fehlenden „DC in use“ sichtbar:

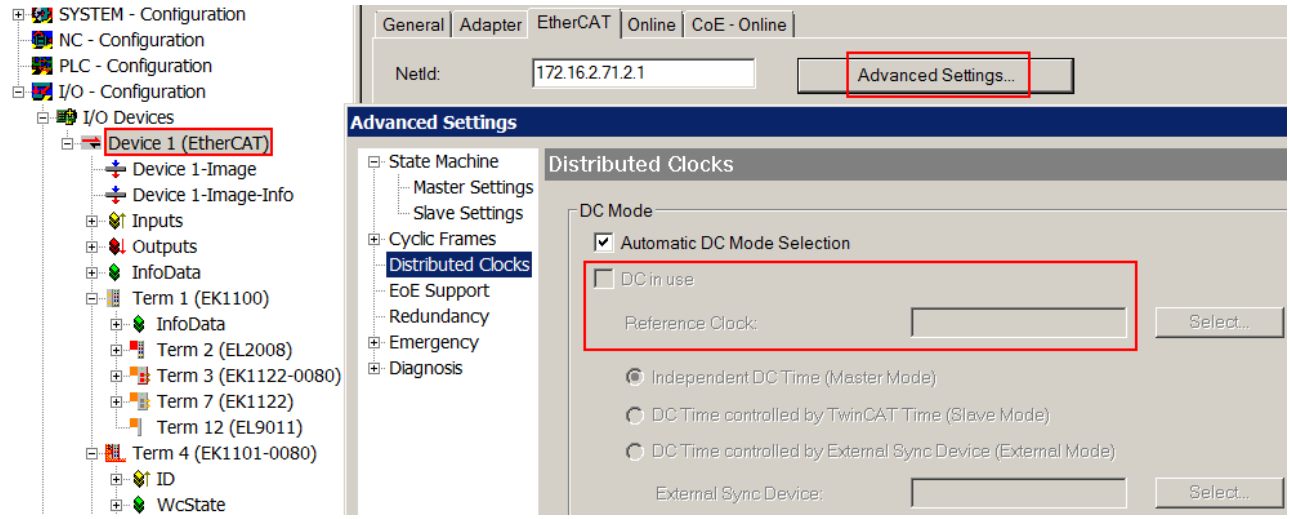


Abb. 41: DC-Master-Einstellung

Diese Einstellung wird vom System Manager automatisch gewählt, wenn keine EtherCAT-Slaves in der Konfiguration enthalten sind, bei denen Distributed Clocks aktiviert ist. Es sollte hier nicht durch den Anwender „DC in use“ willkürlich deaktiviert werden, weil sonst diese Teilnehmer nicht mehr funktionieren.

2.3.2.3 Adressierung

Addressierung und Identifizierung

Zur Zeit sind drei verschiedenen Verfahren zur eindeutigen Identifizierung von Hot-Connect-Stationen definiert:

- **SecondSlaveAdress (SSA)**
  Hierbei wird die Stationsadresse [0..65536] fest im E²PROM des EtherCAT-Slave gespeichert. Beim Start lädt der EtherCAT-Slave-Controller (ESC) diese Adresse in sein Register 0x0012, von wo es vom EtherCAT Master ausgelesen werden kann.
  Geschrieben wird diese Adresse
  - durch den EtherCAT Master/TwinCAT Systemmanager vom Anwender bei Anlagenerrichtung
  - durch eine Bedienoberfläche am Slave (Tastatur, Display, Wahlschalter, ...) bei Geräteinbetriebnahme
- Der Anwendungsfall ist in nahezu allen Fällen nicht die Verwendung als flexible Station sondern die eindeutige Identifizierung eines Gerätes.
  Im Austauschfall muss im Ersatzgerät die bisherige Adresse neu hinterlegt werden.
  Eine geänderte SSA wird i.d.R. vom ESC erst nach einem Power-Neustart des Gerätes übernommen.
  Die SSA kann vom Anwender durch eine Lesen des Registers 0x0012 aus der PLC geprüft werden (siehe TcEtherCAT-lib --> FB_EcPhysicalReadCmd)
- **InputWord/IdentificationValue/Data Word**
  Hierbei wird eine der Stationsadresse [0...65535] entsprechende Bitfolge im Prozessdatenbereich eines EtherCAT-Slaves erwartet und vom Master als Stationsadresse interpretiert.
  TwinCAT 2.10 erwartet die 16 Bit Daten ab Register 0x1000. Ab TwinCAT 2.11R2 kann die Information an beliebiger Stelle im Slave liegen, durch die ESI-Beschreibung wird der EtherCAT Master vom Speicherort im ESC informiert, im Dialog (s. Abb. *Default-Einstellung eines EtherCAT slave der den Identifizierungsmodus DataWord unterstützt*) genannt ADO (AdressOffset).
  Das InputWord wird i.d.R. in der Konfiguration als Prozessdatum beim unterstützenden Slave dargestellt (siehe EK1501, EK1101).
- **Explicit Device Identification**
  Hierbei teilt der Slave dem Master während der Hochlaufphase durch das AL Status Register 0x0134 seine ID [0...65535] auf Anforderung durch den Master mit.
  Dieses Verfahren wird ab TwinCAT 2.11R3 unterstützt.

Adressierungsmethode

Generell ist in der zum EtherCAT Gerät gehörenden ESI-Datei hinterlegt, welche Adressierungsarten der Slave unterstützt. Dies wird im TwinCAT System Manager -> EtherCAT Slave -> Erweiterte Einstellungen angezeigt und sollte im allg. nicht anwenderseitig verändert werden.
Siehe Abb. (*Default-Einstellung eines EtherCAT slave der den Identifizierungsmodus DataWord unterstützt*) Ausnahme: Der Identifizierungsmodus SSA ist in jedem Slave möglich, der über ein ESC-EEPROM verfügt (Herstellerhinweis beachten). Beckhoff EtherCAT Klemmen/Boxen verfügen i.d.R. über ein EEPROM.

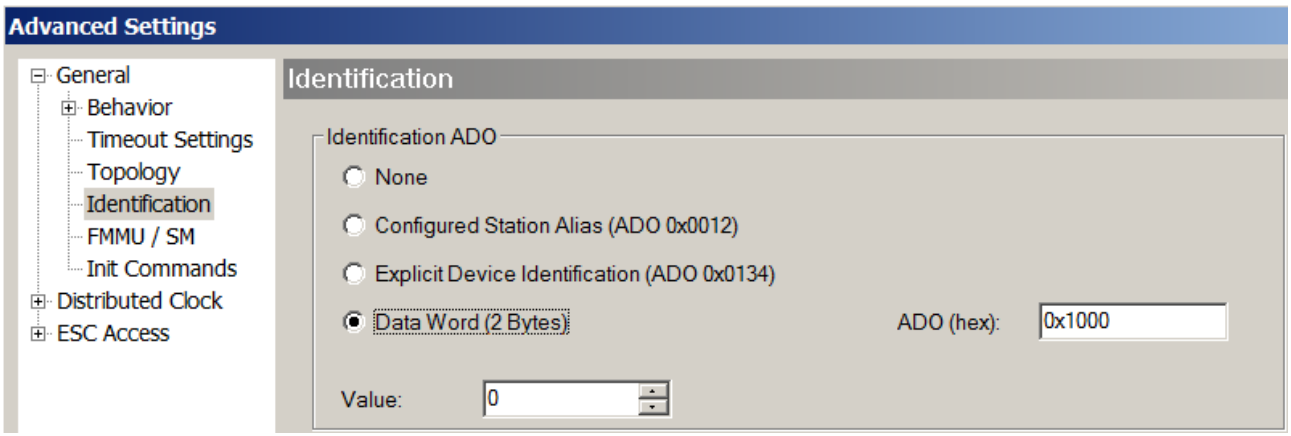


Abb. 42: Default-Einstellung eines EtherCAT slave der den Identifizierungsmodus DataWord unterstützt

Alle EtherCAT-Slaves werden immer über das Autoincrement-Verfahren initial angesprochen, dann versucht der EtherCAT-Master die konfigurierten Stationsadressen im Feld zu finden und den Stationen/Kopplern zuzuordnen. Dazu wird der abgehende Port gezielt geöffnet und geschlossen um die IDs zu ermitteln.

Adressvergabe im Slave

Explicit Device Identification

Die ID-Einstellung ist nach Geräte-Anleitung vorzunehmen.

InputWord/IdentificationValue

In der Regel erlaubt bei solchen Slaves ein von außen zugänglicher Wahlschalter die Einstellung der Position, der Wahlschalter kann verplombbar gestaltet oder über einen Softwaremechanismus geschützt werden. Andere Geräte bieten eine Bedienungsoberfläche die die ID-Einstellung erlaubt.

Beispiel: Beckhoff EK1101 - freie Zugänglichkeit der ID-Schalter

Abb. 43: EK1101 mit ID-Switch

SecondSlaveAddress

Die SSA kann vom Master oder über eine Bedienoberfläche am Slave gesetzt werden. Im Folgenden die Beschreibung des Setzens über den Beckhoff TwinCAT Master:

- Nehmen Sie den EtherCAT-Slave ohne Adressierung in einer einfachen Konfiguration in Betrieb. Der Slave sollte im OP sein, WorkingCounter = 0, keine LostFrames
- Gehen Sie über
  Slave -> EtherCAT -> Advanced Settings -> E²PROM zum Dialog *SecondAddress*.
- Dort wird die aktuelle Adresse angezeigt, schreiben Sie die *New Second Address* in das Slave E²PROM.
- Nach einem Power-Neustart wird die Adresse übernommen.
- In diesem Beispiel wird die neue Adresse $10_{dec}/0A_{hex}$ gesetzt.

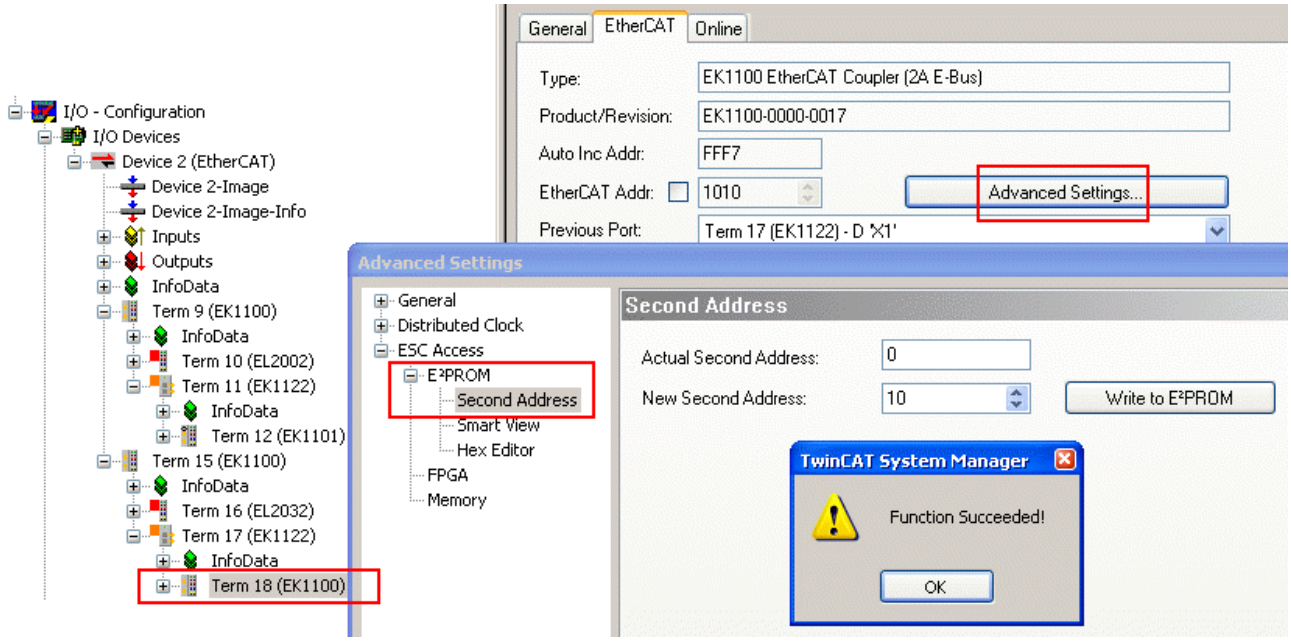


Abb. 44: SSA setzen

Ob der Slave die Adresse übernommen hat, kann wie folgt überprüft werden:

Gehen Sie im selben Dialog auf *Memory* und kontrollieren Sie den Inhalt von Register 0x0012.

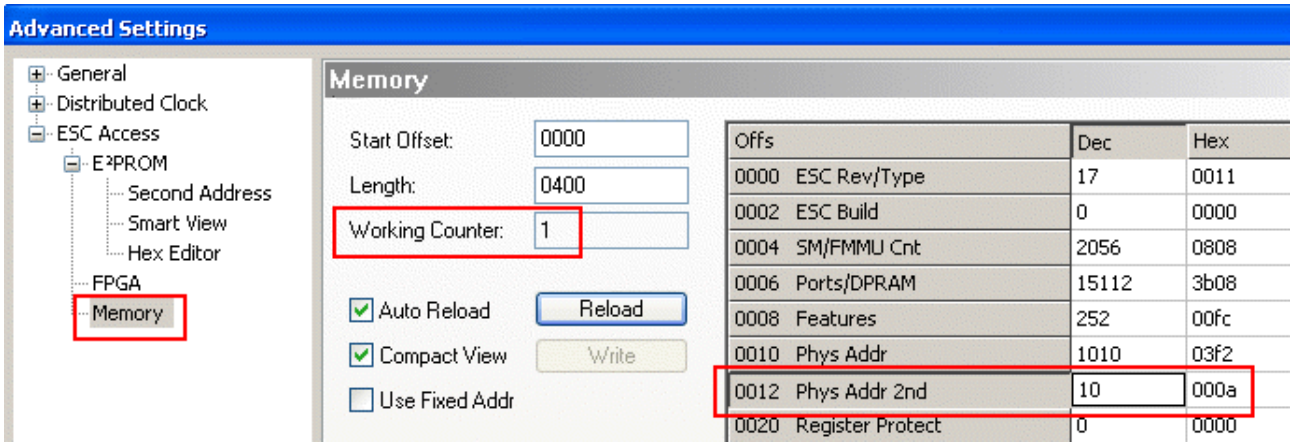


Abb. 45: Kontrolle der SSA

Wenn nichts angezeigt wird:

- Normalerweise ist Wc=0 der "fehlerfreie" Zustand, in diesem Fall muss WorkingCounter=1 stehen, sonst erfolgt aus anderen Gründen keine Kommunikation mit dem Slave
- Kontrollieren Sie, ob Sie sich auf dem richtigen Slave befinden.
- Wie durch *Length=400* ersichtlich, liest der Dialog 400_{hex} Byte in einem Vorgang aus dem ESC. Dies unterstützen die meisten Fremd-ESC nicht, dann werden keine Daten in den Spalten angezeigt. Versuchen Sie ab *Length=2* herauszufinden, wie viele Bytes Ihr ESC unterstützt.

Hinweis TwinCAT 2.11

Der Einstellungsdialog für die SSA wurde in TwinCAT 2.11 leicht verändert, behält aber die gleichen Eigenschaften.

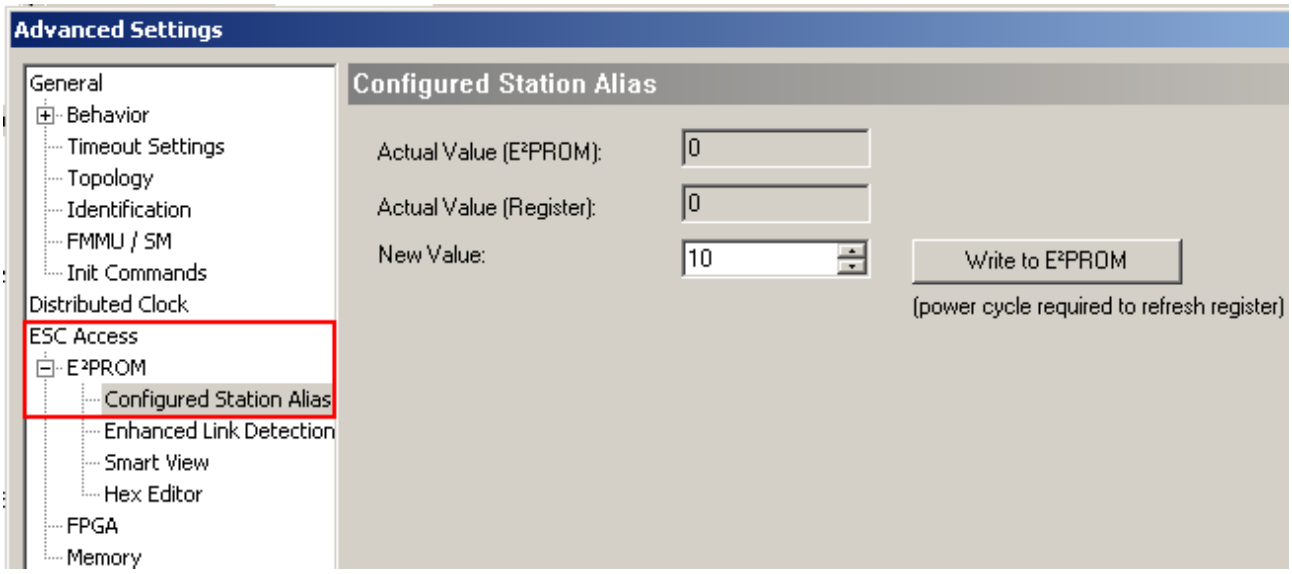


Abb. 46: SSA setzen unter TwinCAT 2.11

Kompatible Beckhoff-Geräte

Im Allgemeinen kann jedem EtherCAT-Slave eine Hot-Connect-Adresse zugeordnet werden, unabhängig davon ob es eine IO-Klemme, ein Antrieb oder eine Baugruppe aus Koppler und Klemmen ist.

Hot-Connect-Fähigkeit

Prüfen Sie, ob der von Ihnen verwendete EtherCAT-Slave Hot-Connect-fähig ist. Beckhoff-Slaves unterstützen die Funktionen aktuell wie folgt (Stand 05/2012): (Siehe folgende Tabelle)

Version	SecondSlaveAdress	InputWord	Explicit Device Identification
EK1100	ab HW18	-	-
EK1101, EK1501, EK1101-0080	alle	alle	-
BK1120	ab HW09	-	-
EL-Klemmen	i.allg. ja für die meisten Bauserien ab einer XML-Version xxxx-xxxx-0016	-	-
EP-Boxen	i.allg. ja für die meisten Bauserien ab einer XML-Version xxxx-xxxx-0016	-	-
AX2xxx	-	-	-
AX5xxx	alle	s. Dokumentation	s. Dokumentation

2.3.2.4 Setup TwinCAT 2.10

Im Folgenden wird nun der Konfigurationsablauf unter TwinCAT 2.10 an einem Beispiel beschrieben.

Schritt 1: Erstellen der Ausgangskonfiguration

Zuerst müssen alle Komponenten in der Konfiguration vorhanden sein, in der (Offline-) Topologieansicht sind die Komponenten dann so verbunden, wie die Reihenfolge in der Konfiguration bzw. die "PreviousPort" Angabe vorgibt. Hier die Offline-Ansicht von 4 Stationen mit EK1100, Ausgangsklemmen (rot) und einer EK1122.

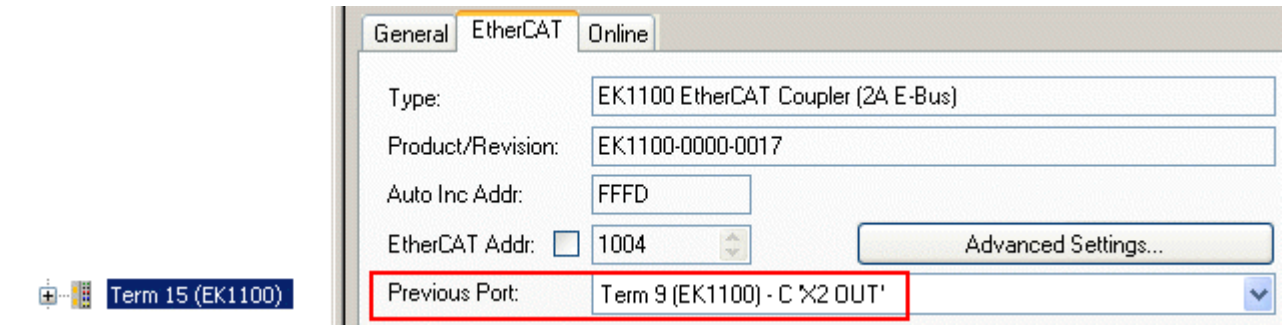


Abb. 47: Anlegen der Konfiguration

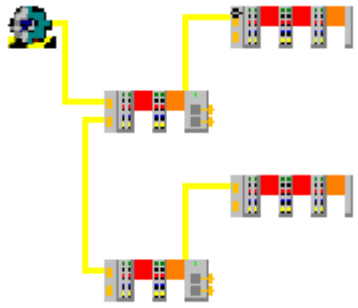


Abb. 48: Anlegen der Konfiguration

Schritt 2: als HotConnect-Gruppe definieren

Durch Rechtsklick auf den EtherCAT-Slave öffnet sich der Dialog zur HotConnect-Gruppe.

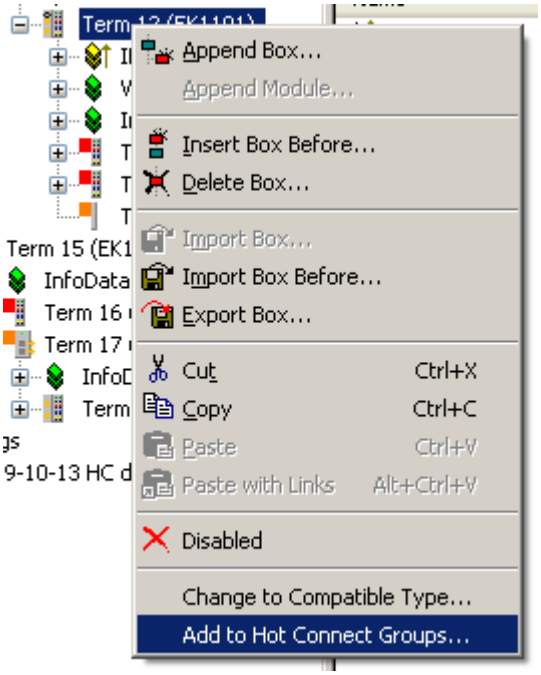


Abb. 49: HotConnect-Gruppe zuordnen

Geben Sie nun die gewünschte Adresse an, achten Sie dabei auf das verwendete Verfahren.

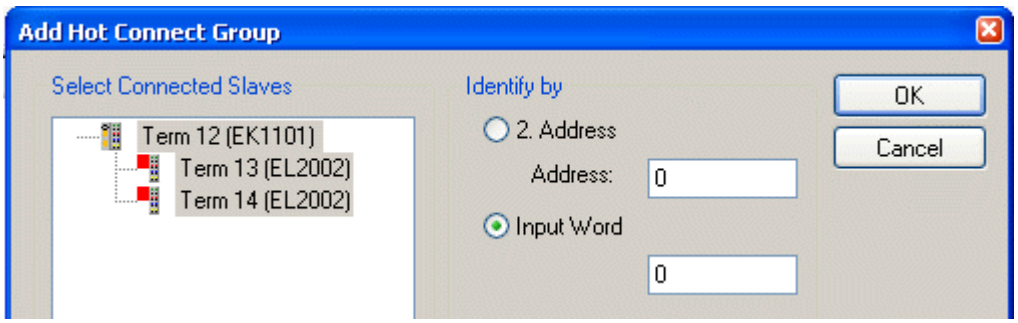


Abb. 50: Adresse zuordnen

Der Erfolg der Maßnahme ist zuerst in der Konfiguration zu sehen: Die Station löst sich aus der Reihefolge und wird unten rot markiert angehängt. Die Eigenschaft "*PreviousPort*" ist nicht mehr einstellbar, da TwinCAT die Station nun überall erwartet.

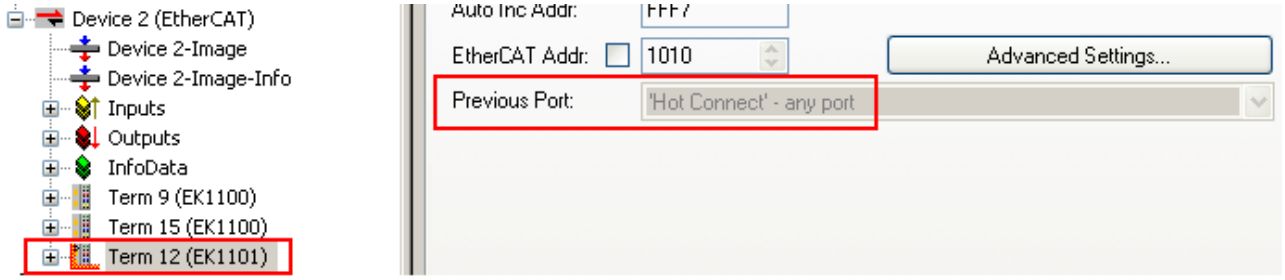


Abb. 51: Hot Connect-Gruppe

In der (Offline-)Topologieansicht löst sich die Station ebenfalls und hängt verbindungslos in der Luft.

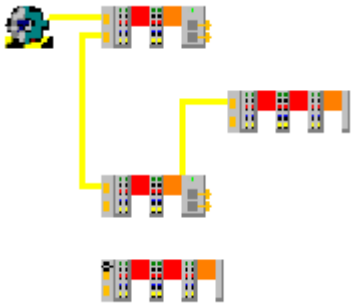


Abb. 52: Topologieansicht

Dies gilt auch für alle weiteren HotConnect-Stationen.

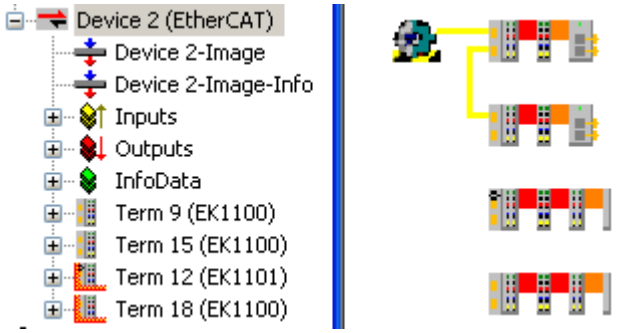


Abb. 53: Topologieansicht

Schritt 3: Online-Betrieb

Erst wenn diese Konfiguration nach der Aktivierung in Betrieb ist, kann der **Online**-Topologieansicht die tatsächliche Anordnung eingesehen werden.

Grüne Balken über EtherCAT Slaves zeigen den OP-State an, rote den INIT-State.

In der Abb. *Online-Topologieansicht* ist erkennbar, dass die HotConnect-Station A an der EK1122 angedockt wurde, während die Station B wegen Abwesenheit nicht am Datenverkehr teilnimmt.

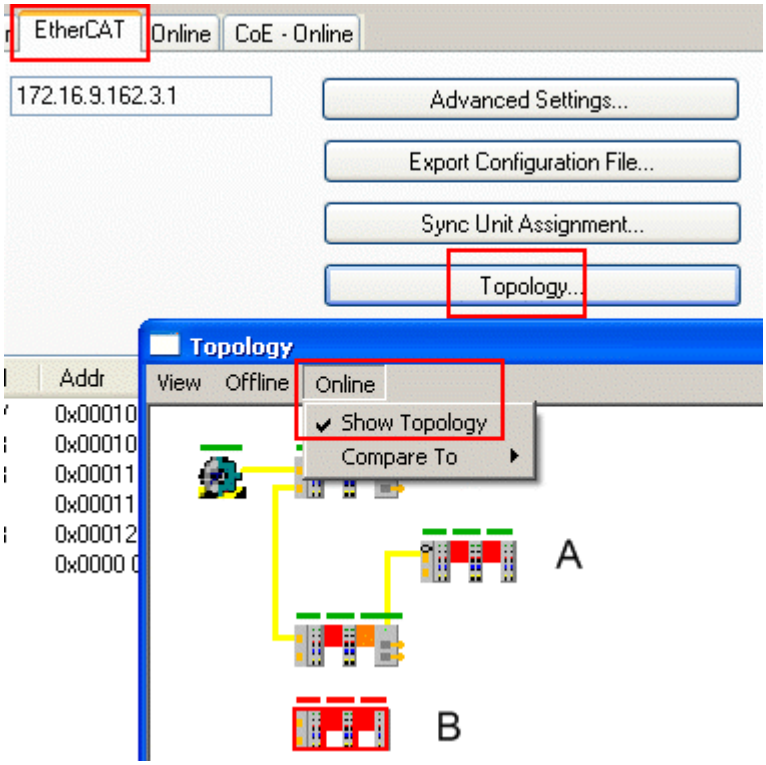


Abb. 54: Online-Topologieansicht

Auch in der Online-Anzeige werden diese Slaves von Station B korrekt als nicht präsent angezeigt.

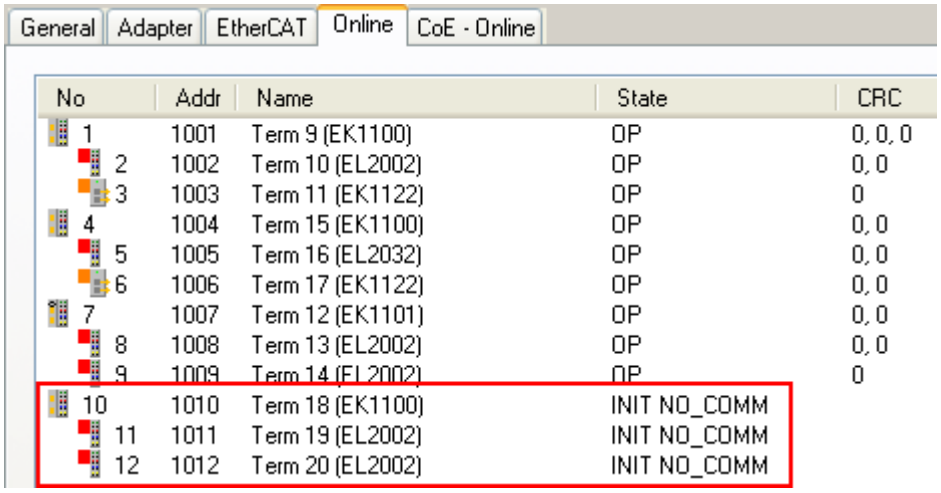


Abb. 55: Online-Anzeige

Die vergebene Adresse kann auch nachträglich über den Reiter *HotConnect* geändert werden.

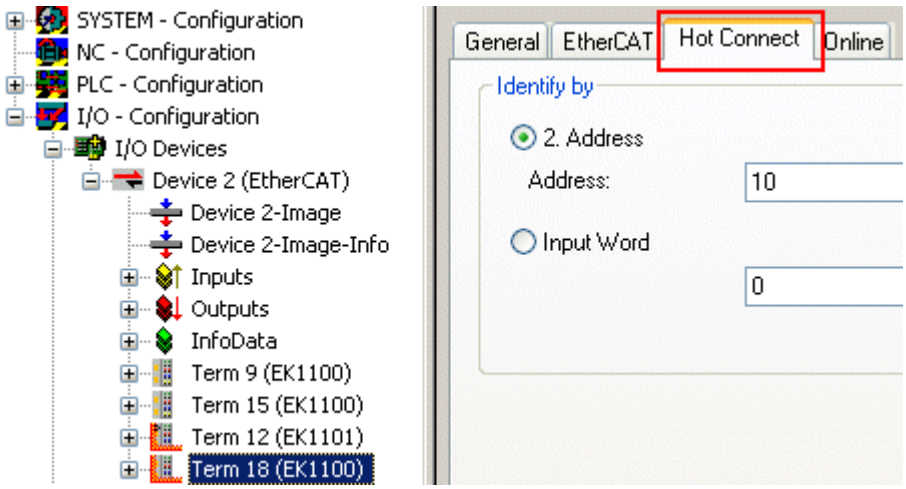


Abb. 56: Reiter HotConnect

2.3.2.5 Setup TwinCAT 2.11R2

TwinCAT 2.11R2 wird mit anderen Dialogen konfiguriert, grundsätzlich ist die Vorgehensweise jedoch die gleich wie unter TwinCAT 2.10.

Parametrierung Konfiguration

Die EtherCAT Konfiguration kann durch Online Scan oder manuell erstellt werden, wie im Folgenden beschrieben.

1. Zuerst ist die Konfiguration in der maximalen Ausbaustufe zu erstellen. D.h. alle maximal jemals in der späteren Anlage vorhandenen Buskopplerstationen müssen in der Konfiguration enthalten sein, auch wenn sie nicht alle *gleichzeitig* in Betrieb sein werden. Die Reihenfolge und damit die entsprechende Angabe "Previous Port" ist dabei nicht wichtig.

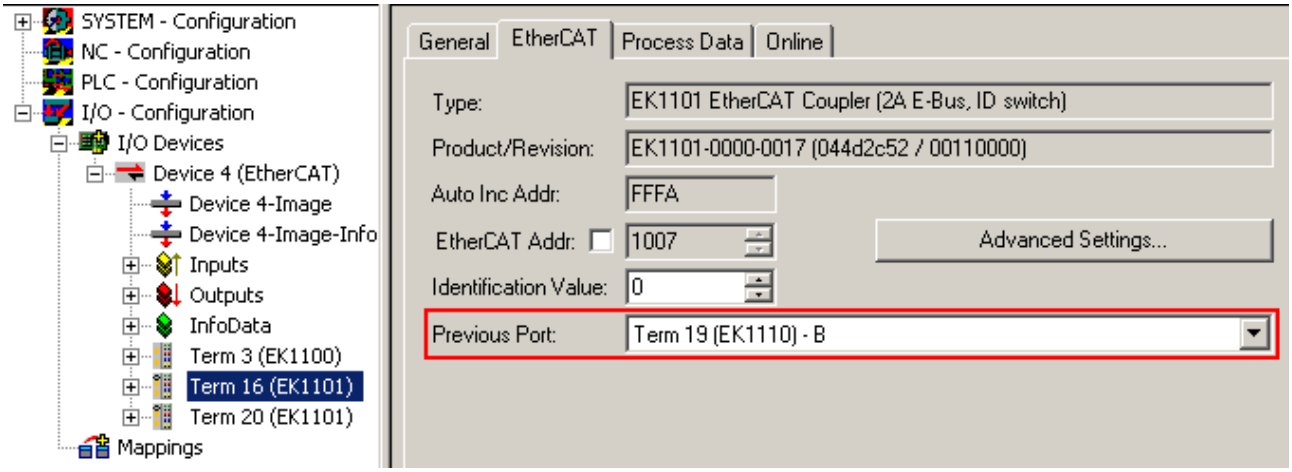


Abb. 57: PreviousPort an Busstation

HotConnect Einheiten

Es ist nur technologisch erlaubt, über Ethernet kommunizierende Einheiten mit einer EtherCAT-ID auszustatten, denn nur diese dürfen/können im Betrieb an- und abgesteckt werden. Dies sind also Geräte mit RJ45/LWL/M8/M12-Ethernet-Anschluss (Koppler, EP-Boxen, Antriebe, ...). Klemmen dürfen nicht unter Spannung gezogen/gesteckt werden, die Festlegung als flexible HotConnect-Gruppe ist also ohne Sinn.
Zur eindeutigen Identifizierung über SSA können natürlich auch Klemmen herangezogen werden.

2. Die designierten flexiblen Busstationen werden nun im System Manager mit der HotConnect-Eigenschaft versehen. Durch Rechtsklick auf eine Kopplerstation/EP-Box wird der Dialog aufgerufen.

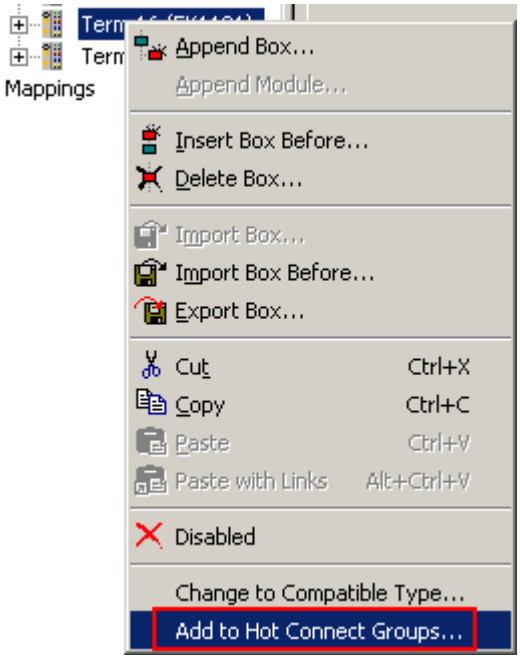


Abb. 58: Aufruf HotConnect-Dialog

3. Über zwei Eigeschaften kann in TwinCAT 2.11R2 eine angekoppelte Busstation als die "Richtige" identifiziert werden:
   - die ID "identification value"
   - die EtherCAT-Adresse des vorangehenden Slaves
   Die entsprechende Eigenschaft ist zu aktivieren und der Wert [0; 65535] anzugeben. Die ID betrifft dabei sowohl Identifizierung durch **InputWord** als auch **SecondSlaveAdress**. Default ist hier die Einstellung InputWord, soll SSA genutzt werden ist sie wie nachfolgend beschrieben zu ändern.

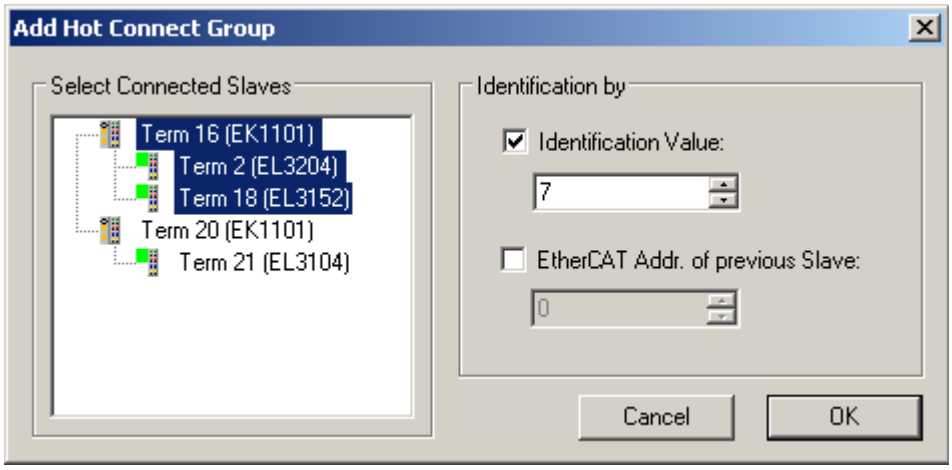


Abb. 59: Eigenschaften HotConnect

Identifizierung über "previous slave"

Wird die Adresse "previous slave" angegeben, kann eine so parametrierte Busstation nicht mehr an jedem beliebigen freien Port eingesteckt bzw. betrieben werden, sondern nur noch an dem einen EtherCAT Port, der direkt nach dem angegebenen Slave kommt. Die Nutzung dieser Methode schränkt also den Einsatzort einer solchen Station erheblich ein. "Vorangehend" meint hier den EtherCAT-kommunikationstechnischen Vorläufer-Slave, nicht unbedingt den in der Topologie-Reihenfolge vorangehenden. Insbesondere bei Nutzung der Mehrport-Slaves EK1122/EK1521 ist darauf zu achten.

4. Die Station erhält dadurch einen zusätzlichen Reiter "HotConnect", wo diese Eigenschaften auch nachträglich modifiziert werden können.

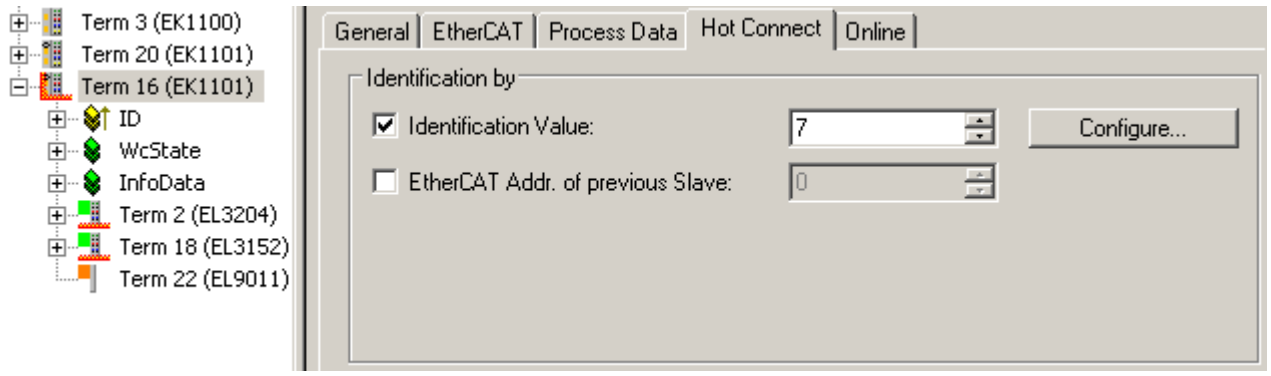


Abb. 60: Eigenschaften HotConnect

In den Eigenschaften kann über *Configure* die Default-Identifizierung InputWord (aus den Prozessdaten, im Dialog genannt "DataWord") auf SecondSlaveAdress (im Dialog: "Configured Station Alias" aus Slave Register 12_{hex}) geändert werden.

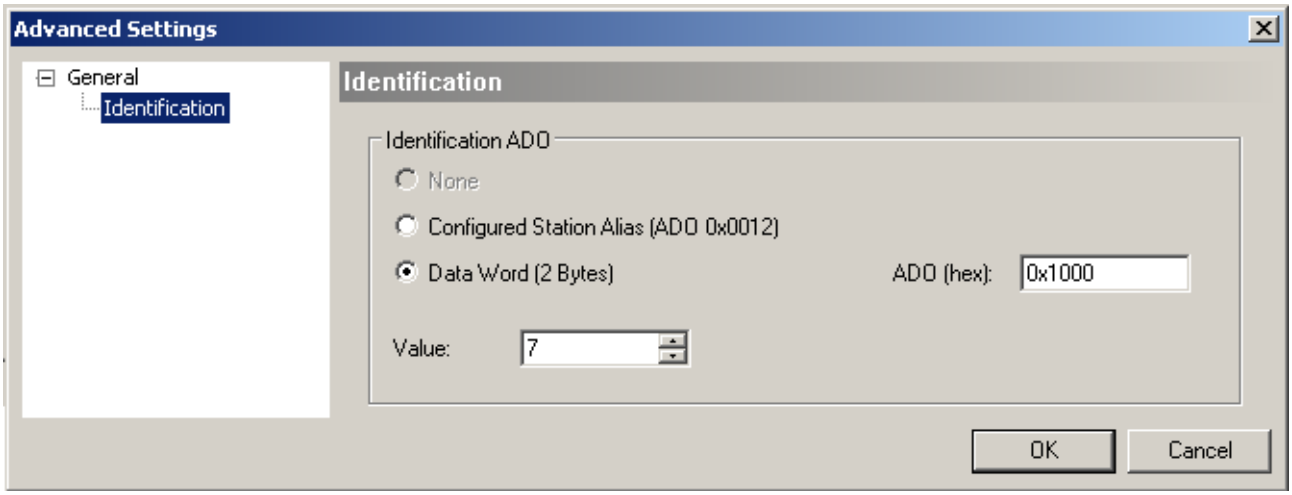


Abb. 61: Detail Einstellung

Das Geräte-Symbol im Systemmanager wird rot als HotConnect gekennzeichnet.

Identifizierung über "Data Word"

Die 16 Bit Identifizierungsdaten werden aus dem Slave an einer vorgegebenen Stelle gelesen. TwinCAT 2.10 unterstützt hier nur Register 1000_{hex}. TwinCAT 2.11 kann von jeder beliebigen Stelle lesen, so wie dies im Dialog Abb. *Detail Einstellung* "ADO (hex)" angegeben ist. Defaulteinstellung ist Reg. 1000_{hex} bzw. die Angabe aus der ESI-Datei.

HotConnect Betrieb

Für den HotConnect Betrieb muss sich TwinCAT im Config/FreeRun oder RUN-Modus befinden. Nach Konfigurationsänderungen (andere ID, ID-Ort etc.) ist die Konfiguration neu zu laden bzw. ein TwinCAT Restart durchzuführen.

Beispiel 1: Gruppen nicht identifiziert aber verbunden

Sind Stationen mit ID über Etehrnet angebunden aber nicht identifiziert, finden sich an den entsprechenden Slaves entsprechende Status-Meldungen:

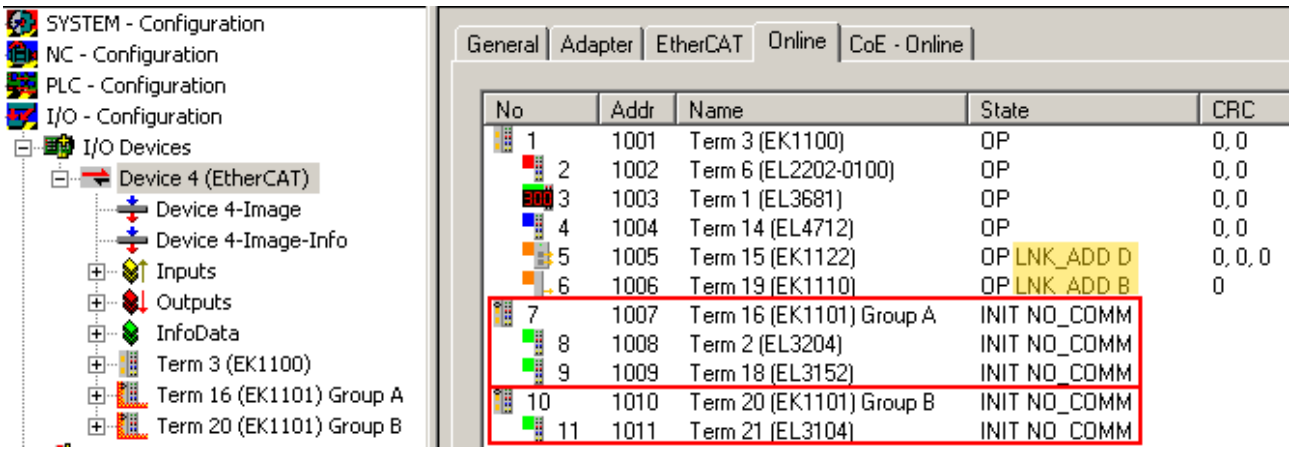


Abb. 62: Statusmeldungen

- An den Slaves, an denen die Stationen angesteckt wurden findet sich "LNK_ADD" mit der Portangabe: dort befindet sich ein unerwarteter LINK zu nicht identifizierten Teilnehmern.
- Die Stationen selbst (hier Group A, Group B) sind noch im INIT, keine Kommunikation möglich

TwinCAT versucht zu einem solchen Zeitpunkt zyklisch fortlaufend die angekoppelten Teilnehmer zu identifizieren.

Beispiel 2: Gruppe identifiziert

Der EK1101 "Group A" wird nun über die ID-Schalter auf die richtige ID gestellt. Nach wenigen Sekunden erkennt TwinCAT dies und nimmt die Gruppe in den Prozessdatenverkehr auf und setzt die Teilnehmer in den OP-State.

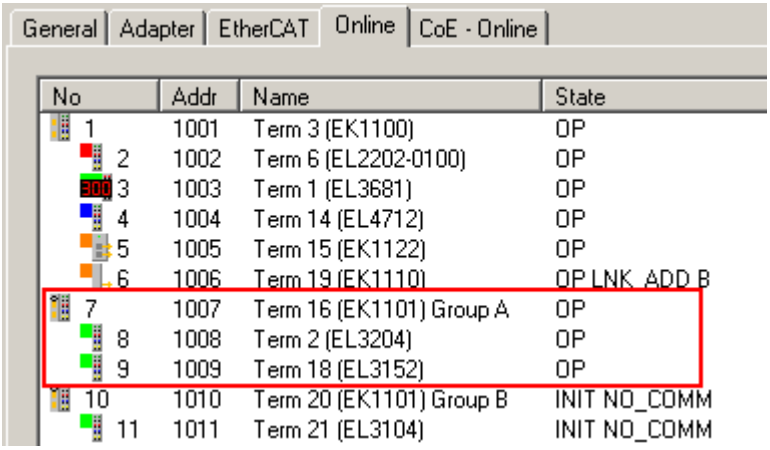


Abb. 63: Group A in Betrieb

Aus der Steuerung erkennbar ist dies über die Prozessdaten (WC-State, State) des Kopplers:

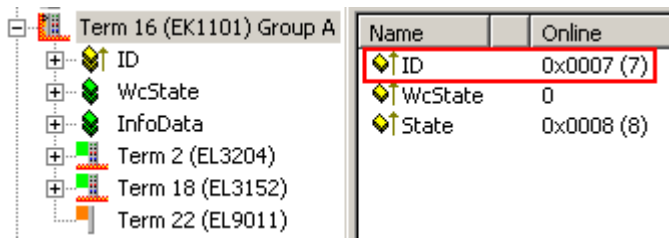


Abb. 64: Prozessdaten der ID-Baugruppe

Beispiel 3: Nutzung der Topologie-Ansicht

Die Online-Topologie-Ansicht bildet den aktuell realen Link-Status und Ankopplungsort ab.

Die Offline Anzeige zeigt die ID-Stationen losgelöst ohne Kommunikationsverbindung, da nicht bekannt ist wo die Stationen angebunden sind.

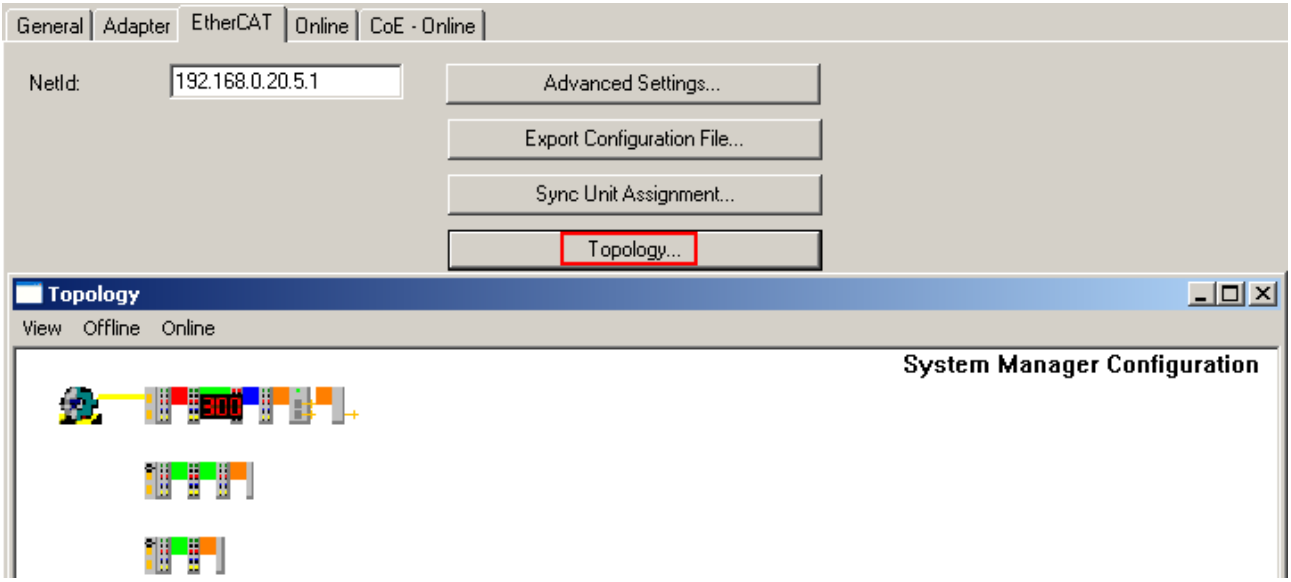


Abb. 65: Offline Ansicht

Sind die Stationen nicht identifiziert und präsent, werden sie in der Online Ansicht rot markiert.

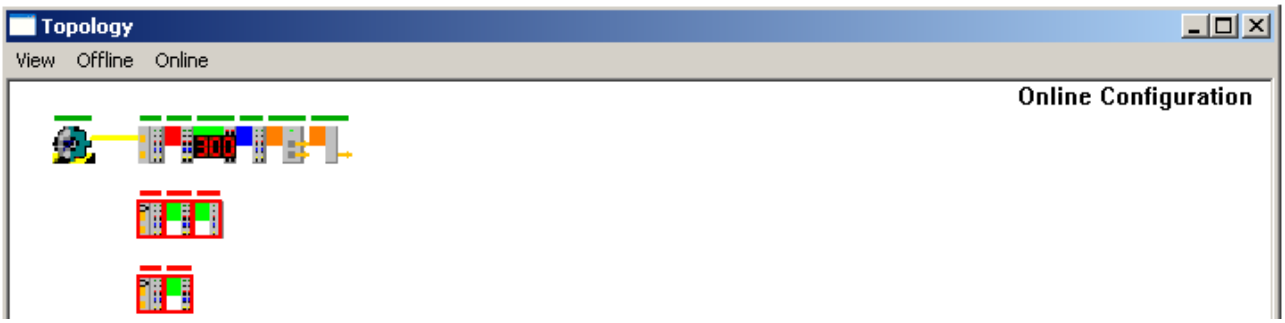


Abb. 66: Online Ansicht

Werden die Stationen dann durch Veränderung der ID (hier: Schalterstellung am EK1101) zugeschaltet, erscheint der grüne OP-Balken.

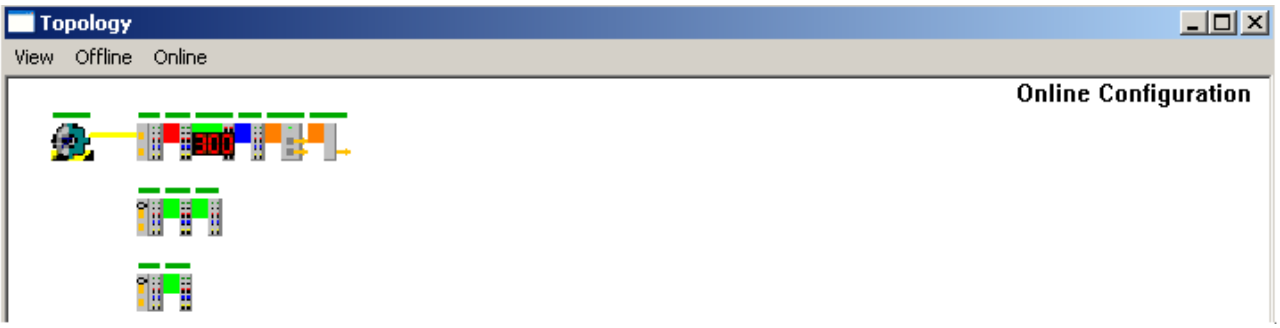


Abb. 67: Online Ansicht

Nach einem Wechsel zur Offline-Ansicht und zurück, wird der aktuelle Verbindungszustand angezeigt.

Abb. 68: Aktueller Verbindungszustand

2.3.2.6 Identifizierung

Wird HotConnect genutzt, kann in der Online-Topologie-Ansicht im System Manager der aktuelle Topologiestatus eingesehen werden, d.h. wo welche HotConnect-Gruppe angebunden ist. Um aus der Steuerung/PLC heraus auf diese Information zuzugreifen sind einige ADS-Zugriffe nötig.

https://infosys.beckhoff.com/content/1031/ethercatsystem/Resources/zip/2469147275.zip (PLC und System Manager) verfolgt folgenden Ansatz:

- ein FunctionBlock enthält den notwendigen Code
- der FB kann mehrfach instanziiert werden und ist mit seinen I/O-Variablen mit einem HotConnect-Kopf (EK-Koppler, EP-Box) zu verlinken. Dadurch erhält er Informationen über den EtherCAT Master und den eigenen ADS Port.
- Sobald der Wc (Working Counter) Betrieb meldet, werden über ADS die eigenen Daten vom EtherCAT Master angefragt, wie auch die Daten der direkt anschließenden Geräte.
- Zusätzlich werden noch Identität und Name der benachbarten Geräte ermittelt. Identifizierungsmerkmal ist die konstant bleibende EtherCAT Adresse.

Dieses https://infosys.beckhoff.com/content/1031/ethercatsystem/Resources/zip/2469149451.zip enthält auch noch eine Visualisierungsvorlage die Platzhalter-Variablen nutzt.

EtherCAT Topologie in der PLC

Im Einleitungsteil der Dokumentation zur TcEtherCAT.lib befindet sich ein umfangreiches Beispielprogramm zur Ermittlung unterschiedlichster Informationen eines EtherCAT Netzwerkes. Diese Dokumentation ist im InformationSystem (Online oder auf DVD) zu finden: TwinCAT --> TwinCAT PLC --> PC Libraries --> EtherCAT --> Übersicht.

Dieses Beispielprogramm verwendet Bausteine von dort.

Es wird folgender Aufbau angenommen:

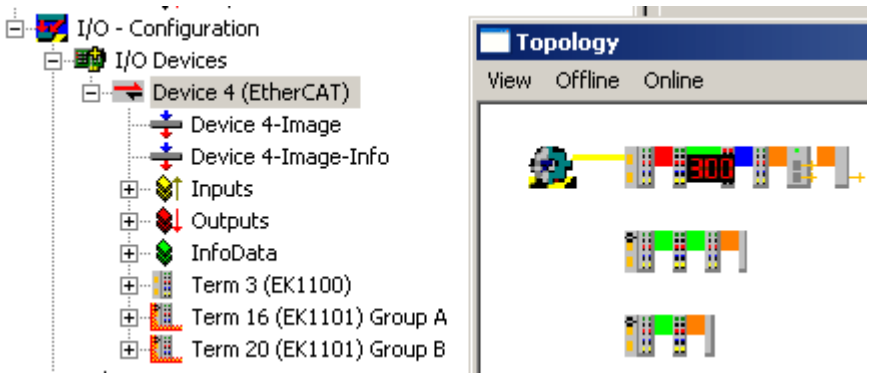


Abb. 69: Demo-Aufbau

Der erste Koppler EK1100 ist mit EK1110 und EK1122 als Abzweigstationen für die EK1101 *GroupA* und *GroupB* ausgerüstet.

Zum Vorgehen: entscheiden ist die Verlinkung der Adressinformationen in den FunctionBlock hinein, s. Abb. *AdsAddr*.

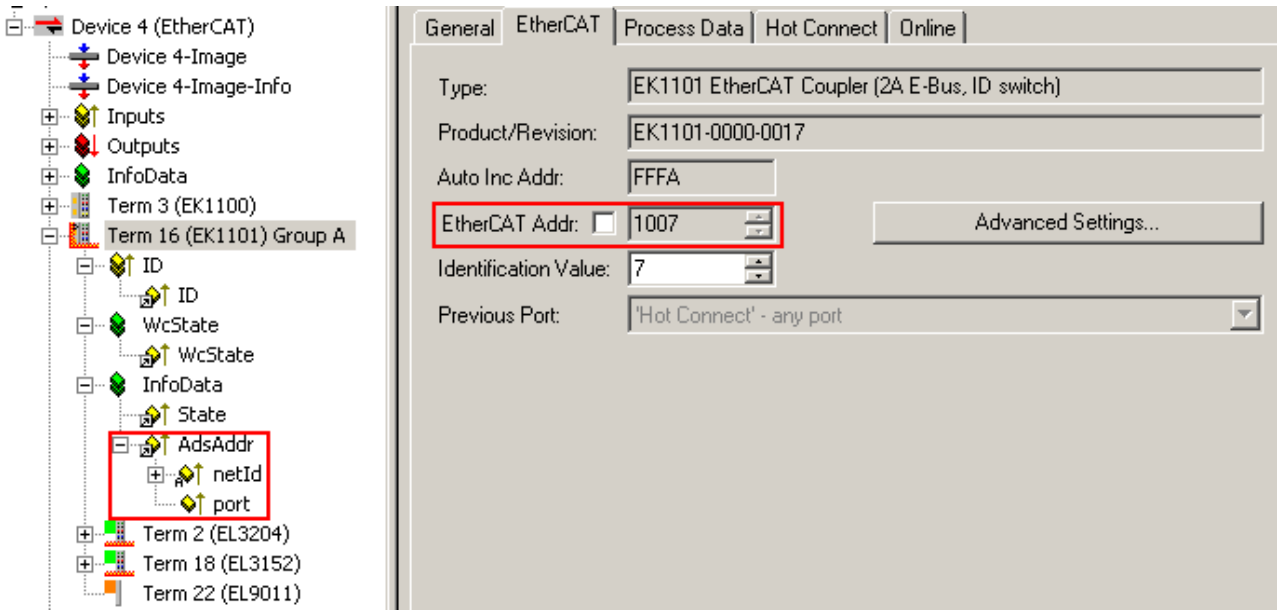


Abb. 70: AdsAddr

Im Beispiel hat der HotConnect-Koppler EK1101 die EtherCAT Adresse 1007. Über die Verlinkung kann auch der FB auf die EtherCAT Master NetId und den ADS Port = EtherCAT Adresse zugreifen.

ADS Adresse

Wird die ADS Adresse in den grünen InfoDaten nicht angezeigt, kann die Anzeige in den Erweiterten Einstellungen des Slaves eingeschaltet werden. (Siehe Abb. *„Einschalten der „ADS Address“ in den Erweiterten Daten (Advanced Settings)“*)

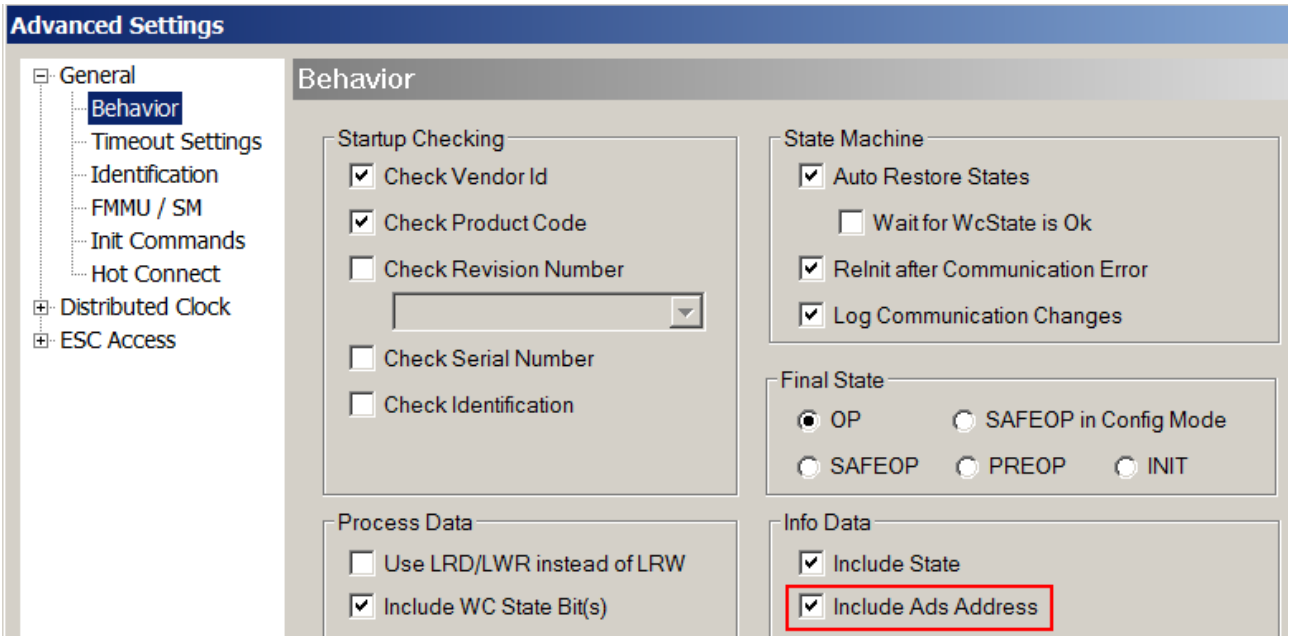


Abb. 71: Einschalten der „ADS Address“ in den Erweiterten Daten (Advanced Settings)

Der FunctionBlock gibt eine StatusStructur aus, die Informationen über die eigene Identität der verlinkten Kopplers und die angeschlossenen Geräte liefert, s. Abb. *Aufbau Information*. Da ein EtherCAT Slave über maximal 4 Ports verfügt, können auch maximal 4 benachbarte Geräte ermittelt werden. Die Gerätenamen werden aus der Konfiguration entnommen.

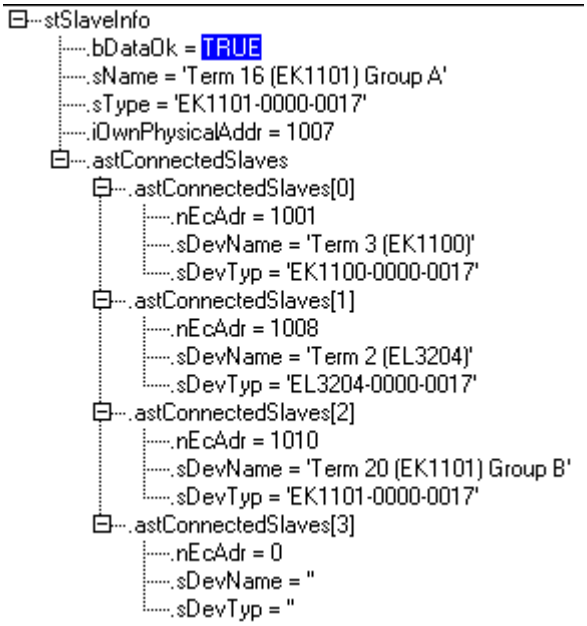


Abb. 72: Aufbau Information

Am Wichtigsten für die Eigenorientierung ist der Vorgänger-Slave an Port A/0.

Über diesen Mechanismus lässt sich auch eine gesamte Topologieermittlung aus der PLC heraus durchführen.

2.3.3 EtherCAT Datenaustausch

Wenn zwischen zwei EtherCAT-Systemen direkter synchroner Datenaustausch erforderlich ist, stehen verschiedenen Komponenten zur Realisierung zur Auswahl. Je nach Anforderungen kann für die Applikation die richtige Methode nach den folgenden Kriterien ausgewählt werden. Unterscheidungsmerkmale sind:

- synchroner Datenaustausch mit vordefinierten, in der Konfiguration festgelegten Prozessdaten
- asynchroner Datenaustausch
- Unterstützung für ADS over EtherCAT (AoE)
- Unterstützung für Synchronisierung der Distributed Clocks (DC) der beiden Systeme untereinander

In der folgenden Tabelle sind einige Charakteristika aufgeführt. Diese Angaben können nur Anhaltspunkte sein, maßgeblich ist die jeweilig online verfügbare Komponentendokumentation!

	EL6692	Publisher/ Subscriber	EL6601	FC1100	CX50x0-B110
Max. synchroner Datenumfang	480 Byte, bidirektional	beliebig	1024 Byte, bidirektional (Publisher/ Subscriber-Verfahren)	1024 Byte, bidirektional	
Max. asynchroner Datenumfang	-	-	beliebig	-	
Unterstützung AoE	ja	ja	-	ja	ja
Unterstützung DC	ja	-	-	-	-
Hinweis	- empfohlen zur Synchronisierung von EtherCAT-Systemen - TwinCAT 2.11 erforderlich	- Verwendung eines Ethernet-Ports in beiden Systemen als RealTime-Device - empfohlen zum synchronen Datenaustausch	- Verlegung des RT-Devices in eine EtehrCAT-Klemme - empfohlen zum synchronen Datenaustausch	- freier PCI Steckplatz im IPC erforderlich - TwinCAT 2.11 R2 erforderlich	- CX5000 als unterlagerte autonome Steuerung mit eigenen IO wird als EtherCAT-Slave in das überlagerte System eingebunden - Option B110 erforderlich - TwinCAT 2.11 R2 erforderlich

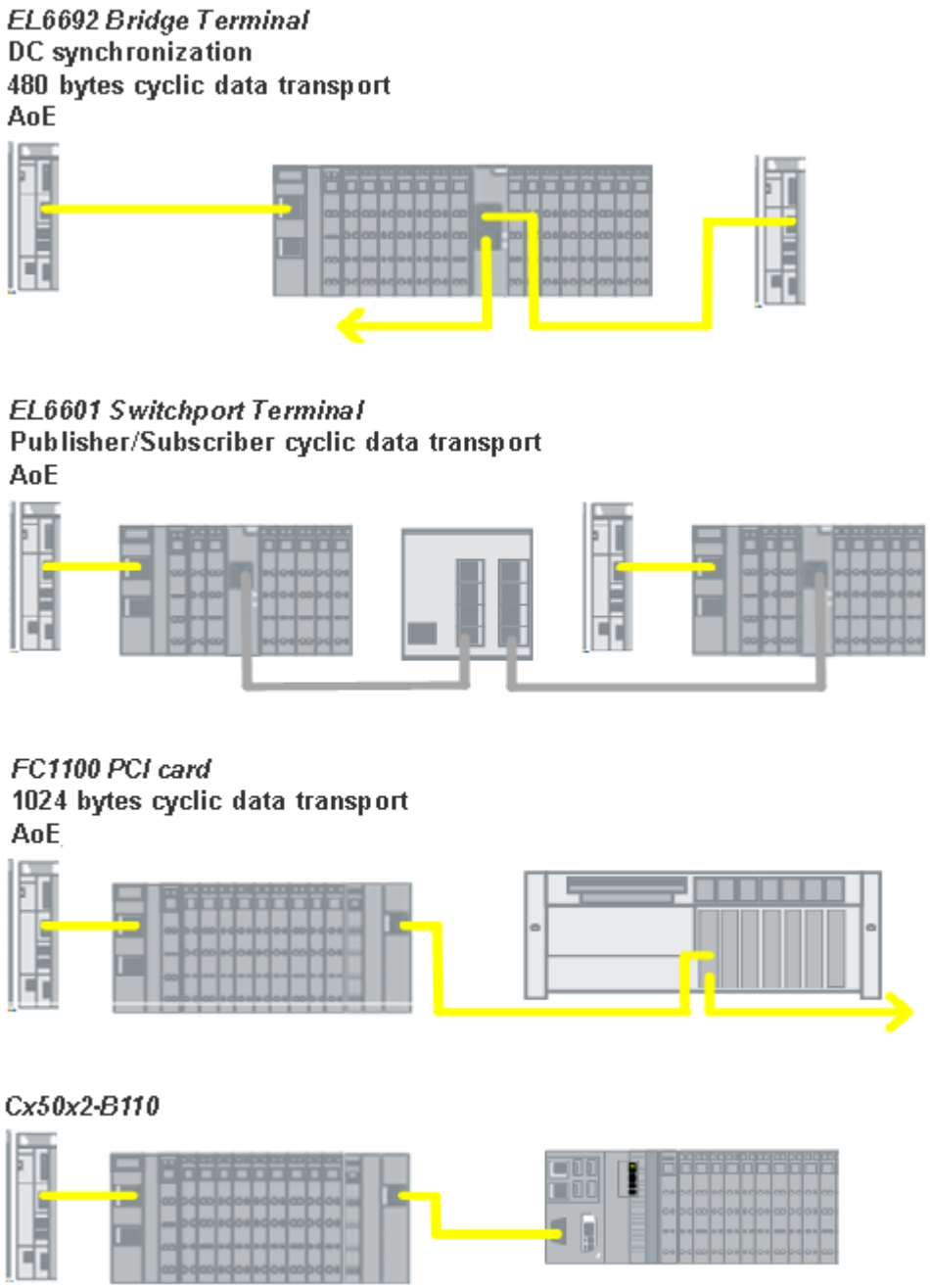


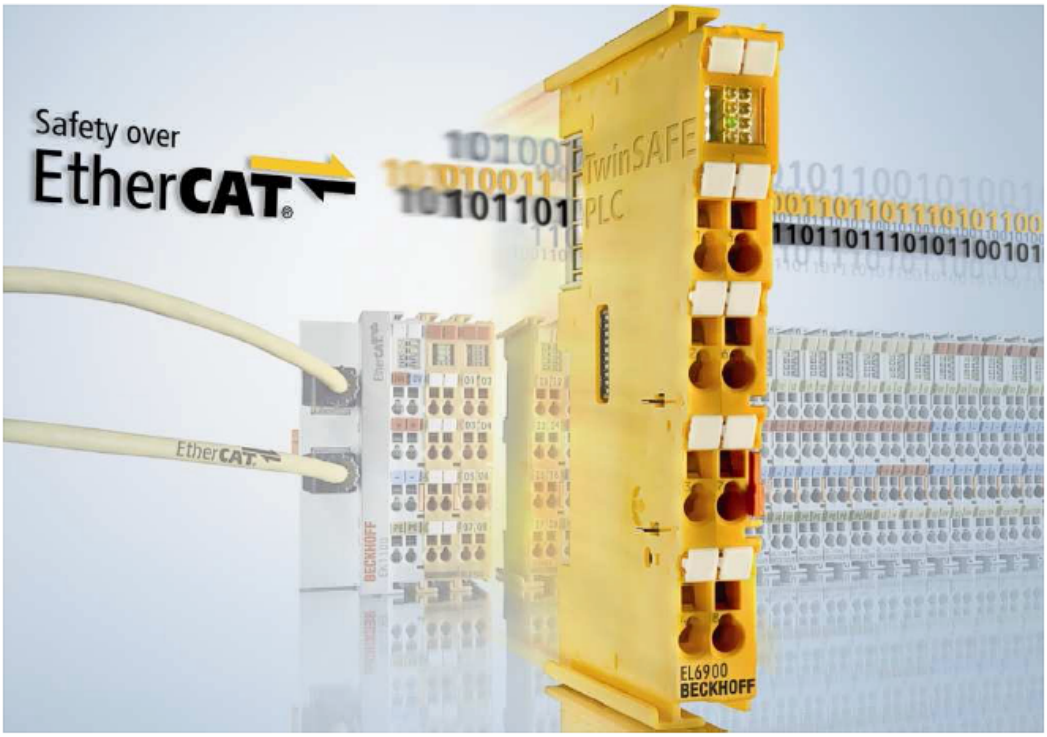
Abb. 73: Topologien verschiedener Datenaustauschverfahren

2.3.4 Safety over EtherCAT - TwinSAFE

Die Beckhoff EtherCAT Safety Produkte arbeiten mit dem TwinSAFE Protokoll, das innerhalb der EtherCAT-Datagramme als Datenanteil transportiert wird. Prinzipiell ist das TwinSAFE-Protokoll unabhängig vom Kommunikationskanal und besitzt eigene Fehlererkennungsmechanismen.

Im EtherCAT-System übernimmt eine Logik-Klemme EL6900 die Bearbeitung der sicherheitsrelevanten Funktionen und überträgt über das TwinSAFE-Protokoll Daten zu den Eingangs- und Ausgangsklemmen (EL1904, EL2904).

Zur Anwendung und Inbetriebnahme sind auf der Beckhoff Webseite im Downloadbereich die entsprechenden Gerätedokumentation und ein TwinSAFE-Applikationshandbuch zugänglich.



Applikationshandbuch

TwinSAFE

Version: 1.1.0
Datum: 22.03.2011

BECKHOFF

Abb. 74: Applikationshandbuch TwinSAFE

3 Einrichtung im TwinCAT Systemmanager

3.1 TwinCAT Quickstart

TwinCAT stellt eine Entwicklungsumgebung für Echtzeitsteuerung mit Multi-SPS-System, NC Achsregelung, Programmierung und Bedienung dar. Das gesamte System wird hierbei durch diese Umgebung abgebildet und ermöglicht Zugriff auf eine Programmierumgebung (inkl. Kompilierung) für die Steuerung. Einzelne digitale oder analoge Eingänge bzw. Ausgänge können auch direkt ausgelesen bzw. beschrieben werden, um diese z.B. hinsichtlich ihrer Funktionsweise zu überprüfen.

Weitere Informationen hierzu erhalten Sie unter http://infosys.beckhoff.de:

- **EtherCAT Systemhandbuch:**
  Feldbuskomponenten → EtherCAT-Klemmen → EtherCAT System Dokumentation → Einrichtung im TwinCAT System Manager
- **TwinCAT 2** → TwinCAT System Manager → E/A- Konfiguration
- Insbesondere zur TwinCAT – Treiberinstallation:
  Feldbuskomponenten → Feldbuskarten und Switche → FC900x – PCI-Karten für Ethernet → Installation

Geräte, d. h. „devices“ beinhalten jeweils die Klemmen der tatsächlich aufgebauten Konfiguration. Dabei gibt es grundlegend die Möglichkeit sämtliche Informationen des Aufbaus über die „Scan“ - Funktion einzubringen („online“) oder über Editorfunktionen direkt einzufügen („offline“):

- **„offline“**: der vorgesehene Aufbau wird durch Hinzufügen und entsprechendes Platzieren einzelner Komponenten erstellt. Diese können aus einem Verzeichnis ausgewählt und Konfiguriert werden.
  - Die Vorgehensweise für den „offline“ – Betrieb ist unter http://infosys.beckhoff.de einsehbar:
    TwinCAT 2 → TwinCAT System Manager → EA - Konfiguration → Anfügen eines E/A-Gerätes
- **„online“**: die bereits physikalisch aufgebaute Konfiguration wird eingelesen
  - Sehen Sie hierzu auch unter http://infosys.beckhoff.de:
    Feldbuskomponenten → Feldbuskarten und Switche → FC900x – PCI-Karten für Ethernet → Installation → Geräte suchen

Vom Anwender –PC bis zu den einzelnen Steuerungselementen ist folgender Zusammenhang vorgesehen:

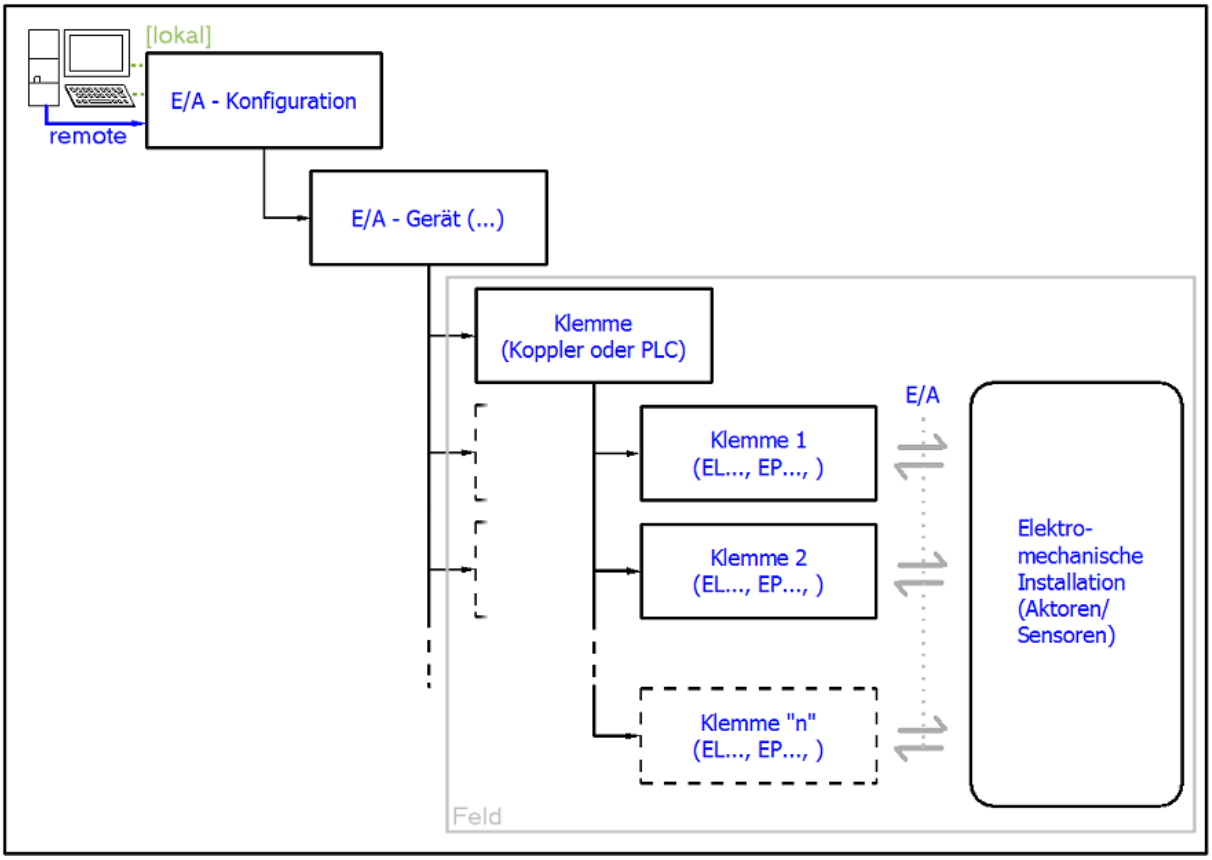


Abb. 75: Bezug von der Anwender Seite (Inbetriebnahme) zur Installation

Das anwenderseitige Einfügen bestimmter Komponenten (E/A – Gerät, Klemme, Box,..) erfolgt bei TwinCAT 2 und TwinCAT 3 auf die gleiche Weise. In den nachfolgenden Beschreibungen wird ausschließlich der „online“ Vorgang angewandt.

Beispielkonfiguration (realer Aufbau)

Ausgehend von der folgenden Beispielkonfiguration wird in den anschließenden Unterkapiteln das Vorgehen für TwinCAT 2 und TwinCAT 3 behandelt:

- Steuerungssystem (PLC) **CX2040** inkl. Netzteil **CX2100-0004**
- Rechtsseitig angebunden am CX2040 (E-Bus):
  EL1004 (4-Kanal-Digital-Eingangsklemme 24 V_{DC})
- Über den X001 Anschluss (RJ-45) angeschlossen: **EK1100** EtherCAT-Koppler
- Rechtsseitig angebunden am EK1100 EtherCAT-Koppler (E-Bus):
  EL2008 (8-Kanal-Digital-Ausgangsklemme 24 V_{DC}; 0,5 A)
- (Optional über X000: ein Link zu einen externen PC für die Benutzeroberfläche)

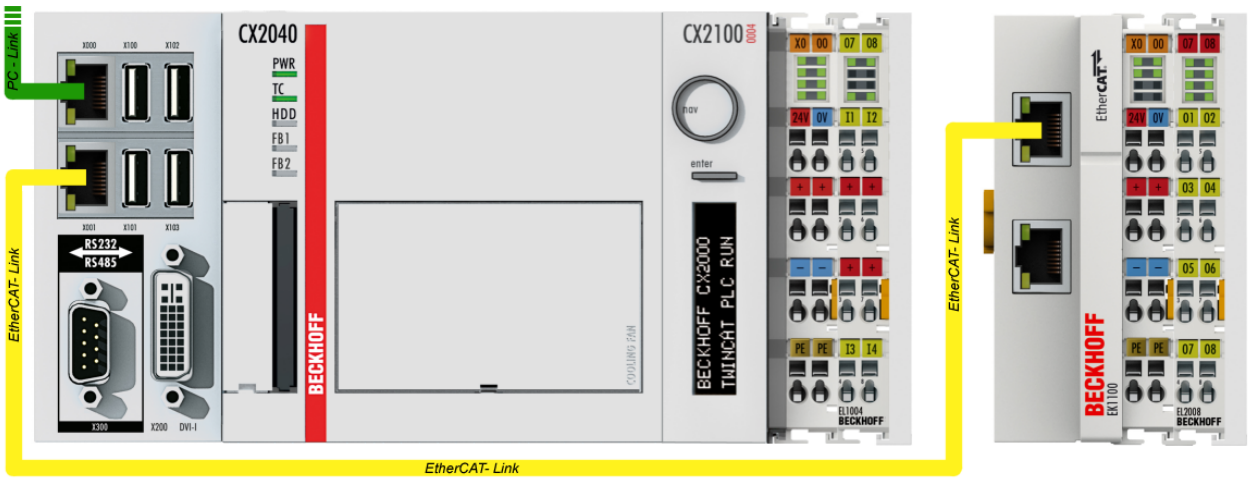


Abb. 76: Aufbau der Steuerung mit Embedded-PC, Eingabe (EL1004) und Ausgabe (EL2008)

Anzumerken ist, dass sämtliche Kombinationen einer Konfiguration möglich sind; beispielsweise könnte die Klemme EL1004 ebenso auch nach dem Koppler angesteckt werden oder die Klemme EL2008 könnte zusätzlich rechts an dem CX2040 angesteckt sein – dann wäre der Koppler EK1100 überflüssig.

3.1.1 TwinCAT 2

Startup

TwinCAT 2 verwendet grundlegend zwei Benutzeroberflächen: den „TwinCAT System Manager“ zur Kommunikation mit den elektromechanischen Komponenten und „TwinCAT PLC Control“ für die Erstellung und Kompilierung einer Steuerung. Begonnen wird zunächst mit der Anwendung des „TwinCAT System Manager“.

Nach erfolgreicher Installation des TwinCAT-Systems auf den Anwender PC der zur Entwicklung verwendet werden soll, zeigt der TwinCAT 2 (System Manager) folgende Benutzeroberfläche nach dem Start:

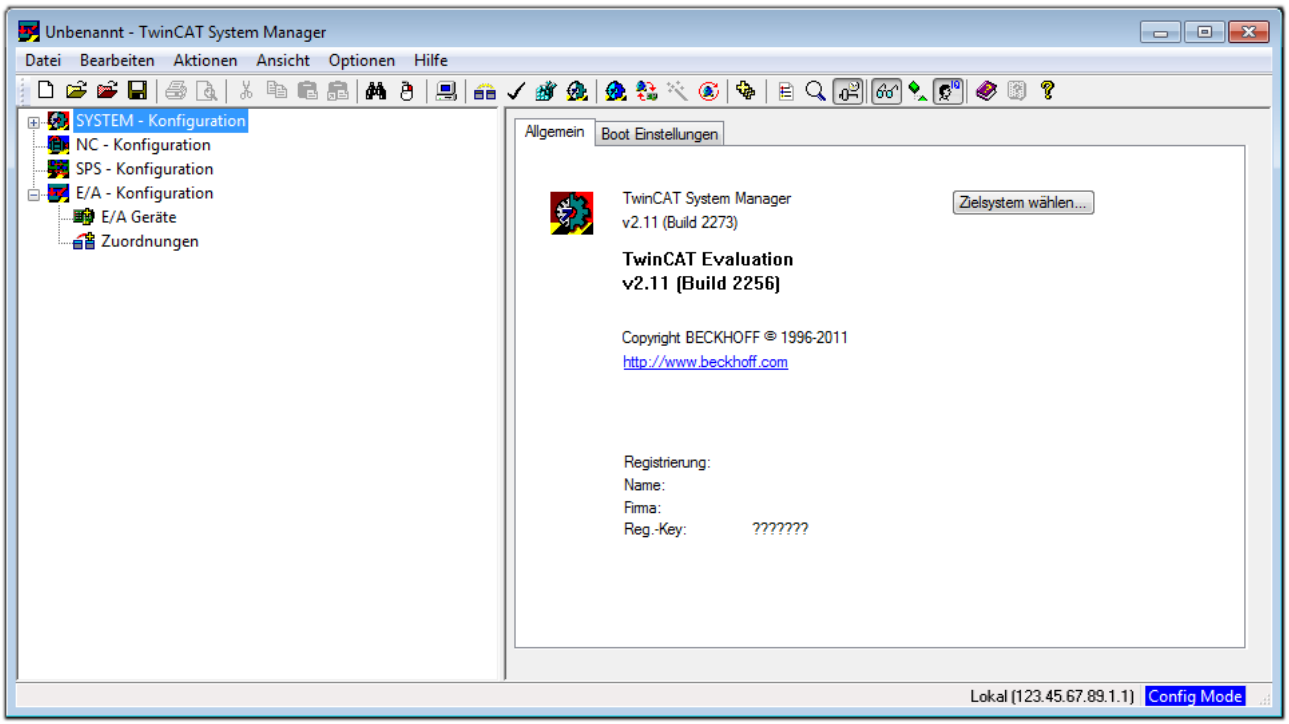


Abb. 77: Initiale Benutzeroberfläche TwinCAT 2

Es besteht generell die Möglichkeit das TwinCAT „lokal“ oder per „remote“ zu verwenden. Ist das TwinCAT System inkl. Benutzeroberfläche (Standard) auf dem betreffenden PLC installiert, kann TwinCAT „lokal“ eingesetzt werden und mit Schritt „Geräte einfügen [▶ 68]“ fortgesetzt werden.

Ist es vorgesehen, die auf einem PLC installierte TwinCAT Laufzeitumgebung von einem anderen System als Entwicklungsumgebung per „remote“ anzusprechen, ist das Zielsystem zuvor bekannt zu machen. Im Menü unter „Aktionen“ → „Auswahl des Zielsystems...“, über das Symbol „  “ oder durch Taste „F8“ wird folgendes Fenster hierzu geöffnet:

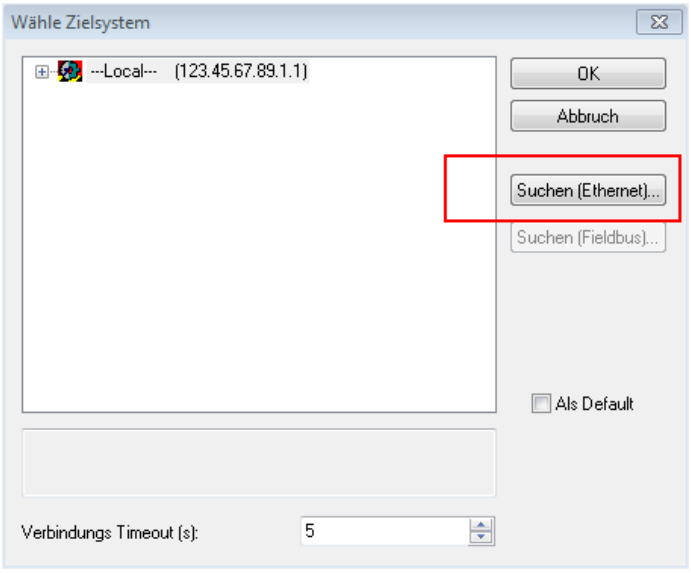


Abb. 78: Wähle Zielsystem

Mittels „Suchen (Ethernet)...“ wird das Zielsystem eingetragen. Dadurch wird ein weiterer Dialog geöffnet um hier entweder:

- den bekannten Rechnernamen hinter „Enter Host Name / IP:“ einzutragen (wie rot gekennzeichnet)
- einen „Broadcast Search“ durchzuführen (falls der Rechnername nicht genau bekannt)
- die bekannte Rechner - IP oder AmsNetId einzutragen

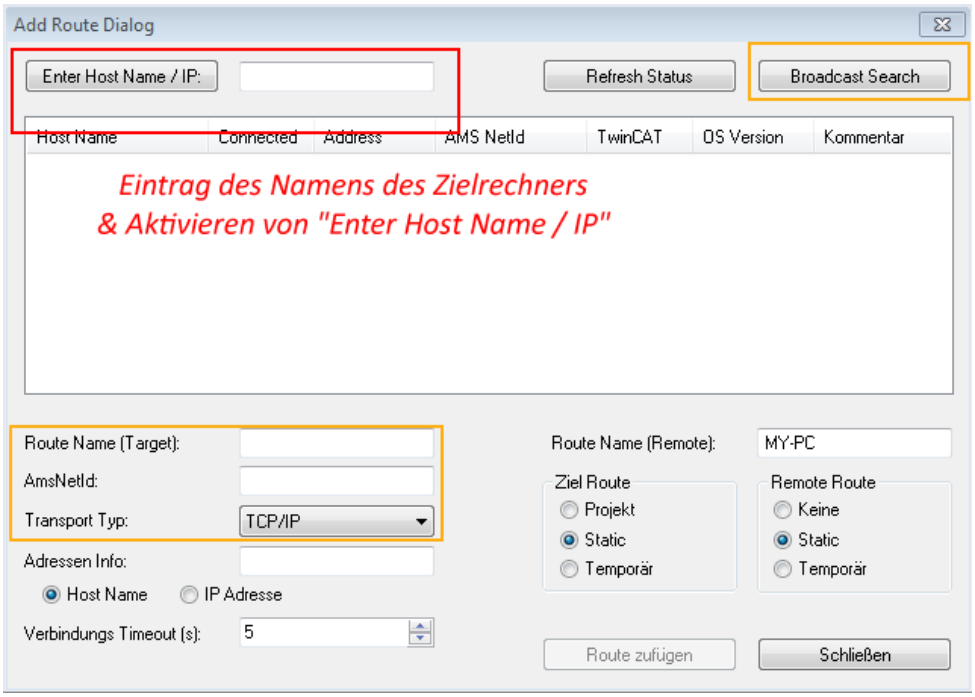
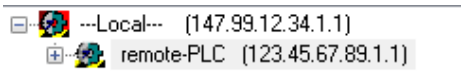


Abb. 79: PLC für den Zugriff des TwinCAT System Managers festlegen: Auswahl des Zielsystems

Ist das Zielsystem eingetragen steht dieses wie folgt zur Auswahl (ggf. muss zuvor das korrekte Passwort eingetragen werden):



Nach der Auswahl mit „OK“ ist das Zielsystem über den System Manager ansprechbar.

Geräte einfügen

In dem linksseitigen Konfigurationsbaum der TwinCAT 2 – Benutzeroberfläche des System Managers wird „E/A Geräte“ selektiert und sodann entweder über Rechtsklick ein Kontextmenü geöffnet und „Geräte Suchen…“ ausgewählt oder in der Menüleiste mit  die Aktion gestartet. Ggf. ist zuvor der TwinCAT System Manager in den „Konfig Modus“ mittels  oder über das Menü „Aktionen“ → „Startet/ Restarten von TwinCAT in Konfig-Modus“(Shift + F4) zu versetzen.

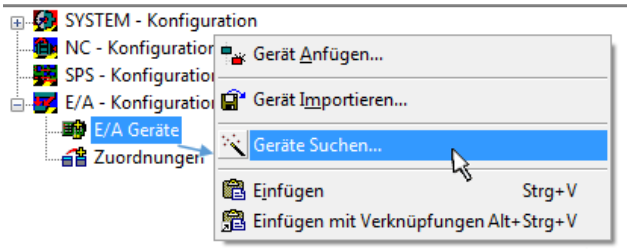


Abb. 80: Auswahl „Gerät Suchen..“

Die darauf folgende Hinweismeldung ist zu bestätigen und in dem Dialog die Geräte „EtherCAT“ zu wählen:

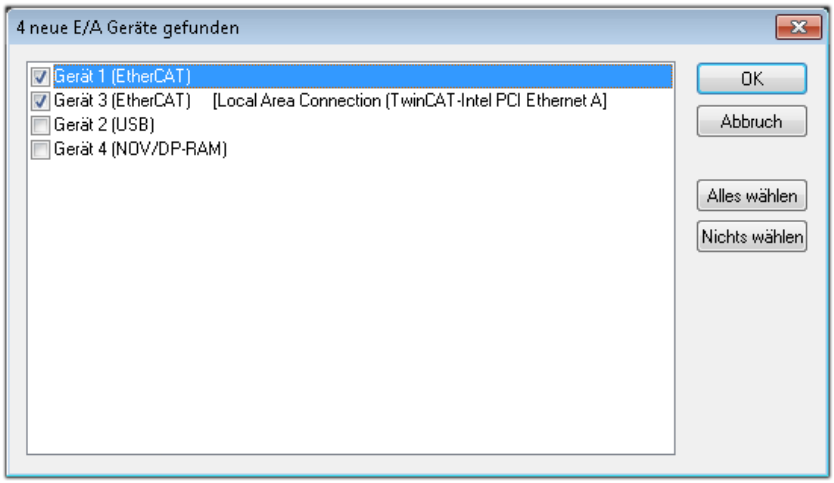


Abb. 81: Automatische Erkennung von E/A Geräten: Auswahl der einzubindenden Geräte

Ebenfalls ist anschließend die Meldung „nach neuen Boxen suchen“ zu bestätigen, um die an den Geräten angebundenen Klemmen zu ermitteln. „Free Run“ erlaubt das Manipulieren von Ein- und Ausgangswerten innerhalb des „Config Modus“ und sollte ebenfalls bestätigt werden.

Ausgehend von der am Anfang dieses Kapitels beschriebenen Beispielkonfiguration [▶ 64] sieht das Ergebnis wie folgt aus:

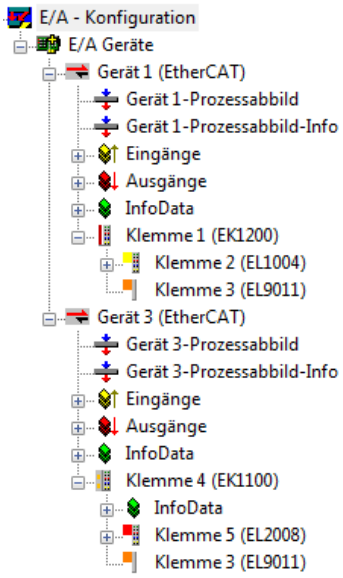


Abb. 82: Abbildung der Konfiguration im TwinCAT 2 System Manager

Der gesamte Vorgang setzt sich aus zwei Stufen zusammen, die auch separat ausgeführt werden können (erst das Ermitteln der Geräte, dann das Ermitteln der daran befindlichen Elemente wie Boxen, Klemmen o. ä.). So kann auch durch Markierung von „Gerät ..“ aus dem Kontextmenü eine „Suche“ Funktion (Scan) ausgeführt werden, die hierbei dann lediglich die darunter liegenden (im Aufbau vorliegenden) Elemente einliest:

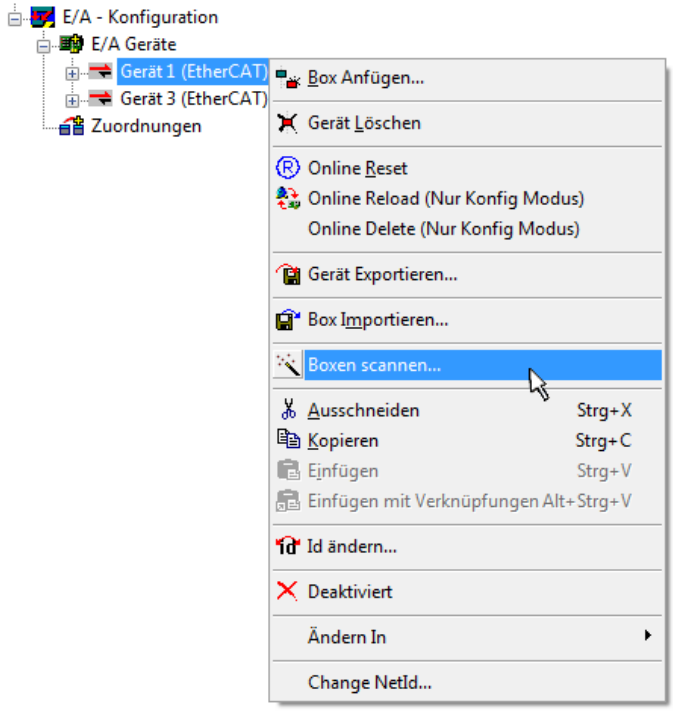


Abb. 83: Einlesen von einzelnen an einem Gerät befindlichen Klemmen

Diese Funktionalität ist nützlich, falls die Konfiguration (d. h. der „reale Aufbau“) kurzfristig geändert wird.

PLC programmieren und integrieren

TwinCAT PLC Control ist die Entwicklungsumgebung zur Erstellung der Steuerung in unterschiedlichen Programmumgebungen: Das TwinCAT PLC Control unterstützt alle in der IEC 61131-3 beschriebenen Sprachen. Es gibt zwei textuelle Sprachen und drei grafische Sprachen.

- **Textuelle Sprachen**
  - Anweisungsliste (AWL, IL)

- Strukturierter Text (ST)

- **Grafische Sprachen**
  - Funktionsplan (FUP, FBD)
  - Kontaktplan (KOP, LD)
  - Freigrafischer Funktionsplaneditor (CFC)
  - Ablaufsprache (AS, SFC)

Für die folgenden Betrachtungen wird lediglich vom strukturierten Text (ST) Gebrauch gemacht.

Nach dem Start von TwinCAT PLC Control wird folgende Benutzeroberfläche für ein initiales Projekt dargestellt:

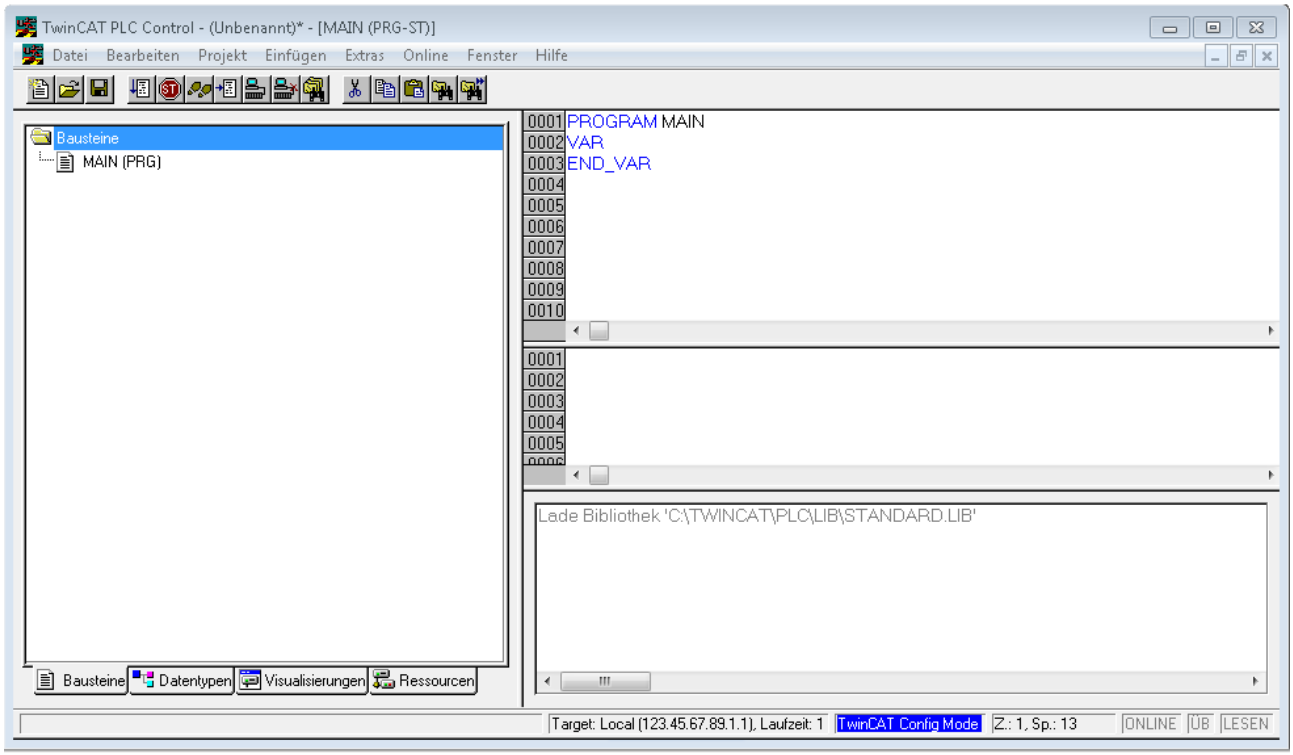


Abb. 84: TwinCAT PLC Control nach dem Start

Nun sind für den weiteren Ablauf Beispielvariablen sowie ein Beispielprogramm erstellt und unter dem Namen „PLC_example.pro“ gespeichert worden:

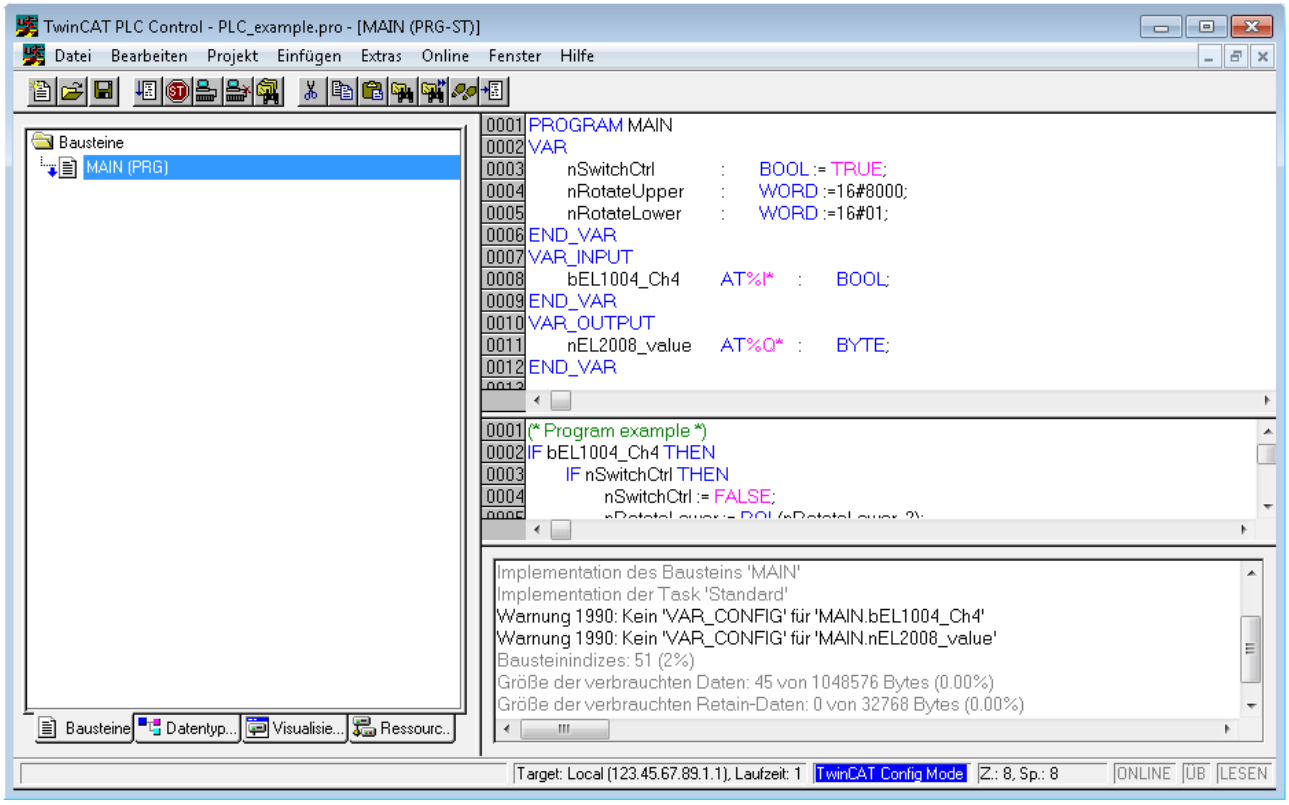


Abb. 85: Beispielprogramm mit Variablen nach einem Kompiliervorgang (ohne Variablenanbindung)

Die Warnung 1990 (fehlende „VAR_CONFIG“) nach einem Kompiliervorgang zeigt auf, dass die als extern definierten Variablen (mit der Kennzeichnung „AT%I*“ bzw. „AT%Q*“) nicht zugeordnet sind. Das TwinCAT PLC Control erzeugt nach erfolgreichen Kompiliervorgang eine „*.tpy“ Datei in dem Verzeichnis in dem das Projekt gespeichert wurde. Diese Datei („*.tpy“) enthält u.a. Variablenzuordnungen und ist dem System Manager nicht bekannt, was zu dieser Warnung führt. Nach dessen Bekanntgabe kommt es nicht mehr zu dieser Warnung.

Im System Manager ist das Projekt des TwinCAT PLC Control zunächst einzubinden. Dies geschieht über das Kontext Menü der „SPS- Konfiguration“ (rechts-Klick) und der Auswahl „SPS Projekt Anfügen…“:

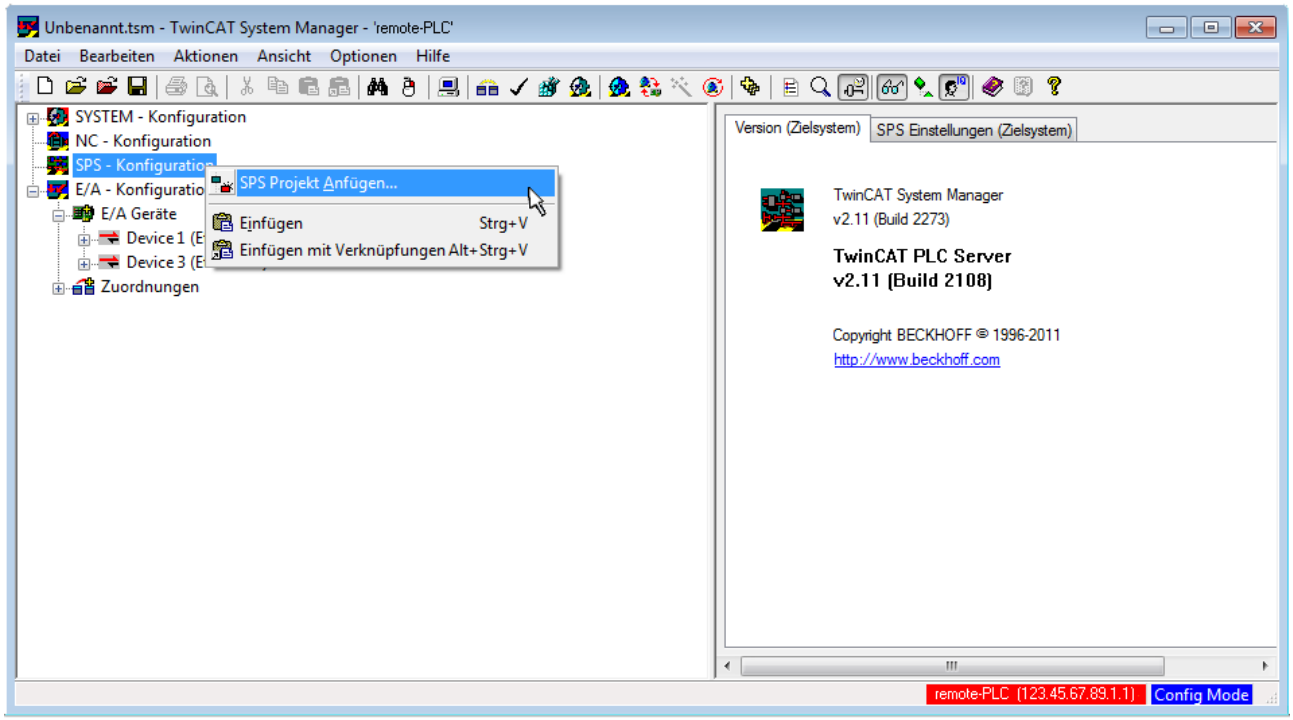


Abb. 86: Hinzufügen des Projektes des TwinCAT PLC Control

Über ein dadurch geöffnetes Browserfenster wird die PLC- Konfiguration „PLC_example.tpy“ ausgewählt. Dann ist in dem Konfigurationsbaum des System Manager das Projekt inklusive der beiden „AT“ – gekennzeichneten Variablen eingebunden:

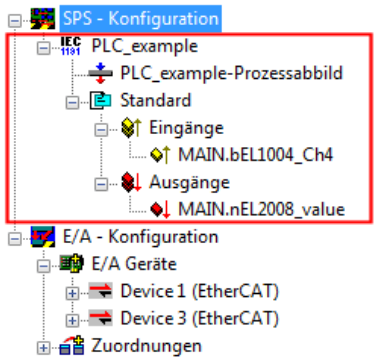


Abb. 87: Eingebundenes PLC Projekt in der SPS- Konfiguration des System Managers

Die beiden Variablen „bEL1004_Ch4“ sowie „nEL2008_value“ können nun bestimmten Prozessobjekten der E/A - Konfiguration zugeordnet werden.

Variablen Zuordnen

Über das Kontextmenü einer Variable des eingebundenen Projekts „PLC_example“ unter „Standard“ wird mittels „Verknüpfung Ändern…“ ein Fenster zur Auswahl eines passenden Prozessobjektes (PDOs) geöffnet:

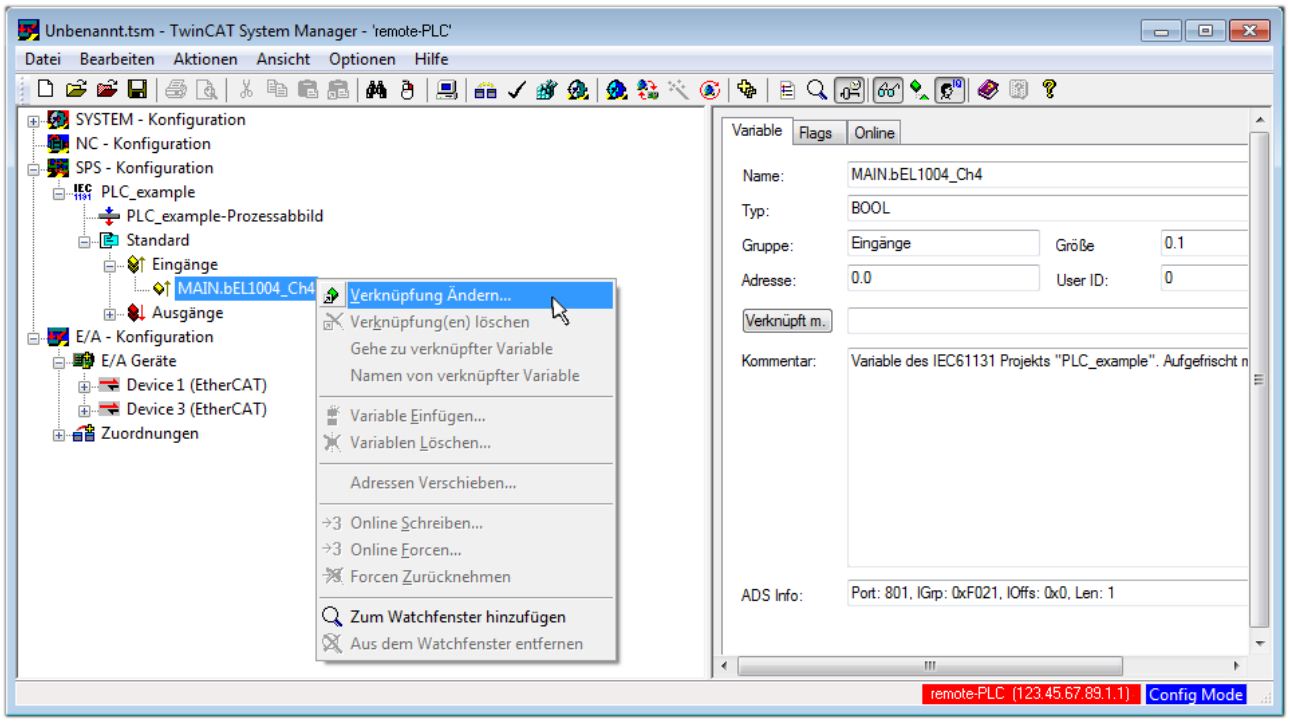


Abb. 88: Erstellen der Verknüpfungen PLC-Variablen zu Prozessobjekten

In dem dadurch geöffneten Fenster kann aus dem SPS-Konfigurationsbaum das Prozessobjekt für die Variable „bEL1004_Ch4“ vom Typ BOOL selektiert werden:

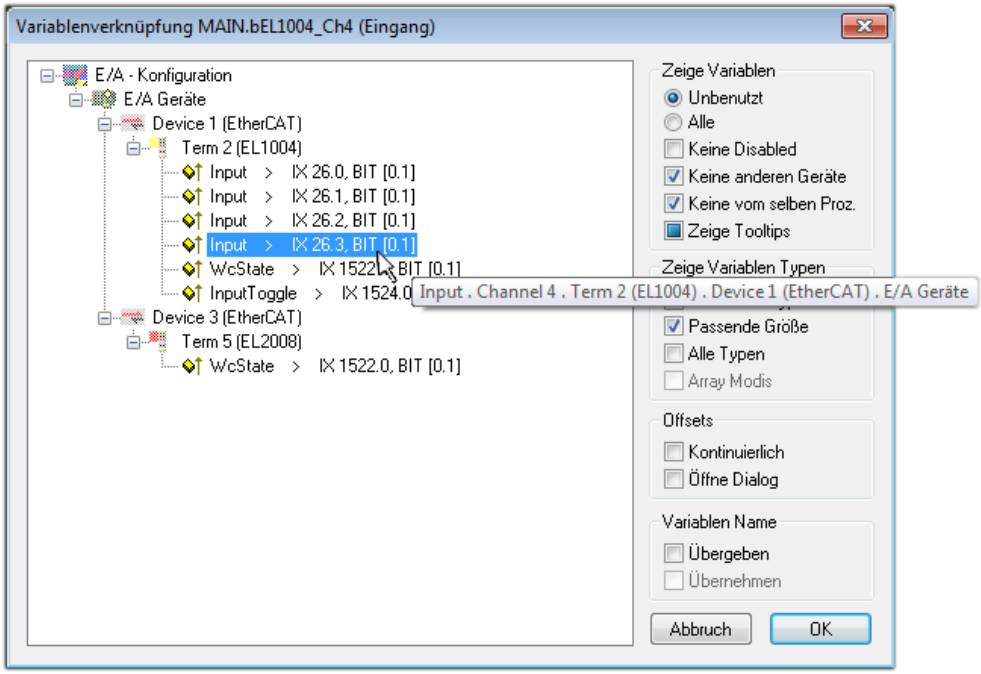


Abb. 89: Auswahl des PDO vom Typ BOOL

Entsprechend der Standarteinstellungen stehen nur bestimmte PDO Objekte zur Auswahl zur Verfügung. In diesem Beispiel wird von der Klemme EL1004 der Eingang von Kanal 4 zur Verknüpfung ausgewählt. Im Gegensatz hierzu muss für das Erstellen der Verknüpfung der Ausgangsvariablen die Checkbox „Alle Typen“ aktiviert werden, um in diesem Fall eine Byte-Variable einen Satz von acht separaten Ausgangsbits zuzuordnen. Die folgende Abbildung zeigt den gesamten Vorgang:

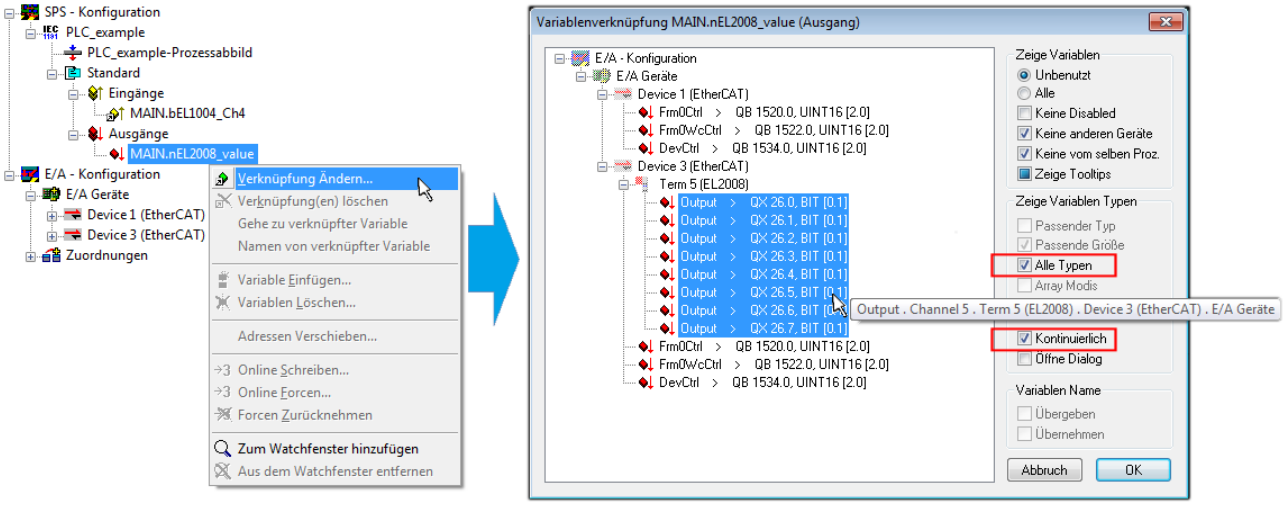


Abb. 90: Auswahl von mehreren PDO gleichzeitig: Aktivierung von „Kontinuierlich“ und „Alle Typen“

Zu sehen ist, dass überdies die Checkbox „Kontinuierlich“ aktiviert wurde. Dies ist dafür vorgesehen, dass die in dem Byte der Variablen „nEL2008_value“ enthaltenen Bits allen acht ausgewählten Ausgangsbits der Klemme EL2008 der Reihenfolge nach zugeordnet werden sollen. Damit ist es möglich, alle acht Ausgänge der Klemme mit einem Byte entsprechend Bit 0 für Kanal 1 bis Bit 7 für Kanal 8 von der PLC im Programm später anzusprechen. Ein spezielles Symbol ( ) an dem gelben bzw. roten Objekt der Variablen zeigt an, dass hierfür eine Verknüpfung existiert. Die Verknüpfungen können z. B. auch überprüft werden, indem „Goto Link Variable“ aus dem Kontextmenü einer Variable ausgewählt wird. Dann wird automatisch das gegenüberliegende verknüpfte Objekt, in diesem Fall das PDO selektiert:

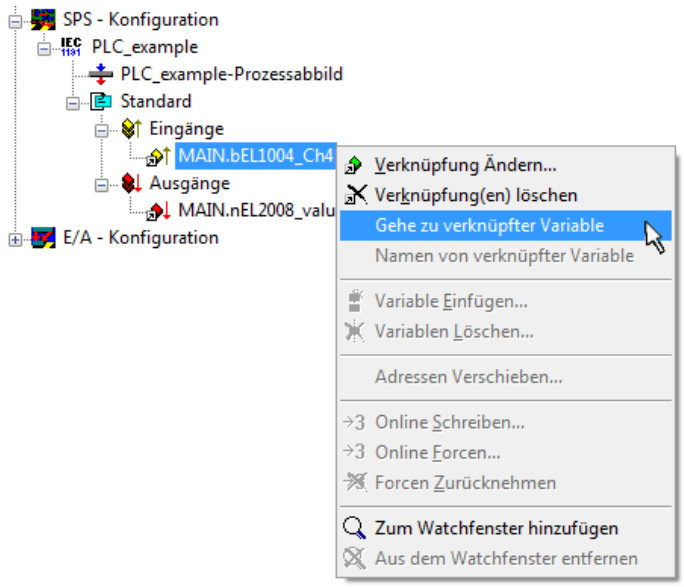
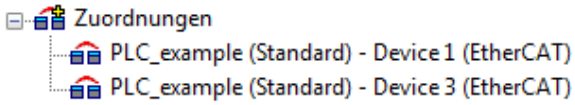


Abb. 91: Anwendung von „Goto Link Variable“ am Beispiel von „MAIN.bEL1004_Ch4“

Anschließend wird mittels Menüauswahl „Aktionen“ → „Zuordnung erzeugen…“ oder über  der Vorgang des Zuordnens von Variablen zu PDO abgeschlossen.

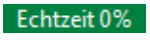
Dies lässt sich entsprechend in der Konfiguration einsehen:



Der Vorgang zur Erstellung von Verknüpfungen kann auch in umgekehrter Richtung, d. h. von einzelnen PDO ausgehend zu einer Variablen erfolgen. In diesem Beispiel wäre dann allerdings eine komplette Auswahl aller Ausgangsbits der EL2008 nicht möglich, da die Klemme nur einzelne digitale Ausgänge zur Verfügung stellt. Hat eine Klemme einen Byte, Word, Integer oder ein ähnliches PDO, so ist es möglich dies wiederum einen Satz von bit-typisierten Variablen (Typ „BOOL“) zuzuordnen. Auch hier kann ebenso in die andere Richtung ein „Goto Link Variable“ ausgeführt werden, um dann die betreffende Instanz der PLC zu selektieren.

Aktivieren der Konfiguration

Die Zuordnung von PDO zu PLC Variablen hat nun die Verbindung von der Steuerung zu den Ein- und Ausgängen der Klemmen hergestellt. Nun kann die Konfiguration aktiviert werden. Zuvor kann mittels  (oder über „Aktionen“ → „Konfiguration überprüfen…“) die Konfiguration überprüft werden. Falls kein Fehler vorliegt, kann mit  (oder über „Aktionen“ → „Aktiviert Konfiguration…“) die Konfiguration aktiviert werden, um dadurch Einstellungen im System Manger auf das Laufzeitsystem zu übertragen. Die darauf folgenden Meldungen „Alte Konfigurationen werden überschrieben!“ sowie „Neustart TwinCAT System in Run Modus“ werden jeweils mit „OK“ bestätigt.

Einige Sekunden später wird der Realtime Status 

unten rechts im System Manager angezeigt. Das PLC System kann daraufhin wie im Folgenden beschrieben gestartet werden.

Starten der Steuerung

Ausgehend von einem remote System muss nun als erstes auch die PLC Steuerung über „Online“ → „Choose Run-Time System…“ mit dem embedded PC über Ethernet verbunden werden:

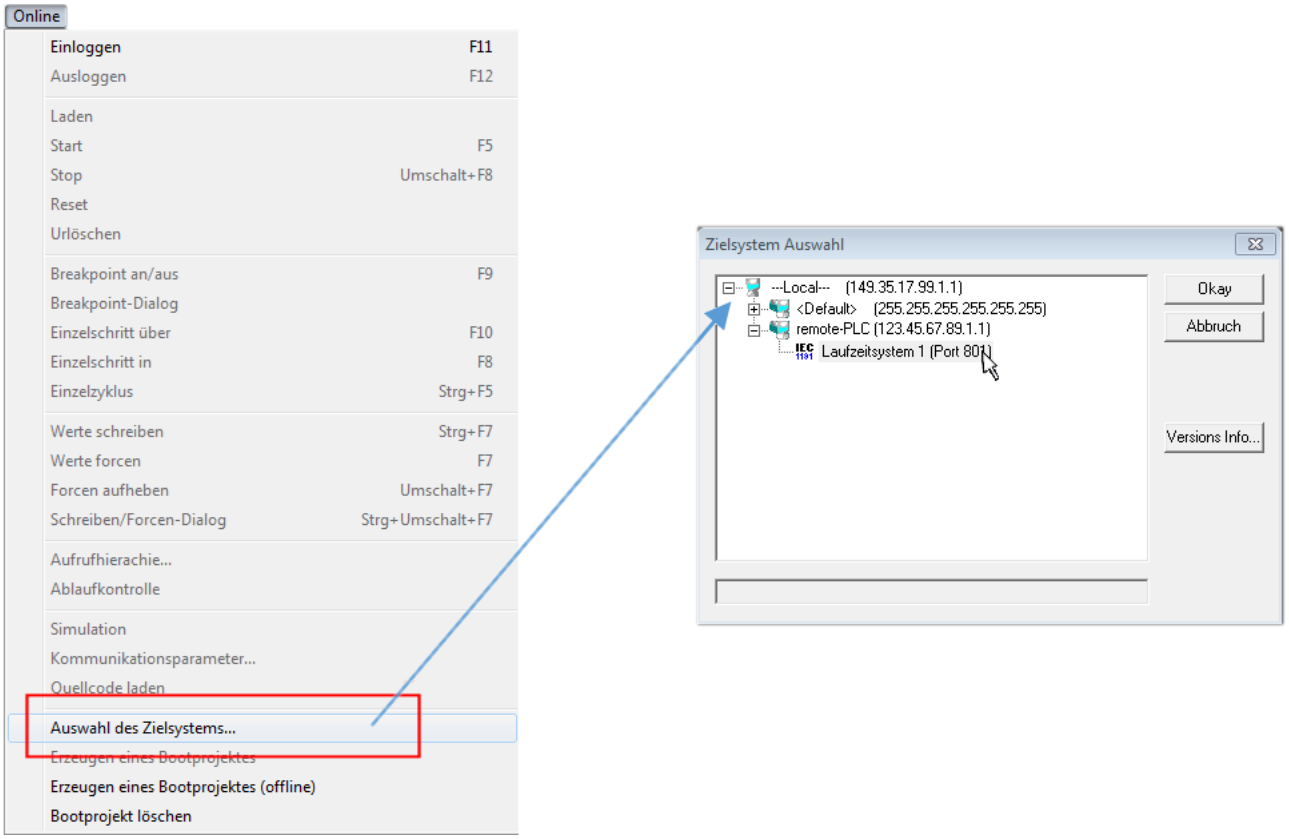


Abb. 92: Auswahl des Zielsystems (remote)

In diesem Beispiel wird das „Laufzeitsystem 1 (Port 801)“ ausgewählt und bestätigt. Mittels Menüauswahl

„Online“ → „Login“, Taste F11 oder per Klick auf  wird auch die PLC mit dem Echtzeitsystem verbunden und nachfolgend das Steuerprogramm geladen, um es ausführen lassen zu können. Dies wird entsprechend mit der Meldung „Kein Programm auf der Steuerung! Soll das neue Programm geladen werden?“ bekannt gemacht und ist mit „Ja“ zu beantworten. Die Laufzeitumgebung ist bereit zum Programstart:

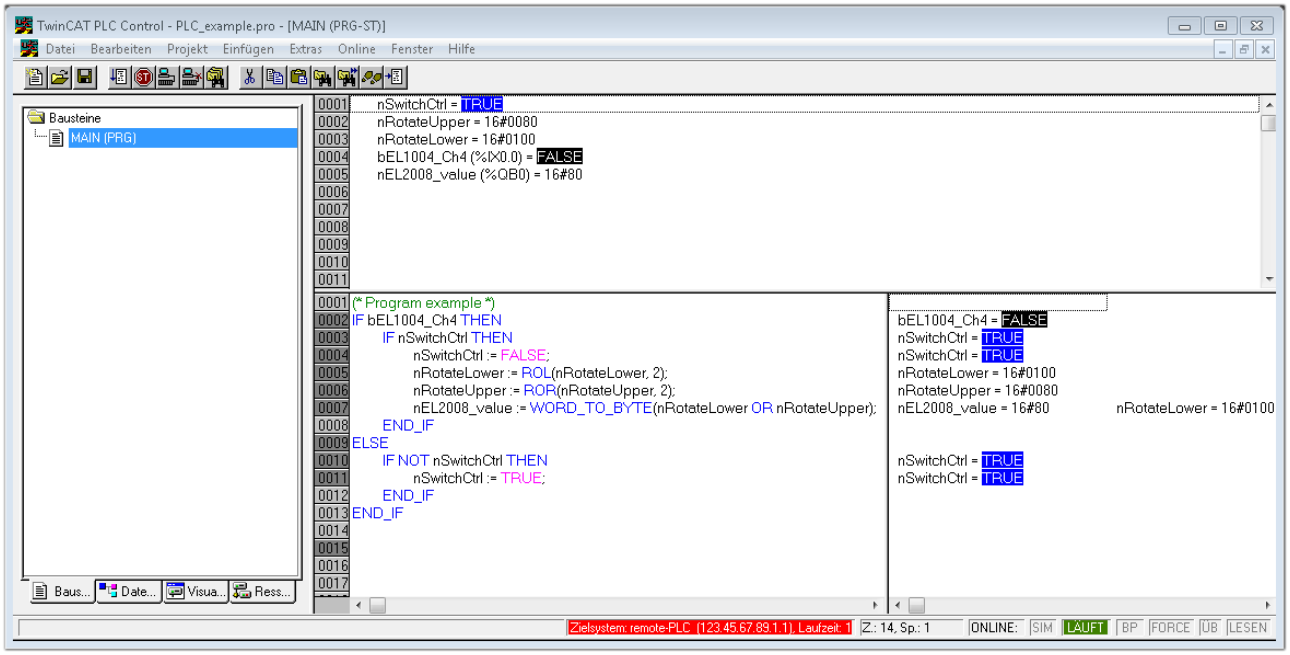


Abb. 93: PLC Control Logged-in, bereit zum Programmstart

Über „Online“ → „Run“, Taste F5 oder  kann nun die PLC gestartet werden.

3.1.2 TwinCAT 3

Startup

TwinCAT 3 stellt die Bereiche der Entwicklungsumgebung durch das Microsoft Visual-Studio gemeinsam zur Verfügung: in den allgemeinen Fensterbereich erscheint nach dem Start linksseitig der Projektmappen-Explorer (vgl. „TwinCAT System Manager“ von TwinCAT 2) zur Kommunikation mit den elektromechanischen Komponenten.

Nach erfolgreicher Installation des TwinCAT-Systems auf den Anwender PC der zur Entwicklung verwendet werden soll, zeigt der TwinCAT 3 (Shell) folgende Benutzeroberfläche nach dem Start:

Abb. 94: Initale Benutzeroberfläche TwinCAT 3

Zunächst ist die Erstellung eines neues Projekt mittels 

(oder unter „Datei“→“Neu“→“Projekt…“) vorzunehmen. In dem darauf folgenden Dialog werden die entsprechenden Einträge vorgenommen (wie in der Abbildung gezeigt):

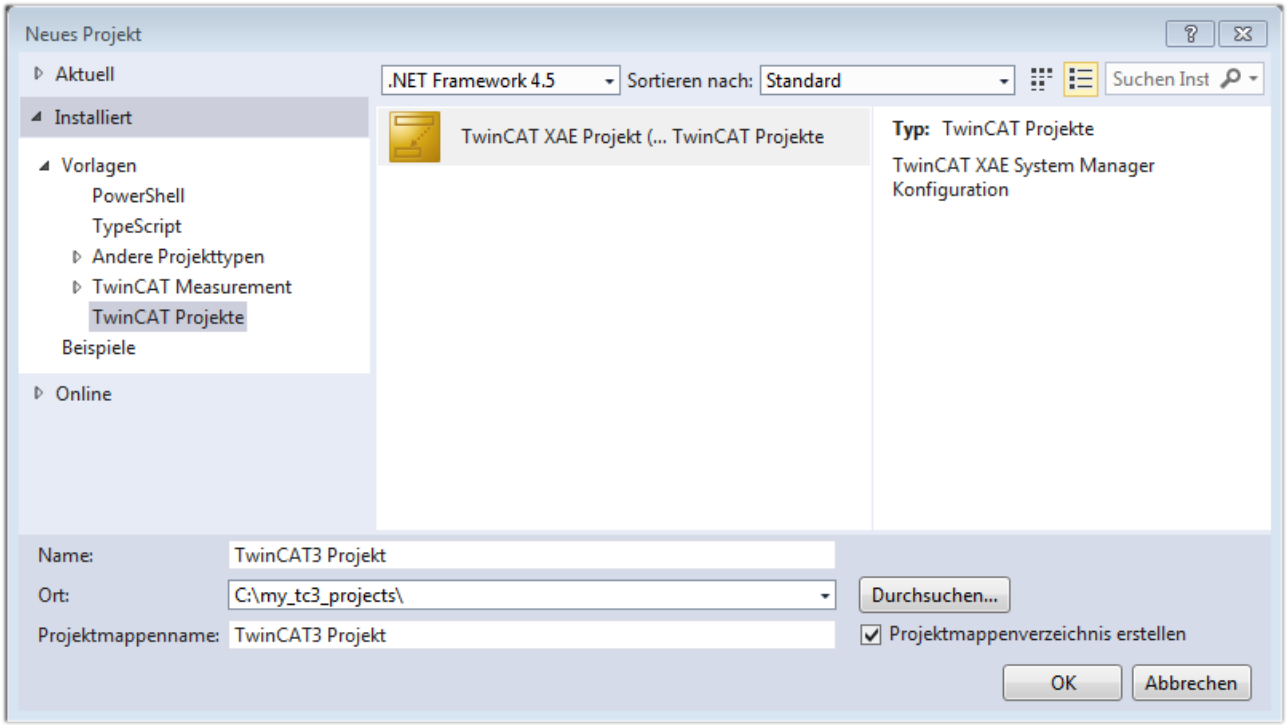


Abb. 95: Neues TwinCAT 3 Projekt erstellen

Im Projektmappen-Explorer liegt sodann das neue Projekt vor:

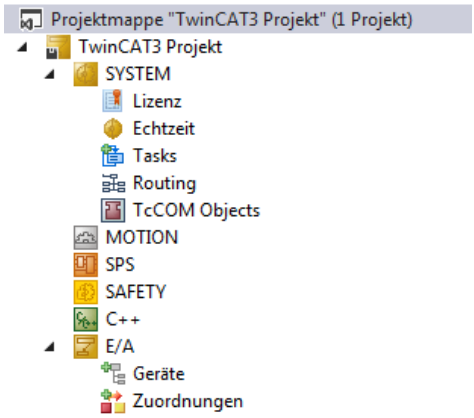
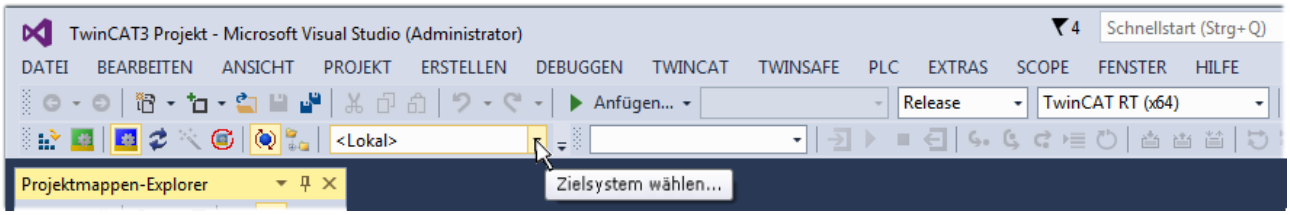


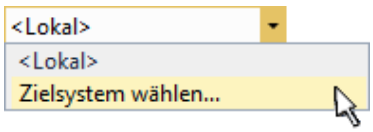
Abb. 96: Neues TwinCAT 3 Projekt im Projektmappen-Explorer

Es besteht generell die Möglichkeit das TwinCAT „lokal“ oder per „remote“ zu verwenden. Ist das TwinCAT System inkl. Benutzeroberfläche (Standard) auf dem betreffenden PLC (lokal) installiert, kann TwinCAT „lokal“ eingesetzt werden und mit Schritt „Geräte einfügen [▶ 79]“ fortgesetzt werden.

Ist es vorgesehen, die auf einem PLC installierte TwinCAT Laufzeitumgebung von einem anderen System als Entwicklungsumgebung per „remote“ anzusprechen, ist das Zielsystem zuvor bekannt zu machen. Über das Symbol in der Menüleiste:



wird das pull-down Menü aufgeklappt:



und folgendes Fenster hierzu geöffnet:

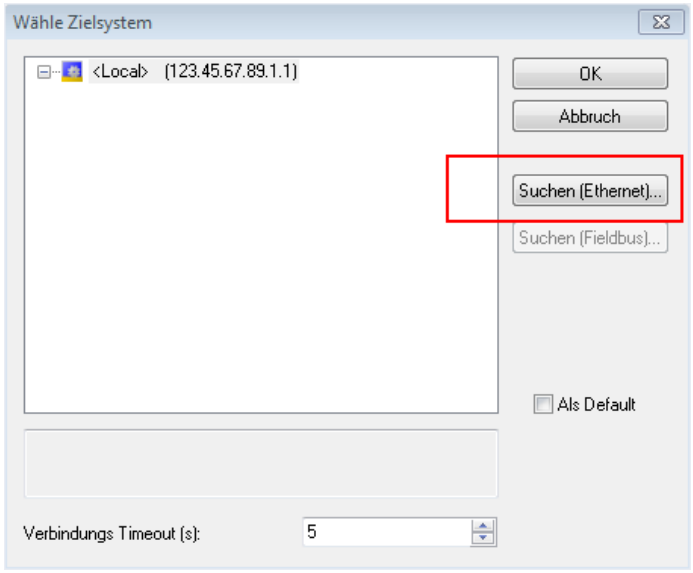


Abb. 97: Auswahldialog: Wähle Zielsystem

Mittels „Suchen (Ethernet)...“ wird das Zielsystem eingetragen. Dadurch wird ein weiterer Dialog geöffnet um hier entweder:

- den bekannten Rechnernamen hinter „Enter Host Name / IP:“ einzutragen (wie rot gekennzeichnet)
- einen „Broadcast Search“ durchzuführen (falls der Rechnername nicht genau bekannt)
- die bekannte Rechner - IP oder AmsNetId einzutragen

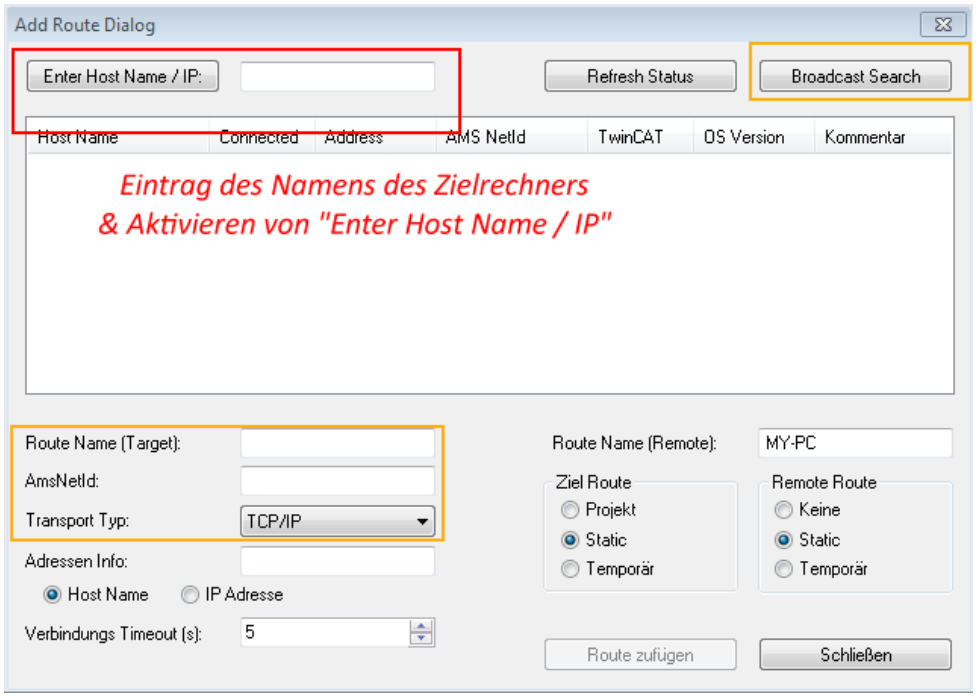
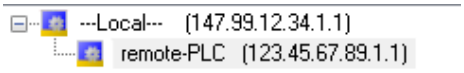


Abb. 98: PLC für den Zugriff des TwinCAT System Managers festlegen: Auswahl des Zielsystems

Ist das Zielsystem eingetragen, steht dieses wie folgt zur Auswahl (ggf. muss zuvor das korrekte Passwort eingetragen werden):



Nach der Auswahl mit „OK“ ist das Zielsystem über das Visual Studio Shell ansprechbar.

Geräte einfügen

In dem linksseitigen Projektmappen-Explorer der Benutzeroberfläche des Visual Studio Shell wird innerhalb des Elementes „E/A“ befindliche „Geräte“ selektiert und sodann entweder über Rechtsklick ein Kontextmenü geöffnet und „Scan“ ausgewählt oder in der Menüleiste mit  die Aktion gestartet. Ggf. ist zuvor der TwinCAT System Manager in den „Konfig Modus“ mittels  oder über das Menü „TWINCAT“ → „Restart TwinCAT (Config Mode)“ zu versetzen.

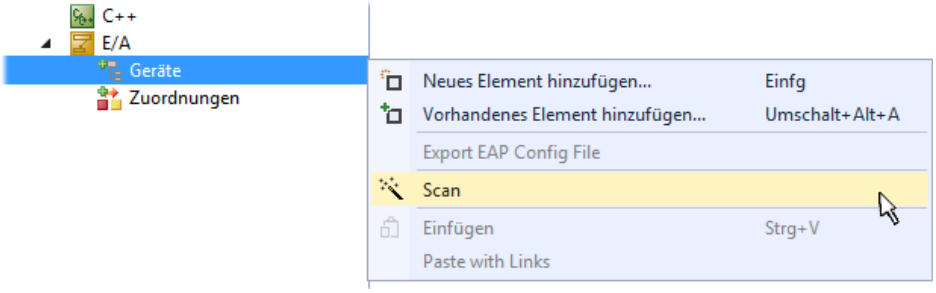


Abb. 99: Auswahl „Scan“

Die darauf folgende Hinweismeldung ist zu bestätigen und in dem Dialog die Geräte „EtherCAT“ zu wählen:

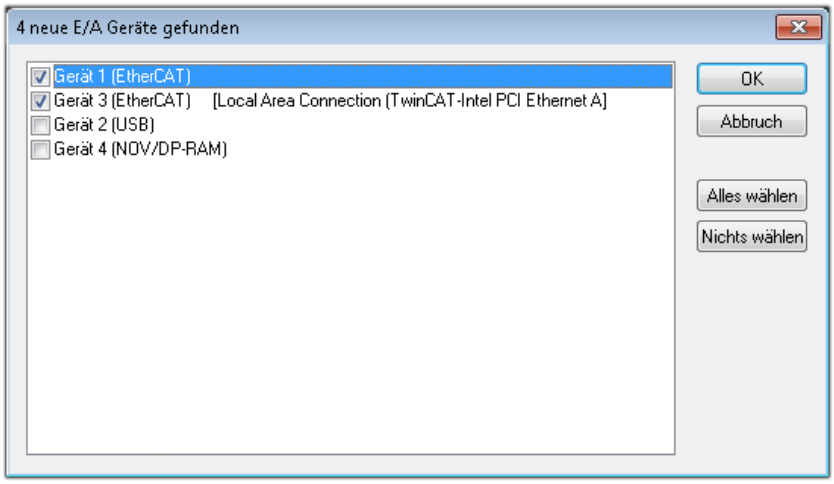


Abb. 100: Automatische Erkennung von E/A Geräten: Auswahl der einzubindenden Geräte

Ebenfalls ist anschließend die Meldung „nach neuen Boxen suchen“ zu bestätigen, um die an den Geräten angebundenen Klemmen zu ermitteln. „Free Run“ erlaubt das Manipulieren von Ein- und Ausgangswerten innerhalb des „Config Modus“ und sollte ebenfalls bestätigt werden.

Ausgehend von der am Anfang dieses Kapitels beschriebenen Beispielkonfiguration [▶ 64] sieht das Ergebnis wie folgt aus:

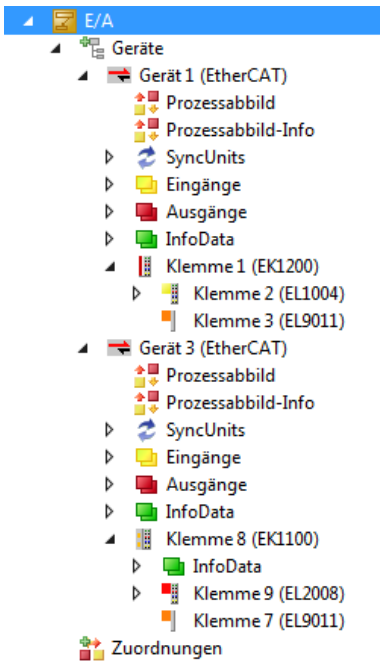


Abb. 101: Abbildung der Konfiguration in VS Shell der TwinCAT 3 Umgebung

Der gesamte Vorgang setzt sich aus zwei Stufen zusammen, die auch separat ausgeführt werden können (erst das Ermitteln der Geräte, dann das Ermitteln der daran befindlichen Elemente wie Boxen, Klemmen o. ä.). So kann auch durch Markierung von „Gerät ..“ aus dem Kontextmenü eine „Suche“ Funktion (Scan) ausgeführt werden, die hierbei dann lediglich die darunter liegenden (im Aufbau vorliegenden) Elemente einliest:

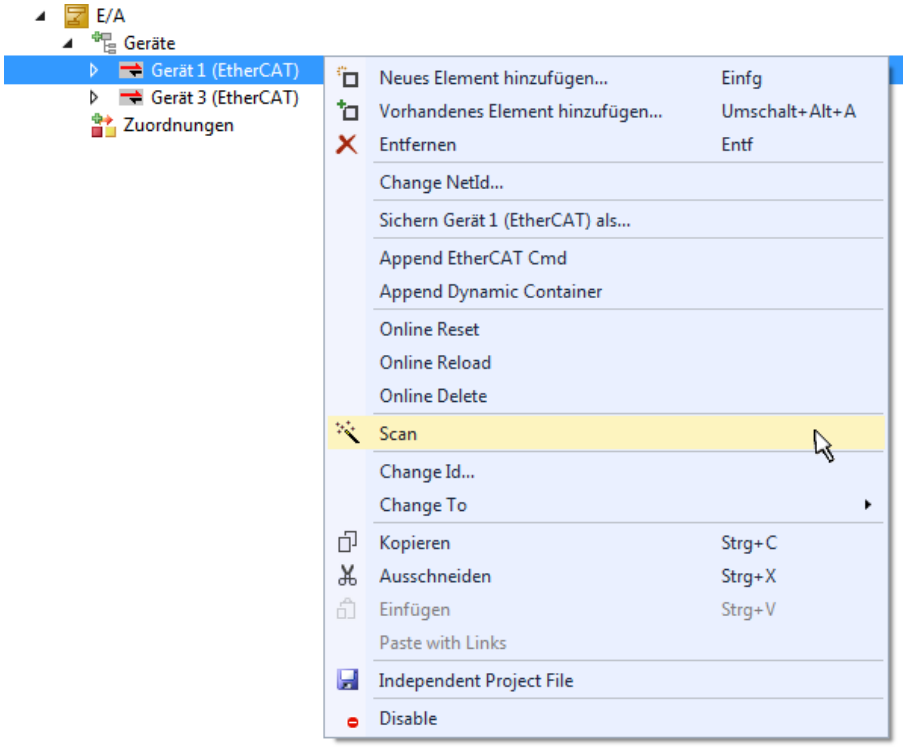


Abb. 102: Einlesen von einzelnen an einem Gerät befindlichen Klemmen

Diese Funktionalität ist nützlich, falls die Konfiguration (d. h. der „reale Aufbau“) kurzfristig geändert wird.

PLC programmieren

TwinCAT PLC Control ist die Entwicklungsumgebung zur Erstellung der Steuerung in unterschiedlichen Programmumgebungen: Das TwinCAT PLC Control unterstützt alle in der IEC 61131-3 beschriebenen Sprachen. Es gibt zwei textuelle Sprachen und drei grafische Sprachen.

- **Textuelle Sprachen**
  - Anweisungsliste (AWL, IL)
  - Strukturierter Text (ST)
- **Grafische Sprachen**
  - Funktionsplan (FUP, FBD)
  - Kontaktplan (KOP, LD)
  - Freigrafischer Funktionsplaneditor (CFC)
  - Ablaufsprache (AS, SFC)

Für die folgenden Betrachtungen wird lediglich vom strukturierten Text (ST) Gebrauch gemacht.

Um eine Programmierumgebung zu schaffen, wird dem Beispielprojekt über das Kontextmenü von „SPS“ im Projektmappen-Explorer durch Auswahl von „Neues Element hinzufügen….“ ein PLC Unterprojekt hinzugefügt:

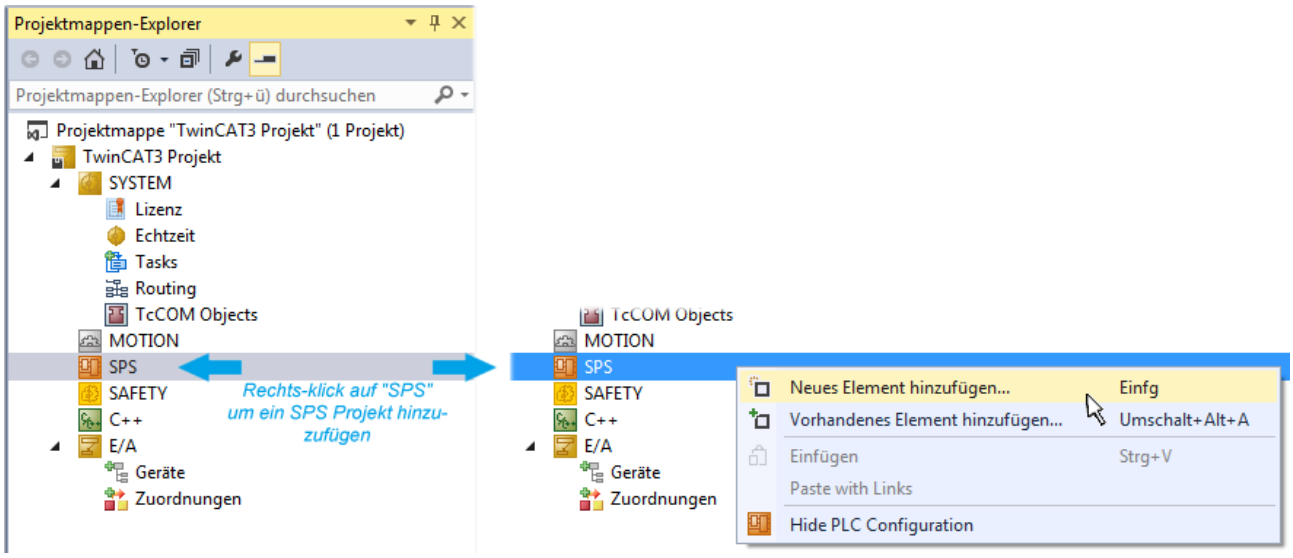


Abb. 103: Einfügen der Programmierumgebung in „SPS“

In dem darauf folgenden geöffneten Dialog wird ein „Standard PLC Projekt“ ausgewählt und beispielsweise als Projektname „PLC_example“ vergeben und ein entsprechendes Verzeichnis ausgewählt:

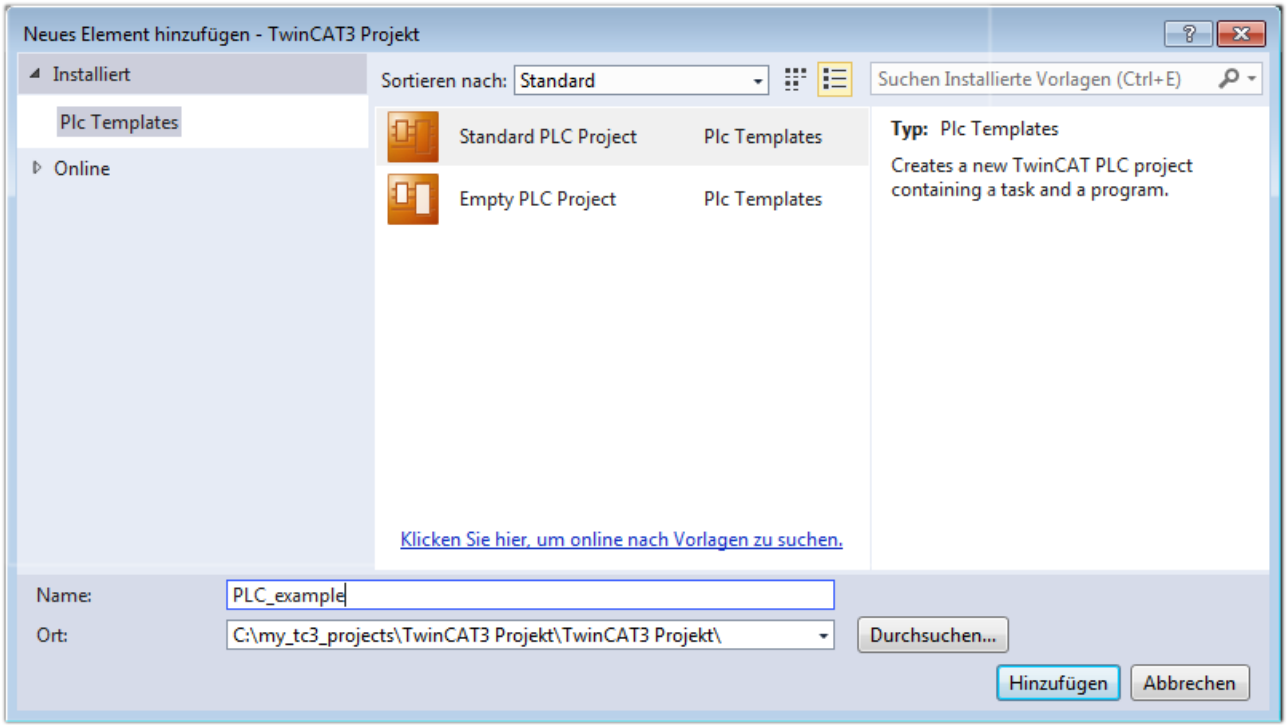


Abb. 104: Festlegen des Namens bzw. Verzeichnisses für die PLC Programmierumgebung

Das durch Auswahl von „Standard PLC Projekt“ bereits existierende Programm „Main“ kann über das „PLC_example_Project“ in „POUs“ durch Doppelklick geöffnet werden. Es wird folgende Benutzeroberfläche für ein initiales Projekt dargestellt:

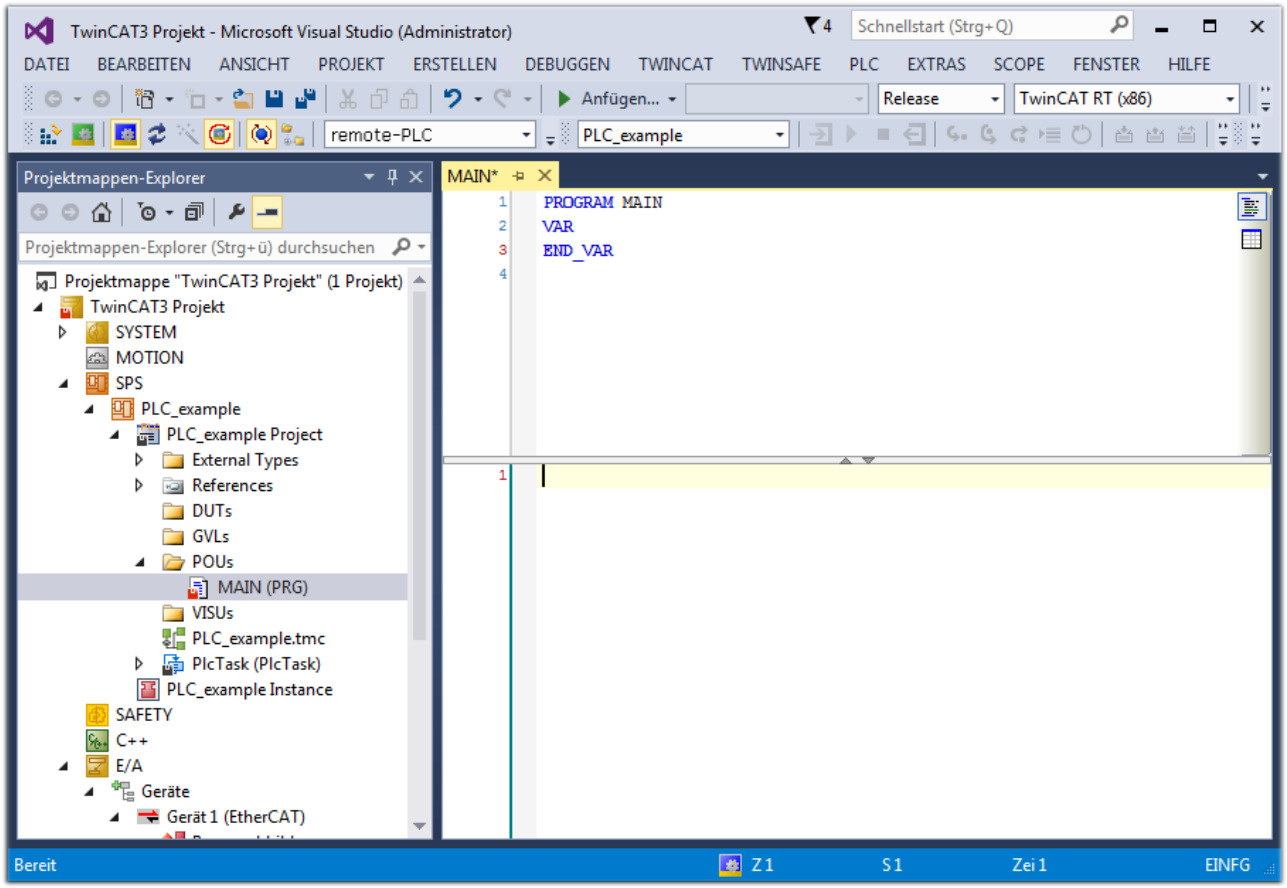


Abb. 105: Initiales Programm „Main“ des Standard PLC Projektes

Nun sind für den weiteren Ablauf Beispielvariablen sowie ein Beispielprogramm erstellt worden:

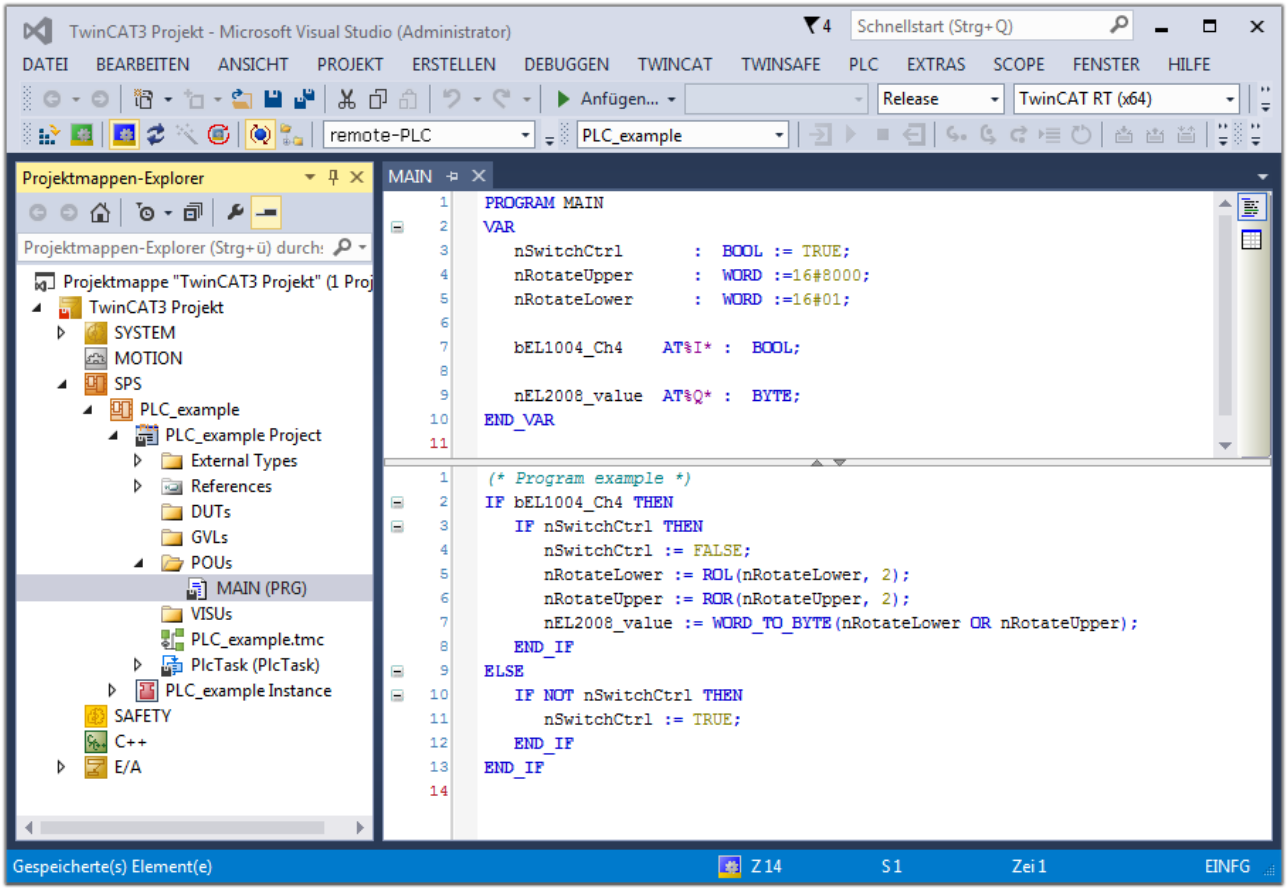


Abb. 106: Beispielprogramm mit Variablen nach einem Kompiliervorgang (ohne Variablenanbindung)

Das Steuerprogramm wird nun als Projektmappe erstellt und damit der Kompiliervorgang vorgenommen:

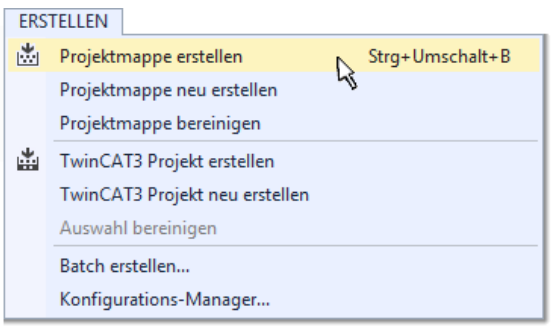
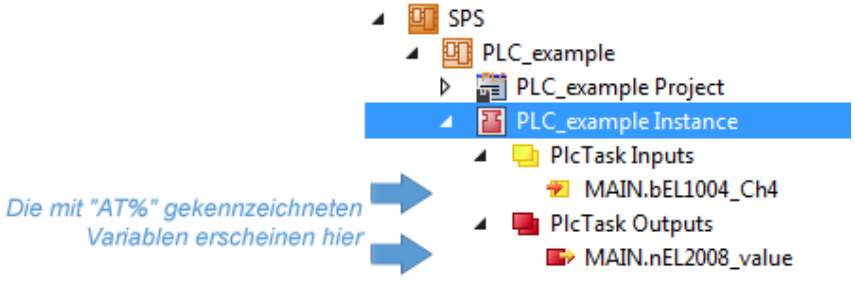


Abb. 107: Kompilierung des Programms starten

Anschließend liegen in den „Zuordnungen“ des Projektmappen-Explorers die folgenden – im ST/ PLC Programm mit „AT%“ gekennzeichneten Variablen vor:



Variablen Zuordnen

Über das Menü einer Instanz – Variablen innerhalb des „SPS“ Kontextes wird mittels „Verknüpfung Ändern…“ ein Fenster zur Auswahl eines passenden Prozessobjektes (PDOs) für dessen Verknüpfung geöffnet:

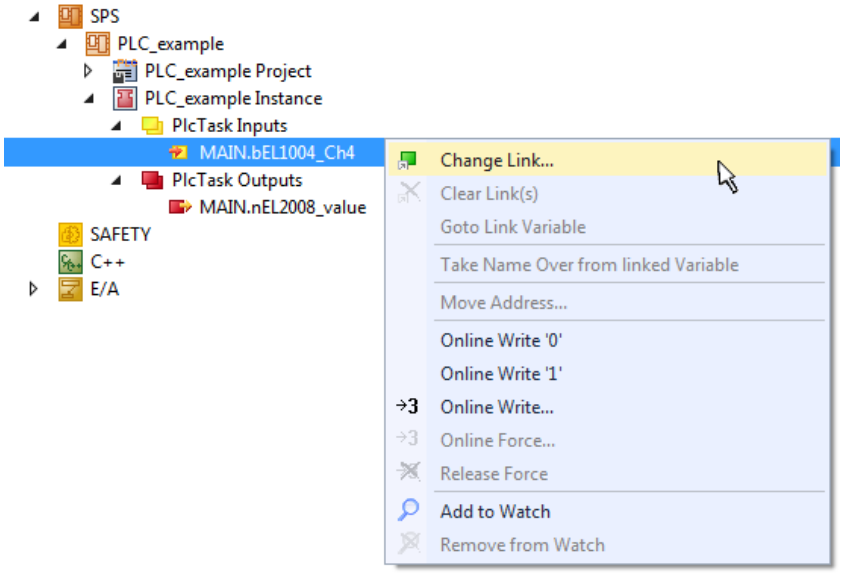


Abb. 108: Erstellen der Verknüpfungen PLC-Variablen zu Prozessobjekten

In dem dadurch geöffneten Fenster kann aus dem SPS-Konfigurationsbaum das Prozessobjekt für die Variable „bEL1004_Ch4“ vom Typ BOOL selektiert werden:

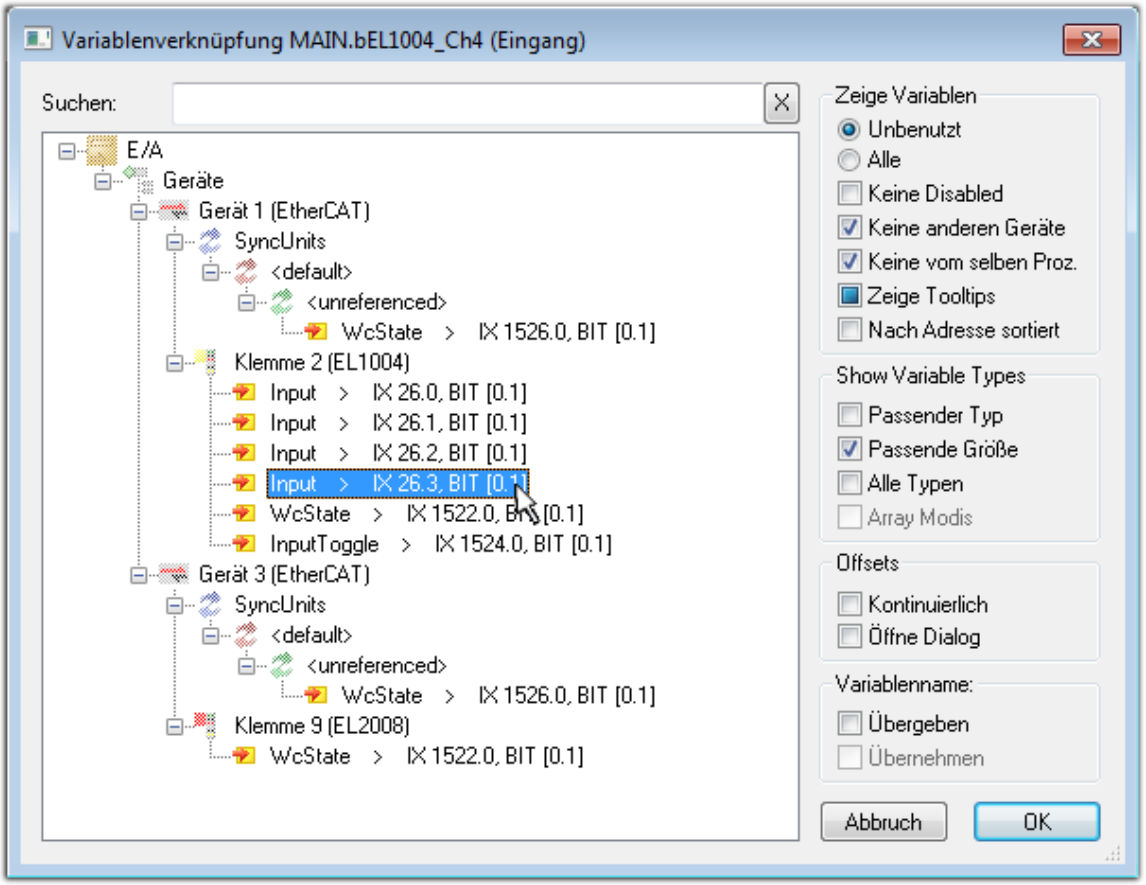


Abb. 109: Auswahl des PDO vom Typ BOOL

Entsprechend der Standarteinstellungen stehen nur bestimmte PDO Objekte zur Auswahl zur Verfügung. In diesem Beispiel wird von der Klemme EL1004 der Eingang von Kanal 4 zur Verknüpfung ausgewählt. Im Gegensatz hierzu muss für das Erstellen der Verknüpfung der Ausgangsvariablen die Checkbox „Alle Typen“ aktiviert werden, um in diesem Fall eine Byte-Variable einen Satz von acht separaten Ausgangsbits zuzuordnen. Die folgende Abbildung zeigt den gesamten Vorgang:

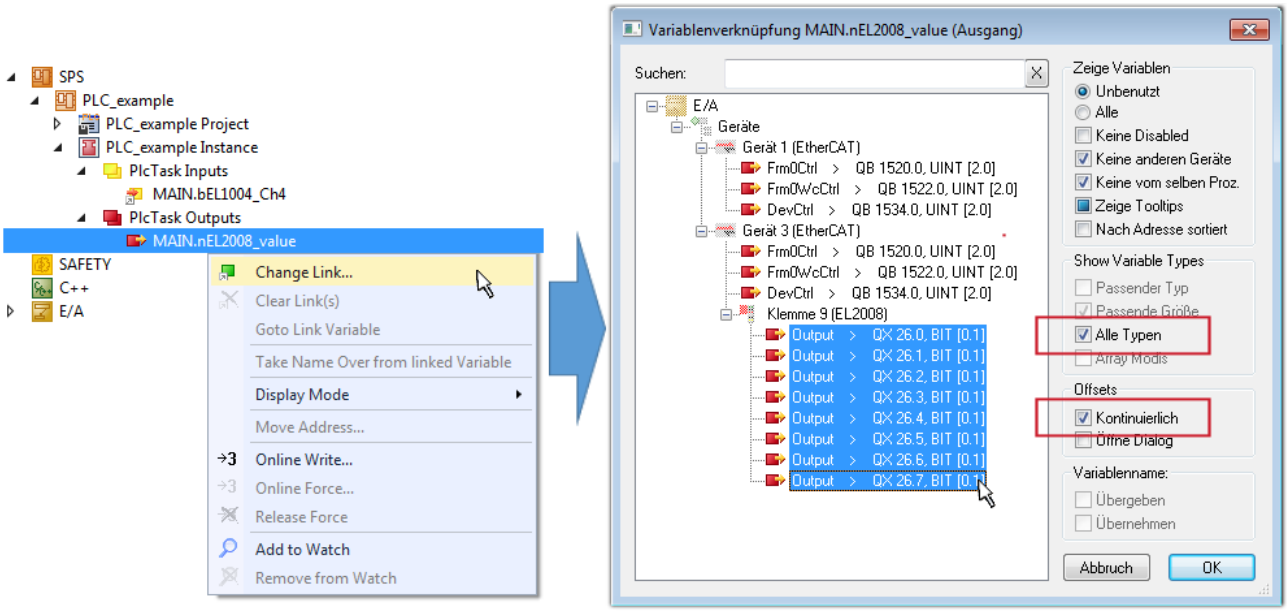


Abb. 110: Auswahl von mehreren PDO gleichzeitig: Aktivierung von „Kontinuierlich“ und „Alle Typen“

Zu sehen ist, dass überdies die Checkbox „Kontinuierlich“ aktiviert wurde. Dies ist dafür vorgesehen, dass die in dem Byte der Variablen „nEL2008_value“ enthaltenen Bits allen acht ausgewählten Ausgangsbits der Klemme EL2008 der Reihenfolge nach zugeordnet werden sollen. Damit ist es möglich, alle acht Ausgänge der Klemme mit einem Byte entsprechend Bit 0 für Kanal 1 bis Bit 7 für Kanal 8 von der PLC im Programm später anzusprechen. Ein spezielles Symbol ( ) an dem gelben bzw. roten Objekt der Variablen zeigt an, dass hierfür eine Verknüpfung existiert. Die Verknüpfungen können z. B. auch überprüft werden, indem „Goto Link Variable“ aus dem Kontextmenü einer Variable ausgewählt wird. Dann wird automatisch das gegenüberliegende verknüpfte Objekt, in diesem Fall das PDO selektiert:

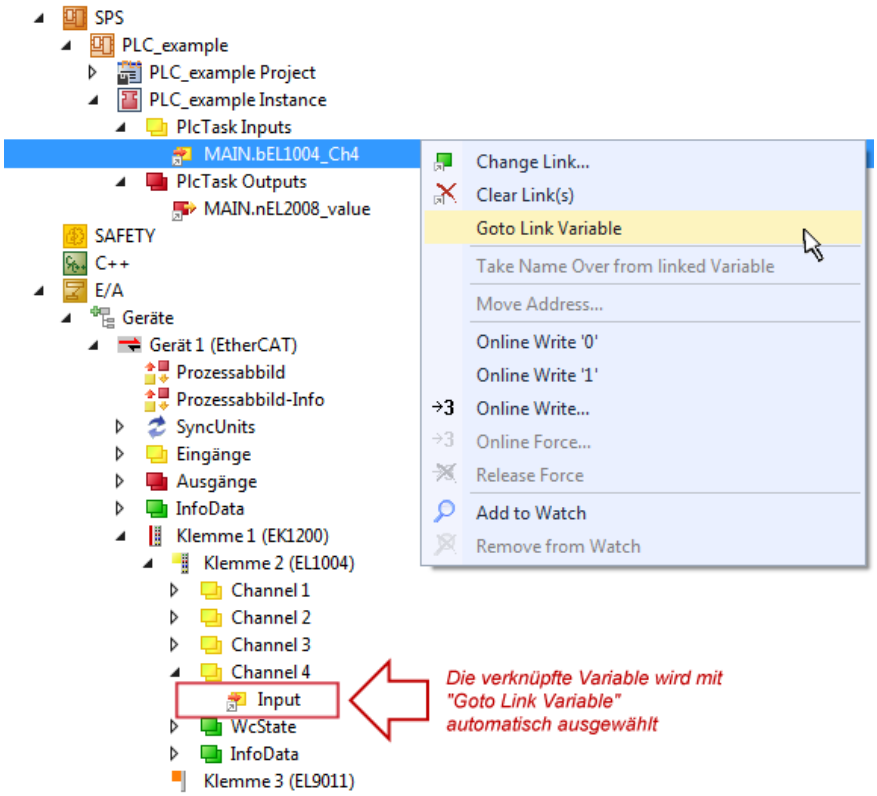


Abb. 111: Anwendung von "Goto Link Variable" am Beispiel von „MAIN.bEL1004_Ch4“

Der Vorgang zur Erstellung von Verknüpfungen kann auch in umgekehrter Richtung, d. h. von einzelnen PDO ausgehend zu einer Variablen erfolgen. In diesem Beispiel wäre dann allerdings eine komplette Auswahl aller Ausgangsbits der EL2008 nicht möglich, da die Klemme nur einzelne digitale Ausgänge zur Verfügung stellt. Hat eine Klemme einen Byte, Word, Integer oder ein ähnliches PDO, so ist es möglich dies wiederum einen Satz von bit-typisierten Variablen (Typ „BOOL“) zuzuordnen. Auch hier kann ebenso in die andere Richtung ein „Goto Link Variable“ ausgeführt werden, um dann die betreffende Instanz der PLC zu selektieren.

Hinweis zur Art der Variablen-Zuordnung

Diese folgende Art der Variablen Zuordnung kann erst ab der TwinCAT Version V3.1.4024.4 verwendet werden und ist ausschließlich bei Klemmen mit einem Mikrocontroller verfügbar.

In TwinCAT ist es möglich eine Struktur aus den gemappten Prozessdaten einer Klemme zu erzeugen. Von dieser Struktur kann dann in der SPS eine Instanz angelegt werden, so dass aus der SPS direkt auf die Prozessdaten zugegriffen werden kann, ohne eigene Variablen deklarieren zu müssen.

Beispielhaft wird das Vorgehen an der EL3001 1-Kanal-Analog-Eingangsklemme -10…+10 V gezeigt.

1. Zuerst müssen die benötigten Prozessdaten im Reiter „Prozessdaten“ in TwinCAT ausgewählt werden.
2. Anschließend muss der SPS Datentyp im Reiter „PLC“ über die Check-Box generiert werden.
3. Der Datentyp im Feld „Data Type“ kann dann über den „Copy“-Button kopiert werden.

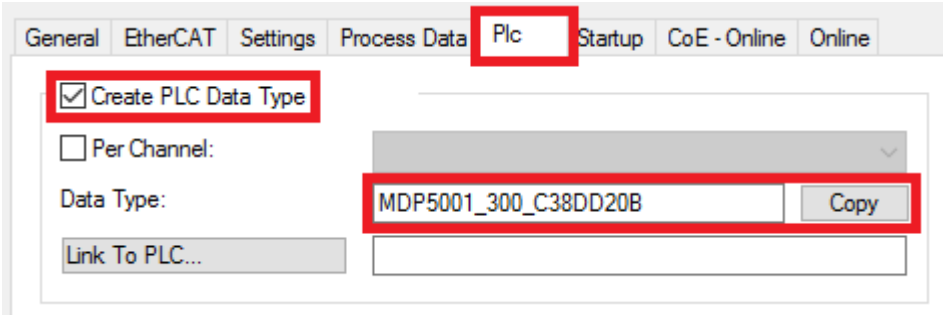


Abb. 112: Erzeugen eines SPS Datentyps

4. In der SPS muss dann eine Instanz der Datenstruktur vom kopierten Datentyp angelegt werden.

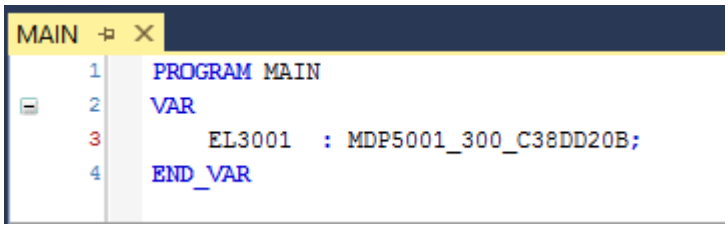


Abb. 113: Instance_of_struct

5. Anschließend muss die Projektmappe erstellt werden. Das kann entweder über die Tastenkombination „STRG + Shift + B“ gemacht werden oder über den Reiter „Erstellen“/ „Build“ in TwinCAT.
6. Die Struktur im Reiter „PLC“ der Klemme muss dann mit der angelegten Instanz verknüpft werden.

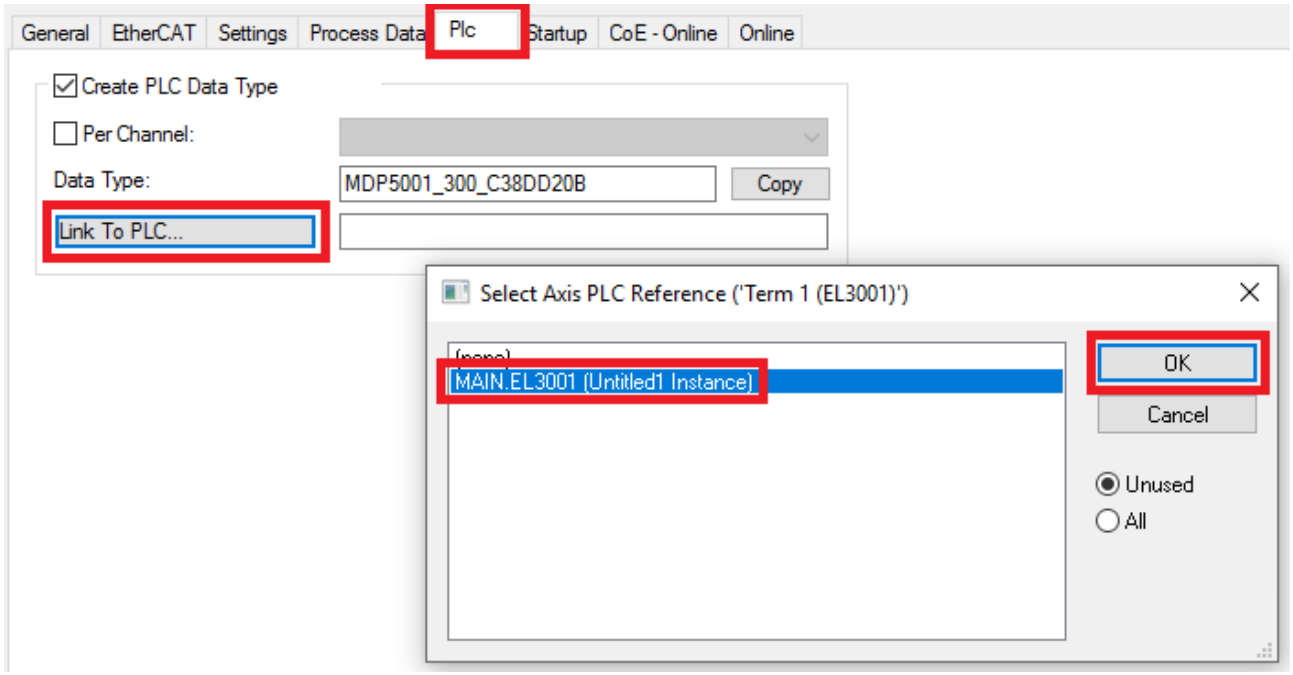


Abb. 114: Verknüpfung der Struktur

7. In der SPS können die Prozessdaten dann über die Struktur im Programmcode gelesen bzw. geschrieben werden.

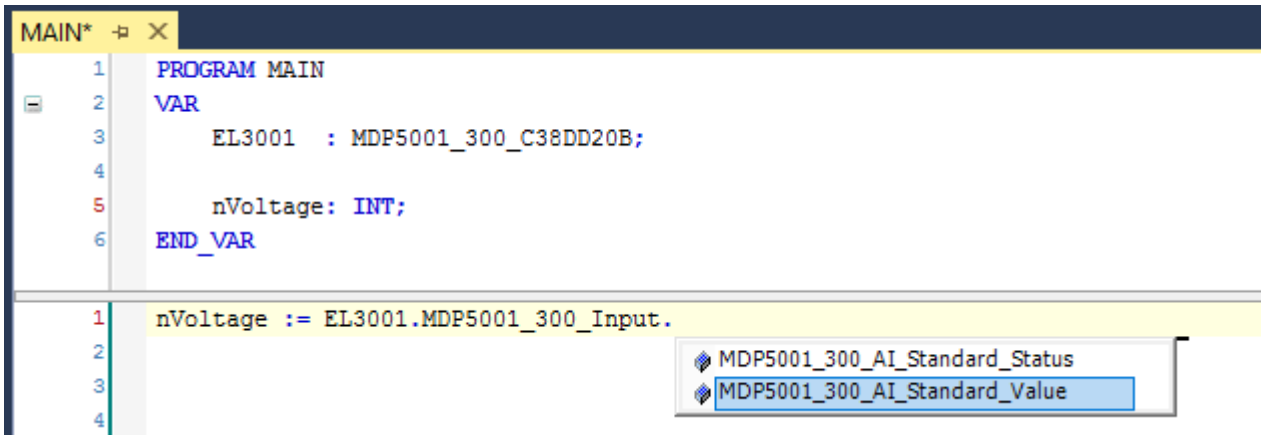
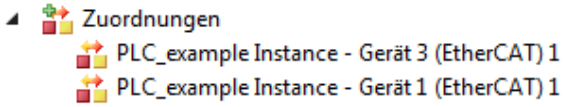


Abb. 115: Lesen einer Variable aus der Struktur der Prozessdaten

Aktivieren der Konfiguration

Die Zuordnung von PDO zu PLC Variablen hat nun die Verbindung von der Steuerung zu den Ein- und

Ausgängen der Klemmen hergestellt. Nun kann die Konfiguration mit  oder über das Menü unter „TWINCAT“ aktiviert werden, um dadurch Einstellungen der Entwicklungsumgebung auf das Laufzeitsystem zu übertragen. Die darauf folgenden Meldungen „Alte Konfigurationen werden überschrieben!“ sowie „Neustart TwinCAT System in Run Modus“ werden jeweils mit „OK“ bestätigt. Die entsprechenden Zuordnungen sind in dem Projektmappen-Explorer einsehbar:



Einige Sekunden später wird der entsprechende Status des Run Modus mit einem rotierenden Symbol  unten rechts in der Entwicklungsumgebung VS Shell angezeigt. Das PLC System kann daraufhin wie im Folgenden beschrieben gestartet werden.

Starten der Steuerung

Entweder über die Menüauswahl „PLC“ → „Einloggen“ oder per Klick auf  ist die PLC mit dem Echtzeitsystem zu verbinden und nachfolgend das Steuerprogramm zu geladen, um es ausführen lassen zu können. Dies wird entsprechend mit der Meldung „*Kein Programm auf der Steuerung! Soll das neue Programm geladen werden?*“ bekannt gemacht und ist mit “Ja” zu beantworten. Die Laufzeitumgebung ist

bereit zum Programmstart mit Klick auf das Symbol  , Taste „F5“ oder entsprechend auch über „PLC“ im Menü durch Auswahl von „Start“. Die gestartete Programmierumgebung zeigt sich mit einer Darstellung der Laufzeitwerte von einzelnen Variablen:

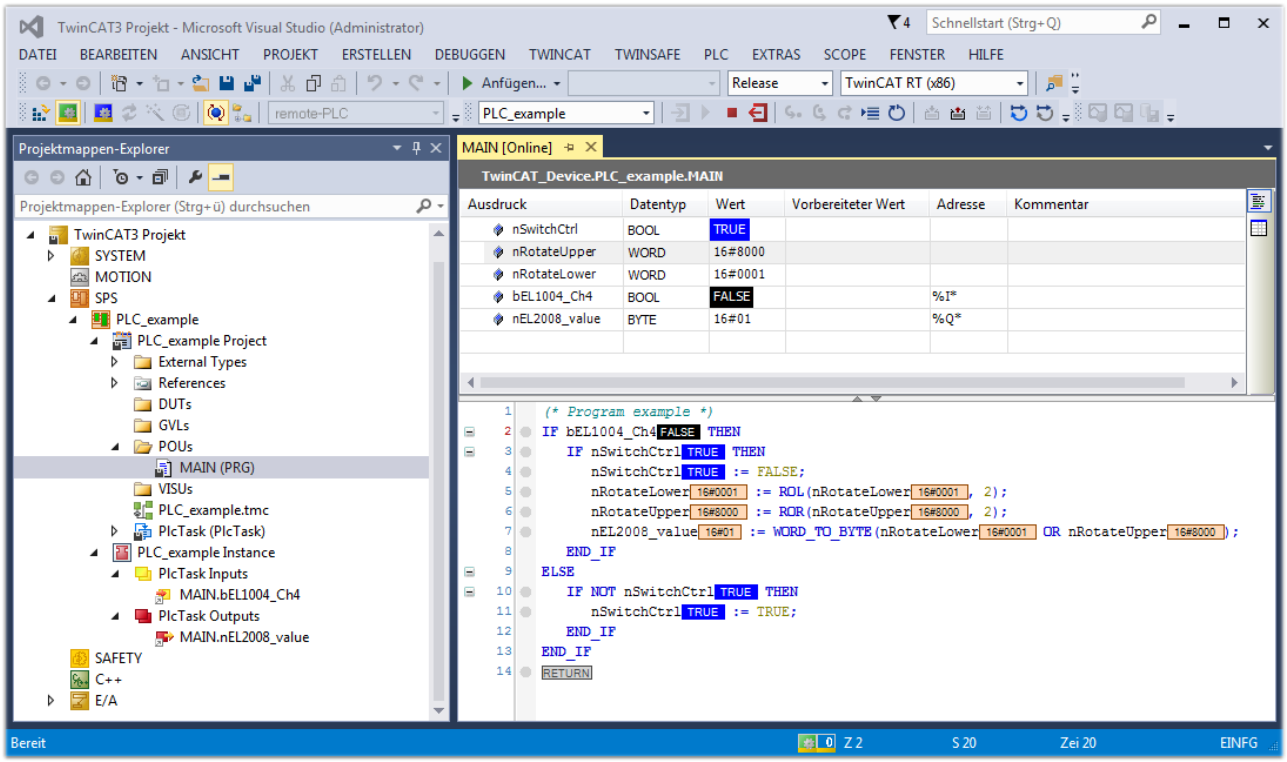


Abb. 116: TwinCAT 3 Entwicklungsumgebung (VS Shell): Logged-in, nach erfolgten Programmstart

Die beiden Bedienelemente zum Stoppen und Ausloggen führen je nach Bedarf zu der gewünschten Aktion (entsprechend auch für Stopp „umschalt-Taste + F5“ oder beide Aktionen über das „PLC“ Menü auswählbar).

3.2 EtherCAT Master in TwinCAT

Der Beckhoff TwinCAT System Manager als Konfigurationsoberfläche für die I/O-Umgebung in der Applikation unterstützt durch verschiedene Automatismen die Einrichtung und Inbetriebnahme eines EtherCAT Feldbusses. Das virtuelle "Gerät" EtherCAT wird im System Manager im Konfigurationsbaum als eigenständiges Element angelegt, seine Eigenschaften sind über entsprechende Eigenschaftenfenster zugänglich. EtherCAT definiert sich hier über die Elemente

- der EtherCAT Master in TwinCAT [▶ 89]
- die Echtzeitumgebung TwinCAT [▶ 91]
- die Ethernet-Schnittstelle [▶ 92]
- die angeschlossenen EtherCAT Slaves [▶ 93]

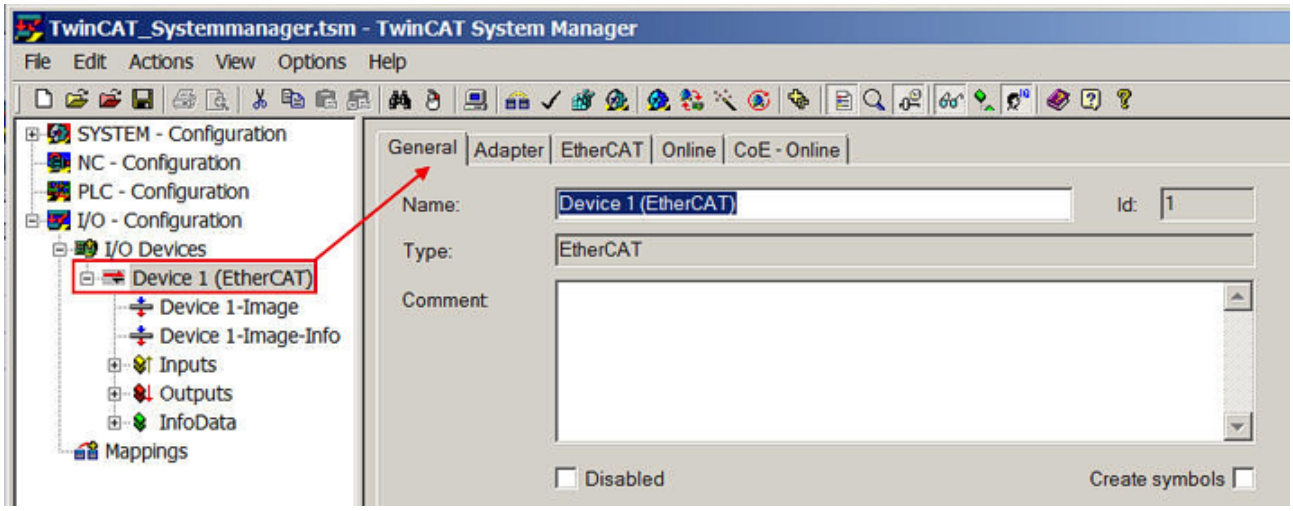


Abb. 117: Gerät EtherCAT im Konfigurationsbaum

Der EtherCAT Master als virtuelles Software Gerät

EtherCAT als Ethernet-basierender Echtzeit-Feldbus ist zur Verbindung mit der I/O-Umgebung auf eine physikalische Ethernet-Schnittstelle an der Steuerung und einen Echtzeit-Trigger nach Anforderung angewiesen. Dadurch kann der EtherCAT Master mit dem angeschlossenen EtherCAT Umfeld kommunizieren.

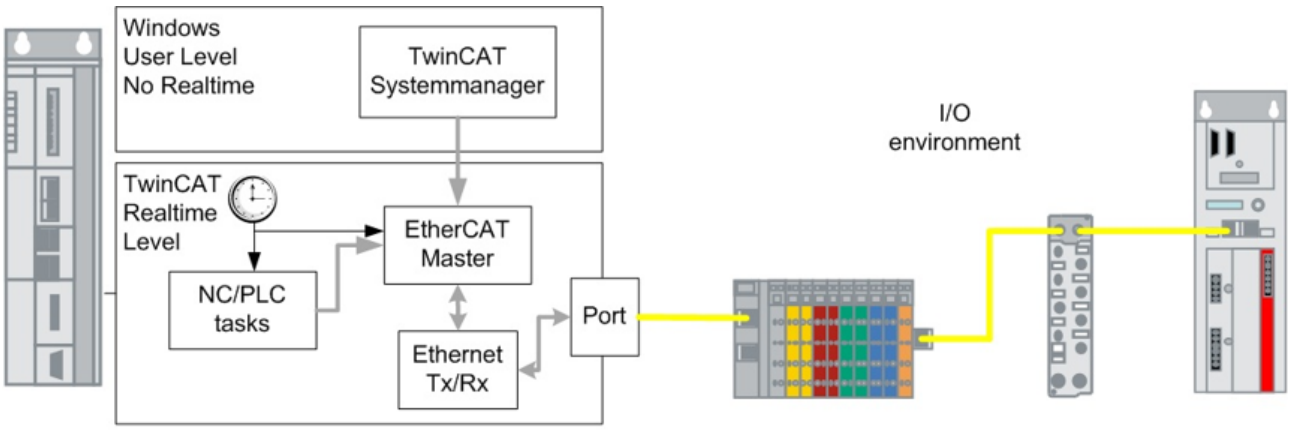


Abb. 118: EtherCAT Master in der IPC-Umgebung

Der EtherCAT Master als Software-Gerät in TwinCAT

- kann aus den angelieferten Prozessdaten der PLC/NC/Task die Ethernet-Telegramme mit den EtherCAT-Datagrammen zusammensetzen und über den zugewiesenen Ethernet Port verschicken.
- kann aus dem I/O-Umfeld empfangene Prozessdaten entpacken und wieder an die Tasks zurückreichen.
- kümmert sich um die Konstruktion von zyklischen und azyklischen Telegrammen.
- regelt die Distributed-Clocks-Synchronisation.
- wertet die Diagnose-Informationen der EtherCAT Slaves aus.
- führt anlassbezogene Diagnose durch oder reagiert auf veränderte Topologien.
- kann nur *ein* EtherCAT-System mit bis zu 65535 Slaves verwalten und wird dazu mit *einer* Ethernet-Schnittstelle ("RJ45 Port") verlinkt. Zum Zwecke der Kabelredundanz kann dem EtherCAT-Master noch ein zweiter Port zugeteilt werden.
- verwaltet die Kommunikation mit anderen EtherCAT Mastern im gleichen TwinCAT-System, falls mehrere in der Konfiguration angelegt werden.
- ist der alleinige Erzeuger von EtherCAT-Telegrammen in einem EtherCAT System.
- ....

EtherCAT Kommunikation besteht aus **zyklischen** und **azyklischen** Ethernet-Telegrammen.

Die **zyklischen** bilden die normalen Prozessdaten ab und sind zur Anlagenlaufzeit üblicherweise nicht veränderlich. Aus der bekannten Konfiguration, d.h. der Menge der angeschlossenen EtherCAT Slaves, ergibt sich immer der gleiche Mindestumfang an Prozessdaten, die zur zyklischen Kommunikation der Teilnehmer mit dem Master verschickt werden müssen. Die zyklisch (also in konstantem Abstand) ausgeführte triggernde Task, z. B. ein PLC-Task mit 10 ms Laufzeit) stößt die Kommunikation mit dem EtherCAT-Feld an und erhält nach Abschluss der Kommunikation die Prozessdaten vom EtherCAT Master zurückgeliefert. TwinCAT arbeitet auf IPC/Embedded-Systemen üblicherweise mit konstanten Zykluszeiten von z. B. 50 µs ... > 100 ms. Innerhalb dieser Zykluszeit müssen die Rechenoperationen der Task abgeschlossen sein. Die hohe Kommunikationsleistung von EtherCAT (hoher 100 MBit/S Datendurchsatz) ermöglicht es auch in großen Konfiguration von > 1000 Teilnehmern, dass die versandten EtherCAT Telegramme schon vor Beginn des nächsten Zyklus wieder in der Steuerung angelangt sind. Deshalb ist die Taskausführung und die Buskommunikation als synchrone Aufgabe in TwinCAT angelegt. Für die hochqualitative Ausführung entscheidend ist dabei ein geringer Jitter, d.h. die Tasks sollen mit hoher zeitlicher Präzision ausgeführt bzw. wiederholt gestartet werden.

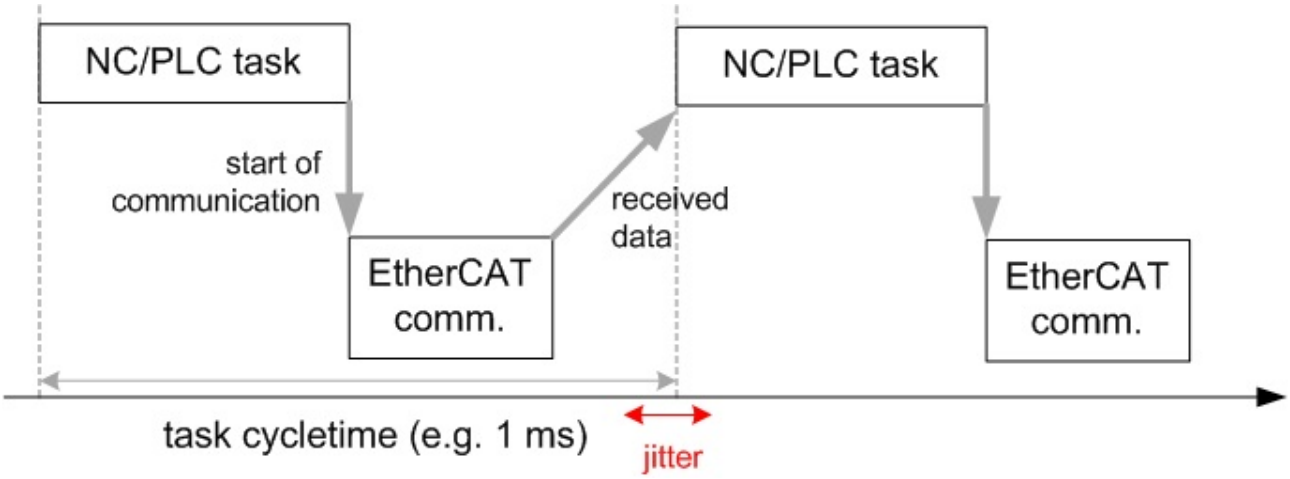


Abb. 119: Synchrone Ausführung von Task und EtherCAT Kommunikation

In Abb. *Synchrone Ausführung von Task und EtherCAT Kommunikation* erfolgt der Start der Kommunikation nach der Task. Auch eine Frame-Versendung gleichzeitig zum Task-Startzeitpunkt ist möglich ("I/O at task begin").

Die **azyklischen** Telegramme werden vom EtherCAT Master nach Erfordernis erstellt und in den Pausen zwischen den zyklischen Daten versendet bzw. empfangen. Mit diesen Telegrammen werden Diagnoseinformationen eingeholt, Mailbox-Kommunikation zu einzelnen Teilnehmern transportiert z. B. zum

Firmware-Update oder zu einem unterlagerten Feldbus. Die azyklischen Telegramme werden ohne Echtzeitanspruch in der Reihenfolge ihrer Anforderung je nach verfügbarem Platz aus einer Pipeline versendet und werden daher auch "queued frames" (aneinandergereihte Frames) genannt.

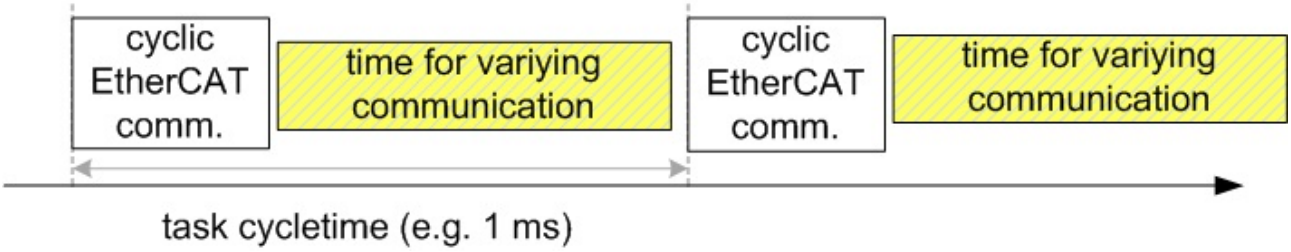


Abb. 120: Zeitliche Aufteilung feste (so genannte zyklische) und variable EtherCAT-Telegramme (so genannte azyklische)

Auch der TwinCAT System Manager bildet diese Zeitverhältnisse in jeder erstellten Konfiguration ab. In Abb. *Konfigurationsabhängige Zeitaufteilung im System Manager* ist die eine 100 µs-Task zu sehen, die mit ca. 20 Slaves kommuniziert und dazu einen Ethernet-Frame (rot) mit mehreren Datagrammen zur zyklischen Kommunikation benötigt.

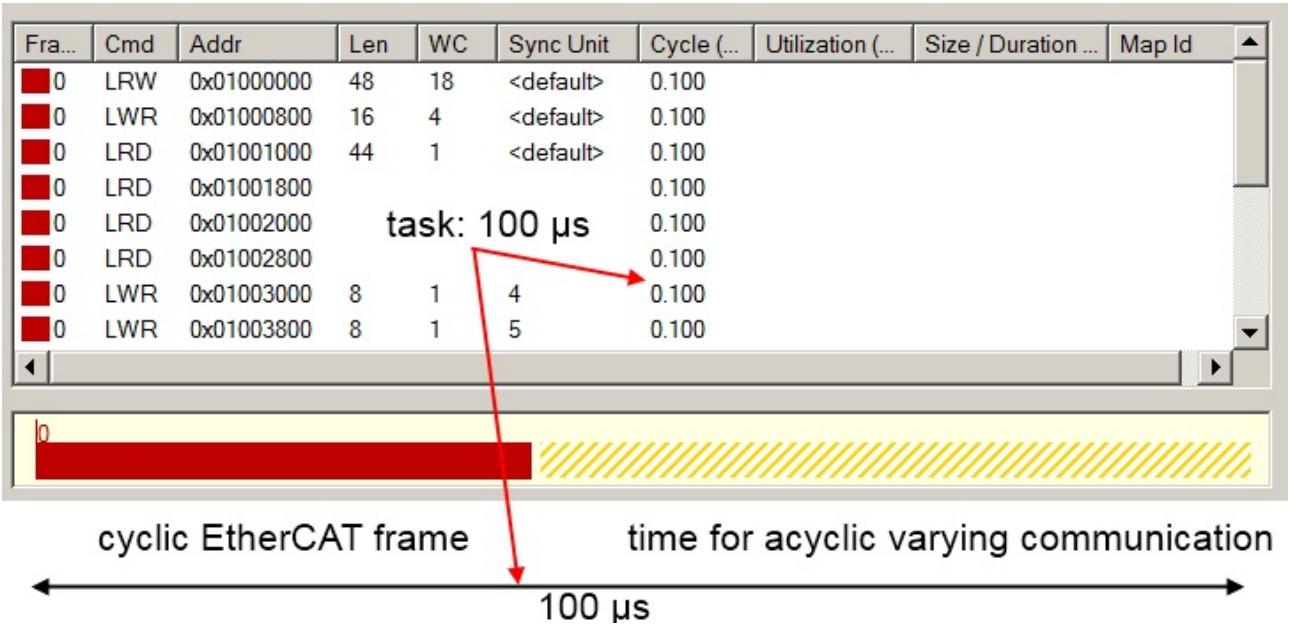


Abb. 121: Konfigurationsabhängige Zeitaufteilung im System Manager

Die Echtzeitumgebung

Der EtherCAT Master selbst regelt "nur" den Aufbau und die Interpretation der EtherCAT Telegramme. Eine TwinCAT-basierende Steuerung stützt sich auf wiederholte zyklische Ausführung einer Task mit konstanter Wiederholrate, der so genannten Zykluszeit. Üblich im TwinCAT Umfeld sind Zykluszeiten von 50 µs über 1 ms bis zu mehreren 100 ms. Die Zykluszeit wird bei der Konfigurationserstellung gewählt in Abhängigkeit von der Rechenleistung der verwendeten Steuerung, den Busteilnehmern, den ausgeführten Programmen, den applikativen Anforderungen u.a.

Als Task wird dabei jede zyklisch auszuführende Aufgabe bezeichnet, z. B.: NC-Berechnungen, PLC-Code, eine Visualisierung oder vom Kunden erstellte und angetriggerte R3-Applikationen. Die Tasks können unterschiedliche Zykluszeiten besitzen, müssen aber in einer Priorität gewichtet sein. Höherpriore Tasks können den Ablauf von niederprioren Taks unterbrechen und damit pausieren lassen. Die niederpriore Task wird fortgesetzt, sobald die Prioritätsliste dies erlaubt. Werden mehr als ein CPU-Core genutzt (erst mit TwinCAT 3 möglich), können auch mehrere Tasks parallel ausgeführt werden.

In einem korrekt dimensionierten und parametrierten TwinCAT System können alle angelegten Tasks auch bei unterschiedlichen Prioritäten in der vorgesehenen Zykluszeit einmal je Gesamtzyklus durchgeführt werden. Die automatischen Einstellungen des System Manager bei Konfigurationserstellung gewährleisten

in der Regel einen stabilen Betrieb der Konfiguration. Als Gesamtzyklus wird hier die langsamste Task bezeichnet. Z. B. muss bei einer 1 ms und einer auf dem gleichen System befindlichen 100 ms Task die schnelle Task 100-mal ausgeführt werden, bis die langsame Task einmal ausgeführt wird.

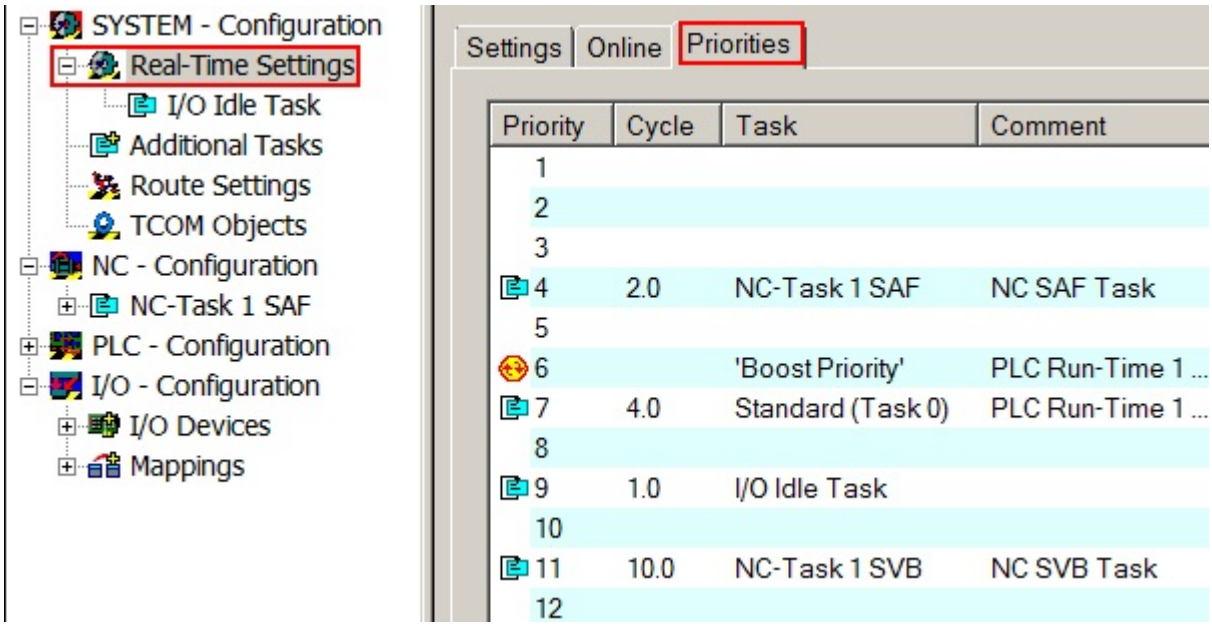


Abb. 122: Prioritätsliste im TwinCAT System Manager

Die Echtzeiteigenschaften von TwinCAT auf Windows-basierenden Systemen (2000, NT, XP, 7, CE; WES, WEC) gewährleisten auch ohne dezidierte Timing-Hardware einen geringen Jitter und damit hohe Konstanz und exaktes Timing auch bei kürzesten Zykluszeiten von 50 µs.

Prinzipiell kann jede Task ihr eigenes I/O-Abbild haben und so einen eigenen Feldbus-Zyklus mit den Teilnehmern führen. In Abb. *2 Tasks mit eigenen Ethernet-Frames (Task 1 ms: „roter“ Frame, Task 10 ms: „gelber“ Frame)* sind 2 Tasks (1 + 10 ms) zusehen die jeweils eigene I/O-Zyklen auslösen und damit eigene Ethernet-Frames bewirken.

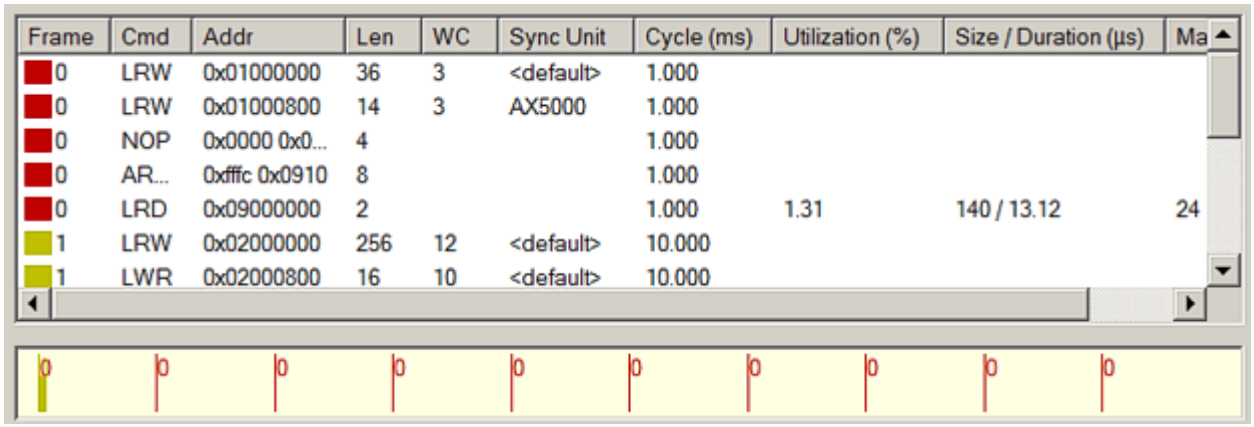


Abb. 123: Tasks mit eigenen Ethernet-Frames (Task 1 ms: „roter“ Frame, Task 10 ms: „gelber“ Frame)

Die Ethernet Schnittstelle

EtherCAT ist aktuell (2011) auf ISO/OSI Layer 2 mit FastEthernet = 100 MBit/s bis in die Slave-Ebene standardisiert. Die Schnittstelle muss also mindestens FastEthernet unterstützen. Siehe dazu auch Hinweise zur EtherCAT Infrastruktur.

Die dem EtherCAT Master zugeteilte Ethernet Schnittstelle muss im Weiteren den Ansprüchen der Applikation genügen:

- äußere Anforderungen: Temperatur, Vibration, Ausziehschutz, Knickschutz, Verschmutzung.

  Als Verbindung zur Steuerung/IPC kommt üblicherweise ein RJ45, M12 oder M8-Übergang zum Einsatz.

- Echtzeitfähigkeit
  Die Ethernet Schnittstelle muss in ihren zeitkritischen Eigenschaften den Ansprüchen der Applikation gerecht werden und sollte zeitrichitg versandte Telegramme des EtherCAT Masters nicht verzögern. Dies betrifft die Software-Implementierung (Treiber, Stack-Verwaltung) wie die elektronische Ausführung (PCI- oder USB-Anbindung, DMA, NDIS-Verwaltung, vorgeschalteter Switch).

EtherCAT Slaves

Bei der Zusammenstellung der Konfiguration sind einige Faktoren zu beachten:

- Die Eigenschaften eines EtherCAT Slaves werden für den EtherCAT Master in der Gerätebeschreibungsdatei ESI (EtherCAT Slave Information) definiert. Diese XML-Datei ist vom jeweiligen Gerätehersteller bereitzustellen.
  Der TwinCAT System Manager sucht diese ESI-Dateien standardmäßig unter C:\TwinCAT\IO.
  Eine ESI-Datei kann mehrere Gerätebeschreibungen und -revisionen enthalten.
- Bei im Klemmenstrang aneinandergereihten Klemmen (EL/ES) in der so genannten LVDS-Physik ist der Bedarf an E-Bus-Strom zu beachten. Der System Manager rechnet anhand des in der ESI-Beschreibung angegebenen Strombedarfs mit und warnt bei Überlastgefahr des vorangehenden Kopplers. Diese haben üblicherweise 2A Versorgungsvermögen. Negative Werte in der Spalte "E-Bus" sind nicht zulässig und führen zu u.U. schwer zu reproduzierbaren Fehlern!

Number	Box Name	Address	Type	In Size	Out Size	E-Bus (mA)
18	Klemme 18 (EL2004)	1018	EL2004		0.4	440
19	Klemme 19 (EL2004)	1019	EL2004		0.4	340
20	Klemme 20 (EL2004)	1020	EL2004		0.4	240
21	Klemme 21 (EL2004)	1021	EL2004		0.4	140
22	Klemme 22 (EL2004)	1022	EL2004		0.4	40
23	Klemme 23 (EL2004)	1023	EL2004		0.4	-60 !
24	Klemme 24 (EL6692)	1024	EL6692	4.0	2.0	-180 !
25	Klemme 25 (EL6731)	1025	EL6731	4.0		-530 !
26	Klemme 26 (EL9010)		EL9010			-530 !

Abb. 124: System Manager Ansicht Stromberechnung

- zwischen Teilnehmern auf FX/TX-Ebene (Ethernet Kabel, Glasfaser, POF) dürfen die lt. Übertragungsphysik vorgegebenen max. Kabellängen und Dämpfungswerte nicht überschritten werden.
- die Anzahl und Anordnung der EtherCAT Slaves (max. 65535) hat Einfluss auf die Durchlaufzeit der EtherCAT Telegramme.
- es ist auf funktionssichere Schirmanbindung der Koppler/Klemmen und Erdung der Gesamtapplikation zu achten.
- Zuleitungen/Sensorleitungen sind ggf. gesondert zu schirmen.

3.3 Allgemeine Hinweise zur Verwendung von Beckhoff EtherCAT IO-Komponenten

Hinweis: dieses Dokument gibt übergeordnete Hilfestellungen. Für Detailfragen (z. B. „Wo finde ich die Revision?“) ziehen Sie bitte die entsprechenden Seiten in dieser EtherCAT Systemdokumentation zu Rate.

Übersicht:

3.3.1 Technische Einordnung

Die Anwendungs- und Kommunikationseigenschaften eines Beckhoff IO-Geräts (EtherCAT Klemme EL/ES/EM/EK, EtherCAT Box EP/EQ, Sonderbauformen CU) werden durch folgende 3 Elemente bestimmt: das **Gerät**, die **Firmware** (optional) und die **EtherCAT Slave Gerätebeschreibung** (ESI).
Ein elektronisches Automatisierungsgerät mit Feldbusanschluss besteht noch aus vielen weiteren (internen)

Komponenten, Software und Bauteilen wie z. B. der ESC (EtherCAT Slave Controller, der Realtime Kommunikations-IC), die 3 oben genannten sind jedoch diejenigen, die das Gerät nach außen hin in seinen Eigenschaften für den Anwender definieren.

Im Einzelnen zu den 3 Elementen:

Das materielle Gerät selbst, die "Hardware" (HW)

- wird durch Beckhoff gekennzeichnet durch den Hardwarestand z. B. HW01.
- im EtherCAT-Sprachgebrauch wird das Gerät auch oft als **Slave** im Sinne der Feldbustopologie bezeichnet, weil es vom EtherCAT **Master** angesprochen wird.
- der Hardwarestand.
  - wird außen auf das Gerät gedruckt, siehe Seriennummer/Batch-Nummer [▶ 101]
  - ist bei Geräten mit CoE-Verzeichnis dort auslesbar
  - ist seit 2012/01 durch den EtherCAT Master aus dem EtherCAT/ESI-EEPROM auslesbar
  - wird je nach Gerätebaureihe dezimal oder hexaddezimal codiert

Die ggf. darauf laufende Firmware

- **Hinweis**: als Firmware (FW) wird der ausführbare und vom Hersteller (Beckhoff) erstellte Programmcode bezeichnet der auf einem µC (Microcontroller, FPGA oder Prozessor) ausgeführt wird. Nicht jedes Gerät (Device) muss zwingend über einen µC und damit eine Firmware verfügen!
- diese kann ggf. aus mehreren Dateien bestehen und muss auf das IO-Gerät aufgespielt werden. Üblicherweise sind das *.efw oder *.rbf-Dateien
- wird durch Beckhoff gekennzeichnet durch den Firmwarestand z. B. FW02
- der Firmwarestand
  - wird außen auf das Gerät gedruckt, siehe Seriennummer/Batch-Nummer [▶ 101].
  - ist bei Geräten mit CoE-Verzeichnis dort auslesbar.
  - ist seit 2012/01 durch den EtherCAT Master aus dem ESI-EEPROM auslesbar.
  - falls ein Firmware-Update des Gerätes vorgenommen wird, ist der Aufdruck am Gehäuse durch den Ausführenden /Anwender entsprechend anzupassen.

Die EtherCAT ESI Kommunikationsbeschreibung als Gerätebeschreibungsdatei für den EtherCAT Master (bei Beckhoff: in die TwinCAT-Software integriert)

- beschreibt die EtherCAT Kommunikationsschnittstelle zwischen Gerät und Master in allen Aspekten die für Datenkommunikation und Synchronisierung relevant sind
- wird einerseits durch Beckhoff auf das IO-Gerät selbst programmiert (in das sog. EtherCAT/ESI-EEPROM)
  - damit das Gerät beim Scannen durch den EtherCAT Master Grundinformationen von sich aus bekannt geben kann: Prozessdatenumfang (PDO), Einstellungsmöglichkeiten (CoE) und weiteres.
  - außerdem beinhaltet diese ESI die Grundeinstellungen für das IO-Gerät selbst die für die Funktion relevant sind und beim PowerOn vom µC oder anderen Steuerkomponenten auf dem Gerät eingelesen werden.
- sollte andererseits dem EtherCAT Master als Datei vorliegen
  - da die EtherCAT-Master-Software nun den Slave auch ohne elektrischen Zugriff "kennt", kann der der Anwender seine Buskonfiguration auch „offline“ erstellen, d.h. ohne live Kontakt zu dem IO-Gerät, wie es beim Scannen zwingend nötig ist.
  - und außerdem ist dem EtherCAT Master dadurch bekannt, wie er den Slave über EtherCAT ansprechen muss und welche Funktionen dieser bietet. Dadurch sind für den Master z. B. zyklische Prozessdaten und CoE-Verzeichnis des Slaves bestimmt. Ohne ESI-Datei kennt der Master das Gerät schlichtweg nicht.
  - Wenn in TwinCAT ein EtherCat Slave mit der NC verknüpft werden soll, ist zwingend das Vorliegen der ESI Gerätebeschreibungsdatei erforderlich
  - **Hinweis**: es gibt EtherCAT Master, die die Geräteinformationen nur durch den Scan-Vorgang aus dem Slave gewinnen und keine ESI-Datei vom Gerät benötigen.
    Auch TwinCAT kann notfalls "online" ohne vorliegende ESI-Datei arbeiten, für TwinCAT ist

allerdings das Vorliegen der korrekten ESI-Datei dringend empfohlen und zwar aus einem einfachen Grund: viele EtherCAT Geräte verfügen mittlerweile über einen großen Funktionsumfang und umfangreiche Einstellmöglichkeiten. Dies resultiert dann in einer großen ESI-Datei die u.U. nicht mehr vollständig im lokalen EtherCAT/ESI-EEPROM gespeichert werden kann - die dann dort vorliegenden reduzierten Informationen reichen dann zwar für einen Grundbetrieb des Gerätes aus, der volle Funktions- und Diagnoseumfang des Geräts ist dann aber nicht verfügbar.
Außerdem ist nur bei Vorliegen der ESi-Datei eine Offline-Konfiguration möglich.

- wird durch Beckhoff gekennzeichnet durch die sog. Revision z. B. -0018
wenn es zu inhaltlichen/funktionellen Änderungen in der ESI kommt, wird in der Regel der Revisionsstand +1 erhöht
- die Revision
  - ist vom EtherCAT Master aus dem EtherCAT/ESI-EEPROM auslesbar.
  - ist bei Geräten mit CoE-Verzeichnis ebenfalls dort auslesbar.
  - wird seit 2014/01 außen auf das Gerät gedruckt →"Rev. xxxx [▶ 101]"
  - falls ein Revision-Update des Gerätes vorgenommen wird, ist der Aufdruck am Gehäuse durch den Ausführenden/Anwender entsprechend anzupassen (wenn vorhanden).

Alle 3 Elemente können durch ihre Eigenschaften Einfluss haben auf

- funktionale Eigenschaften z. B. Filter, SampleRate, Ausgabegeschwindigkeit, Eingangsempfindlichkeit u.a.
- zeitliches Verhalten z. B. beim Hochlauf,
- Verhalten und Diagnose im Fehlerfall
- Kommunikationseigenschaften z. B. Prozessdaten, Parameterverzeichnis,
- Distributed Clocks Eigenschaften z. B. Triggerarten, Synchronität, Latenz u.a.
- äußeres Erscheinungsbild

Diese 3 Elemente sind wie folgt in der Applikationswelt wieder zu finden:

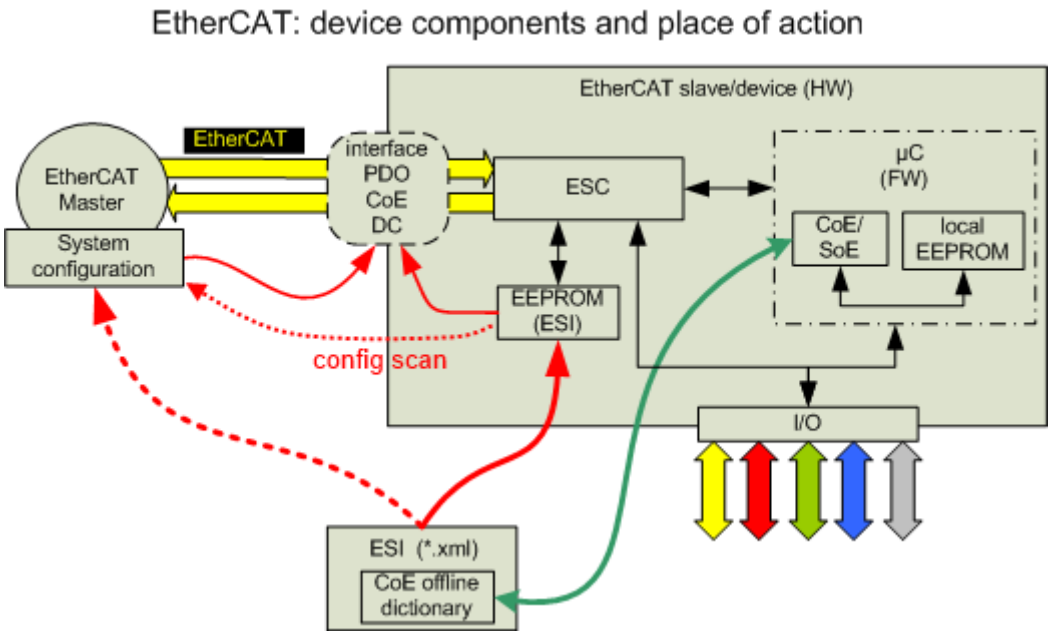


Abb. 125: Komponenten des EtherCAT-Geräts

Erläuterung dazu in Anlehnung an die oben erwähnten Grundlagen:

- die ESI ist im Device einprogrammiert (EtherCAT/ESI-EEPROM) und bestimmt das EtherCAT Interface aus Sicht des µC, die EtherCAT Echtzeitkommunikation übernimmt der ESC (EtherCAT Slave Controller).

- der EtherCAT Master benötigt die ESI um sie z. B. in seine SystemConfiguration einzuarbeiten. Er kann dazu online das EtherCAT/ESI-EEPROM auslesen. Falls ihm allerdings die ESI direkt als Datei vorliegt, ist auch eine Offline-Konfiguration möglich.
  Dadurch haben nun Master und Slave/Gerät die gleiche Definition der EtherCAT Schnittstelle/Interface und können miteinander kommunizieren.
- der µC (falls vorhanden) bildet die Funktion des Gerätes ab und kommuniziert dazu mit dem ESC. Er kontrolliert auch die IO-Seite des Gerätes, falls dies nicht bereits der ESC übernimmt.
- das Parameterverzeichnis "CoE" (CANopen-over-EtherCAT) des Gerätes wird vom µC verwaltet, also "online". Die ESI-Datei enthält auch eine Kopie davon, das sog. "CoE offline dictionary". Damit ist der Anwender in der Lage, bereits offline die möglichen Funktionen einzusehen und ggf. über die StartUp-Liste vorzukonfigurieren.
  Änderungen im Online-CoE werden bei Beckhoff-IO-Geräten und dem AX2000 im Allgemeinen in einem lokalen EEPROM stromausfallsicher gespeichert (wenn nicht anders parametriert). Der AX5000 verfügt dagegen über ein SoE-Verzeichnis (Sercos-over-EtherCAT) das beim PowerOn immer im Default-Zustand startet, anwenderseitig erforderliche Änderungen sind bei jedem Hochlauf vom EtherCAT Master z. B. per StartUp-Liste in den Slave zu laden.

Vereinfacht dargestellt bilden diese 3 Elemente das gesamte Gerät ab wobei die ESI sowohl im Gerät wie auch im Master Verwendung findet:

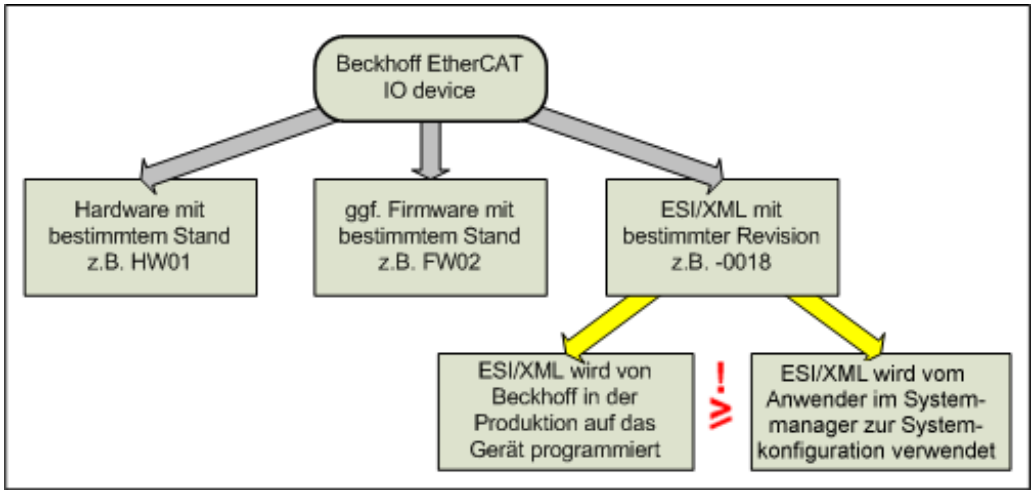


Abb. 126: Vereinfachte Darstellung des gesamten EtherCAT-Geräts

Das gesamte Gerät kann je nach Typ auch mit einer fortlaufenden, eindeutigen Gerätenummer gekennzeichnet sein. Siehe dazu die Indentifizierungshinweise [▶ 101].

3.3.2 Änderungsmanagement durch Beckhoff

Beckhoff behält sich vor, alle 3 Elemente für sich und unangekündigt zur Produktlebenszeit zu ändern um

- Funktionsverbesserungen einzuführen und
- neue Funktionen zu realisieren

Solch eine Änderung durch Beckhoff

- wird dann i.d.R. durch einen geänderten HW/FW/Revisionsstand gekennzeichnet.
- erfolgt so, dass die Beckhoff EtherCAT IO Kompatibilitätsregel in bestehenden Applikationen eingehalten werden kann:

Geräte-Revision in der Anlage >= Geräte-Revision in der Konfiguration

Hintergrund: Mit der ESI-Beschreibung wird auch das Prozessabbild, die Art der Kommunikation zwischen Master und Slave/Gerät und ggf. Geräte-Funktionen definiert. Damit muss das reale Gerät (inkl. Firmware wenn vorhanden) die Kommunikationsanfragen/-einstellungen des Masters unterstützen. Dies ist abwärtskompatibel der Fall, d.h. neuere Geräte (höhere Revision, höherer

Firmwarestand, höherer Hardwarestand) sollen es auch unterstützen, wenn der EtherCAT Master sie infolge einer bestehenden Konfiguration als eine ältere Revision anspricht. Dies erlaubt im Bedarfsfall den späteren Eintausch/Austausch von Geräten ohne Veränderung der Konfiguration, also ohne Zugriff auf die Applikationsdateien.

Beispiel: Wird in der Konfiguration eine EtherCAT Klemme EL2521-0025-1018 (also Revision -1018) vorgesehen, dann kann als reales IO-Gerät eine Klemme eingesetzt werden, die als EL2521-0025-1018 oder höher (-1019, -1020) programmiert ist. Höhere Revision bedeutet in vielen Fällen auch einen neueren, in jedem Fall aber mindestens gleichen Firmwarestand seitens der Beckhoff-Produktion.
Das bedeutet für Beckhoff und den Anwender, dass die weiterentwickelte oder geänderte Ausgabe eines EtherCAT-IO-Produkts mit weitergezähltem Hardware-, Firmware- oder Revisionsstand alle Eigenschaften und Features der Vorgängerprodukte mit niedrigerem FW- oder Revisionsstand unterstützen sollen.

ESI-Gerätebeschreibungen mit Revisionshistorie

In den auf der Beckhoff Website zum Download angebotenen ESI-XML Dateien (gezippt) sind fast alle jemals veröffentlichten EtherCAT-Geräte mit ihrer gesamten Revisionshistorie enthalten.
Der Anwender hat dadurch die Möglichkeit, auch bewusst Vorgängerrevisionen in seiner Konfiguration einzubinden.

Beckhoff ESI-Gerätebeschreibungen, Kompatibilität

Aus der IO-Kompatibilitätsregel folgt, dass auch wenn in einer neueren Firmware/Revision gegenüber dem Vorgängermodell die Default-Prozessdaten geändert worden sind, dieses Gerät dennoch über die PDO-Einstellung (z. B. PredefinedPDOsettings) auf die „alten“ Prozessdaten umgestellt werden kann oder diese zumindest "versteht", denn diese sollte die neue Firmware/Revision unbedingt unterstützen. Die jeweilige Gerätedokumentation gibt dazu auf der Seite Prozessdaten Hilfestellungen.

3.3.3 Bezug/Update der Elemente

Die 3 Elemente können bezogen werden:

HW:

- **Bezug**: nur durch Erwerbung/Austausch von/durch Beckhoff
- **Update**: Ein Update der Hardware ist üblicherweise nur durch Austausch des Gerätes möglich.

FW:

- **Bezug**: durch Nachfrage bei Beckhoff/Support
- **Update**: Ein Update der Firmware wird in der Regel in der Gerätedokumentation beschrieben und ist mit dem Beckhoff System Manager vorzunehmen. Es ist mit Beckhoff bzw. den Dokumentationsangaben zu klären, welchen möglichen Firmwarestand ein bestimmtes vorliegendes Gerät nach seinem Hardwarestand unterstützt.
  Hinweis/Tipp: es ist der außen am Gerät ablesbare FW-Stand durch den Ausführenden händisch anzupassen, damit der neue FW-Stand auch visuell erkennbar ist

ESI/XML:

- **Bezug**: der aktuelle Stand liegt zum Download auf der Beckhoff Website und beim Erstellen eines neuen TwinCAT Builds werden die zum Erstellungszeitpunkt aktuellen ESI in die TwinCAT Installation integriert. Je nachdem welche TwinCAT Version installiert wird, entsprechen also die nach der TwinCAT Installation unter C:\TwinCAT\IO\ vorgefunden Gerätebeschreibungen ggf. nicht mehr dem aktuellen Beckhoff-Stand.
- **Update**:
  - im System Manager: es müssen die ESI-Dateien (*.xml, *.xsd und ggf. weitere Dateien im gleichen Ordnerpfad) im TwinCAT Stammverzeichnis (z. B. TwinCAT 2: C:\TwinCAT\IO\) ausgetauscht werden. Danach ist ein Neustart des System Manager erforderlich.

- im Gerät: Ein Update der Revision wird in der Regel in der Gerätedokumentation beschrieben und ist mit dem Beckhoff System Manager vorzunehmen. Es ist mit Beckhoff bzw. den Dokumentationsangaben zu klären, welchen möglichen Revisionsstand ein bestimmtes vorliegendes Gerät nach seinem Hardwarestand unterstützt.
  Hinweis/Tipp: es ist der außen am Gerät ablesbare Revisionsstand durch den Ausführenden händisch anzupassen damit der neue Revisionsstand auch visuell erkennbar ist

"Never touch a running system!"

Ein Update von Firmware oder Revision sollte nur bei begründetem Anlass vorgenommen werden.
Wenn beides geändert wird, ist in der Regel zuerst die Firmware, dann die XML/Revision im EEPROM zu ändern. Hinweise in der Dokumentation sind zu beachten!

Ein Anspruch auf Herausgabe von Informationen zu HW/FW/Revisionssänderungen besteht gegenüber Beckhoff nicht.

Download ESI-Beschreibungen

Die Firma Beckhoff stellt auf Ihrer Website den aktuellen Stand der ESI-Gerätebeschreibungen zum Download zur Verfügung. Damit stehen diese ESI jedem EtherCAT-Anwender, jedem Konfigurationsersteller und jedem Kunden zur Verfügung. Es ist aber empfehlenswert, dass Konfigurationsersteller/Programmierer intern in Ihrer Firma/Betrieb/Konzern klären welche Revisionsstände zur firmeninternen Anwendung freigegeben werden sollen, damit in ausgelieferten Maschinen über die Jahre ein einheitlicher Konfigurationsstand gepflegt werden kann. Dies vereinfacht die Ersatzteilhaltung.

3.3.4 Applikations-Projektierung im TwinCAT System Manager 2.x/3.x - Erstellen der Konfiguration

Eine EtherCAT Konfiguration kann im TwinCAT System Manager auf 3 Wegen erstellt werden:

- durch manuelle Konfigurationserstellung ohne vorhandenes Gerät, sog. Offline-Erstellung → siehe Kapitel „Konfigurationsserstellung – Manuell [▶ 129]“
- durch Online-Scannen des mit dem Master verbundenen Gerätes --> siehe Kapitel „Konfigurationsserstellung – OnlineScan [▶ 134]“
- durch automatisierte Konfigurationserstellung ohne Bedienereingriff z. B. über AutomationInterface

Der TwinCAT System Manager 2.x und TwinCAT 3.0/3.1 verhält sich wie folgt:

- **Offline-Erstellung:** Kapitel „Konfigurationsserstellung – Manuell [▶ 129]“

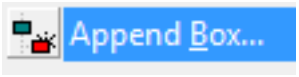


Abb. 127: Kontextmenü „Box anfügen“

In der Auswahlmaske für das einzufügende IO-Gerät bietet der System Manager primär derzeit nur **die höchste/letzte Revision** zum Einfügen an, die auf dem lokalen Programmiergerät im ESI-Verzeichnis vorhanden ist.

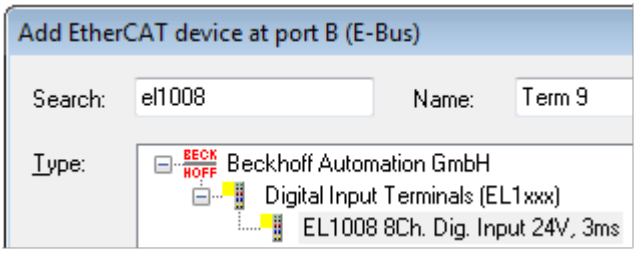


Abb. 128: Auswahl und Anfügen eines EtherCAT-Geräts

Hier zum Beispiel die EL1008.

Um die Revision einzusehen, ist die entsprechende CheckBox zu setzen.

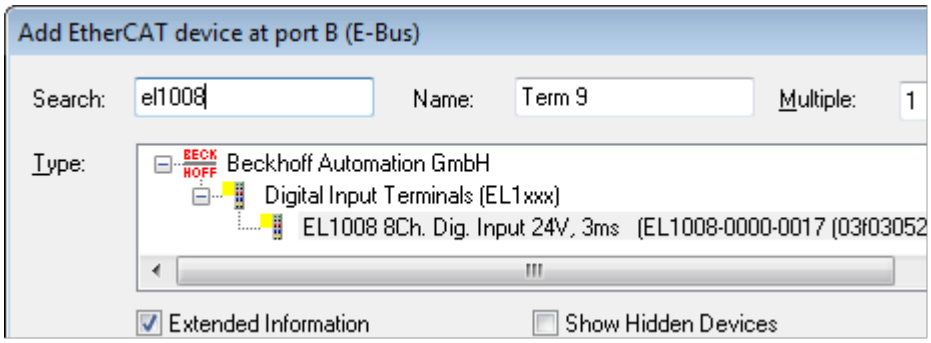


Abb. 129: Checkbox „Extended Information“ setzen für Revisionsanzeige

Soll eine andere/niedrigere Revision in die Konfiguration eingesetzt werden, ist diese über ShowHiddenDevices anzuzeigen und dann auszuwählen.
Bitte Beckhoff Kompatibilitätsregel beachten.

- **Online-Scan:** Kapitel „Konfigurationsserstellung – OnlineScan [▶ 134]“

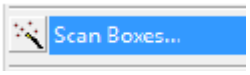


Abb. 130: Kontextmenü „Scan Boxes“

TwinCAT liest aus dem gefunden IO-Gerät Hersteller, Name und Revision aus dem ESI-EEPROM aus, z. B. "Beckhoff EL2502-0000-0018". Dann sucht der System Manager in seinem ESI-Verzeichnis nach der zugehörigen ESI-Datei. Falls diese nicht vorhanden ist, kann der System Manager aus dem Gerät die Online-Beschreibung auslesen und verwenden (dies wird nicht empfohlen, ESi-Beschreibung in der technischen Einordnung und Scan-Seite beachten!).
Es wird in der Konfiguration dann genau diese Revision mit ihren Eigenschaften (Prozessdaten, CoE-Verzeichnis etc.) eingebunden (Beckhoff Kompatibilitätsregel ist zu beachten).
Um alle Funktionen und Diagnosemöglichkeiten des IO-Geräts zu unterstützen wird Anwendern dringend empfohlen, beim Nichtvorhandensein der entsprechenden ESI-Datei den Vorgang abzubrechen und vom Gerätehersteller die entsprechende ESI-Datei zu beschaffen und im ESI-Verzeichnis zu hinterlegen.

- **Automatisierte Erstellung:**
  es wird diejenige Revision in die Konfiguration eingebaut, die im Steuer-Programm hinterlegt ist. Ansonsten gelten vorgehende Sätze.

HINWEIS

Revision und Funktionen des EtherCAT Gerätes

Beim Erstellen einer Konfiguration ist darauf zu achten, dass nur Gerätefunktionen und damit im Endeffekt Revisionen konfiguriert werden, die von den kundenseitig vorliegenden Klemmen (z. B. im Ersatzteilbestand) unterstützt werden.

3.3.5 Konfigurationsvergleich im Projekt

Der TwinCAT System Manager erlaubt einen Vergleich zwischen der Konfiguration laut *.tsm-Datei und der tatsächlich mit dem EtherCAT-Master verbundenen IO-Zusammenstellung. Der Vergleich „Compare“ wird automatisch vorgenommen, wenn beim Scan-Vorgang Geräte vorgefunden werden.

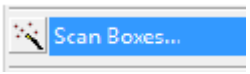


Abb. 131: Kontextmenü „Scan Boxes“

In der erscheinenden Übersicht werden Unterschiede aufgezeigt, weitere Hinweise dazu siehe Kapitel „Konfigurationsserstellung – OnlineScan [▶ 134]“

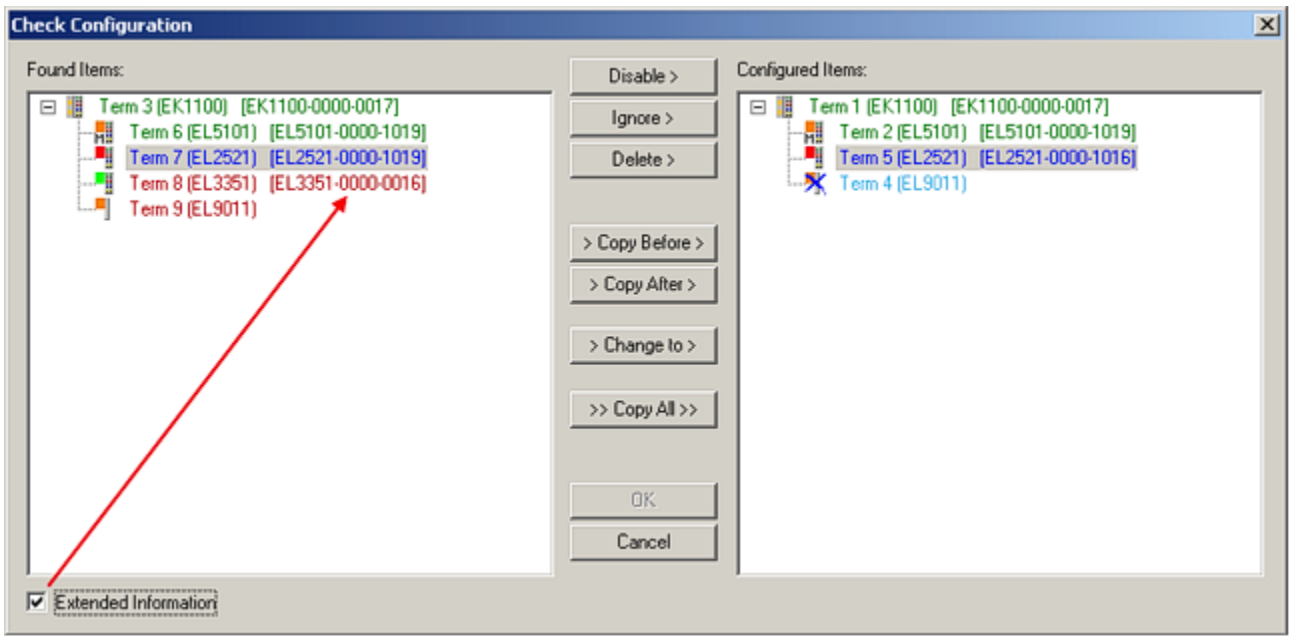


Abb. 132: Konfigurationsvergleich

3.3.6 Installationsstand in der Applikation ermitteln

Wenn eine Übersicht über die real in der Applikation installierten EtherCAT-IO-Komponenten gewünscht wird

- können die aufgedruckten Bezeichnungen und Seriennummern der Klemmen/Boxen abgelesen werden.
- können ab TwinCAT 2.11R3 b2235 über den System Manager online die Informationen HW/FW-Stand und Produktionsdatum aus dem ESI-EEPROM ausgelesen werden.

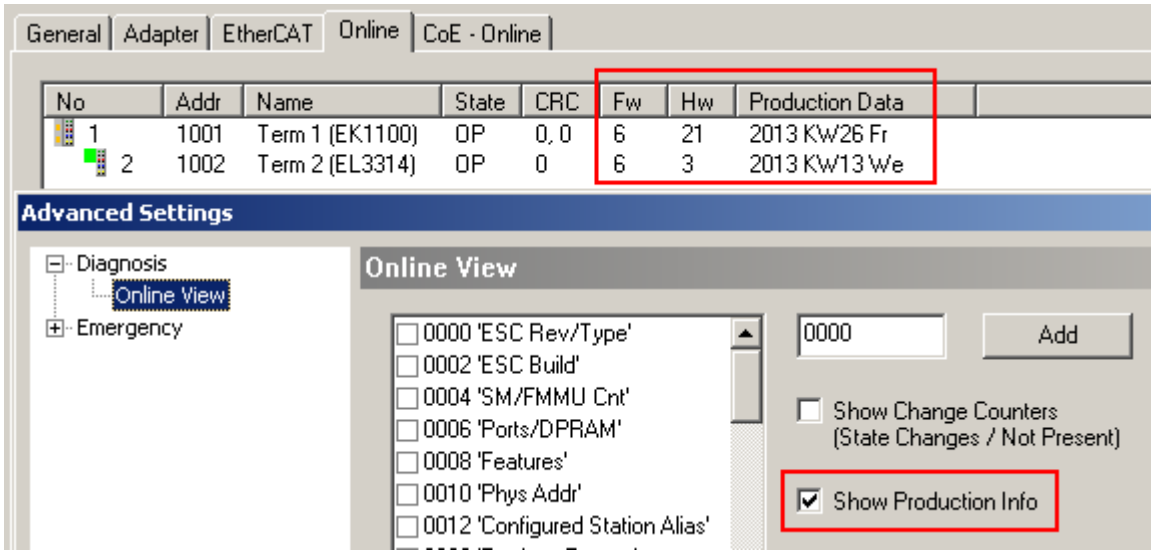


Abb. 133: Auslesen von Produktionsinformationen aus EEPROM

Bausteine zum Auslesen dieser Informationen aus der PLC sind in Vorbereitung.
Ein Export dieser Informationen nach Excel ist möglich:

Abb. 134: Export der Informationen

siehe dazu auch das Kapitel "Versionsidentifikation EtherCAT Geräte - online" [▶ 114]

- können per Online-ADS-Zugriff aus Geräten mit CoE-Verzeichnis die Informationen des Identity-Objekts in die PLC ausgelesen werden

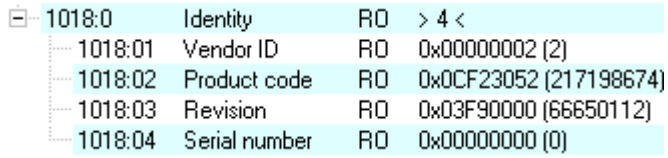


Abb. 135: Auslesen der Informationen aus Identity-Objekt 1018

Siehe dazu auch das Beispielprogramm im Kapitel "Versionsidentifikation EtherCAT Geräte - online" [▶ 114].

3.4 Versionsidentifikation von EtherCAT-Geräten

3.4.1 Allgemeine Hinweise zur Kennzeichnung

Bezeichnung

Ein Beckhoff EtherCAT-Gerät hat eine 14stellige technische Bezeichnung, die sich zusammensetzt aus

- Familienschlüssel
- Typ
- Version
- Revision

Beispiel	Familie	Typ	Version	Revision
EL3314-0000-0016	EL-Klemme (12 mm, nicht steckbare Anschlussebene)	3314 (4 kanalige Thermoelementklemme)	0000 (Grundtyp)	0016
ES3602-0010-0017	ES-Klemme (12 mm, steckbare Anschlussebene)	3602 (2 kanalige Spannungsmessung)	0010 (Hochpräzise Version)	0017
CU2008-0000-0000	CU-Gerät	2008 (8 Port FastEthernet Switch)	0000 (Grundtyp)	0000

Hinweise

- die oben genannten Elemente ergeben die **technische Bezeichnung**, im Folgenden wird das Beispiel EL3314-0000-0016 verwendet.
- Davon ist EL3314-0000 die Bestellbezeichnung, umgangssprachlich bei „-0000“ dann oft nur EL3314 genannt. „-0016“ ist die EtherCAT-Revision.
- Die **Bestellbezeichnung** setzt sich zusammen aus
- Familienschlüssel (EL, EP, CU, ES, KL, CX, ...)
- Typ (3314)
- Version (-0000)
- Die **Revision** -0016 gibt den technischen Fortschritt wie z. B. Feature-Erweiterung in Bezug auf die EtherCAT Kommunikation wieder und wird von Beckhoff verwaltet.
Prinzipiell kann ein Gerät mit höherer Revision ein Gerät mit niedrigerer Revision ersetzen, wenn nicht anders z. B. in der Dokumentation angegeben.
Jeder Revision zugehörig und gleichbedeutend ist üblicherweise eine Beschreibung (ESI, EtherCAT Slave Information) in Form einer XML-Datei, die zum Download auf der Beckhoff Webseite bereitsteht.
Die Revision wird seit 2014/01 außen auf den IP20-Klemmen aufgebracht, siehe Abb. *„EL5021 EL-Klemme, Standard IP20-IO-Gerät mit Chargennummer und Revisionskennzeichnung (seit 2014/01)“*.
- Typ, Version und Revision werden als dezimale Zahlen gelesen, auch wenn sie technisch hexadezimal gespeichert werden.

3.4.2 Versionsidentifikation von EK Kopplern

Als Seriennummer/Date Code bezeichnet Beckhoff im IO-Bereich im Allgemeinen die 8-stellige Nummer, die auf dem Gerät aufgedruckt oder auf einem Aufkleber angebracht ist. Diese Seriennummer gibt den Bauzustand im Auslieferungszustand an und kennzeichnet somit eine ganze Produktions-Charge, unterscheidet aber nicht die Module einer Charge.

Aufbau der Seriennummer: **KK YY FF HH**	Beispiel mit Seriennummer 12 06 3A 02:
KK - Produktionswoche (Kalenderwoche)	12 - Produktionswoche 12
YY - Produktionsjahr	06 - Produktionsjahr 2006
FF - Firmware-Stand	3A - Firmware-Stand 3A
HH - Hardware-Stand	02 - Hardware-Stand 02

Abb. 136: EK1101 EtherCAT Koppler mit Revision 0815 und Seriennummer 41130206

3.4.3 Versionsidentifikation von EL Klemmen

Als Seriennummer/Date Code bezeichnet Beckhoff im IO-Bereich im Allgemeinen die 8-stellige Nummer, die auf dem Gerät aufgedruckt oder auf einem Aufkleber angebracht ist. Diese Seriennummer gibt den Bauzustand im Auslieferungszustand an und kennzeichnet somit eine ganze Produktions-Charge, unterscheidet aber nicht die Module einer Charge.

Aufbau der Seriennummer: **KK YY FF HH**	Beispiel mit Seriennummer 12 06 3A 02:
KK - Produktionswoche (Kalenderwoche)	12 - Produktionswoche 12
YY - Produktionsjahr	06 - Produktionsjahr 2006
FF - Firmware-Stand	3A - Firmware-Stand 3A
HH - Hardware-Stand	02 - Hardware-Stand 02

Abb. 137: EL2872 mit Revision 0022 und Seriennummer 01200815

3.4.4 Versionsidentifikation von ELX Klemmen

Als Seriennummer/Date Code bezeichnet Beckhoff im IO-Bereich im Allgemeinen die 8-stellige Nummer, die auf dem Gerät aufgedruckt oder auf einem Aufkleber angebracht ist. Diese Seriennummer gibt den Bauzustand im Auslieferungszustand an und kennzeichnet somit eine ganze Produktions-Charge, unterscheidet aber nicht die Module einer Charge.

Aufbau der Seriennummer: **KK YY FF HH**	Beispiel mit Seriennummer 12 06 3A 02:
KK - Produktionswoche (Kalenderwoche)	12 - Produktionswoche 12
YY - Produktionsjahr	06 - Produktionsjahr 2006
FF - Firmware-Stand	3A - Firmware-Stand 3A
HH - Hardware-Stand	02 - Hardware-Stand 02

Abb. 138: ELX2002 mit Beckhoff Traceability Number (BTN) 10000030 und Date Code 01200815

3.4.5 Versionsidentifikationen von ELM Klemmen

Als Seriennummer/Date Code bezeichnet Beckhoff im IO-Bereich im Allgemeinen die 8-stellige Nummer, die auf dem Gerät aufgedruckt oder auf einem Aufkleber angebracht ist. Diese Seriennummer gibt den Bauzustand im Auslieferungszustand an und kennzeichnet somit eine ganze Produktions-Charge, unterscheidet aber nicht die Module einer Charge.

Aufbau der Seriennummer: **KK YY FF HH**

KK - Produktionswoche (Kalenderwoche)
YY - Produktionsjahr
FF - Firmware-Stand
HH - Hardware-Stand

Beispiel mit Seriennummer 12 06 3A 02:

12 - Produktionswoche 12
06 - Produktionsjahr 2006
3A - Firmware-Stand 3A
02 - Hardware-Stand 02

Abb. 139: ELM3002-0000 mit eindeutiger BTN 0000wwww und Seriennummer 09200506

3.4.6 Versionsidentifikation von EJ Modulen

Produktgruppe	Beispiel		
	Produktbezeichnung	Version	Revision
EtherCAT-Koppler EJ110x	EJ1101	-0022 (Koppler mit externen Steckern, Netzteil und optionalen ID-Switchen)	-0016
Digital-Eingangs-Module EJ1xxx	EJ1008 8-kanalig	-0000 (Grundtyp)	-0017
Digital-Ausgangs-Module EJ2xxx	EJ2521 1-kanalig	-0224 (2 x 24 V Ausgänge)	-0016
Analog-Eingangs-Module EJ3xxx	EJ3318 8-kanaliges Thermoelement	-0000 (Grundtyp)	-0017
Analog-Ausgangs-Module EJ4xxx	EJ1434 4-kanalig	-0000 (Grundtyp)	-0019
Sonderfunktions-Module EJ5xxx, EJ6xxx	EJ6224 IO-Link-Master	-0090 (mit TwinSAFE SC)	-0016
Motor-Module EJ7xxx	EJ7211 Servomotorendstufe	-9414 (mit OCT, STO und TwinSAFE SC)	-0029

Bezeichnung

Beckhoff EtherCAT-Steckmodule verfügen über eine 14-stellige **technische Bezeichnung**, die sich wie folgt zusammensetzt (z. B. EJ1008-0000-0017):

- **Bestellbezeichnung**:
  - Familienschlüssel: EJ
  - Produktbezeichnung: Die erste Stelle der Produktbezeichnung dient der Zuordnung zu einer Produktgruppe (z. B. EJ2xxx = Digital - Ausgangsmodul)
  - Versionsnummer: Die vierstellige Versionsnummer kennzeichnet verschiedene Produktvarianten
- **Revisionsnummer:**
  Sie wird bei Änderungen am Produkt hochgezählt.

Die Bestellbezeichnung und Revisionsnummer werden auf der Seite der EtherCAT-Steckmodule aufgebracht, siehe folgende Abbildung (A und B).

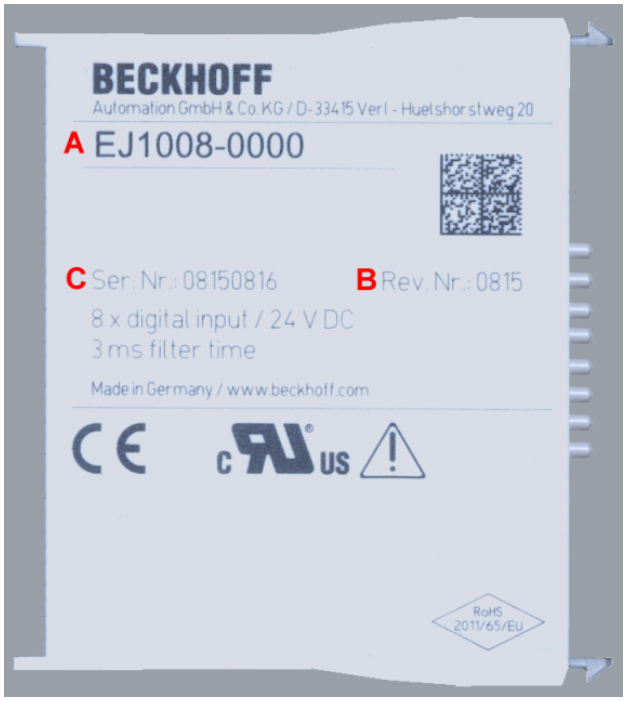


Abb. 140: Bestellbezeichnung (A), Revisionsnummer (B) und Seriennummer (C) am Beispiel EJ1008

Hinweise

- die oben genannten Elemente ergeben die **technische Bezeichnung**, im Folgenden wird das Beispiel EJ1008-0000-0017 verwendet.
- Davon ist EJ1008-0000 die **Bestellbezeichnung**, umgangssprachlich bei „-0000“ dann oft nur EJ1008 genannt.
- Die **Revision** -0017 gibt den technischen Fortschritt wie z. B. Feature-Erweiterung in Bezug auf die EtherCAT Kommunikation wieder und wird von Beckhoff verwaltet.
  Prinzipiell kann ein Gerät mit höherer Revision ein Gerät mit niedrigerer Revision ersetzen, wenn nicht anders z. B. in der Dokumentation angegeben.
  Jeder Revision zugehörig und gleichbedeutend ist üblicherweise eine Beschreibung (ESI, **E**therCAT **S**lave **I**nformation) in Form einer XML-Datei, die zum Download auf der Beckhoff Webseite bereitsteht.
  Die Revision wird auf der Seite der EtherCAT-Steckmodule aufgebracht, siehe folgende Abbildung.
- Produktbezeichnung, Version und Revision werden als dezimale Zahlen gelesen, auch wenn sie technisch hexadezimal gespeichert werden.

Seriennummer

Die 8-stellige Seriennummer ist auf dem EtherCAT-Steckmodul auf der Seite aufgedruckt (s. folgende Abb. C). Diese Seriennummer gibt den Bauzustand im Auslieferungszustand an und kennzeichnet somit eine ganze Produktions-Charge, unterscheidet aber nicht die Module einer Charge.

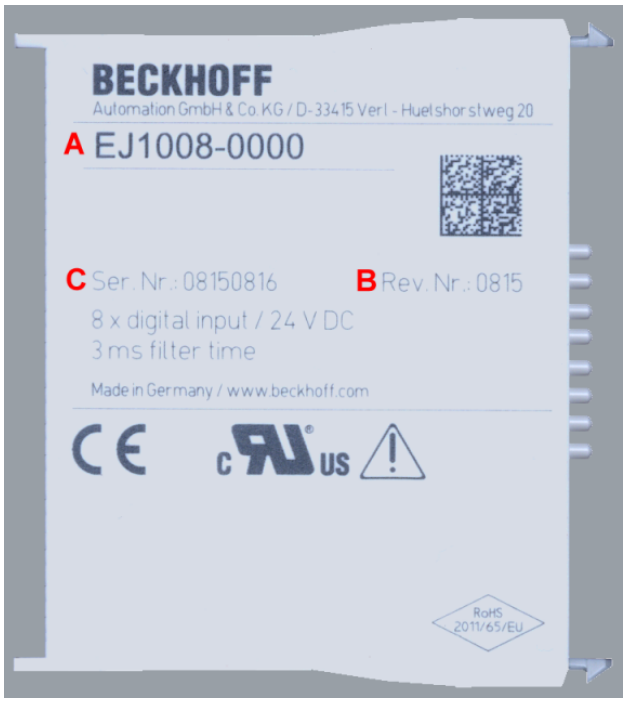


Abb. 141: Bestellbezeichnung (A), Revisionsnummer (B) und Seriennummer (C) am Beispiel EJ1008

Seriennummer	Beispiel Seriennummer: 08 15 08 16
KK - Produktionswoche (Kalenderwoche)	08 - Produktionswoche 08
YY - Produktionsjahr	15 - Produktionsjahr 2015
FF - Firmware-Stand	08 - Firmware-Stand 08
HH - Hardware-Stand	16 - Hardware-Stand 16

3.4.7 Versionsidentifikation von EP/EPI/EPP/ER/ERI Boxen

Als Seriennummer/Date Code bezeichnet Beckhoff im IO-Bereich im Allgemeinen die 8-stellige Nummer, die auf dem Gerät aufgedruckt oder auf einem Aufkleber angebracht ist. Diese Seriennummer gibt den Bauzustand im Auslieferungszustand an und kennzeichnet somit eine ganze Produktions-Charge, unterscheidet aber nicht die Module einer Charge.

Aufbau der Seriennummer: **KK YY FF HH**	Beispiel mit Seriennummer 12 06 3A 02:
KK - Produktionswoche (Kalenderwoche)	12 - Produktionswoche 12
YY - Produktionsjahr	06 - Produktionsjahr 2006
FF - Firmware-Stand	3A - Firmware-Stand 3A
HH - Hardware-Stand	02 - Hardware-Stand 02

Ausnahmen können im **IP67-Bereich** auftreten, dort kann folgende Syntax verwendet werden (siehe jeweilige Gerätedokumentation):

Syntax: D ww yy x y z u

D - Vorsatzbezeichnung
ww - Kalenderwoche
yy - Jahr
x - Firmware-Stand der Busplatine
y - Hardware-Stand der Busplatine
z - Firmware-Stand der E/A-Platine
u - Hardware-Stand der E/A-Platine

Beispiel: D.22081501 Kalenderwoche 22 des Jahres 2008 Firmware-Stand Busplatine: 1 Hardware Stand Busplatine: 5 Firmware-Stand E/A-Platine: 0 (keine Firmware für diese Platine notwendig) Hardware-Stand E/A-Platine: 1

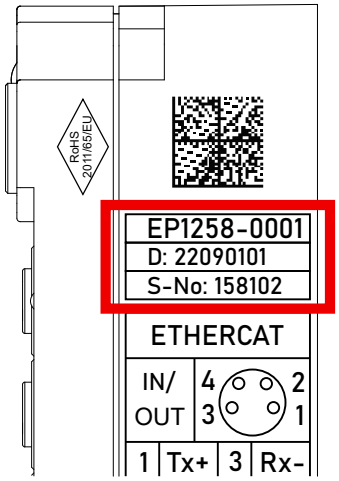


Abb. 142: EP1258-0001 IP67 EtherCAT Box mit Chargennummer/ DateCode 22090101 und eindeutiger Seriennummer 158102

3.4.8 Versionsidentifikation von CU Switches

Als Seriennummer/Date Code bezeichnet Beckhoff im IO-Bereich im Allgemeinen die 8-stellige Nummer, die auf dem Gerät aufgedruckt oder auf einem Aufkleber angebracht ist. Diese Seriennummer gibt den Bauzustand im Auslieferungszustand an und kennzeichnet somit eine ganze Produktions-Charge, unterscheidet aber nicht die Module einer Charge.

Aufbau der Seriennummer: **KK YY FF HH**	Beispiel mit Seriennummer 12 06 3A 02:
KK - Produktionswoche (Kalenderwoche)	12 - Produktionswoche 12
YY - Produktionsjahr	06 - Produktionsjahr 2006
FF - Firmware-Stand	3A - Firmware-Stand 3A
HH - Hardware-Stand	02 - Hardware-Stand 02

Abb. 143: CU1521 mit der Seriennummer 4820/ und der eindeutigen Beckhoff Traceability Number (BTN) 00007su0

3.4.9 Beckhoff Identification Code (BIC)

Der Beckhoff Identification Code (BIC) wird vermehrt auf Beckhoff-Produkten zur eindeutigen Identitätsbestimmung des Produkts aufgebracht. Der BIC ist als Data Matrix Code (DMC, Code-Schema ECC200) dargestellt, der Inhalt orientiert sich am ANSI-Standard MH10.8.2-2016.

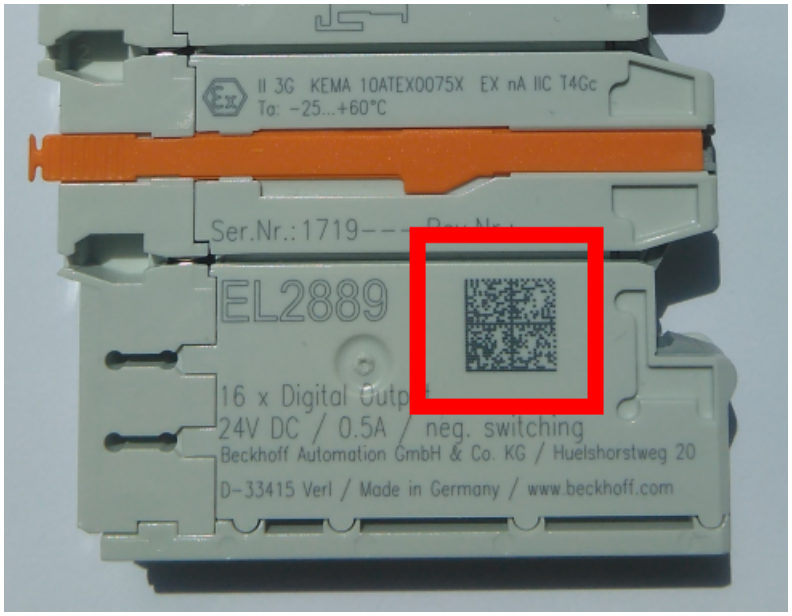


Abb. 144: BIC als Data Matrix Code (DMC, Code-Schema ECC200)

Die Einführung des BIC erfolgt schrittweise über alle Produktgruppen hinweg.
Er ist je nach Produkt an folgenden Stellen zu finden:

- auf der Verpackungseinheit
- direkt auf dem Produkt (bei ausreichendem Platz)
- auf Verpackungseinheit und Produkt

Der BIC ist maschinenlesbar und enthält Informationen, die auch kundenseitig für Handling und Produktverwaltung genutzt werden können.

Jede Information ist anhand des so genannten Datenidentifikators (ANSI MH10.8.2-2016) eindeutig identifizierbar. Dem Datenidentifikator folgt eine Zeichenkette. Beide zusammen haben eine maximale Länge gemäß nachstehender Tabelle. Sind die Informationen kürzer, werden sie um Leerzeichen ergänzt.

Folgende Informationen sind möglich, die Positionen 1 bis 4 sind immer vorhanden, die weiteren je nach Produktfamilienbedarf:

Pos-Nr.	Art der Information	Erklärung	Datenidentifikator	Anzahl Stellen inkl. Datenidentifikator	Beispiel
1	Beckhoff-Artikelnummer	**Beckhoff - Artikelnummer**	1P	8	1P072222
2	Beckhoff Traceability Number (BTN)	**Eindeutige Seriennummer, Hinweis s. u.**	SBTN	12	SBTNk4p562d7
3	Artikelbezeichnung	**Beckhoff Artikelbezeichnung, z. B. EL1008**	1K	32	1KEL1809
4	Menge	**Menge in Verpackungseinheit, z. B. 1, 10…**	Q	6	Q1
5	Chargennummer	Optional: Produktionsjahr und -woche	2P	14	2P401503180016
6	ID-/Seriennummer	Optional: vorheriges Seriennummer-System, z. B. bei Safety-Produkten oder kalibrierten Klemmen	51S	12	51S678294
7	Variante	Optional: Produktvarianten-Nummer auf Basis von Standardprodukten	30P	32	30PF971, 2*K183
...					

Weitere Informationsarten und Datenidentifikatoren werden von Beckhoff verwendet und dienen internen Prozessen.

Aufbau des BIC

Beispiel einer zusammengesetzten Information aus den Positionen 1 bis 4 und dem o.a. Beispielwert in Position 6. Die Datenidentifikatoren sind in Fettschrift hervorgehoben:

1P072222**S**BTNk4p562d7**1K**EL1809 **Q**1 **51S**678294

Entsprechend als DMC:

Abb. 145: Beispiel-DMC **1P**072222**S**BTNk4p562d7**1K**EL1809 **Q**1 **51S**678294

BTN

Ein wichtiger Bestandteil des BICs ist die Beckhoff Traceability Number (BTN, Pos.-Nr. 2). Die BTN ist eine eindeutige, aus acht Zeichen bestehende Seriennummer, die langfristig alle anderen Seriennummern-Systeme bei Beckhoff ersetzen wird (z. B. Chargenbezeichnungen auf IO-Komponenten, bisheriger Seriennummernkreis für Safety-Produkte, etc.). Die BTN wird ebenfalls schrittweise eingeführt, somit kann es vorkommen, dass die BTN noch nicht im BIC codiert ist.

HINWEIS

Diese Information wurde sorgfältig erstellt. Das beschriebene Verfahren wird jedoch ständig weiterentwickelt. Wir behalten uns das Recht vor, Verfahren und Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern. Aus den Angaben, Abbildungen und Beschreibungen in dieser Information können keine Ansprüche auf Änderung geltend gemacht werden.

3.4.10 Elektronischer Zugriff auf den BIC (eBIC)

Elektronischer BIC (eBIC)

Der Beckhoff Identification Code (BIC) wird auf Beckhoff Produkten außen sichtbar aufgebracht. Er soll wo möglich, auch elektronisch auslesbar sein.

Für die elektronische Auslesung ist die Schnittstelle entscheidend, über die das Produkt elektronisch angesprochen werden kann.

K-Bus Geräte (IP20, IP67)

Für diese Geräte sind derzeit keine elektronische Speicherung und Auslesung geplant.

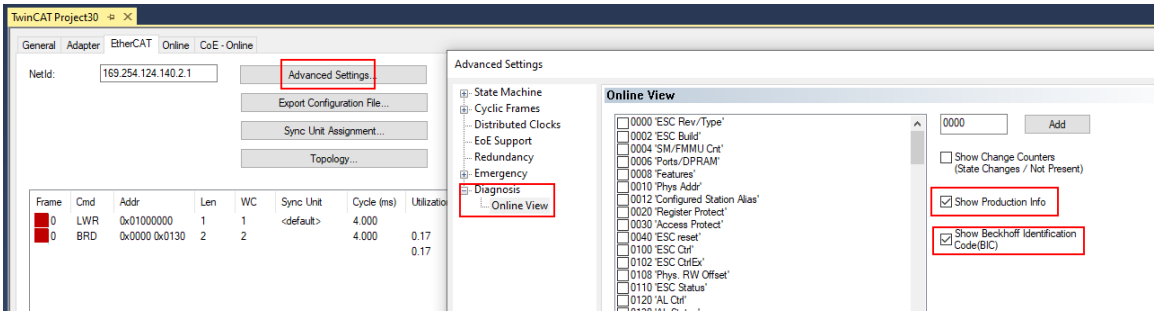
EtherCAT Geräte (P20, IP67)

Alle Beckhoff EtherCAT Geräte haben ein sogenanntes ESI-EEPROM, das die EtherCAT-Identität mit der Revision beinhaltet. Darin wird die EtherCAT-Slave-Information gespeichert, umgangssprachlich auch als ESI/XML-Konfigurationsdatei für den EtherCAT-Master bekannt. Zu den Zusammenhängen siehe die entsprechenden Kapitel im EtherCAT-Systemhandbuch (Link).

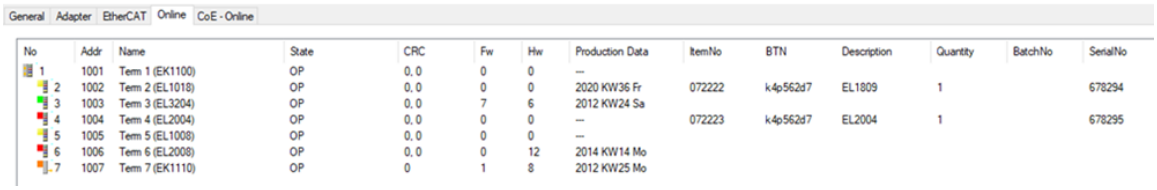
In das ESI-EEPROM wird auch die eBIC gespeichert. Die Einführung des eBIC in die Beckhoff IO Produktion (Klemmen, Boxen) erfolgt ab 2020; mit einer weitgehenden Umsetzung ist in 2021 zu rechnen.

Anwenderseitig ist die eBIC (wenn vorhanden) wie folgt elektronisch zugänglich:

- Bei allen EtherCAT Geräten kann der EtherCAT Master (TwinCAT) den eBIC aus dem ESI-EEPROM auslesen
  - Ab TwinCAT 3.1 build 4024.11 kann der eBIC im Online-View angezeigt werden.
  - Dazu unter
    EtherCAT → Erweiterte Einstellungen → Diagnose das Kontrollkästchen „Show Beckhoff Identification Code (BIC)“ aktivieren:



  - Die BTN und Inhalte daraus werden dann angezeigt:



  - Hinweis: ebenso können wie in der Abbildung zu sehen die seit 2012 programmierten Produktionsdaten HW-Stand, FW-Stand und Produktionsdatum per „Show Production Info“ angezeigt werden.
  - Ab TwinCAT 3.1. build 4024.24 stehen in der Tc2_EtherCAT Library ab v3.3.19.0 die Funktionen *FB_EcReadBIC* und *FB_EcReadBTN* zum Einlesen in die PLC und weitere eBIC-Hilfsfunktionen zur Verfügung.
- Bei EtherCAT Geräten mit CoE-Verzeichnis kann zusätzlich das Objekt 0x10E2:01 zur Anzeige der eigenen eBIC genutzt werden, hier kann auch die PLC einfach auf die Information zugreifen:

◦ Das Gerät muss zum Zugriff in SAFEOP/OP sein:

Index	Name	Flags	Value
1000	Device type	RO	0x015E1389 (22942601)
1008	Device name	RO	ELM3704-0000
1009	Hardware version	RO	00
100A	Software version	RO	01
100B	Bootloader version	RO	J0.1.27.0
1011:0	Restore default parameters	RO	> 1 <
1018:0	Identity	RO	> 4 <
10E2:0	Manufacturer-specific Identification C...	RO	> 1 <
10E2:01	SubIndex 001	RO	1P158442SBTN0008jekp1KELM3704 Q1 2P482001000016
10F0:0	Backup parameter handling	RO	> 1 <
10F3:0	Diagnosis History	RO	> 21 <
10F8	Actual Time Stamp	RO	0x170bfb277e

◦ Das Objekt 0x10E2 wird in Bestandsprodukten vorrangig im Zuge einer notwendigen Firmware-Überarbeitung eingeführt.

◦ Ab TwinCAT 3.1. build 4024.24 stehen in der Tc2_EtherCAT Library ab v3.3.19.0 die Funktionen *FB_EcCoEReadBIC* und *FB_EcCoEReadBTN* zum Einlesen in die PLC und weitere eBIC-Hilfsfunktionen zur Verfügung.

- Hinweis: bei elektronischer Weiterverarbeitung ist die BTN als String(8) zu behandeln, der Identifier „SBTN“ ist nicht Teil der BTN.
- Technischer Hintergrund
  Die neue BIC Information wird als Category zusätzlich bei der Geräteproduktion ins ESI-EEPROM geschrieben. Die Struktur des ESI-Inhalts ist durch ETG Spezifikationen weitgehend vorgegeben, demzufolge wird der zusätzliche herstellerspezifische Inhalt mithilfe einer Category nach ETG.2010 abgelegt. Durch die ID 03 ist für alle EtherCAT Master vorgegeben, dass sie im Updatefall diese Daten nicht überschreiben bzw. nach einem ESI-Update die Daten wiederherstellen sollen.
  Die Struktur folgt dem Inhalt des BIC, siehe dort. Damit ergibt sich ein Speicherbedarf von ca. 50..200 Byte im EEPROM.
- Sonderfälle
  ◦ Sind mehrere ESC in einem Gerät verbaut die hierarchisch angeordnet sind, trägt nur der TopLevel ESC die eBIC Information.
  ◦ Sind mehrere ESC in einem Gerät verbaut die nicht hierarchisch angeordnet sind, tragen alle ESC die eBIC Information gleich.
  ◦ Besteht das Gerät aus mehreren Sub-Geräten mit eigener Identität, aber nur das TopLevel-Gerät ist über EtherCAT zugänglich, steht im CoE-Objekt-Verzeichnis 0x10E2:01 die eBIC des TopLevel-Geräts, in 0x10E2:nn folgen die eBIC der Sub-Geräte.

Profibus/Profinet/DeviceNet... Geräte

Für diese Geräte ist derzeit keine elektronische Speicherung und Auslesung geplant.

3.5 Versionsidentifikation EtherCAT Geräte - online

Die Produktionsinformationen eines EtherCAT Slave Geräts

- Firmwarestand
- Hardwarestand
- Produktionsdatum

sind wie vorhergehend beschrieben durch die seitlich aufgelaserte Seriennummer ablesbar. Firmware/ Hardware (FW/HW)-Stand ist bei "komplexen" Geräten mit CoE-Fähigkeit auch von dort elektronisch auslesbar, jedoch nicht bei einfachen Geräten ohne CoE.

Deshalb wird seit Produktionsdatum Juli 2012 in EtherCAT-Klemmen die o.g. Produktionsinformation von Beckhoff einprogrammiert. Sie befindet sich im EtherCAT ESI EEPROM, das Bestandteil eines jeden Beckhoff EtherCAT Slave ist.

Information	ESI/XML (Konfiguration)	CoE ggf. Default-Daten im Offline Dictionary	ESI EEPROM	seitlicher Aufdruck/Aufkleber
Verfügbarkeit	**alle EtherCAT Slaves**	**nur bei komplexen Geräten**	**alle EtherCAT Slaves**	**alle EtherCAT Slaves**
Device name	x	0x1008	x	x
Hardware version		0x1009	x ("production info")	x
Firmware version		0x100A	x ("production info")	x
Vendor ID (z.B. Beckhoff: 0x2)		0x1018:01	x	
Product code (32 Bit)	x	0x1018:02	x	
Revision (32 Bit)	x	0x1018:03	x	x
Serial number		0x1018:04	x	(x)
ID number		(bei manchen Serien, siehe jeweilige Dokumentation)		(x)
Produktionsdatum			x ("production info")	x

Hinweise zum elektronischen Auslesen

- ESI/XML
  - diese Daten können vom Anwenderprogramm (PLC, non-RT-Task) aus dem verwendeten EtherCAT Master ausgelesen werden.
  - für TwinCAT sind hier die Funktionen aus der TcEtherCAT.lib zu nutzen (andere EtherCAT Master je nach Fähigkeit).
  - die Daten stehen online (mit live Verbindung zum Gerät) und offline zur Verfügung.
- CoE (online)
  - diese Daten können vom Anwenderprogramm (PLC, non-RT-Task) aus dem CoE-verzeichnis des Geräts über EtherCAT ausgelesen werden.
  - für TwinCAT sind hier die Funktionen aus der TcEtherCAT.lib zu nutzen (andere EtherCAT Master je nach CoE-Fähigkeit).
  - die Daten stehen nur online zur Verfügung
    (beim Offline-Zugriff kann ein EtherCAT Master wie TwinCAT fallweise die Daten aus dem ESI-Dictionary der XML-Datei vorlegen - das sind dann aber nur Daten eines vorgesehen Slaves).
- ESI EEPROM

- diese Daten können
  - von einem TwinCAT 2.11R3 ab build 2035 online angezeigt werden
  - vom Anwenderprogramm (PLC, non-RT-Task) aus dem ESI EEPROM des Geräts über EtherCAT ausgelesen werden (siehe Beispielprogramm)
- für TwinCAT sind hier die Funktionen aus der TcEtherCAT.lib zu nutzen (andere EtherCAT Master je nach CoE-Fähigkeit)
- die Daten stehen nur online zur Verfügung

- seitlicher Aufdruck: elektronisch nicht möglich.

Online Production Info (ESI EEPROM)

HINWEIS

Datenspeicherung im ESI EEPROM

Bei einem EEPROM Update des EtherCAT Slave durch den EtherCAT Master z. B. zum Zwecke des Revisionsupdate werden die dort befindlichen Produktionsdaten überschrieben, wenn der EtherCAT Master diese Daten vom Überschreiben nicht ausnimmt. Ab TwinCAT 2.11 überwacht der System Manager beim Update auf diese Daten.

Die Produktionsinformation HW, FW und Datum können im System Manager angezeigt und exportiert werden.

Ab TwinCAT 2.11R3 Build 2035: per Rechtsklick auf der Online-Seite -> Properties

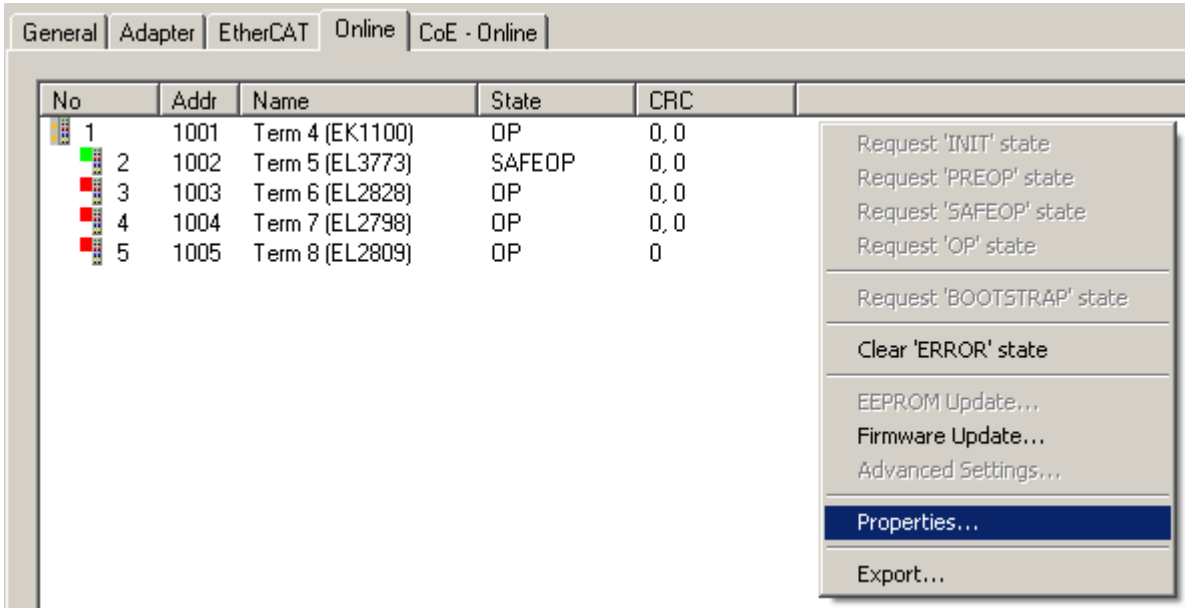


Abb. 146: System Manager Online View -> Properties

Dann „Show Production Info“ aktivieren

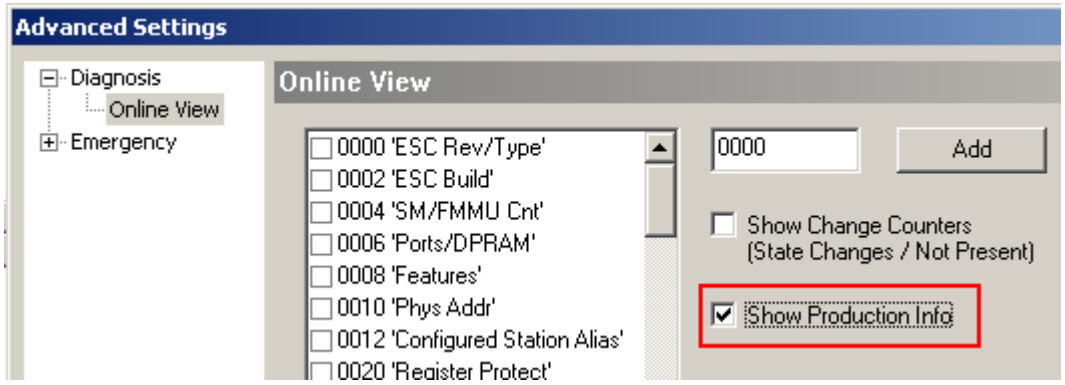


Abb. 147: ShowProductionInfo

Dann werden neue Spalten mit den Informationen eingeblendet

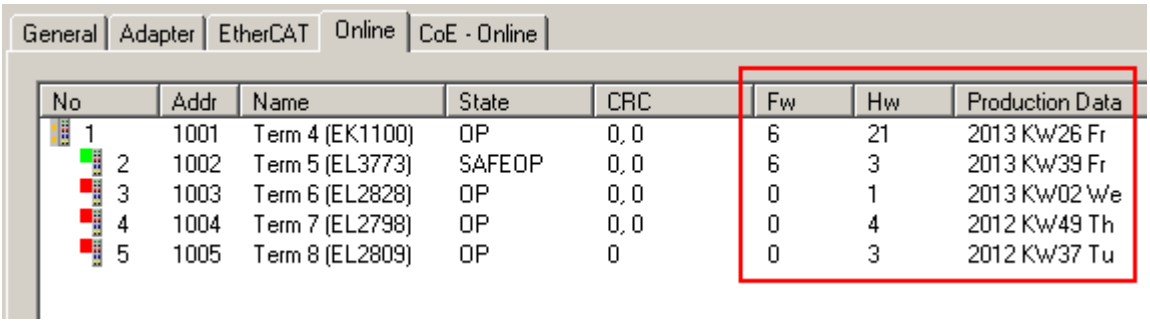


Abb. 148: Online Anzeige

Geräte ohne diese Informationen zeigen in der Übersicht 0 an.

Ab TwinCAT 3.1 erfolgt die Aktivierung der Anzeige über die erweiterten Einstellungen des EtherCAT-Masters:

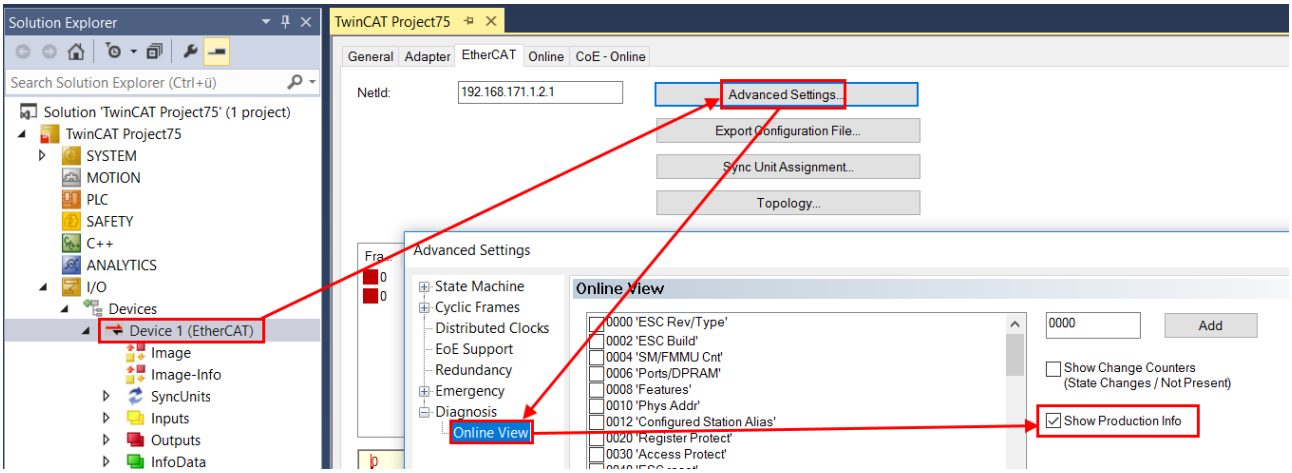


Abb. 149: TwinCAT 3.1: „Show Produktion Info“ aktivieren

Die Daten können dann direkt in eine CSV-Datei exportiert werden.

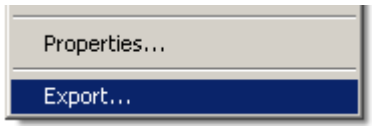


Abb. 150: Export

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Name	PhysAddr	AutoIncAddr	VendorId	ProductionNo	RevisionNo	SerialNo	Hardware Version	Firmware Version	Production Year	Production Week	Production DayOfWeek	State
2	Term 4 (EK1100)	0x3e9	0x0	0x2	0x44c2c52	0x120000	0x0	0x15	0x6	2013	26	Fr	0x8
3	Term 5 (EL3773)	0x3ea	0xffff	0x2	0xebd3052	0x130000	0x0	0x3	0x6	2013	39	Fr	0x4
4	Term 6 (EL2828)	0x3eb	0xfffe	0x2	0xb0c3052	0x100000	0x0	0x1	0x0	2013	2	We	0x8
5	Term 7 (EL2798)	0x3ec	0xfffd	0x2	0xaee3052	0x110000	0x0	0x4	0x0	2012	49	Th	0x8
6	Term 8 (EL2809)	0x3ed	0xfffc	0x2	0xaf93052	0x110000	0x0	0x3	0x0	2012	37	Tu	0x8
7													

Abb. 151: Beispielinhalt der csv-Datei

Beispielprogramm zum Auslesen und Export der "Production Info" aus der PLC

Verwendung der Beispielprogramme

Dieses Dokument enthält exemplarische Anwendungen unserer Produkte für bestimmte Einsatzbereiche. Die hier dargestellten Anwendungshinweise beruhen auf den typischen Eigenschaften unserer Produkte und haben ausschließlich Beispielcharakter. Die mit diesem Dokument vermittelten Hinweise beziehen sich ausdrücklich nicht auf spezifische Anwendungsfälle, daher liegt es in der Verantwortung des Kunden zu prüfen und zu entscheiden, ob das Produkt für den Einsatz in einem bestimmten Anwendungsbereich geeignet ist. Wir übernehmen keine Gewährleistung, dass der in diesem Dokument enthaltene Quellcode vollständig und richtig ist. Wir behalten uns jederzeit eine Änderung der Inhalte dieses Dokuments vor und übernehmen keine Haftung für Irrtümer und fehlenden Angaben.
Das im Folgenden vorgestellte Programm dient als erste Einführung in die Auswertemöglichkeiten der Slave-Daten und Produktionsinformationen. Dem Anwender steht es frei, das Programm nach seinen Vorstellungen zu verändern, oder nur Teile des Codes zu verwenden.

Beispielprogramm (https://infosys.beckhoff.com/content/1031/ethercatsystem/Resources/zip/2469151627.zip)

In diesem Programmbeispiel werden einige Identitätsdaten der verbundenen EtherCAT Slaves ausgelesen und in einer Visualisierung (Abb. *System Manager Online View -> Properties*) angezeigt. Das Auslesen der konfigurierten Daten erfolgt mit den Bausteinen der TcEtherCAT.lib. Onlinedaten wie Hersteller (Vendor) und Produktionsdatum werden durch Zugriff auf das ESI-EEPROM ermittelt. Der Einstieg in die Kommunikation mit dem EEPROM kann durch Verwendung des in diesem Projekt enthaltenen Funktionsblock „FB_ReadEEPROM“ einfach und schnell erfolgen. Zudem besteht eine Exportmöglichkeit, um die Daten in eine Datei der Text-Tabellenformate „*.txt“ und „*.csv“ zu exportieren. In der Option „*.csv“ wird der deutsche Standart mit einem Semikolon als Trennzeichen gewählt.

Abb. 152: Visualisierung im Beispielprogramm

In gängigen Tabellenkalkulationsprogrammen sind diverse Sortier- und Filtereigenschaften auf die exportierte Datei anwendbar, z. B. „Tabelle mit Überschriften“.

	A	B	C	D	E	F	G	H
1	Adress	Name	ConfigType	ProductionCode	RevisionNr	Vendor	ProductionData	2013-10-01-12:00:52.679
2	1001	Terminal 1 (EK1100)	EK1100	72100946	1114112	Beckhoff Automation GmbH	FW: 06, HW: 22, THU, 32/2012	
3	1002	Terminal 2 (EL2032)	EL2032	133181522	1048576	Beckhoff Automation GmbH	FW: 09, HW: 15, TUE, 23/2012	
4	1003	Terminal 3 (EL1024)	EL1024	67121234	1048576	Beckhoff Automation GmbH	FW: 00, HW: 04, MON, 19/2012	
5	1004	Terminal 4 (EL1252-0050)	EL1252-0050	82063442	1048626	Beckhoff Automation GmbH	FW: 00, HW: 00, TUE, 25/2012	
6	1005	Terminal 5 (EL1252)	EL1252	82063442	1376256	Beckhoff Automation GmbH	FW: 01, HW: 06, WED, 09/2013	
7	1006	Terminal 6 (EL1252)	EL1252	82063442	1114112	Beckhoff Automation GmbH	No Productiondata availiable	
8	1007	Terminal 7 (EL1819)	EL1819	119222354	1114112	Beckhoff Automation GmbH	FW: 00, HW: 03, THU, 50/2012	

Abb. 153: Filterung im Tabellenprogramm

3.6 TwinCAT Entwicklungsumgebung

3.6.1 Installation TwinCAT Realtime Treiber

Um einen Standard Ethernet Port einer IPC Steuerung mit den nötigen Echtzeitfähigkeiten auszurüsten, ist der Beckhoff Echtzeit Treiber auf diesem Port unter Windows zu installieren.

Dies kann auf mehreren Wegen vorgenommen werden.

A: Über den TwinCAT Adapter-Dialog

Im System Manager ist über Options → Show realtime Kompatible Geräte die TwinCAT-Übersicht über die lokalen Netzwerkschnittstellen aufzurufen.

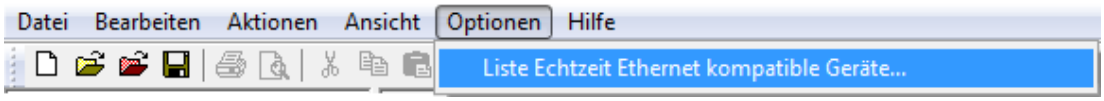


Abb. 154: Aufruf im System Manager (TwinCAT 2)

Unter TwinCAT 3 ist dies über das Menü unter „TwinCAT“ erreichbar:

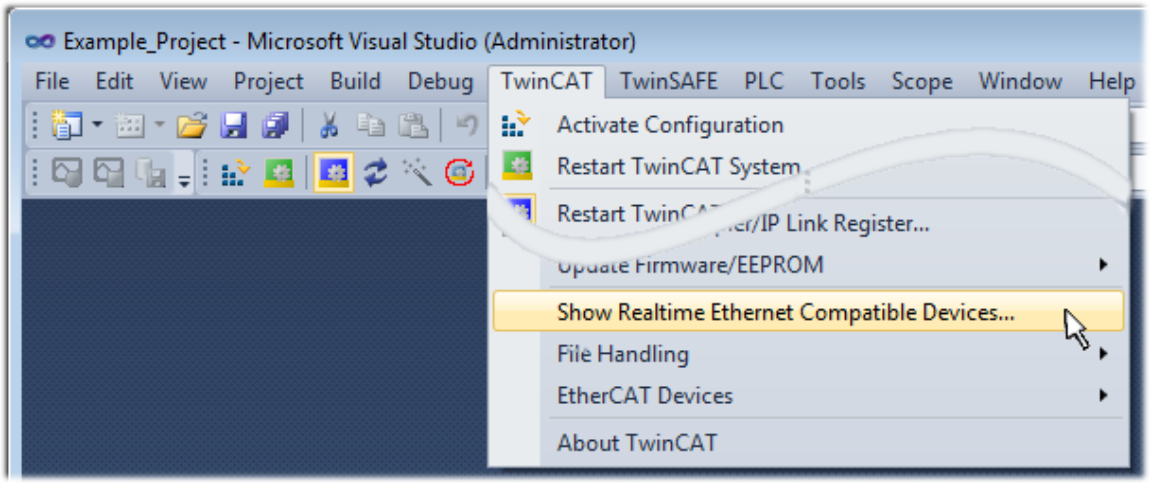


Abb. 155: Aufruf in VS Shell (TwinCAT 3)

B: Über TcRteInstall.exe im TwinCAT-Verzeichnis

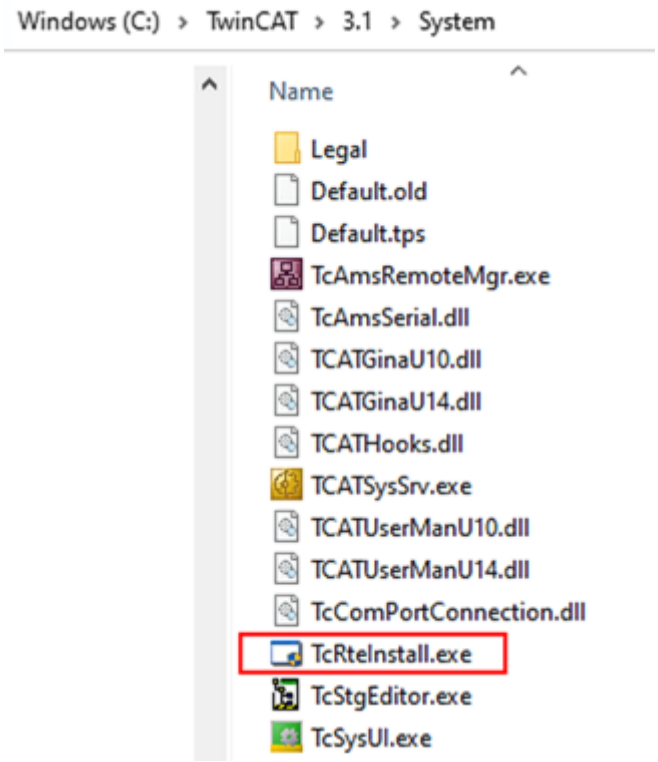


Abb. 156: TcRteInstall.exe im TwinCAT-Verzeichnis

In beiden Fällen erscheint der folgende Dialog:

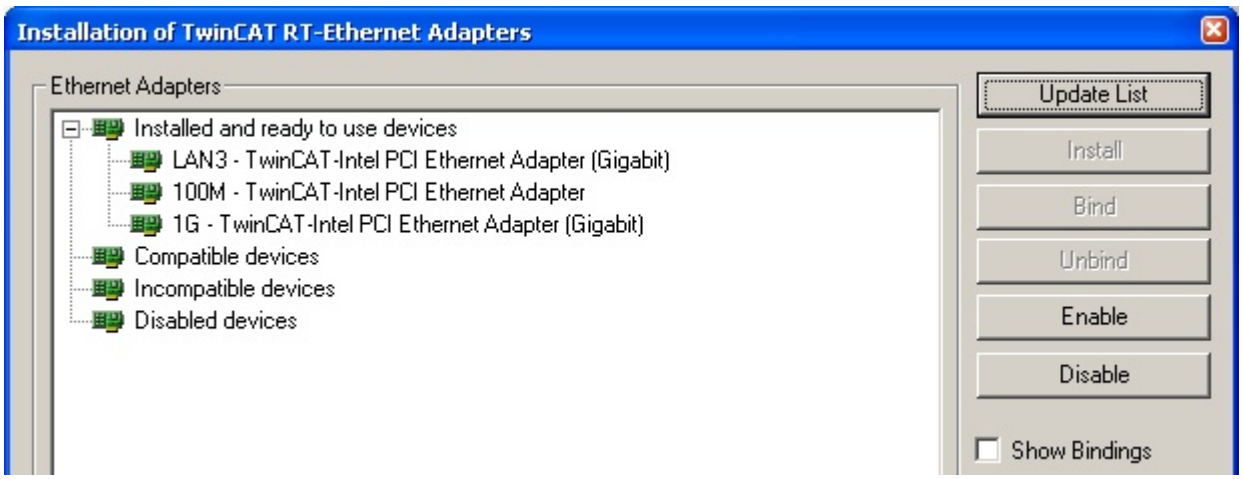


Abb. 157: Übersicht Netzwerkschnittstellen

Hier können nun Schnittstellen, die unter „Kompatible Geräte“ aufgeführt sind, über den „Install“ Button mit dem Treiber belegt werden. Eine Installation des Treibers auf inkompatiblen Devices sollte nicht vorgenommen werden.

Ein Windows-Warnhinweis bezüglich des unsignierten Treibers kann ignoriert werden.

Alternativ kann auch wie im Kapitel Offline Konfigurationserstellung, Abschnitt „Anlegen des Geräts EtherCAT“ [▶ 129] beschrieben, zunächst ein EtherCAT-Gerät eingetragen werden, um dann über dessen Eigenschaften (Karteireiter „Adapter“, Button „Kompatible Geräte…“) die kompatiblen Ethernet Ports einzusehen:

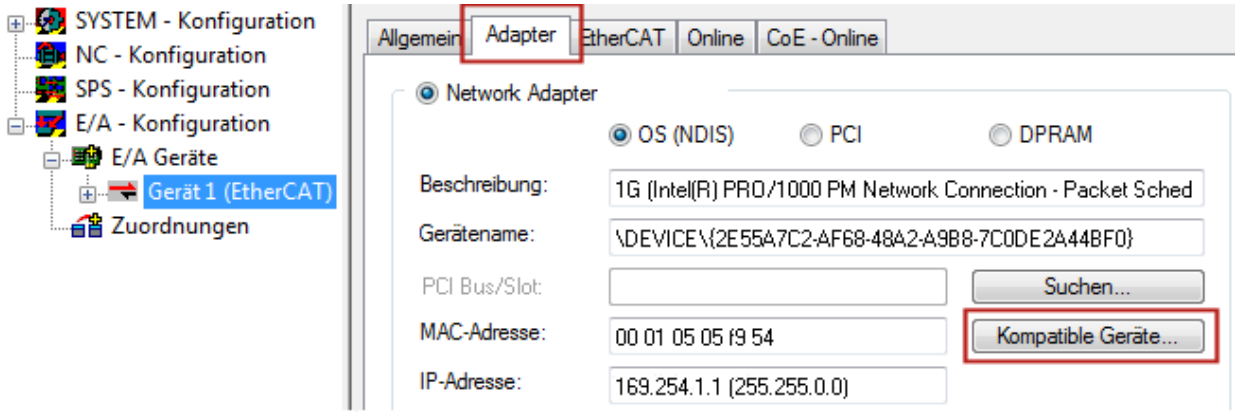
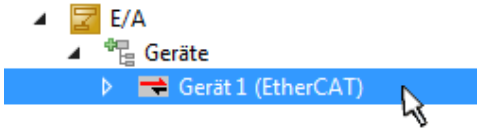


Abb. 158: Eigenschaft von EtherCAT Gerät (TwinCAT 2): Klick auf „Kompatible Geräte…“ von „Adapter“

TwinCAT 3: Die Eigenschaften des EtherCAT-Gerätes können mit Doppelklick auf „Gerät .. (EtherCAT)“ im Projektmappen-Explorer unter „E/A“ geöffnet werden:



Nach der Installation erscheint der Treiber aktiviert in der Windows-Übersicht der einzelnen Netzwerkschnittstelle (Windows Start → Systemsteuerung → Netzwerk)

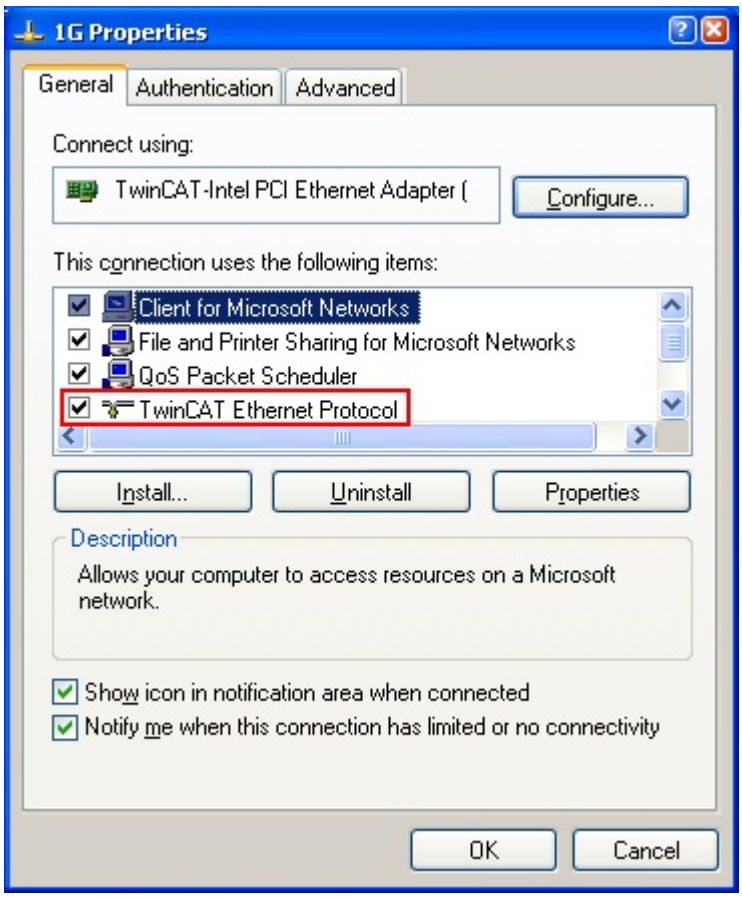


Abb. 159: Windows-Eigenschaften der Netzwerkschnittstelle

Eine korrekte Einstellung des Treibers könnte wie folgt aussehen:

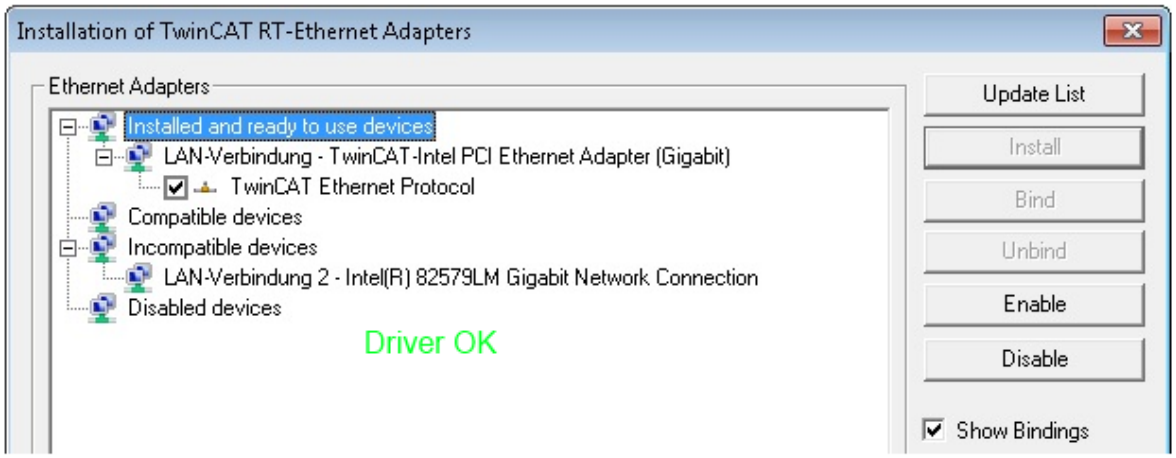
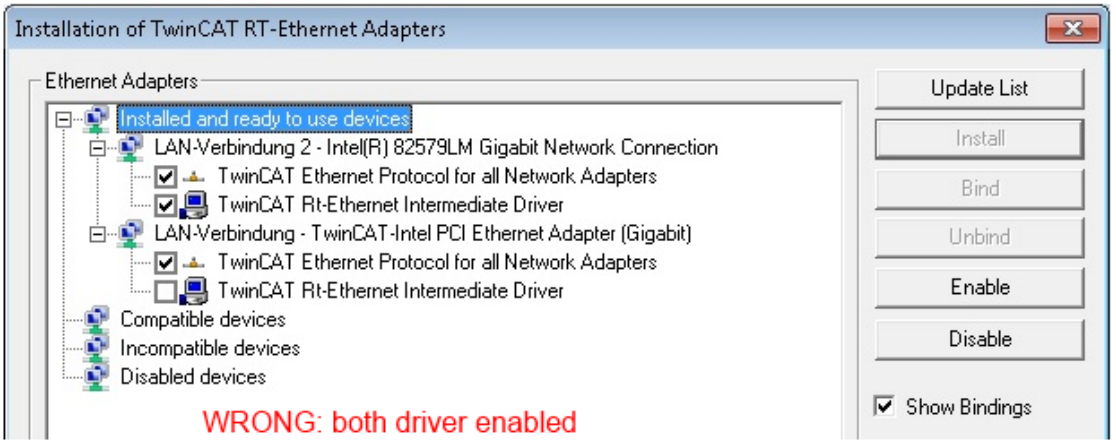
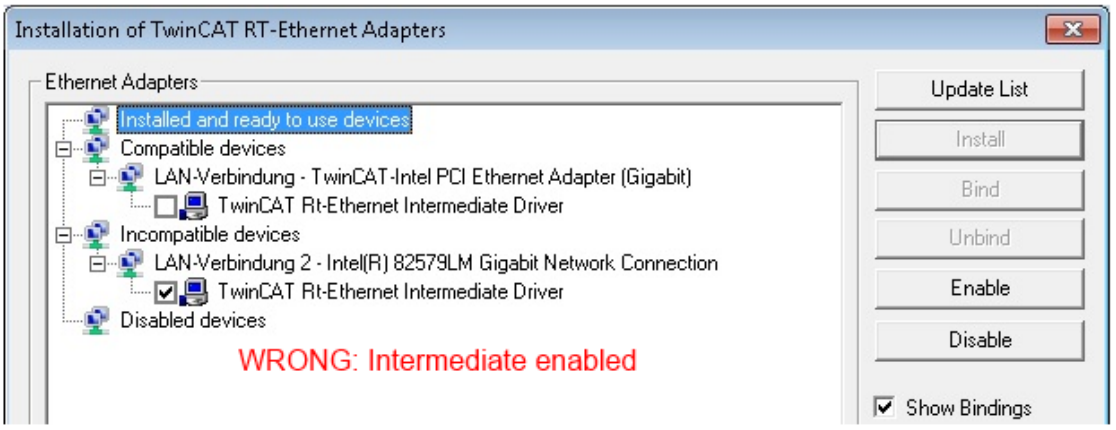
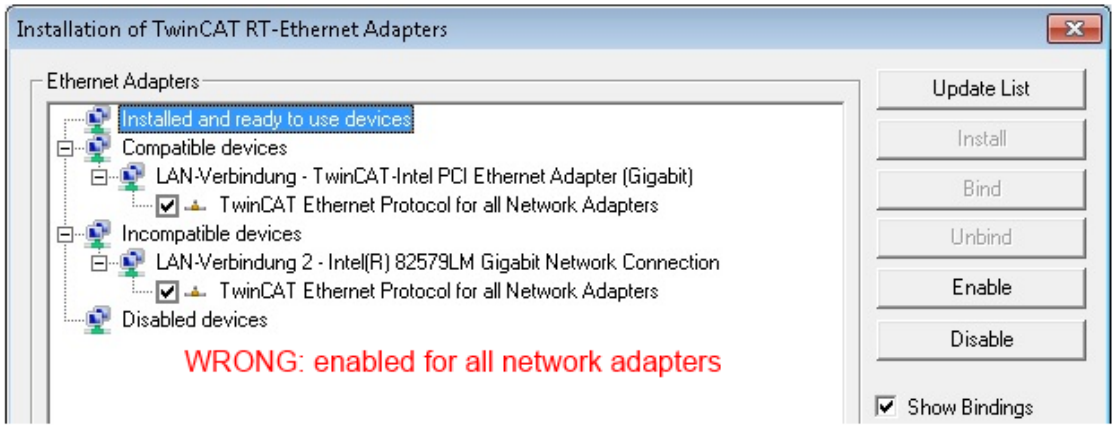


Abb. 160: Beispielhafte korrekte Treiber-Einstellung des Ethernet Ports

Andere mögliche Einstellungen sind zu vermeiden:







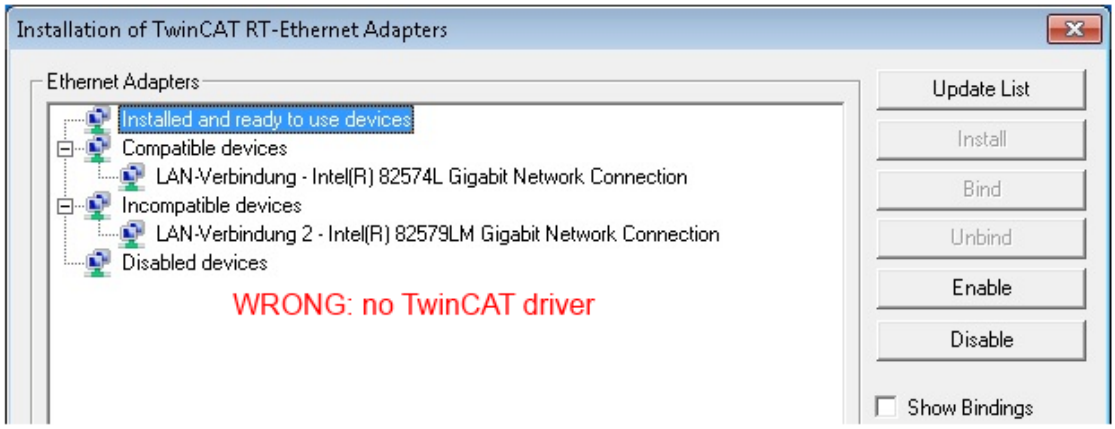


Abb. 161: Fehlerhafte Treiber-Einstellungen des Ethernet Ports

IP-Adresse des verwendeten Ports

IP Adresse/DHCP

In den meisten Fällen wird ein Ethernet-Port, der als EtherCAT-Gerät konfiguriert wird, keine allgemeinen IP-Pakete transportieren. Deshalb und für den Fall, dass eine EL6601 oder entsprechende Geräte eingesetzt werden, ist es sinnvoll, über die Treiber-Einstellung „Internet Protocol TCP/IP“ eine feste IP-Adresse für diesen Port zu vergeben und DHCP zu deaktivieren. Dadurch entfällt die Wartezeit, bis sich der DHCP-Client des Ethernet Ports eine Default-IP-Adresse zuteilt, weil er keine Zuteilung eines DHCP-Servers erhält. Als Adressraum empfiehlt sich z. B. 192.168.x.x.

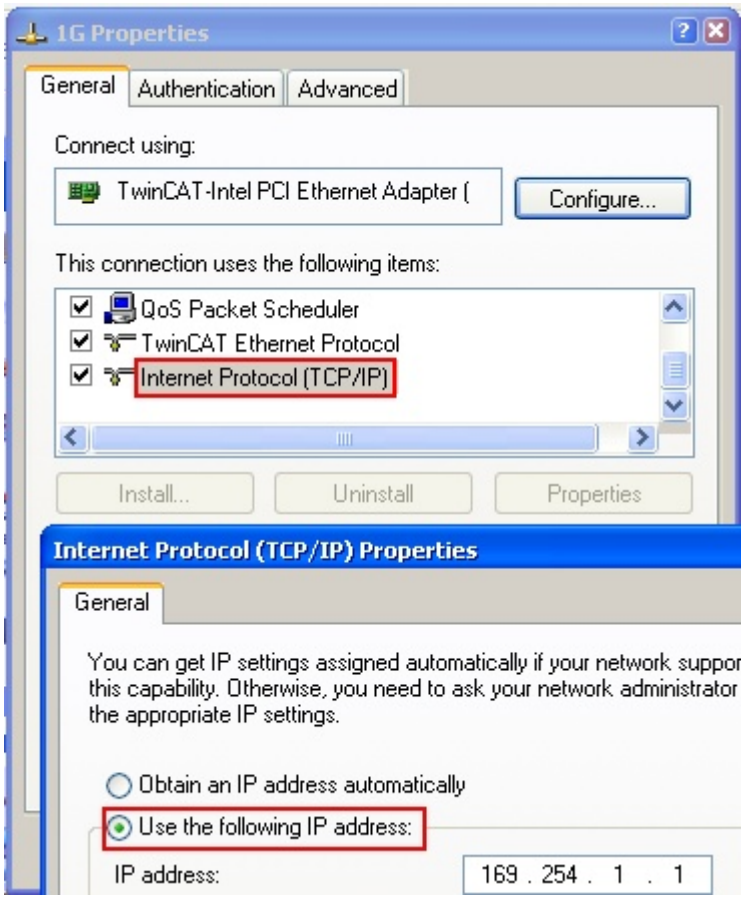


Abb. 162: TCP/IP-Einstellung des Ethernet Ports

3.6.2 Hinweise ESI-Gerätebeschreibung

Installation der neuesten ESI-Device-Description

Der TwinCAT EtherCAT Master/System Manager benötigt zur Konfigurationserstellung im Online- und Offline-Modus die Gerätebeschreibungsdateien der zu verwendeten Geräte. Diese Gerätebeschreibungen sind die so genannten ESI (EtherCAT Slave Information) in Form von XML-Dateien. Diese Dateien können vom jeweiligen Hersteller angefordert werden bzw. werden zum Download bereitgestellt. Eine *.xml-Datei kann dabei mehrere Gerätebeschreibungen enthalten.

Auf der Beckhoff Website werden die ESI für Beckhoff EtherCAT Geräte bereitgehalten.

Die ESI-Dateien sind im Installationsverzeichnis von TwinCAT abzulegen.

Standardeinstellungen:

- **TwinCAT 2**: C:\TwinCAT\IO\EtherCAT
- **TwinCAT 3**: C:\TwinCAT\3.1\Config\Io\EtherCAT

Beim Öffnen eines neuen System Manager-Fensters werden die Dateien einmalig eingelesen, wenn sie sich seit dem letzten System Manager-Fenster geändert haben.

TwinCAT bringt bei der Installation den Satz an Beckhoff-ESI-Dateien mit, der zum Erstellungszeitpunkt des TwinCAT builds aktuell war.

Ab TwinCAT 2.11 / TwinCAT 3 kann aus dem System Manager heraus das ESI-Verzeichnis aktualisiert werden, wenn der Programmier-PC mit dem Internet verbunden ist; unter

TwinCAT 2: Options → „Update EtherCAT Device Descriptions“

TwinCAT 3: TwinCAT → EtherCAT Devices → “Update Device Descriptions (via ETG Website)…”

Hierfür steht der TwinCAT ESI Updater [▶ 128] zur Verfügung.

ESI

Zu den *.xml-Dateien gehören die so genannten *.xsd-Dateien, die den Aufbau der ESI-XML-Dateien beschreiben. Bei einem Update der ESI-Gerätebeschreibungen sind deshalb beide Dateiarten ggf. zu aktualisieren.

Geräteunterscheidung

EtherCAT Geräte/Slaves werden durch vier Eigenschaften unterschieden, aus denen die vollständige Gerätebezeichnung zusammengesetzt wird. Beispielsweise setzt sich die Gerätebezeichnung „EL2521-0025-1018“ zusammen aus:

- Familienschlüssel „EL“
- Name „2521“
- Typ „0025“
- und Revision „1018“

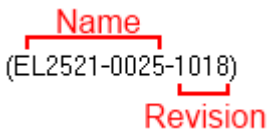


Abb. 163: Gerätebezeichnung: Struktur

Die Bestellbezeichnung aus Typ + Version (hier: EL2521-0010) beschreibt die Funktion des Gerätes. Die Revision gibt den technischen Fortschritt wieder und wird von Beckhoff verwaltet. Prinzipiell kann ein Gerät mit höherer Revision ein Gerät mit niedrigerer Revision ersetzen, wenn z. B. in der Dokumentation nicht anders angegeben. Jeder Revision zugehörig ist eine eigene ESI-Beschreibung. Siehe weitere Hinweise [▶ 101].

Online Description

Wird die EtherCAT Konfiguration online durch Scannen real vorhandener Teilnehmer erstellt (s. Kapitel Online Erstellung) und es liegt zu einem vorgefundenen Slave (ausgezeichnet durch Name und Revision) keine ESI-Beschreibung vor, fragt der System Manager, ob er die im Gerät vorliegende Beschreibung verwenden soll. Der System Manager benötigt in jedem Fall diese Information, um die zyklische und azyklische Kommunikation mit dem Slave richtig einstellen zu können.

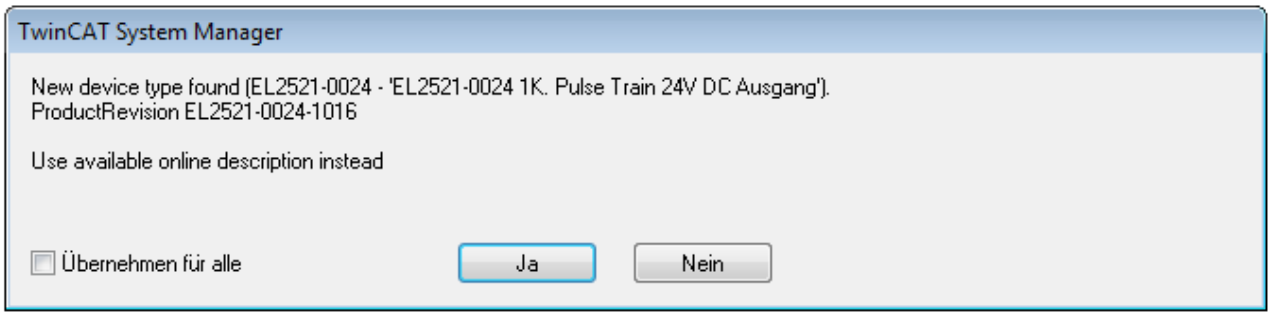


Abb. 164: Hinweisfenster OnlineDescription (TwinCAT 2)

In TwinCAT 3 erscheint ein ähnliches Fenster, das auch das Web-Update anbietet:

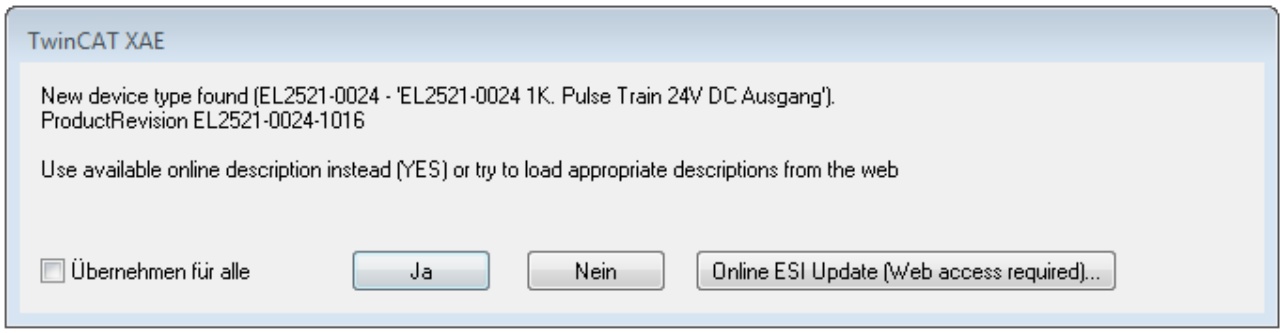


Abb. 165: Hinweisfenster OnlineDescription (TwinCAT 3)

Wenn möglich, ist das *Yes* abzulehnen und vom Geräte-Hersteller die benötigte ESI anzufordern. Nach Installation der XML/XSD-Datei ist der Konfigurationsvorgang erneut vorzunehmen.

HINWEIS

Veränderung der „üblichen“ Konfiguration durch Scan

✓ für den Fall eines durch Scan entdeckten aber TwinCAT noch unbekannten Geräts sind zwei Fälle zu unterscheiden. Hier am Beispiel der EL2521-0000 in der Revision 1019:

a) für das Gerät EL2521-0000 liegt überhaupt keine ESI vor, weder für die Revision 1019 noch für eine ältere Revision. Dann ist vom Hersteller (hier: Beckhoff) die ESI anzufordern.

b) für das Gerät EL2521-0000 liegt eine ESI nur in älterer Revision vor, z. B. 1018 oder 1017. Dann sollte erst betriebsintern überprüft werden, ob die Ersatzteilhaltung überhaupt die Integration der erhöhten Revision in die Konfiguration zulässt. Üblicherweise bringt eine neue/größere Revision auch neue Features mit. Wenn diese nicht genutzt werden sollen, kann ohne Bedenken mit der bisherigen Revision 1018 in der Konfiguration weitergearbeitet werden. Dies drückt auch die Beckhoff Kompatibilitätsregel aus.

Siehe dazu insbesondere das Kapitel „Allgemeine Hinweise zur Verwendung von Beckhoff EtherCAT IO-Komponenten" und zur manuellen Konfigurationserstellung das Kapitel „Offline Konfigurationserstellung [▶ 129]“.

Wird dennoch die Online Description verwendet, liest der System Manager aus dem im EtherCAT Slave befindlichen EEPROM eine Kopie der Gerätebeschreibung aus. Bei komplexen Slaves kann die EEPROM-Größe u. U. nicht ausreichend für die gesamte ESI sein, weshalb im Konfigurator dann eine *unvollständige* ESI vorliegt. Deshalb wird für diesen Fall die Verwendung einer offline ESI-Datei vorrangig empfohlen.

Der System Manager legt bei „online“ erfassten Gerätebeschreibungen in seinem ESI-Verzeichnis eine neue Datei „OnlineDescription0000...xml“ an, die alle online ausgelesenen ESI-Beschreibungen enthält.

OnlineDescriptionCache00000002.xml

Abb. 166: Vom System Manager angelegt OnlineDescription.xml

Soll daraufhin ein Slave manuell in die Konfiguration eingefügt werden, sind „online“ erstellte Slaves durch ein vorangestelltes „>“ Symbol in der Auswahlliste gekennzeichnet (siehe Abbildung *Kennzeichnung einer online erfassten ESI am Beispiel EL2521*).

Abb. 167: Kennzeichnung einer online erfassten ESI am Beispiel EL2521

Wurde mit solchen ESI-Daten gearbeitet und liegen später die herstellereigenen Dateien vor, ist die OnlineDescription....xml wie folgt zu löschen:

- alle System Managerfenster schließen
- TwinCAT in Konfig-Mode neu starten
- „OnlineDescription0000...xml“ löschen
- TwinCAT System Manager wieder öffnen

Danach darf diese Datei nicht mehr zu sehen sein, Ordner ggf. mit <F5> aktualisieren.

OnlineDescription unter TwinCAT 3.x

Zusätzlich zu der oben genannten Datei „OnlineDescription0000...xml“ legt TwinCAT 3.x auch einen so genannten EtherCAT-Cache mit neuentdeckten Geräten an, z. B. unter Windows 7 unter

C:\User\[USERNAME]\AppData\Roaming\Beckhoff\TwinCAT3\Components\Base\EtherCATCache.xml

(Spracheinstellungen des Betriebssystems beachten!)
Diese Datei ist im gleichen Zuge wie die andere Datei zu löschen.

Fehlerhafte ESI-Datei

Liegt eine fehlerhafte ESI-Datei vor die vom System Manager nicht eingelesen werden kann, meldet dies der System Manager durch ein Hinweisfenster.

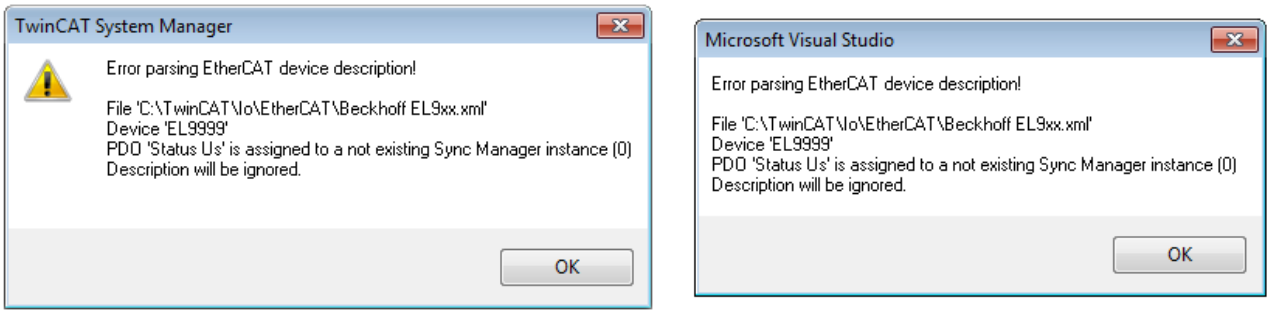


Abb. 168: Hinweisfenster fehlerhafte ESI-Datei (links: TwinCAT 2; rechts: TwinCAT 3)

Ursachen dafür können sein

- Aufbau der *.xml entspricht nicht der zugehörigen *.xsd-Datei → prüfen Sie die Ihnen vorliegenden Schemata
- Inhalt kann nicht in eine Gerätebeschreibung übersetzt werden → Es ist der Hersteller der Datei zu kontaktieren

3.6.3 TwinCAT ESI Updater

Ab TwinCAT 2.11 kann der System Manager bei Online-Zugang selbst nach aktuellen Beckhoff ESI-Dateien suchen:

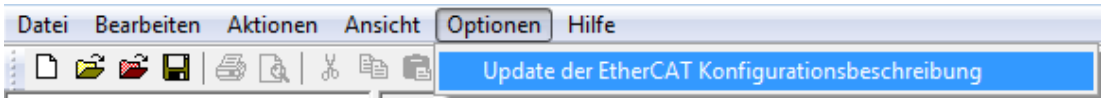


Abb. 169: Anwendung des ESI Updater (>=TwinCAT 2.11)

Der Aufruf erfolgt unter:
„Options“ → „Update EtherCAT Device Descriptions“.

Auswahl bei TwinCAT 3:

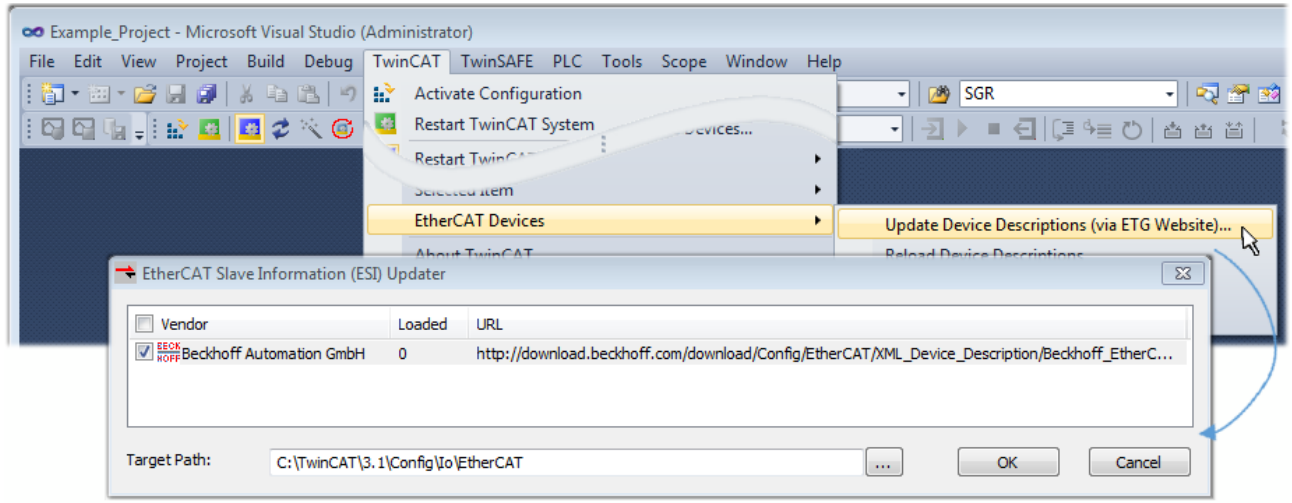


Abb. 170: Anwendung des ESI Updater (TwinCAT 3)

Der ESI Updater ist eine bequeme Möglichkeit, die von den EtherCAT Herstellern bereitgestellten ESIs automatisch über das Internet in das TwinCAT-Verzeichnis zu beziehen (ESI = EtherCAT slave information). Dazu greift TwinCAT auf die bei der ETG hinterlegte zentrale ESI-ULR-Verzeichnisliste zu; die Einträge sind dann unveränderbar im Updater-Dialog zu sehen.

Der Aufruf erfolgt unter:
„TwinCAT“ → „EtherCAT Devices“ → „Update Device Description (via ETG Website)…“.

3.6.4 Unterscheidung Online/Offline

Die Unterscheidung Online/Offline bezieht sich auf das Vorhandensein der tatsächlichen I/O-Umgebung (Antriebe, Klemmen, EJ-Module). Wenn die Konfiguration im Vorfeld der Anlagenerstellung z. B. auf einem Laptop als Programmiersystem erstellt werden soll, ist nur die „Offline-Konfiguration“ möglich. Dann müssen alle Komponenten händisch in der Konfiguration z. B. nach Elektro-Planung eingetragen werden.

Ist die vorgesehene Steuerung bereits an das EtherCAT System angeschlossen, alle Komponenten mit Spannung versorgt und die Infrastruktur betriebsbereit, kann die TwinCAT Konfiguration auch vereinfacht durch das so genannte „Scannen“ vom Runtime-System aus erzeugt werden. Dies ist der so genannte Online-Vorgang.

In jedem Fall prüft der EtherCAT Master bei jedem realen Hochlauf, ob die vorgefundenen Slaves der Konfiguration entsprechen. Dieser Test kann in den erweiterten Slave-Einstellungen parametriert werden. Siehe hierzu den Hinweis „Installation der neuesten ESI-XML-Device-Description“ [▶ 124].

Zur Konfigurationserstellung

- muss die reale EtherCAT-Hardware (Geräte, Koppler, Antriebe) vorliegen und installiert sein.

- müssen die Geräte/Module über EtherCAT-Kabel bzw. im Klemmenstrang so verbunden sein wie sie später eingesetzt werden sollen.
- müssen die Geräte/Module mit Energie versorgt werden und kommunikationsbereit sein.
- muss TwinCAT auf dem Zielsystem im CONFIG-Modus sein.

Der Online-Scan-Vorgang setzt sich zusammen aus:

- Erkennen des EtherCAT-Gerätes [▶ 134] (Ethernet-Port am IPC)
- Erkennen der angeschlossenen EtherCAT-Teilnehmer [▶ 135]. Dieser Schritt kann auch unabhängig vom vorangehenden durchgeführt werden.
- Problembehandlung [▶ 138]

Auch kann der Scan bei bestehender Konfiguration [▶ 139] zum Vergleich durchgeführt werden.

3.6.5 OFFLINE Konfigurationserstellung

Anlegen des Geräts EtherCAT

In einem leeren System Manager Fenster muss zuerst ein EtherCAT Gerät angelegt werden.

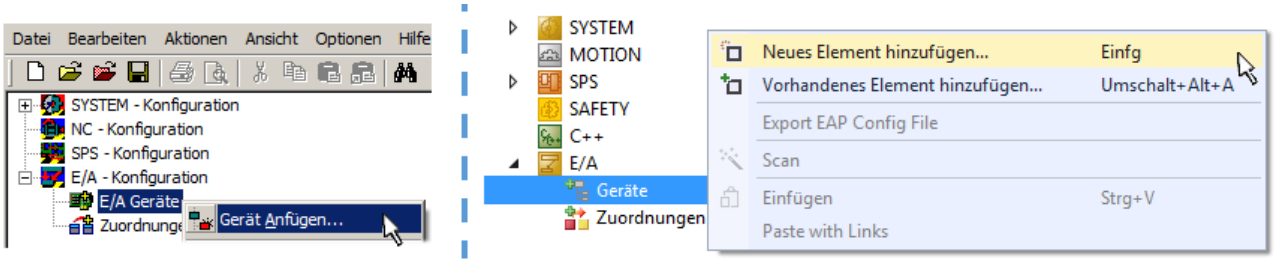


Abb. 171: Anfügen eines EtherCAT Device: links TwinCAT 2; rechts TwinCAT 3

Für eine EtherCAT I/O Anwendung mit EtherCAT Slaves ist der „EtherCAT“ Typ auszuwählen. „EtherCAT Automation Protocol via EL6601“ ist für den bisherigen Publisher/Subscriber-Dienst in Kombination mit einer EL6601/EL6614 Klemme auszuwählen.

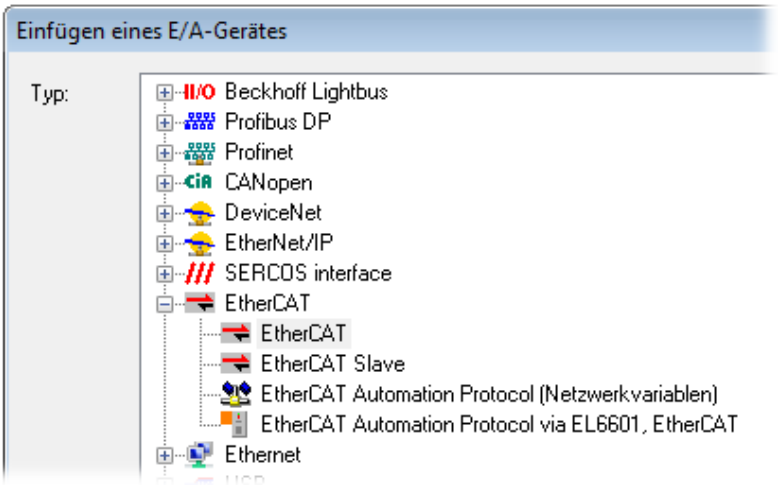


Abb. 172: Auswahl EtherCAT Anschluss (TwinCAT 2.11, TwinCAT 3)

Diesem virtuellen Gerät ist dann ein realer Ethernet Port auf dem Laufzeitsystem zuzuordnen.

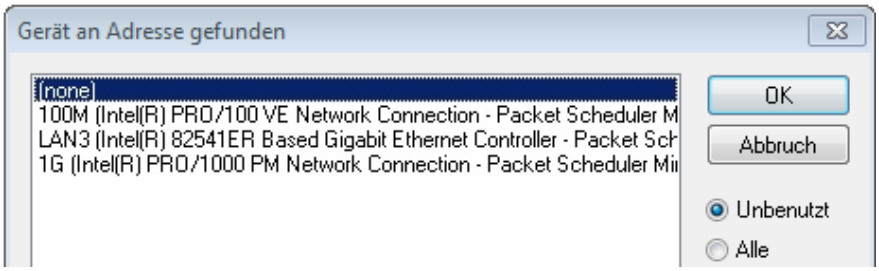


Abb. 173: Auswahl Ethernet Port

Diese Abfrage kann beim Anlegen des EtherCAT-Gerätes automatisch erscheinen, oder die Zuordnung kann später im Eigenschaftendialog gesetzt/geändert werden; siehe Abb. „Eigenschaften EtherCAT Gerät (TwinCAT 2)“.

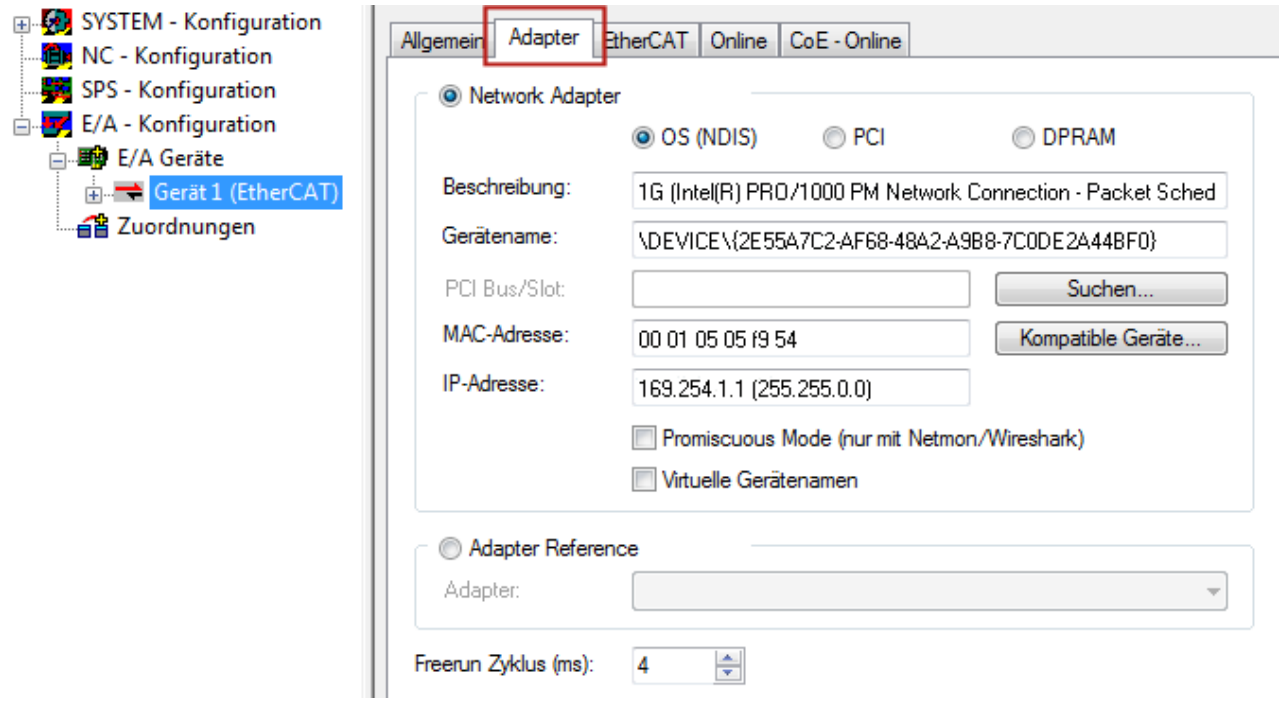
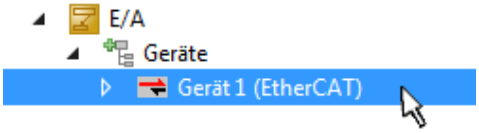


Abb. 174: Eigenschaften EtherCAT Gerät (TwinCAT 2)

TwinCAT 3: Die Eigenschaften des EtherCAT-Gerätes können mit Doppelklick auf „Gerät .. (EtherCAT)“ im Projektmappen-Explorer unter „E/A“ geöffnet werden:



Auswahl Ethernet Port

Es können nur Ethernet Ports für ein EtherCAT Gerät ausgewählt werden, für die der TwinCAT Realtime-Treiber installiert ist. Dies muss für jeden Port getrennt vorgenommen werden. Siehe dazu die entsprechende Installationsseite [▶ 118].

Definieren von EtherCAT Slaves

Durch Rechtsklick auf ein Gerät im Konfigurationsbaum können weitere Geräte angefügt werden.

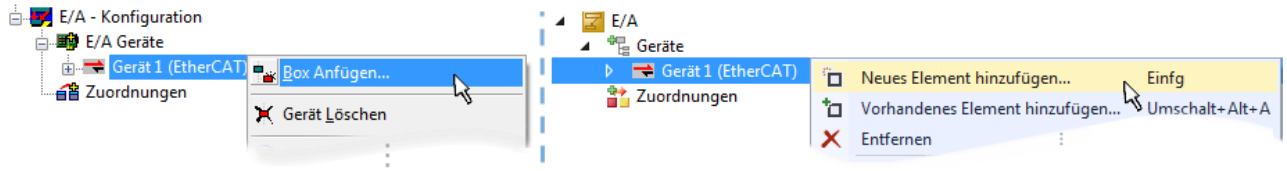


Abb. 175: Anfügen von EtherCAT Geräten (links: TwinCAT 2; rechts: TwinCAT 3)

Es öffnet sich der Dialog zur Auswahl des neuen Gerätes. Es werden nur Geräte angezeigt für die ESI-Dateien hinterlegt sind.

Die Auswahl bietet auch nur Geräte an, die an dem vorher angeklickten Gerät anzufügen sind - dazu wird die an diesem Port mögliche Übertragungsphysik angezeigt (Abb. „Auswahldialog neues EtherCAT Gerät“, A). Es kann sich um kabelgebundene FastEthernet-Ethernet-Physik mit PHY-Übertragung handeln, dann ist wie in Abb. „Auswahldialog neues EtherCAT Gerät“ nur ebenfalls kabelgebundenes Geräte auswählbar. Verfügt das vorangehende Gerät über mehrere freie Ports (z. B. EK1122 oder EK1100), kann auf der rechten Seite (A) der gewünschte Port angewählt werden.

Übersicht Übertragungsphysik

- „Ethernet“: Kabelgebunden 100BASE-TX: EK-Koppler, EP-Boxen, Geräte mit RJ45/M8/M12-Konnector
- „E-Bus“: LVDS „Klemmenbus“„EJ-Module“: EL/ES-Klemmen, diverse anreihbare Module

Das Suchfeld erleichtert das Auffinden eines bestimmten Gerätes (ab TwinCAT 2.11 bzw. TwinCAT 3).

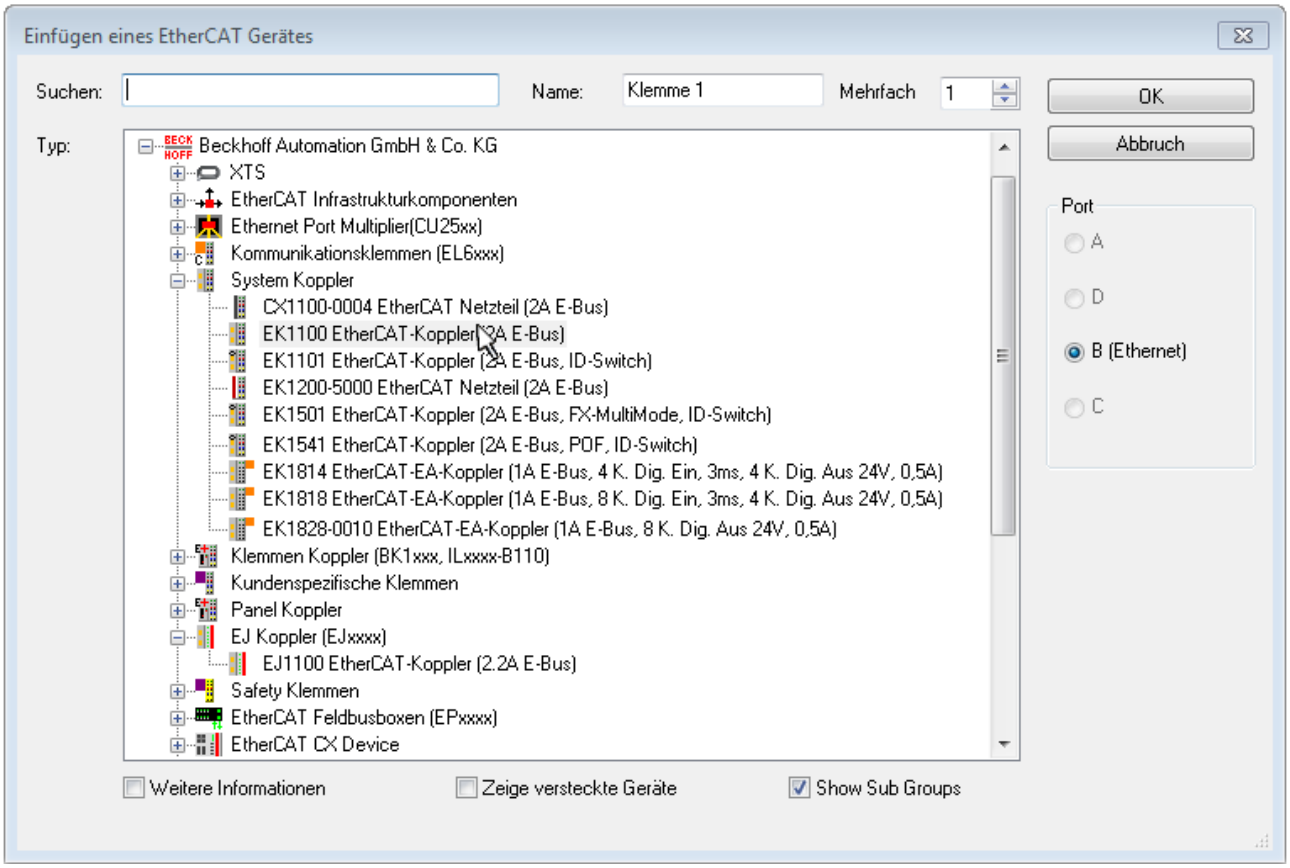


Abb. 176: Auswahldialog neues EtherCAT Gerät

Standardmäßig wird nur der Name/Typ des Gerätes als Auswahlkriterium verwendet. Für eine gezielte Auswahl einer bestimmen Revision des Gerätes kann die Revision als „Extended Information“ eingeblendet werden.

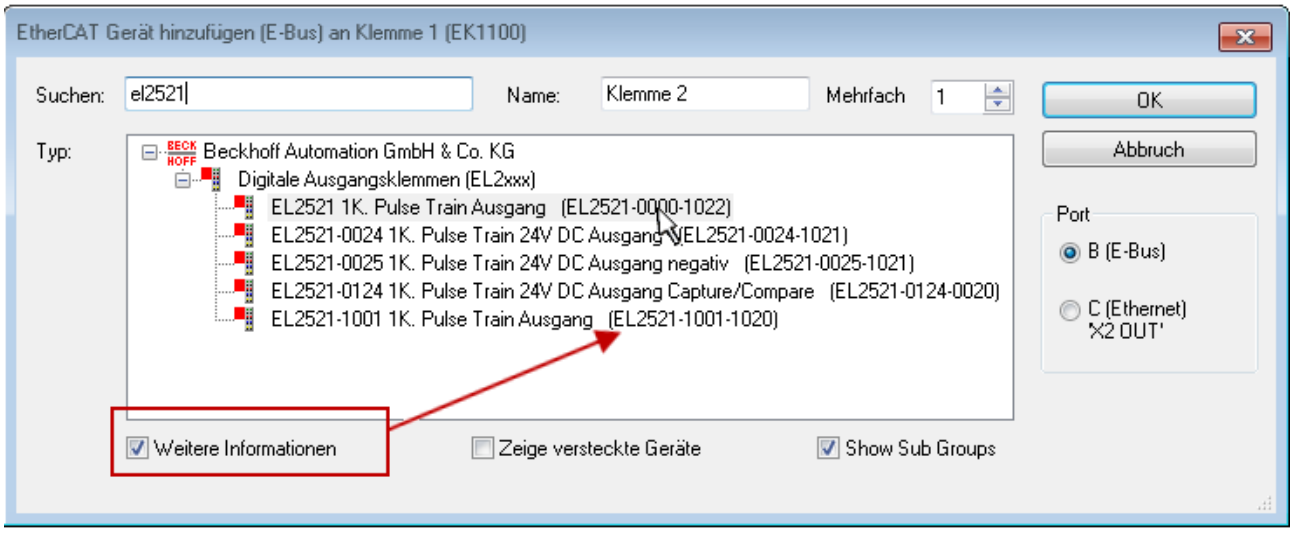


Abb. 177: Anzeige Geräte-Revision

Oft sind aus historischen oder funktionalen Gründen mehrere Revisionen eines Gerätes erzeugt worden, z. B. durch technologische Weiterentwicklung. Zur vereinfachten Anzeige (s. Abb. „Auswahldialog neues EtherCAT Gerät“) wird bei Beckhoff Geräten nur die letzte (=höchste) Revision und damit der letzte Produktionsstand im Auswahldialog angezeigt. Sollen alle im System als ESI-Beschreibungen vorliegenden Revisionen eines Gerätes angezeigt werden, ist die Checkbox „Show Hidden Devices“ zu markieren, s. Abb. „Anzeige vorhergehender Revisionen“.

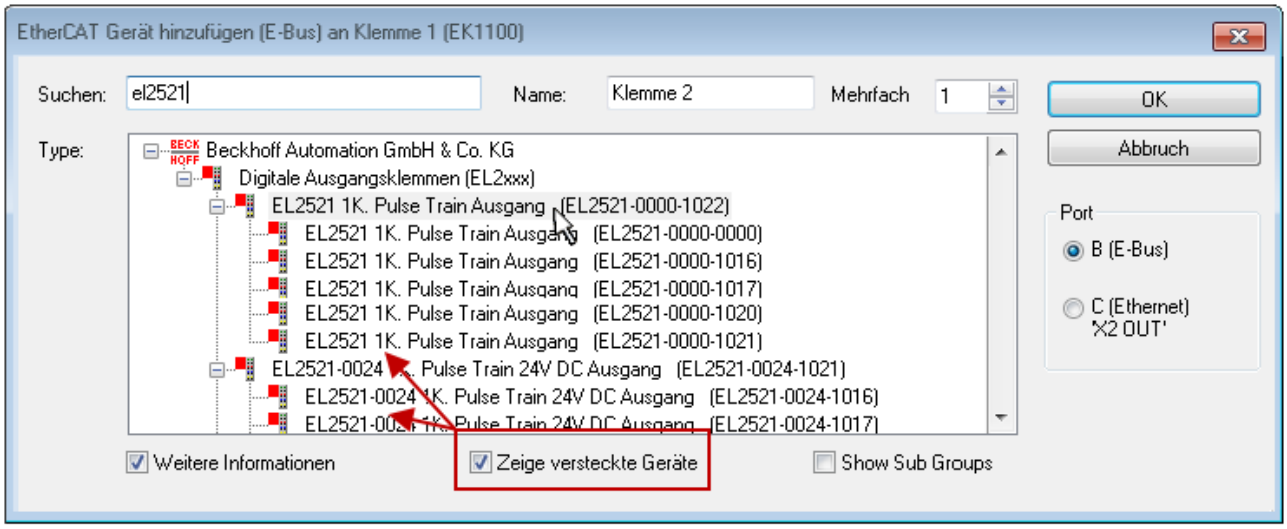


Abb. 178: Anzeige vorhergehender Revisionen

Geräte-Auswahl nach Revision, Kompatibilität

Mit der ESI-Beschreibung wird auch das Prozessabbild, die Art der Kommunikation zwischen Master und Slave/Gerät und ggf. Geräte-Funktionen definiert. Damit muss das reale Gerät (Firmware wenn vorhanden) die Kommunikationsanfragen/-einstellungen des Masters unterstützen. Dies ist abwärtskompatibel der Fall, d. h. neuere Geräte (höhere Revision) sollen es auch unterstützen, wenn der EtherCAT Master sie als eine ältere Revision anspricht. Als Beckhoff-Kompatibilitätsregel für EtherCAT-Klemmen/ Boxen/ EJ-Module ist anzunehmen:

Geräte-Revision in der Anlage >= Geräte-Revision in der Konfiguration

Dies erlaubt auch den späteren Austausch von Geräten ohne Veränderung der Konfiguration (abweichende Vorgaben bei Antrieben möglich).

Beispiel

In der Konfiguration wird eine EL2521-0025-**1018** vorgesehen, dann kann real eine EL2521-0025-**1018** oder höher (-**1019**, -**1020**) eingesetzt werden.

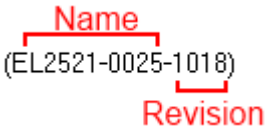


Abb. 179: Name/Revision Klemme

Wenn im TwinCAT System aktuelle ESI-Beschreibungen vorliegen, entspricht der im Auswahldialog als letzte Revision angebotene Stand dem Produktionsstand von Beckhoff. Es wird empfohlen, bei Erstellung einer neuen Konfiguration jeweils diesen letzten Revisionsstand eines Gerätes zu verwenden, wenn aktuell produzierte Beckhoff-Geräte in der realen Applikation verwendet werden. Nur wenn ältere Geräte aus Lagerbeständen in der Applikation verbaut werden sollen, ist es sinnvoll eine ältere Revision einzubinden.

Das Gerät stellt sich dann mit seinem Prozessabbild im Konfigurationsbaum dar und kann nur parametriert werden: Verlinkung mit der Task, CoE/DC-Einstellungen, PlugIn-Definition, StartUp-Einstellungen, ...

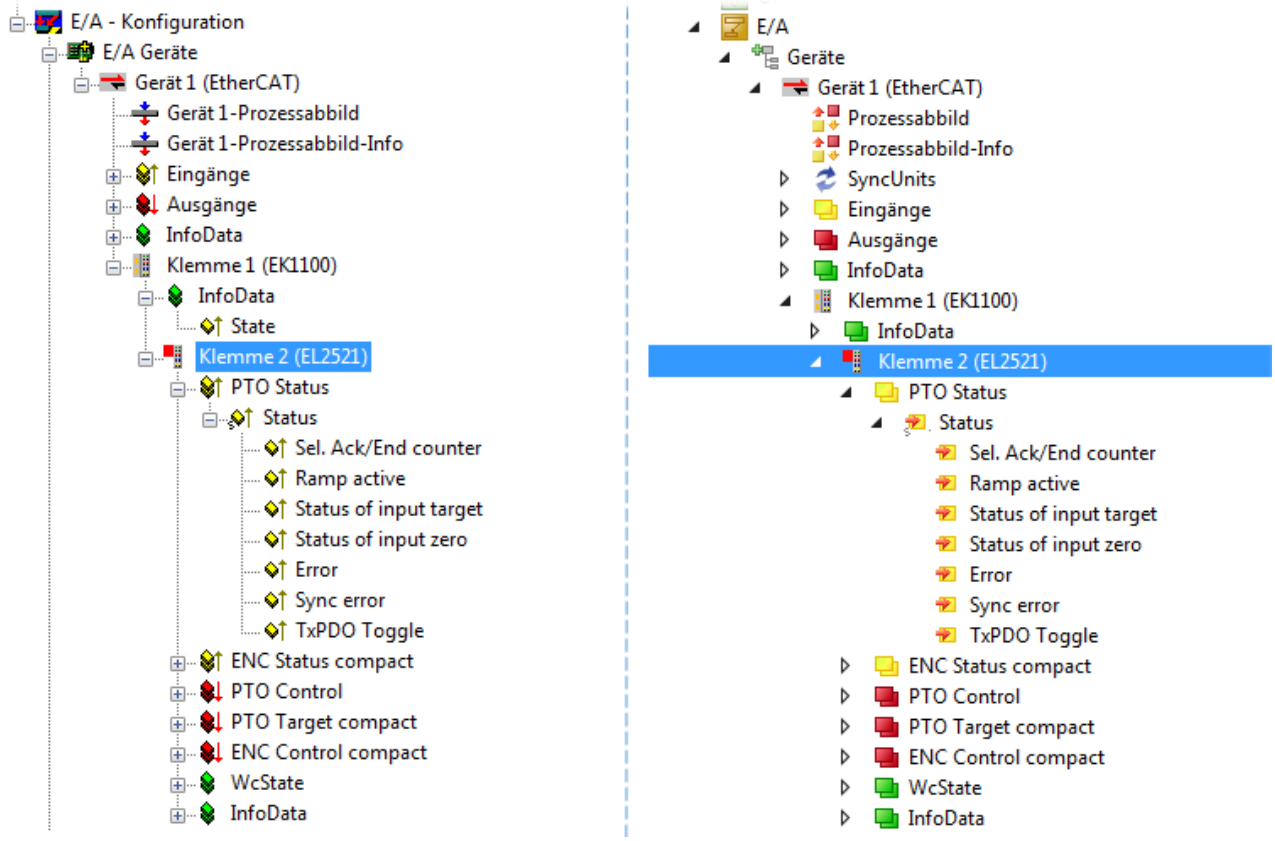


Abb. 180: EtherCAT Klemme im TwinCAT-Baum (links: TwinCAT 2; rechts: TwinCAT 3)

3.6.6 ONLINE Konfigurationserstellung

Erkennen/Scan des Geräts EtherCAT

Befindet sich das TwinCAT-System im CONFIG-Modus, kann online nach Geräten gesucht werden. Erkennbar ist dies durch ein Symbol unten rechts in der Informationsleiste:

- bei TwinCAT 2 durch eine blaue Anzeige „Config Mode“ im System Manager-Fenster: Config Mode .
- bei der Benutzeroberfläche der TwinCAT 3 Entwicklungsumgebung durch ein Symbol .

TwinCAT lässt sich in diesem Modus versetzen:

- TwinCAT 2: durch Auswahl von aus der Menüleiste oder über „Aktionen“ → „Starten/Restarten von TwinCAT in Konfig-Modus“
- TwinCAT 3: durch Auswahl von aus der Menüleiste oder über „TWINCAT“ → „Restart TwinCAT (Config Mode)“

Online Scannen im Config Mode

Die Online-Suche im RUN-Modus (produktiver Betrieb) ist nicht möglich. Es ist die Unterscheidung zwischen TwinCAT-Programmiersystem und TwinCAT-Zielsystem zu beachten.

Das TwinCAT 2-Icon ( ) bzw. TwinCAT 3-Icon ( ) in der Windows Taskleiste stellt immer den TwinCAT-Modus des lokalen IPC dar. Im System Manager-Fenster von TwinCAT 2 bzw. in der Benutzeroberfläche von TwinCAT 3 wird dagegen der TwinCAT-Zustand des Zielsystems angezeigt.

Abb. 181: Unterscheidung Lokalsystem/ Zielsystem (links: TwinCAT 2; rechts: TwinCAT 3)

Im Konfigurationsbaum bringt uns ein Rechtsklick auf den General-Punkt „I/O Devices“ zum Such-Dialog.

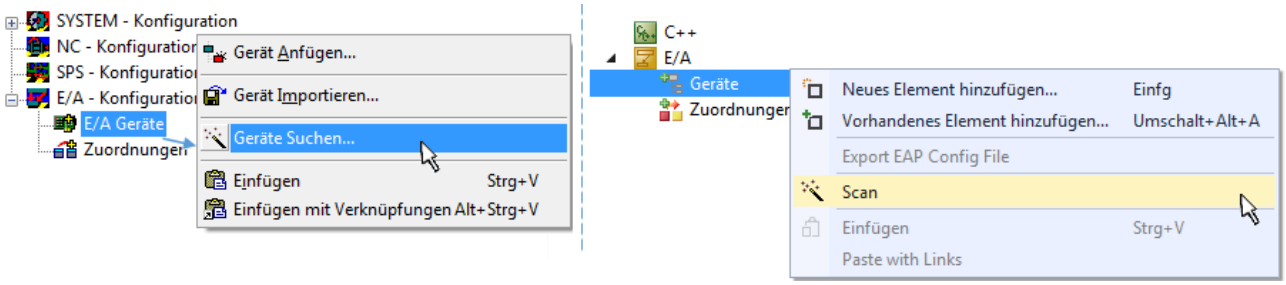


Abb. 182: Scan Devices (links: TwinCAT 2; rechts: TwinCAT 3)

Dieser Scan-Modus versucht nicht nur EtherCAT-Geräte (bzw. die als solche nutzbaren Ethernet-Ports) zu finden, sondern auch NOVRAM, Feldbuskarten, SMB etc. Nicht alle Geräte können jedoch automatisch gefunden werden.

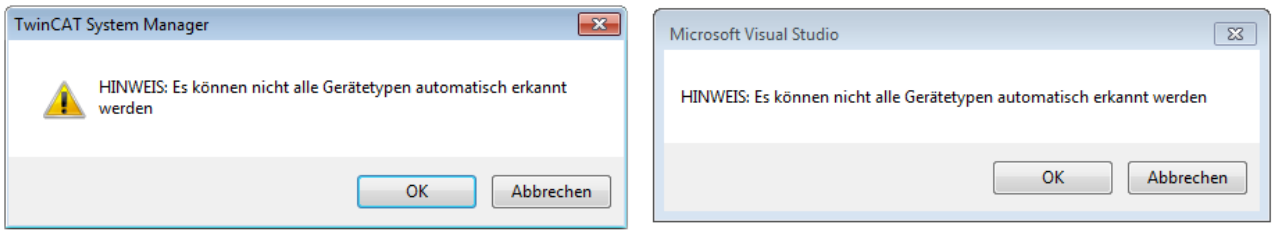


Abb. 183: Hinweis automatischer GeräteScan (links: TwinCAT 2; rechts: TwinCAT 3)

Ethernet Ports mit installierten TwinCAT Realtime-Treiber werden als „RT-Ethernet“ Geräte angezeigt. Testweise wird an diesen Ports ein EtherCAT-Frame verschickt. Erkennt der Scan-Agent an der Antwort, dass ein EtherCAT-Slave angeschlossen ist, wird der Port allerdings gleich als „EtherCAT Device“ angezeigt.

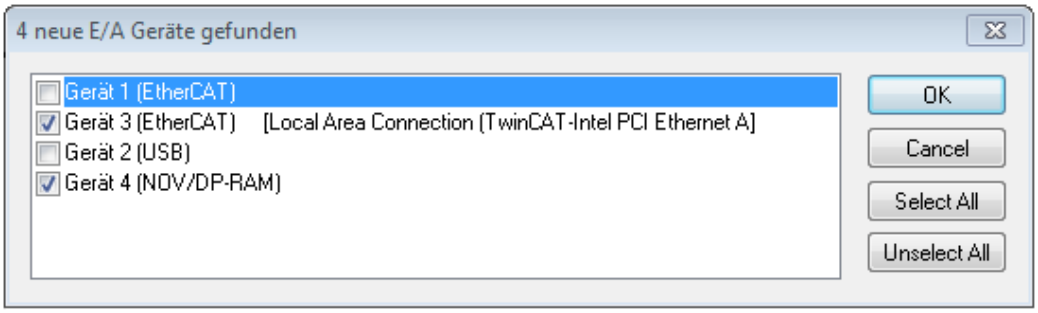


Abb. 184: Erkannte Ethernet-Geräte

Über entsprechende Kontrollkästchen können Geräte ausgewählt werden (wie in der Abb. „Erkannte Ethernet-Geräte“ gezeigt ist z. B. Gerät 3 und Gerät 4 ausgewählt). Für alle angewählten Geräte wird nach Bestätigung „OK“ im nachfolgenden ein Teilnehmer-Scan vorgeschlagen, s. Abb. „Scan-Abfrage nach dem automatischen Anlegen eines EtherCAT Gerätes“.

Auswahl Ethernet Port

Es können nur Ethernet Ports für ein EtherCAT Gerät ausgewählt werden, für die der TwinCAT Realtime-Treiber installiert ist. Dies muss für jeden Port getrennt vorgenommen werden. Siehe dazu die entsprechende Installationsseite [▶ 118].

Erkennen/Scan der EtherCAT Teilnehmer

Funktionsweise Online Scan

Beim Scan fragt der Master die Identity Informationen der EtherCAT Slaves aus dem Slave-EEPROM ab. Es werden Name und Revision zur Typbestimmung herangezogen. Die entsprechenden Geräte werden dann in den hinterlegten ESI-Daten gesucht und in dem dort definierten Default-Zustand in den Konfigurationsbaum eingebaut.

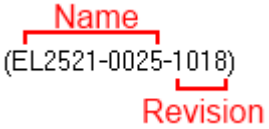


Abb. 185: Beispiel Default-Zustand

HINWEIS

Slave-Scan in der Praxis im Serienmaschinenbau

Die Scan-Funktion sollte mit Bedacht angewendet werden. Sie ist ein praktisches und schnelles Werkzeug, um für eine Inbetriebnahme eine Erst-Konfiguration als Arbeitsgrundlage zu erzeugen. Im Serienmaschinebau bzw. bei Reproduktion der Anlage sollte die Funktion aber nicht mehr zur Konfigurationserstellung verwendet werden sondern ggf. zum Vergleich [▶ 139] mit der festgelegten Erst-Konfiguration.
Hintergrund: da Beckhoff aus Gründen der Produktpflege gelegentlich den Revisionsstand der ausgelieferten Produkte erhöht, kann durch einen solchen Scan eine Konfiguration erzeugt werden, die (bei identischem Maschinenaufbau) zwar von der Geräteliste her identisch ist, die jeweilige Gerätерevision unterscheiden sich aber ggf. von der Erstkonfiguration.

Beispiel:

Firma A baut den Prototyp einer späteren Serienmaschine B. Dazu wird der Prototyp aufgebaut, in TwinCAT ein Scan über die IO-Geräte durchgeführt und somit die Erstkonfiguration "B.tsm" erstellt. An einer beliebigen Stelle sitzt dabei die EtherCAT-Klemme EL2521-0025 in der Revision 1018. Diese wird also so in die TwinCAT-Konfiguration eingebaut:

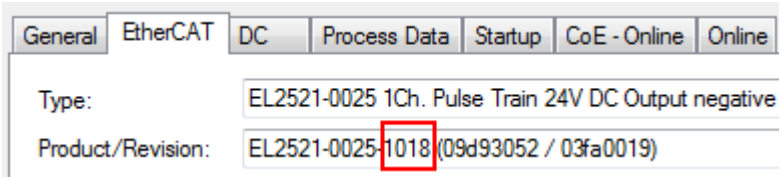


Abb. 186: Einbau EtherCAT-Klemme mit Revision -1018

Ebenso werden in der Prototypentestphase Funktionen und Eigenschaften dieser Klemme durch die Programmierer/Inbetriebnehmer getestet und ggf. genutzt d. h. aus der PLC „B.pro“ oder der NC angesprochen. (sinngemäß gilt das gleiche für die TwinCAT 3-Solution-Dateien).

Nun wird die Prototypenentwicklung abgeschlossen und der Serienbau der Maschine B gestartet, Beckhoff liefert dazu weiterhin die EL2521-0025-0018. Falls die Inbetriebnehmer der Abteilung Serienmaschinenbau immer einen Scan durchführen, entsteht dabei bei jeder Maschine wieder ein eine inhaltsgleiche B-Konfiguration. Ebenso werden eventuell von A weltweit Ersatzteillager für die kommenden Serienmaschinen mit Klemmen EL2521-0025-1018 angelegt.

Nach einiger Zeit erweitert Beckhoff die EL2521-0025 um ein neues Feature C. Deshalb wird die FW geändert, nach außen hin kenntlich durch einen höheren FW-Stand **und eine neue Revision** -1**019**. Trotzdem unterstützt das neue Gerät natürlich Funktionen und Schnittstellen der Vorgängerversion(en), eine Anpassung von „B.tsm“ oder gar „B.pro“ ist somit nicht nötig. Die Serienmaschinen können weiterhin mit „B.tsm“ und „B.pro“ gebaut werden, zur Kontrolle der aufgebauten Maschine ist ein vergleichernder Scan [▶ 139] gegen die Erstkonfiguration „B.tsm“ sinnvoll.

Wird nun allerdings in der Abteilung Seriennmaschinenbau nicht „B.tsm“ verwendet, sondern wieder ein Scan zur Erstellung der produktiven Konfiguration durchgeführt, wird automatisch die Revision **-1019** erkannt und in die Konfiguration eingebaut:

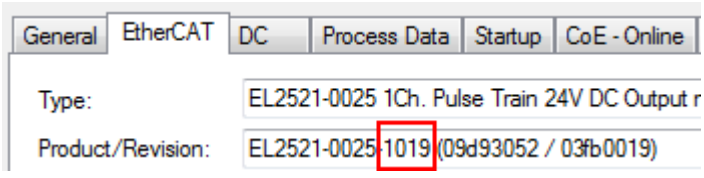


Abb. 187: Erkennen EtherCAT-Klemme mit Revision -1019

Dies wird in der Regel von den Inbetriebnehmern nicht bemerkt. TwinCAT kann ebenfalls nichts melden, da ja quasi eine neue Konfiguration erstellt wird. Es führt nach der Kompatibilitätsregel allerdings dazu, dass in diese Maschine später keine EL2521-0025-**1018** als Ersatzteil eingebaut werden sollen (auch wenn dies in den allermeisten Fällen dennoch funktioniert).

Dazu kommt, dass durch produktionsbegleitende Entwicklung in Firma A das neue Feature C der EL2521-0025-1019 (zum Beispiel ein verbesserter Analogfilter oder ein zusätzliches Prozessdatum zur Diagnose) gerne entdeckt und ohne betriebsinterne Rücksprache genutzt wird. Für die so entstandene neue Konfiguration „B2.tsm“ ist der bisherige Bestand an Ersatzteilgeräten nicht mehr zu verwenden.

Bei etabliertem Serienmaschinenbau sollte der Scan nur noch zu informativen Vergleichszwecken gegen eine definierte Erstkonfiguration durchgeführt werden. Änderungen sind mit Bedacht durchzuführen!

Wurde ein EtherCAT-Device in der Konfiguration angelegt (manuell oder durch Scan), kann das I/O-Feld nach Teilnehmern/Slaves gescannt werden.

Abb. 188: Scan-Abfrage nach dem automatischen Anlegen eines EtherCAT Gerätes (links: TwinCAT 2; rechts TwinCAT 3)

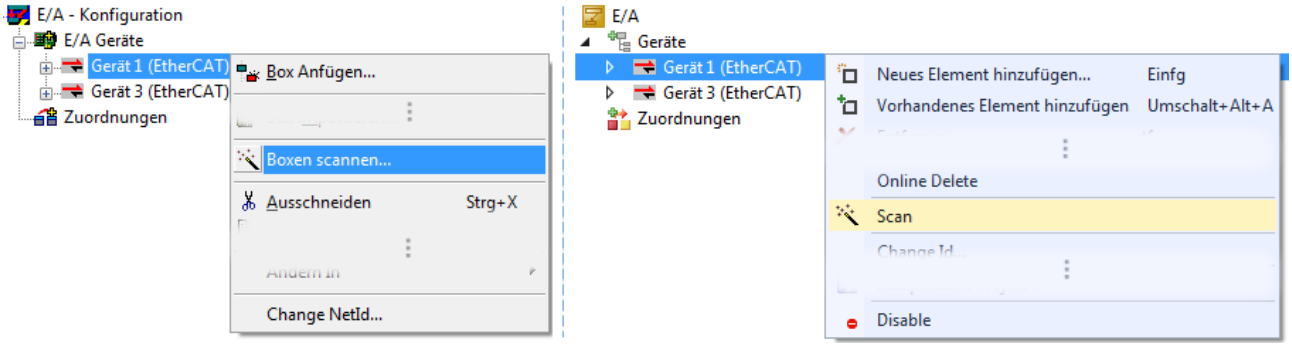


Abb. 189: Manuelles Auslösen des Teilnehmer-Scans auf festegelegtem EtherCAT Device (links: TwinCAT 2; rechts TwinCAT 3)

Im System Manager (TwinCAT 2) bzw. der Benutzeroberfläche (TwinCAT 3) kann der Scan-Ablauf am Ladebalken unten in der Statusleiste verfolgt werden.

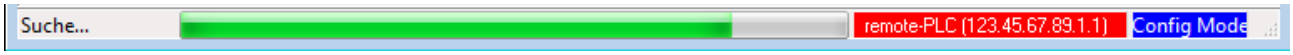


Abb. 190: Scanfortschritt am Beispiel von TwinCAT 2

Die Konfiguration wird aufgebaut und kann danach gleich in den Online-Zustand (OPERATIONAL) versetzt werden.

Abb. 191: Abfrage Config/FreeRun (links: TwinCAT 2; rechts TwinCAT 3)

Im Config/FreeRun-Mode wechselt die System Manager Anzeige blau/rot und das EtherCAT Gerät wird auch ohne aktive Task (NC, PLC) mit der Freilauf-Zykluszeit von 4 ms (Standardeinstellung) betrieben.

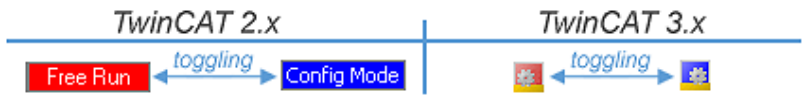


Abb. 192: Anzeige des Wechsels zwischen „Free Run“ und „Config Mode“ unten rechts in der Statusleiste

Abb. 193: TwinCAT kann auch durch einen Button in diesen Zustand versetzt werden (links: TwinCAT 2; rechts TwinCAT 3)

Das EtherCAT System sollte sich danach in einem funktionsfähigen zyklischen Betrieb nach Abb. *Beispielhafte Online-Anzeige* befinden.

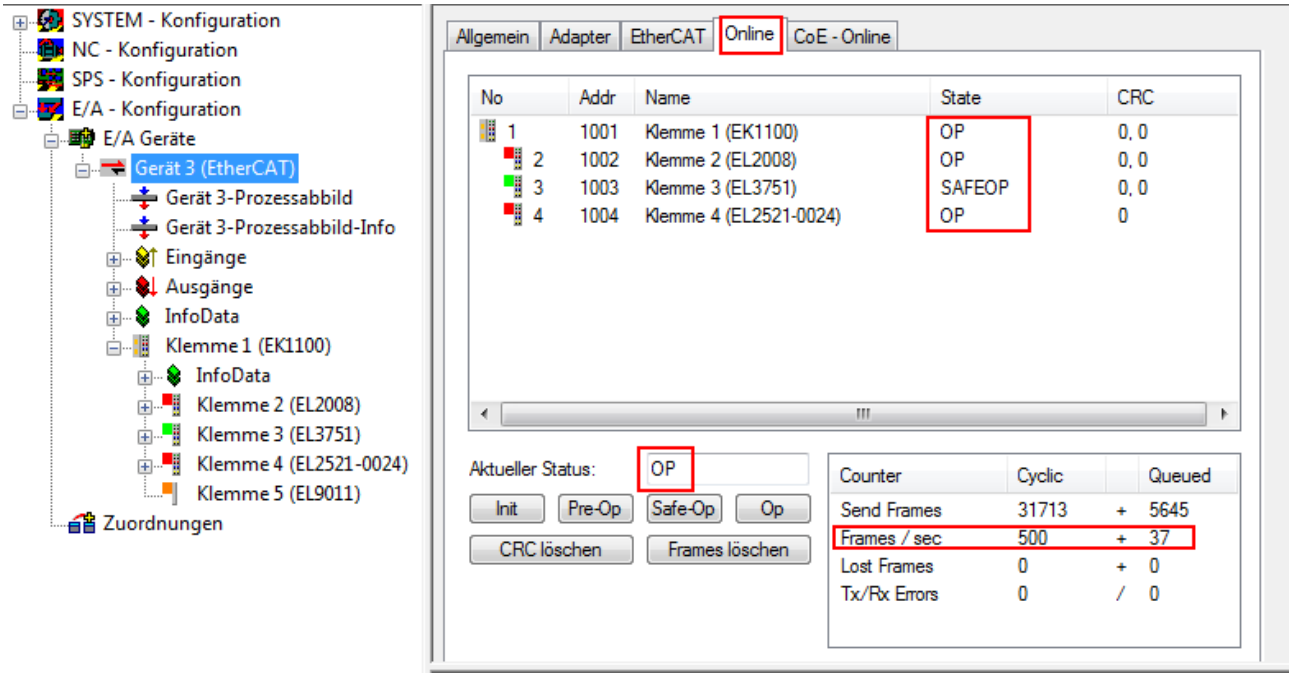


Abb. 194: Beispielhafte Online-Anzeige

Zu beachten sind

- alle Slaves sollen im OP-State sein
- der EtherCAT Master soll im „Actual State“ OP sein
- „Frames/sec“ soll der Zykluszeit unter Berücksichtigung der versendeten Frameanzahl sein
- es sollen weder übermäßig „LostFrames“- noch CRC-Fehler auftreten

Die Konfiguration ist nun fertig gestellt. Sie kann auch wie im manuellen Vorgang [▶ 129] beschrieben verändert werden.

Problembehandlung

Beim Scannen können verschiedene Effekte auftreten.

- es wird ein **unbekanntes Gerät** entdeckt, d. h. ein EtherCAT Slave für den keine ESI-XML-Beschreibung vorliegt.
  In diesem Fall bietet der System Manager an, die im Gerät eventuell vorliegende ESI auszulesen.
  Lesen Sie dazu das Kapitel „Hinweise zu ESI/XML“.
- **Teilnehmer werden nicht richtig erkannt**
  Ursachen können sein
  - fehlerhafte Datenverbindungen, es treten Datenverluste während des Scans auf
  - Slave hat ungültige Gerätebeschreibung

Es sind die Verbindungen und Teilnehmer gezielt zu überprüfen, z. B. durch den Emergency Scan.
Der Scan ist dann erneut vorzunehmen.

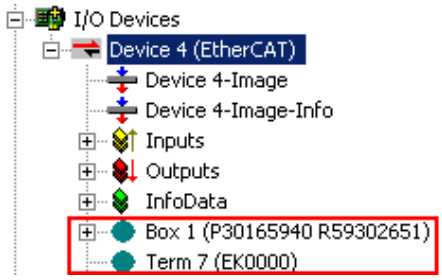


Abb. 195: Fehlerhafte Erkennung

Im System Manager werden solche Geräte evtl. als EK0000 oder unbekannte Geräte angelegt. Ein Betrieb ist nicht möglich bzw. sinnvoll.

Scan über bestehender Konfiguration

HINWEIS

Veränderung der Konfiguration nach Vergleich

Bei diesem Scan werden z. Z. (TwinCAT 2.11 bzw. 3.1) nur die Geräteeigenschaften Vendor (Hersteller), Gerätename und Revision verglichen! Ein „ChangeTo“ oder „Copy“ sollte nur im Hinblick auf die Beckhoff IO-Kompatibilitätsregel (s. o.) nur mit Bedacht vorgenommen werden. Das Gerät wird dann in der Konfiguration gegen die vorgefundene Revision ausgetauscht, dies kann Einfluss auf unterstützte Prozessdaten und Funktionen haben.

Wird der Scan bei bestehender Konfiguration angestoßen, kann die reale I/O-Umgebung genau der Konfiguration entsprechen oder differieren. So kann die Konfiguration verglichen werden.

Abb. 196: Identische Konfiguration (links: TwinCAT 2; rechts TwinCAT 3)

Sind Unterschiede feststellbar, werden diese im Korrekturdialog angezeigt, die Konfiguration kann umgehend angepasst werden.

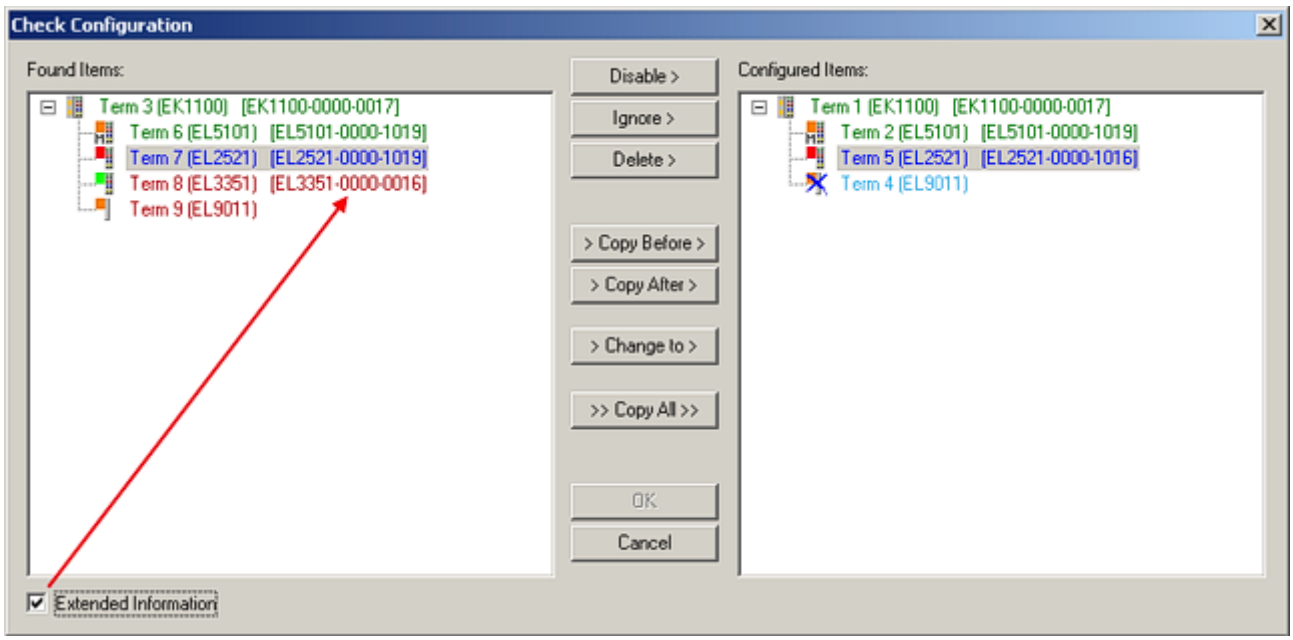


Abb. 197: Korrekturdialog

Die Anzeige der „Extended Information“ wird empfohlen, weil dadurch Unterschiede in der Revision sichtbar werden.

Farbe	Erläuterung
grün	Dieser EtherCAT Slave findet seine Entsprechung auf der Gegenseite. Typ und Revision stimmen überein.
blau	Dieser EtherCAT Slave ist auf der Gegenseite vorhanden, aber in einer anderen Revision. Diese andere Revision kann andere Default-Einstellungen der Prozessdaten und andere/zusätzliche Funktionen haben. Ist die gefundene Revision > als die konfigurierte Revision, ist der Einsatz unter Berücksichtigung der Kompatibilität möglich. Ist die gefundene Revision < als die konfigurierte Revision, ist der Einsatz vermutlich nicht möglich. Eventuell unterstützt das vorgefundene Gerät nicht alle Funktionen, die der Master von ihm aufgrund der höheren Revision erwartet.
hellblau	Dieser EtherCAT Slave wird ignoriert (Button „Ignore“)
rot	• Dieser EtherCAT Slave ist auf der Gegenseite nicht vorhanden • Er ist vorhanden, aber in einer anderen Revision, die sich auch in den Eigenschaften von der angegebenen unterscheidet. Auch hier gilt dann das Kompatibilitätsprinzip: Ist die gefundene Revision > als die konfigurierte Revision, ist der Einsatz unter Berücksichtigung der Kompatibilität möglich, da Nachfolger-Geräte die Funktionen der Vorgänger-Geräte unterstützen sollen. Ist die gefundene Revision < als die konfigurierte Revision, ist der Einsatz vermutlich nicht möglich. Eventuell unterstützt das vorgefundene Gerät nicht alle Funktionen, die der Master von ihm aufgrund der höheren Revision erwartet.

Geräte-Auswahl nach Revision, Kompatibilität

Mit der ESI-Beschreibung wird auch das Prozessabbild, die Art der Kommunikation zwischen Master und Slave/Gerät und ggf. Geräte-Funktionen definiert. Damit muss das reale Gerät (Firmware wenn vorhanden) die Kommunikationsanfragen/-einstellungen des Masters unterstützen. Dies ist abwärtskompatibel der Fall, d. h. neuere Geräte (höhere Revision) sollen es auch unterstützen, wenn der EtherCAT Master sie als eine ältere Revision anspricht. Als Beckhoff-Kompatibilitätsregel für EtherCAT-Klemmen/ Boxen/ EJ-Module ist anzunehmen:

Geräte-Revision in der Anlage >= Geräte-Revision in der Konfiguration

Dies erlaubt auch den späteren Austausch von Geräten ohne Veränderung der Konfiguration (abweichende Vorgaben bei Antrieben möglich).

Beispiel

In der Konfiguration wird eine EL2521-0025-**1018** vorgesehen, dann kann real eine EL2521-0025-**1018** oder höher (-**1019**, -**1020**) eingesetzt werden.

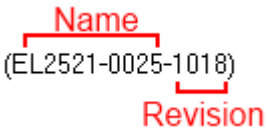


Abb. 198: Name/Revision Klemme

Wenn im TwinCAT System aktuelle ESI-Beschreibungen vorliegen, entspricht der im Auswahldialog als letzte Revision angebotene Stand dem Produktionsstand von Beckhoff. Es wird empfohlen, bei Erstellung einer neuen Konfiguration jeweils diesen letzten Revisionsstand eines Gerätes zu verwenden, wenn aktuell produzierte Beckhoff-Geräte in der realen Applikation verwendet werden. Nur wenn ältere Geräte aus Lagerbeständen in der Applikation verbaut werden sollen, ist es sinnvoll eine ältere Revision einzubinden.

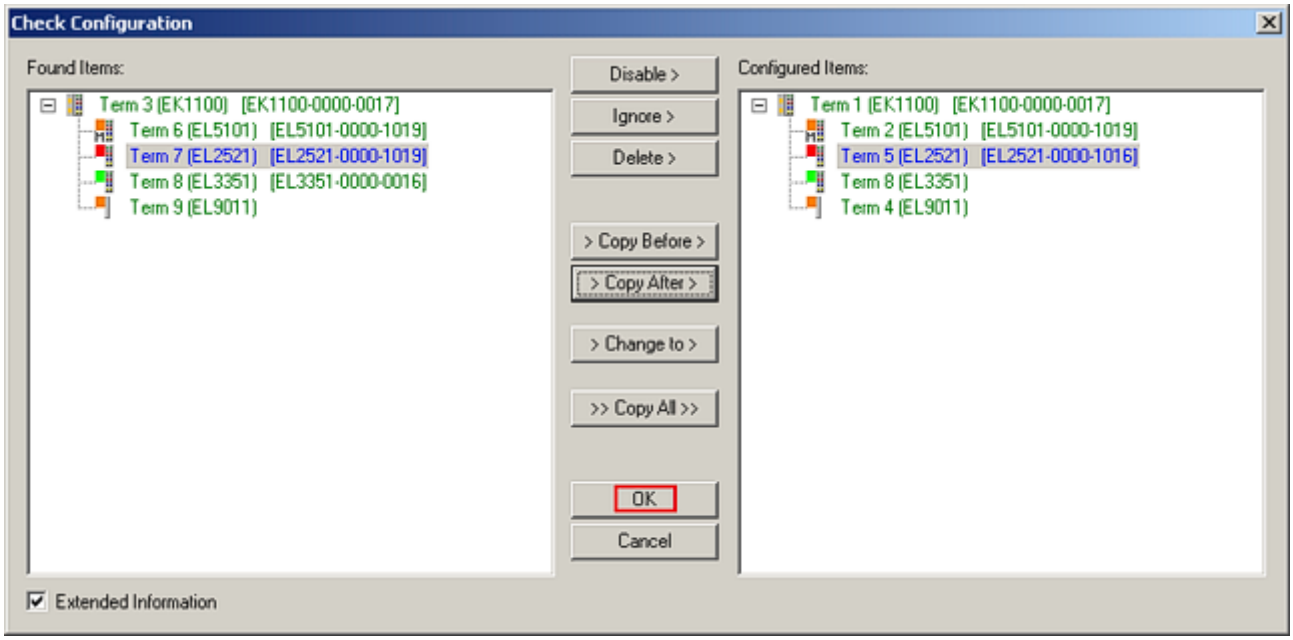


Abb. 199: Korrekturdialog mit Änderungen

Sind alle Änderungen übernommen oder akzeptiert, können sie durch „OK“ in die reale *.tsm-Konfiguration übernommen werden.

Change to Compatible Type

TwinCAT bietet mit „Change to Compatible Type…“ eine Funktion zum Austauschen eines Gerätes unter Beibehaltung der Links in die Task.

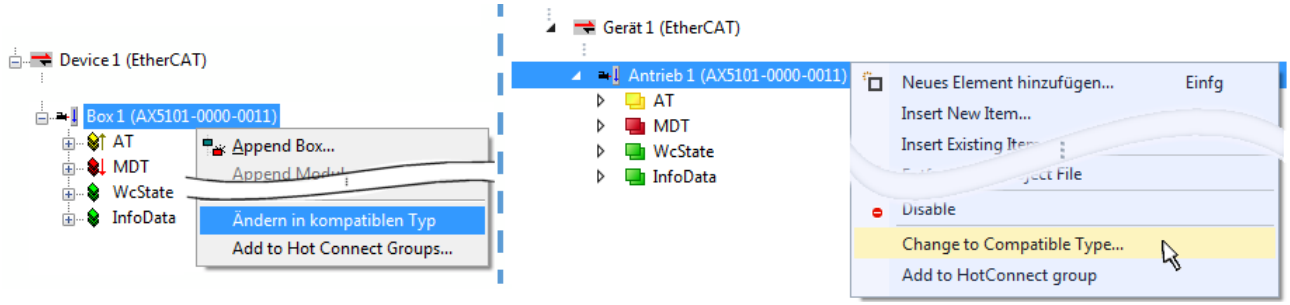


Abb. 200: Dialog „Change to Compatible Type…“ (links: TwinCAT 2; rechts TwinCAT 3)

Folgende Elemente in der ESI eines EtherCAT-Teilenhmers werden von TwinCAT verglichen und als gleich vorausgesetzt, um zu entscheiden, ob ein Gerät als „kompatibel“ angezeigt wird:

- Physics (z.B. RJ45, Ebus…)
- FMMU (zusätzliche sind erlaubt)
- SyncManager (SM, zusätzliche sind erlaubt)
- EoE (Attribute MAC, IP)
- CoE (Attribute SdoInfo, PdoAssign, PdoConfig, PdoUpload, CompleteAccess)
- FoE
- PDO (Prozessdaten: Reihenfolge, SyncUnit SU, SyncManager SM, EntryCount, Entry.Datatype)

Bei Geräten der AX5000-Familie wird diese Funktion intensiv verwendet.

Change to Alternative Type

Der TwinCAT System Manager bietet eine Funktion zum Austauschen eines Gerätes: Change to Alternative Type

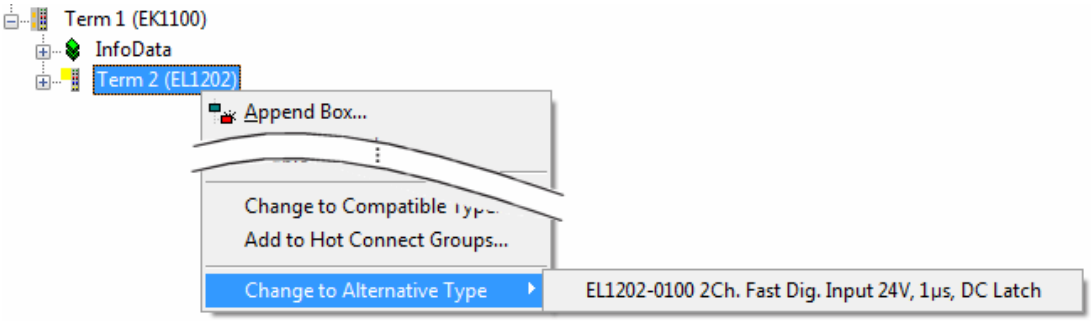


Abb. 201: TwinCAT 2 Dialog Change to Alternative Type

Wenn aufgerufen, sucht der System Manager in der bezogenen Geräte-ESI (hier im Beispiel: EL1202-0000) nach dort enthaltenen Angaben zu kompatiblen Geräten. Die Konfiguration wird geändert und gleichzeitig das ESI-EEPROM überschrieben - deshalb ist dieser Vorgang nur im Online-Zustand (ConfigMode) möglich.

3.6.7 Standard-Verhalten EtherCAT Master

3.6.7.1 Allgemeines

TwinCAT unterstützt zur einfachen und schnellen Inbetriebnahme den Anwender durch einige Default-Einstellungen und Automatismen. In den meisten Fällen sind diese Einstellungen ausreichend für einen stabilen Anlagenbetrieb.
Für kundenspezifische Behandlung oder Sonderverhalten sei hier auf diese Einstellungen, ihre Auswirkungen und die Einstellmöglichkeiten hingewiesen. Folgende Themen werden behandelt:

- Prozesdaten Info
- EtherCAT Master Settings

- Slave Settings
- Sync Task
- Distributed Clock Settings
- EoE

Für einen regulären ordnungsgemäßen Betrieb des EtherCAT Systems sind folgende Elemente zu prüfen:

Element	Kontrollmöglichkeit online/Inbetriebnehmer	Kontrollmöglichkeit durch Applikation
TwinCAT auf dem Zielsystem im RUN-State (oder CONFIG/ FREERUN)	TwinCAT Icon auf dem Zielsystem (nicht Programmiersystem!) ist grün bzw. blau System Manager Angabe grün bzw. blau/rot blinkend (FreeRun) 	Applikation prüft über ADS TwinCAT Zustand
der EtherCAT Master im State OP	s. Abb. *Online Diagnose EtherCAT Device*, A	Abfrage EcMasterState über ADS (PLC: Baustein aus TcEtherCAT.lib) ADS NetId des EcMasters bekannt aus Device Infodaten (s. Abb *Online Diagnose EtherCAT Device*)
alle EtherCAT Slaves im State OP	s. Abb. *Online Diagnose EtherCAT Device*, B	Abfrage EcSlavesState über ADS (PLC: Baustein aus TcEtherCAT.lib)
die zyklischen Telegramme entsprechend der Zykluszeit werden verschickt	s. Abb. *Online Diagnose EtherCAT Device*, C	Abfrage über ADS
gelegentlich werden azyklische Telegramme verschickt	s. Abb. *Online Diagnose EtherCAT Device*, D	Abfrage über ADS
keine oder wenige LostFrames/ CRC in den Slaves	s. Abb. *Online Diagnose EtherCAT Device*, D	Abfrage über ADS
EtherCAT DevState = 0	s. Abb. *Online Diagnose EtherCAT Device*, E	Link in die überwachende Task
alle WorkingCounter der Slaves = 0 durchgehend	s. Abb. *Online Diagnose EtherCAT Device*, F	Link in die überwachende Task oder Sammel-Information Frm0WcState aus den EcMaster Inputs
keine auffälligen Ausgaben im Logger-Fenster		- (Ursachen für Loggerausgaben werden bei korrekter Diagnose über andere Wege festgestellt)
keine Zykluszeitüberschreitungen		PLC: Einbindung TcUtilities.lib, dadurch eingeloggt Zugriff auf SystemInfo und SystemTaskInfoArr[]
keine Ebus-Strom Überschreitung	s. Abb. *Online Diagnose EtherCAT Device*, G	-
div. Watchdogs eingehalten (Klemmen standard 100 ms, FSoE mit Rückbestätigung 100 ms)		wird durch Überwachung der States festgestellt

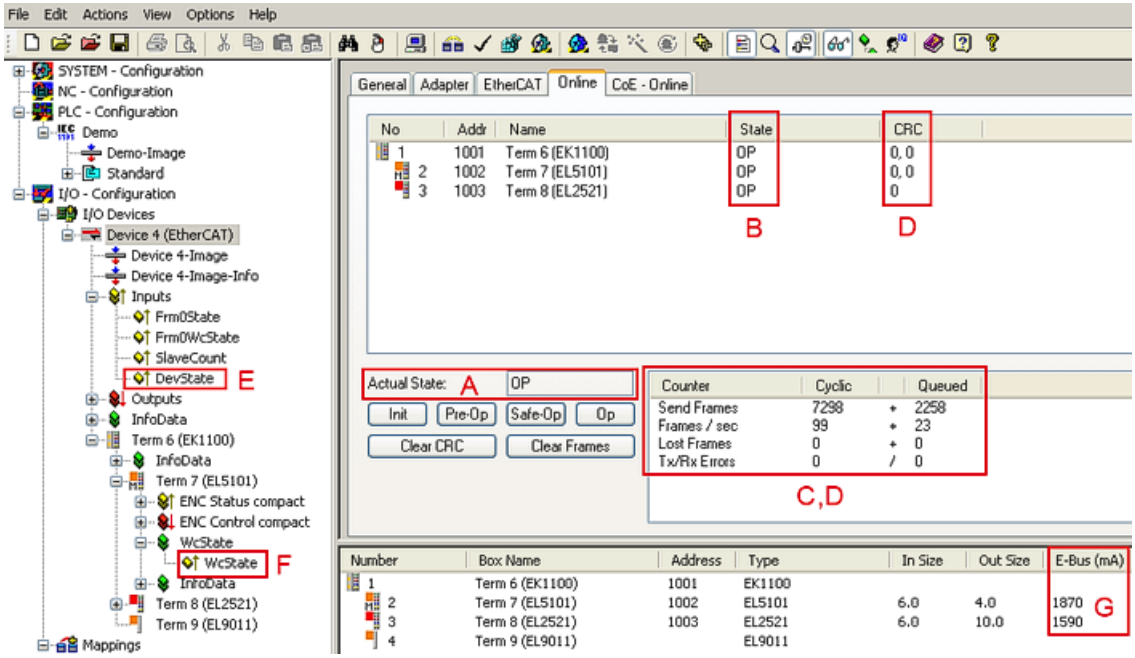


Abb. 202: Online Diagnose EtherCAT Device

Die Standard-Automatismen im System Manager stellen diesen Zustand her sobald die Konfiguration aktiviert wurde und TwinCAT in RUN/CONFIG versetzt wird.

3.6.7.2 Default Einstellungen und Angaben

Prozessdaten Info

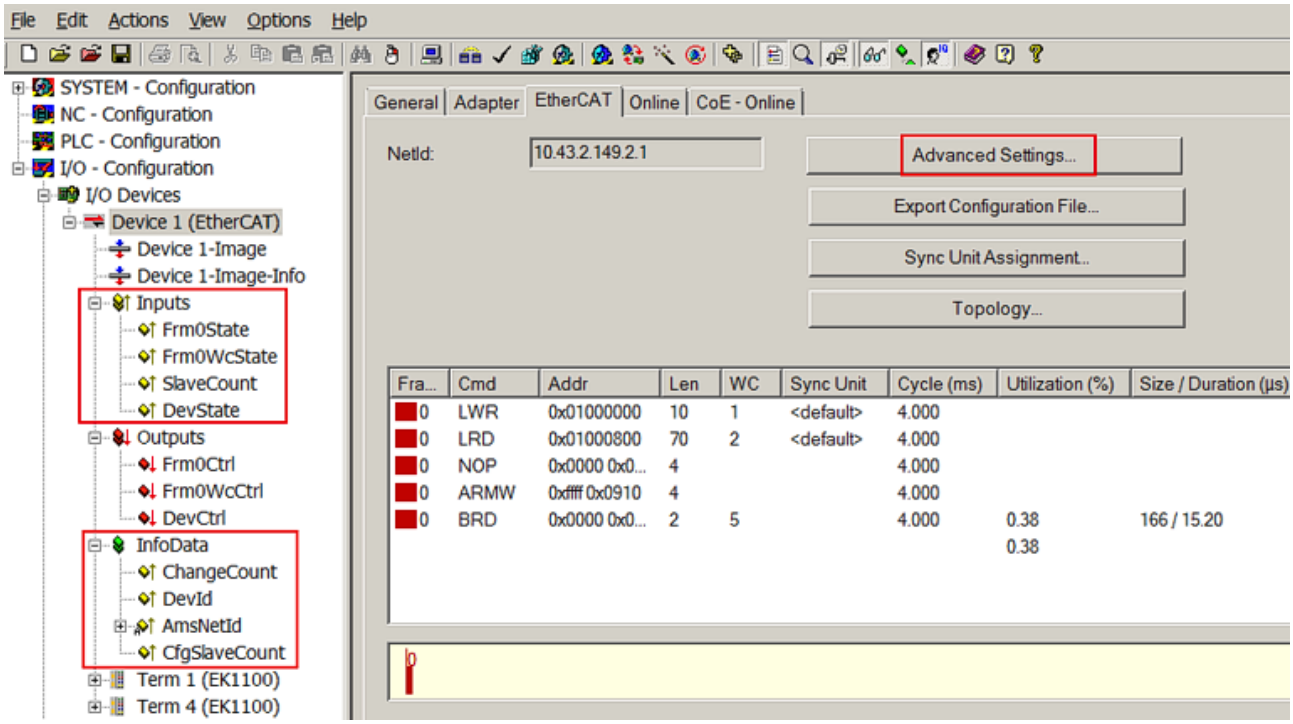


Abb. 203: Prozessdaten

Der EtherCAT Master verfügt über Infodaten die zyklusaktuell eine Diagnose liefern (gelbe Variablen) und allgemeine Info außerhalb des Echtzeitkontext (grüne Variablen). Die wichtigsten im Folgenden:

- **DevState**: soll = 0 sein, dann sind alle Slaves im OP, kein Link Fehler etc.

- **Frm0WcState**: soll ebenfalls =0 sein. Für jeden zyklischen Ethernet-Frame wird eine solche Variable angelegt (Frm0WcState, Frm1WcState, ...)
  Eine Anwendung soll mindestens diese beiden Master-Inputs zyklusaktuell prüfen und überwachen.
- **AmsNetId**: diese AMS-Adresse benötigt die Applikation (PLC, externe Task) um über ADS den EtherCAT Master bzw. die unterlagerten Slaves anzusprechen

Über die Erweiterten Einstellungen/AdvancedSettings sind weitere Einstellungen zugänglich:

Master Settings

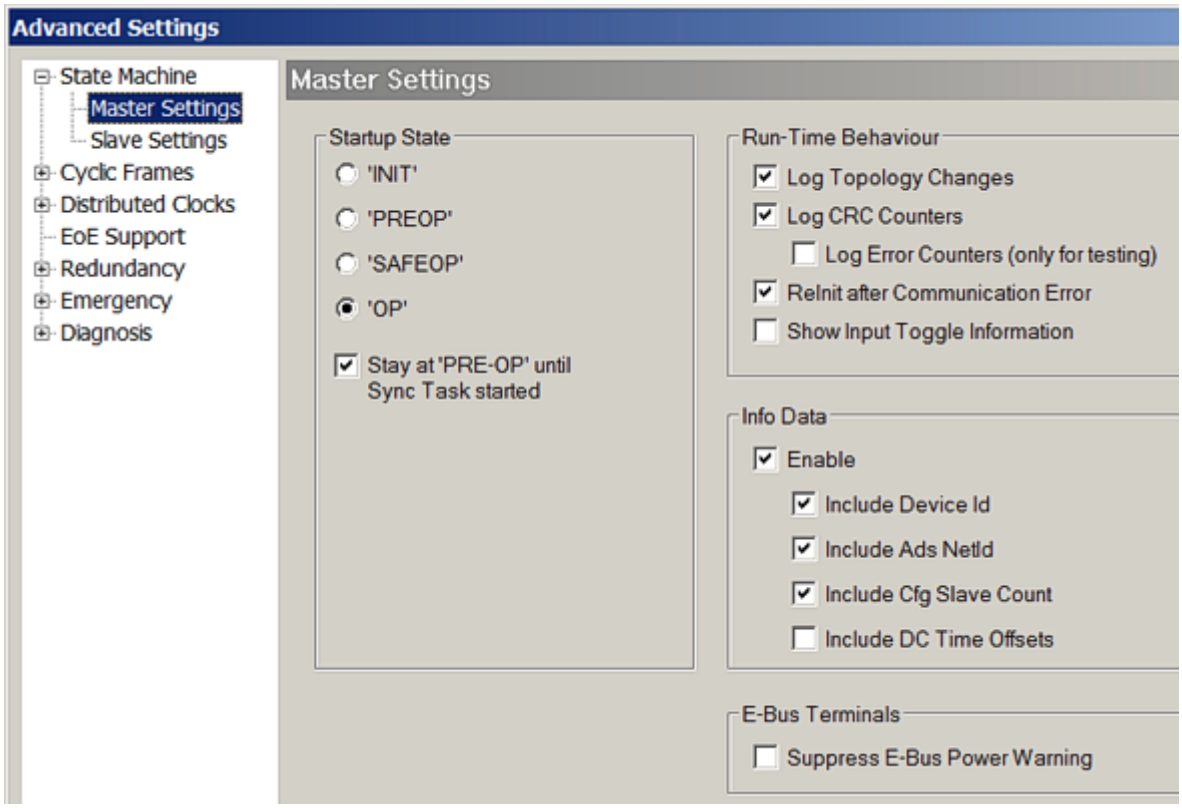
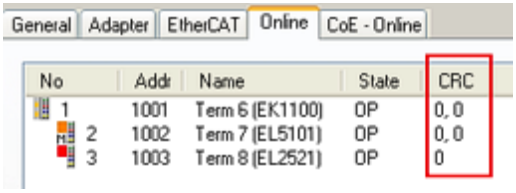
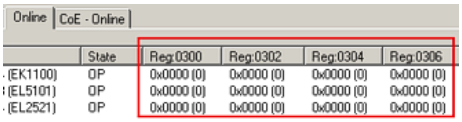
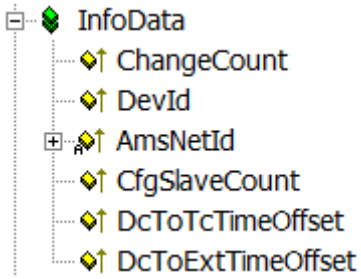


Abb. 204: Master Settings

Element	Detail	Erklärung	Auswirkungen
StartUp State		Sobald TwinCAT "gestartet" wird (RUN oder COnfig/ FreeRun) wird der Master in den hier gewählten State gesetzt. Es wird allerdings mit dem Übergang nach OP gewartet, bis die Sync-Task gestartet ist. Bei mehreren Tasks auf einem System ist die höchstpriore die Sync-Task, die auch die Distributed-Clock-Regelung beinhaltet. DC-fähige Slaves lassen sich jedoch nicht in OP schalten, wenn ihre lokale Clock nicht eingeregelt ist bzw. sie fallen wieder aus dem OP-state wenn die Regelung misslingt ("Sync lost").	Es wird empfohlen, den EcMaster State aus der Applikation (PLC, so vorhanden) zu setzen und zu überwachen (FB_EcGet/SetMasterState aus TcEtherCAT.lib). Dadurch kann der Master auch nach schwerwiegenden Kommunikationsfehlern wieder in den OP gesetzt werden.
Run-Time Behaviour	Log Topology Changes	Standardmäßig aktiviert Online-Ausgaben im Loggerfenster werden aktiviert	Deaktivierung nicht sinnvoll
	Log CRC Counters	Standardmäßig aktiviert; es werden im OnlineView die CRC-Fehler der Slaves aus dem Feld ausgelesen und kumuliert gesammelt. 	Wenn im Online View die CRC-Registerzähler in den Slaves gezielt angezeigt werden sollen, ist diese Option zu deaktivieren - sie löscht nämlich nach dem Auslesen die lokalen Register x0300ff damit diese nicht bei xFF vollaufen. 
	Log Error Counters	keine Funktion	
	ReInit after Communication Error	Nach einem Kommunikationsfehler bei dem der Master den OP-state verlassen hat (Verbindung getrennt und >10 Zyklen lang Lost Frames, Stationen abgeschaltet), versucht TwinCAT den Master wieder in den OP-State zu versetzen.	Wenn der EcMaster State aus der Applikation gesteuert wird, muss diese Option unbedingt deaktiviert werden, da sich sonst beide Mechanismen behindern können. Beide greifen über ADS auf den Master zu.
	Show Input Toggle Information	Wenn aktiviert, wird bei Eingangsklemmen eine zusätzliche Toggle-Variable eingeblendet die verlinkt werden kann. Sie ändert ihren Zustand 0/1 bei jedem neu empfangenen Datagramm.	
Info Data		Das Einblenden dieser (grünen) Nicht-Echtzeit Informationsdaten im System-Manager-Baum kann hier deaktiviert werden.	DeviceId: nützlich für den Zugriff aus der Applikation AdsNetId: nötig für den Zugriff aus der Applikation CfgSlaveCount: Anzahl der bisherigen Konfigurationsänderungen Dc Time Offsets: die zur Laufzeit konstanten Offsets zwischen externer, interner und TwinCAT Clock werden eingeblendet. Nötig für die externe EtehrCAT Synchronisierung. 
Ebus Power Warning		Standardmäßig warnt der System Manager vor Überschreitung der max. Belastung eines EtherCAT Kopplers (z. B. EK1100).	

Slave Settings

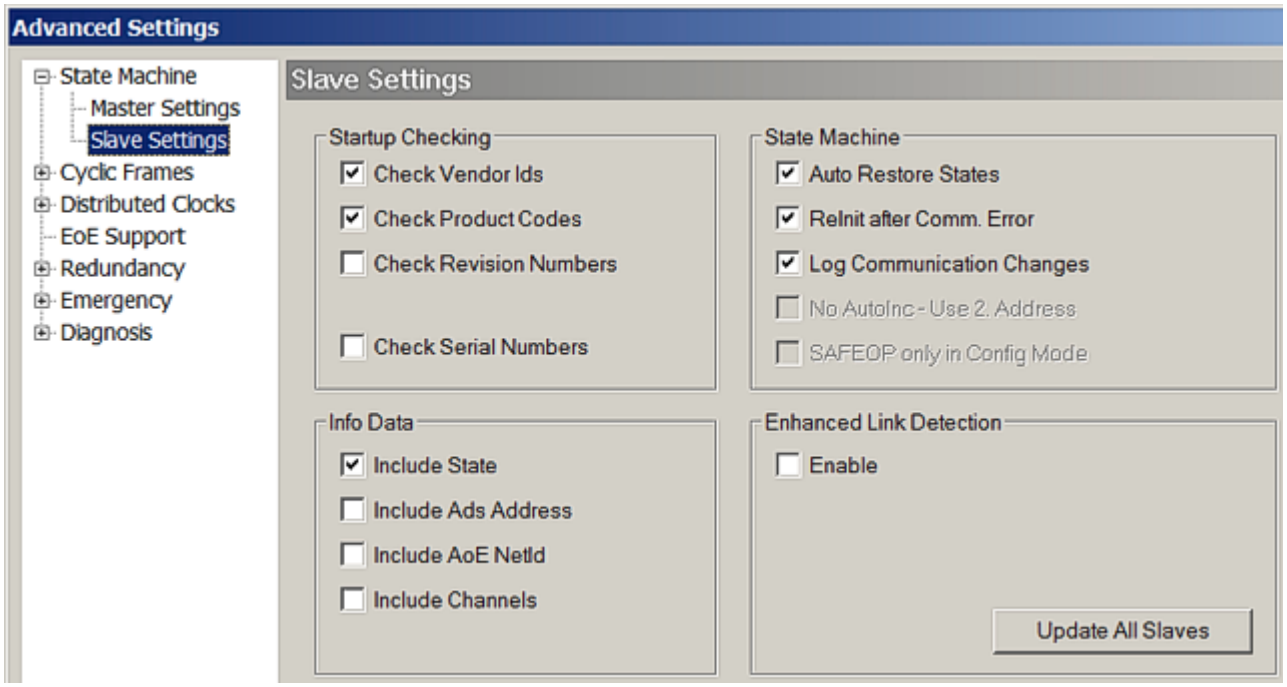
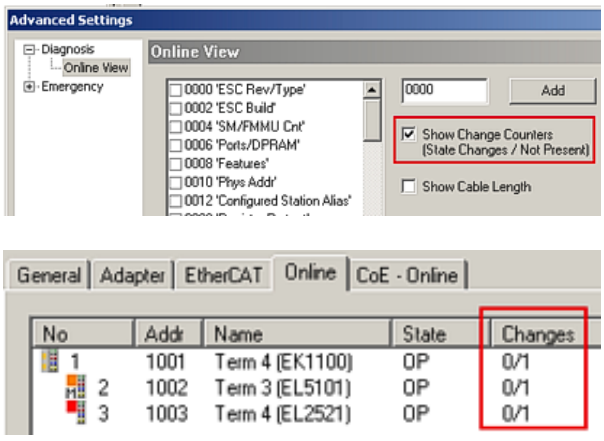


Abb. 205: Slave Settings

Element	Detail	Erklärung	Auswirkungen
StartUp Checking		Beim Hochlauf von EtherCAT werden die hier aktivierten Eigenschaften aller Slaves überprüft. Entsprechende Settings in den Slaves gehen allerdings vor!	Standardmäßig werden VendorID und ProductCode (z. B. EI2521-0010) geprüft. Dies wird empfohlen, denn dadurch können weiterentwickelte aber typgleiche Geräte mit höherer Revision im Austauschfall eingesetzt werden.
State Machine	Auto Restore States	Wenn der Slave aus eigenen Gründen (Energieverlust, Synchronisierungsfehler) den OP-State verlassen hat, versucht der EtherCAT Master bei aktivierter Checkbox den Slave wieder in den OP-State bzw. den zuletzt regulär erreichten State zu setzen.	Es wird empfohlen, den Slave State State aus der Applikation (PLC, so vorhanden) zu setzen und zu überwachen (FB_EcGet/SetMasterState aus TcEtherCAT.lib). Dadurch kann die Applikation den Slave in Übereinstimmung mit applikationsspezifischen Erfordernissen ansteuern. Beispiel Servoachse: der EtherCAT Master würde die Achse sobald möglich einfach wieder in den OP-State setzen, ohne tiefere Kenntnis über Sicherheits- oder Funktionszusammenhänge. Die Applikation hingegen kann entscheiden, ob diese Achse nach dem schweren Fehler "State Ausfall" überhaupt und wann wieder in den OP-State gesetzt werden darf. Des weiteren kann ein DeadLock eintreten: wird der Slave vom Master neu in den OP gesetzt, erreicht aber nur SAFEOP und wird dann bereits wieder gestört, wird der Master im Weiteren nur noch versuchen den Slave in den SAFEOP-State zu setzen, er wird den OP-nicht mehr erreichen. Deshalb wird eine Ansteuerung von Master- und Slave-State durch die Applikation empfohlen.
	ReInit after Comm.Error	Wenn die Kommunikation zu einem Slave unterbrochen wurde, startet der Master den Slave bei wiederhergestellter Verbindung neu durch den INIT-State, auch wenn der Slave für sich nur in den SAFEOP-State zurückgefallen ist. Dadurch wird ein sicherer Hochlauf und eindeutiger Zustand des Slaves erzielt.	Bei einem Neustart eines Slaves INIT --> OP fallen i.d.R. die Ausgänge ab.
	Log Communication Changes	Standardmäßig aktiviert; wird im Online View die Option "Show Change Counter" aktiviert, werden die State-Wechsel angezeigt. 	Deaktivierung nicht sinnvoll
Info Data		Das Einblenden dieser (grünen) Nicht-Echtzeit Informationsdaten im System-Manager-Baum kann hier (de)aktiviert werden.	Das Einblenden der ADS-Adresse in jedem Slave ist z. B. nützlich für die Verlinkung mit einem slavespezifischen FUNCTIONBLOCK, der einen Slave überwachen soll.
Enhanced Link Detection		Diese Funktion ist nicht für den allgemeinen Gebrauch bestimmt.	Wird diese Funktion auf EtherCAT Geräte angewendet, die dies nicht unterstützen kann zur dauerhaften und irreversiblen Störung der EtherCAT Kommunikation kommen. Bei Geräten, die diese Funktion unterstützen, ist diese Funktion bereits herstellerseitig durch die in der Produktion aufgespielte ESI aktiviert.

Sync Task

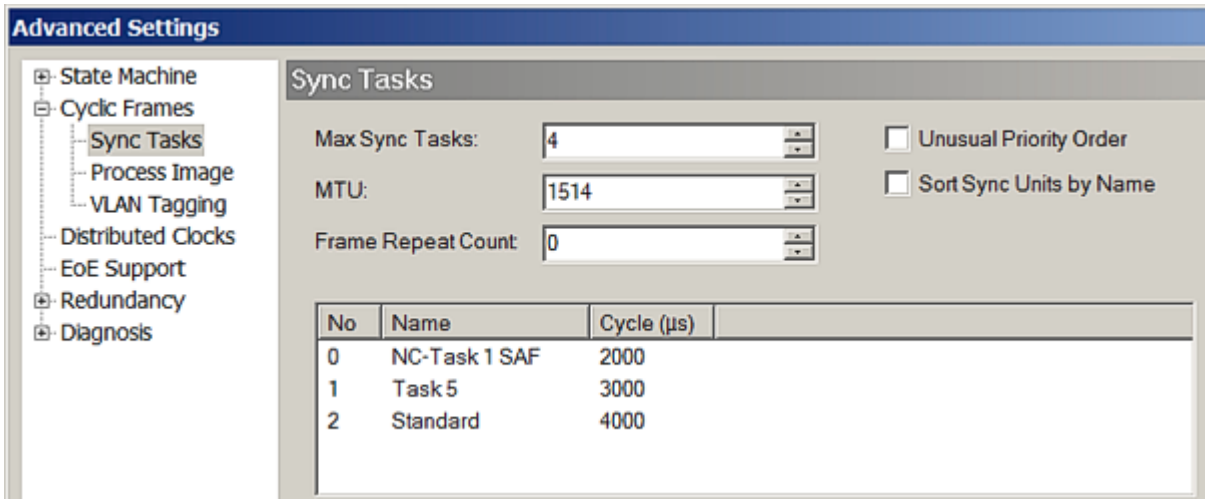


Abb. 206: Sync Task

Element	Erklärung
Max Sync Task	TwinCAT 2.10/2.11 unterstützt max. 4 SyncTasks. Eine SyncTask ist eine Task (PLC, NC) die ein I/O-Update antriggert, also mit eigenen EtherCAT-Frames in zyklischer und fester Zykluszeit mit dem I/O-Feld kommuniziert. In Abb. *Sync Task*. sind in der Konfiguration 3 Tasks in Gebrauch mit Zykluszeiten von 2, 3 und 4 ms. In der Taskpriorisierung ist auf richtige Reihenfolge entsprechend zu achten. Sind mehr als 4 Tasks in der Konfiguration vorhanden, werden von großen Zykluszeiten her die Tasks in die langsamste Task gesetzt. TIPP: Dies kann genutzt werden, wenn eine sehr langsame PLC-Task > 100 ms genutzt werden soll, die I/O-Kommunikation aber wegen dem Slave-Watchdog schneller betrieben werden muss .Dann ist die Anzahl "Max Sync Task" soweit zu reduzieren, bis nur noch Tasks <100 ms übrigbleiben.
MTU	Die "Max Transfer Unit " (MTU) ist die maximale Byte-Länge eines Ethernet-Frames mit EtherCAT Datagrammen.
Frame Repeat Count	Der TwinCAT EtherCAT Master unterstützt das Mehrfach-Senden von EtherCAT Frames zum Zwecke der erhöhten Störsicherheit. ACHTUNG: die verwendeten und betroffenen EtherCAT Slaves müssen dies unterstützten. Der Slave-Hersteller spezifiziert dies in der ESI-Beschreibung.

Distributed Clocks

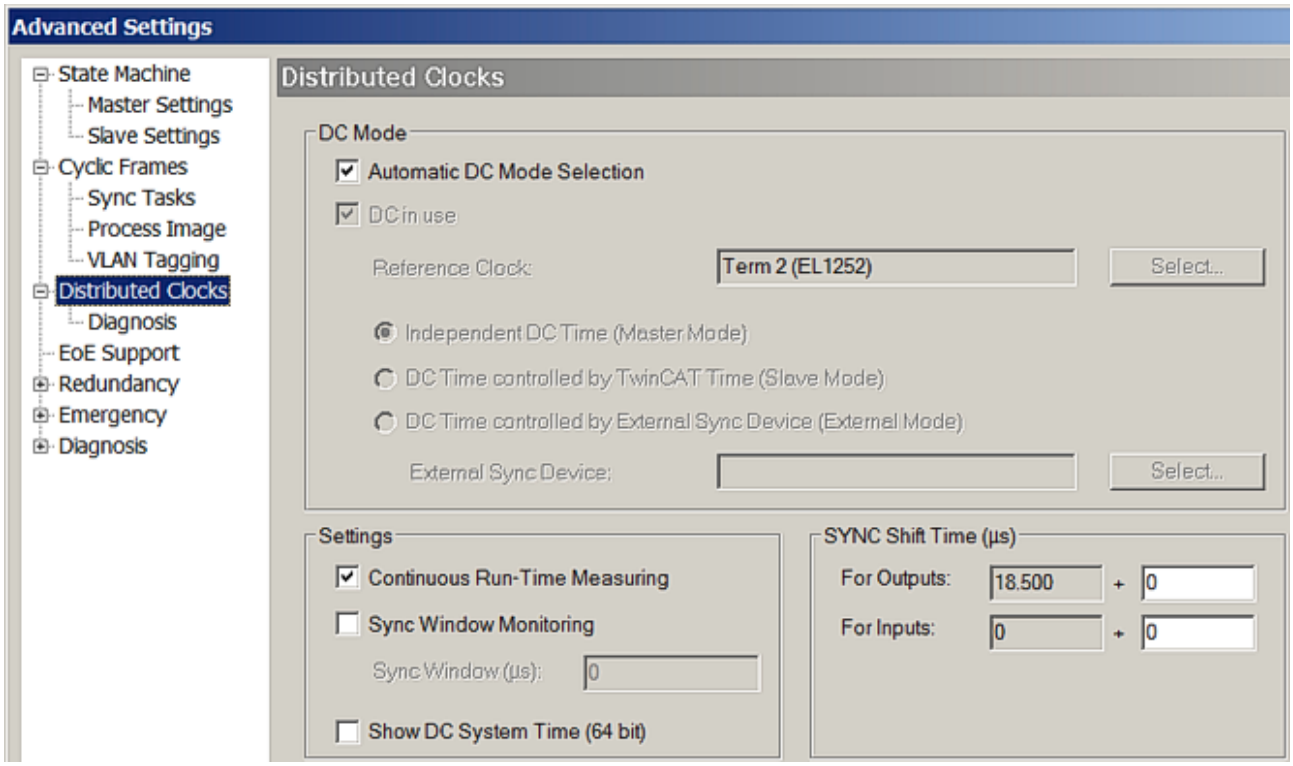


Abb. 207: Distributed Clocks

Diese Einstellungen werden in einem besonderen Kapitel [▶ 154] besprochen.

EoE Support (Ethernet over EtherCAT)

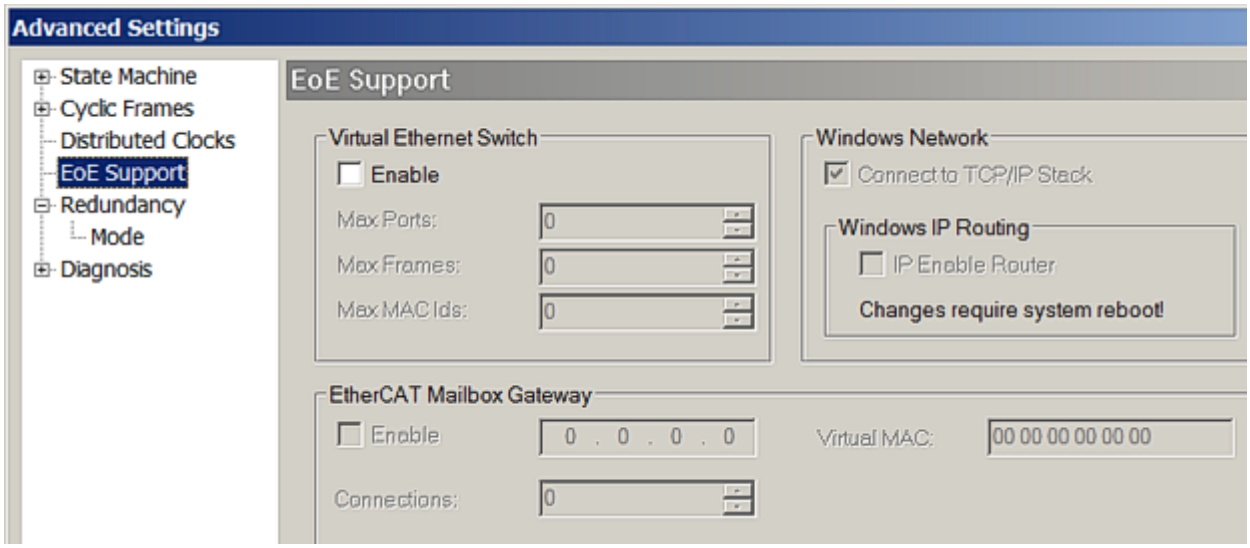


Abb. 208: Ethernet over EtherCAT Support

Element	Detail	Erklärung	Auswirkungen
Virtual Ethernet Switch		Die Durchleitung von Standard TCP/IP-Verkehr über den virtuellen Switch innerhalb des TwinCAT-EtherCAT-System wird in diesen Einstellungen automatisch abhängig von den verwendeten Slaves gesetzt. EL6601 (SwitchPort-Klemmen) z. B. führen hier zu einer Aktivierung des VirtualEthernetSwitch und Hinzufügung von Ports. (s. Abb. *„Aktivierung des VirtualEthetnetSwitch“)* Weiterführende Hinweise sind deshalb den entsprechenden Klemmendokumentationen (EL6601, EL6614) zu entnehmen.	Die ausdrückliche Aktivierung und Vorgabe von Ports ist nötig, wenn z. B. zu einem intelligenten Antrieb über EoE zur Parametrierung oder Firmware-Update kommuniziert werden soll. Dann ist für jedes angeschlossene Gerät ein Port anzulegen. Die Anzahl "Max.Frames" stellt die interne Queue dar und kann erhöht werden, wenn es zu Durchsatzproblemen kommt. In diesem Fall ist allerdings zuerst die verwendete EtherCAT-Zykluszeit und die Mailbox-Größen zu prüfen. Siehe dazu ebenfalls die EL6601-Dokumentation.

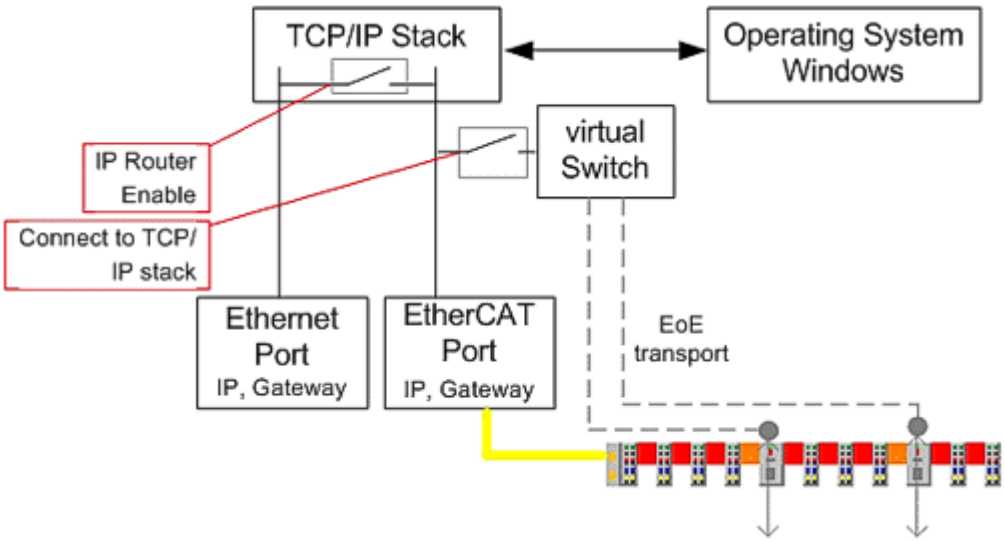


Abb. 209: Aktivierung des VirtualEthetnetSwitch

Element	Detail	Erklärung	Auswirkungen
Windows Network	IP Routing	Siehe dazu das vorangehende Bild.	
EtherCAT Mailbox Gatway		Diese Einstellung wird für spezielle Slaves benötigt.	

Cable Redundancy

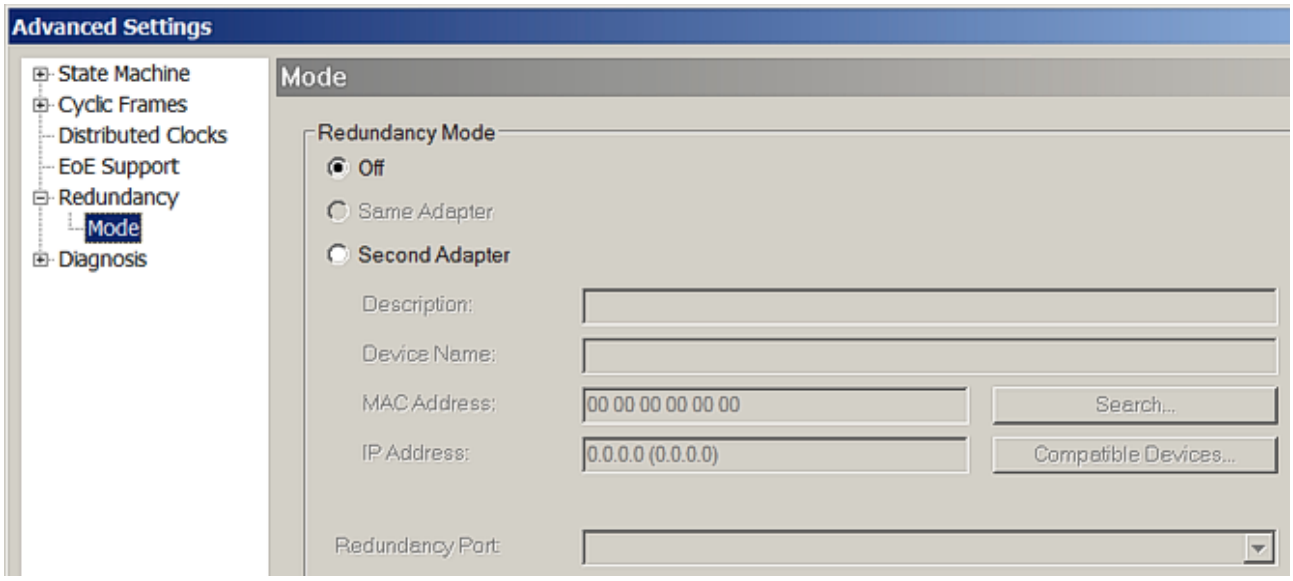


Abb. 210: EtherCAT Kabelredundanz

Im gesonderten Kapitel Kabel-Redundanz [▶ 30] werden die Optionen zur Medienredundanz besprochen. Zur Nutzung ist eine TwinCAT Supplement Lizenz erforderlich.

Wird hier ein Ethernet-Port eingetragen (Installation Realtime-Treiber, siehe hier) kann diese Konfiguration nicht in den RUN-State versetzt werden, wenn die Lizenz fehlt.

Sync Unit Zuordnung

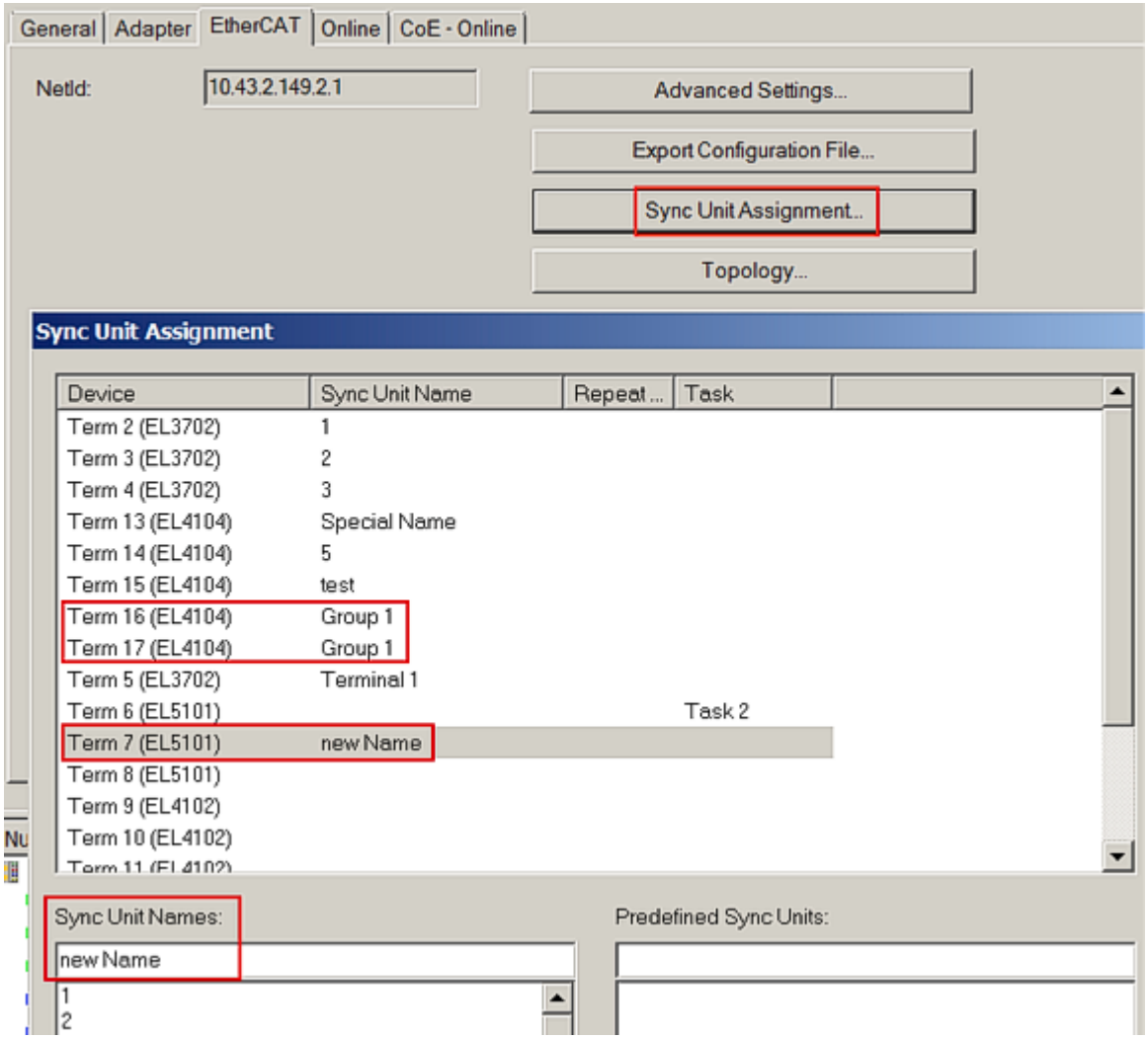


Abb. 211: Sync Unit Zuordnung

Die Sync-Unit-Zurodnung betrifft nur die zyklischen Daten des EtherCAT Systems.

Der System Manager nimmt standardmäßig eine sehr effiziente Zuordnung von zyklischen I/O-Daten und versendeten Datagrammen vor. Das bedeutet, es werden möglichst viele/alle zyklischen Daten in einen/ wenige Datagramme gepackt. Dadurch ergibt sich eine geringe Buslast dank wenig Telegramm-Overhead. Mit wenigen, im besten Fall *einem* Datagram werden möglichst viele, im besten Fall *alle* Slaves angesprochen.

Ein Diagnosemittel im EtherCAT System ist der Working-Counter. Jeder Slave, der auftragsgemäß ein Datagram bearbeitet (Daten hineinschreibt oder herausliest) erhöht den so genannten WorkingCounter (WC). Der Master schickt die Datagramme mit WC=0 los und erwartet sie mit WC>0 zurück. Anhand der WC-Überprüfung kann der Master sofort feststellen, ob alle angesprochenen Slaves das Datagram korrekt bearbeitet haben - ist dies nicht der Fall, kann der Master den zurück gelieferten Daten nicht trauen und er verwirft *alle* Inputdaten aus diesem Datagram. Darüber hinaus beginnt er mit azyklischen Diagnosemaßnahmen zur Feststellung des Fehlerortes.

Wird in einer Anlage mit Auftreten von Working-Counter-Fehlern gerechnet, z. B. weil das flexible Topologiekonzept "HotConnect" genutzt wird, können Koppler-Baugruppen oder einzelne Slaves/Klemmen in diesem Dialog in eigene Datagrame gelegt werden, den sog. SyncUnits. Im Extremfall erhält jeder Slave ein eigenes Datagram, dies bedeutet eine sehr ineffiziente Busausnutzung da viele Datagramme und Ethernet-Frames mit entsprechenden Overhead versendet werden müssen. Hinweis: Ein Ethernet-Frame kann max. 16 EtherCAT Datagramme beinhalten.

3.7 Hinweise Distributed Clocks

3.7.1 EtherCAT Distributed Clocks - Standardeinstellung

Allgemeines

Die Distributed -Clocks-Technologie im EtherCAT-System ermöglicht den synchronisierten Betrieb von lokalen Uhren in allen EtherCAT-Teilnehmern (Master + Slaves). Unterstützt ein EtherCAT-Slave Distributed Clocks (DC), liegt in seinem ESC (EtherCAT Slave Controller) eine Hardware-implementierte Uhr mit einem Umfang von 64 Bit (seltener: nur 32 Bit) und einer Auflösung von 1 Bit = 1 ns vor. Mithilfe dieser lokalen Uhr können nun synchron Ausgaben oder Datenerfassungen (z. B. analoger Eingang) vorgenommen werden. Ein EtherCAT Slave kann, muss aber nicht DC unterstützen. Ein Mischbetrieb im EtherCAT System ist möglich solange der EtherCAT Master DC unterstützt.

Ein Teilnehmer ist dabei die Referenzuhr, alle anderen DC-fähigen Teilnehmer werden dieser Uhr fortlaufend mit einer Genauigkeit von üblicherweise <100 ns Abweichung nachsynchronisiert. Das Synchronisierungsverfahren und die Kommunikationsweise von EtherCAT bedingen, dass der **erste** DC-fähige Slave im System die Referenzuhr darstellt ("M" in Abb. *Topologie eines EtherCAT-System mit DC-fähigen Teilnehmern*). Die nachfolgenden Slaves erhalten zyklisch über ein spezielles Telegramm Information über den Stand der Referenzuhr und können sich dieser Zeit nachregeln.

Daten über den Nachregelungsprozess sind wichtige Diagnoseinformation über den Zustand des Distributed-Clocks-Systems.

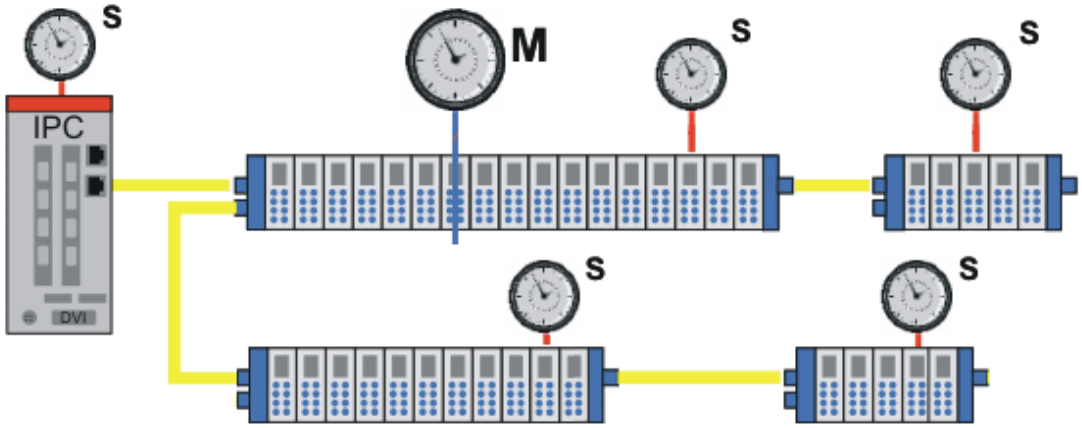


Abb. 212: Topologie eines EtherCAT-System mit DC-fähigen Teilnehmern

Die grundsätzlichen topologieabhängigen Berechnungen, die Einregelung beim EtherCAT-Start und die fortlaufende Synchronisierung obliegen dabei dem EtherCAT-Master TwinCAT. Die Einstellungen dazu werden in den Dialogen im System Manager Konfigurator vorgenommen.

Weitere Hinweise und detaillierte Informationen über das Distributed-Clocks-System sind entsprechenden Kapiteln [▶ 234] zu entnehmen.

Distributed Clocks in Betrieb

Das Distributed-Clock-System wird beim EtherCAT-Hochlauf im Übergang von PREOP nach OP einsynchronisiert, dann werden die Slaves in OP-State gesetzt. In DC-fähigen Slaves ist meist die einwandfreie Synchronisierung im OP-State von Bedeutung, sonst gehen die Slaves eigenständig in den PREOP-State zurück. TwinCAT 2.11 kann solche Teilnehmer wieder einsynchronisieren und in OP setzen.

Standardeinstellungen EtherCAT Master

Wirksamkeit von Änderungen

Das Distributed-Clock-System wird beim EtherCAT-Hochlauf analysiert und berechnet. Einstellungsänderungen in diesem System erfordern also zum Wirksamwerden immer eine Aktivierung der geänderten Konfiguration und einen EtherCAT-Neustart.

Das Distributed-Clocks-System (DC) wird standardmäßig so berechnet, dass die üblichen I/O-Konfigurationen damit stabil lauffähig sind. Dennoch kann unter Benutzung der geg. Diagnosemittel eine Korrektur der Einstellung bei der Maschineninbetriebnahme geboten sein.

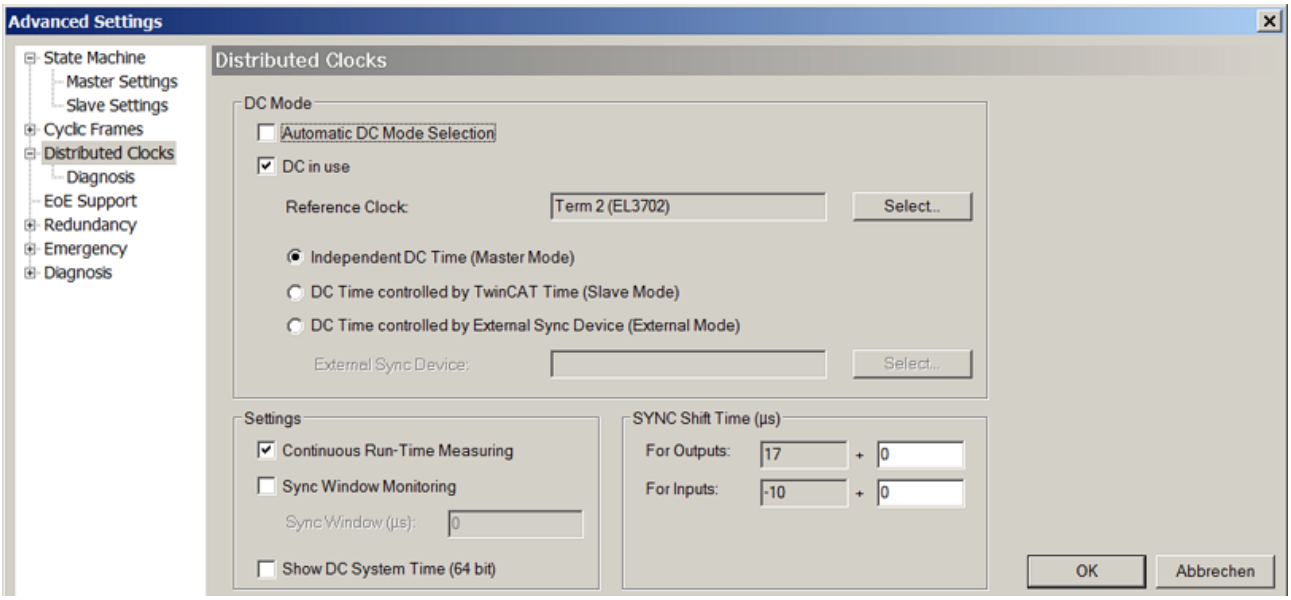


Abb. 213: Distributed Clock Master Settings

Standardmäßig ist die "Automatic DC mode selection" in Betrieb. Nur wenn explizit Änderungen an der automatischen Konfiguration vorgenommen werden sollen, ist "DC in use" auszuwählen.

Element	Detail	Erklärung	Auswirkungen
DC mode	Automatic DC mode selection	Standardeinstellung, automatische Auswahl der ReferenceClock	
	DC in use	Auswahl der ReferenceClock (s. nachfolgenden Absatz) und Synchronisierungsrichtung (s. Kapitel "Kopplung von EtherCAT [▶ 161]") kann manuell vorgenommen werden. Ist nur ein EtherCAT-Device in der Konfiguration vorhanden und werden DC-Slaves genutzt, ist hier "Independent DC Time" zu nutzen (Ausnahme: externe Synchronisierung [▶ 282]).	Diese Einstellungen sind mit Vorsicht zu verändern! Die Stabilität des Gesamtsystem kann beeinträchtigt werden.
Settings	Continuous Runtime Measuring	zyklisch werden während der Laufzeit die zeitlichen Abständen zwischen den Teilnehmern vermessen. Dieser Prozess findet auch bei EtherCAT Start statt.	Für neue Anwendungen unter TwinCAT 2.11 wird eine Deaktivierung dieser Funktion empfohlen.
	Sync Window Monitoring	Wenn aktiviert, wird im EtherCAT *DevState* in Bit 12 angezeigt, ob alle DC-Teilnehmer ihre lokalen Uhren innerhalb des angegebenen Fensters halten (s. Abb. *DevState mit Anzeige SyncWindow Monitoring*). Dafür wird ein zyklisches BRD-Kommando auf x092C (Systemzeit Differenz) verwendet. Die Anzeige ist nur verwertbar, wenn der erste EtherCAT-Teilnehmer auch die ReferenceClock beinhaltet.	
	Show DC System Time (64 bit)	Wenn aktiviert, wird in den Eingängen des EtherCAT Master die aktuelle DC-Zeit als Kopie aus der Masterclock angezeigt. Da der Auslesevorgang dem Feldbustransport unterliegt, sollte zur Gewinnung der aktuellen DC-Systemzeit PLC-Bausteinen der Vorzug gegeben werden. (s. Abb. *“Anzeige „DcSysTime“ im TwinCAT-Baum“*)	

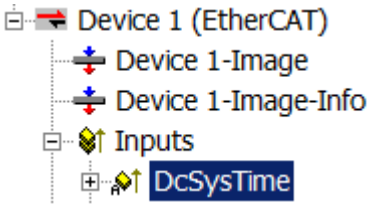


Abb. 214: Anzeige „DcSysTime“ im TwinCAT-Baum

Element	Detail	Erklärung	Auswirkungen
SYNC Shift Time		Es werden die automatisch berechneten Shift-Zeiten für Inputs und Outputs gezeigt (graue Felder). Zusätzlich können die Slave-lokalen Shift-Ereignisse durch manuelle Einträge verschoben werden. Hinweise dazu im allgemeinen DistributedClock-Kapitel [▶ 234].	Ggf. sind Ergebnisse aus der DC-Diagnose hier einzubringen. Im allgemeinen kann bei Synchronisierungsproblemen hier für die Outputs +10..20% der Zykluszeit, für Inputs -10..20% der Zykluszeit als Anhaltswert eingegeben werden (Einheit: µs). Werte >100% der Zykluszeit sind nicht sinnvoll.

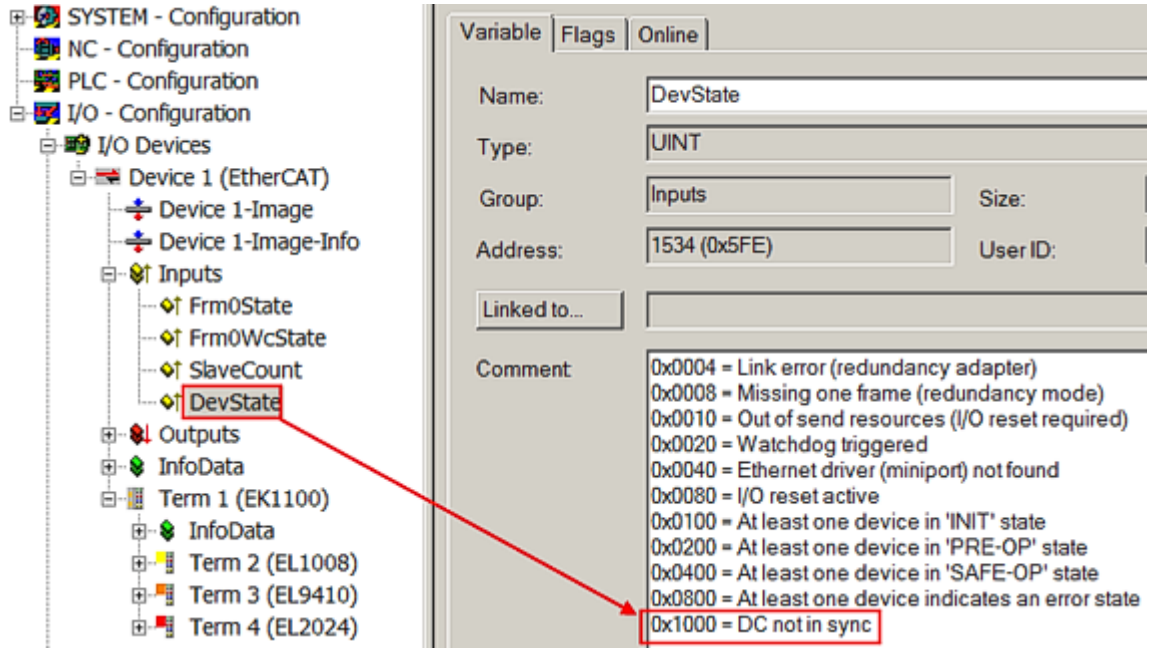


Abb. 215: DevState mit Anzeige SyncWindow Monitoring

Auswahl der Reference Clock

Eine manuelle Auswahl der Reference Clock dieses EtherCAT Systems ist mit Bedacht vorzunehmen! Im Allgemeinen wählt TwinCAT den richtigen, nämlich den ersten DC-unterstützenden Slave aus.

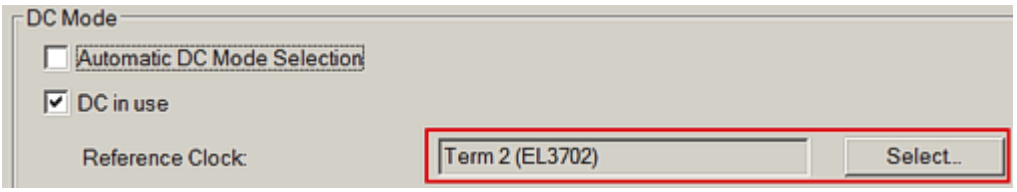


Abb. 216: Manuelle Auswahl ReferenceClock

Wird ein anderer Slave manuell ausgewählt

- muss dieser entweder DC ausdrücklich unterstützen (DC-Reiter ist im Slave angezeigt):

Abb. 217: Distributed Clocks Reiter (DC)

- oder manuell mit

Abb. 218: Checkbox zur manuellen Auswahl

in den erweiterten Slave-Eistellungen markiert werden

Potential Reference clock

Wird ein Slave als "potential Reference Clock" markiert, der dies hardwaretechnisch nicht unterstützt, fehlt dem EtherCAT System nach dem Hochlauf die Referenzuhr. Die Slaves werden nicht in den OP-State wechseln und PREOP_ERR melden. TwinCAT prüft ab Version 2.11R2 build 2032 bei jedem Hochlauf, ob die ausgewählte ReferenceClock diese Funktion auch unterstützt. Es erscheint die Logger-Meldung "slave xx is reference clock device, but does not support dc!" (s. Abb. „Event-Logger Meldung: Fehlende Unterstützung Referenzuhr“)

Im Logger Fenster gibt TwinCAT 2.11R2 außerdem ab build 2028 eine Meldung "DC not synchronized" aus, wenn ein Slave nicht vom PREOP in den OP gesetzt werden kann, weil er nicht synchronisiert wird. (s. Abb. „Event-Logger Meldung: Keine Synchronisation“)

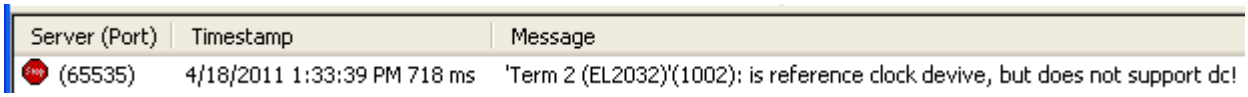


Abb. 219: Event-Logger Meldung: Fehlende Unterstützung Referenzuhr

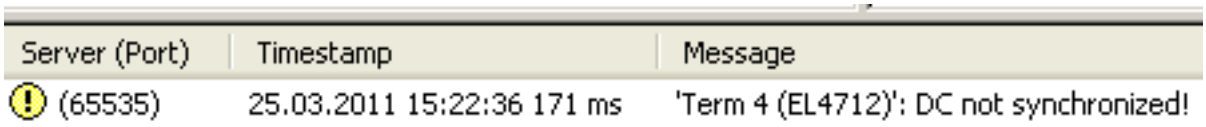


Abb. 220: Event-Logger Meldung: Keine Synchronisation

Standardeinstellungen EtherCAT Slave

In den erweiterten Einstellungen eines EtherCAT Slave sind die Distributed Clocks Einstellungen zu finden.

Wenn der Slave herstellerseitig für DC-Betrieb vorgesehen ist, bringt er einen derartigen "Operation Mode" mit, s. Abb. *Default Slave Einstellung - Slave ohne und mit DC-Fähigkeit* unten. Die Auswahl dieses Operation Modes aktiviert die Integration des Slaves in den masterseitigen Synchronisierungsmechanismus.

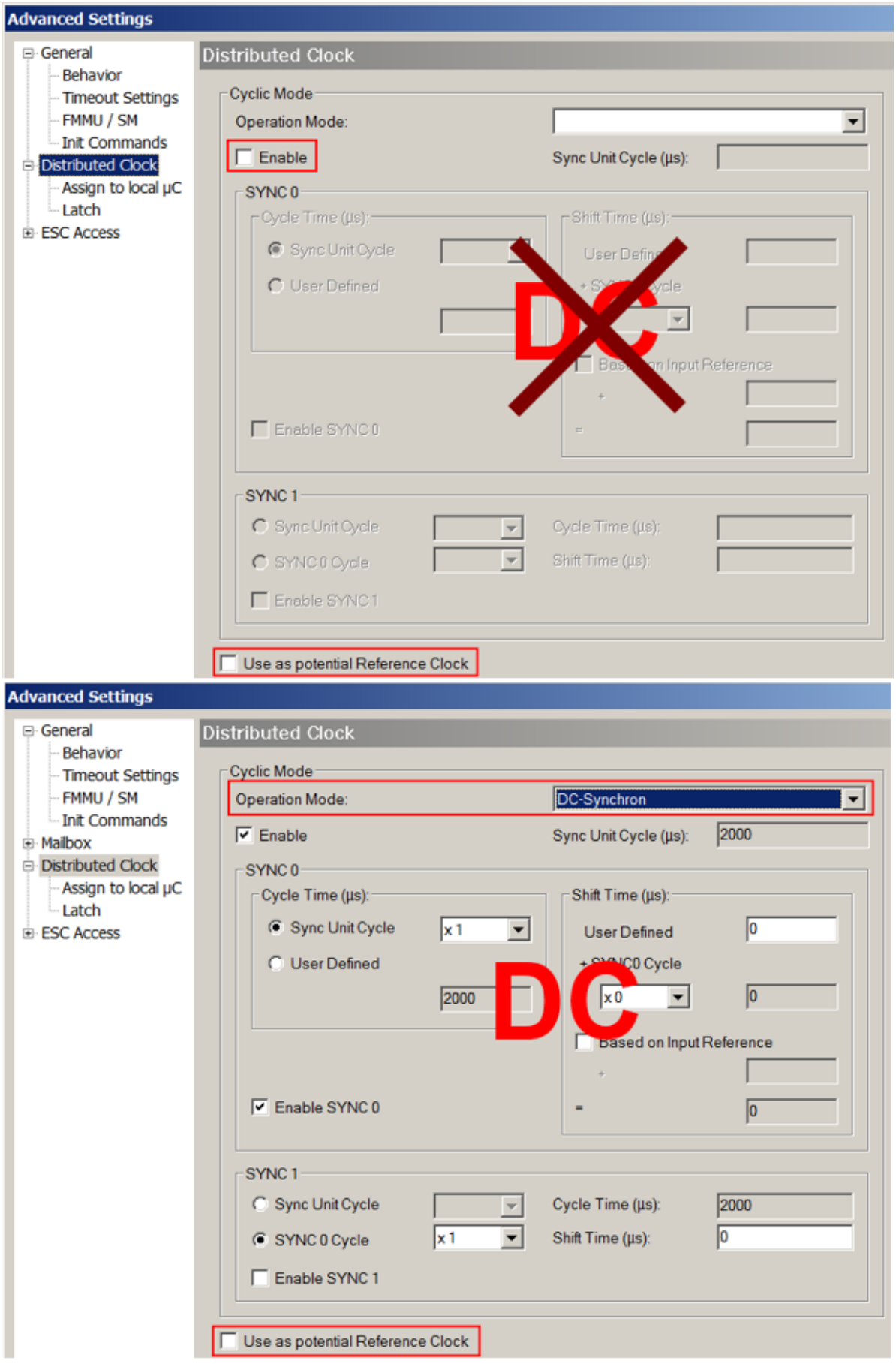


Abb. 221: Default Slave Einstellung - Slave ohne und mit DC-Fähigkeit

Aktivierung Distributed Clock

Werden die Einstellungen "Enable" und "Use as potential Reference Clock" aktiviert, ohne dass das Gerät dies unterstützt, kann es zu Fehlverhalten des Systems kommen.
Der Gerätehersteller hat diese Funktion freizugeben.

Einstellungen Slave Shift Time

Wie in den Master-Einstellungen für alle Slaves, können in den einzelnen DC-Slaves noch gesondert die SYNC-Zeiten verschoben werden.
Es sind die allgemeinen Hinweise [▶ 251] zu beachten.

Prüfung DC-Unterstützung

EtherCAT Slaves mit DC-Funktion werden

- im Übergang SAFEOP-->PREOP mit der DC-Konfiguration geladen.
- im Übergang SAFEOP-->OP synchronisiert.

Kommt es in den entsprechenden Phasen zu Problemen durch falsche Konfiguration, werden die Ziel-States nicht erreicht und im Loggerfenster des System Manager sind entsprechende Hinweise zu finden (z. B. "DC invalid sync cfg")

'PREOP to SAFEOP' failed! Error: 'check device state for SAFEOP'. AL Status '0x0012' read and '0x0004' expected. AL Status Code '0x0030 - DC invalid sync cfg'

Abb. 222: Loggerfenster Hinweis auf falsche DC-Konfiguration des Slaves

Soll überprüft werden, ob ein Gerät und in welchem Umfang es Distributed Clocks unterstützt, können die lokalen Uhrzeitregister online angezeigt werden. In der Online-Anzeige wird durch Rechtsklick auf den freien Bereich der Properties-Dialog aufgerufen.

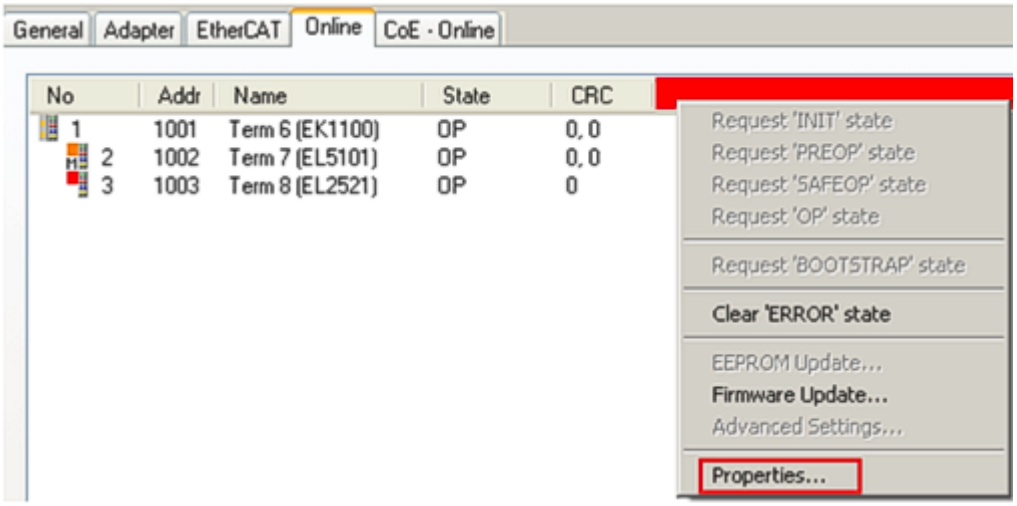


Abb. 223: Einblendung zusätzlicher Registerwerte aus den Slaves

Es sind in diesem Fall die 4 Register 0x0910 bis 0x0916 auszuwählen. In diesen 2-Byte-Registern läuft die lokale DC-Uhr. Sie können (wie einige andere) der Slave-Register vom System Manager online zyklisch ausgelesen werden.

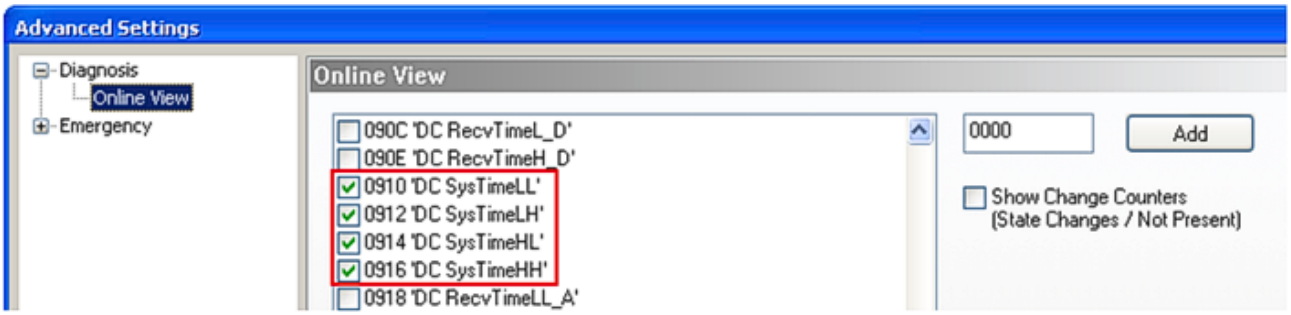


Abb. 224: Auswahl DC-Register

Die Registerwerte werden eingeblendet und vom System Manager automatisch aktualisiert.

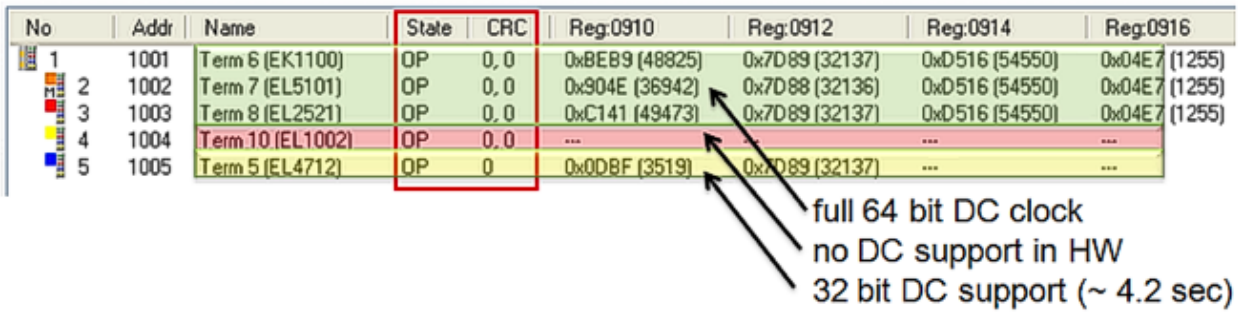


Abb. 225: Anzeige DC-Registerwerte

In Abb. *Anzeige DC-Registerwerte* sind zu sehen

- Slaves mit 64 Bit DC Unterstützung (hier grün hinterlegt)
- Slaves ohne DC-Unterstützung (hier rot hinterlegt)
- Slaves mit 32 Bit DC-Unterstützung (hier gelb hinterlegt)
  32 Bit decken ca. 4.2 Sekunden ab. Dies ist für die Synchronisierung des DC-Systems ausreichend. Hinweise zum Rechnen mit 32/64 Bit Zeiten in der PLC siehe im entsprechenden Kapitel [▶ 260].

Aktualität Online-Anzeige

Die angezeigten Werte werden vom System Manager ohne Anspruch auf Echtzeit und Durchgängigkeit ausgelesen und angezeigt. Sie haben informativen Charakter und können einen ersten Überblick geben. Der Auslesevorgang benötigt Kapazität in der azyklischen EtherCAT Kommunikation - dies kann zu Lasten anderer Anwendungen gehen.
Es können nicht alle Slave-ESC-Registerwerte vom System Manager/EtherCAT Master ausgelesen werden.

3.7.2 EtherCAT Distributed Clocks - Kopplung von EtherCAT Systemen

Ein EtherCAT System mit Distributed-Clocks-Nutzung weist folgende Eigenschaften auf:

- es wird an der Steuerung/IPC *ein* Ethernet Port benutzt
  Hinweis: bei Kabelredundanz werden *zwei* Ethernet Ports benutzt; die Kombination von Distributed Clocks (DC) und Kabelredundanz ist jedoch nur mithilfe des Gerätes CU2508 möglich, s. entsprechende Gerätedokumentation.
- bis zu 65535 Slaves werden an diesem Port von einem Realtime-EtherCAT Master zyklisch/azyklisch mit Prozessdaten versorgt.
- der erste DC-fähige EtherCAT Slave im System stellt die ReferenceClock (Abb. *Topologie Distributed Clocks System*: "M") dar, nach der alle nachfolgenden Slaves synchronisiert werden (Abb. *Topologie Distributed Clocks System*: "S")
- um die Echtzeit synchron zu dem DC-Umfeld zu halten, wird die interne TwinCAT Echtzeit Uhr ebenfalls dieser Feld-ReferenceClock nachgeführt.

Es kann also nur *eine* Referenzuhr an einer TwinCAT Steuerung geben, die die TwinCAT Echtzeit synchronisiert. Alle anderen EtherCAT Systeme müssen sich bzw. ihre lokale ReferenceClock nachregeln. Trotzdem behalten alle EtherCAT Systeme ihre lokale ReferenceClock im jeweils ersten DC-EtherCAT-Slave. Darauf ist in den folgenden Einstellungen Rücksicht zu nehmen.

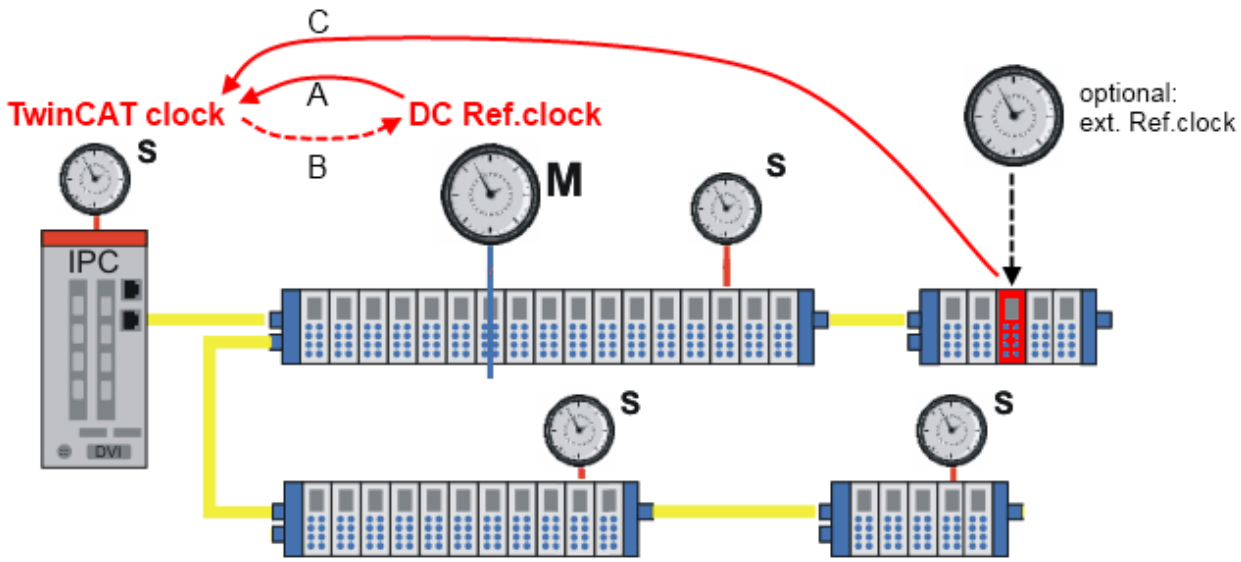


Abb. 226: Topologie Distributed Clocks System

In den erweiterten Einstellungen des EtherCAT Masters kann die Synchronisierungsrichtung eingestellt werden.

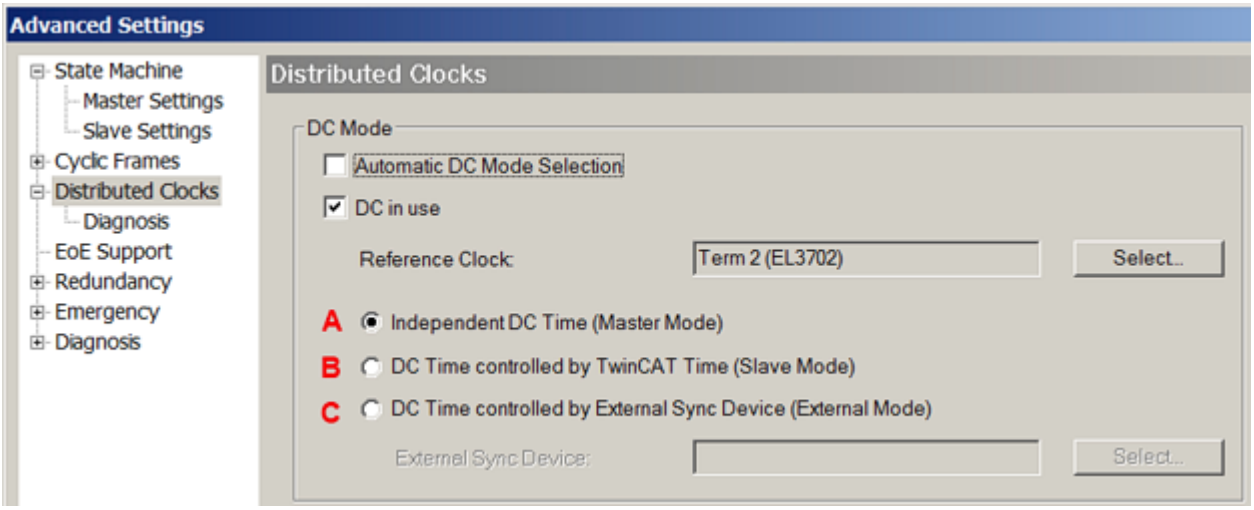


Abb. 227: Synchronisierungsrichtung

Wird mehr als ein EtherCAT System auf einer TwinCAT Steuerung verwendet, ist also mehr als ein "EtherCAT Device" in der I/O-Konfiguration enthalten und benutzen einige oder alle davon Distributed-Clocks-Funktionen, ist wie folgt vorzugehen:

- ein System wird in den DC-Einstellungen auf "Independent mode" gesetzt.
  In diesem System sitzt eine ReferenceClock in einem Slave und regelt alle anderen Slaves in diesem System nach. Ebenso wird die TwinCAT Echtzeit dieser Uhr frequenzsynchron nachgeführt.
- alle anderen DC-Systeme sind auf "Slave-Mode" zu stellen.
  In diesen Systemen sitzen ebenfalls die lokalen ReferenceClocks zur Synchronisierung der nachfolgenden Teilnehmer. Allerdings wird diese ReferenceClock während des EtherCAT Starts und später auch fortlaufend der TwinCAT Zeit nachgeführt und im Folgenden als "nachgeführte ReferenceClock" bezeichnet.

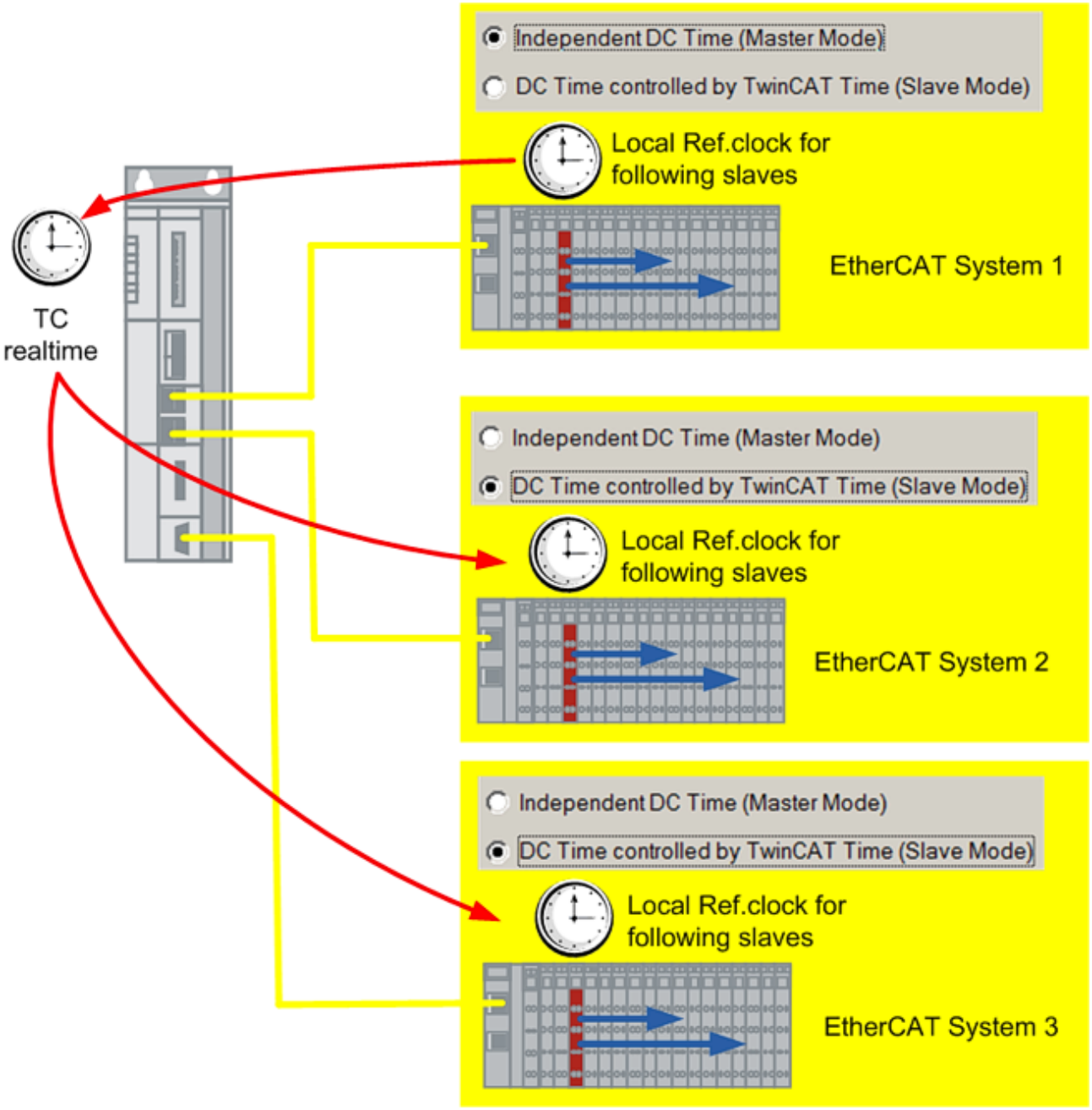


Abb. 228: DC-Kopplung EtherCAT-Systemen

Hinweise

- Diese DC-Kopplung ist erst ab TwinCAT 2.11 möglich

- Als "nachgeführte ReferenceClock" sind ausschließlich EtherCAT Slaves ohne Eigenintelligenz/ Firmware zu verwenden. Solche Slaves sind dadurch erkennbar, dass sie keine CoE/SoE oder keine Prozessdaten besitzen.
Einfache digitale Ein/Ausgangsklemmen (EL1202-0100, EL2202-0100) oder Koppler (EK1100) kommen hier in Frage.
ACHTUNG! Es dürfen nur Geräte als ReferenceClock ausgewählt werden, die dies auch unterstützen. Es sind die Hinweise im Kapitel "Standardeinstellung DC [▶ 154]" zu beachten.

Wird dies nicht beachtet, kann es zu folgenden Effekten kommen: DC-Teilnehmer in den unterlagerten Systemen gehen nicht in OP-State bzw. bleiben in PREOP; Teilnehmer fallen wegen „SyncLost“ aus der Synchronisierung und dem OP-State.

Nachführungsgenauigkeit

Wie in Abb. *DC-Kopplung EtherCAT-Systemen* ersichtlich muss eine Kette von Synchronisierungen bis zur "nachgeführten ReferenceClock" über die TwinCAT Steuerung durchgeführt werden. Dadurch wird nur eine reduzierte Regelungsgenauigkeit der EtherCAT-Systeme untereinander erreicht. Es ist in der Anwendung zu prüfen, ob dieser Ansatz langfristig den applikativen Ansprüchen an Regelungsgenauigkeit und Stabilität genügt.

Für höchste Synchronisierungsgenauigkeit entwickelt ist der Port-Multiplier CU2508. Er unterstützt bis zu 8 angeschlossene EtherCAT-Systeme bzw. 4 mit gleichzeitiger Kabelredundanz. Er ermöglicht auch die Kombination von Distributed-Clocks-Funktion und Kabelredundanz. In der Konfiguration ist der CU2508 transparent, es sind also weiterhin eigenständige EtherCAT Devices in der Konfiguration sichtbar.

Weitere Information siehe die Dokumentation CU2508.

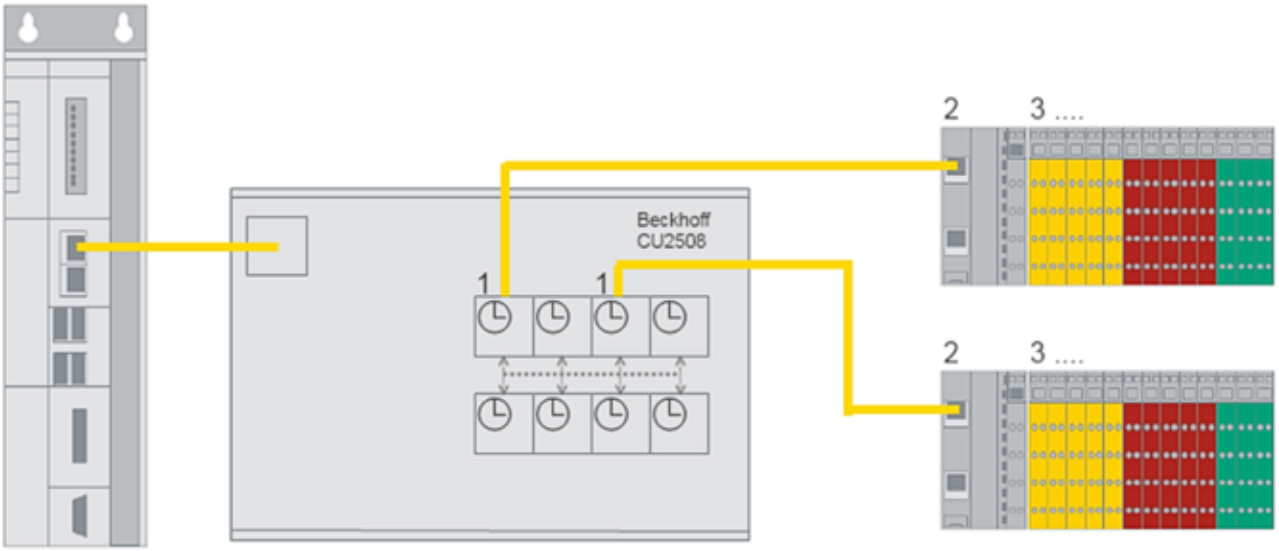


Abb. 229: EtherCAT Topologie mit CU2508 und 2 EtherCAT Systemen

4 EtherCAT Diagnose

4.1 Allgemeine Inbetriebnahmehinweise des EtherCAT Slaves

In dieser Übersicht werden in Kurzform einige Aspekte des EtherCAT Slave Betriebs unter TwinCAT behandelt. Ausführliche Informationen dazu sind entsprechenden Fachkapiteln z.B. in der EtherCAT-Systemdokumentation zu entnehmen.

Diagnose in Echtzeit: WorkingCounter, EtherCAT State und Status

Im Allgemeinen bietet ein EtherCAT Slave mehrere Diagnoseinformationen zur Verarbeitung in der ansteuernden Task an.

Diese Diagnoseinformationen erfassen unterschiedliche Kommunikationsebenen und damit Quellorte und werden deshalb auch unterschiedlich aktualisiert.

Eine Applikation, die auf die Korrektheit und Aktualität von IO-Daten aus einem Feldbus angewiesen ist, muss die entsprechend ihr unterlagerten Ebenen diagnostisch erfassen.

EtherCAT und der TwinCAT System Manager bieten entsprechend umfassende Diagnoseelemente an. Die Diagnoseelemente, die im laufenden Betrieb (nicht zur Inbetriebnahme) für eine zyklusaktuelle Diagnose aus der steuernden Task hilfreich sind, werden im Folgenden erläutert.

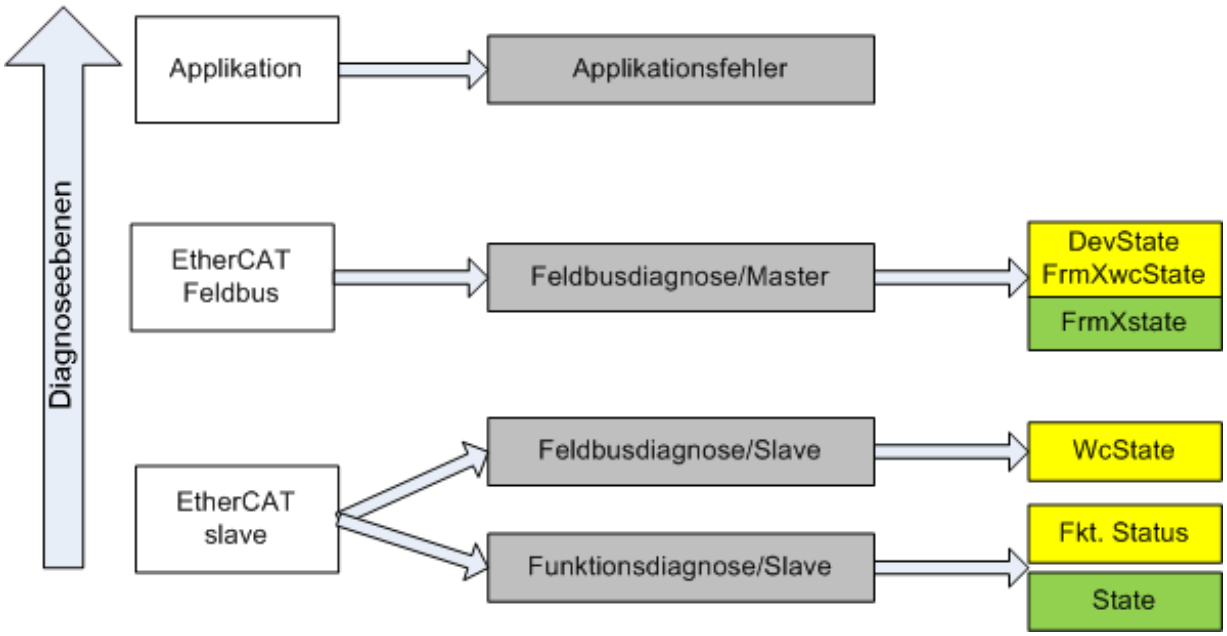


Abb. 230: Auswahl an Diagnoseinformationen eines EtherCAT Slave

Im Allgemeinen verfügt ein EtherCAT Slave über

- slave-typische Kommunikationsdiagnose (Diagnose der erfolgreichen Teilnahme am Prozessdatenaustausch und richtige Betriebsart)
  Diese Diagnose ist für alle Slaves gleich.

  als auch über
- kanal-typische Funktionsdiagnose (geräteabhängig)
  Siehe entsprechende Gerätedokumentation

Die Farbgebung in Abb. *Auswahl an Diagnoseinformationen eines EtherCAT Slave* entspricht auch den Variablenfarben im System Manager, siehe Abb. *Grundlegende EtherCAT Slave Diagnose in der PLC*.

Farbe	Bedeutung
gelb	Eingangsvariablen vom Slave zum EtherCAT Master, die in jedem Zyklus aktualisiert werden
rot	Ausgangsvariablen vom Slave zum EtherCAT Master, die in jedem Zyklus aktualisiert werden
grün	Informationsvariabeln des EtherCAT Masters, die azyklisch aktualisiert werden d. h. in einem Zyklus eventuell nicht den letztmöglichen Stand abbilden. Deshalb ist ein Auslesen solcher Variablen über ADS sinnvoll.

In Abb. *Grundlegende EtherCAT Slave Diagnose in der PLC* ist eine Beispielimplementation einer grundlegenden EtherCAT Slave Diagnose zu sehen. Dabei wird eine Beckhoff EL3102 (2 kanalige analoge Eingangsklemme) verwendet, da sie sowohl über slave-typische Kommunikationsdiagnose als auch über kanal-spezifische Funktionsdiagnose verfügt. In der PLC sind Strukturen als Eingangsvariablen angelegt, die jeweils dem Prozessabbild entsprechen.

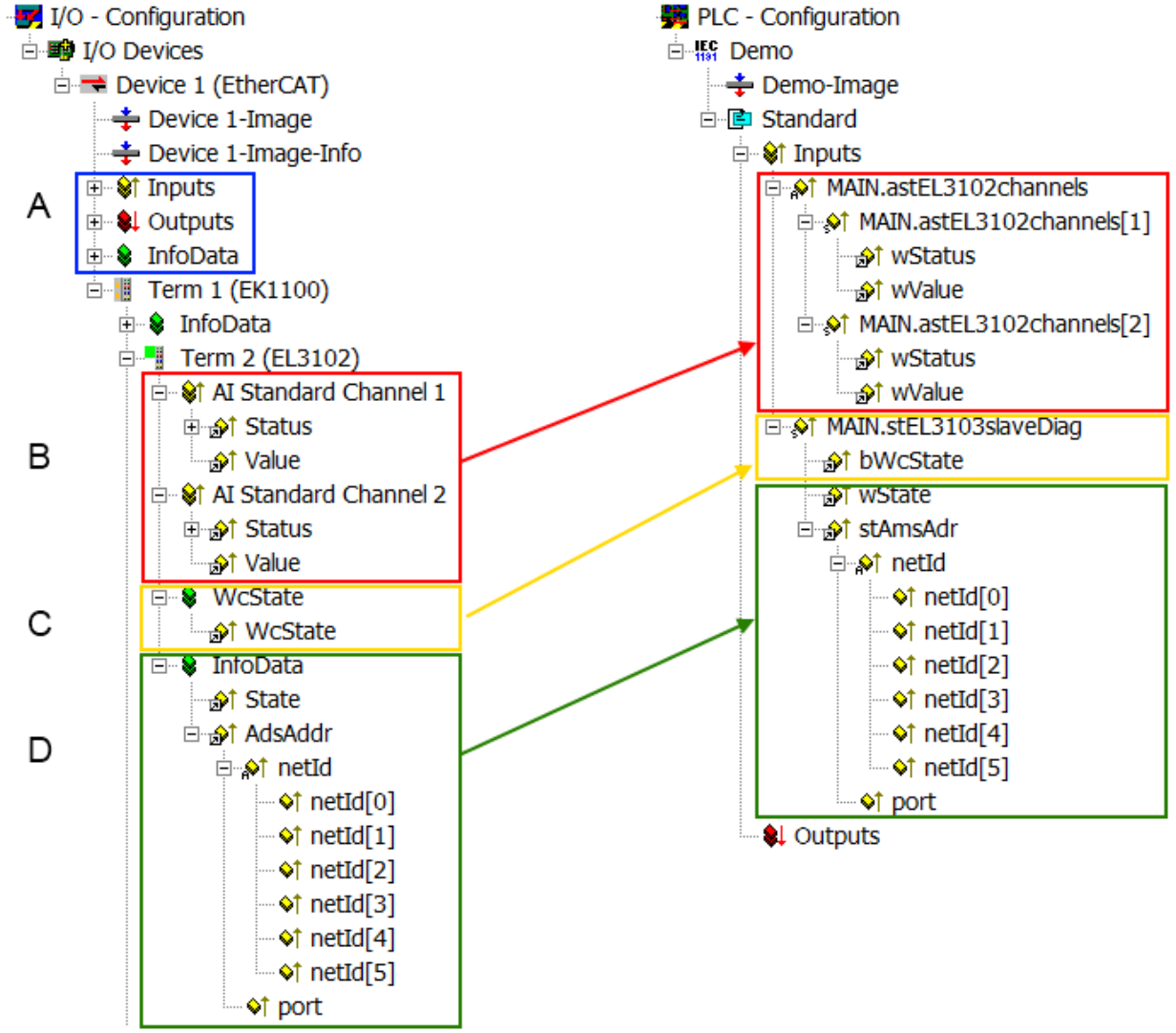


Abb. 231: Grundlegende EtherCAT Slave Diagnose in der PLC

Dabei werden folgende Aspekte abgedeckt:

Kennzeichen	Funktion	Ausprägung	Anwendung/Auswertung
A	Diagnoseinformationen des EtherCAT Master zyklisch aktualisiert (gelb) oder azyklisch bereitgestellt (grün).		Zumindest der DevState ist in der PLC zyklusaktuell auszuwerten. Die Diagnoseinformationen des EtherCAT Master bieten noch weitaus mehr Möglichkeiten, die in der EtherCAT-Systemdokumentation behandelt werden. Einige Stichworte: • CoE im Master zur Kommunikation mit/über die Slaves • Funktionen aus *TcEtherCAT.lib* • OnlineScan durchführen
B	Im gewählten Beispiel (EL3102) umfasst die EL3102 zwei analoge Eingangskanäle, die einen eigenen Funktionsstatus zyklusaktuell übermitteln.	Status • die Bitdeutungen sind der Gerätedokumentation zu entnehmen • andere Geräte können mehr oder keine slavetypischen Angaben liefern	Damit sich die übergeordnete PLC-Task (oder entsprechende Steueranwendungen) auf korrekte Daten verlassen kann, muss dort der Funktionsstatus ausgewertet werden. Deshalb werden solche Informationen zyklusaktuell mit den Prozessdaten bereitgestellt.
C	Für jeden EtherCAT Slave mit zyklischen Prozessdaten zeigt der Master durch einen so genannten Working-Counter an, ob der Slave erfolgreich und störungsfrei am zyklischen Prozessdatenverkehr teilnimmt. Diese elementar wichtige Information wird deshalb im System Manager zyklusaktuell 1. am EtherCAT Slave als auch inhaltsidentisch 2. als Sammelvariable am EtherCAT Master (siehe Punkt A) zur Verlinkung bereitgestellt.	WcState (Working Counter) 0: gültige Echtzeitkommunikation im letzten Zyklus 1: ungültige Echtzeitkommunikation ggf. Auswirkung auf die Prozessdaten anderer Slaves, die in der gleichen SyncUnit liegen	Damit sich die übergeordnete PLC-Task (oder entsprechende Steueranwendungen) auf korrekte Daten verlassen kann, muss dort der Kommunikationsstatus des EtherCAT Slaves ausgewertet werden. Deshalb werden solche Informationen zyklusaktuell mit den Prozessdaten bereitgestellt.
D	Diagnoseinformationen des EtherCAT Masters, die zwar am Slave zur Verlinkung dargestellt werden, aber tatsächlich vom Master für den jeweiligen Slave ermittelt und dort dargestellt werden. Diese Informationen haben keinen Echtzeit-Charakter weil sie • nur selten/nie verändert werden, außer beim Systemstart • selbst auf azyklischem Weg ermittelt werden (z.B. EtherCAT Status)	State aktueller Status (INIT..OP) des Slaves. Im normalen Betriebszustand muss der Slave im OP (=8) sein. *AdsAddr* Die ADS-Adresse ist nützlich, um aus der PLC/Task über ADS mit dem EtherCAT Slave zu kommunizieren, z.B. zum Lesen/Schreiben auf das CoE. Die AMS-NetID eines Slaves entspricht der AMS-NetID des EtherCAT Masters, über den *port* (= EtherCAT Adresse) ist der einzelne Slave ansprechbar.	Informationsvariabeln des EtherCAT Masters, die azyklisch aktualisiert werden, d.h. in einem Zyklus eventuell nicht den letztmöglichen Stand abbilden. Deshalb ist ein Auslesen solcher Variablen über ADS möglich.

HINWEIS

Diagnoseinformationen

Es wird dringend empfohlen, die angebotenen Diagnoseinformationen auszuwerten um in der Applikation entsprechend reagieren zu können.

CoE-Parameterverzeichnis

Das CoE-Parameterverzeichnis (CanOpen-over-EtherCAT) dient der Verwaltung von Einstellwerten des jeweiligen Slaves. Bei der Inbetriebnahme eines komplexeren EtherCAT Slaves sind unter Umständen hier Veränderungen vorzunehmen. Zugänglich ist es über den TwinCAT System Manager, s. Abb. *EL3102, CoE-Verzeichnis*:

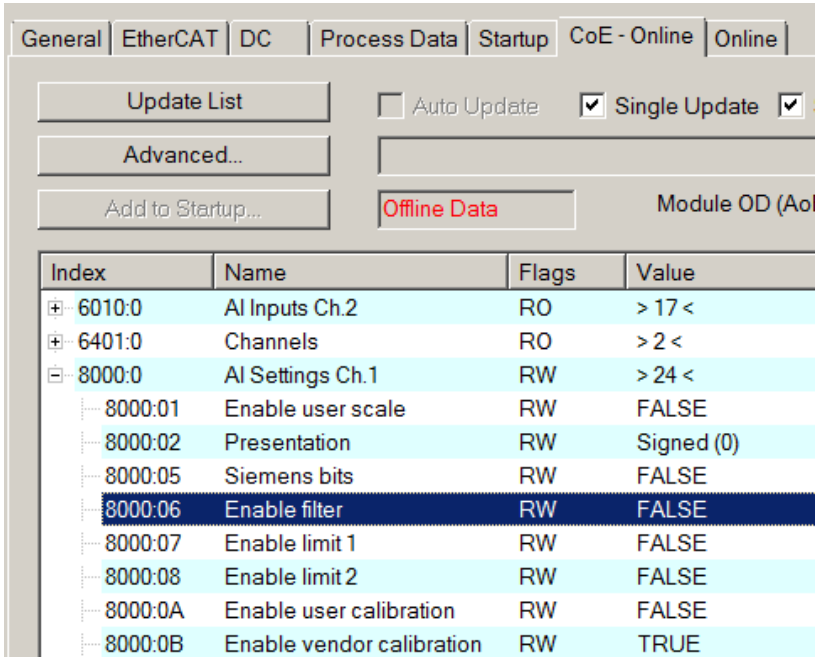


Abb. 232: EL3102, CoE-Verzeichnis

EtherCAT-Systemdokumentation

Es ist die ausführliche Beschreibung in der EtherCAT-Systemdokumentation (EtherCAT Grundlagen --> CoE Interface) zu beachten!

Einige Hinweise daraus in Kürze:

- Es ist geräteabhängig, ob Veränderungen im Online-Verzeichnis slave-lokal gespeichert werden. EL-Klemmen (außer den EL66xx) verfügen über diese Speichermöglichkeit.
- Es ist vom Anwender die StartUp-Liste mit den Änderungen zu pflegen.

Inbetriebnahmehilfe im TwinCAT System Manager

In einem fortschreitenden Prozess werden für EL/EP-EtherCAT Geräte Inbetriebnahmeoberflächen eingeführt. Diese sind in TwinCAT System Managern ab TwinCAT 2.11R2 verfügbar. Sie werden über entsprechend erweiterte ESI-Konfigurationsdateien in den System Manager integriert.

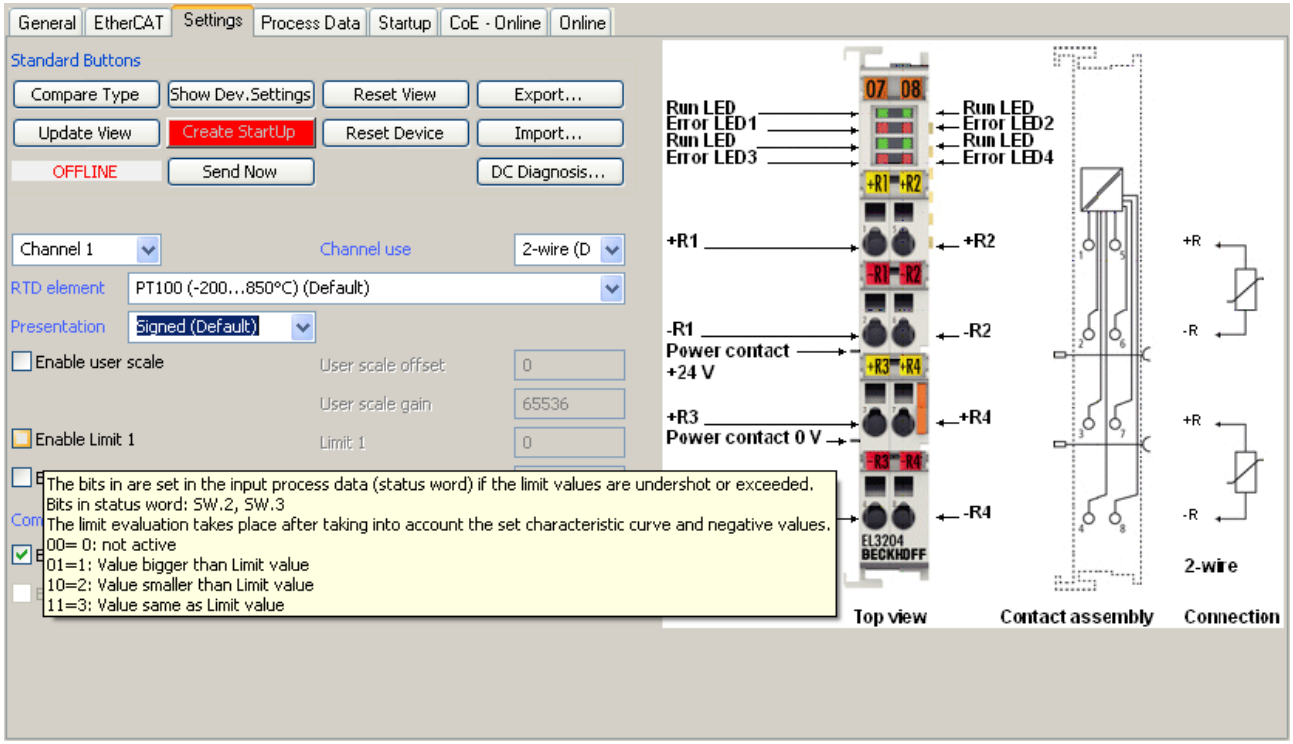


Abb. 233: Beispiel Inbetriebnahmehilfe für eine EL3204

Diese Inbetriebnahme verwaltet zugleich

- CoE-Parameterverzeichnis
- DC/FreeRun-Modus
- die verfügbaren Prozessdatensätze (PDO)

Die dafür bisher nötigen Karteireiter „Process Data“, „DC“, „Startup“ und „CoE-Online“ werden zwar noch angezeigt, es wird aber empfohlen die automatisch generierten Einstellungen durch die Inbetriebnahmehilfe nicht zu verändern, wenn diese verwendet wird.

Das Inbetriebnahme-Tool deckt nicht alle möglichen Einsatzfälle eines EL/EP-Gerätes ab. Sind die Einstellmöglichkeiten nicht ausreichend, können vom Anwender wie bisher DC-, PDO- und CoE-Einstellungen manuell vorgenommen werden.

EtherCAT State: automatisches Default-Verhalten des TwinCAT System Managers und manuelle Ansteuerung

Ein EtherCAT Slave hat für den ordnungsgemäßen Betrieb nach der Versorgung mit Betriebsspannung die Stati

- INIT
- PREOP
- SAFEOP
- OP

zu durchlaufen. Der EtherCAT Master ordnet diese Zustände an in Abhängigkeit der Initialisierungsroutinen, die zur Inbetriebnahme des Gerätes durch die ES/XML und Anwendereinstellungen (Distributed Clocks (DC), PDO, CoE) definiert sind. Siehe dazu auch Kapitel "Grundlagen der Kommunikation, EtherCAT State Machine [▶ 19]. Der Hochlauf kann je nach Konfigurationsaufwand und Gesamtkonfiguration bis zu einigen Sekunden dauern.

Auch der EtherCAT Master selbst muss beim Start diese Routinen durchlaufen, bis er in jedem Fall den Zielzustand OP erreicht.

Der vom Anwender beabsichtigte, von TwinCAT beim Start automatisch herbeigeführte Ziel-State kann im System Manager eingestellt werden. Sobald TwinCAT in RUN versetzt wird, wird dann der TwinCAT EtherCAT Master die Zielzustände anfahren.

Standardeinstellung

Standardmäßig ist in den erweiterten Einstellungen des EtherCAT Masters gesetzt:

- EtherCAT Master: OP
- Slaves: OP
  Diese Einstellung gilt für alle Slaves zugleich.

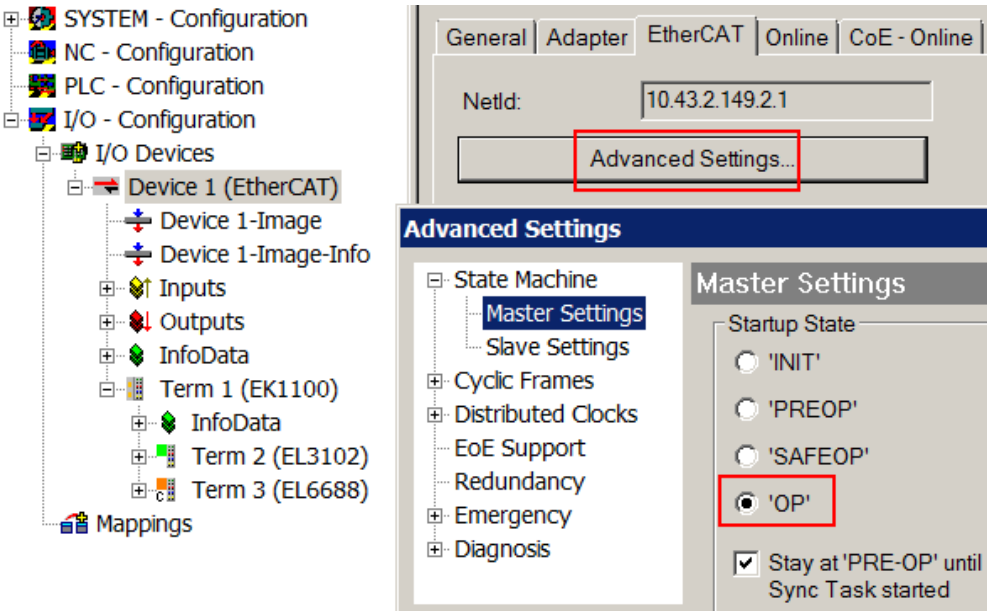


Abb. 234: Default Verhalten System Manager

Zusätzlich kann im Dialog „Erweiterte Einstellung“ beim jeweiligen Slave der Zielzustand eingestellt werden, auch dieser ist standardmäßig OP.

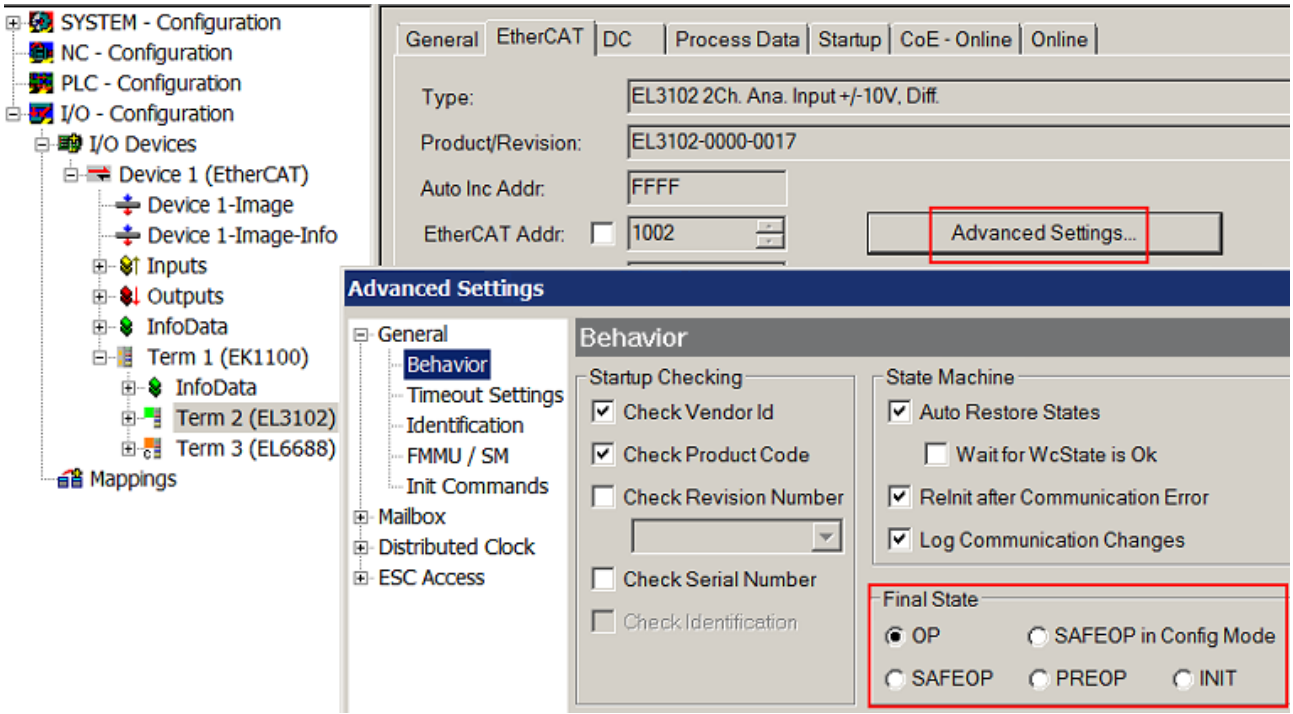


Abb. 235: Default Zielzustand im Slave

Manuelle Führung

Aus bestimmten Gründen kann es angebracht sein, aus der Anwendung/Task/PLc die States kontrolliert zu fahren, z. B.

- aus Diagnosegründen
- kontrolliertes Wiederanfahren von Achsen
- ein zeitlich verändertes Startverhalten ist gewünscht

Dann ist es in der PLC-Anwendung sinnvoll, die PLC-Funktionsblöcke aus der standardmäßig vorhandenen *TcEtherCAT.lib* zu nutzen und z. B. mit *FB_EcSetMasterState* die States kontrolliert anzufahren.

Die Einstellungen im EtherCAT Master sind dann sinnvollerweise für Master und Slave auf INIT zu setzen.

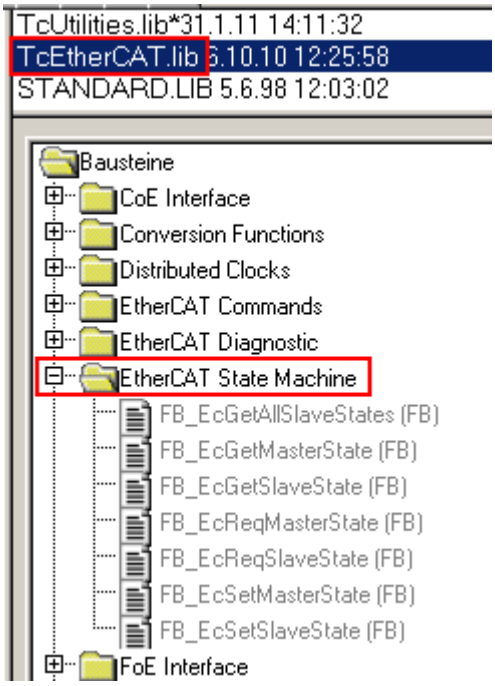
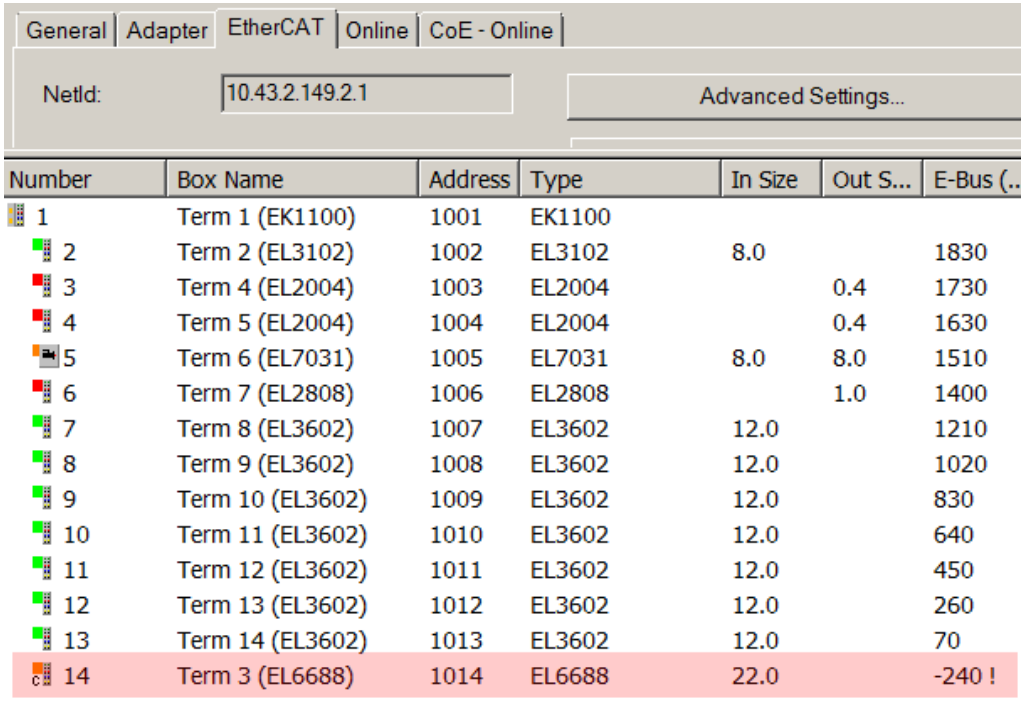


Abb. 236: PLC-Bausteine

Hinweis E-Bus-Strom

EL/ES-Klemmen werden im Klemmenstrang auf der Hutschiene an einen Koppler gesetzt. Ein Buskoppler kann die an ihm angefügten EL-Klemmen mit der E-Bus-Systemspannung von 5 V versorgen, i.d.R. ist ein Koppler dabei bis zu 2 A belastbar. Zu jeder EL-Klemme ist die Information, wie viel Strom sie aus der E-Bus-Versorgung benötigt, online und im Katalog verfügbar. Benötigen die angefügten Klemmen mehr Strom als der Koppler liefern kann, sind an entsprechenden Positionen im Klemmenstrang Einspeiseklemmen (z. B. EL9410) zu setzen.

Im TwinCAT System Manager wird der vorberechnete theoretische maximale E-Bus-Strom als Spaltenwert angezeigt. Eine Unterschreitung wird durch negativen Summenbetrag und Ausrufezeichen markiert, vor einer solchen Stelle ist eine Einspeiseklemme zu setzen.



Number	Box Name	Address	Type	In Size	Out S...	E-Bus (..
1	Term 1 (EK1100)	1001	EK1100			
2	Term 2 (EL3102)	1002	EL3102	8.0		1830
3	Term 4 (EL2004)	1003	EL2004		0.4	1730
4	Term 5 (EL2004)	1004	EL2004		0.4	1630
5	Term 6 (EL7031)	1005	EL7031	8.0	8.0	1510
6	Term 7 (EL2808)	1006	EL2808		1.0	1400
7	Term 8 (EL3602)	1007	EL3602	12.0		1210
8	Term 9 (EL3602)	1008	EL3602	12.0		1020
9	Term 10 (EL3602)	1009	EL3602	12.0		830
10	Term 11 (EL3602)	1010	EL3602	12.0		640
11	Term 12 (EL3602)	1011	EL3602	12.0		450
12	Term 13 (EL3602)	1012	EL3602	12.0		260
13	Term 14 (EL3602)	1013	EL3602	12.0		70
14	Term 3 (EL6688)	1014	EL6688	22.0		-240 !

Abb. 237: Unzulässige Überschreitung E-Bus Strom

Ab TwinCAT 2.11 wird bei der Aktivierung einer solchen Konfiguration eine Warnmeldung „E-Bus Power of Terminal...“ im Logger-Fenster ausgegeben:

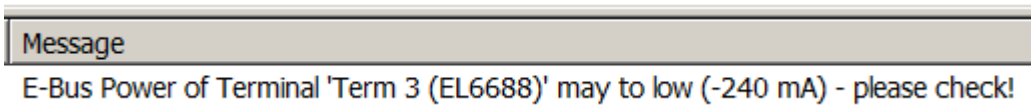


Abb. 238: Warnmeldung E-Bus-Überschreitung

HINWEIS

Achtung! Fehlfunktion möglich!

Die E-Bus-Versorgung aller EtherCAT-Klemmen eines Klemmenblocks muss aus demselben Massepotential erfolgen!

4.2 EtherCAT AL Status Codes

4.2.1 Error Code 0x0000

Meaning

No error

Description

No error

Current State (or state change)

Any

Resulting state

Current state

Solution

n/a

4.2.2 Error Code 0x0001

Meaning

Unspecified error

Description

No error code is defined for occurred error

Current State (or state change)

Any

Resulting state

Any + E

Solution

Read user manual or contact device manufacturer

4.2.3 Error Code 0x0002

Meaning

No Memory

Description

Less hardware memory, slave needs more memory.

Example: For slave configuration, application configuration files are downloaded (possibly via FoE or large CoE objects). The size of those files exceeds the local memory

Current State (or state change)

Any

Resulting state

Any + E

Solution

Download smaller files or objects.

Check user manual.

4.2.4 Error Code 0x0004

Meaning

Invalid Revision

Description

Output/Input mapping is not valid for this hardware or software revision (0x1018:03)

Current State (or state change)

P→S

Resulting state

P+E

Solution

Change mapping or use different hardware

4.2.5 Error Code 0x0011

Meaning

Invalid requested state change

Description

The EtherCAT State Machine (ESM) defines which state changes are allowed. All other state changes are not allowed

Example: If the master requests the slave to go from OP (AL Control = 0x08) directly to BOOT (AL Control = 0x03).

Current State (or state change)

P→S, I→O, P→O, O→B, S→B, P→B

Resulting state

Current State + E

Solution

Go step-by-step from the original state to the desired state.

4.2.6 Error Code 0x0012

Meaning

Unknown requested state change

Description

The ESM defines the following states. They are coded with fixed values (only lower (=right) nibble):

BOOT: AL Control = 0x03

INIT: AL Control = 0x01

PREOP: AL Control = 0x02

SAFEOP: AL Control = 0x04

OP: AL Control = 0x08

The fifth bit of the AL Control (left nibble is 1) is the “Error Acknowledge Bit”. If the slave is in AL STATUS = 0x14, i.e. ERROR SAFEOP the master acknowledges this by setting the Acknowledge bit.

Example: If any other value for AL Control than those specified are sent.

Current State (or state change)

Any

Resulting state

Current State + E

Solution

Do only request the defined states

4.2.7 Error Code 0x0013

Meaning

Boot state not supported

Description

Device does not support BOOT state, but the master requests the slave to go to BOOT (AL Control = 0x03

Current State (or state change)

I→B

Resulting state

I + E

Solution

n/a

4.2.8 Error Code 0x0014

Meaning

No valid firmware

Description

This error code may be returned after a firmware download, if the downloaded file cannot be used by the application controller

Current State (or state change)

I→P

Resulting state

I + E

Solution

Download a firmware that can be supported by the hardware and bootloader. Check Product Code and Revision Number (CoE object 0x1018). If this cannot be read from the firmware any more you may see this in the network configuration (CoE object dictionary) or probably in the ESI file (element Profile: ObjectDictionary:Objects:Object).

4.2.9 Error Code 0x0015

Meaning

Invalid mailbox configuration

Description

Mailbox communication (= acyclic parameter exchange) is done via two memory areas on the EtherCAT Slave Controller (ESC) – the “Output Mailbox” (master -> slave) and the “Input Mailbox” (slave-> master). Those memory areas are protected by SyncManagers to prevent from simultaneous access from master and salve controller at the same time. SyncManagers are hardware entities on the ESC. They are configured via certain registers in the ESC register area (starting at 0x0800). The configuration includes start address, length, and direction (output or input). If those settings differ from those expected by the host controller of the slave this error is returned

Current State (or state change)

I→B

Resulting state

n/a

Solution

Replace previous network description of old slave with the one of the new slave

4.2.10 Error Code 0x0016

Meaning

Invalid mailbox configuration

Description

Example: The slave hardware was replaced while the network configuration remained unchanged. The new hardware expects different mailbox SyncManager settings

Current State (or state change)

I→S

Resulting state

I + E

Solution

Replace previous network description of old slave with the one of the new slave

4.2.11 Error Code 0x0017

Meaning

Invalid Sync Manager configuration

Description

Process data communication (cyclic communication) is done via extra memory areas on the ESC, separated for outputs and inputs. The process data length and the process data SyncManager length have to be the same. If this is not the case or the start address or direction does not match this error is returned.

Example: The process data configuration was changed of the slaves which also changed the length of the data. The change was not activated in the configuration so that the configuration tool would have recalculated the SyncManager settings.

Current State (or state change)

P→S, S→O

Resulting state

Current State + E

Solution

Issue a re-calculation of the EtherCAT configuration

4.2.12 Error Code 0x0018

Meaning

No valid inputs available

Description

The slave application cannot provide valid input values

Example: A certain hardware which needs to be connected to the slave was disconnected

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

n/a

4.2.13 Error Code 0x0019

Meaning

No valid outputs available

Description

The slave application cannot recieve valid output values.

Example: The slave has a RxPdoToggle output or an “Output Valid” information in its process data. The RxPdoToggle does not toggle or the OutputValid is not true. Therefore the slave has no process data which the application can use. If supported, check the RxPDO Toggle Failed Counter in object 0x1C3x.0E). Also, the Synchronization may have problems (see object 0x10F1:SI2 Sync Error Counter Limit) so that process data are received too late by the slave so that the local slave cycle misses the toggle event. Another reason can be that the PLC stopped working

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

The RxPdoToggle may need to be handled by the PLC program

The outputs valid may have to be set by the PLC program

PLC may have stopped, restart PLC

4.2.14 Error Code 0x001A

Meaning

Synchronization error

Description

If too many RxPDO Toggle error occur, i.e. the RxPDO Toggle Failed Counter increases the internal limit the slave returns to SAFEPERROR with 0x001A. Multiple synchronization errors. Device is not synchronized any more (used if the causes mirrored by the AL Status Codes 0x2C, 0x2D, 0x32, 0x33, 0x34 cannot be distinguished).

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

n/a

4.2.15 Error Code 0x001B

Meaning

Sync manager watchdog

Description

The slave did not receive process data within the specified watchdog time. Usually, the WD time is 100ms. The WD is re-started every time it receives new process data, usually when the Output SyncManager (SyncManager2) is written. For devices which have only inputs usually no WD is used. Increasing the WD is not a solution.

Reason: PLC stopped

Current State (or state change)

O, S

Resulting state

S + E

Solution

n/a

4.2.16 Error Code 0x001C

Meaning

Invalid Sync Manager Types

Description

n/a

Current State (or state change)

O, S, O, P→S

Resulting state

S + E

Solution

n/a

4.2.17 Error Code 0x001D

Meaning

Invalid Output Configuration

Description

SM configuration for output process data is invalid

Current State (or state change)

O, S, O, P→S

Resulting state

S + E

Solution

n/a

4.2.18 Error Code 0x001E

Meaning

Invalid Input Configuration

Description

SM configuration for input process data is invalid

Current State (or state change)

O, S, O, P→S

Resulting state

S + E

Solution

n/a

4.2.19 Error Code 0x001F

Meaning

Invalid Watchdog Configuration

Description

The Watchdog is configured in the ESC register 0x0400 and 0x0420. EtherCAT defines default watchdog settings (100ms) or they are defined in the ESI file. If the slave does not accept a change of the expected settings it returns this AL Status Code Example: A slave may not accept that the WD is deactivated.

Current State (or state change)

O, S, O, P→S

Resulting state

P + E

Solution

Use default WD settings

4.2.20 Error Code 0x0020

Meaning

Slave needs cold start

Description

Slave device require a power off - power on reset

Current State (or state change)

Any

Resulting state

Current State + E

Solution

n/a

4.2.21 Error Code 0x0021

Meaning

Slave needs INIT

Description

Slave application requests INIT state

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.2.22 Error Code 0x0022

Meaning

Slave needs PREOP

Description

Slave application requests PREOP state

Current State (or state change)

S, O

Resulting state

S + E, O + E

Solution

n/a

4.2.23 Error Code 0x0023

Meaning

Slave needs SAFEOP

Description

Slave application requests SAFEOP state

Current State (or state change)

O

Resulting state

O + E

Solution

n/a

4.2.24 Error Code 0x0024

Meaning

Invalid Input Mapping

Description

The process data are described by the configuration (PdoConfig) and PDO assignment (PdoAssign).

PdoConfig: list of actual variables (usually indexes 0x6nnn for inputs and 0x7nnn for outputs). Variables are also called PDO entries. There can be one or several variables with in one list (i.e. within one PDO). The Input PDOs have the index 0x1Amm. The Output PDOs have the index 0x16mm.

PdoAssign: The list of PDOs (object index 0x16nn, 0x1Amm) which are actually part of the process data and hence, are transferred cyclically, are listed in the PDO Assign Objects 0x1C12 (output PDOs) and 0x1C13 (input PDOs). All this can be seen in the SystemManager on the TAB “Process Data”. If the mapping which was set by the user on the Process Data tab and which was expected by the slave do not match this Status Code is returned.

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.2.25 Error Code 0x0025

Meaning

Invalid Output Mapping

Description

The process data are described by the configuration (PdoConfig) and PDO assignment (PdoAssign).

PdoConfig: list of actual variables (usually indexes 0x6nnn for inputs and 0x7nnn for outputs). Variables are also called PDO entries. There can be one or several variables with in one list (i.e. within one PDO). The Input PDOs have the index 0x1Amm. The Output PDOs have the index 0x16mm. Example: Slave does only support one or certain PDO combinations but a different setting was made by the user. For a bus coupler the connected terminals differ from the configured terminals in the SystemManager

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.2.26 Error Code 0x0026

Meaning

Inconsistent Settings

Description

General settings mismatch

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.2.27 Error Code 0x0027

Meaning

Freerun not supported

Description

n/a

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.2.28 Error Code 0x0028

Meaning

Synchronization not supported

Description

n/a

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.2.29 Error Code 0x0029

Meaning

Freerun needs 3 Buffer Mode

Description

FreeRun mode, SM has to run in 3-buffer mode

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.2.30 Error Code 0x002A

Meaning

Background Watchdog

Description

n/a

Current State (or state change)

S, O

Resulting state

P + E

Solution

n/a

4.2.31 Error Code 0x002B

Meaning

No Valid Inputs and Outputs

Description

n/a

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

n/a

4.2.32 Error Code 0x002C

Meaning

Fatal Sync Error

Description

The hardware interrupt signal (so called Sync signal) generated by the ESC is not generated any more. The master sets and activated the cycle time of the Sync signal during state transition from PREOP to SAFEOP. If a slave was disconnected and reconnected (also due to lost frames or CRC errors) the generation of the SyncSignal may be lost.

Current State (or state change)

O

Resulting state

S + E

Solution

Set master to INIT and back to OP so that the DCs are initialized again

4.2.33 Error Code 0x002D

Meaning

ana

Description

SyncSignal not received: In SAFEOP the slave waits for the first Sync0/Sync1 events before switching to OP, if these events were not received during the SAFEOP to OP-Timeout time the slave refuses the state transition to OP

Current State (or state change)

n/a

Resulting state

n/a

Solution

n/a

4.2.34 Error Code 0x0030

Meaning

Invalid DC SYNC Configuration

Description

Distributed Clock Configuration is invalid due to application requirements

Current State (or state change)

O, S→O, P→S

Resulting state

P + E, S + E

Solution

n/a

4.2.35 Error Code 0x0031

Meaning

Invalid DC Latch Configuration

Description

DC Latch configuration is invalid due to application requirements

Current State (or state change)

O, S→O, P→S

Resulting state

P + E, S + E

Solution

n/a

4.2.36 Error Code 0x0032

Meaning

PLL Error

Description

Master not synchronized, at least one DC event recieved

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

n/a

4.2.37 Error Code 0x0033

Meaning

DC Sync IO Error

Description

Multiple Synchronization Errors: At least one SycnSignal was received before. However, the PLL between slave and master is not synchronized any more. This may occur if the master application jitters too much

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

Use specific industrial pc, standard office PCs may have power saving options, graphic accelerateds and other system services which disturb the real-time of the master.

CPU power may be too small for the PLC/NC program.

Increase EtherCAT and PLC/NC cycle time.

Use SyncUnits for the slaves using DCs.

4.2.38 Error Code 0x0034

Meaning

DC Sync Timeout Error

Description

Multiple Synchronization Errors, too much SM events missed

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

n/a

4.2.39 Error Code 0x0035

Meaning

DC Invalid Sync Cycle Time

Description

n/a

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.2.40 Error Code 0x0036

Meaning

DC Sync0 Cycle Time

Description

DC Sync0 cycle time does not fit to the application requirements

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.2.41 Error Code 0x0037

Meaning

DC Sync1 Cycle Time

Description

DC Sync1 cycle time does not fit to the application requirements

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.2.42 Error Code 0x0041

Meaning

MBX_AOE

Description

n/a

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.2.43 Error Code 0x0042

Meaning

MBX_EOE

Description

n/a

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.2.44 Error Code 0x0043

Meaning

MBX_COE

Description

n/a

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.2.45 Error Code 0x0044

Meaning

MBX_FOE

Description

n/a

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.2.46 Error Code 0x0045

Meaning

MBX_SOE

Description

n/a

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.2.47 Error Code 0x004F

Meaning

MBX_VOE

Description

n/a

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.2.48 Error Code 0x0050

Meaning

EEPROM No Access

Description

EEPROM not assigned to PDI

Current State (or state change)

Any

Resulting state

Any + E

Solution

n/a

4.2.49 Error Code 0x0051

Meaning

EEPROM Error

Description

EEPROM access error

Current State (or state change)

Any

Resulting state

Any + E

Solution

n/a

4.2.50 Error Code 0x0060

Meaning

Slave Requested Locally

Description

n/a

Current State (or state change)

Any

Resulting state

I

Solution

n/a

4.2.51 Error Code 0x0061

Meaning

Device Identification Value updated

Description

n/a

Current State (or state change)

P

Resulting state

P + E

Solution

n/a

4.2.52 Error Code 0x00F0

Meaning

Application Controller available

Description

n/a

Current State (or state change)

n/a

Resulting state

n/a

Solution

n/a

4.3 EtherCAT AL Status Codes

4.3.1 Error Code 0x0000

Meaning

No error

Description

No error

Current State (or state change)

Any

Resulting state

Current state

Solution

n/a

4.3.2 Error Code 0x0001

Meaning

Unspecified error

Description

No error code is defined for occurred error

Current State (or state change)

Any

Resulting state

Any + E

Solution

Read user manual or contact device manufacturer

4.3.3 Error Code 0x0002

Meaning

No Memory

Description

Less hardware memory, slave needs more memory.

Example: For slave configuration, application configuration files are downloaded (possibly via FoE or large CoE objects). The size of those files exceeds the local memory

Current State (or state change)

Any

Resulting state

Any + E

Solution

Download smaller files or objects.

Check user manual.

4.3.4 Error Code 0x0004

Meaning

Invalid Revision

Description

Output/Input mapping is not valid for this hardware or software revision (0x1018:03)

Current State (or state change)

P→S

Resulting state

P+E

Solution

Change mapping or use different hardware

4.3.5 Error Code 0x0011

Meaning

Invalid requested state change

Description

The EtherCAT State Machine (ESM) defines which state changes are allowed. All other state changes are not allowed

Example: If the master requests the slave to go from OP (AL Control = 0x08) directly to BOOT (AL Control = 0x03).

Current State (or state change)

P→S, I→O, P→O, O→B, S→B, P→B

Resulting state

Current State + E

Solution

Go step-by-step from the original state to the desired state.

4.3.6 Error Code 0x0012

Meaning

Unknown requested state change

Description

The ESM defines the following states. They are coded with fixed values (only lower (=right) nibble):

BOOT: AL Control = 0x03

INIT: AL Control = 0x01

PREOP: AL Control = 0x02

SAFEOP: AL Control = 0x04

OP: AL Control = 0x08

The fifth bit of the AL Control (left nibble is 1) is the “Error Acknowledge Bit”. If the slave is in AL STATUS = 0x14, i.e. ERROR SAFEOP the master acknowledges this by setting the Acknowledge bit.

Example: If any other value for AL Control than those specified are sent.

Current State (or state change)

Any

Resulting state

Current State + E

Solution

Do only request the defined states

4.3.7 Error Code 0x0013

Meaning

Boot state not supported

Description

Device does not support BOOT state, but the master requests the slave to go to BOOT (AL Control = 0x03

Current State (or state change)

I→B

Resulting state

I + E

Solution

n/a

4.3.8 Error Code 0x0014

Meaning

No valid firmware

Description

This error code may be returned after a firmware download, if the downloaded file cannot be used by the application controller

Current State (or state change)

I→P

Resulting state

I + E

Solution

Download a firmware that can be supported by the hardware and bootloader. Check Product Code and Revision Number (CoE object 0x1018). If this cannot be read from the firmware any more you may see this in the network configuration (CoE object dictionary) or probably in the ESI file (element Profile: ObjectDictionary:Objects:Object).

4.3.9 Error Code 0x0015

Meaning

Invalid mailbox configuration

Description

Mailbox communication (= acyclic parameter exchange) is done via two memory areas on the EtherCAT Slave Controller (ESC) – the “Output Mailbox” (master -> slave) and the “Input Mailbox” (slave-> master). Those memory areas are protected by SyncManagers to prevent from simultaneous access from master and salve controller at the same time. SyncManagers are hardware entities on the ESC. They are configured via certain registers in the ESC register area (starting at 0x0800). The configuration includes start address, length, and direction (output or input). If those settings differ from those expected by the host controller of the slave this error is returned

Current State (or state change)

I→B

Resulting state

n/a

Solution

Replace previous network description of old slave with the one of the new slave

4.3.10 Error Code 0x0016

Meaning

Invalid mailbox configuration

Description

Example: The slave hardware was replaced while the network configuration remained unchanged. The new hardware expects different mailbox SyncManager settings

Current State (or state change)

I→S

Resulting state

I + E

Solution

Replace previous network description of old slave with the one of the new slave

4.3.11 Error Code 0x0017

Meaning

Invalid Sync Manager configuration

Description

Process data communication (cyclic communication) is done via extra memory areas on the ESC, separated for outputs and inputs. The process data length and the process data SyncManager length have to be the same. If this is not the case or the start address or direction does not match this error is returned.

Example: The process data configuration was changed of the slaves which also changed the length of the data. The change was not activated in the configuration so that the configuration tool would have recalculated the SyncManager settings.

Current State (or state change)

P→S, S→O

Resulting state

Current State + E

Solution

Issue a re-calculation of the EtherCAT configuration

4.3.12 Error Code 0x0018

Meaning

No valid inputs available

Description

The slave application cannot provide valid input values

Example: A certain hardware which needs to be connected to the slave was disconnected

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

n/a

4.3.13 Error Code 0x0019

Meaning

No valid outputs available

Description

The slave application cannot recieve valid output values.

Example: The slave has a RxPdoToggle output or an “Output Valid” information in its process data. The RxPdoToggle does not toggle or the OutputValid is not true. Therefore the slave has no process data which the application can use. If supported, check the RxPDO Toggle Failed Counter in object 0x1C3x.0E). Also, the Synchronization may have problems (see object 0x10F1:SI2 Sync Error Counter Limit) so that process data are received too late by the slave so that the local slave cycle misses the toggle event. Another reason can be that the PLC stopped working

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

The RxPdoToggle may need to be handled by the PLC program

The outputs valid may have to be set by the PLC program

PLC may have stopped, restart PLC

4.3.14 Error Code 0x001A

Meaning

Synchronization error

Description

If too many RxPDO Toggle error occur, i.e. the RxPDO Toggle Failed Counter increases the internal limit the slave returns to SAFEPERROR with 0x001A. Multiple synchronization errors. Device is not synchronized any more (used if the causes mirrored by the AL Status Codes 0x2C, 0x2D, 0x32, 0x33, 0x34 cannot be distinguished).

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

n/a

4.3.15 Error Code 0x001B

Meaning

Sync manager watchdog

Description

The slave did not receive process data within the specified watchdog time. Usually, the WD time is 100ms. The WD is re-started every time it receives new process data, usually when the Output SyncManager (SyncManager2) is written. For devices which have only inputs usually no WD is used. Increasing the WD is not a solution.

Reason: PLC stopped

Current State (or state change)

O, S

Resulting state

S + E

Solution

n/a

4.3.16 Error Code 0x001C

Meaning

Invalid Sync Manager Types

Description

n/a

Current State (or state change)

O, S, O, P→S

Resulting state

S + E

Solution

n/a

4.3.17 Error Code 0x001D

Meaning

Invalid Output Configuration

Description

SM configuration for output process data is invalid

Current State (or state change)

O, S, O, P→S

Resulting state

S + E

Solution

n/a

4.3.18 Error Code 0x001E

Meaning

Invalid Input Configuration

Description

SM configuration for input process data is invalid

Current State (or state change)

O, S, O, P→S

Resulting state

S + E

Solution

n/a

4.3.19 Error Code 0x001F

Meaning

Invalid Watchdog Configuration

Description

The Watchdog is configured in the ESC register 0x0400 and 0x0420. EtherCAT defines default watchdog settings (100ms) or they are defined in the ESI file. If the slave does not accept a change of the expected settings it returns this AL Status Code Example: A slave may not accept that the WD is deactivated.

Current State (or state change)

O, S, O, P→S

Resulting state

P + E

Solution

Use default WD settings

4.3.20 Error Code 0x0020

Meaning

Slave needs cold start

Description

Slave device require a power off - power on reset

Current State (or state change)

Any

Resulting state

Current State + E

Solution

n/a

4.3.21 Error Code 0x0021

Meaning

Slave needs INIT

Description

Slave application requests INIT state

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.3.22 Error Code 0x0022

Meaning

Slave needs PREOP

Description

Slave application requests PREOP state

Current State (or state change)

S, O

Resulting state

S + E, O + E

Solution

n/a

4.3.23 Error Code 0x0023

Meaning

Slave needs SAFEOP

Description

Slave application requests SAFEOP state

Current State (or state change)

O

Resulting state

O + E

Solution

n/a

4.3.24 Error Code 0x0024

Meaning

Invalid Input Mapping

Description

The process data are described by the configuration (PdoConfig) and PDO assignment (PdoAssign).

PdoConfig: list of actual variables (usually indexes 0x6nnn for inputs and 0x7nnn for outputs). Variables are also called PDO entries. There can be one or several variables with in one list (i.e. within one PDO). The Input PDOs have the index 0x1Amm. The Output PDOs have the index 0x16mm.

PdoAssign: The list of PDOs (object index 0x16nn, 0x1Amm) which are actually part of the process data and hence, are transferred cyclically, are listed in the PDO Assign Objects 0x1C12 (output PDOs) and 0x1C13 (input PDOs). All this can be seen in the SystemManager on the TAB “Process Data”. If the mapping which was set by the user on the Process Data tab and which was expected by the slave do not match this Status Code is returned.

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.3.25 Error Code 0x0025

Meaning

Invalid Output Mapping

Description

The process data are described by the configuration (PdoConfig) and PDO assignment (PdoAssign).

PdoConfig: list of actual variables (usually indexes 0x6nnn for inputs and 0x7nnn for outputs). Variables are also called PDO entries. There can be one or several variables with in one list (i.e. within one PDO). The Input PDOs have the index 0x1Amm. The Output PDOs have the index 0x16mm. Example: Slave does only support one or certain PDO combinations but a different setting was made by the user. For a bus coupler the connected terminals differ from the configured terminals in the SystemManager

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.3.26 Error Code 0x0026

Meaning

Inconsistent Settings

Description

General settings mismatch

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.3.27 Error Code 0x0027

Meaning

Freerun not supported

Description

n/a

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.3.28 Error Code 0x0028

Meaning

Synchronization not supported

Description

n/a

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.3.29 Error Code 0x0029

Meaning

Freerun needs 3 Buffer Mode

Description

FreeRun mode, SM has to run in 3-buffer mode

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.3.30 Error Code 0x002A

Meaning

Background Watchdog

Description

n/a

Current State (or state change)

S, O

Resulting state

P + E

Solution

n/a

4.3.31 Error Code 0x002B

Meaning

No Valid Inputs and Outputs

Description

n/a

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

n/a

4.3.32 Error Code 0x002C

Meaning

Fatal Sync Error

Description

The hardware interrupt signal (so called Sync signal) generated by the ESC is not generated any more. The master sets and activated the cycle time of the Sync signal during state transition from PREOP to SAFEOP. If a slave was disconnected and reconnected (also due to lost frames or CRC errors) the generation of the SyncSignal may be lost.

Current State (or state change)

O

Resulting state

S + E

Solution

Set master to INIT and back to OP so that the DCs are initialized again

4.3.33 Error Code 0x002D

Meaning

ana

Description

SyncSignal not received: In SAFEOP the slave waits for the first Sync0/Sync1 events before switching to OP, if these events were not received during the SAFEOP to OP-Timeout time the slave refuses the state transition to OP

Current State (or state change)

n/a

Resulting state

n/a

Solution

n/a

4.3.34 Error Code 0x0030

Meaning

Invalid DC SYNC Configuration

Description

Distributed Clock Configuration is invalid due to application requirements

Current State (or state change)

O, S→O, P→S

Resulting state

P + E, S + E

Solution

n/a

4.3.35 Error Code 0x0031

Meaning

Invalid DC Latch Configuration

Description

DC Latch configuration is invalid due to application requirements

Current State (or state change)

O, S→O, P→S

Resulting state

P + E, S + E

Solution

n/a

4.3.36 Error Code 0x0032

Meaning

PLL Error

Description

Master not synchronized, at least one DC event recieved

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

n/a

4.3.37 Error Code 0x0033

Meaning

DC Sync IO Error

Description

Multiple Synchronization Errors: At least one SycnSignal was received before. However, the PLL between slave and master is not synchronized any more. This may occur if the master application jitters too much

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

Use specific industrial pc, standard office PCs may have power saving options, graphic accelerateds and other system services which disturb the real-time of the master.

CPU power may be too small for the PLC/NC program.

Increase EtherCAT and PLC/NC cycle time.

Use SyncUnits for the slaves using DCs.

4.3.38 Error Code 0x0034

Meaning

DC Sync Timeout Error

Description

Multiple Synchronization Errors, too much SM events missed

Current State (or state change)

O, S→O

Resulting state

S + E

Solution

n/a

4.3.39 Error Code 0x0035

Meaning

DC Invalid Sync Cycle Time

Description

n/a

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.3.40 Error Code 0x0036

Meaning

DC Sync0 Cycle Time

Description

DC Sync0 cycle time does not fit to the application requirements

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.3.41 Error Code 0x0037

Meaning

DC Sync1 Cycle Time

Description

DC Sync1 cycle time does not fit to the application requirements

Current State (or state change)

P→S

Resulting state

P + E

Solution

n/a

4.3.42 Error Code 0x0041

Meaning

MBX_AOE

Description

n/a

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.3.43 Error Code 0x0042

Meaning

MBX_EOE

Description

n/a

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.3.44 Error Code 0x0043

Meaning

MBX_COE

Description

n/a

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.3.45 Error Code 0x0044

Meaning

MBX_FOE

Description

n/a

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.3.46 Error Code 0x0045

Meaning

MBX_SOE

Description

n/a

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.3.47 Error Code 0x004F

Meaning

MBX_VOE

Description

n/a

Current State (or state change)

B, P, S, O

Resulting state

Current State + E

Solution

n/a

4.3.48 Error Code 0x0050

Meaning

EEPROM No Access

Description

EEPROM not assigned to PDI

Current State (or state change)

Any

Resulting state

Any + E

Solution

n/a

4.3.49 Error Code 0x0051

Meaning

EEPROM Error

Description

EEPROM access error

Current State (or state change)

Any

Resulting state

Any + E

Solution

n/a

4.3.50 Error Code 0x0060

Meaning

Slave Requested Locally

Description

n/a

Current State (or state change)

Any

Resulting state

I

Solution

n/a

4.3.51 Error Code 0x0061

Meaning

Device Identification Value updated

Description

n/a

Current State (or state change)

P

Resulting state

P + E

Solution

n/a

4.3.52 Error Code 0x00F0

Meaning

Application Controller available

Description

n/a

Current State (or state change)

n/a

Resulting state

n/a

Solution

n/a

5 EtherCAT Betrieb - Ansteuerung

5.1 Operation

In Vorbereitung

6 EtherCAT Betrieb - Timing

6.1 Konzept Mapping

EtherCAT Slaves werden im System Manager direkt mit dem Prozessabbild einer Task verlinkt. Dies führt zum so genannten synchronen Mapping - der Schreib/Lesezugriff auf das Prozessabbild erfolgt abwechselnd durch I/O-Update/Feldbus und Task.

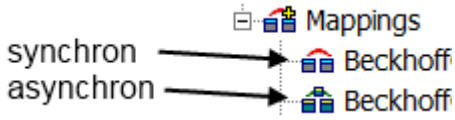


Abb. 239: System Manager Icons bei synchronem/asynchronem Mapping

Bei einem asynchronen Mapping greifen die beiden Prozesse "Task" und "Feldbus Update" in ihrer jeweiligen Zykluszeit auf das Prozessabbild zu.

Auswirkung Link

Sind I/Os direkt mit einer Task verlinkt, wird das entsprechende EtherCAT-Feldbusupdate auch mit dieser Task-Zykluszeit synchron durchgeführt. Es werden jedoch nicht mehr als "max. Sync Task" I/O-Zyklen ausgeführt. (EtherCAT Master --> Erweiterte Einstellungen --> Cyclic Task).

Sind Geräte nicht mit einer Task verlinkt, werden sie mit der langsamsten verfügbaren Task im asynchronen Mapping angesprochen. Dies ist auch bei automatischer eingerichteter Querkommunikation wie z. B. bei Safety-Geräten der Fall.

Watchdog

In den Mapping-Einstellungen ist zu prüfen, ob solcherart eingerichtete Timings nicht die Watchdog-zeiten überschritten.
Beispiele für Watchdogs:

- I/O Watchdog: default 100 ms
- FSoE Watchdog: 100 ms (bestätigte Kommunikation, Zykluszeiten <= 25 ms empfohlen bei Standardeinstellung)

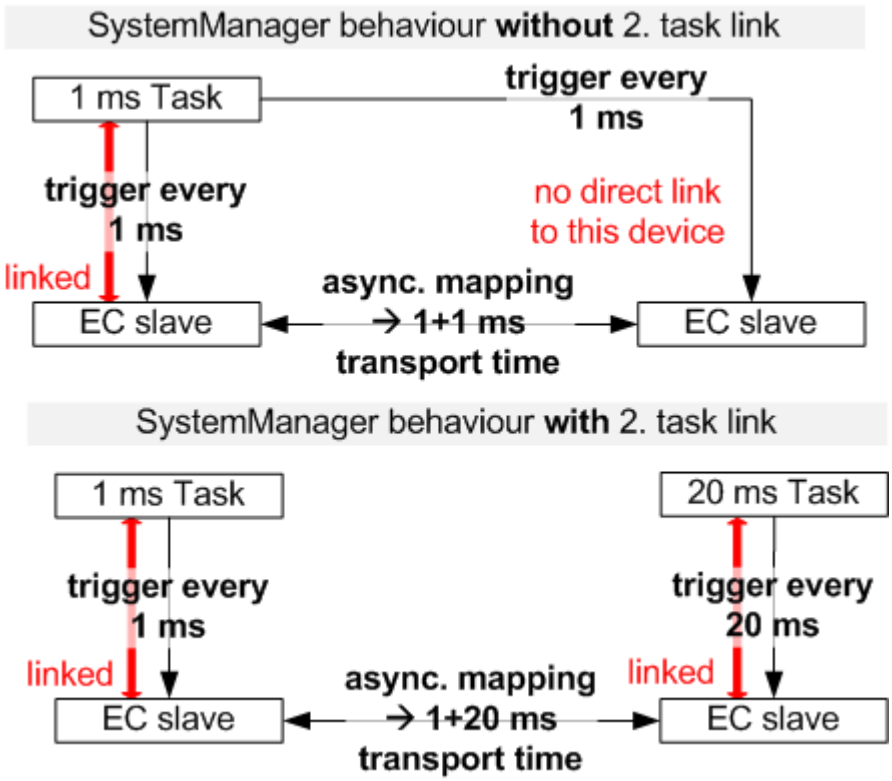


Abb. 240: Automatisch eingerichtetes asynchrones Mapping

6.2 Zuordnungsarten und grafische Darstellung

6.2.1 Zuordnung

Unterhalb des Baumeintrags *Zuordnungen*, gibt es eine Auflistung aller Prozessabbild-Verknüpfungen.

Abb. 241: TC-Baum: Zuordnungen, Liste der Verknüfungen

Auf der rechten Seite erscheint der entsprechende Dialog für die angewählte Zuordnung, wie im Folgenden zu sehen.

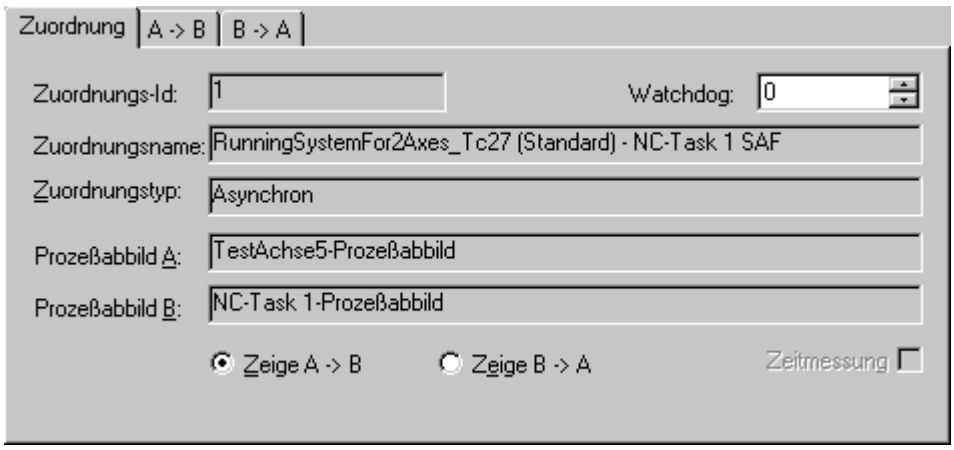


Abb. 242: Reiter „Zuordnung“

Zuordnungs-ID

Identifikationsnummer zur internen Verwaltung der verschiedenen Zuordnungen (Mappings).

Zuordnungsname

Nennt die Namen der beiden verknüpften Prozessabbilder.

Zuordnungstyp

Synchron: Ein Prozessabbild ist Master, ein zweites Slave. Der Master gibt die Ausgänge zum Schreiben frei (z. B. auf Feldbuskarte C1220, ..) und schaut ob die Gegenseite mit ihrem I/0-Zyklus fertig ist um die aktuellen Eingänge zu lesen. Die Gegenseite hat dabei keinen eigenen Zeittakt (engl. Timekeeper).

Asynchron: Wird z. B. bei Verknüpfungen von zwei Tasks eingesetzt, oder bei Geräten, die mit eigenem Zeittakt arbeiten (z. B. COM-Port) und damit die Auffrischung von Ein- und Ausgängen selbständig regeln. Der Austausch der Informationen zwischen zwei Prozessabbildern geschieht daher beim asynchronen Mapping mit Hilfe des Drei-Puffer-Prinzips.

Bei manchen Gerätetypen (z. B. die Multitasking-fähige Profibuskarte *FC310x*) wird eine Mischform aus synchroner und asynchroner Zuordnung (Mapping) verwendet. Die Task mit der höheren Priorität verhält sich daher synchron, die niederpriore asynchron zum Gerät.

Watchdog

Beim asynchronen Mapping (*sieheZuordnungstyp*) kann es passieren, dass sich eine Task nicht mehr korrekt beendet wird (Endlosschleife), die andere Task aber weiterbearbeitet wird und daher immer ständig alte Werte aus den Puffern ausliest. Daher ist hier ein Maximalwert für einen Lebenszykluszähler eingebbar. Wird der Wert erreicht, werden alle Werte der Puffer auf '0' gesetzt.

Zeitmessung

Aktiviert die Zeitmessung bei einer synchronen Zuordnung (*siehe Zuordnungstyp*).

Prozessabbild A

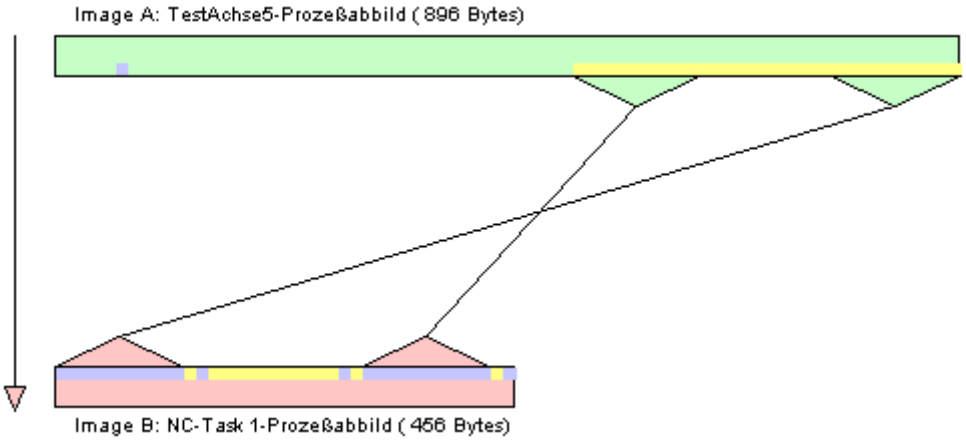
Hier ist der Taskname, der dem Prozessabbild A zugeordnet ist, eingetragen.

Prozessabbild B

Hier ist der Taskname, der dem Prozessabbild B zugeordnet ist, eingetragen.

Zeige A -> B bzw. B-> A

Vertauscht die Sichtweise auf die beiden Prozessabbilder in der dargestellten Ansicht.



Prozessabbilder A, B

Die Farben Grün und Rose bedeuten Prozessabbild A und B, die Farben gelb und blau Ein- bzw. Ausgänge des Prozessabbildes. Wenn man mit der Maus auf die Ein- bzw. Ausgänge geht, erscheint ein sogenannter "Tooltip" mit dem Variablennamen.

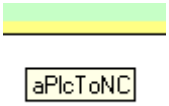


Abb. 243: Variablenname

6.2.2 Kontext-Menü

Bei rechtem Mausklick erscheint das Kontext-Menü zur Einstellung der Zoom-Auflösung des Zuordnungsgraphen. Je höher der eingestellte Pixelwert pro Byte, desto einfacher ist es, die Verknüpfungen einzelner Variablen zu begutachten.

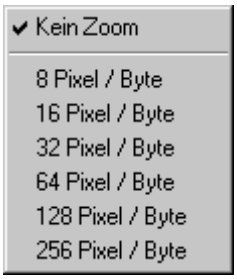


Abb. 244: Einstellung Zoomfaktor

6.2.3 Karteireiter „A -> B“ bzw. „B -> A“

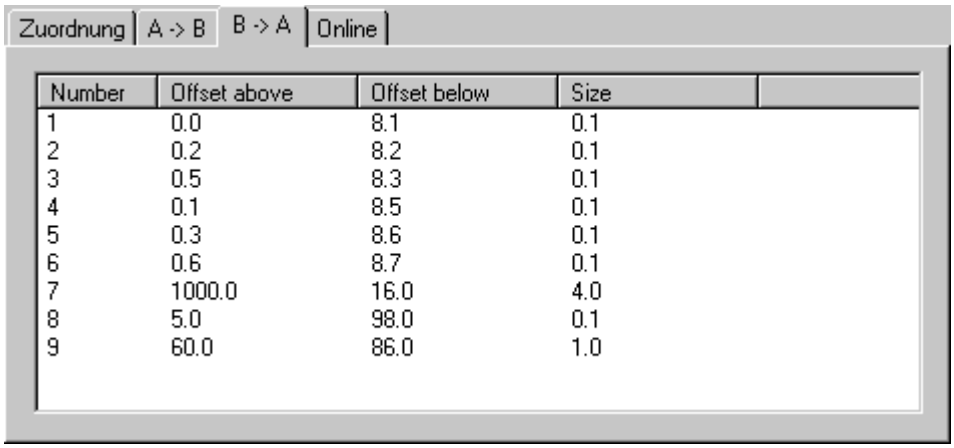


Abb. 245: Reiter „B -> A“

Number

Laufende Nummer der Kopieraktionen.

Offset A

Gibt den Offset innerhalb von Prozessabbild A an, von dem die Kopieraktion ausgeht.

Offset B

Gibt den Offset innerhalb von Prozessabbild B an, von dem die Kopieraktion ausgeht.

Size

Länge der zu kopierenden Werte ab jeweiligem Offset (bei Wert *0.1* würde z. B. ein Bit kopiert).

6.2.4 Karteireiter Online

Hier findet man, bei aktiver Konfiguration und gestartetem System, eine grafische Darstellung der Zeit (in Nanosekunden), welche für die Kopieraktion von Prozessabbild A nach B, bzw. B nach A aktuell benötigt wird. Dieser Dialog existiert seit TwinCAT 2.8 bei synchronen und asynchronen Zuordnungen (siehe auch "Zuordnungstypen" weiter oben).

Asynchrone Zuordnungen:

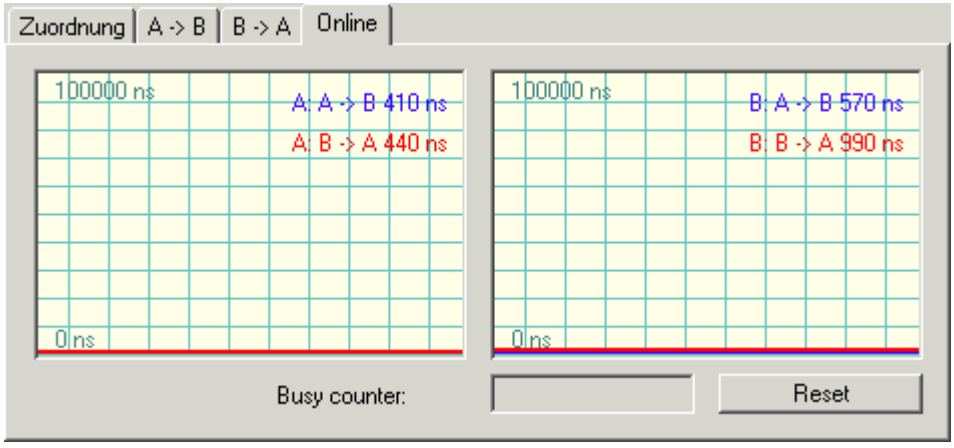


Abb. 246: Online-Darstellung „Asynchrone Zuordnungen“

Die angegebene Zeit neben

A: A->B bezieht sich auf das Kopieren aller Daten dieser Zuordnung von Prozessabbild A in den Puffer für B.

A: B->A bezieht sich auf das Kopieren aller Daten dieser Zuordnung von Prozessabbild A aus dem Puffer von B.

B: A->B bezieht sich auf das Kopieren aller Daten dieser Zuordnung von Prozessabbild B aus dem Puffer von A.

B: B->A bezieht sich auf das Kopieren aller Daten dieser Zuordnung von Prozessabbild B in den Puffer für A.

Synchrone Zuordnungen:

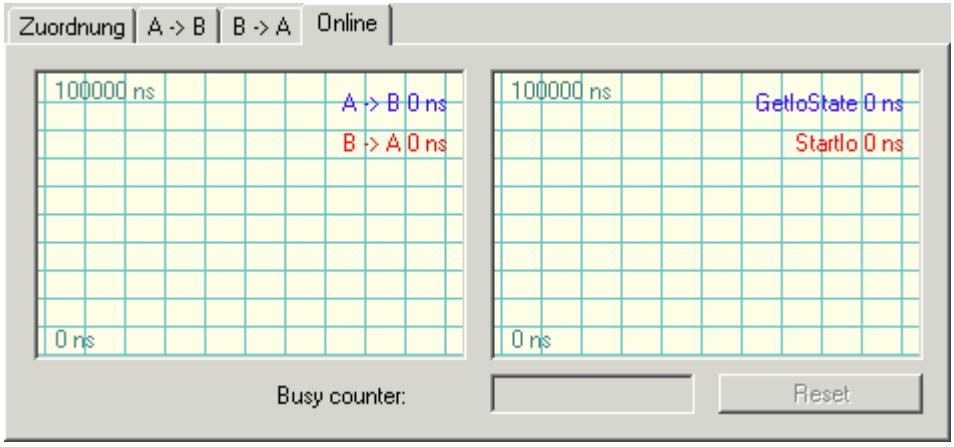


Abb. 247: Online-Darstellung „Synchrone Zuordnungen“

Die angegebene Zeit neben

A->B bezieht sich auf das Kopieren aller Daten dieser Zuordnung von Prozessabbild A nach B.

B->A bezieht sich auf das Kopieren aller Daten dieser Zuordnung von Prozessabbild A von B.

GetIoState bezieht sich auf die Dauer der Überprüfungsfunktion GetIoState(), die abhängig von der verwendeten Feldbuskarte unterschiedlich aufwendig ist.

StartIo bezieht sich auf die Dauer der Function StartIo(), die den Bus startet und abhängig von der verwendeten Feldbuskarte unterschiedlich aufwendig ist.

Hinweis:

Alle Zeiten werden als Differenzzeiten zwischen Beginn und Ende der jeweiligen Aktion gemessen und können daher bei Unterbrechungen stark schwanken.

Details zur kontinuierlich fortlaufenden, grafischen Anzeige des aktuellen Online-Wertes finden sie unter: Einstellungen History-Anzeige.

Busy counter

Bei synchroner Zuordnung von Prozessabbilder schaut das Master-Prozessabbild nach, ob der Slave mit seinem I/O-Zyklus fertig ist und neue Eingänge zur Abholung bereit stehen. Sollte dies nicht der Fall sein, wird der *Busy counter* durch den Master inkrementiert und in dem oben gezeigten Feld angezeigt. Bei *asynchronen Zuordnungen* bleibt das Feld leer.

Reset

Setzt den *Busy counter* zurück auf '0'.

7 Distributed Clocks

7.1 TwinCAT & Zeit

7.1.1 TwinCAT Zeitquellen

In der Beckhoff TwinCAT Automatisierungssuite können mehrere unabhängige Zeitquellen ausgewertet werden. Siehe dazu auch das Beispielprogramm [▶ 232].

Angaben zur Architektur:

Name	BIOS Mother-board	CPU Zeit	Windows/NT Zeit	TwinCAT/TC Zeit	Distributed Clocks/DC Zeit
Beschreibung	RTC (RealTime-Clock), batteriege-speist auf dem Motherboard	CPU Counter aus der Hardware der Steuerung, nicht geregelt Initialisiert durch RTC	Lokale Systemzeit des Win-dows-Betriebssystems (NT) Initialisiert durch RTC	Fortlaufende TwinCAT-Uhr Initialisiert durch Windows	Die Startzeit des aktuel-len Task-Zyklus' wird zurückgegeben. Initialisiert durch TwinCAT.
Daten		64 bit Auflösung: 100 $ns_{PC\ Base}$	ab 1.1.1601 00:00 Auflösung: 1 ms Umfang: Struktur mit Jahr, Mo-nat, Tag, Stunde usw.	ab 1.1.1601 00:00 Auflösung: 100 ns Umfang: 64 bit	ab 1.1.2000 00:00 Auflösung: 1 ns Umfang: 64 bit
Bezug			Lokal	Lokalzeit unter Berücksichtigung der eingestellten Zeitzone, meist al-so UTC	Lokalzeit unter Berück-sichtigung der einge-stellten Zeitzone, meist also UTC
PLC-Format		T_ULARGE_IN-TEGER	TIMESTRUCT	T_FILETIME	T_DcTime
Aufruf		GetCpuCounter	NT_GetTime()	GetSystemTime()	F_GetActualDcTime*() (ab TwinCAT 2.11)* *F_GetCurDcTickTime() (= GetSystemTime)* *F_GetCurDcTaskTime() (ab TwinCAT 2.11)*
Aktualisierung		bei jedem Aufruf, auch mehrmals innerhalb eines PLC-Zyklus		bei jedem Basi-stick (System Ma-nager | BaseTime)	ActualDcTime bei jedem Aufruf, auch mehrmals innerhalb ei-nes PLC-Zyklus *TickTime* bei jedem Basistick (System Manager | Ba-seTime) *TaskTime* bei Zyklusbeginn der Sync-Task
beispielsweise Verwendung	kann durch den PLC-Baustein *Nt_SetTime-ToRtcTime* zur Korrektur der NT-Zeit verwendet werden	relative Zeitmes-sungen	Logging, Zeitstempelung auf Betriebssystemebene	hochgenaue rela-tive zeitbasierte Aktionen inner-halb eines oder über mehrere Task-Zyklen	- hochgenaue relative zeitbasierte Aktionen in-nerhalb des EtherCAT-Systems - endgültiger Bezug zu Globalzeit durch exter-ne Synchronisierung möglich
Manipulations-möglichkeit			- durch den PLC-Baustein *Nt_SetTimeToRtcTime* kann sie auf die aktuelle RTC-Zeit geän-dert werden, dies löst auch eine Korrektur der RTC-Zeit aus ACHTUNG eine Verwendung dieser Funkti-on in Verbindung mit EtherCAT Distributed Clocks Systemen wird nicht empfohlen - Synchronisierung auf Netz-werkebene (SNTP, NTP)		Synchronisierung zu ex-terner Referenzzeit ab TwinCAT 2.11

Tab. 1: Zeitarchitektur, benötigte Bibliotheken: TcEtherCAT.lib, TcUtilities.lib, TcSystem.lib

Einsatzszenario 1: Lokale Steuerung ohne netzwerkseitige Zwangssynchronisierung

Die lokale Windowsuhr ist freilaufend und kann durch *Nt_SetTimeToRtcTime* an die RTC gekoppelt werden. Beim Einsatz von Distributed Clocks Komponenten wird diese Möglichkeit nicht empfohlen!

Es wird die Verwendung der TC- oder DC-Uhr empfohlen, wenn absolute Zeitbezüge benötigt werden.

Ideal ist die Kopplung der DC-Zeit an eine externe Referenzzeit durch entsprechende EtherCAT-Komponenten.

Einsatzszenario 2: Lokale Steuerung mit netzwerkseitiger Zwangssynchronisierung

Die lokale Windowsuhr wird durch eine Netzwerkuhr/-server/Internet Zeitserver zyklisch mit der Weltzeit synchronisiert.

Eine Kopplung der Windowsuhr durch *Nt_SetTimeToRtcTime* an die RTC wird nicht empfohlen.

Es wird die Verwendung der TC- oder DC-Uhr empfohlen. Der Bezug zur Absolutzeit kann applikationsseitig durch Offsetberechnung zur NT-Zeit hergestellt werden.

Ideal ist die Kopplung der DC-Zeit an eine externe Referenzzeit durch entsprechende EtherCAT-Komponenten.

Einsatzszenario 3: Lokale Steuerung mit externer Referenzzeit über EtherCAT

Durch Kopplung der DC-Zeit an eine externe Zeitquelle (GPS, Funkuhr, PTP/IEEE1588, EtherCAT) steht eine in Frequenz und Phase synchronisierte stetige Zeit zur Verfügung. NT- und TC-Zeit werden in der Applikation nicht benötigt.

Übliche Zeitsynchronisation

Die auf Betriebssystemebene übliche Zeitsynchronisation arbeitet in diskreten Intervallen von mehreren Sekunden bis Tagen. Im Synchronisationsfall führt dies zu einer sprunghaften/unsteten Änderung der unterlagerten Zeit! Davon betroffen sind übliche Netzwerksynchronisationen (SNTP, NTP u.ä.) oder auch Nt_SetTimeToRtcTime. Die Applikation muss diese sprunghaften Änderungen der "Absolut"zeit erwarten!
Alle von Beckhoff in EtherCAT integrierten Verfahren arbeiten mit stetiger, nicht sprungbehafteter (Auf)Synchronisation.

7.1.2 Interne und externe EtherCAT Synchronisierung

7.1.2.1 Allgemeines

In einer Maschinensteuerung mit verteilten Komponenten (I/O, Antrieben, div. Mastern) kann es zweckmäßig sein, dass die Komponenten in engem zeitlichen Bezug zueinander arbeiten. In den Komponenten muss also lokal eine "Uhrzeit" vorhanden sein, auf die die Komponente (z. B. eine I/O-Klemme) jederzeit Zugriff hat.

Solche Anforderungen können sein:

1. Mehrere Ausgänge in einer Steuerung müssen gleichzeitig gesetzt werden, unabhängig davon wann die betreffende Station die Ausgangsdaten bekommt.
2. Antriebe/Achsen in einer Steuerung müssen synchron ihre Achsposition einlesen, unabhängig von der Topologie oder Zykluszeit.

Beide Anforderungen bedeuten, dass ein Synchronisierungsmechanismus zwischen den lokalen Uhrzeiten der Komponenten einer Steuerung besteht.

1. Wenn Eingänge auf die Steuerung einwirken, muss die (absolute) Zeit festgehalten werden - dies kann zur späteren Analyse hilfreich sein, wenn durch Analyse der Abfolge von Ereignissen Wirkungsketten nachvollzogen werden müssen.
   Dies bedeutet, die in den Komponenten laufende Uhrzeit an eine global gültige Zeit, z. B. die Weltzeit nach Greenwich oder eine Netzwerkuhr, angekoppelt sein muss.
2. Tasks auf verschiedenen Steuerungen sollen synchron und ohne Phasenverschiebung laufen.

Die Begriffe "enger zeitl. Bezug" oder "gleichzeitig" können je nach Anforderung quantitativ umgesetzt werden: für einen "Gleichzeitigkeit" im 10 ms Bereich kann eine serielle Kommunikationsstruktur ausreichend sein, in manchen Bereichen sind hier 100 ns und weniger gefordert.

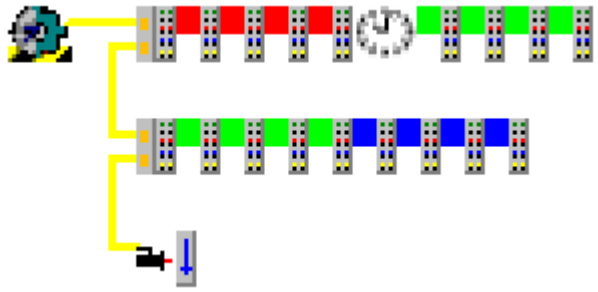


Abb. 248: Einfache I/O-Topologie

In Abb. *Einfache I/O-Topologie* ist eine einfache EtherCAT-Topologie dargestellt, bestehend aus Master, diversen E/A und einer Achse. In verschiedenen Komponenten solle nun eine lokale Uhrzeit betrieben werden. Die Aufgaben:

- Synchronisierung der lokalen Uhren
- Ankopplung an einer übergeordneten Referenzzeit
- Tasksynchronisierung

werden im Folgenden besprochen.

7.1.2.2 Anforderung 1 + 2: Synchronisierung

In einem EtherCAT-System wird das Distributed-Clocks-Konzept (DC) zur Synchronisierung der lokalen Uhren in den EtherCAT-Komponenten benutzt.

Synchronisierung der lokalen EtherCAT-Teilnehmer

Allgemein:

- Auflösung der Uhrzeit 1 ns entsprechend 1 digit, Umfang 64 Bit entsprechend ca. 584 Jahre
- Der EtherCAT-Master muss mit Synchronisierungsdatagrammen die verteilten Uhren im Rahmen der Systemgenauigkeit (EtherCAT: <100 ns) synchron halten.
- Nicht jeder EtherCAT-Teilnehmer muss dieses Feature unterstützen. Wenn ein Slave dieses Konzept nicht unterstützt, wird er vom Master nicht in die Synchronisierung mit aufgenommen. Wenn der verwendete EtherCAT-Master dieses Feature nicht unterstützt, ist DC auch in allen Slaves wirkungslos.
- Auch im EtherCAT-Master läuft eine solche Uhr, dort softwarebasiert.
- im System wird *eine* der vorhandenen Clocks als Reference-Clock ausgewählt - auf sie werden alle anderen Clocks synchronisiert. Diese Referenzuhr ist üblicherweise eine der Uhren der EtherCAT-Slaves, nicht die des EtherCAT-Masters. Üblicherweise wird der erste EtherCAT-Slave in der Topologie, der die Distributed-Clocks unterstützt, als Referenzuhr automatisch ausgewählt.
- es ist im Folgenden also zu unterscheiden zwischen
  - dem EtherCAT-Master (die Software die mit Ethernet-Frames die EtherCAT-Slaves "verwaltet") und den von ihm verwalteten EtherCAT-Slaves.
  - der Reference-Clock die üblicherweise im ersten DC-Slave sitzt und den ihr nachgeregelten Slave-Clocks, einschließlich der Uhr im EtherCAT-Master.

Zum Master:

- der EtherCAT-Master muss in der Systemstartphase die lokalen Uhr der Reference-Clock und der anderen Slave-Clocks auf die aktuelle Zeit setzen und im Folgenden durch zyklische Synchronisierungsdatagramme die Abweichungen der Uhren untereinander minimieren.
- bei Topologieänderungen muss der EtherCAT-Master entsprechend die Uhren neu synchronisieren
- nicht jeder EtherCAT-Master unterstützt dieses Verfahren
- der EtherCAT-Master in der Beckhoff TwinCAT Automatisierungssuite unterstützt Distributed Clocks in vollem Umfang.

Zum Slave:

- Auf Grund der hohen erforderlichen Exaktheit wird diese lokale Uhr in Hardware (ASIC, FPGA) ausgeführt.
- Distributed Clocks wird im EtherCAT Slave Controller (ESC) in den Registern 0x0900 - 0x09FF verwaltet, konkret läuft in den 8 Byte ab 0x0910 die lokale synchronisierte Uhrzeit.

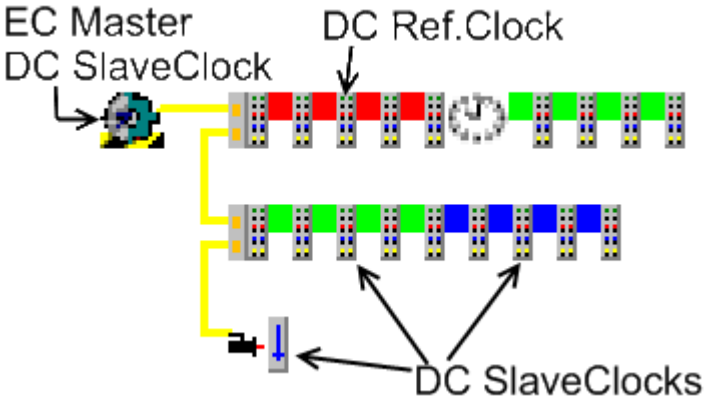


Abb. 249: Abbildung DC auf die Topologie

In Abb. *Abbildung DC auf die Topologie* wurde beispielhaft der 3. EtherCAT-Slave als DC-Reference-Clock ausgewählt - nach dessen lokaler Uhrzeit werden nun alle anderen Ausprägungen der verteilten Uhren nachgeregelt, also alle anderen EtherCAT-Slaves und die Uhr im EtherCAT-Master. Dies geschieht durch Synchronisierungsdatagramme, die der EtherCAT-Master zyklisch verschickt.

Nachregelung TwinCAT-Clock

Damit die PLC/NC-Tasks im Echtzeitkontext der Steuerung gleichzeitig mit den Distributed Clocks laufen, muss die allein echtzeitbestimmende TwinCAT-Uhr der DC-ReferenceClock nachgeführt werden. Wird mehr als 1 EtherCAT-System auf einer Steuerung eingesetzt, kann nur eines dieser Systeme die ReferenceClock stellen, der TwinCAT nachgeführt wird. Die anderen EtherCAT-Systeme wiederum müssen dann der TwinCAT-Uhr folgen.
Siehe dazu die Hinweise im Kapitel "Kopplung von EtherCAT Systemen [▶ 161]"

Durch dieses Verfahren ist gewährleistet, dass in allen DC-unterstützenden Teilnehmern jederzeit lokal auf eine Uhrzeit zurückgegriffen werden kann, die im Rahmen der DC-Synchronisierungsgenauigkeit in allen Teilnehmern gleich ist.

Das System arbeitet nun auf Basis der Zeitbasis der ausgewählten DC-Reference-Clock bzw. deren lokalem Taktgeber/Quarz mit T_{DC}. Diese Zeitbasis wird durch Produktions-/Fertigungsschwankungen kaum jemals gleich der amtlichen Sternzeit/koordinierte Weltzeit UTC T_{UTC} oder einer anderen Referenzzeit sein. Das bedeutet, 1 ms_{UTC} entspricht nie exakt 1 $ms_{DC,}$ $T_{DC} \neq T_{UTC}$. Über längere Zeiträume können auch Driftvorgänge das Verhältnis verändern. Solange DC für relative Vorgänge innerhalb des EtherCAT-Systems verwendet wird, spielt diese Abweichung von der UTC keine Rolle. Soll die DC-Zeit z. B. für Datenlogging mit globalem Zeitmaßstab verwendet werden, muss die $Zeitbasis_{DC}$ zur $Zeitbasis_{UTC}$ synchronisiert werden. Dies wird im Kapitel der Anforderung 3 beschrieben.

7.1.2.3 Anforderung 3: übergeordnete Globalzeit - Absolutzeit

Soll die Zeitbasis T_{DC} einer übergeordneten Zeitbasis nachgeregelt werden, ist dazu die Zeitbasis und das Verfahren zu wählen. Üblicherweise werden gängige Synchronisationsprotokolle zur Synchronisation verwendet, Zeitquellen und Synchronisationsverfahren können sein

- Quellen: Weltzeit UTC, Netzwerkzeit, benachbarte Steuerung, Funkuhren (in Mitteleuropa: DCF77)
- Verfahren: GPS, Funkuhren, NTP (NetworkTimeProtokoll), SNTP (Simple NTP), PTP (IEEE1588), DistributedClocks DC

Erreichbare Synchronisationsgenauigkeiten liegen dabei (je nach Hardware) bei

- NTP/SNTP: ms-Bereich
- PTP: < 1 µs

- DC: < 100 ns

Dabei sind die folgenden beiden Regelungsziele zu erreichen:

- die Frequenz der unterlagerten Zeitbasis ist der übergeordneten nachzuführen.
- ein ggf. bestehender Offset zwischen beiden Absolutzeiten muss nicht unbedingt zu 0 geregelt werden, es reicht ihn bekanntzugeben und konstant zu halten. Der Offset wird max. um ±½ Zykluszeit angepasst.

Externe EtherCAT-Synchronisation

Externe Synchronisationsquellen (EL6688, EL6692 u.a.) können erst ab TwinCAT 2.11 verwendet werden. In früheren Versionen von TwinCAT haben solche EtherCAT-Slaves keine sinnvolle Funktion.

Wird eine übergeordnete Master-Clock in ein EtherCAT-System eingebunden, wird dazu üblicherweise ein spezieller EtherCAT-Teilnehmer für den physikalischen Anschluss verwendet. Dieser kann, da er beide Zeitbasen beobachtet, die Zeitdifferenz ermitteln.

Bitte informieren Sie sich unter www.beckhoff.de über die für diesen Zweck geeigneten aktuell verfügbaren Produkte.

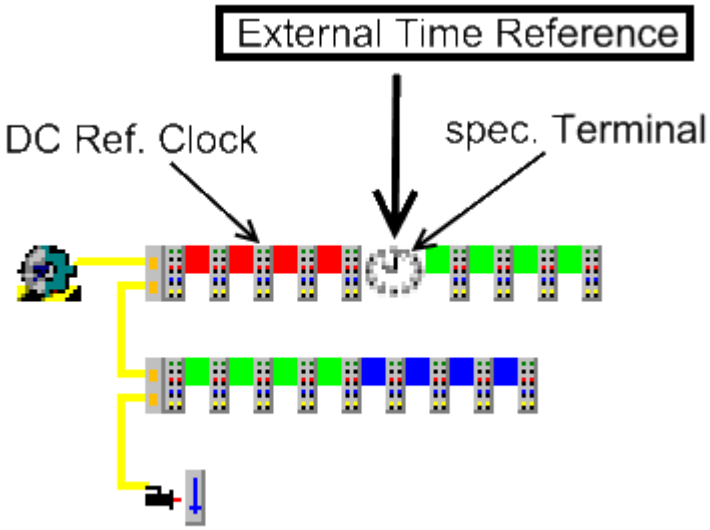


Abb. 250: EtherCAT-Topologie mit externer Referenz-Clock

Die unterschiedlichen Zeitbasen lassen sich hierarchisch anordnen - beim Start des jeweiligen System wird die aktuelle absolute Zeit vom jeweils unterlagerten System übernommen, ggf. wird eine Synchronisierung Top-Down wirksam, falls Externe Zeitbasis bzw. DC-Komponenten im System vorhanden sind..

Nachregelung Lokalzeit vs. übergeordnete Absolutzeit

Die lokale DC-Zeit wird im Synchronisierungsfall nicht der übergeordneten Absolutzeit vollständig angeglichen, sondern nur auf einen konstanten Offset nachgeregelt. Dem Anwender wird dieser Offset als Prozessdatum zur Verfügung gestellt. Dabei wird der Offset um ±½ Zykluszeit korrigiert, damit beide Tasks in Phase laufen.

- Wenn TwinCAT den EtherCAT-Master startet, wird umgehend das lokale DC-System in den Slaves in Betrieb genommen und synchronisiert.
- Ein ExternalReference-Slave wie z. B. EL6688 (IEEE1588 PTP) liefert aber erst nach einigen Sekunden eine mit der übergeordneten Uhr abgestimmte Referenzzeit.
- Sobald diese externe Referenzzeit zur Verfügung steht, wird der Offset zur Lokalzeit berechnet, um ±½ Zykluszeit korrigiert, damit beide Tasks in Phase laufen und dem Anwender in den Info-Daten des EtherCAT-Masters zur Verrechnung mit seinen lokalen Zeitwerten zur Verfügung gestellt.
- Ab diesem Zeitpunkt wird dieser Offset je nach gewählter Regelungsrichtung konstant gehalten.

7.1.2.4 Systemverhalten TwinCAT

Ausfall der externen Referenzclock

Fällt das Signal der externen Referenzclock aus, driften naturgemäß beide Zeitbasen wieder auseinander. Setzt das Signal wieder ein, wird auf den bisherigen Offsetwert stetig zurückgeregelt.

TwinCAT kann auch ohne Signal der externen Uhr starten, beim erstmaligen stabilen Empfang der externen Referenzclock wird der Offset wie oben beschrieben berechnet und beibehalten.

7.1.2.5 Einstellungen in TwinCAT 2.11

Ab TwinCAT 2.11 wird die externe Synchronisierung über EtherCAT unterstützt. Im entsprechenden Dialog kann die Synchronisierungsrichtung eingestellt werden.

Einstellungen Distributed Clocks Timing

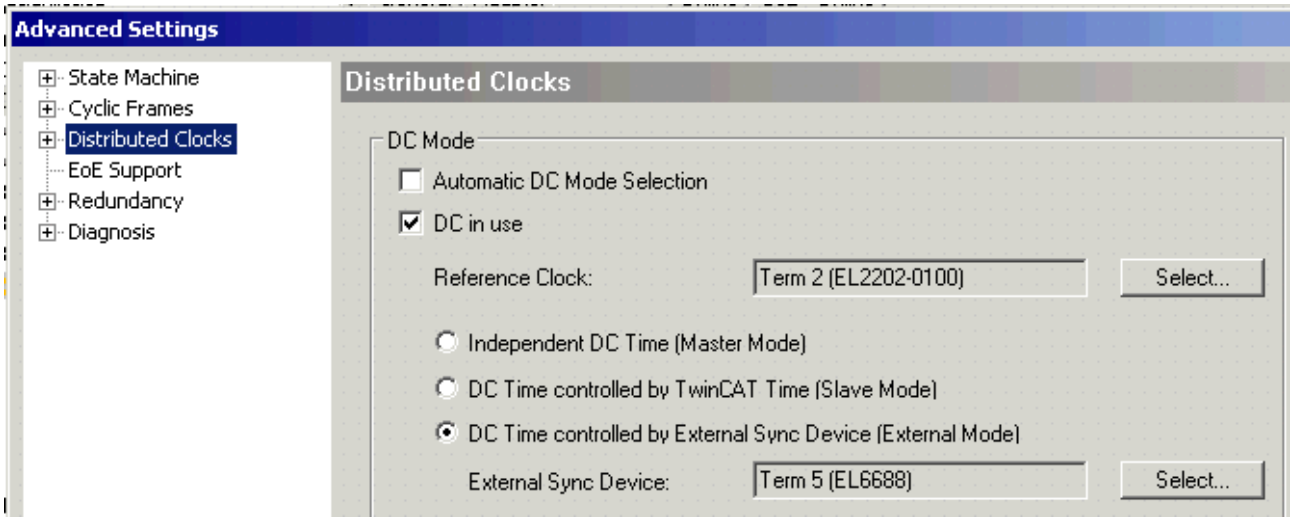


Abb. 251: TwinCAT 2.11 Distributed Clocks Settings - Beispiel für EL6688 im PTP-Slave-Modus als Zeitreferenz für das lokale EtherCAT-System

In Abb. *Synchronisierungsrichtung* ist die zur jeweiligen Synchronisierungsart gehörenden Synchronisierungsrichtung angeben - d.h. welche Quelle ihre Zeit dem Synchronisierten auferlegt.

- **Independent DC Time (A):**
  Eine der EL Klemmen (üblicherweise die erste Distributed Clocks (DC) unterstützende Klemme) ist die Referenzclock, alle anderen DC-Klemmen werden dieser nachgeregelt. Auswahl der Referenzclock im Dialog darüber.
- **DC Time controlled by TwinCAT (B):**
  Die DC-Referenzclock wird der lokalen TwinCAT-Zeit nachgeregelt. Diese Einstellung wird benutzt, wenn auf einer Steuerung mehrere EtherCAT-Systeme jeweils mit Distributed-Clocks-Funktion betrieben werden. Dieser Nachführungsmodus ist jedoch von verringerter Genauigkeit.
  Bei der Anforderung von hoher Genauigkeit muss der externe EtherCAT-Verteiler CU2508 benutzt werden.
  Hinweis: als ReferenceClock in dem unterlagerten EtherCAT-System muss ein Gerät ohne Firmware-Intelligenz (z. B. ein Koppler Ek1100) gesetzt werden.
  Bitte das Kapitel "Kopplung con EtherCAT Systemen [▶ 161]" beachten.
- **DC Time controlled by External Sync Device (C)**:
  Wenn das EtherCAT-System einer übergeordneten Uhr nachgeregelt werden soll, kann hier das External Sync Device ausgewählt werden.
  Bitte das Kapitel "Externe Synchronisierung" [▶ 282] beachten.

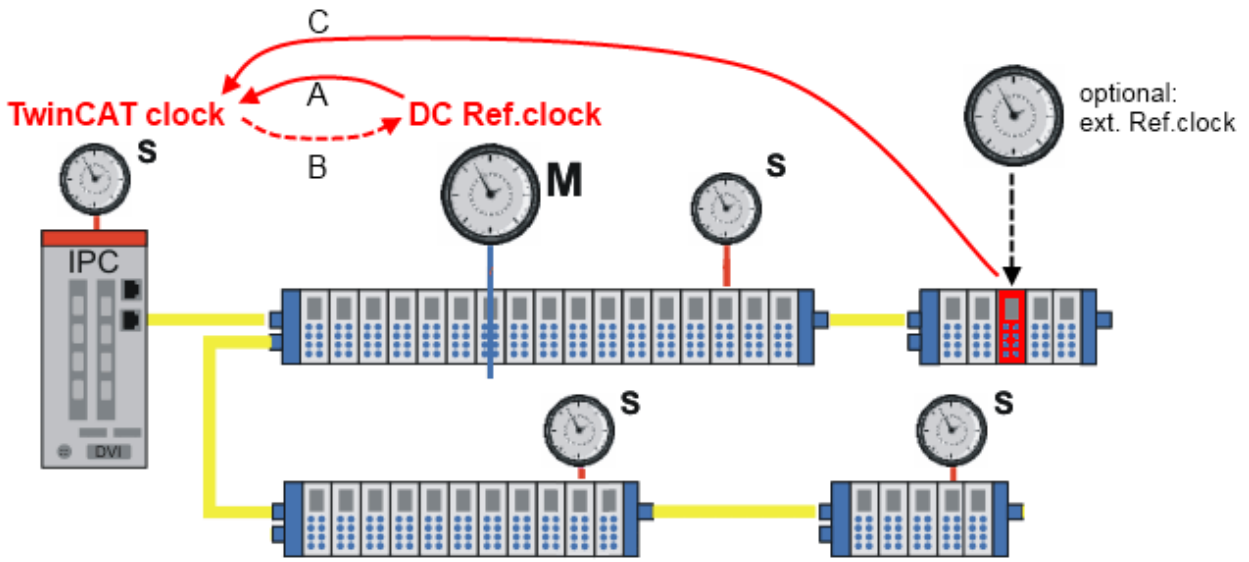


Abb. 252: Synchronisierungsrichtung

Einstellungen Prozessdaten

TwinCAT 2.11 kann in den EtherCAT-Master-Infodaten die aktuellen Offsets in [ns] anzuzeigen.

- Diese Offsets werden nach dem EtherCAT-Start einmalig berechnet.
- Die Synchronisationsregelung hält diese Offsets konstant.
- Sollen auf dem aufsynchronisierten EtherCAT-System lokale DC-Zeitwerte (z. B. aus Zeitstempelklemmen EL1252) in den absoluten Bezug des übergeordneten EtherCAT-Systems gesetzt werden, muss der Anwender diesen Offset mit jedem lokalen Zeitstempel verrechnen.

Beispiel: $t_{EL1252\ timestamp\ channel\ 1,\ absolute\ time} = t_{EL1252\ timestamp\ channel\ 1,\ local\ DC\ time} + t_{ExtToDcOffset} + t_{TcToDcOffset}$

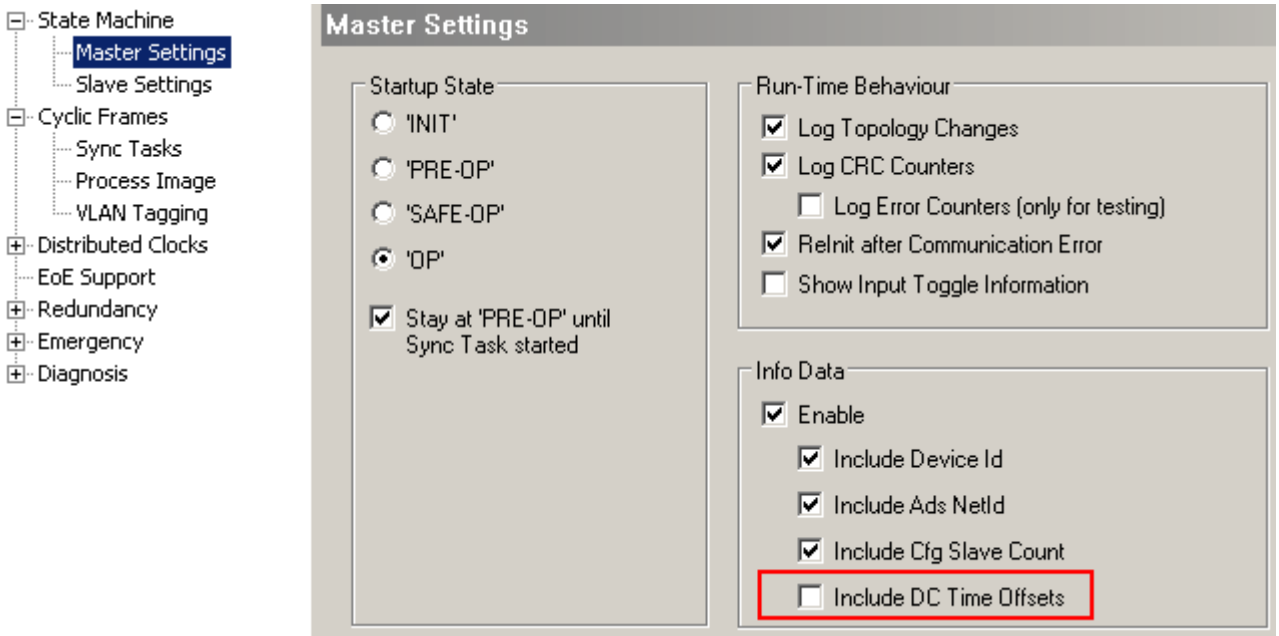


Abb. 253: Anzeige aktueller Offsets

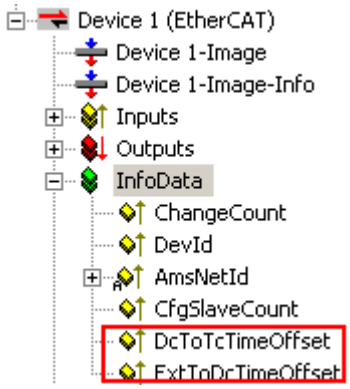


Abb. 254: Aktuelle Offsets

7.1.3 Beispielprogramme

Verwendung der Beispielprogramme

Dieses Dokument enthält exemplarische Anwendungen unserer Produkte für bestimmte Einsatzbereiche. Die hier dargestellten Anwendungshinweise beruhen auf den typischen Eigenschaften unserer Produkte und haben ausschließlich Beispielcharakter. Die mit diesem Dokument vermittelten Hinweise beziehen sich ausdrücklich nicht auf spezifische Anwendungsfälle, daher liegt es in der Verantwortung des Anwenders zu prüfen und zu entscheiden, ob das Produkt für den Einsatz in einem bestimmten Anwendungsbereich geeignet ist. Wir übernehmen keine Gewährleistung, dass der in diesem Dokument enthaltene Quellcode vollständig und richtig ist. Wir behalten uns jederzeit eine Änderung der Inhalte dieses Dokuments vor und übernehmen keine Haftung für Irrtümer und fehlenden Angaben.

Beispiel 1: Anzeige und Auswertung der verschiedenen Zeiten in TwinCAT

Das Beispielprogramm ermittelt mehrere unabhängige lokale Zeiten in einem TwinCAT System unter Windows XPe, berechnet aktuelle Abweichungen und rechnet sie in verschiedene Darstellungen um. Die Funktion *Nt_SetTimeToRtcTime* kann testweise aktiviert werden.

Hinweise:

- verwendete Zykluszeit: 1 ms
- ermittelte Zeiten:
  - lokale Windows NT Zeit (Anzeige in der Taskleiste)
  - lokale TwinCAT Zeit
  - Distributed Clocks Zeit
- im Beispielaufbau werden EtherCAT-Distributed Clocks Klemmen verwendet, um die Distributed Clocks-Zeit (DC) ermitteln zu können.
- die einzelnen Umrechnungen, insbesondere die zyklischen String-Darstellungen, benötigen signifikante Rechenzeit, zum Test des Beispielprogramms wird eine Plattform ab CX1000 empfohlen.

Beachten Sie die allgemeinen Hinweise zur EtherCAT Synchronisation.

Starten des Beispielprogramms

Die Applikationsbeispiele sind mit einem Prüfaufbau getestet und entsprechend beschrieben worden. Etwaige Abweichungen bei der Einrichtung an realen Applikationen sind möglich.

Für den Prüfaufbau wurde folgende Hardware und Software verwendet:

- TwinCAT-Master-PC mit Betriebssystem Windows XP Professional SP 3, TwinCAT Version 2.10 (Build 1330) und INTEL PRO/100 VE Ethernet-Adapter
- Beckhoff EtherCAT Koppler EK1100, Klemmen EL2202-0100, EL2252 und EL9011

7.1.3.1 Beispielprogramm TwinCAT 2

https://infosys.beckhoff.com/content/1031/ethercatsystem/Resources/zip/2469153803.zip

Vorbereitungen zum Starten des Programms

- Nach Klick auf den Download-Button speichern Sie das Zip-Archiv lokal auf ihrer Festplatte und entpacken die *.TSM (Konfigurationsdatei) und *.PRO (PLC-Programmdatei) in einem temporären Arbeitsordner.
- Die *.pro-Datei kann per Doppelklick geöffnet werden oder über die TwinCAT PLC Control Anwendung mit die Menüauswahl „Datei/ Öffnen“. Die *.tsm-Datei ist für den der TwinCAT-System Manager vorgesehen (um hier Konfigurationen einzusehen oder zu übernehmen).
- Dieses Beispiel erfordert eine PLC Steuerung mit einer Klemme EL2202-0100. Sie können entweder einen embedded PC verwenden, an dem die Klemme rechtsseitig angebracht wird, oder einen IPC mit einer EtherCAT-Verbindung eines z.B. RJ-45 Anschlusses zum EK1100 Koppler mit der Klemme (z.B. C6915 + EK1100 + EL2202-0100).
- In diesem Beispiel ist es ist nicht erforderlich die Klemme ausgangsseitig zu beschalten (da lediglich von der DC-Betriebsart Gebrauch gemacht wird). Dennoch wird für den Sync Master ein Link zu einer Variable benötigt. Die externe Variable „bDummyOut“ ist daher dafür vorgesehen, sie mit einen von den beiden Kanälen der Klemme zu verknüpfen.

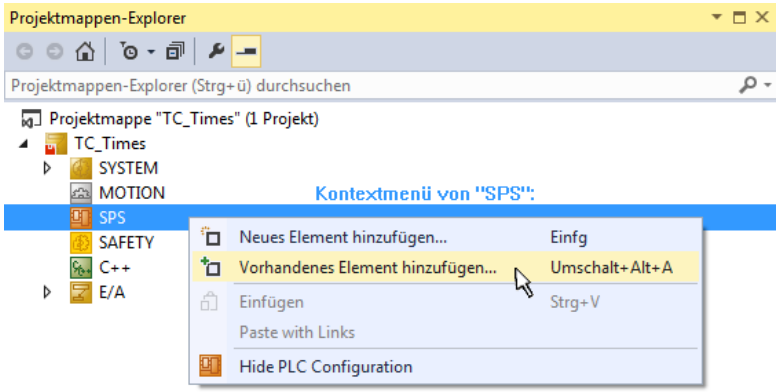
Für die weitere Vorgehensweise sehen Sie bitte das Kapitel TwinCAT Quickstart, TwinCAT 2 [▶ 66].

7.1.3.2 Beispielprogramm TwinCAT 3

https://infosys.beckhoff.com/content/1031/ethercatsystem/Resources/zip/3722257291.zip

Vorbereitungen zum Starten des Beispielprogramms (tpzip - Datei/ TwinCAT 3)

- Nach Klick auf den Download-Button speichern Sie das Zip-Archiv lokal auf ihrer Festplatte und entpacken die *.tpzip -Archivdatei in einem temporären Arbeitsordner.
- Erstellen Sie ein neues TwinCAT Projekt wie im Kapitel
  TwinCAT Quickstart, TwinCAT 3, Startup [▶ 76]
  beschrieben.
- Öffnen Sie das Kontextmenü von „SPS“ im „Projektmappen-Explorer“ und wählen „Vorhandenes Element hinzufügen...“:



- Wählen Sie die zuvor entpackte .tpzip Datei (Beispielprogramm) aus.
- Dieses Beispiel erfordert eine PLC Steuerung mit einer Klemme EL2202-0100. Sie können entweder einen embedded PC verwenden, an dem die Klemme rechtsseitig angebracht wird, oder einen IPC mit einer EtherCAT-Verbindung eines z.B. RJ-45 Anschlusses zum EK1100 Koppler mit der Klemme (z.B. C6915 + EK1100 + EL2202-0100).

- In diesem Beispiel ist es ist nicht erforderlich die Klemme ausgangsseitig zu beschalten (da lediglich von der DC-Betriebsart Gebrauch gemacht wird). Dennoch wird für den Sync Master ein Link zu einer Variable benötigt. Die externe Variable „bDummyOut“ ist daher dafür vorgesehen, sie mit einen von den beiden Kanälen der Klemme zu verknüpfen.

Sehen Sie hierzu auch weitere Hinweise in dem Kapitel:
Inbetreibnahme, TwinCAT Quickstart, TwinCAT 3, Startup [▶ 76].

7.2 Grundlagen

7.2.1 EtherCAT Distributed Clocks

Im Folgenden wird eine Einführung in die Distributed Clocks Technologie als Feature des EtherCAT-Protokolls gegeben. In diesem Abschnitt wird ein anwendernaher Überblick mit den prinzipiellen Aspekten aufgeführt, der für die übliche Anwendung im EtherCAT-System ausreichend sein sollte. Im nachfolgenden Abschnitt werden für den interessierten Anwender Interna und genauere Beschreibungen geliefert, für den üblichen Betrieb eines EtherCAT-Slave sind diese Kenntnisse nicht erforderlich.

Inhalt dieser Dokumentation

Diese Einführung bleibt auf die anwendungsrelevante Funktionsbeschreibung beschränkt. Weitere und ausführlichere Informationen zu EtherCAT im Allgemeinen und Distributed Clocks im Besonderen sind unter http://www.ethercat.org/verfügbar.

Wichtige neueingeführte Begriffe sind im folgenden kursiv gedruckt.

Die folgende Seite besteht aus den Abschnitten

- Prinzip der Distributed Clocks [▶ 234]
- Nutzbare Zusatzfunktionen des ESC [▶ 238]
- Anwendung der Distributed Clocks und Synchronität mit der Steuerung im PC

Prinzip der Distributed Clocks

Der Begriff "Distributed Clocks" bezeichnet in der EtherCAT-Terminologie einen logischen Verbund aus verteilten Uhren. Mit den Distributed Clocks ist es bei dem Echtzeit-Ethernet-Protokoll EtherCAT möglich, in allen Busteilnehmern lokal eine in einem sehr engen Bereich gleiche Uhrzeit vorzuhalten. Falls ein *EtherCAT-Slave* die Distributed Clocks-Funktionalität unterstützt, beinhaltet er eine eigene Uhr, die nach dem Einschalten zunächst lokal arbeitet, basierend auf einem eigenen Taktgeber im EtherCAT-Slave (Quarz, Oszillator, ...). Im EtherCAT-Strang existiert ein ausgewählter EtherCAT-Slave, der die Referenzuhr/*Reference Clock* (M, siehe Abb. *Distributed Clocks im EtherCAT-System*) darstellt, auf die sich die Slave Clocks (S) der anderen Teilnehmer und der Steuerung synchronisieren. Diese Reference Clock stellt somit die Systemzeit/*System Time* dar. Diese Abstimmung und Synchronisierung wird automatisch und fortlaufend vom *EtherCAT-Master* vorgenommen, wenn dieser die Distributed Clocks Funktionalität unterstützt - wie z. B. der Beckhoff TwinCAT EtherCAT-Master. Dazu sendet der EtherCAT-Master in kurzen Abständen (so häufig, dass die Slave-Clocks innerhalb der spezifizierten Grenzen nicht auseinander laufen) ein spezielles *EtherCAT-Datagramm*, in das der EtherCAT-Slave mit der Reference Clock seine aktuelle Uhrzeit einträgt. Diese Information wird dann von allen anderen EtherCAT-Slaves mit Slave-Clock aus demselben umlaufenden Datagramm gelesen. Dies ist auf Grund der Ringstruktur von EtherCAT möglich, wenn die Reference Clock topologisch *vor* allen anderen Slave-Clocks angeordnet ist. Deshalb wird standardmäßig der erste Distributed Clocks fähige EtherCAT-Slave vom EtherCAT-Master als Reference Clock ausgewählt.

In einer EtherCAT-Konfiguration gibt es also einen EtherCAT-Master, der den Bus mit den angeschlossenen EtherCAT-Slaves betreibt und verwaltet. Von diesen EtherCAT-Slaves beinhaltet *ein* Teilnehmer die Reference Clock, alle anderen EtherCAT-Teilnehmer (also auch der EtherCAT-Master) stellen Slave-Clocks dar.

Das Handling der EtherCAT-Kommunikation und insbesondere der Distributed Clocks Funktionalität in einem EtherCAT-Slave übernimmt der *EtherCAT-Slave-Controller (ESC)*. Dies ist ein elektronisches Bauteil (Chip), das als ASIC oder programmierbares FPGA o.ä. ausgeführt sein kann. Jeder EtherCAT-Slave verfügt über solch einen ESC, damit zyklische und azyklische Prozessdaten vom Master mit dem Slave über den

EtherCAT-Feldbus ausgetauscht werden können. Dieser ESC kann direkt einfache Funktionen wie digitale Ein-/Ausgänge verwalten, er kann aber auch über serielle/parallele Interfaces an einen weiteren Prozessor im EtherCAT-Slave angeschlossen werden, der komplexere Aufgabe wie z. B. eine Antriebsregelung übernimmt. Insbesondere aber verwaltet der ESC die lokale Distributed Clocks Funktionalität mit den zugehörigen Aktionen, wenn der EtherCAT-Slave dieses Feature unterstützen soll.

In Abb. *Beispielhafter Funktionsumfang eines ESC und seine Einbettung in einen typischen EtherCAT-Slave* sei das Schema eines EtherCAT-Slaves dargestellt. Dazu die Einbettung des ESC in diesen mit einer Auswahl seiner Grundfunktionen.

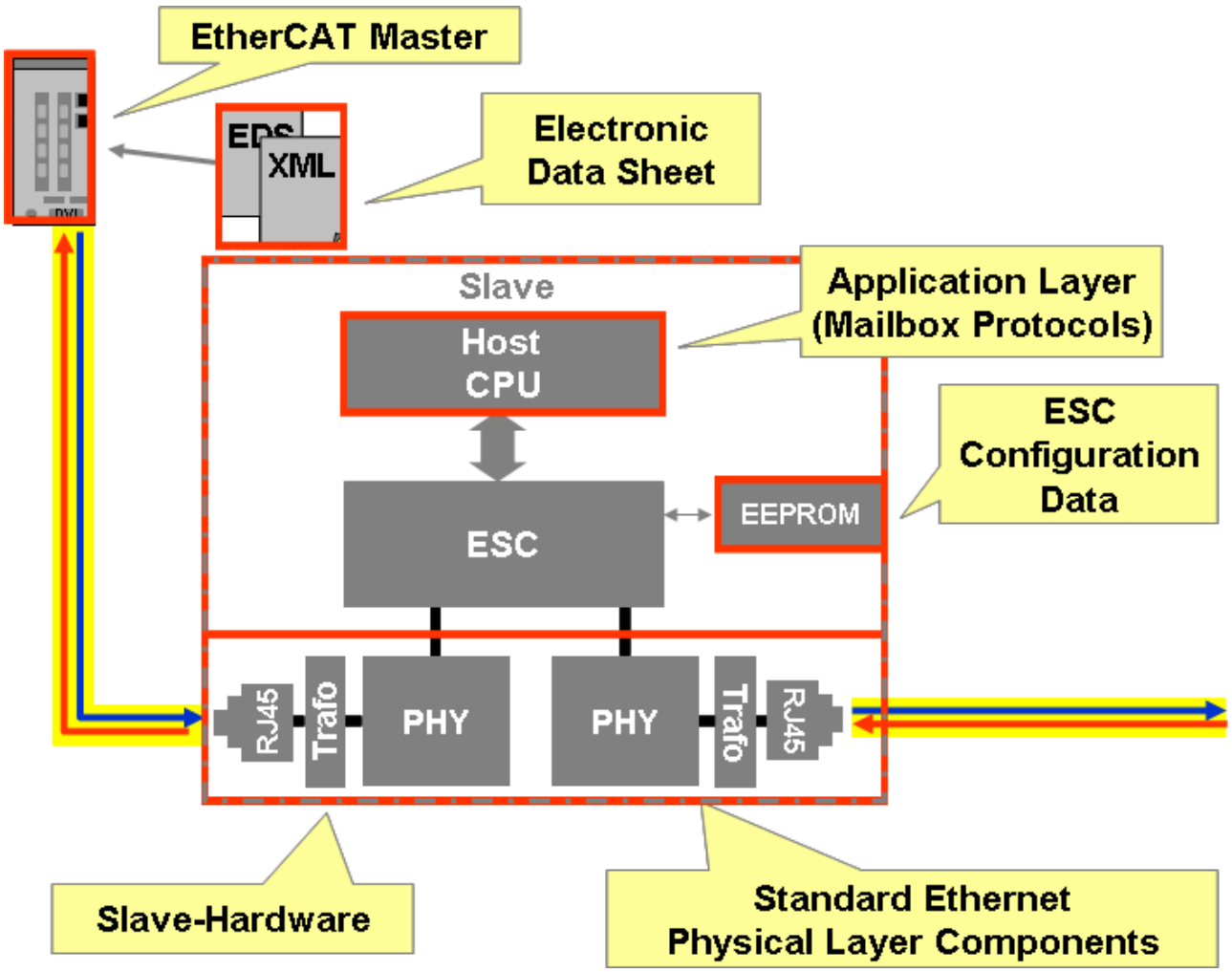


Abb. 255: Beispielhafter Funktionsumfang eines ESC und seine Einbettung in einen typischen EtherCAT-Slave

Von der RJ45-Buchse werden die elektrischen Signale über den Trafo zum PHY (PHYsikalische Schnittstelle) geleitet. Er extrahiert aus dem codierten Ethernet-Signal die Nutzdaten und leitet sie dem ESC zur Verarbeitung weiter. Minimal verzögert (da im Durchlauf verarbeitet) wird das EtherCAT-Telegramm dann wieder über PHY und Buchse zum nächsten EtherCAT-Slave weitergeleitet. Der ESC parametriert sich beim Slave-Start selbsttätig mit Konfigurationsdaten aus einem EEPROM. Falls eine weitere CPU im Slave existiert, kann er mit dieser über Schnittstellen kommunizieren.

Folgende Features bietet die Distributed Clocks Einheit des ESC im Vollausbau (geräteimplementierungsabhängig):

- Clock Synchronisierung zwischen den EtherCAT-Slaves und dem Master
- synchrone Erzeugung von Output Signalen (Sync Signale)
- synchrones Einlesen von Input Signalen
- präzise Zeitstempelung von Eingangssignalen (Latch Signale)
- Erzeugung synchroner Interrupts

In Abb. *Distributed Clocks im EtherCAT-System* ist die Reference Clock (M) der erste Slave im EtherCAT-Strang mit Distributed Clocks Funktionalität nach dem EtherCAT-Master-IPC. Da jeder Slave auf dem Hin- und Rückweg eine geringe Verzögerung – sowohl im Teilnehmer (S) selbst als auch durch die dazwischen liegende Übertragungsstrecke – verursacht, sind die Laufzeiten (Δt) zwischen Reference Clock und jeweiliger Slave Clock bei der Synchronisierung der Slave Clocks zu berücksichtigen. Es ist also nicht zweckmäßig, einfach die lokale Uhrzeit der Reference Clock in alle nachfolgenden Slave-Uhren zu kopieren, sondern für jeden Slave ist ein eigener Offset-Wert in Abhängigkeit von mehreren Parametern zu berechnen.

Zur Messung der Offset-Zeiten sendet der EtherCAT-Master in der Startphase ein Broadcast-Read-Datagramm auf eine spezielle Adresse in allen ESC's, womit jeder Slave veranlasst wird, den Empfangszeitpunkt des Telegramms (bezüglich seiner lokalen Uhr) sowohl auf dem Hin- als auch auf dem Rückweg zu speichern. Diese gespeicherten Zeitpunkte werden dann vom Master eingelesen und entsprechend verrechnet. Diese Messzyklen finden für alle EtherCAT-Slaves mehrmals statt. Dadurch kann der EtherCAT-Master ein sehr exaktes Abbild der Topologie erstellen, bezogen auf die Frame-Verzögerungen zwischen den EtherCAT-Slaves.

Mit den bisher beschriebenen Aktionen ist ein synchroner Betrieb aller Distributed Clocks im EtherCAT-Verbund gegeben, z. B. in einer Produktionsanlage - hochgenaue relative Zeitangaben sind damit innerhalb der Applikation möglich. Der absolute Bezug zur globalen Realität wird über die *Master Clock* hergestellt - sie trägt i.d.R. eine Ausprägung der globalen Weltzeit (DCF77, GPS, Internetzeitserver, ...) oder eine andere, als systemübergreifend gültig bezeichnete Zeit (Netzwerkserver, PC-Uhr, BIOS-Uhr, IEEE1588, ...). So startet ein Beckhoff TwinCAT EtherCAT-Master mit den Angaben der lokalen PC-Uhr als Master Clock, um damit die Reference Clock zu initialisieren. Sowohl beim Systemstart als auch im weiteren Betrieb ist allerdings die Nachregelung der Reference Clock (M) nach einer Master Clock möglich, entweder über direkten Kontakt der Master Clock zur Steuerung (z. B. Netzwerkserver) oder über Einspeisung in einen speziellen EtherCAT-Slave (z. B. Beckhoff EL6692, EtherCAT-Bridge-Klemme).
Diese übergeordnete Master Clock verletzt nicht das Primat der Reference Clock, da die kurzfristige und für Echtzeitsteuerungen kritische Synchronisierung im sub-Millisekundenbereich allein von der Reference Clock bestimmt wird - die Synchronisierung der gesamten EtherCAT-Applikation inkl. Steuerungs-PC mit der Master Clock verläuft dagegen in längeren Intervallen.

Folgende Effekte müssen somit von der Distributed-Clocks-Regelung im EtherCAT-Master berücksichtigt werden:

- Offset-Kompensation jedes Slaves zur Reference Clock - nach dem Systemstart beginnen die lokalen Uhren ggf. mit unterschiedlichen Startwerten zu arbeiten.
- Offset-Kompensation der Reference Clock zur Master Clock - berücksichtigt beim Aufstart des Systems.
- Propagation Delay Measurement/Messung der Offset-Zeiten - abhängig von der Teilnehmeranzahl, Kabellängen, dynamischen Konfigurationsänderungen, usw.
- Drift compensation/Driftkorrektur - da jede Slave-Clock üblicherweise ihre eigene Taktquelle besitzt (Quarz, PLL, ...), bleiben Offset-Zeiten über einen längeren Zeitraum (Minuten, Tage) nicht konstant. Die Driftkorrektur fängt diesen Missstand ab.

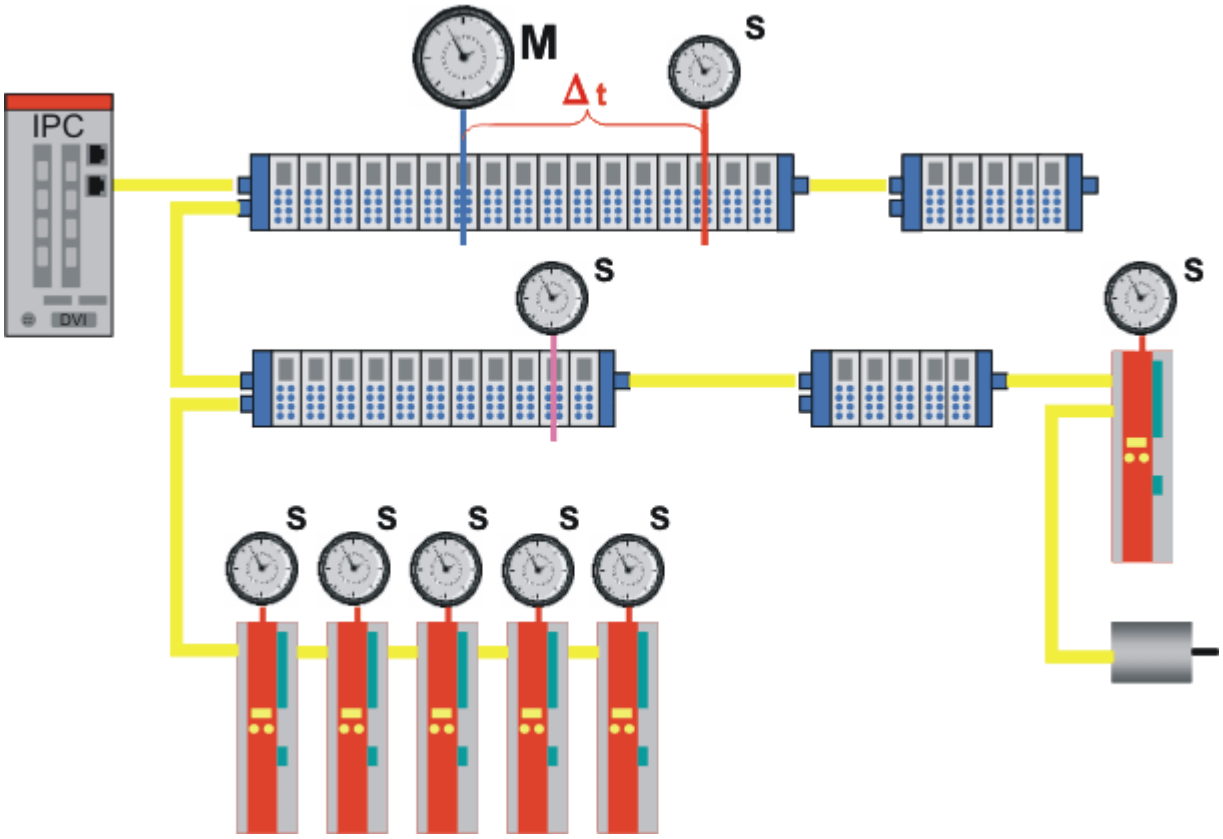


Abb. 256: Distributed Clocks im EtherCAT-System

Wenn bei einer Busunterbrechnung ein EtherCAT-Slave nicht mehr mit Synchronisierungsdatagrammen versorgt wird, kann er dennoch weiterhin durch seine lokale Uhr alle Aufgaben erfüllen. Die Distributed Clocks Regelung erfolgt ruckfrei - im Normalbetrieb und auch beim Wiedereinsetzen des EtherCAT-Verkehrs.

Zusammenfassung Technische Daten

Die Distributed Clock Funktion im EtherCAT-Slave-Controller (ESC) hat folgenden Eigenschaften:

- Einheit *1 ns*
- universaler Nullpunkt *1.1.2000 00:00*
- Umfang bis zu *64 Bit* (ausreichend für 584 Jahre); manche EtherCAT-Slaves unterstützen jedoch nur einen Umfang von 32 Bit, d.h. nach ca. 4,2 Sekunden läuft das Register lokal über und beginnt wieder bei 0.
- Jede lokale Uhr wird vom EtherCAT-Master automatisch mit der Reference Clock im EtherCAT-System mit einer Genauigkeit < 100 ns synchronisiert, unabhängig von der Entfernung zwischen einzelnen EtherCAT-Slaves
- Beim Start des EtherCAT-Systems übernimmt der EtherCAT-Master i.d.R. die aktuelle Uhrzeit von einer Master Clock, z. B. der hardwarebasierten BIOS-Uhr des eigenen PCs. Dadurch ist der Zeitbezug zur aktuellen Weltzeit hergestellt. Mit dieser Zeit wird beim EtherCAT-Start die gewählte Reference Clock geladen, die diese Zeit dann i.d.R. selbsttätig auf der Basis ihres lokalen Taktgebers fortführt. Eine fortlaufende Synchronisierung der Reference Clock mit der Master Clock ist aber möglich.
- Die effektive Schrittweite der lokalen Uhr beträgt üblicherweise 10 ns - die verbleibende Stelle ("Einer") wird zur Regelung der Distributed Clocks benutzt.

Bis jetzt wurde die Distributed Clocks Funktion im ESC ohne Interaktion zur Umgebung betrachtet. Auf der Basis dieser lokalen/globalen Uhrzeit können nun aber zusätzliche Funktionen im EtherCAT-Slave realisiert werden.

Nutzbare Zusatzfunktionen des ESC

Die hochgenau synchronisierte Distributed Clocks Zeit wird vom ESC genutzt, mittels einer Capture/Compare-Einheit auf vorgebbare Zeiten oder Signale von außerhalb des ESC zu reagieren. Wie sich der ESC verhält, wird während des Startups durch die Konfiguration des EtherCAT-Slaves definiert und ist vom Anwender i.d.R. nicht veränderbar.

Aktion nach Zeitvorgabe: Compare - Sync0/1

Die Distributed Clock Einheit im ESC verfügt i.d.R. über 2 Interrupts, die zeitgesteuert ausgelöst werden können. Diese Interrupts heißen *SYNC0* und *SYNC1*. In diesem Fall wäre die Compare-Einheit im ESC aktiv: stimmt die lokale Distributed Clocks Zeit mit einer vom Anwender definierten Vorgabezeit überein, löst der ESC einen Interrupt und die damit verbundenen Vorgänge aus. Dabei kann diese Vorgabezeit *einmalig* gesetzt werden, was dann auch eine *einmalige* Aktion im ESC zur Folge hat - Verwendung z. B. bei Beckhoff Timestamp Klemmen.
Der ESC kann aber auch selbsttätig neue Vorgabewerte nachladen, was in diesem Fall eine zyklische Abfolge von ESC-Aktionen zur Folge hat - Verwendung z. B. bei Beckhoff Oversampling Klemmen. Durch die beim ESC-Start z. B. aus einem Slave-eigenen EEPROM geladene Konfigurationsdaten ist parametrierbar, welche Aktion ein ESC beim Auftreten eines SYNC0/SYNC1-Signals letztendlich durchführt. Beispielsweise kann er dann Ausgangsdaten schreiben, Eingangsdaten einlesen oder die Kommunikation mit einem angeschlossenen Microcontroller beginnen.

Reaktion auf externes Signal: Capture - Latch 0/1

Wird ein ESC entsprechend konfiguriert, kann er beim Eintreten eines externen Ereignisses die aktuelle lokale Uhrzeit speichern, sie also mit einer Capture-Einheit ohne Zeitverzug in einen Puffer legen. Solche externen Ereignisse können sein: Ankunft des EtherCAT-Frames, Ende des EtherCAT-Frames, Flanke an einem dezidierten Pin des ESC, Kommunikation mit einem angeschlossenen Microcontroller und noch viele andere mehr.

Anschluss an eine externe Logik - SPI/µC parallel/IO/IRQ

Ein ESC ist nicht nur als Stand-Alone-Einheit zu verwenden, sondern verfügt über Schnittstellen, um mit anderen elektronischen Einheiten zu kommunizieren. Das kann z. B. ein Controller sein, der einen Leistungsantrieb regelt, oder eine Auswertungselektronik eines Drehgebers, s. Abb. *Beispielhafter Funktionsumfang eines ESC und seine Einbettung in einen typischen EtherCAT-Slave*. Auch die Kommunikation über diese Schnittstellen kann Distributed Clocks gesteuert erfolgen. Damit ist z. B. die Positionsabfrage an eine Drehgeberelektronik zeitlich im ns-Bereich äquidistant.

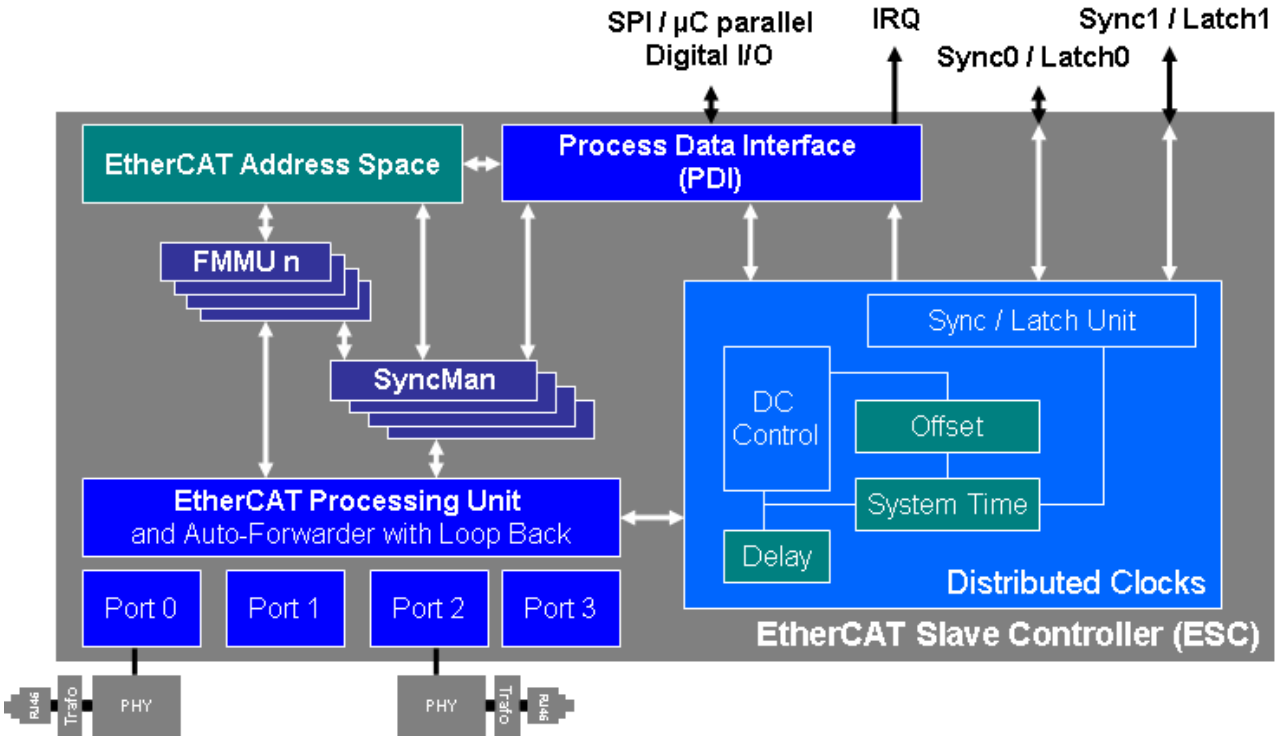


Abb. 257: Schnittstellen der Distributed Clocks Unit

In Abb. *Schnittstellen der Distributed Clocks Unit* ist die Distributed Clocks Unit mit Ihren Schnittstellen und dem Zusammenwirken mit dem EtherCAT Bus schematisiert.

Anwendung der Distributed Clocks und Synchronität mit der Steuerung im PC

Durch den Distributed Clocks Verbund werden alle EtherCAT-Slaves, die dieses Feature unterstützen, mit einer Abweichung von <100 ns synchron betrieben. An einer Beispielanwendung mit Eingangskanälen soll dieses Prinzip verdeutlicht werden. Auf die Anwendung auf Ausgangskanäle wird im Anschluss eingegangen.

Im Folgenden wird an einem Beispiel mit drei Beckhoff EL1202-0010 die synchrone Erzeugung von SYNC-Signalen [▶ 238] zum gleichzeitigen Lesen der Eingänge beschrieben. Andere Funktionen des ESC (Latch-Signale [▶ 238], Anschluß externer Logik) werden ausführlich in den entsprechenden Slave Dokumentationen beschrieben.

Distributed Clocks mit Eingangskanälen

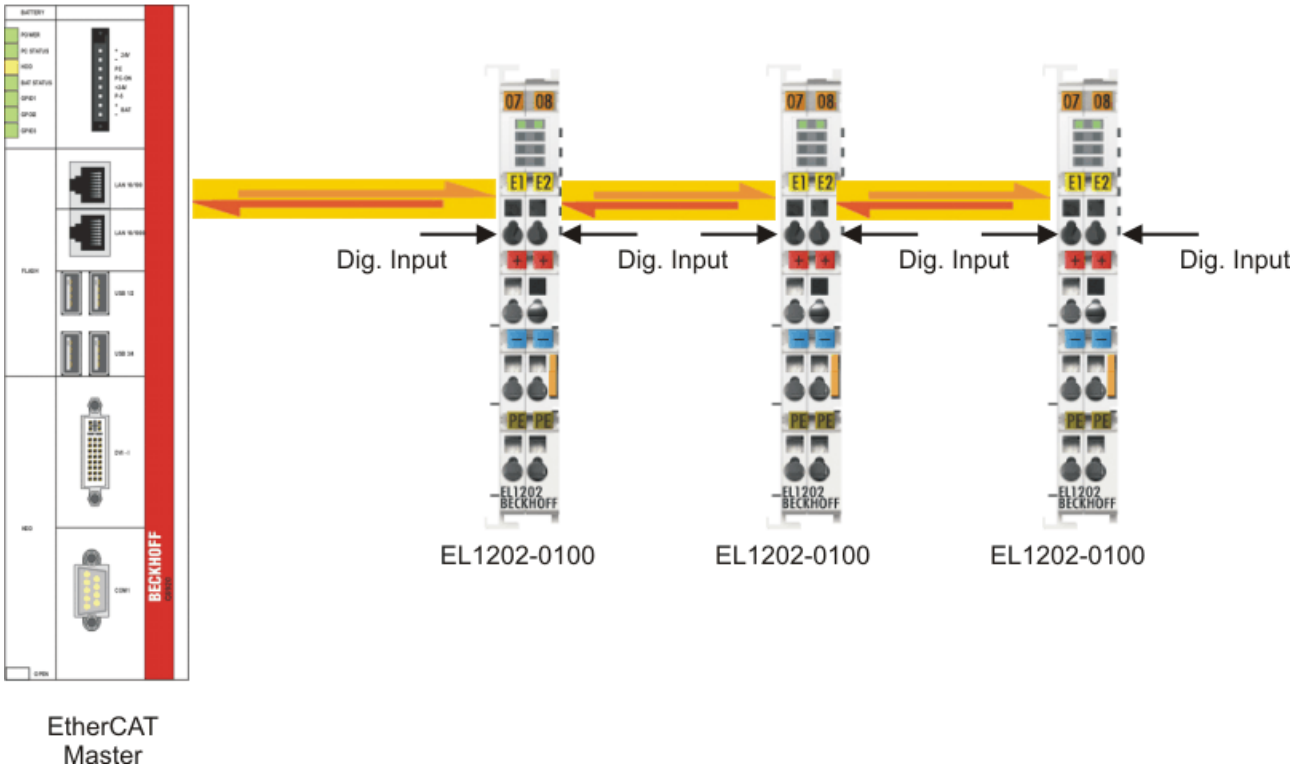


Abb. 258: Beispielanwendung Distributed Clocks, Zykluszeit 100 µs

In Abb. *Beispielanwendung Distributed Clocks, Zykluszeit 100 µs* betreibt der Master PC mit seiner PLC an einem EtherCAT-Master eine EtherCAT-Konfiguration mit 3 digitalen zweikanaligen Eingangsklemmen EL1202-0100 mit Distributed Clocks Unterstützung (die Distributed Clocks-Unterstützung wird durch Umstellung der EL1202 auf die EL1202-0100 im System Manager erreicht, siehe die entsprechende Dokumentation). Welche EtherCAT-Slaves noch eingesetzt werden und welche Entfernungen zwischen den Slaves liegen, spielt dabei keine Rolle. Die steuernde PLC und damit der EtherCAT-Feldbus werden mit 1 ms Zykluszeit betrieben, das heißt alle 1 ms werden die Eingangsklemmen EL1202-0100 nach ihren Eingängen abgefragt. In der EL1202-0100 wird nun die lokale Distributed Clock zum Samplen der beide Eingangskanäle verwendet: mit jedem SYNC-Interrupt liest der ESC direkt den anliegenden Eingangswert (0 oder 1) ein. Passiert dann kurz darauf der EtherCAT-Frame die Klemmen, legt jede EL1202-0100 ihre beiden Bits an der vorgesehenen Stelle im Frame ab. Durch die Verwendung der Distributed Clock erfolgt so das Samplen der Eingänge hochkonstant im fortlaufend gleichen Abstand mit einer Abweichung von < 100 ns. Außerdem lesen *alle* EL1202-0100 im gesamten EtherCAT-Verbund in der Standardeinstellung ihre Eingänge zum *gleichen* globalen Zeitpunkt ein, unabhängig von ihrer Anordnung. In einem Zeitdiagramm sieht das wie folgt aus:

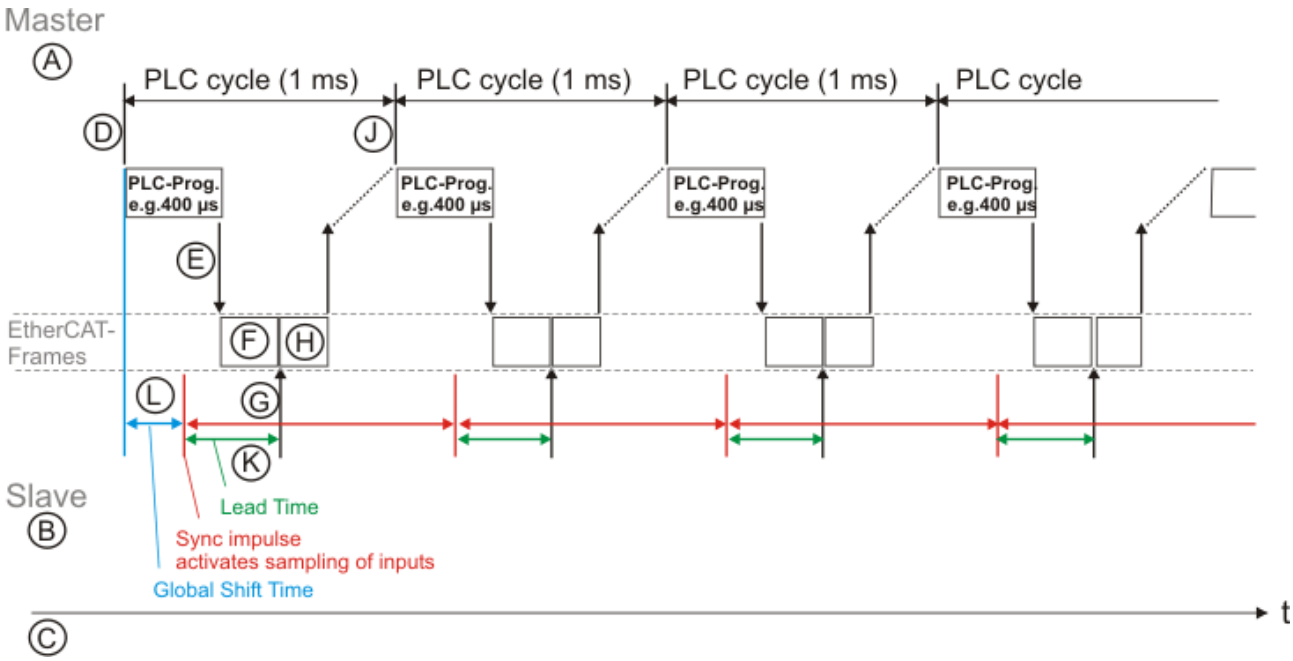


Abb. 259: Zeitdiagramm Einlesen der EL1202-0100

In Abb. *Zeitdiagramm Einlesen der EL1202-0100* ist der Ablauf von 4 kompletten I/O-Zyklen am Beispiel einer EL1202-0100 (vgl. Abb. *Beispielanwendung Distributed Clocks, Zykluszeit 100 µs*) dargestellt. Die Vorgänge während eines solchen Zyklus' lassen sich wie folgt aufschlüsseln:

- **A, B:** Abb. *Zeitdiagramm Einlesen der EL1202-0100* ist grau unterteilt in eine Master- und eine Slave-Seite - dazwischen bewegen sich als Transportschicht die Ethernet-Frames.
- **C:** die nach rechts verlaufende Zeitachse ermöglicht eine zeitliche Zuordnung des Ablaufs der beschriebenen Aktionen.
- **D:** Die PC-eigene Echtzeituhr startet einen neuen Verarbeitungszyklus, die Zykluszeit sei 1 ms. Die Berechnung dauere hier 400 µs. Aus dem vorliegenden Eingangsprozessabbild wird durch den vorgegebenen Programmcode das Ausgangsprozessabbild berechnet.
- **E:** Nach der Berechnung stehen neue Ausgabedaten für den Feldbus zur Verfügung. Der EtherCAT-Frame (bzw. die EtherCAT-Frames, wenn mehrere) wird mit den Prozessdaten abgeschickt. Ein Ethernet-Frame ist je nach Datenumfang zwischen 7 und 128 µs lang. Es dauert also eine gewisse Zeit, bis er komplett auf dem Feldbus unterwegs ist, alle Slaves durchlaufen hat und dann wieder vollständig von der Netzwerkkarte empfangen wurde.
- **F:** Der EtherCAT-Frame durchläuft nun alle EtherCAT-Slaves, die vor der betrachteten EL1202-0100 liegen.
- **G:** Abhängig von der Anzahl der EtherCAT-Slaves, kommt nach einer gewissen Verzögerung von einigen µs der Frame an der EL1202-0100 vorbei, um die Eingangsdaten abzuholen (genauer: er wird vom ESC im Durchlauf verarbeitet). Diese Daten müssen nun bereits abholbereit zur Verfügung stehen.
- **H:** danach durchläuft der Frame noch alle nachfolgenden Slaves bis er wieder komplett vom Netzwerkport des PCs empfangen wurde.
- **J:** wird der nächste PLC-Berechnungszyklus von der Echtzeituhr gestartet, stehen somit aktuelle Eingangsdaten aus dem Feldbus zur Verfügung.

Es folgen nähere Erläuterungen zu den Vorgängen in der beispielhaften EL1202-0100:

- **K:** Damit die bei (G) in den Frame übertragenen Eingangsdaten rechtzeitig im ESC bereitstehen um in den Frame zu gelangen, müssen durch eine sinnvolle *Vorlaufzeit (Lead Time, grün)* vor der erwarteten Frame-Passage die Eingänge gelesen werden. Ansonsten würden ggf. veraltete Daten in den Frame eingeblendet. Das Lesen der physikalischen Eingänge wird von der Distributed Clocks Unit im ESC durch das SYNC-Signal ausgelöst.
  Diese Zeit kann als Sicherheitsabstand gewertet werden: je kürzer dieser Sicherheitsabstand ist, desto aktueller sind die Eingangsdaten. Wird diese Zeit zu kurz gewählt, besteht die Möglichkeit, dass die

Eingangsdaten noch nicht zur Verfügung stehen, da der EtherCAT Frame die Klemme zu früh erreicht - es werden dann keinen neuen Prozessdaten in den Frame gekoppelt! Deshalb wird die Berechnung dieser Vorlaufzeit standardmäßig automatisch vom EtherCAT-Master durchgeführt.

- **L:** Vom Master mit seiner eigenen Echtzeituhr aus gesehen findet der klemmenlokale SYNC-Puls somit um eine *globale Shift Time* nach dem Echtzeit-Tick statt. Diese globale Shift Time wird einmalig beim EtherCAT-Aufstart berücksichtigt, denn ab dann laufen sowohl der Echtzeit-Tick im PC als auch die SYNC-Pulse in den EtherCAT-Slaves mit der konstanten Zykluszeit von (hier) 1 ms.
- Die globale Shift Time wird automatisch vom Beckhoff TwinCAT EtherCAT-Master nach den Randbedingungen wie Konfiguration, max. Programmrechendauer, Zykluszeit u.a. so berechnet, dass ein möglichst sicherer, aber dennoch aktueller Betrieb aller EtherCAT-Slaves gewährleistet wird.
- Die automatisch berechnete, globale Shift Time ist vom Anwender durch eine *manuelleShift Time* veränderbar, und zwar sowohl auf globaler Ebene für alle EtherCAT-Slaves, als auch Slave-lokal für jeden EtherCAT-Slave einzeln.

Nachführung der PC-Echtzeit

In Abb. *Zeitdiagramm Einlesen der EL1202-0100* arbeiten 2 Zeiträume nebeneinander: der PC mit seinem Echtzeit-Tick *und* das Distributed Clocks System mit seinen verteilten Uhren in den EtherCAT-Slaves und dem EtherCAT-Master (der gleichwohl im PC sitzt) - dargestellt durch die grauen Trennlinien A//B.
Der Hintergrund für die Einführung einer zweiten Zeitbasis für die EtherCAT-Slaves mit den daraus abgeleiteten SYNC-Interrupts liegt in dem Jitter, mit dem die EtherCAT-Frames die einzelnen EtherCAT-Slaves passieren. Durch den Durchlauf durch die Slaves, die evtl. variable PLC-Laufzeit und die Qualität des Echtzeit-Ticks kann der Zeitpunkt, wann ein Frame vom PC abgeschickt wird und er dann endlich einen Slave passiert, im µs-Bereich variieren. Wenn Slave-lokale Ereignisse z. B. allein durch die Passage des Kommunikationsframes gestartet werden würden, ergäben sich so gleichermaßen variante Aktionen im Slave. Dies ist für hochgenaue Anwendungen nicht tolerabel. Durch die Einführung von Slave-lokalen geregelten Uhren besitzt nun jeder entsprechende EtherCAT-Slave ein hochgenaue lokale Zeitbasis mit einer Abweichung von < 100 ns zu allen anderen Uhren, was die Qualität aller zeitbasierten Aktionen um mehrere Größenordnungen verbessert.

Durch die beiden unterschiedlichen Zeitbasen würde jedoch schon nach kurzer Zeit die "Feldzeit" (Distributed Clocks/System Time) von der "PC-Zeit" (Betriebssystem/BIOS Uhr) abweichen - und der Abstand würde fortlaufend größer.

Deshalb greift der TwinCAT EtherCAT-Master in die Uhr des PCs ein und stellt diese zyklisch der EtherCAT Reference Clock nach - so bleiben dauerhaft der Echtzeit-Tick der Steuerung und die klemmenlokalen SYNC-Interrupts synchron mit der eingestellten/berechneten globalen Shift Time (Abb. *Zeitdiagramm Einlesen der EL1202-0100*, blau)

> **Abgleich auf eine externe Master Clock**
>
> Soll ein solcherart geregeltes Automatisierungssystem auf Dauer mit einer äußeren Referenzuhr übereinstimmen (z. B. Weltzeit oder Netzwerk-lokale Zeit), sind z. B. entsprechende EtherCAT-Slaves zu verwenden, die den EtherCAT-Master über die Uhrzeit der Master Clock informieren oder die Master Clock ist direkt an der Steuerung anzuschließen. Der EtherCAT-Master unternimmt dann die entsprechenden Schritte zur Einhaltung einer dauerhaft konstant geringen Regeldifferenz zwischen Master und Reference Clock.

Berechnung der globalen Shift Time

In Abb. *Zeitdiagramm Einlesen der EL1202-0100* ist eine blaue "globale Shift Time" angeben. Deren Wert bestimmt, wann in Relation zum Gesamtsystemverhalten der ESC z. B. eine SYNC0/1-Aktion lokal im Slave startet. Die globale Shift Time setzt sich aus verschiedenen, mitunter auch vom Anwender veränderbaren Elementen zusammen, s. Abb. *Zeitliche Festlegung des Slave-lokalen SYNC-Pulses in Bezug auf den Echtzeit-Tick des Master-PCs*.

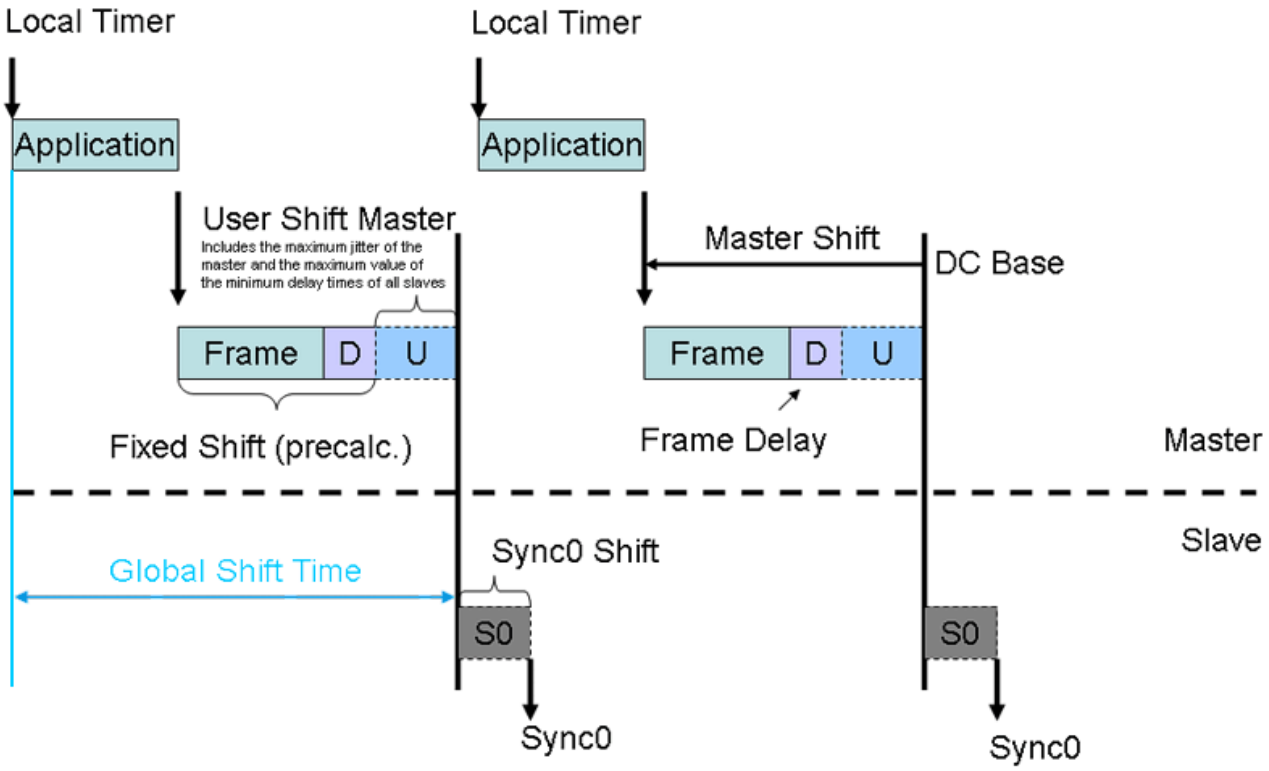


Abb. 260: Zeitliche Festlegung des Slave-lokalen SYNC-Pulses in Bezug auf den Echtzeit-Tick des Master-PCs

Die Elemente der globalen Shift Time nach Abb. *Zeitliche Festlegung des Slave-lokalen SYNC-Pulses in Bezug auf den Echtzeit-Tick des Master-PCs* sind

- **Application**
  Die maximal erwartete Berechnungszeit für den Programmcode - nicht manipulierbar.
- **Frame**
  Die Framelänge des Ethernet-Frames (7..128 µs, auch mehrere Frames) - nicht manipulierbar.
- **D**
  Summe der Verzögerungen, die durch den Durchlauf des/der Frames durch die EtherCAT-Slaves generiert werden, inklusive der Jitter-Kalkulation.
- **U**
  User Shift Master - Shift Time die vom EtherCAT-Master je nach Baugruppe mit Standardwerten vorbelegt wird - vom Anwender manipulierbar.
  Ein- und Ausgabebaugruppen haben hier unterschiedliche Werte eingetragen
- **S0**
  User Shift Time im Slave - gewöhnlich 0, wird aber bei manchen EtherCAT-Slaves bereits standardmäßig mit Werten <>0 belegt, vom Anwender manipulierbar (s. jew. Slave Dokumentation)

Da sich die o.a. Elemente zur globalen Shift Time aufaddieren lassen, kann es u.U. sinnvoll sein, in den entsprechenden Dialogen *negative* Werte einzutragen - etwa um mit einer negativen Slave Sync0 Shift Time (s. Abb. *Zeitliche Festlegung des Slave-lokalen SYNC-Pulses in Bezug auf den Echtzeit-Tick des Master-PCs*) ein früheres Einlesen von Eingangssignalen auszulösen als vom EtherCAT-Master standardmäßig vorgesehen.

HINWEIS

Achtung! Keine Plausibilitätskontrolle!

Die aufgeführten Hinweise und Erläuterungen sollten mit Bedacht angewendet werden! Die genannten Einstellungen werden vom EtherCAT-Master automatisch mit Werten belegt, die eine zuverlässige und aktuelle Prozessdatenerfassung unterstützen.
Anwenderseitige Eingriffe an dieser Stelle können zu unerwünschtem Verhalten führen!
Bei der Manipulation dieser Einstellungen im Beckhoff TwinCAT System Manager wird softwareseitig keine Plausibilitätskontrolle durchgeführt!
Eine korrekte Funktion der EtherCAT-Slaves in allen denkbaren Einstellungsvarianten kann nicht gewährleistet werden!
Falls in der entsprechenden Slave-Dokumentation nicht anders angeben, wird dringend davon abgeraten, die automatisch gesetzten Einstellungen zu verändern.

Distributed Clocks mit Ausgangskanälen

Im Wesentlichen unterscheidet sich die Logik die bei Ausgangskanälen angewendet wird nicht von der bereits beschriebenen bei Eingangskanälen. Der einzige Unterschied ist, dass der Slave-lokale SYNC-Takt nun meist zweckmäßigerweise *nach* der Passage eines datenbringenden EtherCAT-Frames zu erfolgen hat, statt wie bei den Eingangskanälen *vor* dem Frame. Deshalb findet die Berechnung der oben angegebenen globalen Shift Time (Abb. *Zeitdiagramm Einlesen der EL1202-0100*, blau) für Ein- und Ausgangs-Slaves getrennt statt. Die standardmäßig ermittelte globale Shift Time für die Gruppe der Ausgangsbaugruppen ist damit größer als die, die für die Gruppe der Eingangsbaugruppen berechnet wird. Beide liegen jedoch in dem sinnvollen Bereich von 0..100% der EtherCAT-Zykluszeit.

In den folgenden Abschnitten wird nun der Zugriff auf die Distributed Clocks-Einstellungen aus dem TwinCAT System Manager heraus erklärt.

7.2.2 Einstellungen Distributed Clocks im Beckhoff TwinCAT System Manager (2.10)

Für den EtherCAT- Master und die Slaves mit Distributed Clocks Unterstützung gibt es unterschiedliche Konfigurationsdialoge.

HINWEIS

Achtung! Keine Plausibilitätskontrolle!

Die aufgeführten Hinweise und Erläuterungen sollten mit Bedacht angewendet werden!
Die genannten Einstellungen werden vom EtherCAT-Master automatisch mit Werten belegt, die eine zuverlässige und aktuelle Prozessdatenerfassung unterstützen.
Anwenderseitige Eingriffe an dieser Stelle können zu unerwünschtem Verhalten führen!
Bei der Manipulation dieser Einstellungen im Beckhoff TwinCAT System Manager wird softwareseitig keine Plausibilitätskontrolle durchgeführt!
Eine korrekte Funktion der EtherCAT-Slave in allen denkbaren Einstellungsvarianten kann nicht gewährleistet werden!
Falls in der entsprechenden Slave-Dokumentation nicht anders angeben, wird dringend davon abgeraten, die automatisch gesetzten Einstellungen zu verändern.

Gültigkeit der nachstehenden Einstellungen

Die gezeigten Einstellmöglichkeiten sind einem Beckhoff TwinCAT 2.10 Build 1320 entnommen. Neuere Ausgaben können eine abweichende Oberflächengestaltung aufweisen, die Verwendung bleibt aber sinngemäß die gleiche.

Einstellungen des Masters

Jedes EtherCAT-Gerät im System Manager bietet in seinen Erweiterten Einstellungen Zugang zu den Distributed-Clocks-Einstellungen, falls EtherCAT-Slave in der Konfiguration präsent ist:

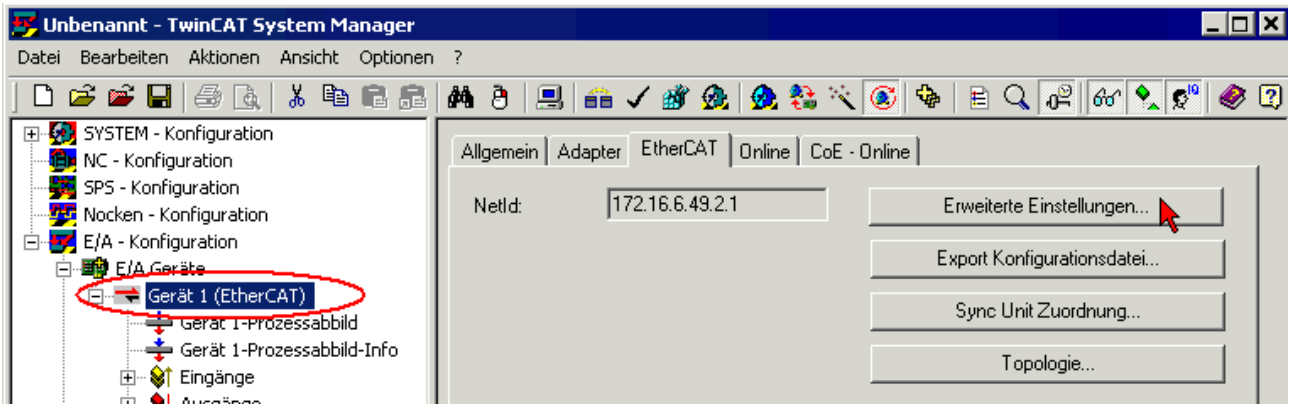


Abb. 261: EtherCAT- Master - Erweiterte Einstellungen

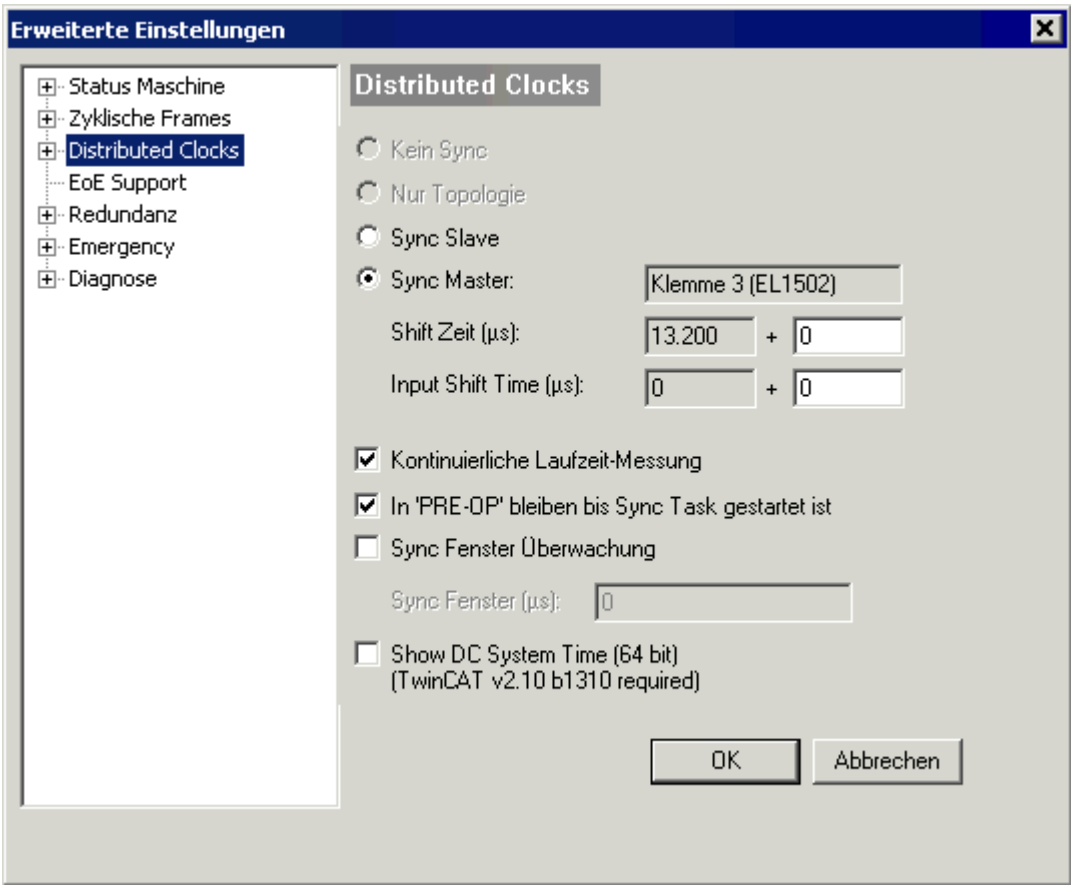


Abb. 262: EtherCAT- Master - Distributed Clocks

- **Sync Slave**
  Wenn aktiviert, ist dieses EtherCAT-Gerät/dieser EtherCAT-Strang ein Sync-Slave zu einem anderen EtherCAT-Strang im selben PC, dem Sync Master - dieser enthält die Reference Clock und regelt die Echtzeit auf dem PC. Im Sync Slave EtherCAT-Strang gibt es keine Reference Clock.
- **Sync Master**
  Defaulteinstellung - dieses EtherCAT Gerät (hier: Gerät 1) regelt die PC Echtzeit; die Reference Clock liegt in diesem EtherCAT Strang und ist in diesem Beispiel der EtherCAT-Slave "Klemme 3", eine EL1502.
- **Shift Zeit (µs)**
  Das ist die automatisch vom System Manager berechnete Shift Zeit für alle EtherCAT-Slaves - in diesem Fall lösen alle lokalen Distributed Clocks in den EtherCAT-Slaves ihren SYNC 13,2 µs nach dem Echtzeit-Tick aus. Dies betrifft sowohl Slaves, die als Eingangsbaugruppe als auch Slaves, die als Ausgangsbaugruppe deklariert sind.
  Der Anwender kann hier mit einem zusätzlichen Wert eingreifen und so die SYNC-Pulse um positive oder negative Werte verschieben.

- **Input Shift Zeit (µs)**
  Diese Shift Time betrifft nur Slaves, die als Eingangsbaugruppe deklariert sind (s. XML Device Description). Auch hier kann manuell verändert werden.
- **Show DC System Time (64 bit)**
  Wenn aktiviert, erscheint im Prozessabbild des EtherCAT-Masters die neue 64 Bit Eingangsvariable *DcSysTime*, s. Abb. *Aktivierbare Eingangsvariable DcSysTime*. Sie ist eine Kopie der Uhrzeit in der EtherCAT Reference Clock.

Genauigkeit der Variable DcSysTime

Die Uhrzeit aus der Reference Clock wird ohne Anspruch auf gleichmäßige Abtastung gewonnen. Sie soll dem Anwender lediglich als grobe Orientierung dienen, in welchem Zeitbereich sich das EtherCAT-System gerade befindet. Sie wird zwar zyklisch abgetastet, der ermittelte Wert kann aber durch die "weiche" Abtastung um bis zu +/- 1 Zykluszeit jittern.
Für hochgenaue relative zeitbasierte Aktionen sind die Zeitwerte geeigneter EtherCAT-Slaves zu verwenden - z. B. die Zeitstempel einer EL1252.

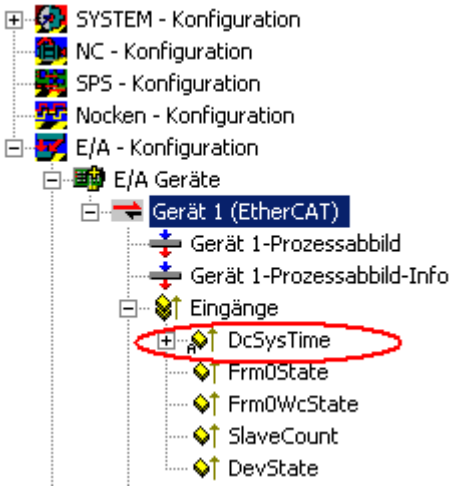


Abb. 263: Aktivierbare Eingangsvariable DcSysTime

HINWEIS

Achtung! Keine Plausibilitätskontrolle!

Die aufgeführten Hinweise und Erläuterungen sollten mit Bedacht angewendet werden!
Die genannten Einstellungen werden vom EtherCAT-Master automatisch mit Werten belegt, die eine zuverlässige und aktuelle Prozessdatenerfassung unterstützen.
Anwenderseitige Eingriffe an dieser Stelle können zu unerwünschtem Verhalten führen!
Bei der Manipulation dieser Einstellungen im Beckhoff TwinCAT System Manager wird softwareseitig keine Plausibilitätskontrolle durchgeführt!
Eine korrekte Funktion der EtherCAT-Slave in allen denkbaren Einstellungsvarianten kann nicht gewährleistet werden!

Falls in der entsprechenden Slave-Dokumentation nicht anders angeben, wird dringend davon abgeraten, die automatisch gesetzten Einstellungen zu verändern.

Einstellungen der Slaves

Gültigkeit der nachstehenden Einstellungen

Anhand der Klemme EL1202-0100 soll stellvertretend die DC-Funktionalität erläutert werden. Jeder EtherCAT-Slave mit Distributed Clocks Unterstützung nutzt dieses Feature auf individuelle Weise und wird daher in den zugehörigen Dokumentationen entsprechend beschrieben.

HINWEIS

Achtung! Keine Plausibilitätskontrolle!

Die aufgeführten Hinweise und Erläuterungen sollten mit Bedacht angewendet werden!
Die genannten Einstellungen werden vom EtherCAT-Master automatisch mit Werten belegt, die eine zuverlässige und aktuelle Prozessdatenerfassung unterstützen.
Anwenderseitige Eingriffe an dieser Stelle können zu unerwünschtem Verhalten führen!
Bei der Manipulation dieser Einstellungen im Beckhoff TwinCAT System Manager wird softwareseitig keine Plausibilitätskontrolle durchgeführt! Eine korrekte Funktion der EtherCAT-Slave in allen denkbaren Einstellungsvarianten kann nicht gewährleistet werden!
Falls in der entsprechenden Slave-Dokumentation nicht anders angeben, wird dringend davon abgeraten, die automatisch gesetzten Einstellungen zu verändern.

Karteireiter "DC"

Falls ein EtherCAT-Slave die Distributed Clocks unterstützt, erscheint ein Reiter "DC" für die Parametrierung. Wenn ein EtherCAT-Slave mehrere Betriebsarten/Operation Modi anbietet, kann dieser hier ausgewählt werden. Die EL1202-0100 z. B. kann nur in einer Betriebsart benutzt werden, weshalb hier keine Auswahl möglich ist.

Über den Button "Advanced Settings" gelangen Sie in einen erweiterten Distributed Clocks Dialog:

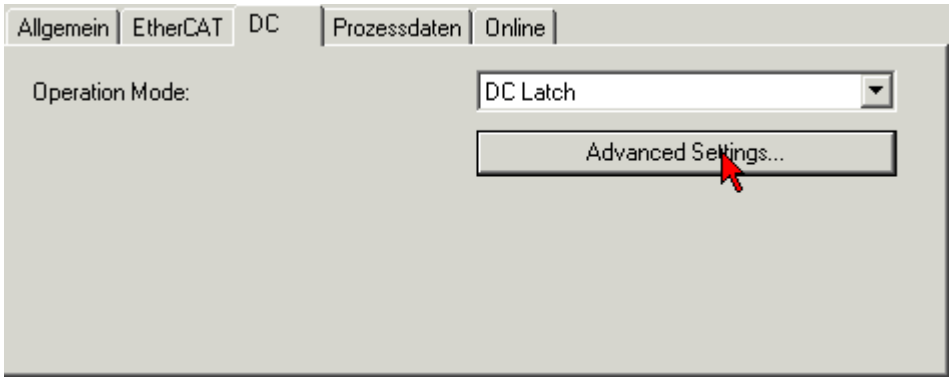


Abb. 264: Karteireiter DC

In Abb. *Dialog Distributed Clocks im EtherCAT-Slave* ist die Grundseite eines jeden EtherCAT-Slave mit Distributed Clocks zu sehen (TwinCAT 2.10, build 1320):

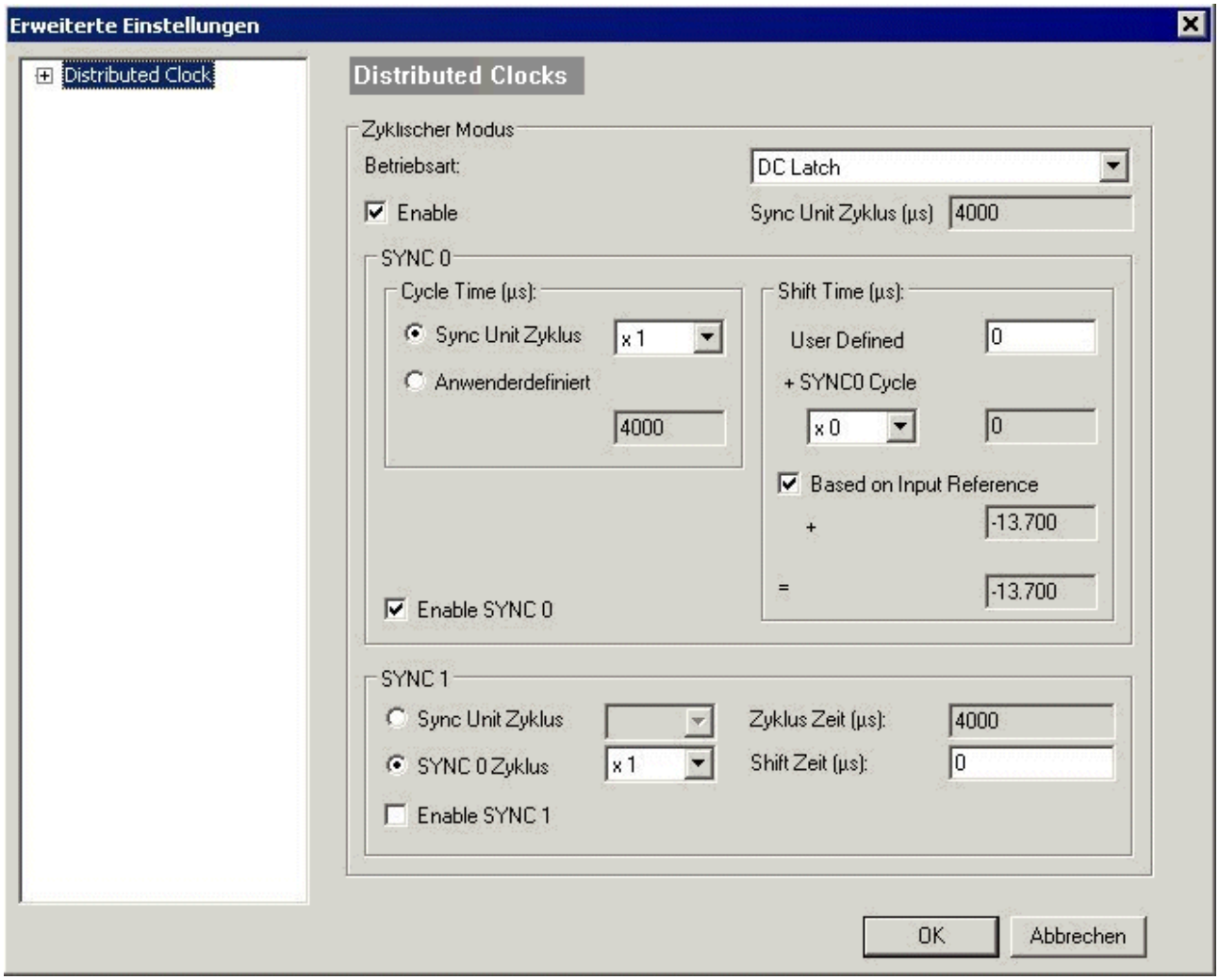


Abb. 265: Dialog Distributed Clocks im EtherCAT-Slave

- **Betriebsart**
  Gleiche Funktionalität wie im übergeordneten Dialog.
- **Zyklischer Modus/Enable**
  Schaltet Distributed Clocks an.

Verwendung eines EtherCAT Slave als Reference Clock

Wenn ein EtherCAT Slave Distributed Clocks herstellerseitig unterstützt, muss dies nicht notwendigerweise aktiviert sein. Um einen EtherCAT Slave als Reference Clock verwenden zu können, muss mit der Checkbox "Enable" die Slave-lokale Clock angeschaltet werden, auch wenn Distributed Clocks für den eigentlichen Einsatz dieses Slaves nicht benötigt wird.

- **Sync Unit Zyklus (µs)**
  Grundzyklus im EtherCAT-Slave - entspricht der EtherCAT Zykluszeit, die diesen EtherCAT-Slave gerade behandelt. In diesem Beispiel fragt eine Task mit 4 ms Zykluszeit (4000 µs) diese EL1202-0100 ab. Werden mehrere Tasks mit unterschiedlichen Zykluszeiten auf einem EtherCAT-Strang betrieben, steht hier nur die Zykluszeit der Task, die mit dem gerade betrachteten Slave im Prozessdatenaustausch steht. Sind mehrere Tasks auf einen Slave angesetzt, steht hier die schnellste Taskzykluszeit.

Es folgen nun 2 Abschnitte, um jeweils die beiden durch die Distributed Clocks Einheit im ESC erzeugten Interrupt-Signale näher zu spezifizieren.

- **Enable SYNC0**
  Aktiviert das SYNC0-Signal.

- **SYNC0 - Zyklus Cycle Time**
  Hier kann ein Mehrfaches oder ein Bruchteil des o.a. Grundzyklus eingestellt werden. Das Ergebnis erscheint im Fenster darunter (hier: 4000 µs bei Faktor 1). In diesen Abständen wird das SYNC0-Signal vom ESC generiert, wenn der SYNC0 bzw. Distributed Clock überhaupt aktiviert sind.
- **Anwenderdefiniert**
  Alternativ ist ein beliebiger Wert einzugeben.
- **Shift Time**
  Wie bereits in der allgemeinen Einführung zu Distributed Clocks besprochen, kann der SYNC-Puls eines EtherCAT-Slave um eine konstante Zeit vor oder zurück verschoben werden (S0 User Shift Time [▶ 242]). Die EL1202-0100 ist den Eingabebaugruppen zugeordnet weshalb hier das
- **Based on Input Reference**
  aus der globalen Distributed Clocks Einstellung des EtherCAT-Masters auf diesen Slave angewendet wird. Sowohl
- **User defined** als auch das
- **Mehrfache der SYNC0 Cycletime** sind standardmäßig 0. Somit addieren sich die Zeitkomponenten in dieser EL1202-0100 in diesem Beispiel zur
- **Gesamt-Shift-Time des SYNC0** von -13,7 µs.

Ein etwas reduzierter Dialog steht für die Einstellung des SYNC1- Signals zur Verfügung:

- **Enable SYNC1**
  Aktiviert das SYNC1-Signal.
- **SYNC1 - Zyklus Cycle Time**
  Die SYNC1-Cycletime kann entweder aus einem Mehrfachen/einem Bruchteil des Grundzyklus oder der SYNC0-Zykluszeit abgeleitet sein.
- **Shift Zeit (µs)**
  Hier kann manuell eine konstante Verschiebezeit in µs zwischen SYNC0- und SYNC1-Signal eingetragen werden.

Zusammenhang zwischen SYNC0 und SYNC1

Im Gegensatz zum SYNC0-Signal ist das SYNC1-Signal kein völlig unabhängiger Interrupt, wie schon an den reduzierten und anderslautenden Eigenschaftsdialogen erkennbar ist. Weitere Erläuterungen dazu finden Sie unter www.ethercat.org z. B. bei den Spezifikationen der ESC.

HINWEIS

Achtung! Keine Plausibilitätskontrolle!

Die aufgeführten Hinweise und Erläuterungen sollten mit Bedacht angewendet werden!
Die genannten Einstellungen werden vom EtherCAT-Master automatisch mit Werten belegt, die eine zuverlässige und aktuelle Prozessdatenerfassung unterstützen.
Anwenderseitige Eingriffe an dieser Stelle können zu unerwünschtem Verhalten führen!
Bei der Manipulation dieser Einstellungen im Beckhoff TwinCAT System Manager wird softwareseitig keine Plausibilitätskontrolle durchgeführt!
Eine korrekte Funktion der EtherCAT-Slave in allen denkbaren Einstellungsvarianten kann nicht gewährleistet werden!
Falls in der entsprechenden Slave-Dokumentation nicht anders angeben, wird dringend davon abgeraten, die automatisch gesetzten Einstellungen zu verändern.

Zeitbezogene Zusammenarbeit mit anderen Klemmen

Zum Abschluss dieser Einführung soll noch ein weiteres Beispiel aufgeführt werden, wie die Distributed Clocks in einem EtherCAT-System eingesetzt werden können.
Es bestehe die Aufgabe, einen analogen Eingangswert im Bereich +/- 10 V in einem Abstand von 50 µs in einem exakten Abstand abzutasten und zur Steuerung zu übertragen, s. Abb. *Anwendungsbeispiel einer manuellen Shift Zeit auf SYNC0*.

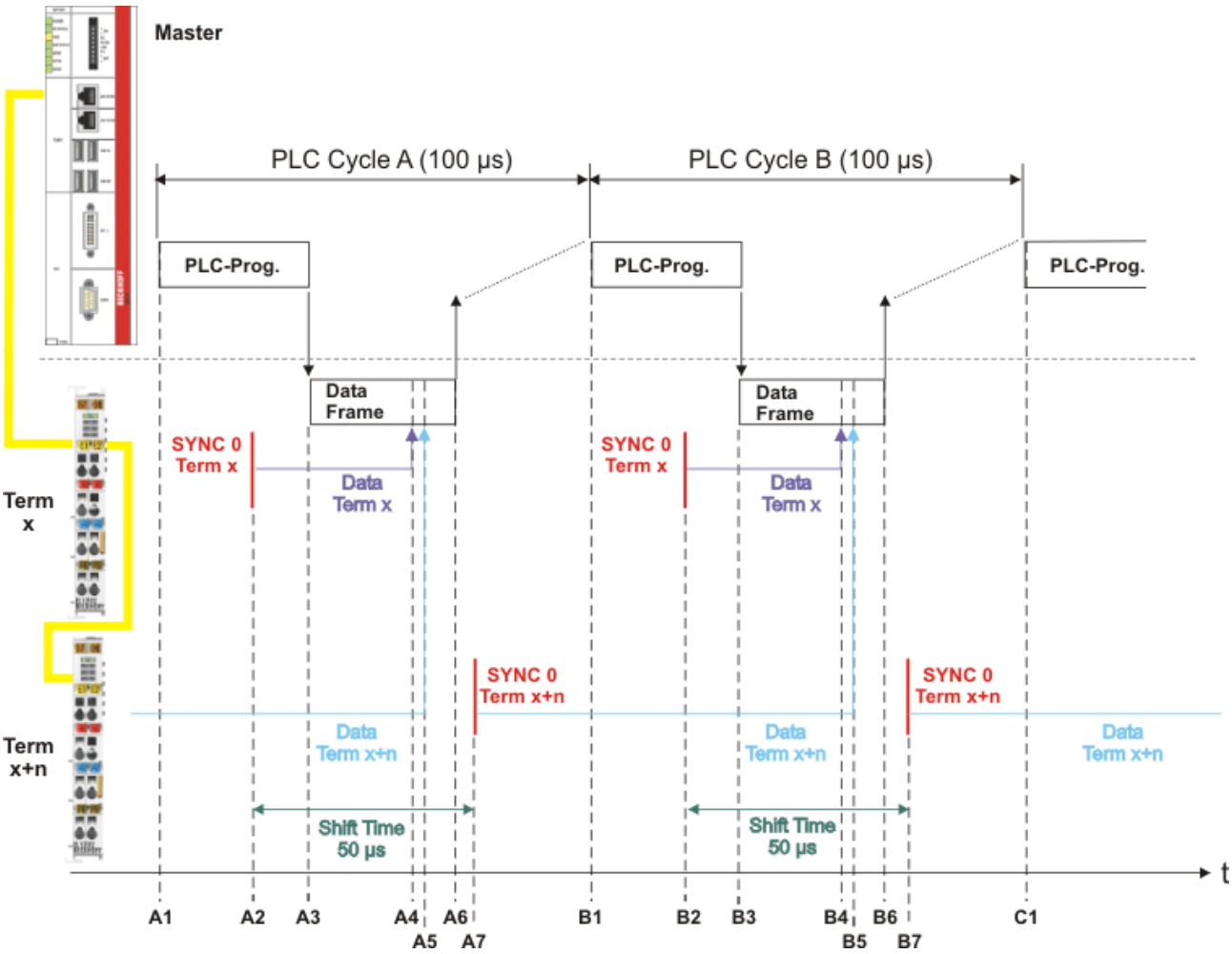


Abb. 266: Anwendungsbeispiel einer manuellen Shift Zeit auf SYNC0

Im Beispiel wird eine entsprechend schnelle analoge Eingabebaugruppe (z. B. Beckhoff EL3702) zusammen mit einer EtherCAT Zykluszeit von 50 µs verwendet. Außerdem soll hier eine andere analoge Eingabebaugruppe verwendet werden, die nur eine Analog/Digital-Wandlungszeit von 60 µs hat. Damit ist eine EtherCAT Zykluszeit von 50 µs nicht sinnvoll.

Hier bietet sich folgender Lösungsweg an: 2 solcher Eingabebaugruppen werden direkt nebeneinander betrieben (hier als "Term x" und "Term x+1" bezeichnet) und mit demselben Eingangssignal beaufschlagt. Die Zykluszeit sei 100 µs (A1 bis B1), somit hat jede Eingabebaugruppe ausreichend Zeit (100 µs > 60 µs) um den analogen Eingangswert zu wandeln. Ausgelöst wird das Wandeln vom SYNC0-Signal in den Eingabebaugruppen in einem Abstand von 100 µs, standardmäßig zum Zeitpunkt A2, B2 usw.. Dazu wird "Term x" betrachtet: zum Zeitpunkt A2 löst das SYNC-Signal das Wandeln der Eingangswerte aus (der Analog/Digital-Konverter wird gestartet). Die Vorlaufzeit (A2 bis A4) ist so bemessen, dass ausreichend Zeit für Wandlung und Datenbereitstellung im Slave besteht, bis zum Zeitpunkt A4 der Slave "Term x" seine Eingangsdaten in den passierenden Ethernet-Frame einkoppelt. Zum Zeitpunkt A6 ist der Frame wieder am Master angekommen - die nächste Datenverarbeitung beginnt dann zum Zeitpunkt B1.

Um zur gewünschten Zeitauflösung von 50 µs zu gelangen, wird in der zweiten Eingabebaugruppe manuell vom Anwender eine konstante User Shift Zeit von 50 µs auf das SYNC0-Signal eingetragen, wie in den o.a. Dialogen gezeigt. Damit wandelt "Term x+1" seine Eingangssignale immer um 50 µs nach "Term x" zum Zeitpunkt A7, B7. usw. Entsprechend koppelt dann "Term x+1" seine zum Zeitpunkt A7 ermittelten Daten zum Zeitpunkt B5 in den Ethernet Frame ein - kurz *nach* B4, weil in diesem Beispiel "Term x+1" *nach* "Term x" angeordnet ist.
Anmerkung: über die Position der Daten im *Ethernet Frame* sagt Abb. *Anwendungsbeispiel einer manuellen Shift Zeit auf SYNC0* nichts aus - nach rechts ist nur die Zeitachse dargestellt!

Damit wandeln beide Eingabebaugruppen zyklisch alle 100 µs, aber mit einem konstanten Versatz von 50 µs den Eingangswert mit einer Abweichung von < 100 ns. Die entstehenden Prozessdaten müssen nun vom Anwender im PLC-Programm in der zeitlich richtigen Reihenfolge interpretiert werden. Die entsprechenden Klemmen können dies durch einen Zeitstempel des Prozessdatums unterstützen, wie in der jeweiligen Dokumentation angeben.

7.2.3 Einstellungen Distributed Clocks im Beckhoff TwinCAT System Manager (2.11)

HINWEIS

Achtung! Keine Plausibilitätskontrolle!

Die aufgeführten Hinweise und Erläuterungen sollten mit Bedacht angewendet werden!
Die genannten Einstellungen werden vom EtherCAT-Master automatisch mit Werten belegt, die eine zuverlässige und aktuelle Prozessdatenerfassung unterstützen.
Anwenderseitige Eingriffe an dieser Stelle können zu unerwünschtem Verhalten führen!
Bei der Manipulation dieser Einstellungen im Beckhoff TwinCAT System Manager wird softwareseitig keine Plausibilitätskontrolle durchgeführt! Eine korrekte Funktion der EtherCAT-Slave in allen denkbaren Einstellungsvarianten kann nicht gewährleistet werden!

Falls in der entsprechenden Slave-Dokumentation nicht anders angeben, wird dringend davon abgeraten, die automatisch gesetzten Einstellungen zu verändern!

Gültigkeit der nachstehenden Einstellungen

Die gezeigten Einstellmöglichkeiten sind einem Beckhoff TwinCAT 2.11 Build 1540 entnommen. Neuere Ausgaben können eine abweichende Oberflächengestaltung aufweisen, die Verwendung bleibt aber sinngemäß die gleiche.

Mit der Markteinführung von TwinCAT 2.11 stehen in System Manager und PLC neue Funktionen zur Verfügung, die die Inbetriebnahme von Distributed Clocks-Slaves (DC-Slaves) vereinfachen bzw. anschaulicher machen. Dies sind im

- System Manager
  - individuelle Anzeige der ShiftTime bei jedem Slave
  - externe Synchronisation möglich
- PLC
  - neue Funktion *F_GetCurDcTaskTime*
  - neue Funktion *F_GetActualDcTime*

Im Rahmen von TwinCAT 2.10 in dieser Dokumentation getroffene Aussagen behalten ihre Gültigkeit.

System Manager - Anzeige Wirkzeitpunkt

Im System Manager ab TwinCAT 2.11 erlaubt eine optionale Anzeige die Onlineverrechnung, wann Ausgänge aus Sicht der Steuerung gesetzt werden bzw. von wann gelesene Eingänge stammen. Die nachfolgenden Informationen gelten ausschließlich für Distributed Clocks-Slaves.

Zum Grundverständnis wird hier ein vollständiger EtherCAT-Update-Zyklus an einem Beispiel erklärt - es wird im Beispiel ein digitaler Eingang eingelesen und auf einen digitalen Ausgang ausgegeben. In beiden Fällen handelt es um DC basierte Slaves, deshalb wird eine EL1202-0100 (Eingang, gelb) und eine EL2202-0100 (Ausgang, rot) verwendet.

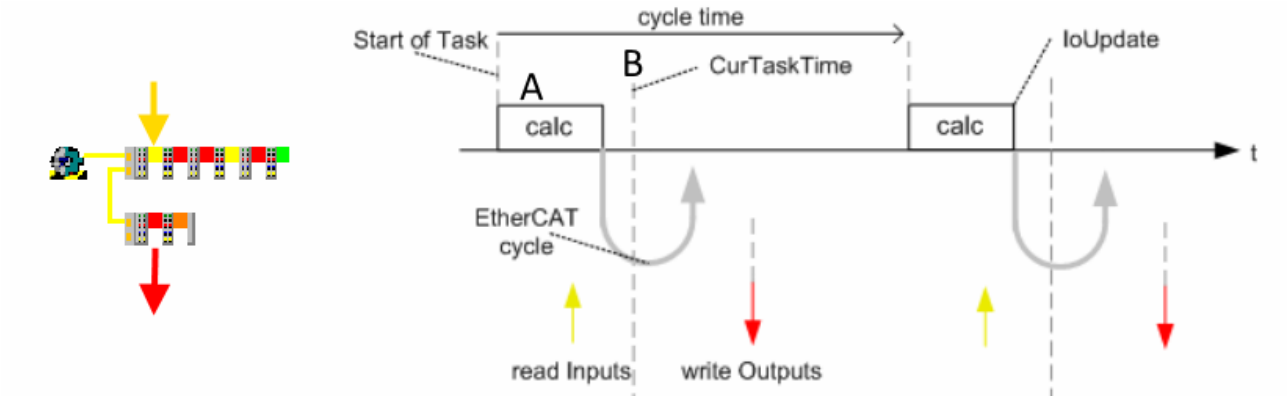


Abb. 267: Im Beispiel verwendete Topologie und Arbeitsprinzip

Einleitende Bemerkungen:

- EtherCAT ist korrekt konfiguriert und führt nach jedem PLC-Zyklus "calc" einen Feldbuszyklus auf EtherCAT "Io-Update" mit der Zykluszeit "cycletime" (im Beispiel: 100 µs) durch. *SeperateInputUpdate* oder "*IO am Taskanfang*" werden in diesem Beispiel nicht verwendet.
- Im EtherCAT-Zyklus werden zugleich die Eingangsdaten von den Slaves eingesammelt wie auch die Ausgangsdaten zu den Slaves geschrieben.
- Die Slaves-Uhren sind durch das DC-System synchronisiert zum EtherCAT-Master.
- Eine externe Synchronisation von TwinCAT gegenüber einer übergeordneten Referenzuhr findet in diesem Beispiel nicht statt.

Neu in TwinCAT 2.11 ist die Möglichkeit, in der PLC den Zeitpunkt *CurTaskTime* durch Aufruf der Funktion *F_GetCurDcTaskTime* online zur Laufzeit zu ermitteln. Der Zeitpunkt, an dem der EtherCAT Frame beim ersten DC-fähigen Teilnehmer ankommt (Abb. *Im Beispiel verwendete Topologie und Arbeitsprinzip*, B), ist **der** zentrale Zeitpunkt, auf den das gesamte System geregelt wird. Er entspricht der **CurTaskTime** des vorangehendes Zyklus', das bedeutet: wird im PLC-Zyklus A die Funktion *F_GetCurDcTaskTime* aufgerufen, meldet sie die Zeit B zurück. Dieser Wert bleibt auch bei mehrmaliger Ermittlung innerhalb dieses Taskzyklus konstant. Aus der Sicht des ablaufenden PLC-Programms ist das der "Jetzt"-Zeitpunkt. Von diesem "Jetzt"-Standpunkt aus sieht die Steuerung

- ihre Eingangsdaten: diese wurden vor x Zeit im Feld gewonnen.
- ihre Ausgangsdaten: diese werden y Zeit später im Feld wirksam werden

Die Zeiten x bzw. y sind bei einem einwandfreien und stabil laufenden EtherCAT-System mit Beckhoff TwinCAT EtherCAT-Master in jedem Zyklus konstant.

Diese Betrachtungsweise ist aber nur zulässig, da durch das DC-System exakt bestimmt wird, wann Eingangsdaten gelesen bzw. Ausgangsdaten ausgegeben werden und ist deshalb für Slaves ohne DC-Funktionalität (wie EL200x oder EL100x) nicht anwendbar.

In TwinCAT 2.11 stehen nun die Zeiten x und y als vorberechnete Werte im System Manager zur Verknüpfung mit der Steuerung zur Verfügung:

- DcInputShift: so "alt" sind die Eingangsdaten aus Sicht des "Jetzt"-Zeitpunktes
- DcOutputShift: soviel später werden die Ausgangsdaten im Feld wirksam.

Hinweis

TwinCAT triggert auf Basis der internen Echtzeit die Task/NC/PLC/... Ziel der Regelung ist dabei, den (ersten) EtherCAT Frame immer im Zyklus-Abstand (z. B. 1 ms) am ersten DC-fähigen EtherCAT Slave ankommen zu lassen. Dies wird erschwert bzw. wird sogar unmöglich, falls die PLC/Task eine extrem ungleichmäßige/schwankende Ausführungszeit aufweist. Dann kommt es zu Synchronisierungsproblemen am Feldbus. Durch die TwinCAT Einstellung "percent of cycle time" kann hier entsprechender Puffer vorgehalten werden.

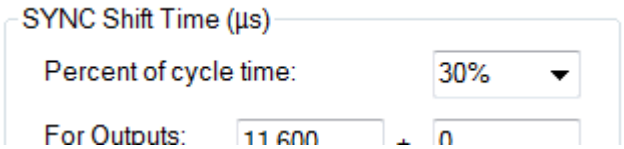


Abb. 268: TwinCAT EtherCAT Master Einstellung, Abschnitt DistributedClocks

Die TwinCAT Echtzeitregelung versucht durch entsprechendes Triggern/Starten der PLC/NC/sonstige Task, trotz der evtl. schwanken Ausführungszeit

Beispiel

Betrachten wir die EL1202-0100 aus obigem Beispiel bei einer Zykluszeit von 100 µs (100.000 ns) in der Konfiguration.

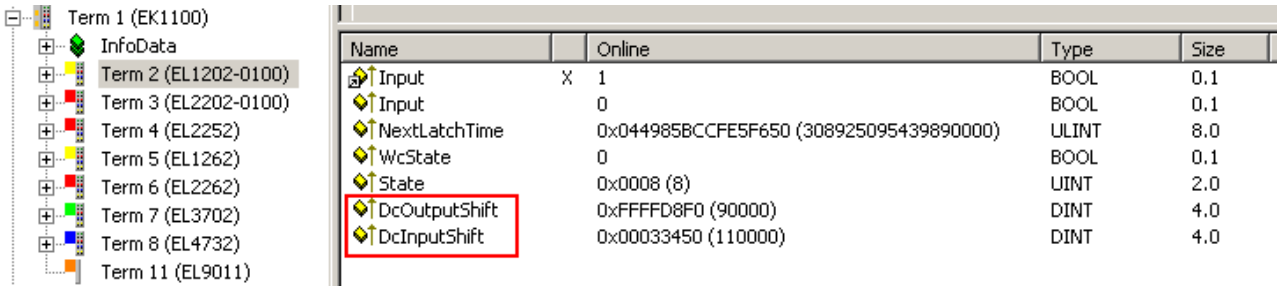


Abb. 269: Prozessdaten

Da so konfiguriert, informiert die Klemme im Online-Zustand über

- DcOutputShift = 90.000 [ns]
- DcInputShift = 110.000 [ns]

Anmerkungen

- Beide Zeitwerte umfassen 32 Bit [1 digit = 1 ns] und sind damit ausreichend für ~4.2 Sekunden
- Beide sind "logisch" zu sehen:
  - positive Werte für Outputs weisen in die Zukunft
  - positive Werte für Inputs deuten in die Vergangenheit
- Jeder Slave zeigt beide Werte an, es ist nach Vorhandensein von In- und/oder Ausgangsvariablen zu beurteilen welcher Wert berücksichtigt werden muss.

Die EL1202-0100 ist eine Eingangsklemme und damit in der DC-Gruppe "InputBased" - sie arbeitet standardmäßig also im Gleichtakt mit allen DC-Eingangsbaugruppen, in Abb. *Im Beispiel verwendete Topologien und Arbeitsprinzip* die gelben Symbole. Außerdem verfügt sie nur über Eingangsvariablen. Besonders zweckmäßig ist also die Betrachtung des Wertes *DcInputShift*:
Aus Sicht des "Jetzt"-Zeitpunktes der Steuerung ist der Eingangswert *Input*=1 also 110.000 ns "alt".

Anmerkung: die EL1202-0100 gibt zusätzlich auch noch den exakten Latch-Zeitpunktes als Prozessdatum *NextLatchTime* an, ein Feature das nicht alle DC-Slaves unterstützen.

Ablauf an einem Beispiel

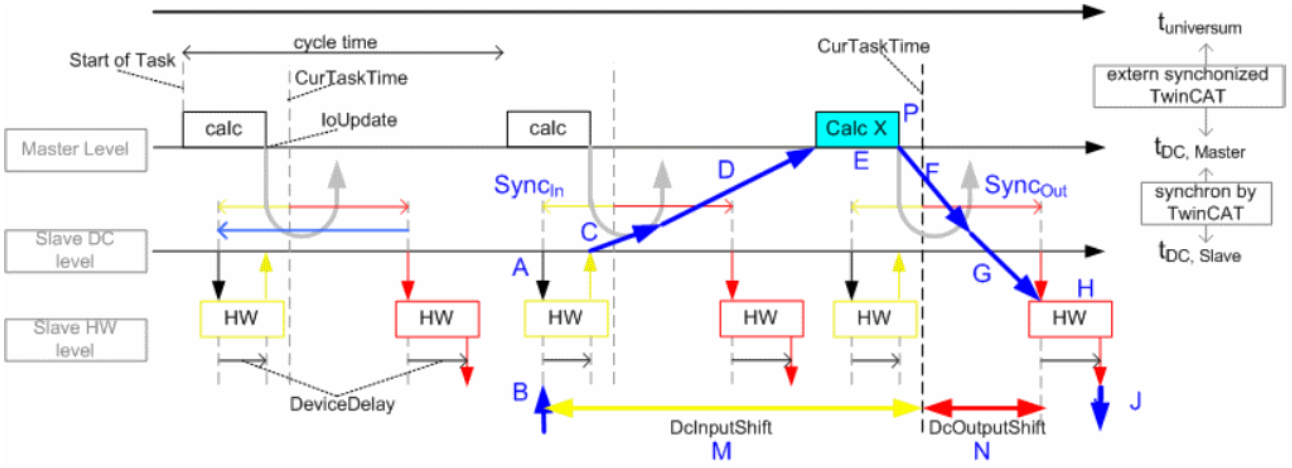


Abb. 270: EtherCAT Update (schematisch)

In diesem Beispiel soll der digitale Eingang B eingelesen und umgehend auf einem digitalen Ausgang J wieder ausgegeben werden. Wir folgen dem Ablauf (blau):

- A: die DC-Unit in der EL1202-0100 stößt mit dem $Sync_{In}$ das "Latchen" der Eingänge an - dieser Zeitpunkt ist fix und wird durch die Shiftzeiten im Master/Slave definiert.
  Dieses "Latchen" muss so rechtzeitig erfolgen, dass die Daten trotz etwaiger Schwankungen sicher vor dem abholenden Frame bereit stehen.
- B: die Eingänge werden gelesen, je nach Gerät tritt noch eine geräteseitige Verzögerung auf - zum Zeitpunkt C stehen die Daten zur Abholung bereit.
  Anmerkung: in der EL1202 ist die Verzögerung ~ 0 µs.
- D: Die Daten werden vom EtherCAT-Telegramm abgeholt und in das Eingangsprozessabbild der Steuerung abgelegt.
  Der EtherCAT Frame durchläuft die Slaves nacheinander, kommt also zu aufeinander folgenden Zeitpunkten an den Slaves vorbei. Eine besondere Rolle nimmt dabei der ERSTE DC-fähige Slave, die sog. Refernz-Uhr ein: der Zeitpunkt, zu dem bei diesem Slave der (erste) EtherCAT Frame ankommt, ist der zentrale Zeitpunkt *CurTaskTime*
  - auf den die Echtzeitregelung ausgerichtet ist: da in diesem Slave die zentrale fixe Referenzzeit läuft, wird aus dem zu frühen/späten Ankommen des Frames die Echtzeit nachgeregelt.
  - die Zeit, die auch in der PLC als Bezugszeitpunkt für alle Shift-Zeiten *DcInputShift/DcOutputShift* Verwendung findet.
    Beispiel: Im PLC-Zyklus "Calc X" ergibt die Funktion *F_GetCurDcTaskTime* als Rückgabe die Zeit P (obwohl diese von Calc X aus gesehen leicht in der Zukunft liegt). Davon ausgehend können dann *DcInputShift und DcOutputShift* entsprechend verrechnet werden.
- Nach der Berechnung E erfolgt mit dem nächsten IO-Zyklus bzw. EtherCAT-Update die Ausgabe der Ausgangsdaten an das Feld: F, G.
- Mit genügend Sicherheitsabstand zum Feldbuszyklus erfolgt, getriggert durch das $Sync_{Out}$ der DC-Unit in der EL2202-0100 die Ausgabe der Daten.
- H, J: Je nach Gerät ist noch eine geräteseitige Verzögerung zu berücksichtigen, bis die Daten ausgegeben werden.
  Anmerkung: in der EL2202 ist die Verzögerung < 1 µs.

Die für den Anwender entscheidenden Vorgänge B und J können jederzeit durch die Angaben DcInputShift (M) und DcOutputShift (N) in Bezug zur Jetzt-Zeit (P) gesetzt werden.

Hinweis: Die Funktion *SeparateInputUpdate* hat derzeit (2015-06, TwinCAT 3.1 b4018) keine automatische Auswirkung auf die Berechnung der *DcInputShift/DcOutoutShift.* Im Anwendungsfall sollten manuell die (Input-) Shiftzeiten der betreffenden Klemmen angepasst werden.

Überprüfung der DC-Wirksamkeit aus der Steuerungssicht

Die o.a. Werte *DcInputShift* und *DcOutputShift* sind theoretische Werte die gültig sind, wenn das EtherCAT-System stabil läuft. Die Default-Einstellungen des System Manager sind so definiert, dass diese Werte in den allermeisten Fällen zuverlässig eingehalten werden.

Typische Gründe warum sie in der Applikation gelegentlich oder systematisch nicht eingehalten werden können sein:

- qualitativ schlechte Echtzeit z. B. bei Verwendung von Fremd-Hardware
- Zykluszeitüberschreitungen durch Programmfehler
- falsche Taskpriorisierung
- LostFrames oder anderweitig fehlerhafte Ethernet-Frames, z. B. durch fehlerhafte Verkabelung
- durch manuelle DC-Einstellung kommen die SYNC-Impulse im Slave zum falschen Zeitpunkt

Dennoch kann eine Überprüfung der tatsächlich erfolgten Lese/Schreiboperationen im Feld sinnvoll sein, z. B. bei der Inbetriebnahme, bei manueller DC-Optimierung von Slaves. Als Überprüfung wird empfohlen:

- bei Input-Slaves
  - CycleCounter:
    Manche Slaves (siehe Dokumentation) wie EL37xx, EL12xx bieten einen fortlaufenden CycleCounter an, der in jedem Slave-Zyklus inkrementiert. Dies kann in der Steuerung überwacht werden.
  - InputToggle:
    Manche Slaves bieten dies Eingangsvariable an, die in jedem erfolgreichen Slave-Zyklus toggelt.
  - WorkingCounter, Status:
    Die Überwachung dieser Statusinformationen ist obligatorisch
- bei Output-Slaves
  - WorkingCounter, Status:
    Die Überwachung dieser Statusinformationen ist obligatorisch
  - lokale Fehlerzähler:
    Manche Slaves (siehe Dokumentation) erwarten ihrerseits einen inkrementierten CycleCounter aus der Steuerung und zählen lokal einen Fehlerzähler hoch, wenn die Steuerung dem nicht folgt. Dieser Fehlerzähler im Slave kann von der Steuerung rückgelesen werden. Die EL2262 oder EL47xx verfügen über diesen Mechanismus.

System Manager - Einstellungen Shiftzeit

Im Advanced-Dialog von EtherCAT-Master und Slave sind in Bezug auf die Shiftzeit sehr ähnliche Dialoge zu finden, weshalb sie hier zusammen besprochen werden.

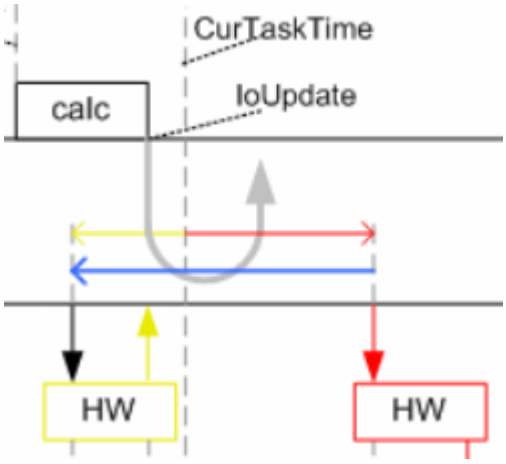


Abb. 271: Lokale Slave- Shiftzeit

Der Begriff "Shiftzeit" in den Einstellungsdialogen unterscheidet sich in der Bedeutung von dem o.a. *DcInput/OutputShift*, vergleichen Sie dazu Abb. *EtherCAT Update (schematisch)* und *Lokale Slave- Shiftzeit*!

- In den TwinCAT-Einstellungsdialogen wird die Shiftzeit angegeben ab dem IoUpdate-Zeitpunkt und stammt sinnvollerweise aus dem Bereich [-Zykluszeit...0...+Zykluszeit].
  Um so viel früher bei Eingängen (Abb. *Lokale Slave- Shiftzeit*, gelb, "A") bzw. später bei Ausgängen (Abb. *Lokale Slave- Shiftzeit*, rot, "B") wird in Bezug auf die IoUpdate-Zeit der Slave-interne SYNC - Impuls ausgelöst.
  Das berücksichtigt aber nicht, dass evtl. einige Zeit benötigt wird um die Daten von/zu den Slaves zu transportieren! Dies macht erst
- DcInput/OutputShift nach Abb. *EtherCAT Update (schematisch)*, die auch *SeperateInputUpdate* oder "*IO am Taskanfang*" berücksichtigt.

Zur Einstellung gilt: effektive Shiftzeit = ShiftzeitMaster + ShiftZeitSlave

Im EtherCAT-Master :

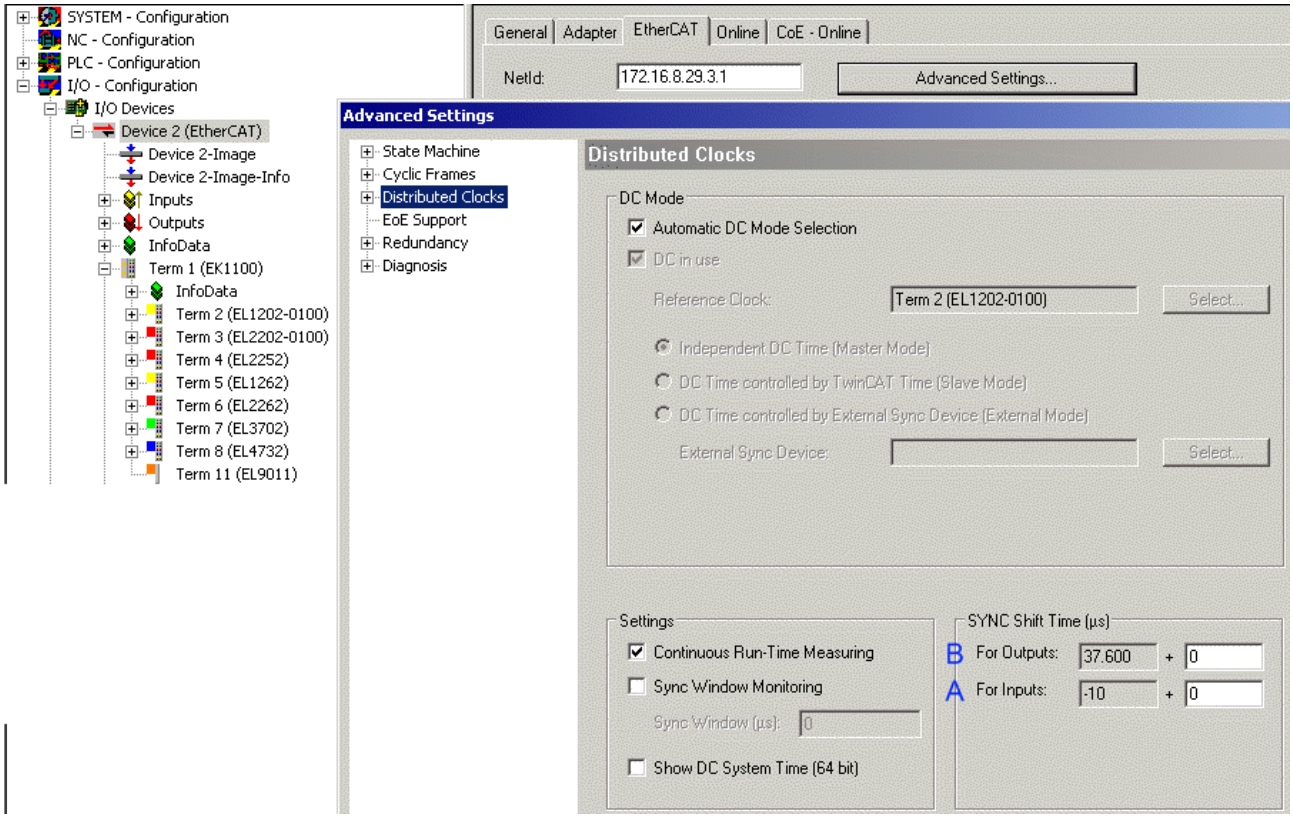


Abb. 272: Einstellung EtherCAT-Master

U.a. auf der Basis der Zykluszeit hat TwinCAT hier die Gruppe der Ausgangs-Baugruppen aus der Sicht der *CurTaskTime* um 37.6 µs in die Zukunft verschoben, die Eingangs-Baugruppen werden dagegen 10 µs vor der *CurTaskTime* ihre Eingänge lesen. Diese beiden Zeiten entsprechen A/B aus Abb. *Lokale Slave-Shiftzeit*. Im Bedarfsfall können sie durch Zusatzeinträge modifiziert werden.

Im EtherCAT-Slave:

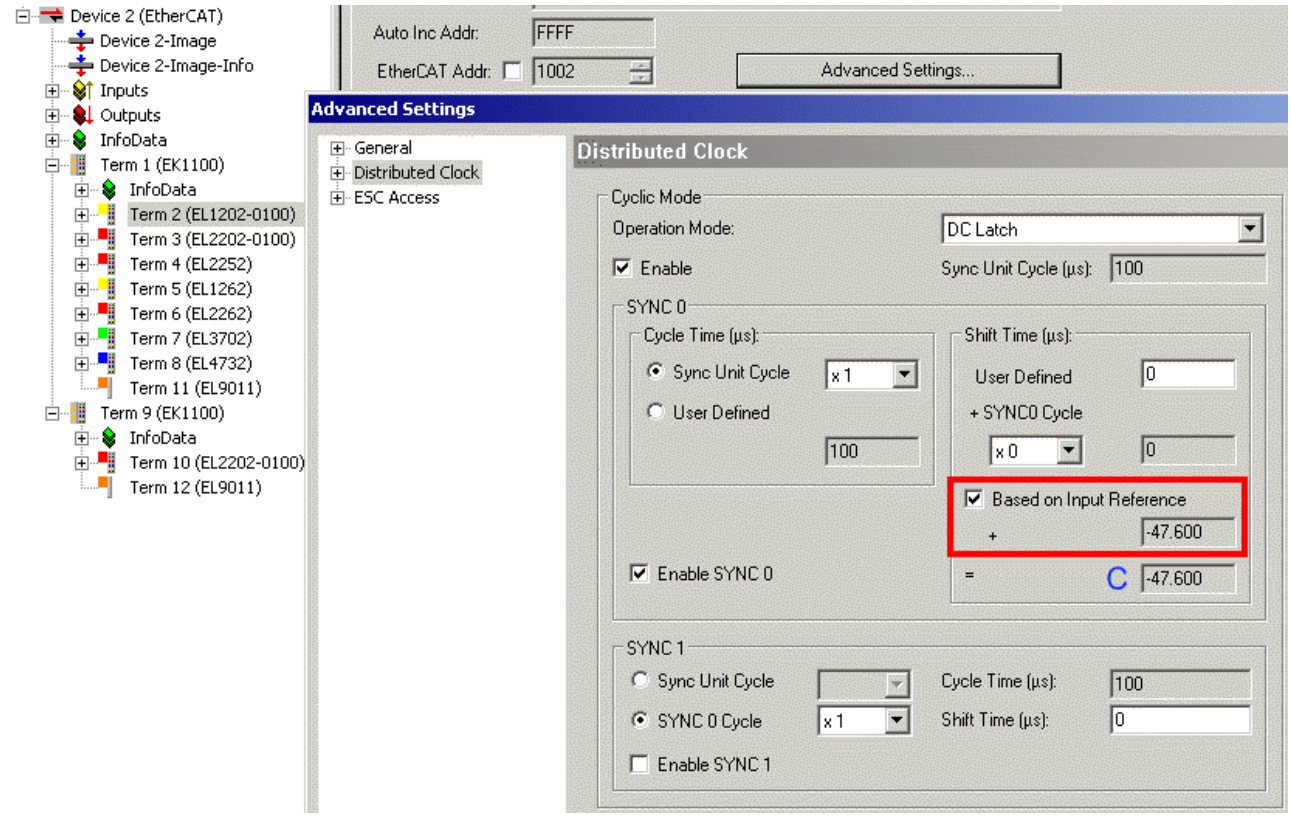


Abb. 273: Einstellung EtherCAT-Slave

Im Slave-Dialog erfolgt die Anzeige der Shifttime aus der (Default)-Sicht der Ausgangsbaugruppen - die hier gezeigte EL1202-0100 ist eine Eingangsbaugruppe, diese wird also automatisch auf *BasedOnInputReference* gelegt und damit wird ihr SYNC-Signal (-37.6 - 10 µs) vor den Ausgangsbaugruppen liegen.

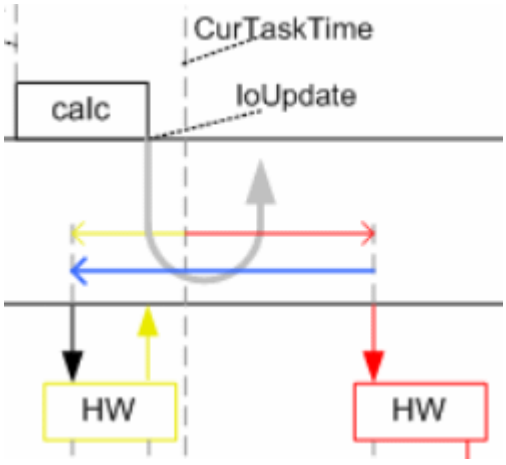


Abb. 274: Shift-Berechnung im Slave-Dialog

Wenn ein Slave manuell "geshiftet" werden soll, könne sinnvolle Werte im Feld *UserDefined* eingetragen werden.

- Werte im Bereich 10..30% der Zykluszeit sind sinnvoll.
- Änderungen in den DC-Einträgen werden erst nach Aktivierung der Konfiguration und Neustart des EtherCAT-Systems wirksam.

Anzeige der Shift-Zeit

Die Anzeige der individuellen Slave Shift-Zeiten ist über die erweiterten Einstellungen des Slaves möglich.

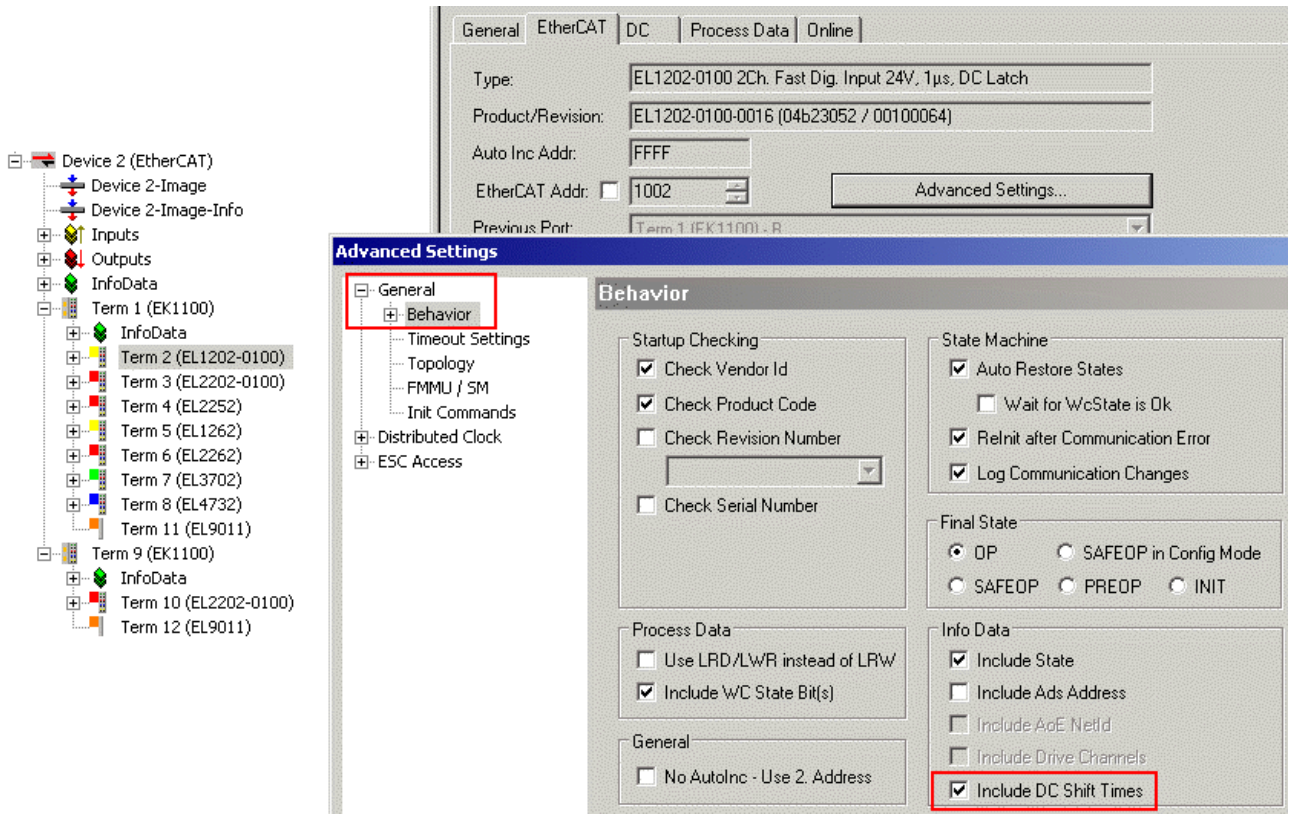


Abb. 275: Anzeige DC Shift-Zeiten

Weitere Einstellungen

- Continuous Runtime Measuring

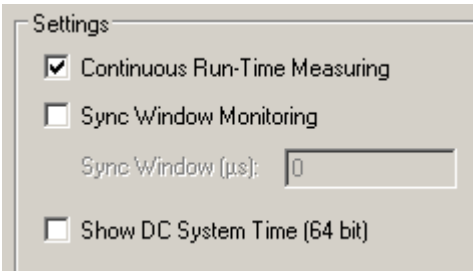


Abb. 276: Einstellung Continuous

Wenn aktiviert, misst TwinCAT zyklisch die Laufzeiten zwischen den EtherCAT Teilnehmern. Es wird empfohlen diese Einstellung beizubehalten. Eine Deaktivierung kann bei sehr kurzer Zykluszeit Raum für zyklische Daten schaffen, weil dann das NOP-Datagramm entfällt.

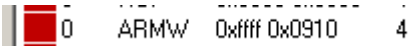


Abb. 277: Runtime Measuring

- SyncWindowMonitoring

  Wenn aktiviert, wird im EtherCAT *DevState* in Bit 12 angezeigt, ob alle DC-Teilnehmer ihre lokalen Uhren innerhalb des angegebenen Fensters halten.

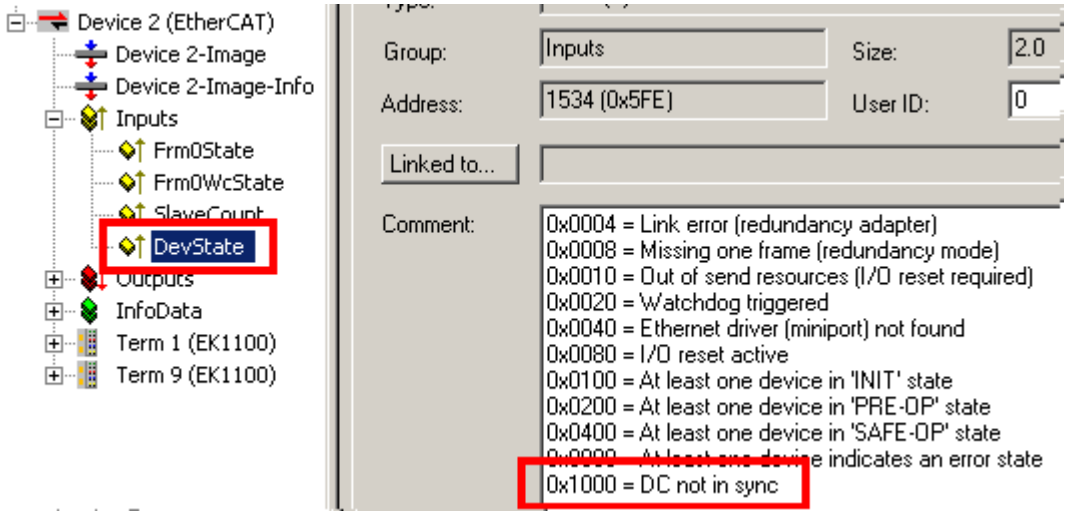


Abb. 278: Eingang DevState

Dafür wird ein zyklisches BRD-Kommando auf 0x092C (Systemzeit Differenz) verwendet.
Die Anzeige ist nur verwertbar, wenn der erste EtherCAT-Teilnehmer auch die Masterclock beinhaltet.

- Show DC System Time

  Wenn aktiviert, wird in den Eingängen des EtherCAT-Master die aktuelle DC-Zeit als Kopie aus der Masterclock angezeigt. Da der Auslesevorgang dem Feldbustransport unterliegt, sollte zur Gewinnung der aktuellen DC-Systemzeit PLC-Bausteinen der Vorzug gegeben werden.

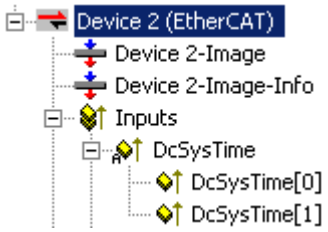


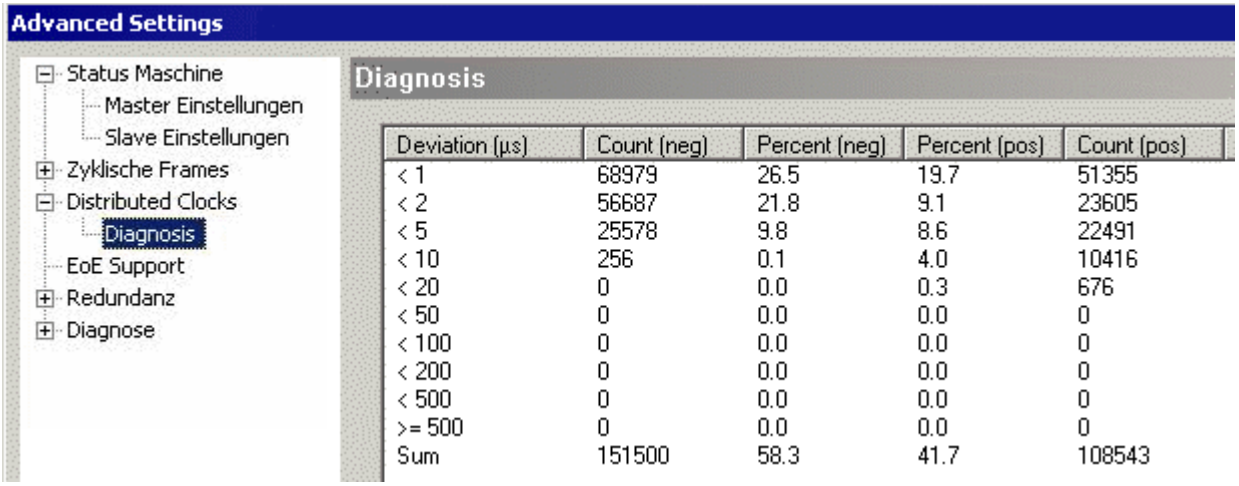
Abb. 279: Eingänge DcSysTime

Distributed Clocks Diagnose

Der TwinCAT System Manager bietet die Möglichkeit, im Online-Zustand eine vorläufige Aussage über die Qualität der aktuellen Echtzeit zu treffen.

Der Ablauf:

- Kommt eine Task zum Aufruf, berechnet sie mit der aktuellen Uhrzeit und der eigenen Zykluszeit den Zeitpunkt des nächsten Aufrufs.
- Wird sie dann zum nächsten Zyklus aufgerufen, vergleicht sie diesen Erwartungswert mit der tatsächlichen Uhrzeit.
- Die Abweichung wird wie u.a. dargestellt.

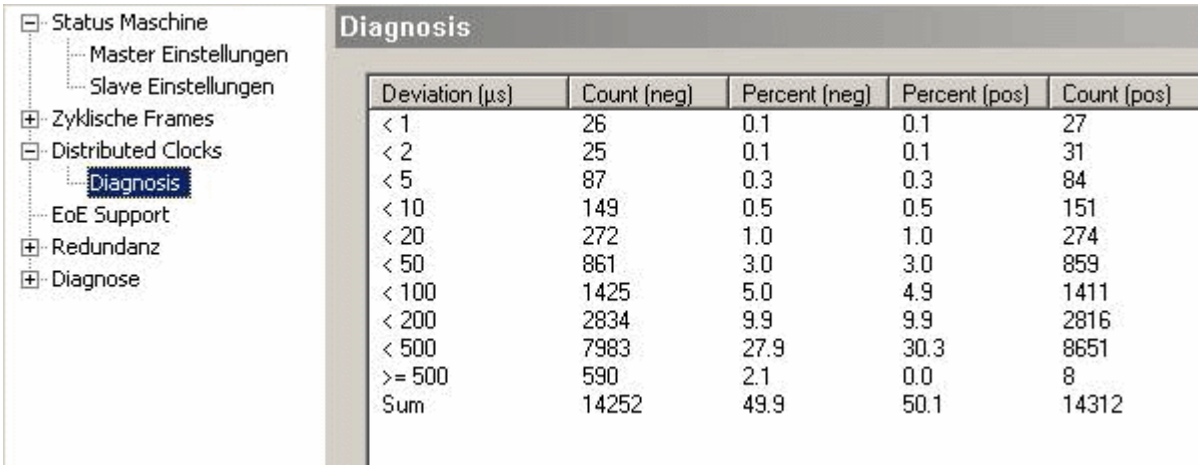


Deviation (µs)	Count (neg)	Percent (neg)	Percent (pos)	Count (pos)
< 1	68979	26.5	19.7	51355
< 2	56687	21.8	9.1	23605
< 5	25578	9.8	8.6	22491
< 10	256	0.1	4.0	10416
< 20	0	0.0	0.3	676
< 50	0	0.0	0.0	0
< 100	0	0.0	0.0	0
< 200	0	0.0	0.0	0
< 500	0	0.0	0.0	0
>= 500	0	0.0	0.0	0
Sum	151500	58.3	41.7	108543

Abb. 280: Online DC Diagnose

Kriterium	Gut	Schlecht
Asymmetrie	Eine Asymmetrie von positiven und negativen Abweichungen ist erforderlich. Dies bildet das Driftverhältnis zwischen Masterclock und TwinCAT-Clock ab.	Bei einem Verhältnis 0:100 oder 100:0 ist das DC-System außer Betrieb.
Verteilung der Deviation	Die Deviation-Werte sollten überwiegend in niedrigen Stufen stehen, s. Abb. *Online DC-Diagnose - Anpassungsbedarf*	Wenn ausschließlich in der Klasse ">=500 µs" Werte auftreten, ist das DC-System außer Betrieb.

Ein System nach Abb. *Online DC-Diagnose - Anpassungsbedarf* ist beispielsweise für schnelle EtherCAT-Applikationen mit z. B. 100 µs Zykluszeit nicht geeignet und verlangt ggf. nach Anpassungen.



Deviation (µs)	Count (neg)	Percent (neg)	Percent (pos)	Count (pos)
< 1	26	0.1	0.1	27
< 2	25	0.1	0.1	31
< 5	87	0.3	0.3	84
< 10	149	0.5	0.5	151
< 20	272	1.0	1.0	274
< 50	861	3.0	3.0	859
< 100	1425	5.0	4.9	1411
< 200	2834	9.9	9.9	2816
< 500	7983	27.9	30.3	8651
>= 500	590	2.1	0.0	8
Sum	14252	49.9	50.1	14312

Abb. 281: Online DC-Diagnose - Anpassungsbedarf

7.2.4 Distributed Clocks & TwinCAT PLC

Gültigkeit der nachstehenden Einstellungen

Nachfolgende Ausführungen basieren auf TwinCAT 2.11 build 1539. Neuere Ausgaben können eine abweichende Oberflächengestaltung aufweisen, die Verwendung bleibt aber sinngemäß die gleiche. Teilfunktionen können schon in Vorgängerversionen enthalten sein, es ist jedoch zu beachten, dass ggf. auf dem Ziel- wie auch dem Programmiersystem die gleiche TwinCAT Version (z. B. 2.11) vorliegen muss, um die aktuellsten Funktionen zu unterstützen.

Aus der Sicht der PLC ist TwinCAT 2.11 eine stetige Weiterentwicklung von TwinCAT 2.10. Die PLC-Bibliotheken wurden um weitere Features ergänzt. Für die Verwendung mit Distributed Clocks-Funktionen sind insbesondere folgende Bibliotheken von Interesse:

- TcUtilities.lib
  - Funktionen zur Behandlung von 64-Bit-Werten: UINT64 und INT64
  - Funktionen zum Auslesen versch. Zeitquellen
- TcEtherCAT.lib
  - Funktionen zur Manipulation von EtherCAT Geräten
  - Funktionen zum Auslesen der DC-Zeiten

Diese Bibliotheken sind in jeder TwinCAT-Installation enthalten.

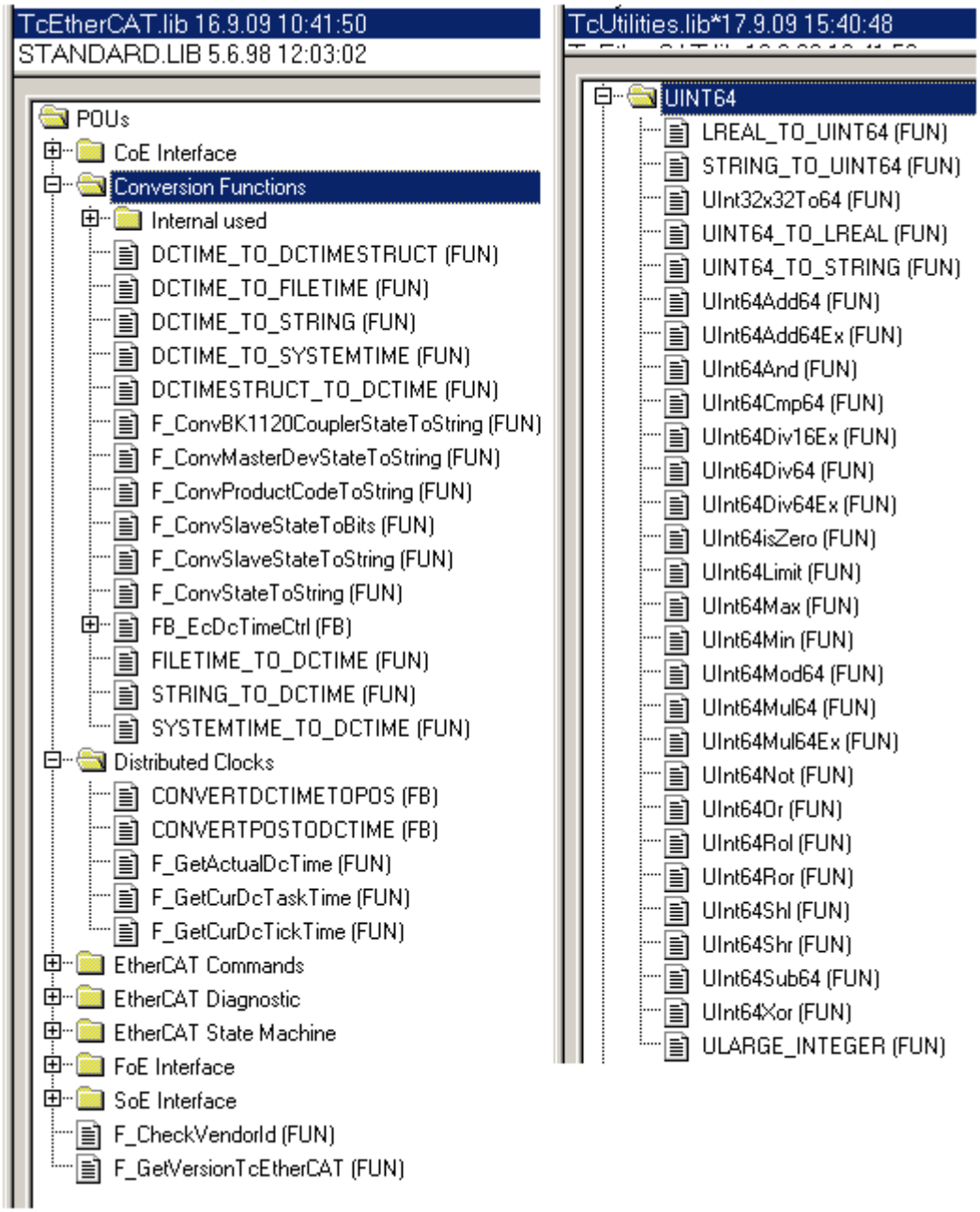


Abb. 282: PLC-Libraries

Arbeiten mit DC-Zeiten in der Steuerung

Die Distributed-Clock-Zeit hat aus der Sicht der Steuerung folgenden Eigenschaften:

- Einheit *1 ns*
- universaler Nullpunkt *1.1.2000 00:00* , d.h. bei Auswertungen der Variable ist ein Offset von 2000 Jahren zu addieren

- Umfang bis zu *64 Bit* (ausreichend für 584 Jahre); manche EtherCAT-Slaves unterstützen jedoch nur einen Umfang von 32 Bit, d.h. nach ca. 4,2 Sekunden läuft das Register lokal über und beginnt wieder bei 0.

Zur Bearbeitung von DC-Zeiten werden folgende 3 Datentypen empfohlen

- **T_DCTIME** aus TcEtherCAT.lib
  Basiert auf T_ULARGE_INTEGER und ist damit vorzeichenlos. Kann zur Verlinkung mit entsprechenden Hardware-Variablen verwendet werden
- **T_ULARGE_INTEGER** aus der TcUtilities.lib
  Vorzeichenloser 64-Bit-Datentyp
- **T_LARGE_INTEGER** aus der TcUtilities.lib
  Vorzeichenbehafteter 64-Bit-Datentyp, negative Zahlen werden im 2er-Komplement dargestellt (Unterlauf unter 0 --> 0xFFFF FFFF FFFF FFFF usw.)
  Dazu stehen in der TcUtilities.lib im Kapitel INT64 viele Funktionen zur Verfügung, wichtig insbesondere die cast-Funktionen LARGE_TO_ULARGE und umgekehrt.
  Sobald mit Zeitdifferenzen gearbeitet wird, bei denen negative Zeiten auftreten können, ist dieser Typ zu verwenden.
  Wird TwinCAT externe Synchronisierung eingesetzt, treten zwangsläufig negative Zeiten in den Offset-Werten auf.

64- vs. 32-Bit-Darstellung

Manche EtherCAT-Slaves können die DC-Zeit nur als 32-Bit-Wert darstellen bzw. als Prozessdatum verarbeiten. Um durch Überlauf verursachten Problemen (alle 4.2 Sekunden) entgegenzuwirken wird jedoch dringend empfohlen, in der Steuerung generell nur mit 64-Bit-Zeiten zu rechnen.

- An die PLC gelieferte 32-Bit-Zeiten sind um den aktuellen High-Teil zu erweitern
- An die HW ist in diesem Fall nur der Low-Part (untere 32 Bit) zu liefern

In diesem Beispielprojekt

(https://infosys.beckhoff.com/content/1031/ethercatsystem/Resources/zip/2469155979.zip) ist ein Funktionsblock enthalten, der zyklisch eine 32-Bit-DC-Zeit um den High-Part zu 64 Bit ergänzt.

Auslesen der DC-Zeit

Wie beschrieben [▶ 224], gibt es auf einem TwinCAT-PC mehrere Zeitquellen, deren maßgebliche für das EtherCAT-System die Distributed Clock Zeit ist. Üblicherweise ist der erste DC-fähige Slave die EtherCAT Masterclock, die Steuerung mit Ihrer Software-Ausprägung der DC-Clock wird dieser Masterclock nachsynchronisiert. Damit steht in der Steuerung die aktuelle DC-Zeit zur Verfügung.

In der PLC gibt es 4 Möglichkeiten, zur Laufzeit der Task an die aktuelle DC-Zeit, die "Jetzt"-Zeit zu gelangen.

- Auslesen der Eingangsvariable *DcSysTime* in den Eingangsdaten des EtherCAT-Device (siehe Beschreibung System Manager, nicht empfohlen)
- TcEtherCAT.lib: *F_GetActualDcTime*
- TcEtherCAT.lib: *F_GetCurDcTickTime*
- TcEtherCAT.lib: *F_GetCurDcTaskTime*

Kompatibilität

Die Funktionen F_GetActualDcTime und F_GetCurDcTaskTime können erst ab der TwinCAT Version 2.11 (auf Programmier- und Zielsystem) benutzt werden.

Beispiel:

Konfiguriert wird eine PLC-Task im System Manager mit 500 µs Zykluszeit und 100 µs Basistick (Base Time).

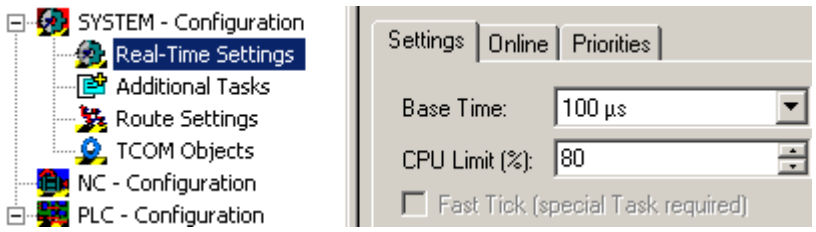


Abb. 283: Beispielkonfiguration mit 500 µs Zykluszeit

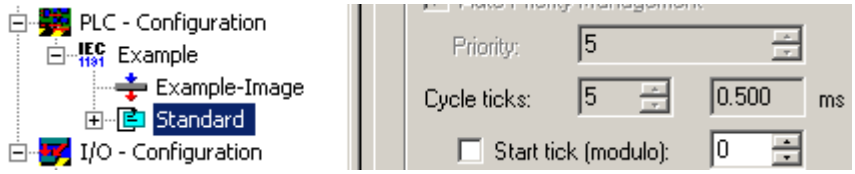


Abb. 284: Beispielkonfiguration mit 500 µs Zykluszeit

Die Bewertung der o.a. Beschaffungswege ermöglicht Abb. *Wege zum Auslesen der DC-Zeit*:

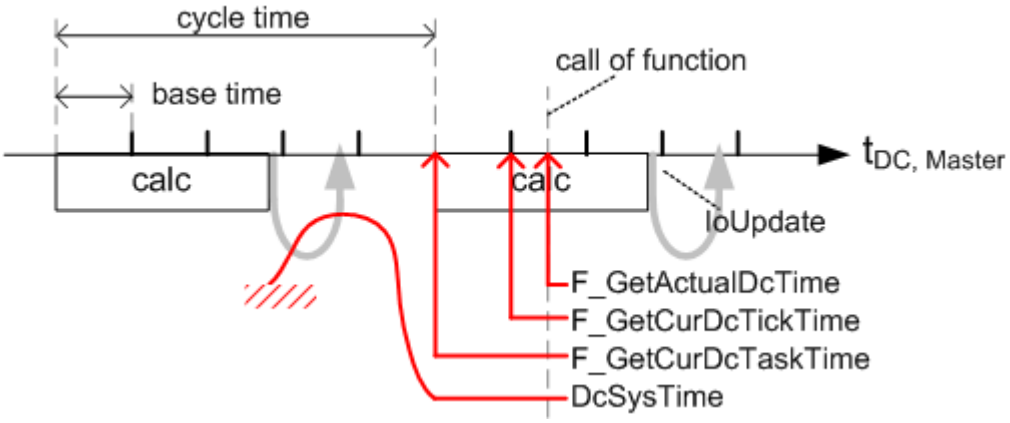


Abb. 285: Wege zum Auslesen der DC-Zeit

Zur Laufzeit des PLC-Programms werden nun die 3 Funktionen F_Get... an beliebiger Stelle aufgerufen, dann liefern sie ab TwinCAT 2.11 zurück:

- F_GetActualDcTime: die tatsächliche aktuelle DC-Zeit. Bei einem mehrmaligen Aufruf innerhalb einer Task liefert *F_GetActualDcTime* also auch unterschiedliche Werte zurück.
- F_GetCurDcTickTime: Zeitpunkt des letzten Basisticks. Ein mehrmaliger Aufruf liefert also nur dann unterschiedliche Werte zurück, wenn mind. 1x Basetime dazwischen liegt.
  Wenn BaseTime = Zykluszeit oder am Anfang einer Task aufgerufen, dann liefert diese Funktion das gleiche Ergebnis wie
- F_GetCurDcTaskTime: "Jetzt"-Zeit dieser Task, auf die sich die Slave-Shift-Zeiten beziehen.
  Die Verwendung dieser Funktion als Grundlage von DC-Operationen wird empfohlen.

Der Inhalt von *DcSysTime* kommt aus der feldseitigen Masterclock - dieser Kopiervorgang unterliegt zeitlichen Schwankungen und schreitet daher nicht exakt mit *cycle time* fort. Dies gilt ebenfalls für andere lokale DcSystem-Zeiten, die als Prozessdatum aus der HW übertragen werden.

7.2.5 Synchronisationsmodi eines EtherCAT-Slaves

Informationen für fortgeschrittene Anwender

Die nachfolgenden Informationen sind für den üblichen Anwendungsfall nicht notwendig! Erweitertes Grundwissen über CoE (CAN over EtherCAT) und tieferes Verständnis des EtherCAT-Protokoll sind nötig, um die nachfolgenden Informationen verwerten zu können.

HINWEIS

Achtung! Manipulation der Parameter kann ungewünschte Effekte bewirken!

Die unbedachte Manipulation der im Folgenden beschriebenen Parameter kann den EtherCAT-Slave in seiner Funktionalität behindern!

Ein EtherCAT-Slave ist ein elektronisches Gerät, das in einem bestimmten Zeitabstand (Takt) immer gleichlaufende Abfolgen von Berechnungen und/oder Datenkopieraktionen durchführt. Dieser Arbeitstakt im EtherCAT-Slave (z. B. im Bereich von wenigen µs bis einigen ms) kann aus verschiedenen Quellen abgeleitet werden. Insbesondere ist es möglich, mehrere EtherCAT-Slaves und/oder den EtherCAT-Master synchron zu betreiben, z. B. über das Verfahren der Distributed Clocks [▶ 234]. Entsprechende Beispiele wurden bereits auf der Einführungsseite zu den Distributed Clocks genannt.

Mit der Synchronisierung oder Abstimmung ist deshalb *nicht* eine physikalische Kopplung z. B. des Prozessortaktes elektronischer Geräte im MHz oder GHz-Bereich gemeint, sondern auf einer übergeordneten logischen Ebene werden Arbeitseinheiten nach den unten beschriebenen Parametern gestartet. Solche Arbeitseinheiten können sein: Ausgänge setzen, Eingänge lesen, Speicher kopieren, Berechnungen starten u.a.

Die folgenden Synchronisationsmodi werden in diesem Dokument beschrieben:

- Free Run [▶ 266] - der EtherCAT-Slave ist nicht mit EtherCAT synchronisiert. Der Slave arbeitet autonom nach seinem eigenen Takt und insbesondere nicht synchron mit dem EtherCAT-Zyklus.
- Synchron mit SM Event [▶ 268] - der EtherCAT-Slave ist mit dem SyncManager 2 (SM2) Event synchronisiert (wenn die zyklischen Ausgänge übertragen werden) oder mit dem SM3 Event (wenn nur die zyklischen Eingänge übertragen werden). Das SM2/SM3 Event wird vom SyncManager bei Bearbeitung eines durchlaufenden Frames ausgelöst.
- Synchron mit SYNC Event (Distributed Clocks) [▶ 271] - der EtherCAT-Slave ist mit SYNC0 oder SYNC1 Event des Distributed-Clocks-System synchronisiert. Dieser Anwendungsfall wurde auf den voranstehenden Seiten ausführlich beschrieben.

Alle nachstehenden Parameter sind Objekte im CoE-Verzeichnis des EtherCAT-Slaves. Diese können online vom Slave ausgelesen werden oder offline über den Slave beschreibende XML-Datei ermittelt werden.

Präsenz der CoE-Objekte

Auch wenn ein EtherCAT-Slave die nachstehend beschriebenen Verfahren unterstützt, muss er deshalb nicht unbedingt die beschriebenen Parameter für den Anwender oder den Master über ein CoE-Verzeichnis zugänglich gemacht haben! Nur EtherCAT-Slaves mit ausreichend "Intelligenz" verfügen über die Fähigkeit, ein CoE-Verzeichnis zu verwalten.
Insbesondere verfügen einige Beckhoff EtherCAT Klemmen über kein CoE-Verzeichnis - die nachfolgenden Parameter können in diesen EtherCAT-Slaves also weder angezeigt noch manipuliert werden. Dennoch können die Funktionen verfügbar sein, da sie dann alternativ über interne Register verwaltet werden.

Bestimmung des Synchronisationsmodus

Die unterschiedlichen Synchronisationsmodi können anhand der verschiedenen Kombinationen der Subindizes 0x1C32 und 0x1C33 bestimmt werden.

"--" innerhalb der Tabelle zeigt an, dass der Subindex entweder nicht benutzt wird, "0" sein kann oder nicht existiert.

	Sync Mode	Synchronization Type 0x1C32:01	Synchronization Type 0x1C33:01	Output Shift Time 0x1C32:03	Input Shift Time 0x1C33:03	Calc and Copy Time 0x1C32:06	Calc and Copy Time 0x1C33:06	Delay Time (0x1C32:09)	Delay Time (0x1C33:09)	Fixed SYNC0 Cycle Time
	1 Free Run Mode									
1	**Free Run**	0x00	0x00	--	--	--	--	--	--	--
	2 SM Event Mode									
2	**SM2 ***	0x01	0x22	--	--	--	--	--	--	--
2	**SM3 ***	--	0x01	--	--	--	--	--	--	--
3	**SM2, Shift Input Latch ***	0x01	0x22	--	!=0 **	--	!=0	--	--	--
3	**SM3, Shift Input Latch ***	--	0x01	--	!=0 **	--	!=0	--	--	--
	3 DC Mode									
4	**DC**	0x02	0x02	--	--	!=0	!=0	!=0	--	--
5	**DC, Shift Outputs Valid and Input Latch with Shift**	0x02	0x02	!=0 **	!=0 **	!=0	!=0	!=0	!=0	--
5	**DC, Shift of Outputs Valid with SYNC1**	0x03	0x02	!=0 ***	!=0 **	!=0	!=0	!=0	!=0	--
5	**DC, Shift of Input Latch with SYNC1**	0x02	0x03	!=0 **	!=0 ***	!=0	!=0	!=0	!=0	--
	4 Subordinated Application Controller Cycles									
4	**DC, Shift Outputs Valid/ Input Latch**	0x03	0x02 or 0x03	!=0 **	!=0 **	!=0	!=0	!=0	!=0	!=0

Tabelle 1: Bestimmung des Synchronisationsmodus

* Falls Ausgänge verfügbar sind, wird der Slave grundsätzlich mit dem SM2 Event synchronisiert. Falls keine Ausgänge verfügbar sind, wird der Slave mit dem SM3 Event synchronisiert.

** Veränderbar, wenn die Klemme mit variablen Shiftzeiten arbeiten kann

*** Shiftzeit kann überschrieben werden mit (SYNC1 Cycle Time + Delay Time)

Terminologie

Copy and Prepare Outputs

Mit einem Trigger Event (Local Timer Event, SM2/3 Event oder SYNC0/1 Event) werden Ausgangsdaten aus dem SyncManager Ausgangsdatenbereich gelesen und ggf. mathematische Berechnungen mit diesen Ausgangswerten durchgeführt. Danach wird das physikalische Ausgangssignal generiert und mit der "Outputs Valid" Kennzeichnung für den Prozess zur Verfügung gestellt.

"Copy and Prepare Outputs" beschreibt die Summe der Zeit für das Kopieren von Prozessdaten vom SyncManager in den lokalen Speicher, ggf. die Durchführung von weiteren mathematischen Berechnungen und Hardware-Verzögerungen (abhängig von der Implementierung einschließlich Software Verarbeitungszeit). Die einzelnen Zeiten werden nicht weiter bestimmt. Sie entsprechen den Werten die im SyncManager Objekt 0x1C32 beschrieben sind:

Beschriebene Zeit	SyncManager Objekt 0x1C32
Kopieren von Prozessdaten aus dem SyncManager und mathematische Berechnungen	Calc and Copy Time (0x1C32:06)
Hardware Verzögerungszeit	Delay Time (0x1C32:09)

"Get and Copy Inputs"

"Get and Copy Inputs" summiert die Summe der Zeiten für Hardware-Verzögerungen beim Lesen des Eingangssignals, ggf. für die Ausführung von mathematischen Berechnungen und für den Kopiervorgang der Eingangsprozessdaten in den Eingangsdatenbereich des SyncManger 3. Die einzelnen Zeiten werden nicht weiter bestimmt. Sie entsprechen den Werten die im SyncManager Objekt 0x1C33 beschrieben sind:

Beschriebene Zeit	SyncManager Objekt 0x1C32
Mathematische Berechnungen und Kopieren von Prozessdaten vom lokalen Speicher zum SyncManager	Calc and Copy Time (0x1C33:06)
Hardware-Verzögerung bis zum "Input Latch"	Delay Time (0x1C33:09)

Die Eingangswerte sind im Eingangsdatenbereich des SyncManger 3 nach der Min Cycle Time (0x1C32:05) verfügbar.

Outputs Valid
Mit dem "Outputs Valid"-Zeitpunkt sind die Ausgänge (z. B. als elektrisches Signal) für den Prozess verfügbar.

Start Driving Outputs
Beim "Start Driving Outputs"-Zeitpunkt hat der µC seine Ausgänge gesetzt. Die Hardware "Delay Time" (0x1C32:09) ist die Verzögerung zwischen "Start Driving Outputs" und "Outputs Valid"-Zeitpunkt.

Start Latch
Der "Start Latch"-Zeitpunkt kennzeichnet den Start des "Input Latch"-Prozesses. Zwischen "Start Latch"- und "Input Latch"-Zeitpunkt wird eine Verzögerung durch die Hardware, durch Abhängigkeiten bei der Slave-Implementierung, sowie durch Softwareverarbeitungszeit beschrieben und in der "Delay Time" 0x1C33:09 abgebildet.

Input Latch
Beim "Input Latch"-Zeitpunkt ist das Erfassen der Eingangsdaten abgeschlossen. Zu diesem Zeitpunkt sind evt. mathematische Berechnungen noch nicht durchgeführt und die Daten noch nicht in den Datenbereich des SyncManagers kopiert.

User Shift Time
Die "User Shift Time" beschreibt den Jitter des Masters.

SYNC1 Cycle Time
Die "SYNC1 Cycle Time" kann zur Verschiebung des "Start Input Latch" oder "Start Driving Outputs" verwendet werden. Die "SYNC1 Cycle Time" ist im Register 0x0984:0x0987 dargestellt. Sie beschreibt die die Verschiebung zwischen dem SYNC0 and SYNC1 Signal (SYNC0 ist immer das Referenzsignal)

Shift Time
Die "Shift Time" beschreibt die Zeit zwischen den Sync Events (SM2 Event, SM3 Event, SYN0, SYNC1) und dem Zeitpunkten "Outputs Valid" oder "Input Latch". Beschreibbarer Wert, falls der Slave die Verschiebung von "Outputs Valid" oder "Input Latch" unterstützt.

Betriebsart 1 - Free Run

Im "Free Run"- Modus wird der lokale Zyklus durch einen lokalen Timer-Interrupt des Application-Controllers ausgelöst. Die Zykluszeit kann vom Master geändert werden (optional) um den Timer-Interrupt zu ändern. Im "Free Run" -Modus arbeitet der lokale Zyklus unabhängig vom Kommunikationszyklus und /oder vom Master-Zyklus.

Optionale Features

Der Slave kann eine variable "Cycle Time" (0x1C32:02 veränderbar). In diesem Fall ist auch "Minimum Cycle Time" (0x1C32:05) variabel.

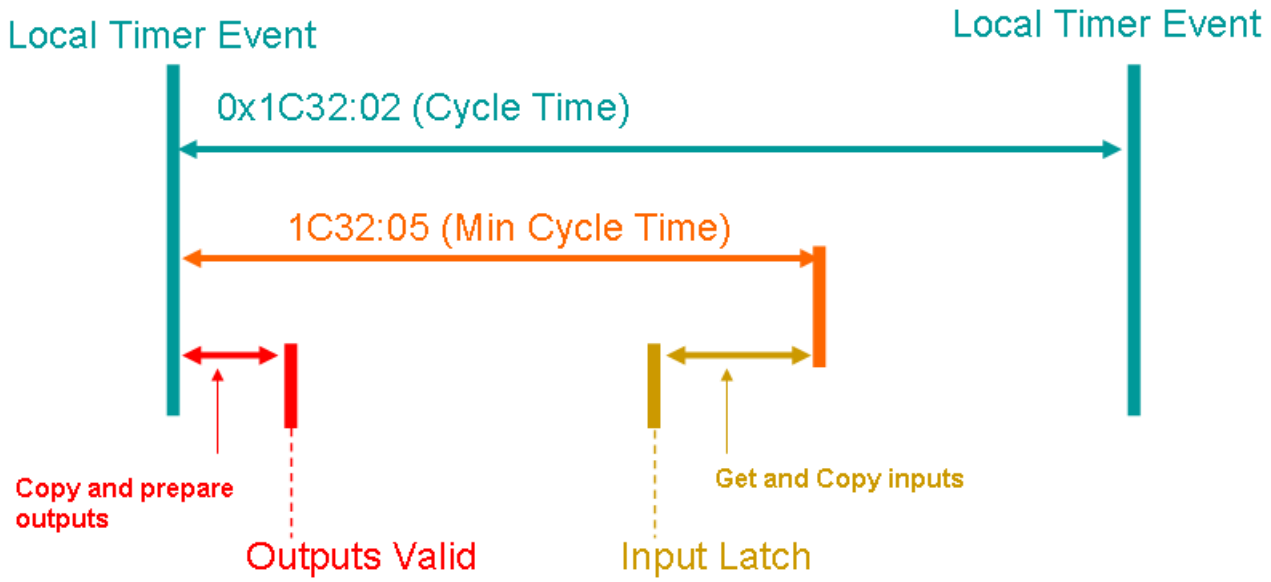


Abb. 286: Lokaler Zyklus "Free Run" Synchronisation

Die Tabellen "0x1C32 Free Run" and "0x1C33 Free Run" erläutern die Anwendung dieser Objekte im "Free Run"- Modus.

Subindex	Beschreibung	Flag	Verwendung	Beschreibung/ Default Wert
1	Synchronization Type	r oder rw	erforderlich	0x00: Free Run
2	Cycle Time	r oder rw	optional	Lokale Zyklus Zeit vom Application Controller
3	Shift Time	--	--	
4	Synchronization Types supported	r	erforderlich	Bit 0: Free Run unterstützt
5	Minimum Cycle Time	r	bedingt	erforderlich falls 0x1C32:02 variabel
6	Calc and Copy Time	--	--	
7	--	--	--	
8	Get Cycle Time	--	--	
9	Delay Time	--	--	
10	SYNC0 Cycle Time	--	--	
11	Cycle Time Too Small	--	--	
12	SM-Event missed	--	--	
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	--	--	
31:15	--	--	--	
32	Sync Error	--	--	

Tabelle 2: 0x1C32 Free Run

Subindex	Beschreibung	Flag	Verwendung	Beschreibung/ Default Wert
1	Synchronization Type	r or rw	erforderlich	0x00: Free Run
2	Cycle Time	r or rw	optional	Gleicher Wert wie 0x1C32:02
3	Shift Time	--	--	
4	Synchronization Types supported	r	erforderlich	Gleicher Wert wie 0x1C32:04
5	Minimum Cycle Time	r	bedingt	Gleicher Wert wie 0x1C32:05
6	Calc and Copy Time	--	--	
7	--	--	--	
8	Get Cycle Time	--	--	
9	Delay Time	--	--	
10	SYNC0 Cycle Time	--	--	
11	Cycle Time Too Small	--	--	
12	SM-Event missed	--	--	
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	--	--	
31:15	--	--	--	
32	Sync Error	--	--	

Tabelle 3: 0x1C33 Free Run

Betriebsart 2 - Synchron mit SM Event

Der lokale Zyklus wird gestartet, wenn der SM2 Event [mit zyklischen Ausgängen] bzw. der SM3 Event [ohne zyklische Ausgänge] empfangen wird.
Wenn die Ausgänge zur Verfügung stehen, wird der Slave grundsätzlich auf den SM2 Event synchronisiert. Stehen keine Ausgänge zur Verfügung, wird der Slave auf den SM3 Event z. B. für zyklische Eingänge synchronisiert.

In dieser Betriebsart sind möglich

- Synchron mit SM2/3 Event [▶ 268]
- Synchron mit SM2/3 Event, Verschiebung des "Input Latch"-Zeitpunktes [▶ 270]

Synchron mit SM2/3 Event

Der lokale Zyklus wird gestartet, wenn der SM2/3 Event empfangen wird.

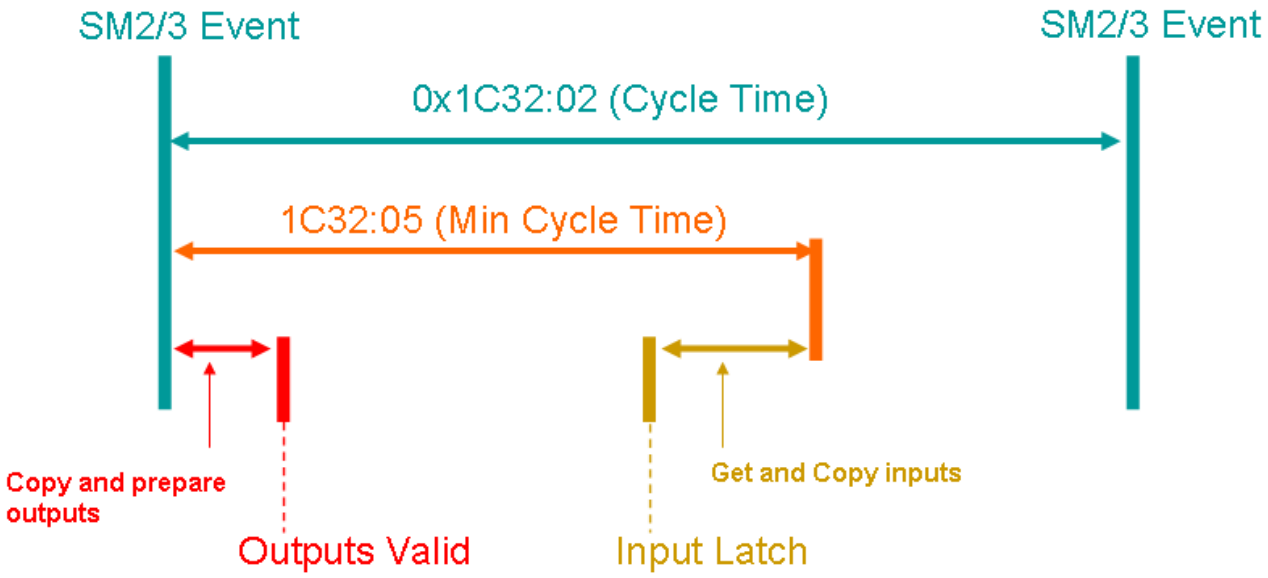


Abb. 287: Lokaler Zyklus mit Synchronisation auf SM2/3 Event

Die Tabellen "0x1C32 Synchron mit SM 2/3 Event" und "0x1C33 Synchron mit SM 2/3 Event" erläutern die Anwendung dieser Objekte im Modus "Synchron mit SM 2/3 Event".

Subindex	Beschreibung	Flag	Verwendung	Beschreibung/ Default Wert
1	Synchronization Type	r oder rw	erforderlich	0x01: Synchron – synchronisiert mit SM 2 Event
2	Cycle Time	r oder rw	optional	Kommunikationszykluszeit
3	Shift Time	--	--	
4	Synchronization Types supported	r	erforderlich	Bit 1: Synchron SM unterstützt
5	Minimum Cycle Time	r	erforderlich	
6	Calc and Copy Time	--	--	
7	--	--	--	
8	Get Cycle Time	rw	bedingt****	
9	Delay Time	--	--	
10	SYNC0 Cycle Time	--	--	
11	Cycle Time Too Small	r	erforderlich	
12	SM-Event Missed	r	optional	
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	r	optional	
31:15	--	--	--	
32	Sync Error	r	bedingt	wird unterstützt, wenn "SM-Event Missed" Counter verwendet wird

**** wird im Synchron Modus oder in DC Mode mit variabler Zykluszeit verwendet

Tabelle 4: 0x1C32 Synchron mit SM 2/3 Event

Subindex	Beschreibung	Flag	Verwendung	Beschreibung/ Default Wert
1	Synchronization Type	r oder rw	erforderlich	0x01: Synchron - synchronisiert mit SM 3 Event (wenn Übertragung der Eingänge in SAFE-OP und OP Status) 0x22: Synchron mit SM2 Event (wenn Übertragung der Ausgänge in SAFE-OP und OP Status)
2	Cycle Time	r or rw	optional	Gleicher Wert wie 0x1C32:02
3	Shift Time	--	--	
4	Synchronization Types supported	r	erforderlich	Gleicher Wert wie 0x1C32:04
5	Minimum Cycle Time	r	erforderlich	Gleicher Wert wie 0x1C32:05
6	Calc and Copy Time	--	--	
7	--	--	--	
8	Get Cycle Time	rw	bedingt****	Gleicher Wert wie 0x1C32:08
9	Delay Time	--	--	
10	SYNC0 Cycle Time	--	--	
11	Cycle Time Too Small	r	erforderlich	Gleicher Wert wie 0x1C32:0B
12	SM-Event missed	r	optional	Gleicher Wert wie 0x1C32:0C
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	r	optional	Gleicher Wert wie 0x1C32:0E
31:15	--	--	--	
32	Sync Error	r	bedingt	Gleicher Wert wie 0x1C32:20

**** wird im Synchron Modus oder in DC Modus mit variabler Zykluszeit verwendet

Tabelle 5: 0x1C33 Synchron mit SM 2/3 Event

Synchron mit SM2/3 Event, Verschiebung des "Input Latch"-Zeitpunktes

Die Eingangsdaten sollten so zeitnah wie möglich zum nächsten SM2/3 Event erfasst werden, um die aktuellsten Eingangsdaten dem Kontrollsystem (Master) zur Verfügung zu stellen. Dies kann mit der Verschiebung des "Input Latch"-Zeitpunktes näher an den SM2/3 Event durch Veränderung der "Shift Time" (0x1C33:03) erreicht werden.

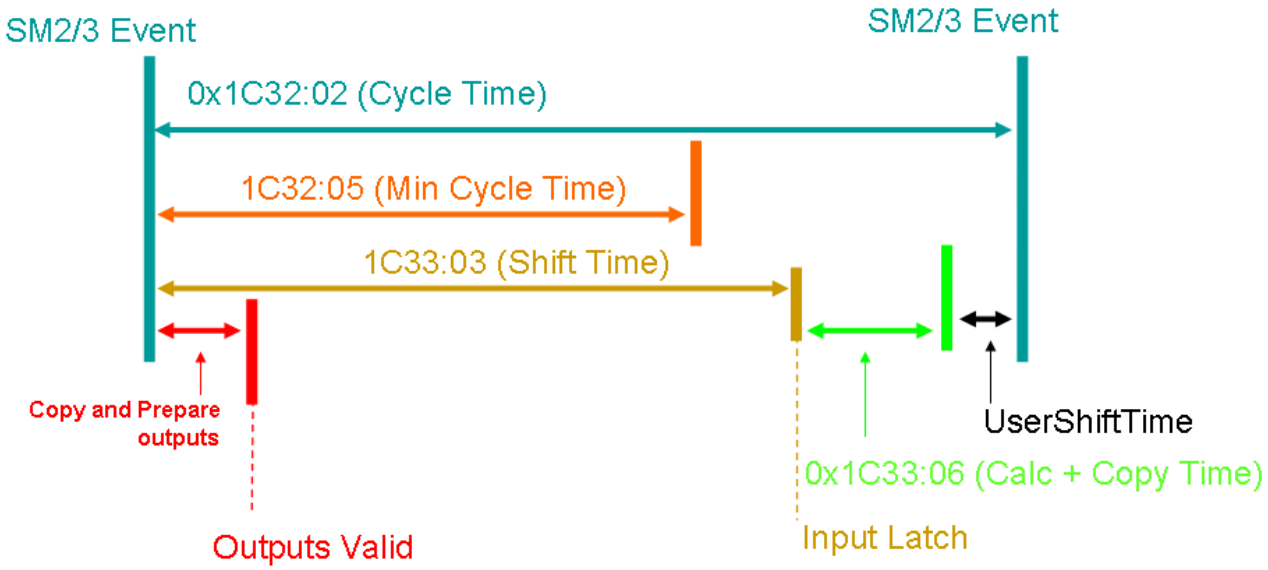


Abb. 288: Lokaler Zyklus mit Synchronisation auf SM2/3 Event, Verschiebung "Input Latch"

Die Tabellen "0x1C32 Synchron mit SM 2/3 Event, Verschiebung Input Latch" und "0x1C33 Synchron mit SM 2/3 Event, Verschiebung Input Latch" erläutern die Anwendung dieser Objekte im Modus "Synchron mit SM 2/3 Event bei Verschiebung des "Input Latch".

Subindex	Beschreibung	Flag	Verwendung	Beschreibung/ Default Wert
1	Synchronization Type	r or rw	erforderlich	0x01: Synchron - synchronisiert mit SM 2/3 Event
2	Cycle Time	r or rw	optional	Kommunikationszykluszeit
3	Shift Time	--	--	
4	Synchronization Types supported	r	erforderlich	Bit 1: Synchron SM unterstützt
5	Minimum Cycle Time	r	erforderlich	
6	Calc and Copy Time	--	--	
7	--	--	--	
8	Get Cycle Time	rw	bedingt****	
9	Delay Time	--	--	
10	SYNC0 Cycle Time	--	--	
11	Cycle Time Too Small	r	erforderlich	
12	SM-Event missed	r	optional	
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	r	optional	
31:15	--	--	--	
32	Sync Error	r	bedingt	wird unterstützt, wenn SM-Event Missed Counter verwendet wird

**** wird im Synchron Modus oder in DC Modus mit variabler Zykluszeit verwendet

Tabelle 6: 0x1C32 Synchron mit SM 2/3 Event, Verschiebung "Input Latch"

Subindex	Beschreibung	Flag	Verwendung	Beschreibung/ Default Wert
1	Synchronization Type	r or rw	erforderlich	0x01: Synchron - synchronisiert mit SM 3 Event (wenn nur Eingänge verfügbar) 0x22: Synchron mit SM2 Event (wenn Ausgänge verfügbar)
2	Cycle Time	r or rw	optional	Gleicher Wert wie 0x1C32:02
3	Shift Time	rw	erforderlich	
4	Synchronization Types supported	r	erforderlich	Gleicher Wert wie 0x1C32:04
5	Minimum Cycle Time	r	erforderlich	Gleicher Wert wie 0x1C32:05
6	Calc and Copy Time	r	erforderlich	
7	--	--	--	
8	Get Cycle Time	rw	bedingt****	Gleicher Wert wie 0x1C32:08
9	Delay Time	--	--	
10	SYNC0 Cycle Time	--	--	
11	Cycle Time Too Small	r	erforderlich	Gleicher Wert wie 0x1C32:0B
12	SM-Event missed	r	optional	Gleicher Wert wie 0x1C32:0C
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	r	optional	Gleicher Wert wie 0x1C32:0E
31:15	--	--	--	
32	Sync Error	r	bedingt	Gleicher Wert wie 0x1C32:20

**** wird im Synchron Modus oder in DC Modus mit variabler Zykluszeit verwendet

Tabelle 7: 0x1C33 Synchron mit SM 2/3 Event, Verschiebung "Input Latch"

Betriebsart 3 - DC Modus (Distributed Clock Modus)

Die Synchronisierung mehrerer EtherCAT-Slaves untereinander und mit dem EtherCAT-Master wurde auf den vorangehenden Seiten bereits praxisnah beschrieben. Hier folgen nun Angaben über die internen Parameter in den Betriebsarten

- DC Modus (Synchron mit SYNC0 Event) [▶ 271]
- DC Modus, Verschiebung von "Outputs Valid" und /oder "Input Latch" [▶ 273]
- Distributed Clocks - Applikation mit untergeordnetem Controller Zyklus [▶ 278]

DC Modus (Synchron mit SYNC0 Event)

Der lokale Zyklus wird gestartet wenn der SYNC0 Event empfangen wird. Der Prozessdatenrahmen muss im Slave komplett verarbeitet werden bevor der nächste SYNC0 Event empfangen wird.

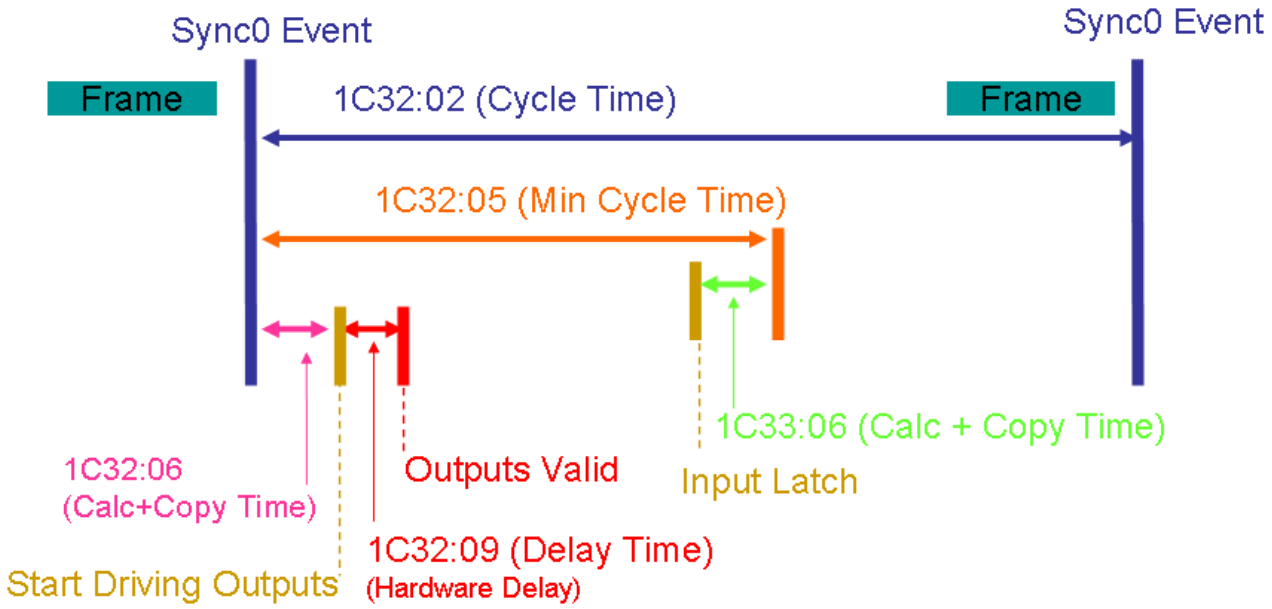


Abb. 289: Lokaler Zyklus mit Synchronisation auf SYNC0 Event

Die Tabellen "0x1C32 DC Modus" und "0x1C33 DC Modus" erläutern die Anwendung dieser Objekte im DC Modus.

Subindex	Beschreibung	Flag	Verwendung	Beschreibung/ Default Wert
1	Synchronization Type	r or rw	erforderlich	0x02: DC SYNC0 – synchronisiert mit SYNC0 Event
2	Cycle Time	r	optional	SYNC0 Zykluszeit (Register 0x09A3:0x09A0) Zeit zwischen zwei SYNC0 Events SYNC0 Zykluszeit wird in diesen Index eingetragen
3	Shift Time	--	--	
4	Synchronization Types supported	r	erforderlich	Bit 3:2 : DC unterstützt: 01 = DC
5	Minimum Cycle Time	r	erforderlich	
6	Calc and Copy Time	r	erforderlich	
7	--	--	--	
8	Get Cycle Time	rw	bedingt****	
9	Delay Time	r	erforderlich	
10	SYNC0 Cycle Time	--	--	
11	Cycle Time Too Small	r	erforderlich	
12	SM-Event missed	r	optional	
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	r	optional	
31:15	--	--	--	
32	Sync Error	r	bedingt	wird unterstützt, wenn SM-Event Missed Counter verwendet wird

**** wird im Synchron Modus oder in DC Modus mit variabler Zykluszeit verwendet

Tabelle 8: 0x1C32 DC Modus

Subindex	Beschreibung	Flag	Verwendung	Beschreibung/ Default Wert
1	Synchronization Type	r	erforderlich	0x02: DC SYNC0 – synchronisiert mit SYNC0 Event
2	Cycle Time	r	optional	Gleicher Wert wie 0x1C32:02
3	Shift Time	--	--	
4	Synchronization Types supported	r	erforderlich	Gleicher Wert wie 0x1C32:04
5	Minimum Cycle Time	r	erforderlich	Gleicher Wert wie 0x1C32:05
6	Calc and Copy Time	r	erforderlich	
7	--	--	--	
8	Get Cycle Time	rw	bedingt****	Gleicher Wert wie 0x1C32:08
9	Delay Time	--	--	
10	SYNC0 Cycle Time	--	--	
11	Cycle Time Too Small	r	erforderlich	Gleicher Wert wie 0x1C32:0B
12	SM-Event missed	r	optional	Gleicher Wert wie 0x1C32:0C
13	Shift Time Too Short	--	--	
14	RxPDO Toggle Failed	r	optional	Gleicher Wert wie 0x1C32:0E
31:15	--	--	--	
32	Sync Error	r	bedingt	Gleicher Wert wie 0x1C32:20

**** wird im Synchron Modus oder in DC Modus mit variabler Zykluszeit verwendet

Tabelle 9: 0x1C33 DC Modus

DC Modus, Verschiebung von "Outputs Valid" und /oder "Input Latch"

Der "Outputs Valid"-Zeitpunkt kann entweder durch die "Shift Time" (0x1C32:03 = 0x02) oder den SYNC1 Event (0x1C32:01 = 0x03) verzögert werden. Die "Shift Time" beschreibt die Zeit zwischen dem SYNC0 Event und "Outputs Valid"-Zeitpunkt während die SYNC1 Zykluszeit die Zeit bis zum "Start Driving Outputs"-Zeitpunkt beschreibt.

Der "Input Latch"-Zeitpunkt kann entweder durch die "Shift Time" (0x1C33:03 = 0x02) oder den SYNC1 Event (0x1C33:01 = 0x03) verzögert werden. In diesem Fall beschreibt die "Shift Time" die Zeit zwischen dem SYNC0 Event und "Input Latch"-Zeitpunkt während die SYNC1 Zykluszeit die Zeit bis zum "Start Latch"-Zeitpunkt beschreibt.

Das SYNC1 Signal kann entweder zur Verschiebung des "Outputs Valid"- oder "Input Latch"-Zeitpunkt verwendet werden.

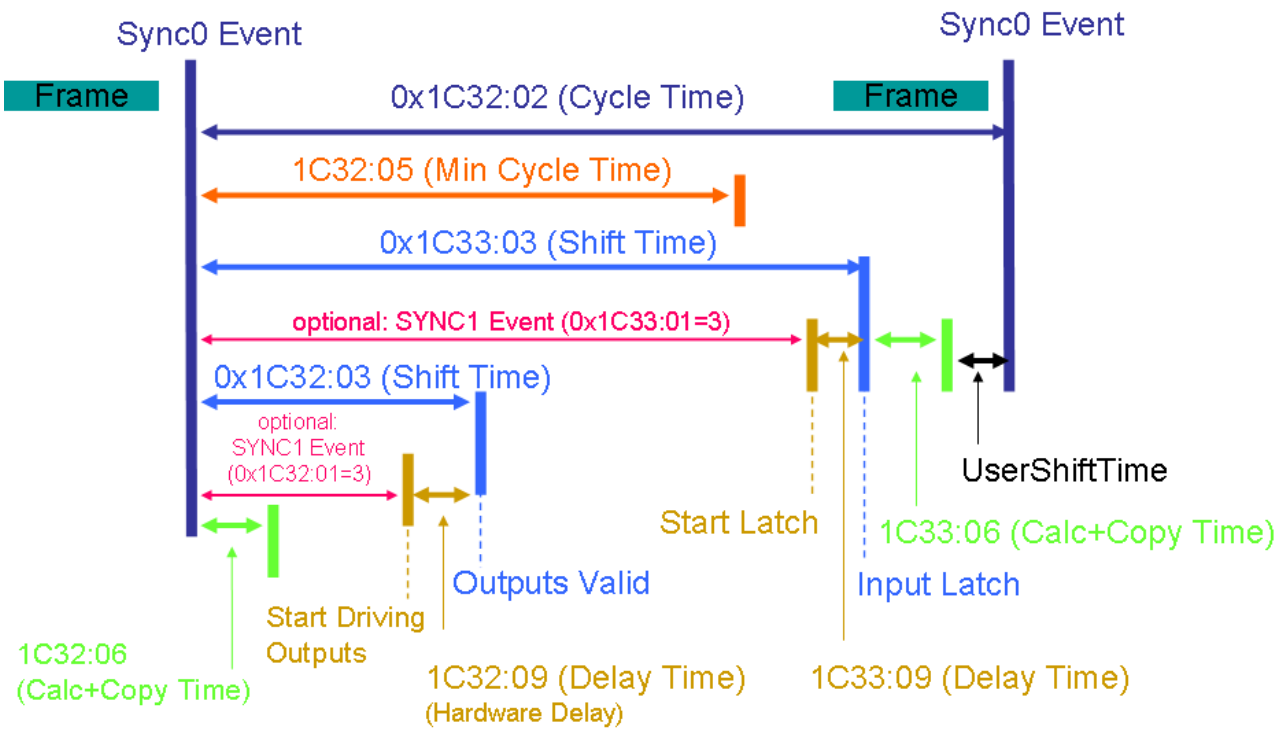


Abb. 290: Lokaler Zyklus mit Synchronisation auf SYNC0 Event, Verschiebung der Eingänge/Ausgänge

Die Tabellen "0x1C32 DC Modus, *Verschiebung der Eingänge/Ausgänge*" und "0x1C33 DC Modus, *Verschiebung der Eingänge/Ausgänge*" erläutern die Anwendung dieser Objekte im DC Modus bei Verschiebung der Eingänge/Ausgänge.

Subindex	Beschreibung	Flag	Verwendung	Beschreibung/ Default Wert
1	Synchronization Type	r or rw	erforderlich	Wenn "Shift Time" zur Verzögerung des "Outputs Valid"-Zeitpunktes verwendet wird: 0x02: DC SYNC0 – synchronisiert mit SYNC0 Event Wenn "SYNC1" Signal zur Verzögerung des "Outputs Valid"-Zeitpunktes verwendet wird: 0x03: DC SYNC1 – synchronisiert mit SYNC1 Event
2	Cycle Time	r	erforderlich	SYNC0 Zykluszeit (Register 0x09A3:0x09A0) Zeit zwischen zwei SYNC0 Events SYNC0 Zykluszeit wird in diesen Index eingetragen
3	Shift Time	r or rw	erforderlich	Eintrag beschreibbar, wenn "Shift Time" zur Verschiebung des "Outputs Valid"-Zeitpunktes verwendet wird Wenn SYNC1 zur Verschiebung des "Outputs Valid"-Zeitpunktes verwendet wird, kann die SYNC1 Zykluszeit + Verzögerungszeit [Delay Time (0x1C32:09)] von der Slave Anwendung in diesen Eintrag kopiert werden.
4	Synchronization Types supported	r	erforderlich	Bit 3:2 : DC unterstützt: 01 = Normal DC Bit 5:4: Shift Einstellungen 00 = Keine Ausgangszeitverschiebung Wenn "Shift Time" zur Verschiebung des "Outputs Valid"-Zeitpunktes verwendet wird: 01 = Ausgangszeitverschiebung mit lokalem Timer (Shift Time) Wenn "SYNC1" Signal zur Verschiebung des "Outputs Valid"-Zeitpunktes verwendet wird: 10 = Ausgangszeitverschiebung mit SYNC1
5	Minimum Cycle Time	r	erforderlich	
6	Calc and Copy Time	r	erforderlich	
7	--	--	--	
8	Get Cycle Time	rw	bedingt****	
9	Delay Time	r	erforderlich	
10	SYNC0 Cycle Time	--	--	
11	Cycle Time Too Small	r	erforderlich	
12	SM-Event missed	r	optional	
13	Shift Time Too Short	r	optional	
14	RxPDO Toggle Failed	r	optional	
31:15	--	--	--	

Subindex	Beschreibung	Flag	Verwendung	Beschreibung/ Default Wert
32	Sync Error	r	bedingt	wird unterstützt, wenn "SM-Event Missed" oder "Shift Time Too Short " Counter verwendet wird

**** wird im Synchron Modus oder in DC Modus mit variabler Zykluszeit verwendet

Tabelle 10: 0x1C32 DC Modus, Verschiebung der Eingänge/Ausgänge

Subindex	Beschreibung	Flag	Verwendung	Beschreibung/ Default Wert
1	Synchronization Type	r or rw	erforderlich	Wenn "Shift Time" zur Verzögerung des "Input Latch"-Zeitpunktes verwendet wird: 0x02: DC SYNC0 – synchronisiert mit SYNC0 Event Wenn "SYNC1" Signal zur Verzögerung des "Input Latch"-Zeitpunktes verwendet wird: 0x03: DC SYNC1 – synchronisiert mit SYNC1 Event
2	Cycle Time	r	erforderlich	Gleicher Wert wie 0x1C32:02
3	Shift Time	r or rw	erforderlich	Eintrag beschreibbar, wenn "Shift Time" zur Verschiebung des "Input Latch"-Zeitpunktes verwendet wird Wenn SYNC1 zur Verschiebung des "Input Latch"-Zeitpunktes verwendet wird, kann die SYNC1 Zykluszeit + Verzögerungszeit [Delay Time (0x1C33:09)] von der Slave Anwendung in diesen Eintrag kopiert werden.
4	Synchronization Types supported	r	erforderlich	Bit 3:2 : DC unterstützt: 01 = DC Bit 5:4: Shift Einstellungen 00 = Keine Eingangszeitverschiebung Wenn "Shift Time" zur Verschiebung des "Input Latch"-Zeitpunktes verwendet wird: 01 = Eingangszeitverschiebung mit lokalem Timer (Shift Time) Wenn "SYNC1" Signal zur Verschiebung des "Outputs Valid"-Zeitpunktes verwendet wird: 10 = Eingangszeitverschiebung mit SYNC1
5	Minimum Cycle Time	r	erforderlich	Gleicher Wert wie 0x1C32:05
6	Calc and Copy Time	r	erforderlich	
7	--	--	--	
8	Get Cycle Time	rw	bedingt****	Gleicher Wert wie 0x1C32:08
9	Delay Time	r	bedingt	Wird verwendet, wenn "SYNC1" Signal zur Verschiebung des "Input Latch"-Zeitpunktes verwendet wird
10	SYNC0 Cycle Time	--	--	
11	Cycle Time Too Small	r	erforderlich	Gleicher Wert wie 0x1C32:0B
12	SM-Event missed	r	optional	Gleicher Wert wie 0x1C32:0C
13	Shift Time Too Short	r	optional	Gleicher Wert wie 0x1C32:0D
14	RxPDO Toggle Failed	r	optional	Gleicher Wert wie 0x1C32:0E
31:15	--	--	--	
32	Sync Error	r	bedingt	Gleicher Wert wie 0x1C32:20

**** wird im Synchron Modus oder in DC Modus mit variabler Zykluszeit verwendet

Tabelle 11: 0x1C33 DC Modus, Verschiebung der Eingänge/Ausgänge

Distributed Clocks - Applikation mit untergeordnetem Controller Zyklus

Bei Slaves mit schnellen lokalen Zykluszeiten (z. B. Control Loops) kann die Zykluszeit des Applikationscontrollers deutlich schneller sein als die Kommunikationszykluszeit und/oder die SYNC0 Zykluszeit. Ein Anwendungsfall sind die Beckhoff XFC-Oversampling-Klemmen.

In diesem Fall werden zwei Synchronisationsfeatures angewendet:

1. Verschiebung des "Outputs Valid"-Zeitpunktes
2. Verschiebung des "input Latch"-Zeitpunktes

DC, untergeordneter µC Zyklus, Verschiebung von "Outputs Valid" und/oder "Input Latch"

Das "SYNC0" Signal wird als Trigger des lokalen µC Zyklus verwendet. Der "Outputs Valid"- und "Input Latch"-Zeitpunkt wird vom SYNC1 Event getriggert und kann nur von der "Output Shift Time" oder "Input Shift Time" verschoben werden.

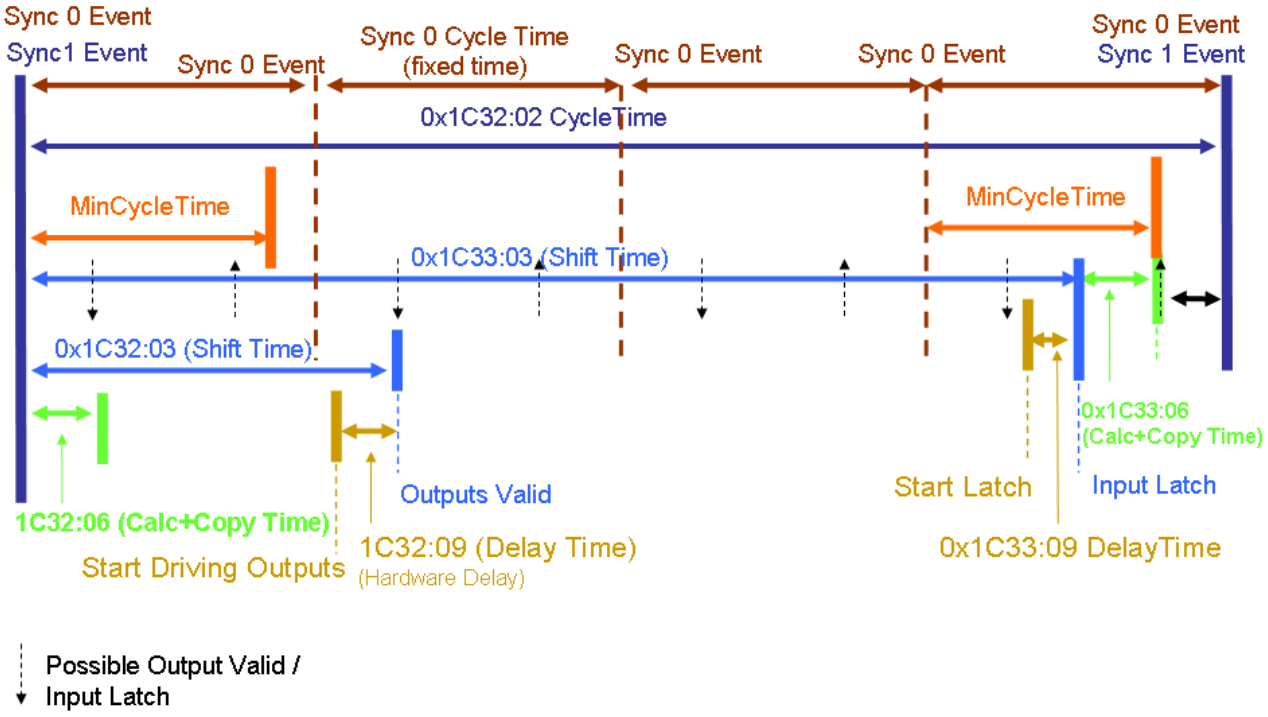


Abb. 291: DC, untergeordnete µC Zyklen, Eingänge/Ausgänge verzögert

Die Tabellen "0x1C32 DC Modus, untergeordnete µC Zyklen, Verschiebung der Eingänge/Ausgänge" und "0x1C33 DC Modus, untergeordnete µC Zyklen, Verschiebung der Eingänge/Ausgänge" erläutern die Anwendung dieser Objekte im DC Modus bei untergeordneten µC Zyklen und Verschiebung des "Outputs Valid"/"Input Latch"-Zeitpunktes.

Subindex	Beschreibung	Flag	Verwendung	Beschreibung/ Default Wert
1	Synchronization Type	r or rw	erforderlich	0x03: DC SYNC1 – synchronisiert mit SYNC1 Event
2	Cycle Time	r or rw	erforderlich	SYNC1 Zykluszeit (Register 0x09A7:0x09A4) Zeit zwischen zwei SYNC1 Events SYNC1 Zykluszeit kann von der Slave Anwendung in diesen Eintrag kopiert werden.
3	Shift Time	rw	optional	
4	Synchronization Types supported	r	erforderlich	Bit 3:2 : DC unterstützt: 10 = Untergeordnete Applikation Bit 5:4: Shift Einstellungen 00 = Keine Ausgangszeitverschiebung 01 = Ausgangszeitverschiebung mit lokalem Timer (Shift Time)
5	Minimum Cycle Time	r	erforderlich	
6	Calc and Copy Time	r	erforderlich	
7	--	--	--	
8	Get Cycle Time	rw	bedingt****	
9	Delay Time	r	erforderlich	
10	SYNC0 Cycle Time	r	erforderlich	
11	Cycle Time Too Small	r	erforderlich	
12	SM-Event missed	r	optional	
13	Shift Time Too Short	r	optional	
14	RxPDO Toggle Failed	r	optional	
31:15	--	--	--	
32	Sync Error	r	bedingt	wird unterstützt, wenn "SM-Event Missed" oder "Shift Time Too Short " Counter verwendet wird

**** wird im Synchron Modus oder in DC Modus mit variabler Zykluszeit verwendet

Tabelle 12: 0x1C32 DC Modus, untergeordnete µC Zyklen, Verschiebung der Eingänge/Ausgänge

Subindex	Beschreibung	Flag	Verwendung	Beschreibung/ Default Wert
1	Synchronization Type	r or rw	erforderlich	0x01: DC SYNC1 – synchronisiert mit SYNC1 Event
2	Cycle Time	r	erforderlich	Gleicher Wert wie 0x1C32:02
3	Shift Time	r	optional	
4	Synchronization Types supported	r	erforderlich	Bit 3:2 : DC unterstützt: 10 = Untergeordnete Applikation Bit 5:4: Shift Einstellungen 00 = Keine Eingangszeitverschiebung 01 = Eingangszeitverschiebung mit lokalem Timer (Shift Time)
5	Minimum Cycle Time	r	erforderlich	Gleicher Wert wie 0x1C32:05
6	Calc and Copy Time	r	erforderlich	
7	--	--	--	
8	Get Cycle Time	rw	bedingt****	Gleicher Wert wie 0x1C32:08
9	Delay Time	r	erforderlich	
10	SYNC0 Cycle Time	r	erforderlich	Gleicher Wert wie 0x1C32:0A
11	Cycle Time Too Small	r	erforderlich	Gleicher Wert wie 0x1C32:0B
12	SM-Event missed	r	optional	Gleicher Wert wie 0x1C32:0C
13	Shift Time Too Short	r	optional	Gleicher Wert wie 0x1C32:0D
14	RxPDO Toggle Failed	r	optional	Gleicher Wert wie 0x1C32:0E
31:15	--	--	--	
32	Sync Error	r	bedingt	Gleicher Wert wie 0x1C32:20

**** wird im Synchron Modus oder in DC Modus mit variabler Zykluszeit verwendet

Tabelle 13: 0x1C33 DC Modus, untergeordnete µC Zyklen, Verschiebung der Eingänge/Ausgänge

7.2.6 EKxxxx - Optionale Distributed Clocks Unterstützung

Grundlagen Distributed Clocks (DC)

Das EtherCAT Distributed-Clocks-System umfasst in den EtherCAT Slaves integrierte lokale Uhren, die über spezielle Datagramme vom EtherCAT Master synchronisiert werden. Nicht alle EtherCAT Slaves unterstützen das Distributed Clocks Verfahren, sondern nur Slaves, deren Funktion dieses erfordert. Im TwinCAT System Manager zeigt eine Slave seine DC-Fähigkeiten, indem er über einen Einstellungsdialog „DC“ verfügt.

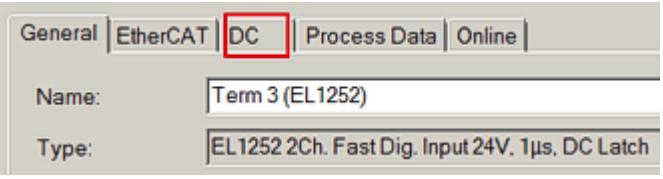


Abb. 292: DC-Reiter zur Anzeige der Distributed Clocks Funktion

Eine dieser lokalen Uhren ist die Referenz-Uhr, nach der alle anderen synchronisiert werden. Siehe dazu entsprechende Erläuterungen in der EtherCAT Grundlagendokumentation. Prinzipbedingt muss das der erste DC-fähige EtherCAT Slave sein. Deshalb wählt TwinCAT standardmäßig den ersten DC-fähigen Teilnehmer als Referenzuhr aus. In den erweiterten Eigenschaften des EtherCAT Masters wird dies dargestellt bzw. kann vom Anwender verändert werden. Die Standard-Einstellung soll nicht verändert werden, außer es wird in entsprechenden Dokumentationen z. B. zur externen Synchronisierung empfohlen.

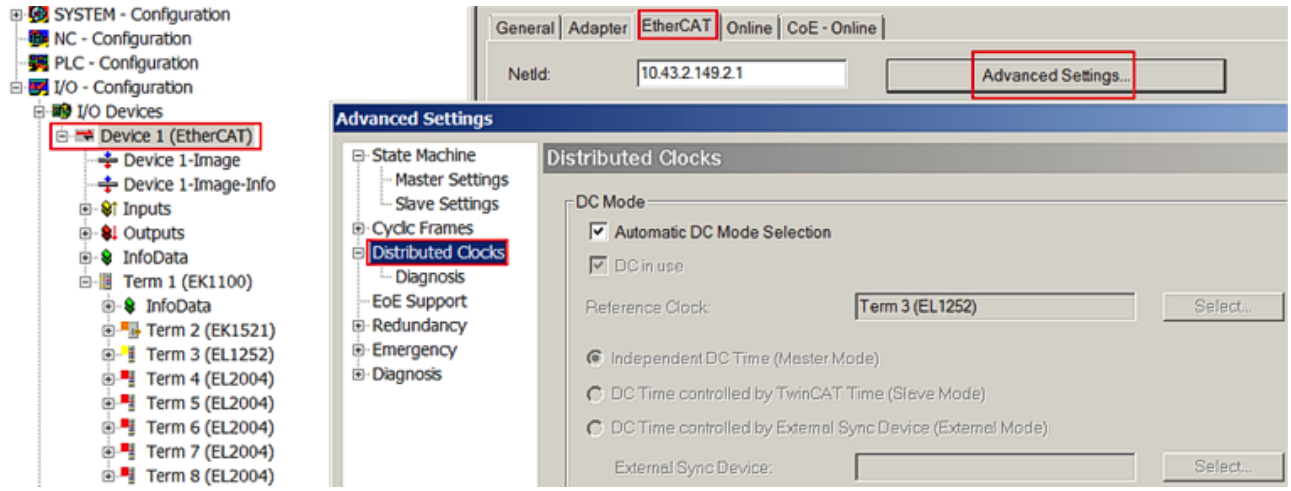


Abb. 293: Erweiterte Einstellung Distributed Clocks im EtherCAT Master

In Abb. *Erweiterte Einstellung Distributed Clocks im EtherCAT Master* ist zu erkennen, wie TwinCAT standardmäßig die EL1252 als Referenzuhr auswählt, da die vorhergehenden Komponenten kein DC unterstützen.

Einstellung EtherCAT Device

System- und Infrastrukturteilnehmer wie die Koppler und Abzweige EK1100, EK1122 etc. benötigen zur Funktion keine Distributed Clocks. Dennoch kann es topologisch sinnvoll sein, z. B. den ersten Koppler im EtherCAT System als Referenzuhr festzulegen. Deshalb sind die Infrastrukturkomponenten ab einem bestimmten Bauzustand in der Lage als Referenzuhr zu arbeiten, wenn in der Konfiguration besondere Einstellungen vorgenommen werden.

Die Komponenten unterstützen lt. der folg. Tabelle die Aktivierung der Distributed Clocks:

Gerät	XML-Revision in der Konfiguration	Seriennummer der Komponente
BK1150	ab BK1150-0000-0016	ab Firmware 01: xxxx01yy
CU1128	ab CU1128-0000-0000	ab Firmware 00: xxxx00yy
EK1100	ab EK1100-0000-0017	ab Firmware 06: xxxx06yy
EK1101	ab EK1101-0000-0017	ab Firmware 01: xxxx01yy
EK1501	ab EK1501-0000-0017	ab Firmware 01: xxxx01yy
EK1501-0010	ab EK1501-0010-0017	ab Firmware 02: xxxx02yy
EK1122	ab EK1122-0000-0017	ab Firmware 01: xxxx02yy
EK1521	ab EK1521-0000-0018	ab Firmware 03: xxxx03yy
EK1541	ab EK1541-0000-0016	ab Firmware 01: xxxx01yy
EK1561	ab EK1561-0000-0016	ab Firmware 01: xxxx01yy
EK1521-0010	ab EK1521-0010-0018	ab Firmware 03: xxxx03yy
EK1814	ab EK1814-0000-0016	ab Firmware 00: xxxx00yy

Tab. 1: DC-Unterstützung ab Rev/FW-Stand

Damit TwinCAT eine solche Komponente als DC-Referenzuhr verwendet, ist ein manueller Eingriff bei der Konfigurationserstellung erforderlich, der hier anhand des EK1100 gezeigt wird.
Die Checkboxen „*Cyclic Mode Enable*“ und „*Use as potential Reference Clock*“ müssen gesetzt werden.

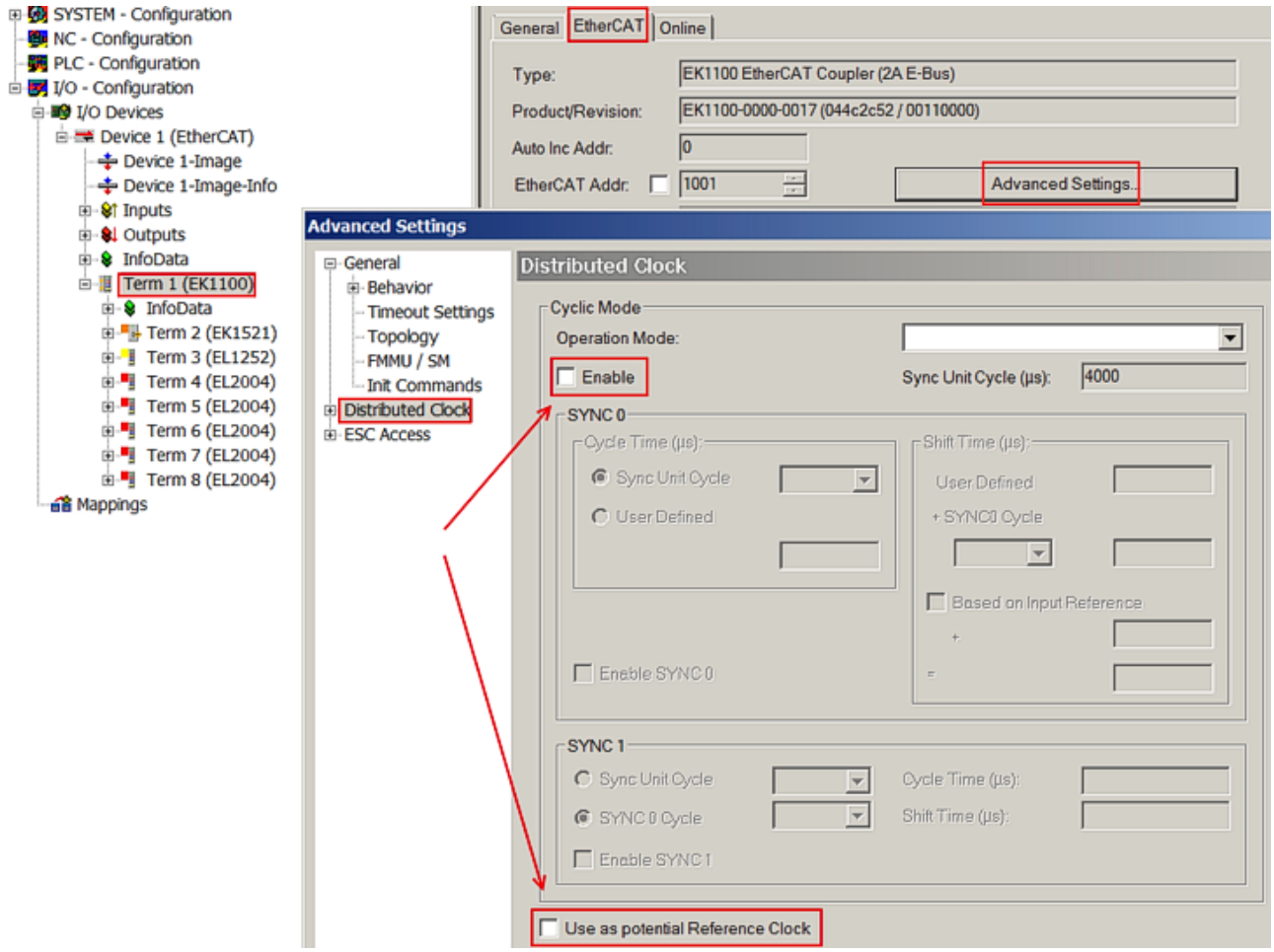


Abb. 294: TwinCAT-Einstellung, um diese Komponente als Referenzuhr zu verwenden

Aktivierung Distributed Clocks Unterstützung

Das hier beschriebene Vorgehen führt nur bei den o.a. Komponenten zum (Synchronisierungs-)Erfolg. Auch bei anderen Komponenten können diese Checkboxen gesetzt werden, die Hardware unterstützt diese Funktion jedoch nicht, wenn nicht entsprechend in der jeweiligen Dokumentation angegeben.
Insbesondere darf nach der Inbetriebnahme die Komponente nicht durch eine frühere Version ausgetauscht werden, die den DC-Support nicht leisten kann.

7.3 Externe Synchronisierung

7.3.1 Grundlagen

Verwendung der Beispielprogramme

Dieses Dokument enthält exemplarische Anwendungen unserer Produkte für bestimmte Einsatzbereiche. Die hier dargestellten Anwendungshinweise beruhen auf den typischen Eigenschaften unserer Produkte und haben ausschließlich Beispielcharakter. Die mit diesem Dokument vermittelten Hinweise beziehen sich ausdrücklich nicht auf spezifische Anwendungsfälle, daher liegt es in der Verantwortung des Kunden zu prüfen und zu entscheiden, ob das Produkt für den Einsatz in einem bestimmten Anwendungsbereich geeignet ist. Wir übernehmen keine Gewährleistung, dass der in diesem Dokument enthaltene Quellcode vollständig und richtig ist. Wir behalten uns jederzeit eine Änderung der Inhalte dieses Dokuments vor und übernehmen keine Haftung für Irrtümer und fehlenden Angaben.

TwinCAT Uhrenhierarchie

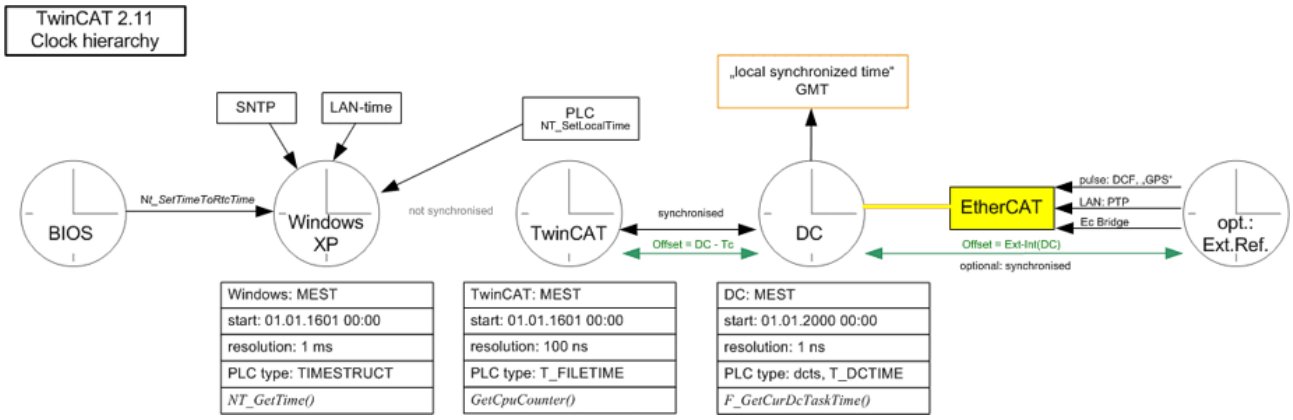


Abb. 295: Uhrenhierarchie TwinCAT 2.11 (ohne Gewähr)

Im Betriebssystem CE ist eine andere Betriebssystemzeit verankert, deshalb wird auch dort der Einsatz der DC-Uhr empfohlen.

Grundlagen

Bei der externen Synchronisierung von TwinCAT bei Benutzung von EtherCAT-Komponenten wird auf der lokalen Steuerung eine Zeit bereitgestellt, die in ihrem Wert der übergeordneten Zeit entspricht. EtherCAT als Feldbus stellt dabei die nötigen Betriebsmittel zur Verfügung, insbesondere den EtherCAT eigenen Synchronisierungsmechanismus der Distributed Clocks. Es werden also

- auf der TwinCAT-Steuerung die EtherCAT-Slaves und der EtherCAT-Master in TwinCAT lokal synchronisiert, s. dazu vorangegangene Seiten.
- danach die Steuerung insgesamt als Slave-Clock mit seiner DC-Clock der übergeordneten Uhr nachgeregelt.

Dabei tritt folgender Ablauf ein:

- die Frequenzsynchronität beider Zeitbasen wird hergestellt
- der Offset zwischen beiden Zeitbasen wird ermittelt und bekanntgegeben

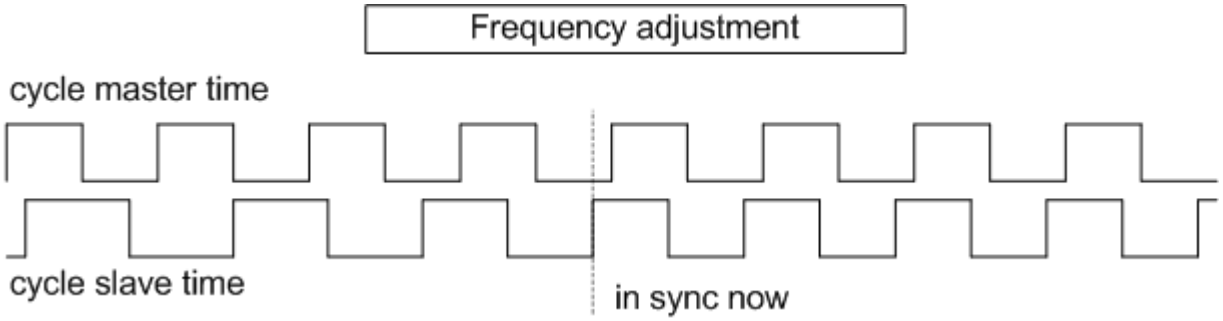


Abb. 296: Frequenzsynchronität

Folgendes ist zu beachten:

- Die Slave-Clock-Steuerung regelt nach dem TwinCAT-Start ihre Distributed-Clocks-Zeit der übergeordneten Zeit in der Frequenz nach:
  - beim EtherCAT-Start wird der erstmalige Offset zwischen beiden Zeiten festgestellt.
  - die nachfolgende Regelung hält diesen Offset konstant und gibt ihn bekannt.
  - die Nachregelung wird kontinuierlich vollzogen.
- Für den Fall, dass die Synchronisierung aussetzt (Verbindung unterbrochen, Neustart eines der Systeme) ist das Verhalten wie folgt:
  - setzt die Regelung in der Slave-Clock-Steuerung wieder ein, wird dort ein erneuter Offset berechnet und bekanntgegeben.

- die Applikation hat diesen Offset deshalb ständig zu beachten.

- Es wird ebenfalls ein neuer Offset berechnet, wenn die Regelungsgrenze von ± 1 Zykluszeit überschritten wird.
- Sowohl die BIOS-Uhr (Motherboard) als auch die Betriebssystemuhr (Windows) wird davon nicht berührt!
- Die TwinCAT-Uhr wird ebenfalls nicht verändert.
- Für Aufgaben in Bezug auf die jeweilige Stationshardware (EtherCAT-Slaves, Klemmen) muss weiterhin die lokale DC-Zeit verwendet werden.
- Wird in der Applikation außerdem die TwinCAT-Zeit verwendet, ist der TcToDc-Offset zwischen TwinCAT- und DC-Uhr zu berücksichtigen.

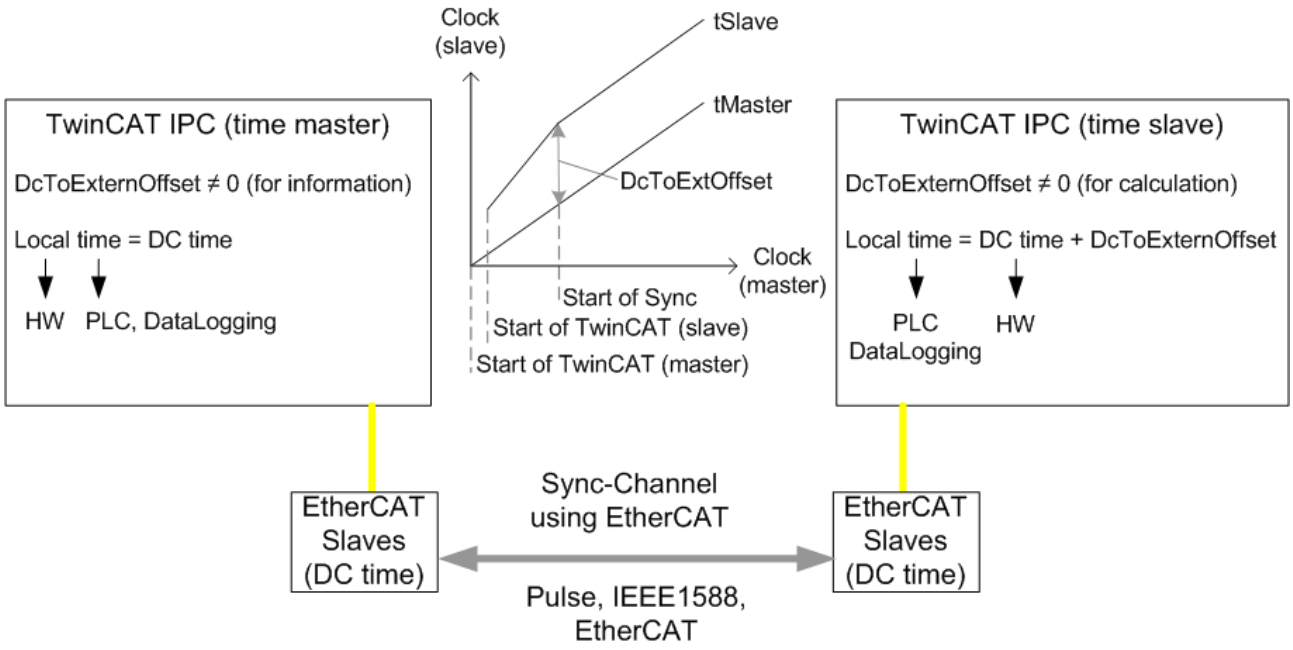


Abb. 297: Synchronisierung von 2 TwinCAT-IPC mithilfe von EtherCAT-Komponenten

Verwendung der synchronisierten Uhrzeit

In der nachgeregelten Station ist die "andere" Zeit aus dem Master-PC bekannt durch:

Synchronisierte DC-Zeit = Lokale DC-Zeit + Offset

Diese synchronisierte Zeit kann nun für Datalogging verwendet werden. Für Aufgaben in Bezug auf die jeweilige Stationshardware (EtherCAT Slaves, Klemmen) muss weiterhin die lokale DC-Zeit verwendet werden.

Kaskadierung von synchronisierten TwinCAT-Systemen

Es wird davon abgeraten mehrere zeitsynchronisierte TwinCAT-Systeme zu kaskadieren. Eine einfache Kaskadierung tritt allerdings bereits dann auf, wenn ein TwinCAT-System durch externe Uhr z. B. gegen GPS geregelt ist und seine lokale Zeit wiederum über eine Bridgeklemme EL6692 an ein unterlagertes EtherCAT-System weitergibt.

Dann ist in den jeweils unterlagerten Systemen der jeweilige DcToExt-Offset der übergeordneten Systeme zu berücksichtigen!

Synchronisierte DC-Zeit = Lokale DC-Zeit + $\text{DcToExtOffset}_{\text{lokal}}$ + Σ $\text{DcToExtOffset}_{\text{übergeodnet}}$

Die übergeordneten jeweiligen DcToExtOffset können durch Netzwerkvariablen, ADS, über die Bridgeklemme EL6692 oder beliebige andere Kanäle transportiert werden. Das unterlagerte System muss diese Offsets mit verrechnen.

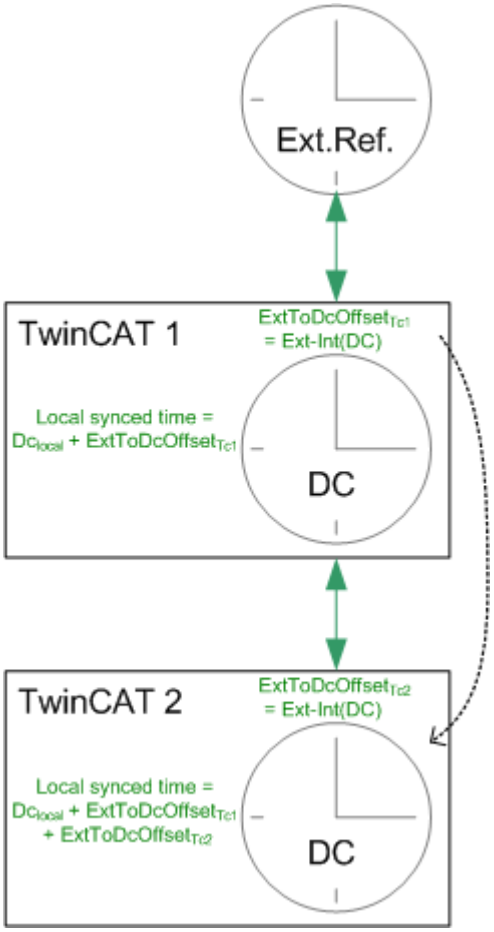


Abb. 298: Kaskade aus geregelten TwinCAT-Systemen

7.3.2 Beispiel: Bridge Klemme EL6692

Verwendung der Beispielprogramme

Dieses Dokument enthält exemplarische Anwendungen unserer Produkte für bestimmte Einsatzbereiche. Die hier dargestellten Anwendungshinweise beruhen auf den typischen Eigenschaften unserer Produkte und haben ausschließlich Beispielcharakter. Die mit diesem Dokument vermittelten Hinweise beziehen sich ausdrücklich nicht auf spezifische Anwendungsfälle, daher liegt es in der Verantwortung des Kunden zu prüfen und zu entscheiden, ob das Produkt für den Einsatz in einem bestimmten Anwendungsbereich geeignet ist. Wir übernehmen keine Gewährleistung, dass der in diesem Dokument enthaltene Quellcode vollständig und richtig ist. Wir behalten uns jederzeit eine Änderung der Inhalte dieses Dokuments vor und übernehmen keine Haftung für Irrtümer und fehlenden Angaben.

Im vorliegenden Beispiel werden zwei Beckhoff IPC mit TwinCAT 2.11, b1539 untereinander synchronisiert. Ein PC ist die Master-Clock, der andere (Slave-Clock) synchronisiert seine "Zeit" auf ihn auf. EtherCAT als Feldbus stellt dabei die nötigen Betriebsmittel zur Verfügung, insbesondere den EtherCAT eigenen Synchronisierungsmechanismus der Distributed Clocks.

Der Vorgang erfolgt nach den Erläuterungen im vorangegangenen Kapitel.

Zu Beachten ist:

- Der Master-PC arbeitet autonom auf Basis seiner DC-Zeit
- Der Slave-PC regelt nach dem TwinCAT-Start seine Distributed Clocks Zeit dem Master-IPC nach:
  - beim EtherCAT-Start wird der erstmalige Offset zwischen beiden Zeiten festgestellt.
  - die nachfolgende Regelung hält diesen Offset konstant und gibt ihn bekannt.
  - die Nachregelung wird kontinuierlich vollzogen.

- Für den Fall, dass die Synchronisierung aussetzt (Verbindung unterbrochen, Neustart eines der Systeme) ist das Verhalten wie folgt
  - Setzt die Regelung im Slave-PC wieder ein, wird dort ein erneuter Offset berechnet und bekanntgegeben.
  - die Applikation hat diesen Offset deshalb ständig zu beachten.
- Für Aufgaben in Bezug auf die jeweilige Stationshardware (EtherCAT Slaves, Klemmen) muss weiterhin die lokale DC-Zeit verwendet werden.
- Die EtherCAT-Zykluszeit muss in beiden Systemen identisch sein.
- Werden in beiden Systemen unterschiedliche Konfigurationen verwendet, d.h. es kommen eine unterschiedliche Anzahl/Typen/Reihenfolge an Slaves zum Einsatz, werden auch die jeweils automatisch berechneten Shiftzeiten differieren.
  Beispiel: in beiden Systemen arbeitet jeweils eine EL2202-0100, die beide gleichzeitig ihren Ausgang schalten sollen. Da unterschiedliche Output-Shiftzeiten berechnet wurden, wird eine konstante Differenz gemessen werden.
  Es ist dann im System mit der kleineren Output-Shifttime die des anderen Systems einzutragen.

HINWEIS

Beeinflussung von Geräten bei Veränderung der Shiftzeiten

Seiteneffekte in Bezug auf die Funktion der anderen Slaves bei Veränderung dieser Shiftzeiten sind zu bedenken!

- Im geregelten System unterliegt der Zeitoffset zwischen den Systemen gewissen Schwankungen.

Beispielprogramm (https://infosys.beckhoff.com/content/1031/ethercatsystem/Resources/zip/2469158155.zip), TwinCAT 2.11

Beachten Sie im Programm die nach Bedarf erfolgte Verwendung von "signed" und "unsigned" 64 Bit-Variablen.

Topologie

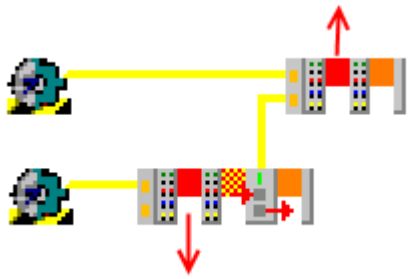


Abb. 299: Topologie des Beispielprogramms

Station Master: EK1100, EL2202, EL6692

Station Slave: EK1100, EL2202

Synchronisiert wird in diesem Beispiel über die EL6692, Richtung *PrimarySide --> SecondarySide (RJ45 Anschluss)*. Auch eine Synchronisierung in der anderen Richtung ist möglich.

EL6692 Dokumentation

Bitte beachten Sie die Angaben in der Dokumentation zur EL6692 zum Systemverhalten dieser Klemme.

Demoprogramm

In diesem Demoprogramm wird auf der Slave-Seite die eigene lokale DC-Zeit aus der ReferenceClock im EtherCAT-Strang mit dem Offset verrechnet, der sich aus der Zeitdifferenz zum externen Synchronisierungsgerät ergibt. Diese Verrechnung macht demzufolge nur auf einer Plattform Sinn, die Synchronisierungslave zu einem Master ist.

Der Synchronisierungsweg kann sein
- ein anderes EtherCAT-System, Mittel: Beckhoff EL6692 Bridge Klemme (dieses Beispiel)
- ein IEEE1588-System, Mittel: Beckhoff EL6688 PTP-Klemme
- ein beliebiger Zeitgeber mit Zeitinfo (GPS, Funkuhr), Mittel: TwinCAT Supplement "Externe Synchronisierung"

Das Prinzip:
TwinCAT bekommt zyklisch (z. B. sekündlich) ein Pärchen (64 Bit, Einheit 1 ns) aus Internem (DC) und externem Zeitstempel. Diese beiden Zeitstempel sind ursprünglich zum selben Zeitpunkt gewonnen. Aus der erstmaligen Differenz wird der Offset zwischen beiden Zeitbasen berechnet und im System Manager | Gerät EtherCAT | InfoData
bekanntgegeben. Weiterhin regelt das Slave-TwinCAT aus dem Verlauf der beiden Zeitstempel zueinander die lokale eigene DC-Zeit nach.

Berechnungen:

- aktuelle Regelabweichung = DcToExtOffset - (Externer Zeitstempel - Interner Zeitstempel)
  Dieser Wert ("signed", 64 Bit) wird mit einer Applikationsspezifischen Schranke verglichen, bei Einhaltung wird die Gültigkeit der Zeit ausgegeben
- lokale synchronisierte Zeit = lokale DC-Zeit + DcToExtOffset
  Diese "nuLocalTime" ("unsigned", 64 Bit) kann nun für Datalogging und Ereignisse mit Zeitbezug zur Master-PC-Clock verwendet werden.

Einrichtung TwinCAT 2.11

Im folgenden Ablauf wird das Gesamtsystem wie folgt eingestellt:

- EL6692 primäre Seite (E-Bus): Sync Master (also Referenzuhr)
- EL6692 sekundäre Seite (RJ45-Buchsen): Sync Slave (also synchronisierte Seite)

Die Synchronisierungsrichtung der Zeit kann auch andersherum eingerichtet werden, den Hinweisen ist dann sinngemäß zu folgen.

Sync Master Seite

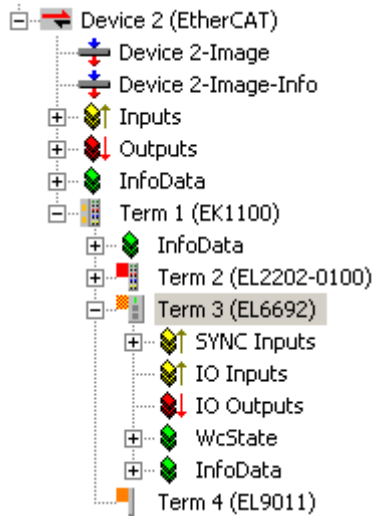


Abb. 300: Teilnehmer auf der Master Seite

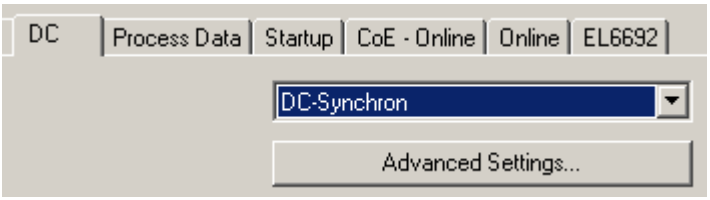


Abb. 301: EL6692 PrimarySide auf DC stellen

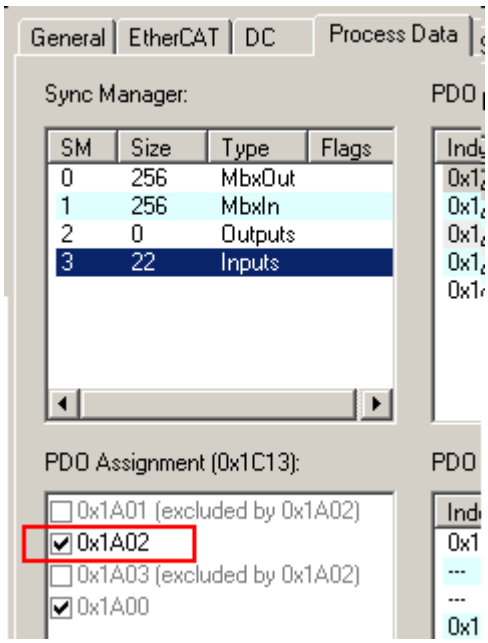


Abb. 302: PDO 0x1A02 aktivieren zur Darstellung der Zeitstempel

Zeitstempel PDO

Das Aktivieren der Zeitstempel-PDO ist für die TwinCAT-Software der jeweiligen Seite der Hinweis, dass diese synchronisiert werden soll, also der SyncSlave ist.
Auf der Sync-Master-Seite (also die Seite die die Referenzuhr darstellt) ist das Aktivieren der Zeitstempel-PDO nicht erforderlich.

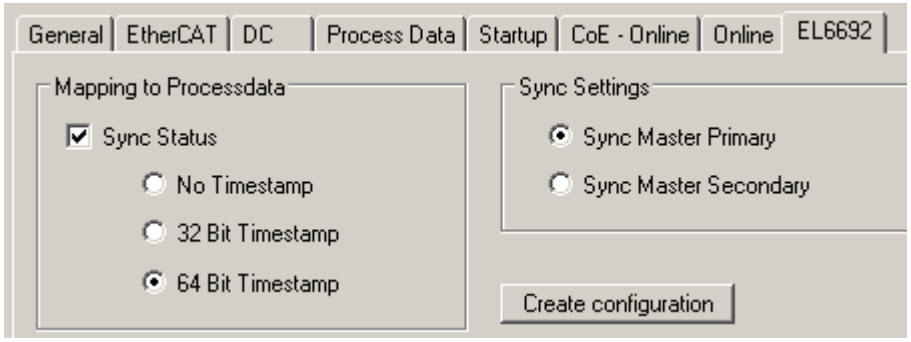


Abb. 303: Synchronisierungsrichtung auf der PrimarySide einstellen, hier SyncSettings: Primary --> Secondary

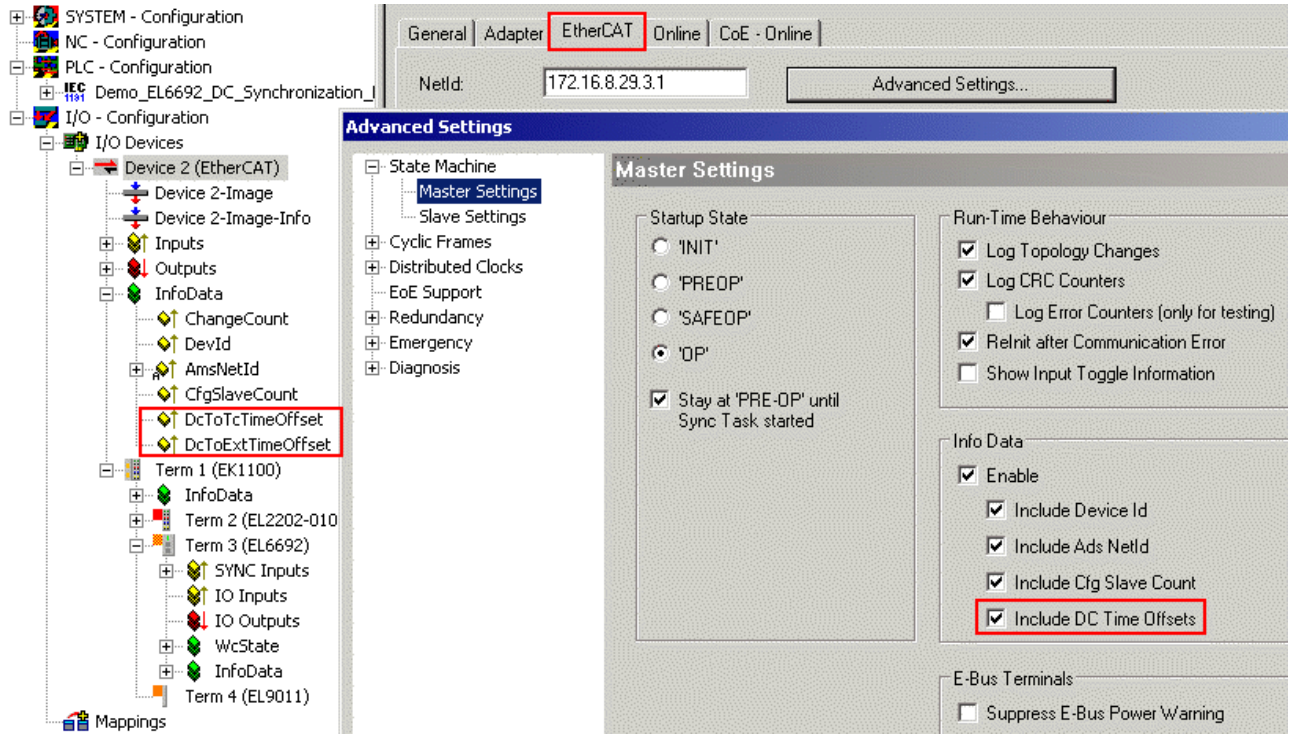


Abb. 304: Anzeige der DC Offsets im EtherCAT Master aktivieren, können auf der Master Seite ausgewertet werden

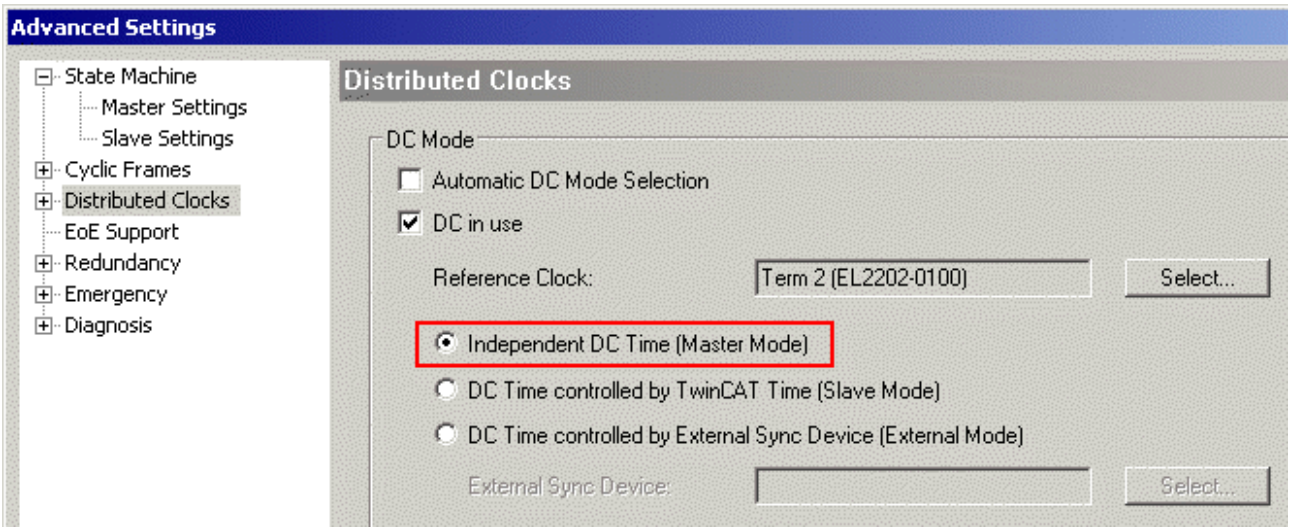


Abb. 305: Master PC arbeitet mit eigener ReferenceClock als Basis

Nun kann TwinCAT auf dieser Seite aktiviert und gestartet werden. Alle Teilnehmer müssen in OP sein, WorkingCounter = 0, keine LostFrames. Die Zeitstempel der EL6692, PrimarySide bleiben auf 0, da die SecondarySide noch nicht konfiguriert wurde.

Sync Slave Seite

Die EL6692, *SecondarySide* wird entsprechend Abb. *EL6692 PrimarySide auf DC stellen* und *PDO 0x1A02 aktivieren zur Darstellung der Zeitstempel* auf DC und 0x1A02 umgestellt.

Nach einem Reload der Konfiguration (oder Neustart im *ConfigMode*, FreeRun) kann durch *GetConfiguration* auf der *SecondarySide* die Synchronisierungsrichtung ausgelesen werden, s. Abb. *SecondarySide der EL6692*.

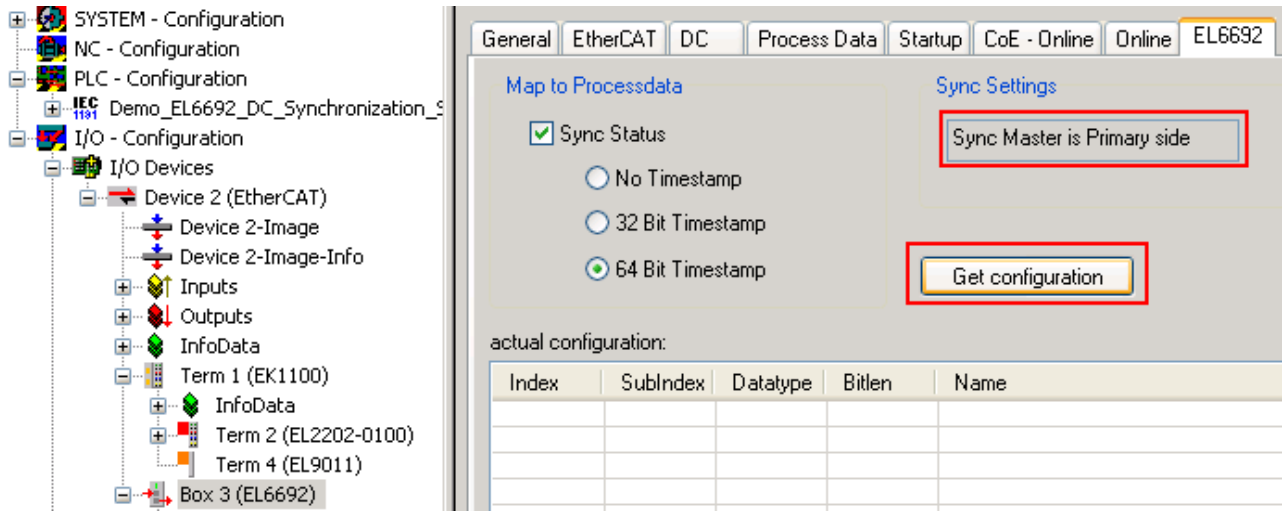


Abb. 306: SecondarySide der EL6692

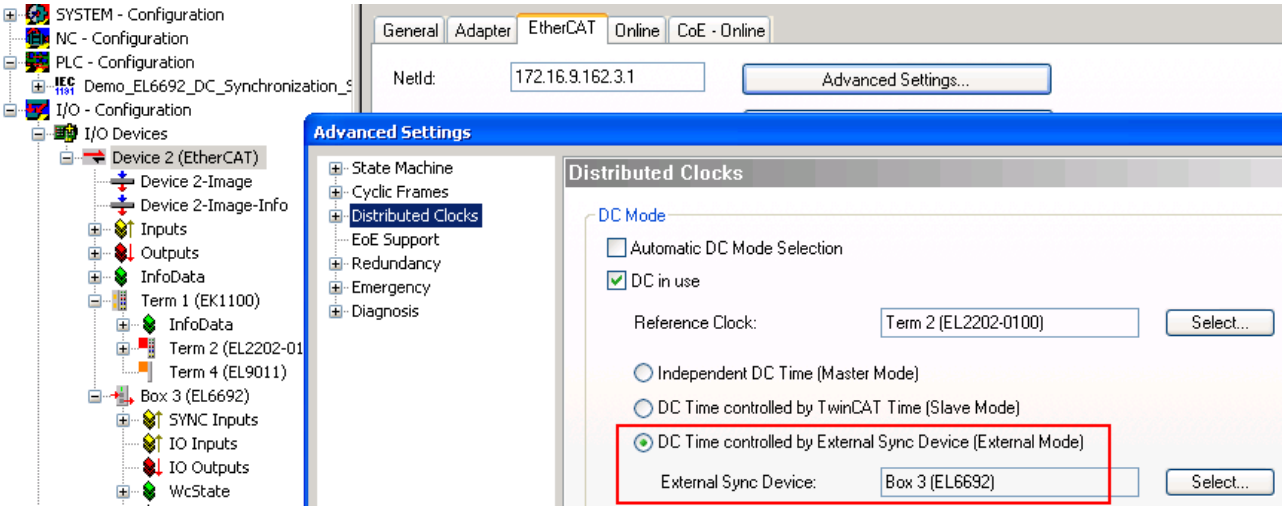


Abb. 307: Einstellung EtherCAT Master, Slave Seite

Nach dem Neustart ist dem EtherCAT Master die DC-Funktion der EL6692 bekannt, deshalb bietet er nun im DC-Dialog diese EL6692 als *ExternalSyncDevice* an.

Für die Auswertungen ist die Verknüpfung der folgenden Variablen nötig, s. Abb. *Slave Seite*.

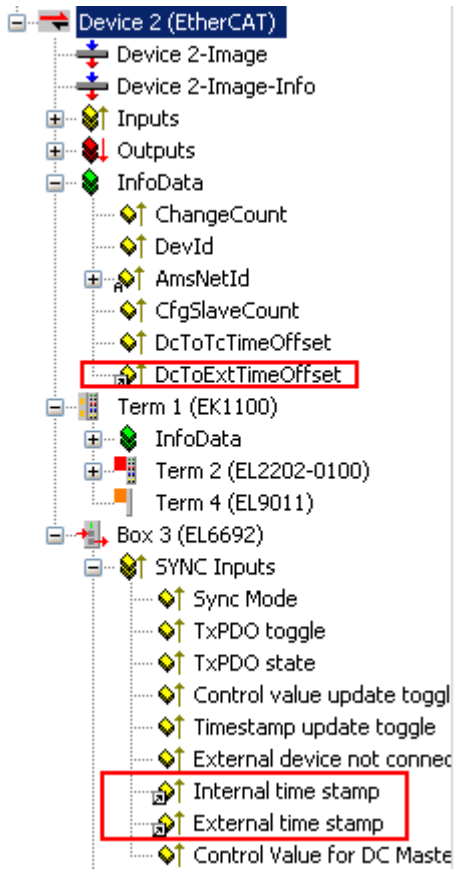


Abb. 308: Slave Seite

HINWEIS

Demoprogramm

Die nachfolgenden Screenshots und Angaben beziehen sich ausschließlich auf das hier besprochene PLC-Demoprogramm und den darin beispielhaft dargestellten Code, nicht auf Analysefunktionen des System Manager.
Beachten Sie dazu auch den Hinweis [▶ 285].

Auf der Slave-Seite kann mit der enthaltenen Visualisierung der Synchronisierungsstart beobachtet werden.

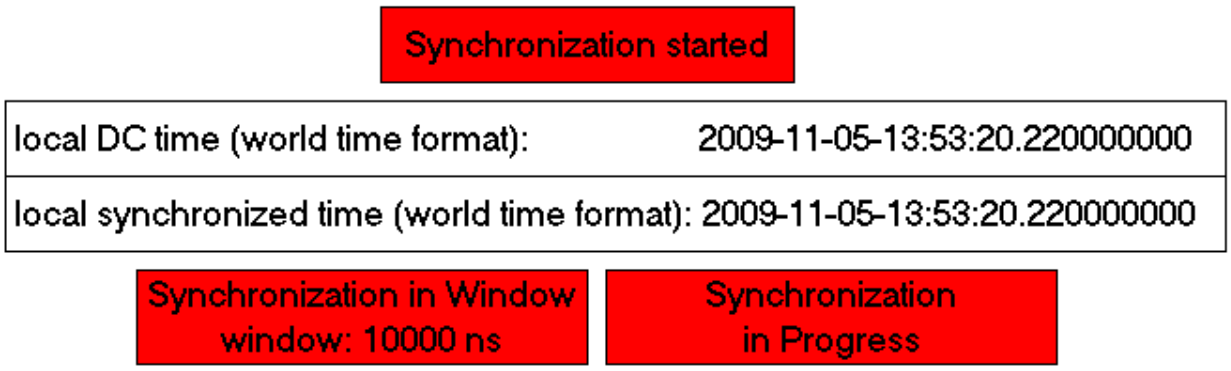


Abb. 309: Start Slave Seite

Sofort nach dem Start steht auf der Slave-Seite nur die lokale DC-Zeit zur Verfügung.

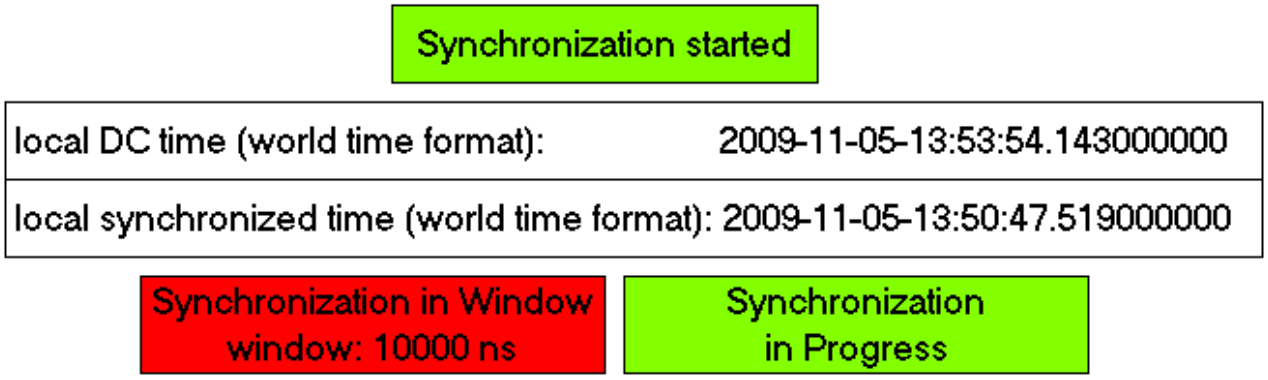


Abb. 310: Zeitstempel bekannt

Nach dem Erhalt der ersten Zeitstempel über die EL6692 ist der Offset bekannt, er beträgt hier rd. 3 Minuten Unterschied in der Zeit der verwendeten IPC. Die Einsynchronisierung hat begonnen, in diesem Beispiel ist ein Fenster von ± 10 µs zu erreichen.

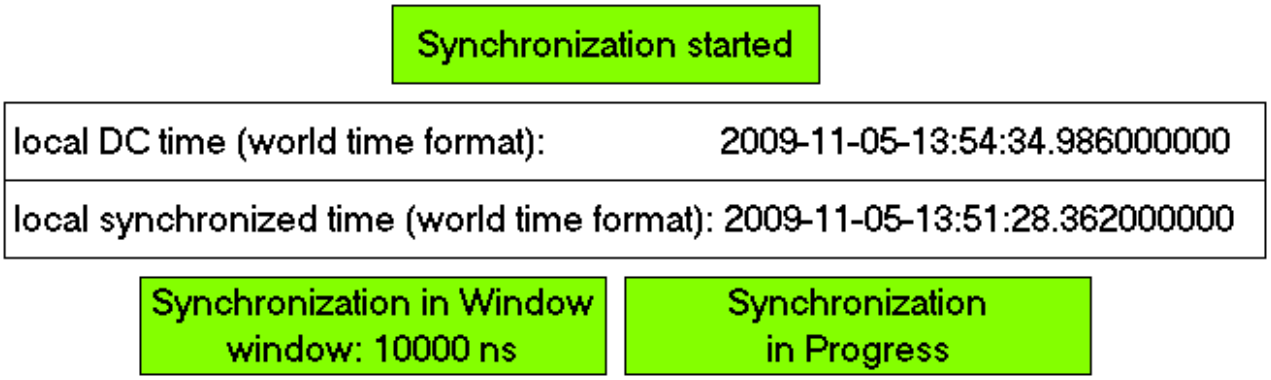


Abb. 311: Synchronisierung erfolgreich

8 Konfiguration der Klemmen

8.1 Allgemeine Konfiguration

8.1.1 EtherCAT Teilnehmerkonfiguration

Klicken Sie im linken Fenster des TwinCAT 2 System Managers bzw. bei der TwinCAT 3 Entwicklungsumgebung im Projektmappen-Explorer auf das Element der Klemme im Baum, die Sie konfigurieren möchten (im Beispiel: Klemme 3: EL3751).

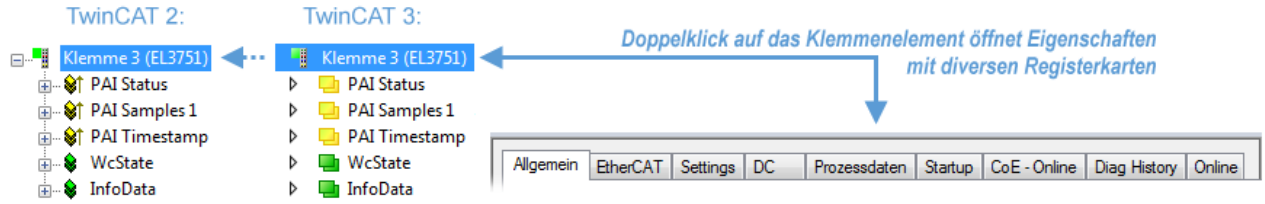


Abb. 312: „Baumzweig“ Element als Klemme EL3751

Im rechten Fenster des System Managers (TwinCAT 2) bzw. der Entwicklungsumgebung (TwinCAT 3) stehen Ihnen nun verschiedene Karteireiter zur Konfiguration der Klemme zur Verfügung. Dabei bestimmt das Maß der Komplexität eines Teilnehmers welche Karteireiter zur Verfügung stehen. So bietet, wie im obigen Beispiel zu sehen, die Klemme EL3751 viele Einstellmöglichkeiten und stellt eine entsprechende Anzahl von Karteireitern zur Verfügung. Im Gegensatz dazu stehen z. B. bei der Klemme EL1004 lediglich die Karteireiter „Allgemein“, „EtherCAT“, „Prozessdaten“ und „Online“ zur Auswahl. Einige Klemmen, wie etwa die EL6695 bieten spezielle Funktionen über einen Karteireiter mit der eigenen Klemmenbezeichnung an, also „EL6695“ in diesem Fall. Ebenfalls wird ein spezieller Karteireiter „Settings“ von Klemmen mit umfangreichen Einstellmöglichkeiten angeboten (z. B. EL3751).

Karteireiter „Allgemein“

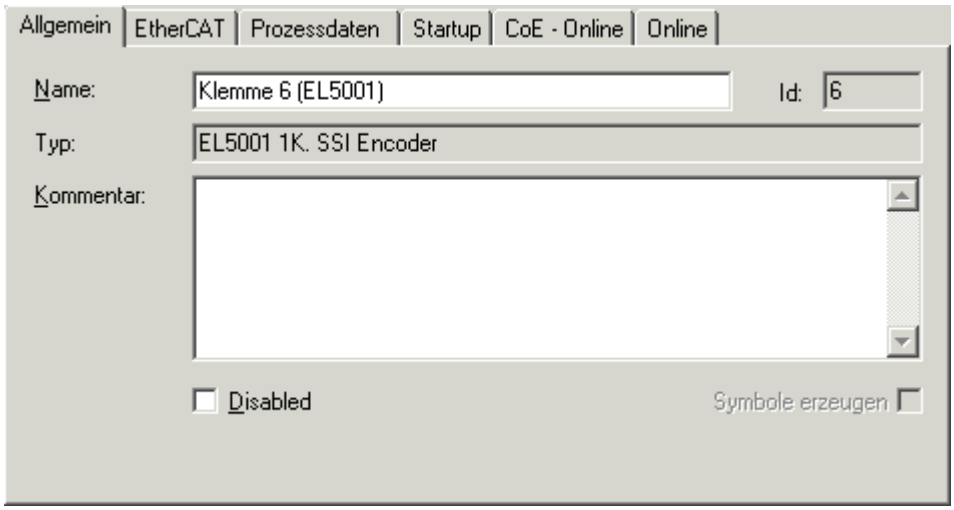


Abb. 313: Karteireiter „Allgemein“

Name	Name des EtherCAT-Geräts
Id	Laufende Nr. des EtherCAT-Geräts
Typ	Typ des EtherCAT-Geräts
Kommentar	Hier können Sie einen Kommentar (z. B. zum Anlagenteil) hinzufügen.
Disabled	Hier können Sie das EtherCAT-Gerät deaktivieren.
Symbole erzeugen	Nur wenn dieses Kontrollkästchen aktiviert ist, können Sie per ADS auf diesen EtherCAT-Slave zugreifen.

Karteireiter „EtherCAT“

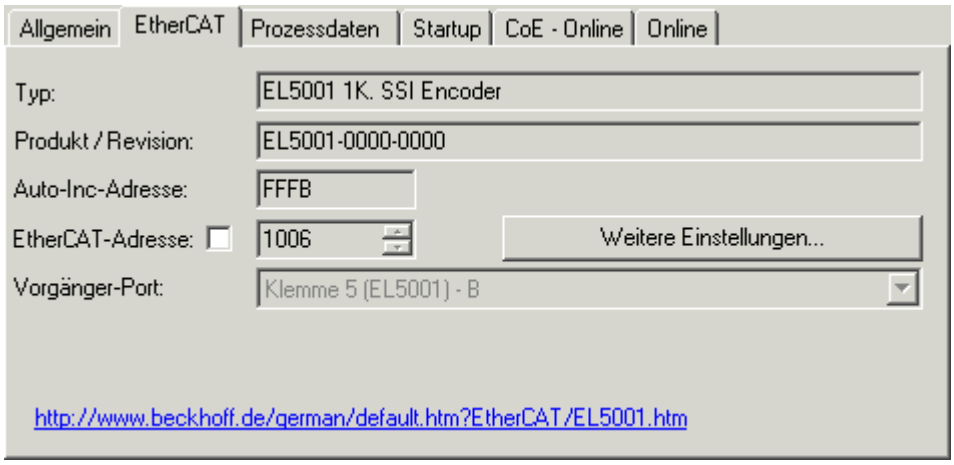


Abb. 314: Karteireiter „EtherCAT“

Typ	Typ des EtherCAT-Geräts
Product/Revision	Produkt- und Revisions-Nummer des EtherCAT-Geräts
Auto Inc Adr.	Auto-Inkrement-Adresse des EtherCAT-Geräts. Die Auto-Inkrement-Adresse kann benutzt werden, um jedes EtherCAT-Gerät anhand seiner physikalischen Position im Kommunikationsring zu adressieren. Die Auto-Inkrement-Adressierung wird während der Start-Up-Phase benutzt, wenn der EtherCAT-master die Adressen an die EtherCAT-Geräte vergibt. Bei der Auto-Inkrement-Adressierung hat der erste EtherCAT-Slave im Ring die Adresse 0000_{hex} und für jeden weiteren Folgenden wird die Adresse um 1 verringert ($FFFF_{hex}$, $FFFE_{hex}$ usw.).
EtherCAT Adr.	Feste Adresse eines EtherCAT-Slaves. Diese Adresse wird vom EtherCAT-Master während der Start-Up-Phase vergeben. Um den Default-Wert zu ändern, müssen Sie zuvor das Kontrollkästchen links von dem Eingabefeld markieren.
Vorgänger Port	Name und Port des EtherCAT-Geräts, an den dieses Gerät angeschlossen ist. Falls es möglich ist, dieses Gerät mit einem anderen zu verbinden, ohne die Reihenfolge der EtherCAT-Geräte im Kommunikationsring zu ändern, dann ist dieses Kombinationsfeld aktiviert und Sie können das EtherCAT-Gerät auswählen, mit dem dieses Gerät verbunden werden soll.
Weitere Einstellungen	Diese Schaltfläche öffnet die Dialoge für die erweiterten Einstellungen.

Der Link am unteren Rand des Karteireiters führt Sie im Internet auf die Produktseite dieses EtherCAT-Geräts.

Karteireiter „Prozessdaten“

Zeigt die (Allgemeine Slave PDO-) Konfiguration der Prozessdaten an. Die Eingangs- und Ausgangsdaten des EtherCAT-Slaves werden als CANopen Prozess-Daten-Objekte (**P**rocess **D**ata **O**bjects, PDO) dargestellt. Falls der EtherCAT-Slave es unterstützt, ermöglicht dieser Dialog dem Anwender ein PDO über PDO-Zuordnung auszuwählen und den Inhalt des individuellen PDOs zu variieren.

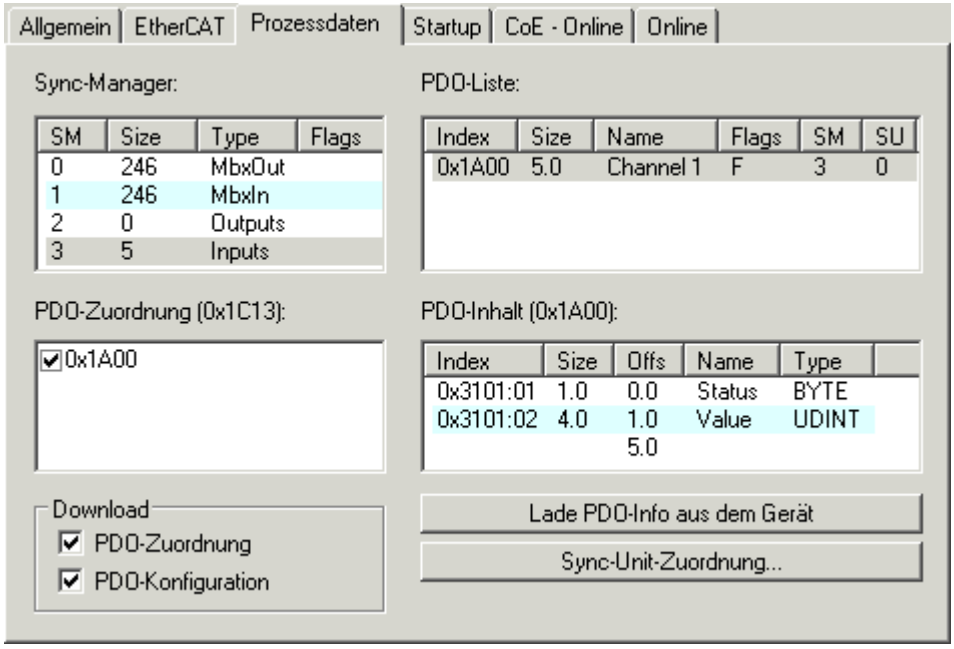


Abb. 315: Karteireiter „Prozessdaten“

Die von einem EtherCAT Slave zyklisch übertragenen Prozessdaten (PDOs) sind die Nutzdaten, die in der Applikation zyklusaktuell erwartet werden oder die an den Slave gesendet werden. Dazu parametriert der EtherCAT Master (Beckhoff TwinCAT) jeden EtherCAT Slave während der Hochlaufphase, um festzulegen, welche Prozessdaten (Größe in Bit/Bytes, Quellort, Übertragungsart) er von oder zu diesem Slave übermitteln möchte. Eine falsche Konfiguration kann einen erfolgreichen Start des Slaves verhindern.

Für Beckhoff EtherCAT Slaves EL, ES, EM, EJ und EP gilt im Allgemeinen:

- Die vom Gerät unterstützten Prozessdaten Input/Output sind in der ESI/XML-Beschreibung herstellerseitig definiert. Der TwinCAT EtherCAT Master verwendet die ESI-Beschreibung zur richtigen Konfiguration des Slaves.
- Wenn vorgesehen, können die Prozessdaten im System Manager verändert werden. Siehe dazu die Gerätedokumentation.
  Solche Veränderungen können sein: Ausblenden eines Kanals, Anzeige von zusätzlichen zyklischen Informationen, Anzeige in 16 Bit statt in 8 Bit Datenumfang usw.
- Die Prozessdateninformationen liegen bei so genannten „intelligenten“ EtherCAT-Geräten ebenfalls im CoE-Verzeichnis vor. Beliebige Veränderungen in diesem CoE-Verzeichnis, die zu abweichenden PDO-Einstellungen führen, verhindern jedoch das erfolgreiche Hochlaufen des Slaves. Es wird davon abgeraten, andere als die vorgesehene Prozessdaten zu konfigurieren, denn die Geräte-Firmware (wenn vorhanden) ist auf diese PDO-Kombinationen abgestimmt.

Ist laut Gerätedokumentation eine Veränderung der Prozessdaten zulässig, kann dies wie folgt vorgenommen werden, s. Abb. *Konfigurieren der Prozessdaten*.

- A: Wählen Sie das zu konfigurierende Gerät
- B: Wählen Sie im Reiter „Process Data“ den Input- oder Output-Syncmanager (C)
- D: die PDOs können an- bzw. abgewählt werden
- H: die neuen Prozessdaten sind als link-fähige Variablen im System Manager sichtbar
  Nach einem Aktivieren der Konfiguration und TwinCAT-Neustart (bzw. Neustart des EtherCAT Masters) sind die neuen Prozessdaten aktiv.
- E: wenn ein Slave dies unterstützt, können auch Input- und Output-PDO gleichzeitig durch Anwahl eines so genannten PDO-Satzes („Predefined PDO-settings“) verändert werden.

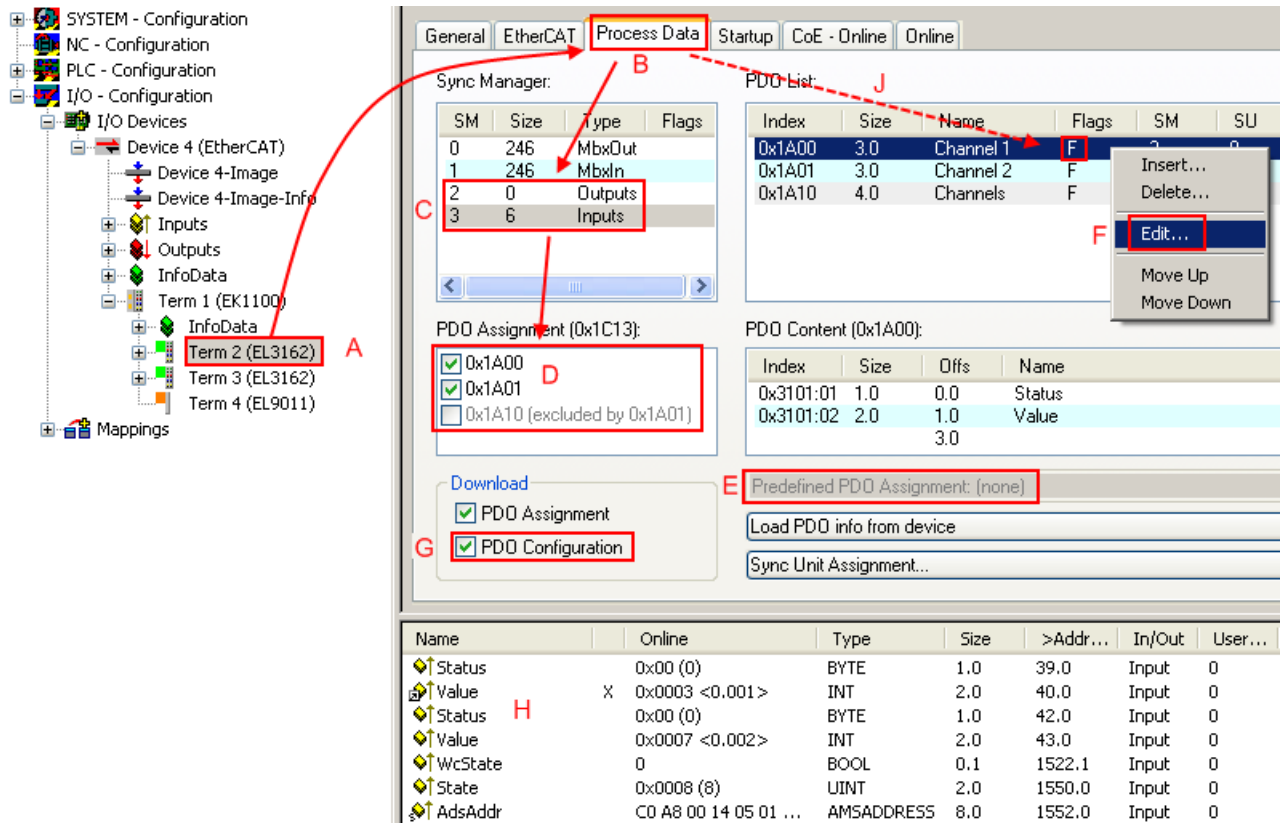


Abb. 316: Konfigurieren der Prozessdaten

Manuelle Veränderung der Prozessdaten

In der PDO-Übersicht kann laut ESI-Beschreibung ein PDO als „fixed“ mit dem Flag „F“ gekennzeichnet sein (Abb. *Konfigurieren der Prozessdaten*, J). Solche PDOs können prinzipiell nicht in ihrer Zusammenstellung verändert werden, auch wenn TwinCAT den entsprechenden Dialog anbietet („Edit“). Insbesondere können keine beliebigen CoE-Inhalte als zyklische Prozessdaten eingeblendet werden. Dies gilt im Allgemeinen auch für den Fall, dass ein Gerät den Download der PDO Konfiguration „G“ unterstützt. Bei falscher Konfiguration verweigert der EtherCAT Slave üblicherweise den Start und Wechsel in den OP-State. Eine Logger-Meldung wegen „invalid SM cfg“ wird im System Manager ausgegeben: Diese Fehlermeldung „invalid SM IN cfg“ oder „invalid SM OUT cfg“ bietet gleich einen Hinweis auf die Ursache des fehlgeschlagenen Starts.

Eine detaillierte Beschreibung [▶ 301] befindet sich am Ende dieses Kapitels.

Karteireiter „Startup“

Der Karteireiter *Startup* wird angezeigt, wenn der EtherCAT-Slave eine Mailbox hat und das Protokoll *CANopen over EtherCAT* (CoE) oder das Protokoll *Servo drive over EtherCAT* unterstützt. Mit Hilfe dieses Karteireiters können Sie betrachten, welche Download-Requests während des Startups zur Mailbox gesendet werden. Es ist auch möglich neue Mailbox-Requests zur Listenanzeige hinzuzufügen. Die Download-Requests werden in derselben Reihenfolge zum Slave gesendet, wie sie in der Liste angezeigt werden.

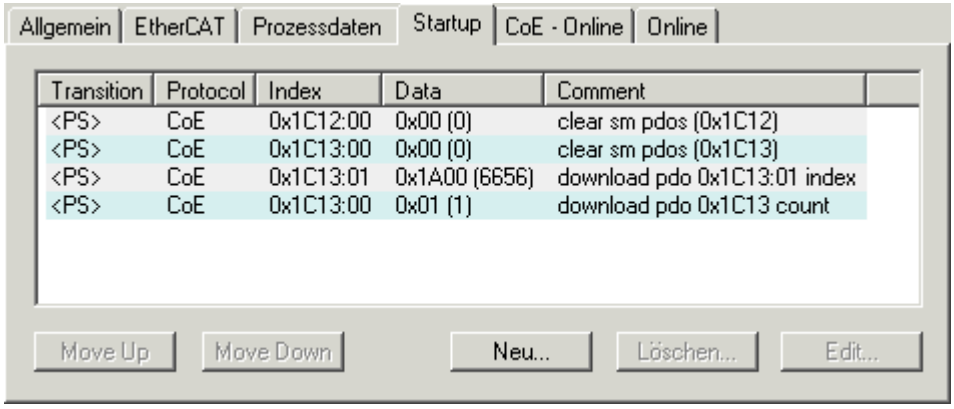


Abb. 317: Karteireiter „Startup“

Spalte	Beschreibung
Transition	Übergang, in den der Request gesendet wird. Dies kann entweder • der Übergang von Pre-Operational to Safe-Operational (PS) oder • der Übergang von Safe-Operational to Operational (SO) sein. Wenn der Übergang in „<>“ eingeschlossen ist (z. B. <PS>), dann ist der Mailbox Request fest und kann vom Anwender nicht geändert oder gelöscht werden.
Protokoll	Art des Mailbox-Protokolls
Index	Index des Objekts
Data	Datum, das zu diesem Objekt heruntergeladen werden soll.
Kommentar	Beschreibung des zu der Mailbox zu sendenden Requests

Move Up	Diese Schaltfläche bewegt den markierten Request in der Liste um eine Position nach oben.
Move Down	Diese Schaltfläche bewegt den markierten Request in der Liste um eine Position nach unten.
New	Diese Schaltfläche fügt einen neuen Mailbox-Download-Request, der währen des Startups gesendet werden soll hinzu.
Delete	Diese Schaltfläche löscht den markierten Eintrag.
Edit	Diese Schaltfläche editiert einen existierenden Request.

Karteireiter „CoE - Online“

Wenn der EtherCAT-Slave das Protokoll *CANopen over EtherCAT* (CoE) unterstützt, wird der zusätzliche Karteireiter *CoE - Online* angezeigt. Dieser Dialog listet den Inhalt des Objektverzeichnisses des Slaves auf (SDO-Upload) und erlaubt dem Anwender den Inhalt eines Objekts dieses Verzeichnisses zu ändern. Details zu den Objekten der einzelnen EtherCAT-Geräte finden Sie in den gerätespezifischen Objektbeschreibungen.

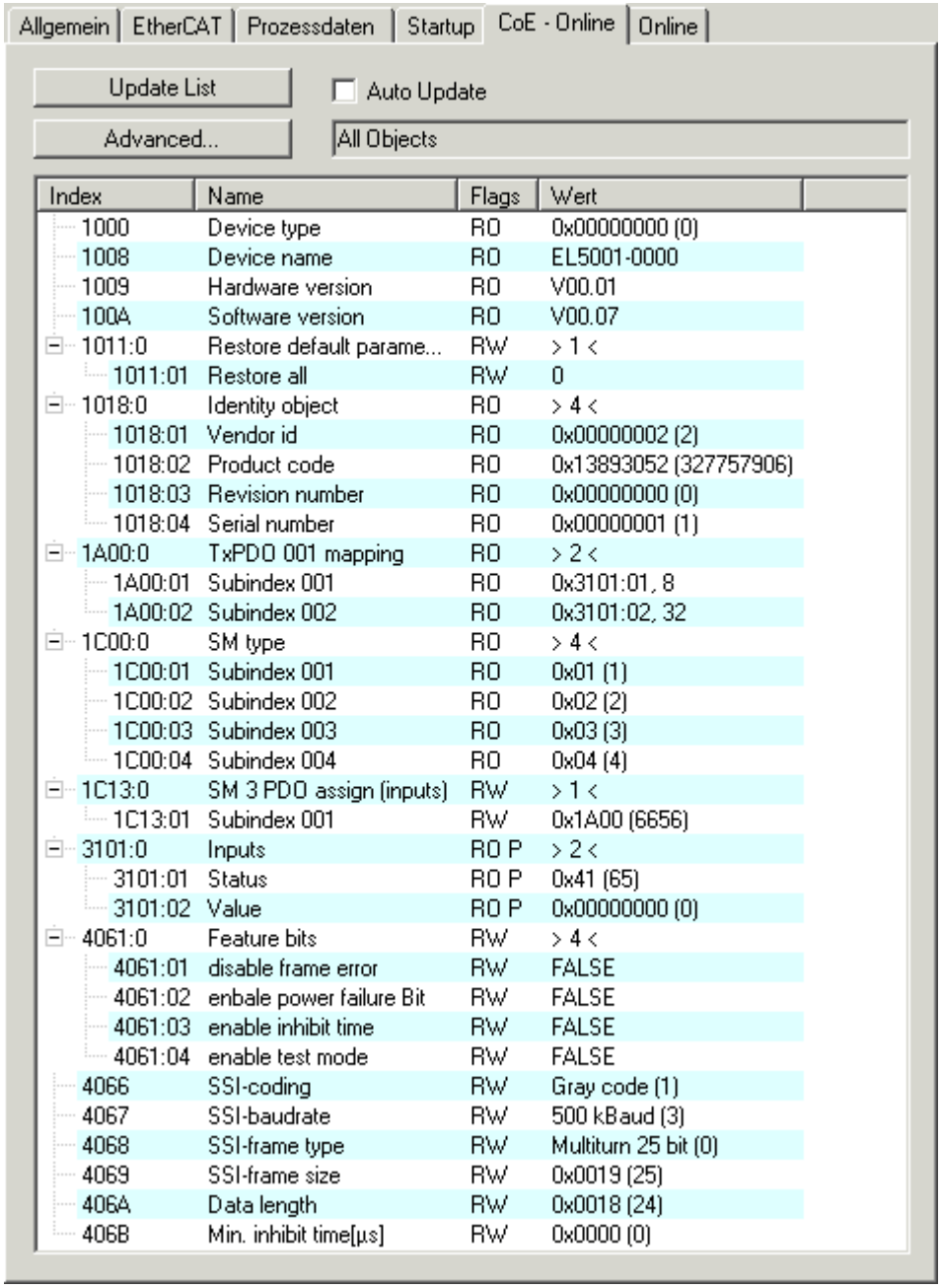


Abb. 318: Karteireiter „CoE - Online“

Darstellung der Objekt-Liste

Spalte	Beschreibung	
Index	Index und Subindex des Objekts	
Name	Name des Objekts	
Flags	RW	Das Objekt kann ausgelesen und Daten können in das Objekt geschrieben werden (Read/Write)
	RO	Das Objekt kann ausgelesen werden, es ist aber nicht möglich Daten in das Objekt zu schreiben (Read only)
	P	Ein zusätzliches P kennzeichnet das Objekt als Prozessdatenobjekt.
Wert	Wert des Objekts	

Update List	Die Schaltfläche *Update List* aktualisiert alle Objekte in der Listenanzeige
Auto Update	Wenn dieses Kontrollkästchen angewählt ist, wird der Inhalt der Objekte automatisch aktualisiert.
Advanced	Die Schaltfläche *Advanced* öffnet den Dialog *Advanced Settings*. Hier können Sie festlegen, welche Objekte in der Liste angezeigt werden.

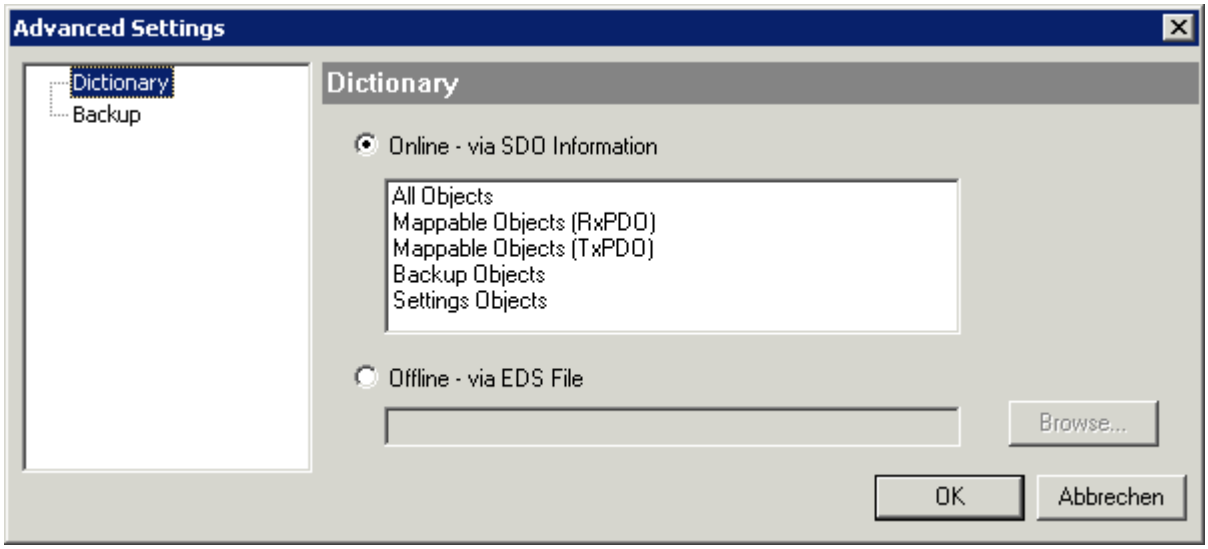


Abb. 319: Dialog „Advanced settings“

Online - über SDO-Information	Wenn dieses Optionsfeld angewählt ist, wird die Liste der im Objektverzeichnis des Slaves enthaltenen Objekte über SDO-Information aus dem Slave hochgeladen. In der untenstehenden Liste können Sie festlegen welche Objekt-Typen hochgeladen werden sollen.
Offline - über EDS-Datei	Wenn dieses Optionsfeld angewählt ist, wird die Liste der im Objektverzeichnis enthaltenen Objekte aus einer EDS-Datei gelesen, die der Anwender bereitstellt.

Karteireiter „Online“

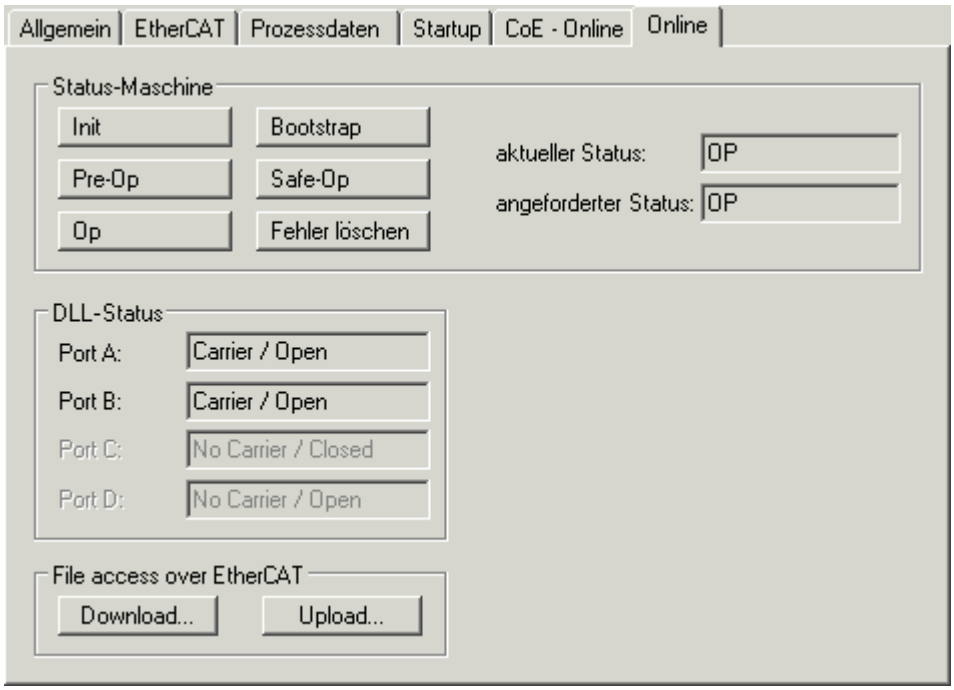


Abb. 320: Karteireiter „Online“

Status Maschine

Init	Diese Schaltfläche versucht das EtherCAT-Gerät auf den Status *Init* zu setzen.
Pre-Op	Diese Schaltfläche versucht das EtherCAT-Gerät auf den Status *Pre-Operational* zu setzen.
Op	Diese Schaltfläche versucht das EtherCAT-Gerät auf den Status *Operational* zu setzen.
Bootstrap	Diese Schaltfläche versucht das EtherCAT-Gerät auf den Status *Bootstrap* zu setzen.
Safe-Op	Diese Schaltfläche versucht das EtherCAT-Gerät auf den Status *Safe-Operational* zu setzen.
Fehler löschen	Diese Schaltfläche versucht die Fehleranzeige zu löschen. Wenn ein EtherCAT-Slave beim Statuswechsel versagt, setzt er eine Fehler-Flag. Beispiel: ein EtherCAT-Slave ist im Zustand PREOP (Pre-Operational). Nun fordert der Master den Zustand SAFEOP (Safe-Operational) an. Wenn der Slave nun beim Zustandswechsel versagt, setzt er das Fehler-Flag. Der aktuelle Zustand wird nun als ERR PREOP angezeigt. Nach Drücken der Schaltfläche *Fehler löschen* ist das Fehler-Flag gelöscht und der aktuelle Zustand wird wieder als PREOP angezeigt.
Aktueller Status	Zeigt den aktuellen Status des EtherCAT-Geräts an.
Angeforderter Status	Zeigt den für das EtherCAT-Gerät angeforderten Status an.

DLL-Status

Zeigt den DLL-Status (Data-Link-Layer-Status) der einzelnen Ports des EtherCAT-Slaves an. Der DLL-Status kann vier verschiedene Zustände annehmen:

Status	Beschreibung
No Carrier / Open	Kein Carrier-Signal am Port vorhanden, der Port ist aber offen.
No Carrier / Closed	Kein Carrier-Signal am Port vorhanden und der Port ist geschlossen.
Carrier / Open	Carrier-Signal ist am Port vorhanden und der Port ist offen.
Carrier / Closed	Carrier-Signal ist am Port vorhanden, der Port ist aber geschlossen.

File Access over EtherCAT

Download	Mit dieser Schaltfläche können Sie eine Datei zum EtherCAT-Gerät schreiben.
Upload	Mit dieser Schaltfläche können Sie eine Datei vom EtherCAT-Gerät lesen.

Karteireiter „DC“ (Distributed Clocks)

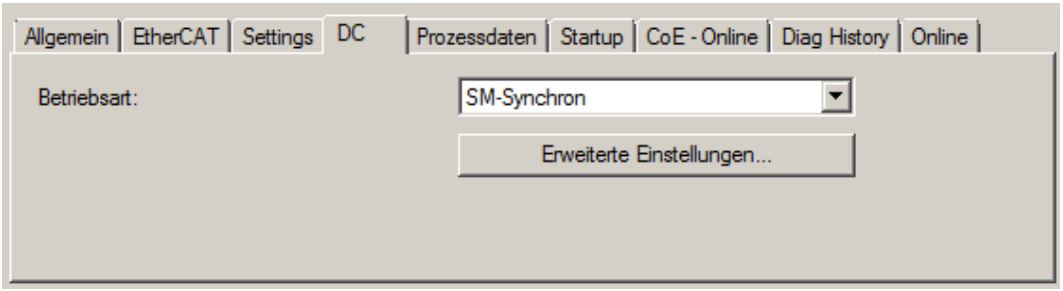


Abb. 321: Karteireiter „DC“ (Distributed Clocks)

Betriebsart	Auswahlmöglichkeiten (optional): • FreeRun • SM-Synchron • DC-Synchron (Input based) • DC-Synchron
Erweiterte Einstellungen…	Erweiterte Einstellungen für die Nachregelung der echtzeitbestimmende TwinCAT-Uhr

Detaillierte Informationen zu Distributed Clocks sind unter http://infosys.beckhoff.de angegeben:

Feldbuskomponenten → EtherCAT-Klemmen → EtherCAT System Dokumentation → Distributed Clocks

8.1.1.1 Detaillierte Beschreibung Karteireiter „Prozessdaten“

Sync-Manager

Listet die Konfiguration der Sync-Manager (SM) auf.
Wenn das EtherCAT-Gerät eine Mailbox hat, wird der SM0 für den Mailbox-Output (MbxOut) und der SM1 für den Mailbox-Intput (MbxIn) benutzt.
Der SM2 wird für die Ausgangsprozessdaten (Outputs) und der SM3 (Inputs) für die Eingangsprozessdaten benutzt.

Wenn ein Eintrag ausgewählt ist, wird die korrespondierende PDO-Zuordnung in der darunter stehenden Liste *PDO-Zuordnung* angezeigt.

PDO-Zuordnung

PDO-Zuordnung des ausgewählten Sync-Managers. Hier werden alle für diesen Sync-Manager-Typ definierten PDOs aufgelistet:

- Wenn in der Sync-Manager-Liste der Ausgangs-Sync-Manager (Outputs) ausgewählt ist, werden alle RxPDOs angezeigt.
- Wenn in der Sync-Manager-Liste der Eingangs-Sync-Manager (Inputs) ausgewählt ist, werden alle TxPDOs angezeigt.

Die markierten Einträge sind die PDOs, die an der Prozessdatenübertragung teilnehmen. Diese PDOs werden in der Baumdarstellung dass System-Managers als Variablen des EtherCAT-Geräts angezeigt. Der Name der Variable ist identisch mit dem Parameter *Name* des PDO, wie er in der PDO-Liste angezeigt wird. Falls ein Eintrag in der PDO-Zuordnungsliste deaktiviert ist (nicht markiert und ausgegraut), zeigt dies an, dass dieser Eintrag von der PDO-Zuordnung ausgenommen ist. Um ein ausgegrautes PDO auswählen zu können, müssen Sie zuerst das aktuell angewählte PDO abwählen.

Aktivierung der PDO-Zuordnung

✓ Wenn Sie die PDO-Zuordnung geändert haben, muss zur Aktivierung der neuen PDO-Zuordnung

a) der EtherCAT-Slave einmal den Statusübergang PS (von Pre-Operational zu Safe-Operational) durchlaufen (siehe Karteireiter Online [▶ 299])

b) der System-Manager die EtherCAT-Slaves neu laden

(Schaltfläche  bei TwinCAT 2 bzw.  bei TwinCAT 3)

PDO-Liste

Liste aller von diesem EtherCAT-Gerät unterstützten PDOs. Der Inhalt des ausgewählten PDOs wird der Liste *PDO-Content* angezeigt. Durch Doppelklick auf einen Eintrag können Sie die Konfiguration des PDO ändern.

Spalte	Beschreibung	
Index	Index des PDO.	
Size	Größe des PDO in Byte.	
Name	Name des PDO. Wenn dieses PDO einem Sync-Manager zugeordnet ist, erscheint es als Variable des Slaves mit diesem Parameter als Namen.	
Flags	F	Fester Inhalt: Der Inhalt dieses PDO ist fest und kann nicht vom System-Manager geändert werden.
	M	Obligatorisches PDO (Mandatory). Dieses PDO ist zwingend Erforderlich und muss deshalb einem Sync-Manager Zugeordnet werden! Als Konsequenz können Sie dieses PDO nicht aus der Liste *PDO-Zuordnungen* streichen
SM	Sync-Manager, dem dieses PDO zugeordnet ist. Falls dieser Eintrag leer ist, nimmt dieses PDO nicht am Prozessdatenverkehr teil.	
SU	Sync-Unit, der dieses PDO zugeordnet ist.	

PDO-Inhalt

Zeigt den Inhalt des PDOs an. Falls das Flag F (fester Inhalt) des PDOs nicht gesetzt ist, können Sie den Inhalt ändern.

Download

Falls das Gerät intelligent ist und über eine Mailbox verfügt, können die Konfiguration des PDOs und die PDO-Zuordnungen zum Gerät herunter geladen werden. Dies ist ein optionales Feature, das nicht von allen EtherCAT-Slaves unterstützt wird.

PDO-Zuordnung

Falls dieses Kontrollkästchen angewählt ist, wird die PDO-Zuordnung die in der PDO-Zuordnungsliste konfiguriert ist beim Startup zum Gerät herunter geladen. Die notwendigen, zum Gerät zu sendenden Kommandos können in auf dem Karteireiter Startup [▶ 296] betrachtet werden.

PDO-Konfiguration

Falls dieses Kontrollkästchen angewählt ist, wird die Konfiguration des jeweiligen PDOs (wie sie in der PDO-Liste und der Anzeige PDO-Inhalt angezeigt wird) zum EtherCAT-Slave herunter geladen.

8.1.2 Fast-Mode

Der Fast Mode bei Beckhoff EtherCAT Klemmen hat sich historisch entwickelt und ist eine Betriebsart, um EL-Klemmen vorrangig der Gruppen EL3xxx und EL4xxx (analoge Ein/Ausgangs-Klemmen) mit einer deutliche schnelleren Wandlungszeit zu betreiben. Somit kann ein analoger Eingangswert schneller/öfter gewandelt werden bzw. entsprechend über die Steuerung ausgegeben werden. Dies geht auf Kosten anderer Features, deshalb ist eine Abwägung erforderlich.

Wenn eine EL-Klemme diesen Modus unterstützt, ist dies in der entsprechenden Dokumentation angegeben.

Es gibt 2 Gruppen von Klemmen, die den FastMode unterstützen:

- Baumuster von EL3xx2 und EL4xx2 mit Produkteinführung vor 2009:
  Die zweikanaligen analogen Ein- und Ausgangsklemmen können Sie durch **Abschalten des zweiten Kanals** in den *Fast-Mode* schalten. Im einkanaligen Betrieb (*Fast-Mode*) verringert sich die Wandlungszeit der Klemme gegenüber dem zweikanaligen Betrieb um ungefähr ein Drittel. Die genauen Werte für die Wandlungszeit im einkanaligen und zweikanaligen Betrieb entnehmen Sie bitte den technischen Daten zur jeweiligen Klemme.
- EL3xxx und EL4xxx ab Herstellungsjahr 2009
  Bei diesen Klemmen kann (wenn laut Dokumentation möglich) der **CoE-Zugriff deaktiviert** werden. Dadurch können alle vorhandenen Kanäle schneller wandeln.

Für beide Gruppen dazu je ein Beispiel.

FastMode durch Kanaldeaktivierung

Beispiel 1

Auf dem Karteireiter *Prozessdaten* der EL3101 können Sie unter *PDO-Zuordnung* mit Hilfe des Kontrollkästchens (siehe roter Pfeil) den zweiten Eingangskanal ein- und ausschalten.

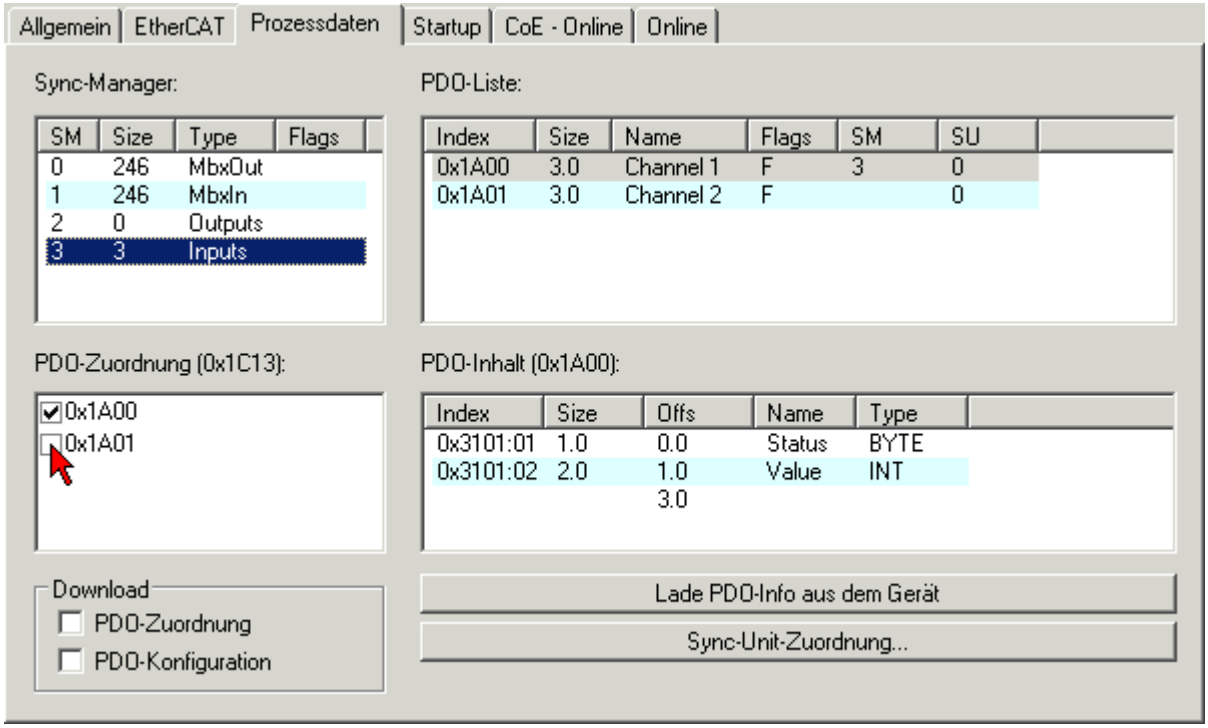


Abb. 322: Reiter „Prozessdaten“

Beispiel 2

Auf dem Karteireiter *Prozessdaten* der EL4101 können Sie unter *PDO-Zuordnung* mit Hilfe des Kontrollkästchens (siehe roter Pfeil) den zweiten Ausgangskanal ein- und ausschalten.

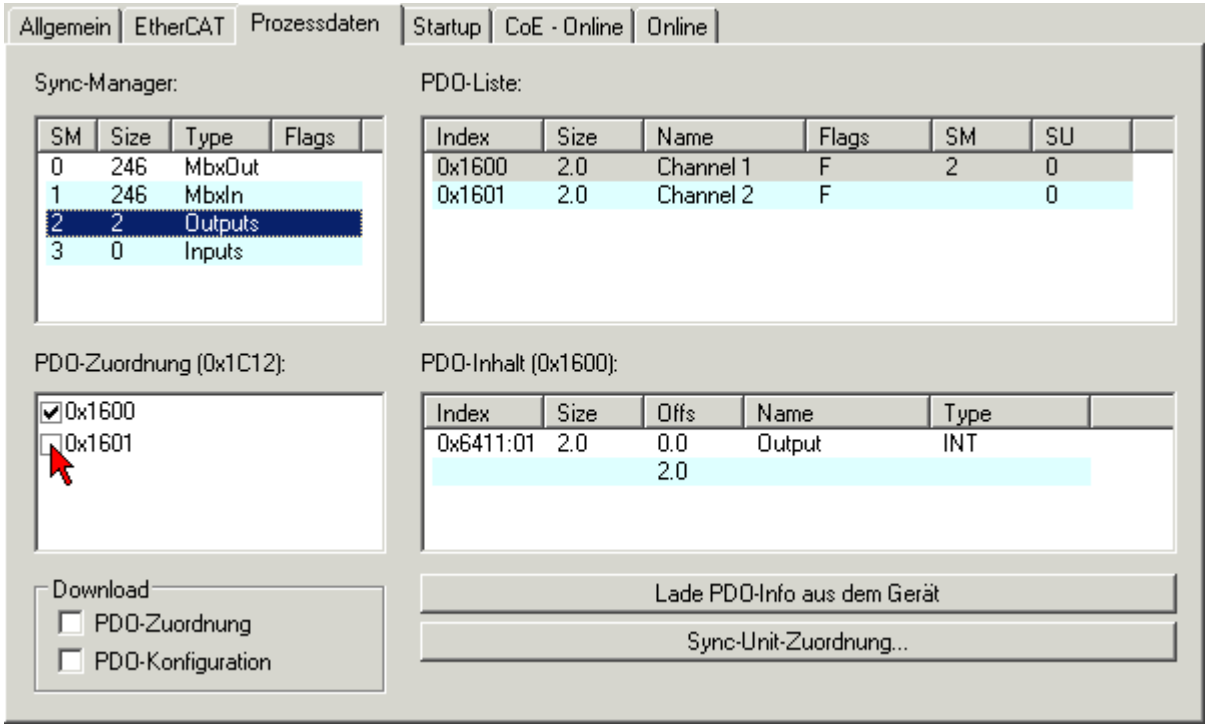


Abb. 323: Reiter „Prozessdaten“

FastMode durch CoE-Deaktivierung

Zum Deaktivieren der CoE-Unterstützung muss in die StartUp-Liste im System Manager bei der Klemme ein

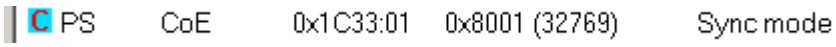


Abb. 324: Eintrag in StartUp-Liste

eingetragen werden. Dadurch ist das CoE später im SAFEOP und OP deaktiviert.

Ganz allgemein wird durch diesen FastMode der Mailbox-Verkehr dieser Klemme abgeschaltet.

Der CoE-Zugriff kann wieder aktiviert werden, indem der ursprüngliche Wert oder z. B. 0x00 in der PREOP-Phase nach CoE 0x1C33:01 geschrieben wird. Siehe dazu auch die Einträge in der Synchronisationsmodi [▶ 226]-Übersicht.

8.2 Erweiterte Konfiguration

8.2.1 Verhalten

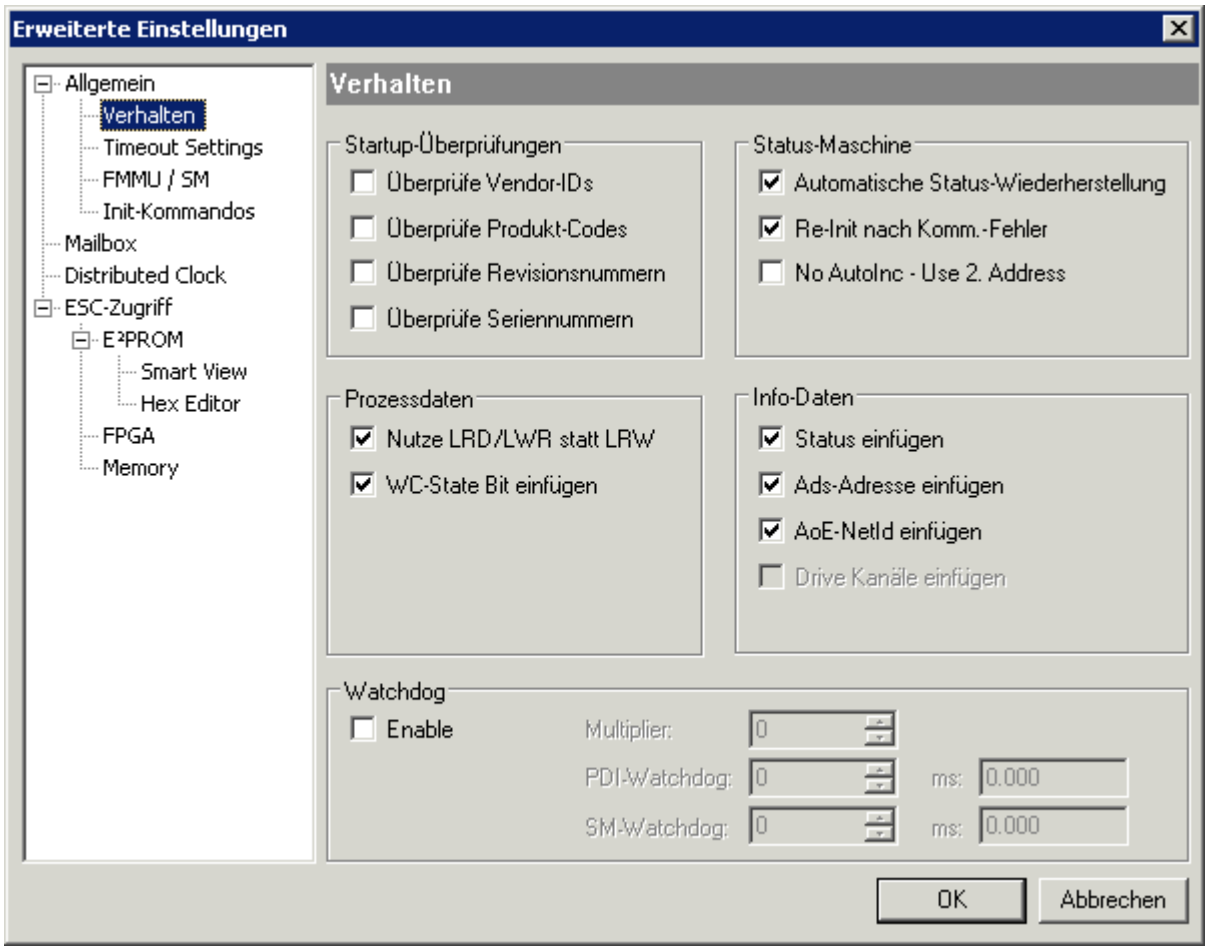


Abb. 325: Advanced Settings: „Verhalten“-Dialog

Startup-Überprüfungen

Hier können Sie festlegen, welche Slave-Informationen der EtherCAT-Master während des Start-Ups überprüfen soll.

Überprüfe Vendor-IDs

Hier können Sie einstellen, dass der EtherCAT-Master die Vendor-ID eines jeden Slaves mit der konfigurierten Vendor-ID vergleicht (Default: inaktiv).

Überprüfe Produkt-Codes

Hier können Sie einstellen, dass der EtherCAT-Master den Produkt-Code eines jeden Slaves mit dem konfigurierten Produkt-Code vergleicht (Default: inaktiv).

Überprüfe Revisions-Nummern

Hier können Sie einstellen, dass der EtherCAT-Master die Revisions-Nummer eines jeden Slaves mit der konfigurierten Revisions-Nummer vergleicht (Default: inaktiv).

Überprüfe Serien-Nummern

Hier können Sie einstellen, dass der EtherCAT-Master die Serien-Nummer eines jeden Slaves mit der konfigurierten Serien-Nummer vergleicht (Default: inaktiv).

Status-Maschine

Auto Status-Wiederherstellung

Hier können Sie einstellen, dass der EtherCAT-Master versucht den Status eines EtherCAT-Slaves automatisch wiederherzustellen (Default: aktiv). Wenn ein EtherCAT-Slave seinen Status von einem Fehler-Status (ERR SAVE-OP, ERR OP usw.) in einen regulären Status (SAFE-OP, OP usw.) ändert, versucht der Master den Slave in den Zustand des Masters zu setzen.

Re-Init nach Kommunikationsfehler

Hier können Sie einstellen, dass der EtherCAT-Master den Slave nach einem Kommunikationsfehler in den Status *Init* zurücksetzt (Default: aktiv).

No AutoInc - Use 2. Address

Wenn dieses Kontrollkästchen aktiviert ist, dann Adressiert der EtherCAT-Master diesen EtherCAT-Slave in der Hochlaufphase nicht anhand der Position im EtherCAT-Ring, sondern liest eine feste Adresse aus dem Slave aus (Default: inaktiv).

Prozessdaten

Nutze LRD/LWR statt LRW

Hier können Sie einstellen, dass ein LRD-Kommando (Logical Read) für das lesen der Eingänge und ein LWR-Kommando (Logical Write) für das setzen der Ausgänge verwendet wird (Default aktiv). Andernfalls wird ein LRW-Kommando (Logical Read/Write) für das lesen der Eingänge und das setzen der Ausgänge verwendet.

WC-Status-Bit einfügen

Hier können Sie einstellen, dass eine Eingangsvariable die den des Working-Counter-Status des EtherCAT-Slaves anzeigt zu dessen Prozessabbild hinzugefügt wird (Default: aktiv).

Info-Daten

Um diese Optionen nutzen zu können, müssen Sie die Info-Daten den Master-Einstellungen aktiviert haben.

Status einfügen

Hier können Sie einstellen, dass die Eingangsvariable *Status* zum InfoData-Eintrag eine jeden EtherCAT-Slaves hinzugefügt wird (Default: aktiv). Diese Variable enthält den aktuellen EterCAT-Status und Link-Status eines EtherCAT-Slaves.

ADS-Adresse einfügen

Hier können Sie einstellen, dass die Eingangsvariable *AdsAddress* zum InfoData-Eintrag eine jeden EtherCAT-Slaves hinzugefügt wird (Default: aktiv für alle EtherCAT-Slaves, die Mailbox-Protokolle wie CoE (CANopen over EtherCAT) oder SoE (Servo over EtherCAT) unterstützen.)

AoE-NetID einfügen

Wenn dieses Kontrollkästchen aktiviert ist, wird die NetID für ADS over EtherCAT eingefügt (Default: inaktiv).

Drive-Kanäle einfügen

Hier können Sie einstellen, dass die Eingangsvariable *ChnX* (X = Kanalnummer) zum InfoData-Eintrag eine jeden EtherCAT-Drives hinzugefügt wird (Default: inaktiv).

Watchdog

Hier könne Sie das Verhalten der Watchdogs konfigurieren.

Enable

Wenn dieses Kontrollkästchen aktiviert ist, sind die Watchdogs eingeschaltet.

Multiplier

Multiplikator

PDI-Watchdog

Watchdog für das Prozessdaten-Interface

SM-Watchdog

Watchdog für die Sync-Manager

8.2.1.1 Kanäle in InfoData einblenden (Include Channels)

Es kann hilfreich sein, dass der Steuerung/PLC bekannt ist, welchen Kanal eines IO Moduls (Klemme, Box) sie nutzt. Beispiele:

- Es wird eine 4-kanalige Analoge Eingangsklemme EL3104 verwendet. In der Steuerung nimmt ein Funktionsblock (FB) die Daten an. Der FB erwartet die Information, auf welchen Kanal der EL3104 er verlinkt ist, z. B. der 3. oder 4.
- Ein Modul verfügt über technologisch unterschiedliche Kanäle und die Steuerung erwartet die Information, mit welchem Technologie-Kanal sie verlinkt ist.
  Beispiel: die analoge Brückenklemme EL3356 arbeitet in der Standardeinstellung als 1-kanalige Last-Klemme und gibt das gemessene Gewicht in kg aus. Alternativ kann sie auf 2-kanalige Spannungsmessung umkonfiguriert werden (Speisespannung und Brückenspannung). Insgesamt bietet die Klemme also drei Kanäle zur Nutzung an (wenn auch nicht alle gleichzeitig).

Lösung: in TwinCAT 2.11 und 3 können die Kanäle eines Moduls in den InfoDaten eingeblendet werden.

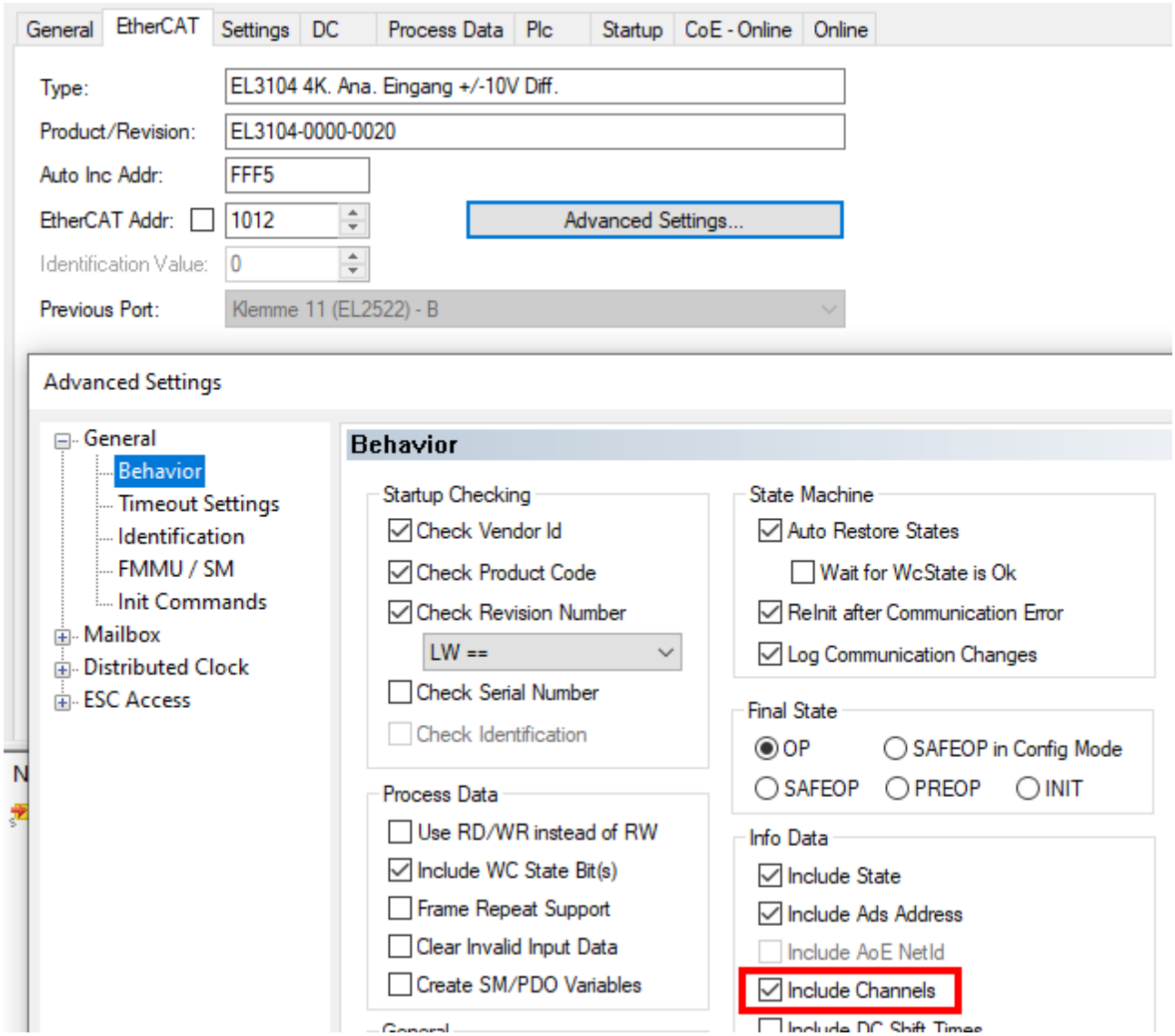


Abb. 326: Einblenden der Kanäle in InfoData über „Erweiterte Einstellungen“ -> „Verhalten“ „Include Channels“

TwinCAT zeigt dann verlinkbare Channels an:

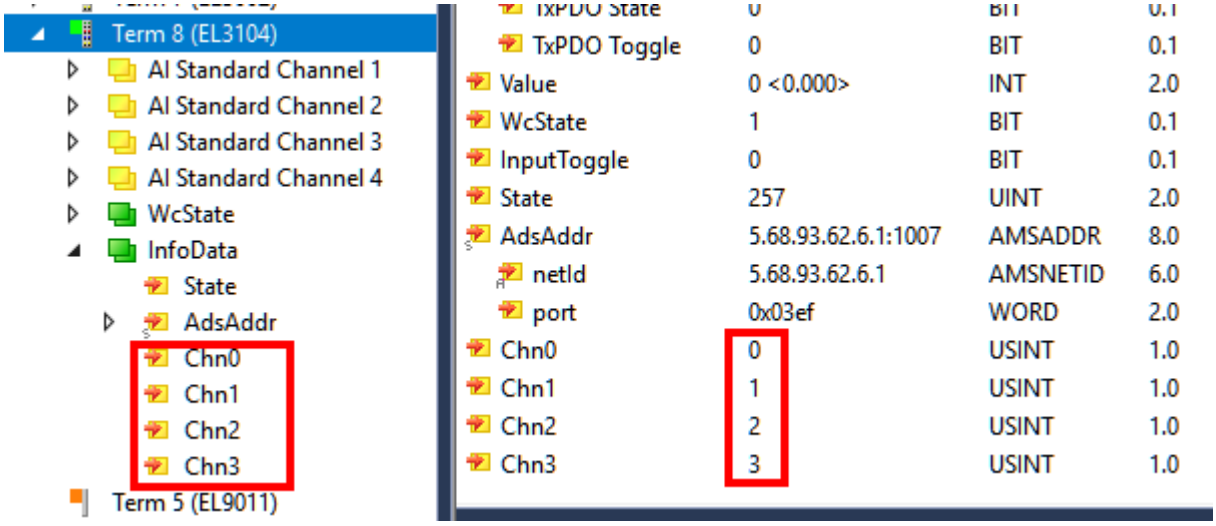
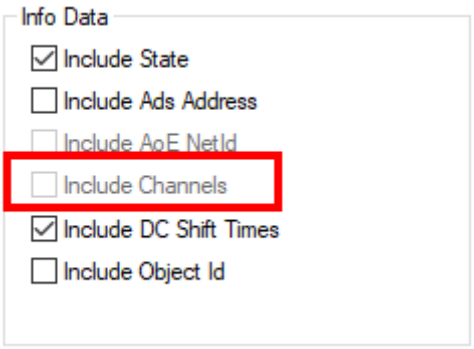


Abb. 327: Anzeige verlinkbarer Kanäle im TwinCAT

Der Inhalt der Bytes ist fix und nicht veränderbar, nämlich die fortlaufende Kanalnummer.

Hintergrund:

- TwinCAT orientiert sich dabei an den Profilangaben in der ESI-Datei → „Profile“. Bei Modulen die diese Info (noch) nicht in der ESI haben, können die Kanäle also nicht aktiviert werden



In den Advanced Settings des EtherCAT Masters wird informativ angezeigt, wenn in einem der Slaves die Kanäle aktiviert sind.

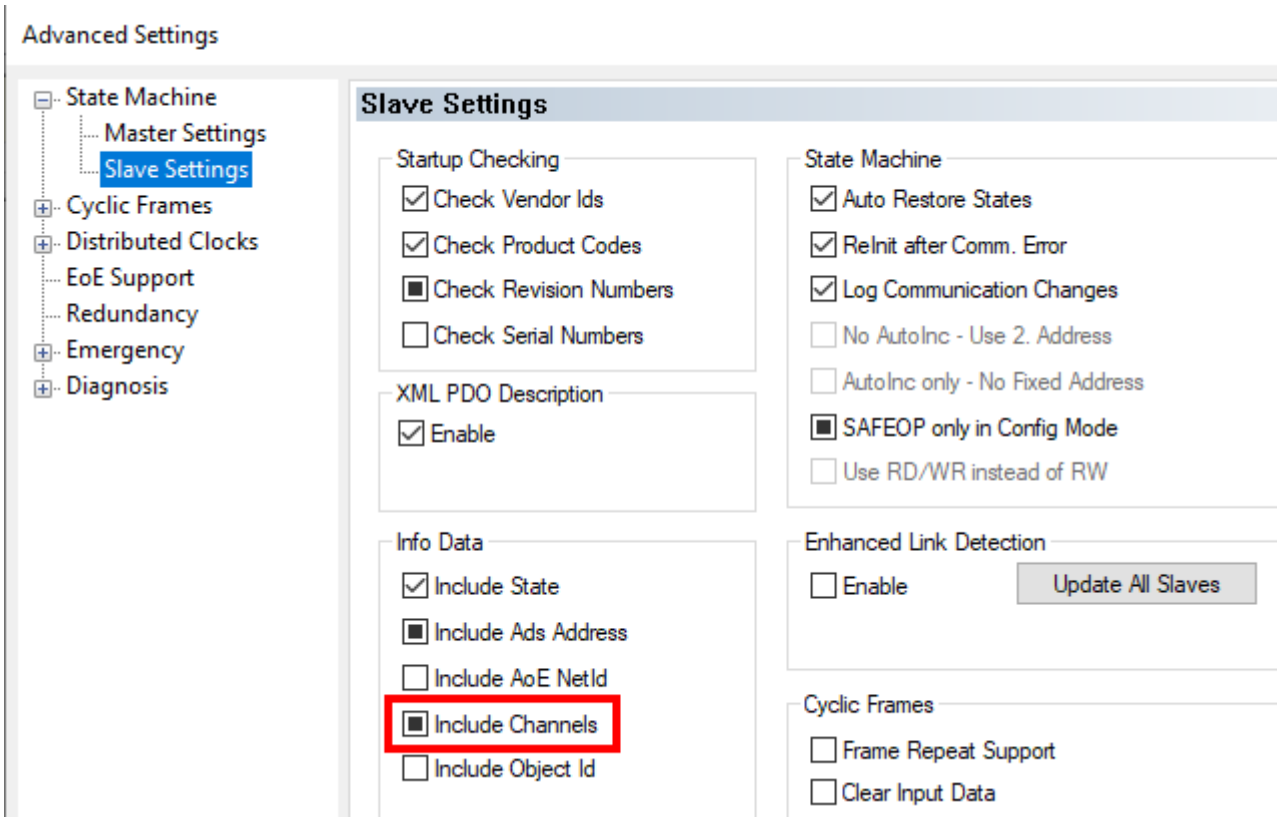


Abb. 328: Anzeige der Aktivierung von „Include Channels“ im EtherCAT Master

8.2.2 Timeout-Einstellungen

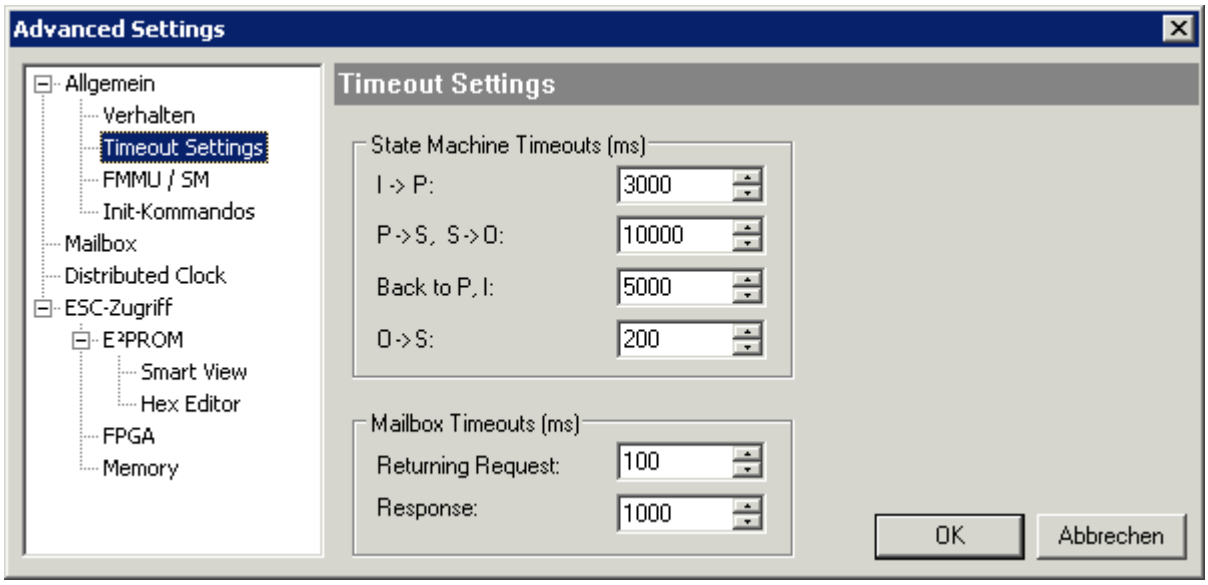


Abb. 329: Advanced Settings: „Timeout Settings“-Dialog

State-Machine Timeouts (ms)

Hier können Sie die Zeiten für die Timeouts beim Übergang von einem Status der State-Machine zu einem anderen parametrieren.

I -> P

Timeout für den Übergang vom Status **I**nit in den Status **P**re-Operational.

P -> S, S -> O

Timeout für den Übergang

- vom Status **P**re-Operational in den Status **S**afe-Operational oder
- vom Status **S**afe-Operational in den Status **O**perational .

Back to P, I

Timeout für die Rückkehr zum Status **P**re-Operational oder **I**nit.

O -> S

Timeout für den Übergang vom Status **O**perational nach in den Status **S**afe-Operational.

Mailbox Timeouts (ms)

Hier können Sie die Timeouts für azyklische Kommandos auf die Mailbox (Mailbox-Interface) parametrieren.

Returning Request

Timeout für die Rückkehr eines Requests aus dem EtherCAT-Ring.

Response

Timeout für die Antwort (Response) des angesprochenen EtherCAT-Geräts auf den Request.

8.2.3 FMMU / SM

Dieser Dialog zeigt die aktuelle Konfiguration der Fieldbus Memory Management Units (FMMU) und der Sync-Manager (SM) an und erlaubt Ihnen diese zu ändern.

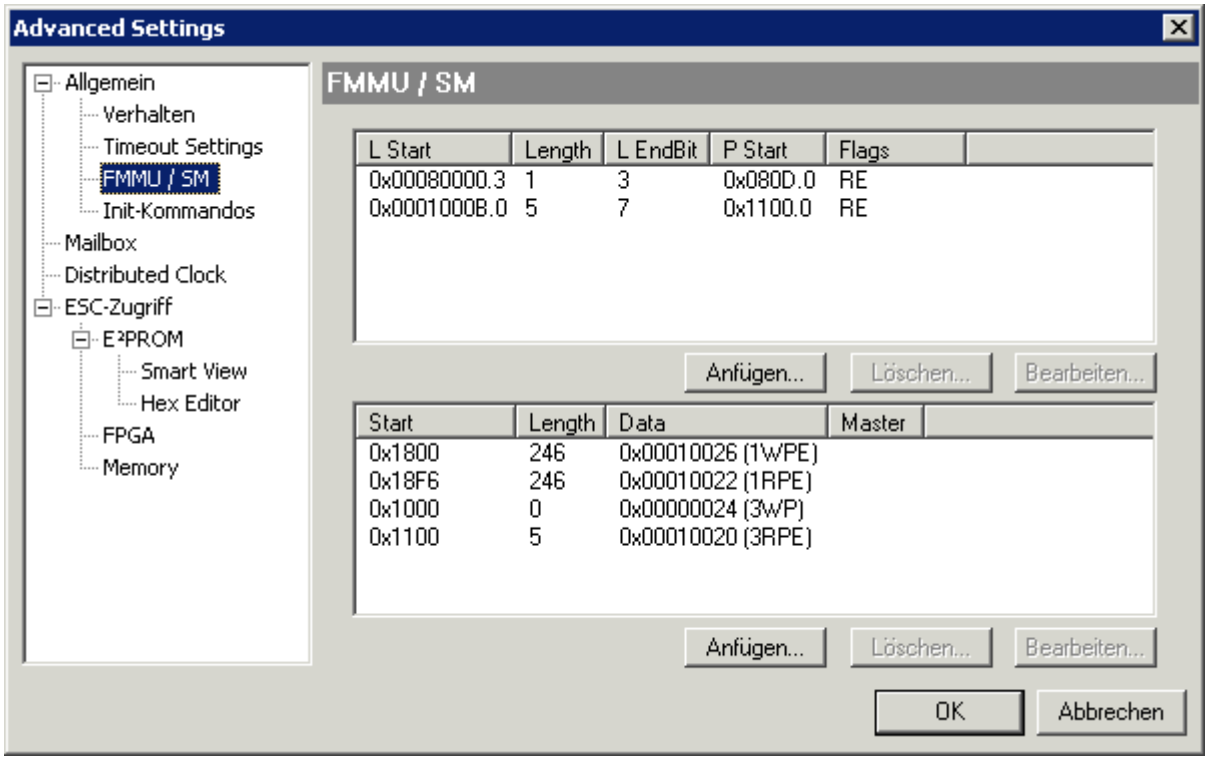


Abb. 330: Advanced Settings: „FMMU/SM“-Dialog

Konfiguration der FMMUs (obere Liste)

L Start

Spezifiziert von welcher logischen Adresse aus die FMMU beginnt die Daten zu mappen. Das Start-Bit wird passend zur Nummer gesetzt, die dem Punkt folgt (0xnnnnnnnn.**Start-Bit**)

Lenght

Spezifiziert wie viele Bytes von der logischen Adressierung gemapped werden.

L EndBit

End-Bit der logischen Adresse. Wenn die Logische Adresse zu einem Byte konfiguriert werden soll, muss das Start-Bit auf 0 konfiguriert werden (L Start = 0xnnnnnnnn.**0**).

P Start

Spezifiziert die Physikalische Adresse, auf die die logische Adresse zeigt.

Flags

RE: Read Enabled
WE: Write enabled

Konfiguration der Sync-Manager (untere Liste)

Start

Spezifiziert von welcher Adresse an der Sync-Channel aktiv ist.

Lenght

Länge des Sync-Channels in Byte. Wenn der Sync-Channel nicht aktiviert ist, ist die Länge 0.

Data

Zum Sync-Channel zu schreibende Konfigurationsdaten.

Master

(in Vorbereitung)

8.2.4 Mailbox

Wenn der EtherCAT-Slave ein oder mehrere Mailbox-Protokolle unterstützt, wird der zusätzliche Karteireiter *Mailbox* angezeigt. Dieser Dialog listet die vom EtherCAT-Slave unterstützten Mailbox-Protokolle auf und ermöglicht Änderungen an deren Konfiguration.

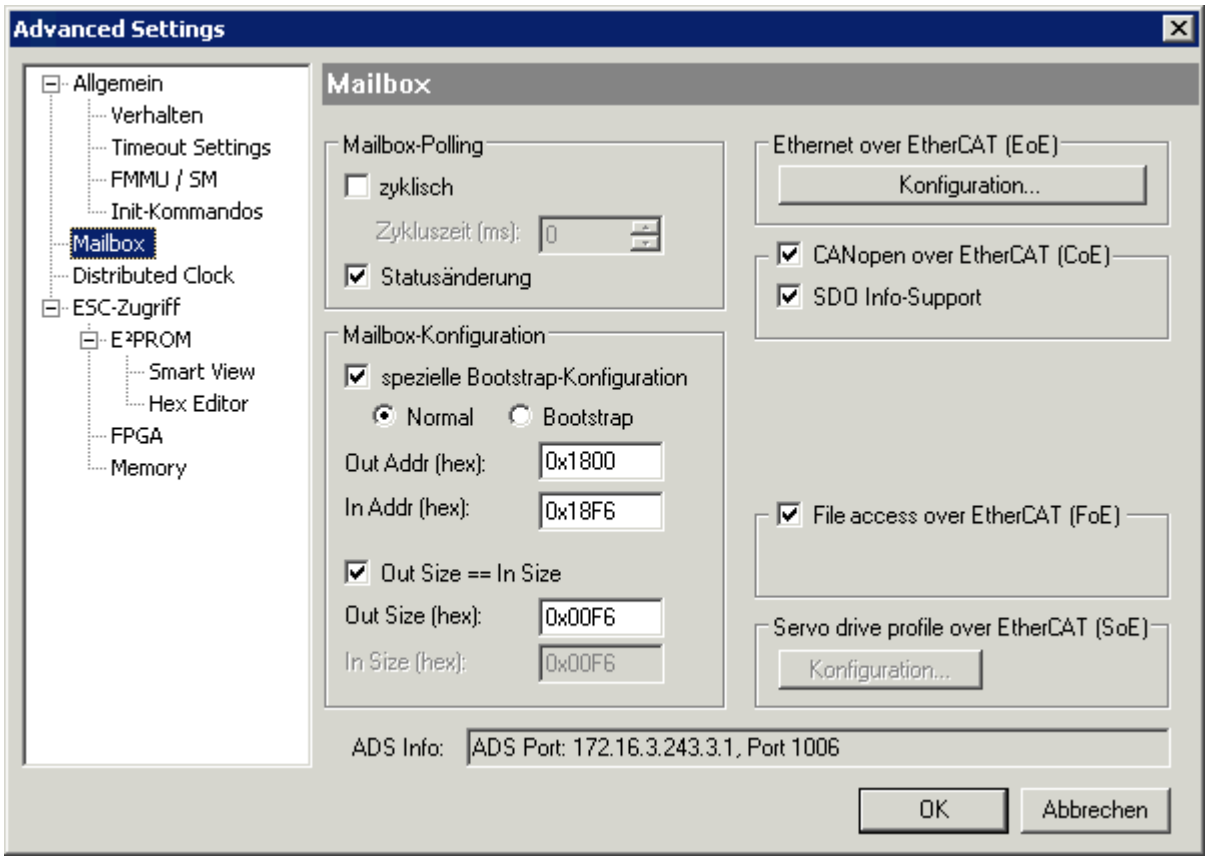


Abb. 331: Advanced Settings: „Mailbox“-Dialog

Mailbox Polling

- **Zyklisch**
  Wenn dieses Kontrollkästchen angewählt ist, ließt der Master zyklisch die Mailbox es EtherCAT-Slaves aus.

- **Zykluszeit (ms)**
  Wenn das Kontrollkästchen *Zyklisch* angewählt ist, spezifiziert dieser Wert wie oft der Master die Mailbox des EtherCAT-Slaves ausliest.
- **Statusänderung**
  Wenn dieses Kontrollkästchen angewählt ist, überprüft der Master ein Statusbit des Slaves um festzustellen, ob ungelesene Daten in der Mailbox zur Verfügung stehen. Nur dann liest der Master die Mailbox aus. Dieser Modus ist effizienter als der zyklische Modus, weil der Master den Status der Mailboxen mehrerer EtherCAT-Slaves mit einem einzigen EtherCAT-Kommando (LRD) überprüfen kann.

Mailbox Konfiguration

- **Spezielle Bootstrap-Konfig**
  Normal: für Prozessdatenverkehr oder Mailbox-Protokolle
  Bootstrap: für Firmware-Download
- **Out-Adr.**
  Physikalische Startadresse der Ausgangs-Mailbox im Slave-Controller.
- **In-Adr.**
  Physikalische Startadresse der Eingangs-Mailbox im Slave-Controller.
- **Out-Size == In Size**
  Wenn Sie dieses Kontrollkästchen markieren, dann sind Out-Size und In-Size identisch.
- **Out-Size**
  Größe der Ausgangs-Mailbox in Byte.
- **In Size**
  Größe der Eingangs-Mailbox in Byte.

Mailbox Protokolle

- **Ethernet over EtherCAT (EoE)**
  - **Ethernet over EtherCAT (EoE)**
    Wenn diese Gruppe freigeschaltet ist, unterstützt der EtherCAT-Slave das Mailbox-Protokoll *Ethernet over EtherCAT* (EoE).
  - **Konfiguration**
    Diese Schaltfläche öffnet einen Dialog zur Konfiguration des Mailbox-Protokolls *Ethernet over EtherCAT*.
- **CANopen over EtherCAT (CoE)**
  - **CANopen over EtherCAT (CoE)**
    Wenn dieses Kontrollkästchen aktiviert ist, unterstützt der EtherCAT-Slave das Mailbox-Protokoll *CANopen over EtherCAT* (CoE).
  - **SDO-Info-Support**
    Wenn dieses Kontrollkästchen aktiviert ist, kann das Objektverzeichnis des EtherCAT-Slaves vom Master geladen werden.
- **File access over EtherCAT (FoE)**
  - **File access over EtherCAT (FoE)**
    Wenn dieses Kontrollkästchen aktiviert ist, unterstützt der EtherCAT-Slave das Mailbox-Protokoll *File access over EtherCAT* (FoE).
- **Servo drive profile over EtherCAT (SoE)**
  - **Servo drive profile over EtherCAT**
    Wenn diese Gruppe freigeschaltet ist, unterstützt der EtherCAT-Slave das Mailbox-Protokoll *Servo drive over EtherCAT* (SoE).
  - **Konfiguration**
    Diese Schaltfläche öffnet einen Dialog zur Konfiguration des Mailbox-Protokolls *Servo drive over EtherCAT*.

ADS-Info

ADS-Identifizierung eines EtherCAT-Slaves.

- Die ADS Net ID ist die gleiche, wie die NetId eines EtherCAT-Geräts.
- Der ADS-Port ist der gleiche wie die feste Adresse eines EtherCAT-Geräts (siehe EtherCAT Adr).

Mit Hilfe von ADS können Sie mit der Mailbox des EtherCAT-Slaves kommunizieren (z. B. SDO Upload Request).

8.2.5 Smart View

Dieser Dialog zeigt die Einstellungen, die im EEPROM des EtherCAT-Slave-Controllers (ESC) gespeichert sind.

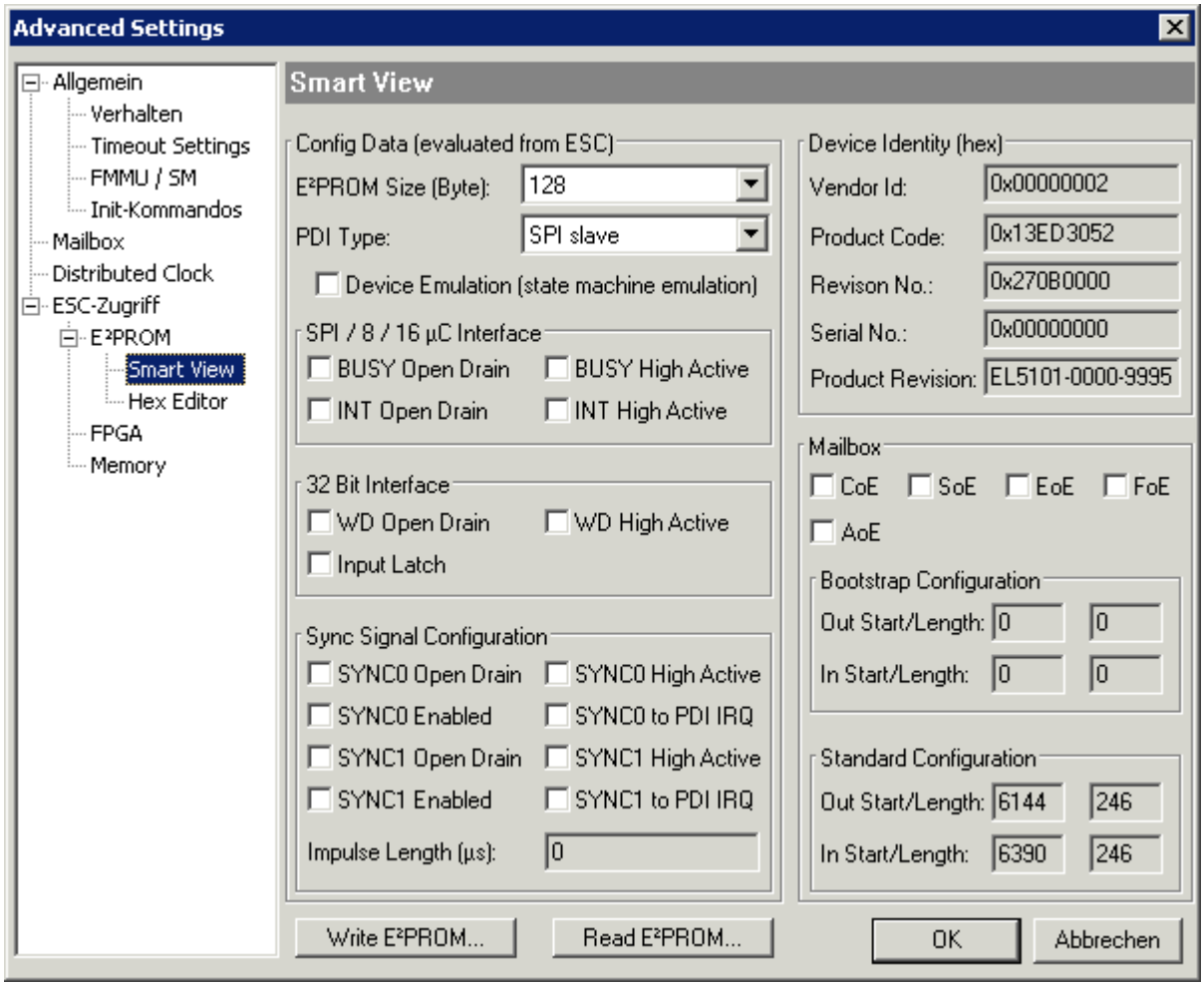


Abb. 332: Advanced Settings: „Smart View“-Dialog

Config Data (evaluated from ESC)

Dieser Abschnitt zeigt interne Parameter des EtherCAT-Slave-Controllers (ESC) an.

- EEPROM Size (Byte): Größe des EEPROMs in Byte
- PDI Type: Typ des Prozessdaten-Interfaces
- Device Emulation (state machine emulation)

SPI / 8 / 18 µC Interface

Dieser Abschnitt zeigt interne Parameter des 16 Bit Prozessdaten-Interfaces an:

- BUSY Open Drain
- BUSY High Active
- INT Open Drain

- INT High Active

32 Bit Interface

Dieser Abschnitt zeigt interne Parameter des 32 Bit Prozessdaten-Interfaces an:

- WD Open Drain: Watchdog Open Drain
- WD High Active: Watchdog Open High Active
- Input Latch

Sync Signal Configuration

Dieser Abschnitt zeigt interne Parameter zur Konfiguration des Sync-Signals an:

- Sync 0 Open Drain
- Sync 0 High Active
- Sync 0 Enabled
- Sync 0 to PDI IRQ
- Sync 1 Open Drain
- Sync 1 High Active
- Sync 1 Enabled
- Sync 1 to PDI IRQ

Schaltflächen für das EEPROM

- Write EEPROM: Schaltfläche zum Beschreiben des EEPROM.
- Read EEPROM: Schaltfläche zum Auslesen des EEPROM.

Device Identity (hex)

Dieser Abschnitt zeigt Informationen zur EtherCAT-Gerät an.

- Vendor ID: Identifikationsnummer des Geräteherstellers.
- Product Code: Produkt-Code des EtherCAT-Geräts.
- Revision No.: Revisionsnummer des EtherCAT-Geräts.
- Serial No.: Seriennummer des EtherCAT-Geräts.
- Product Revision: Produkt-Revision des EtherCAT-Geräts.

Mailbox

Zeigt an, welche Varianten der Mailbox-Kommunikation unterstützt werden.

- CoE: CanOpen over EtherCAT
- SoE: Servo-Profile over EtherCAT
- EoE: Ethernet over EtherCAT
- FoE: Servo-Profile over EtherCAT
- AoE: ADS over EtherCAT

Bootstrap Configuration

Konfiguration für den Bootstrap-Mode (z. B. Firmware-Update)

- Out Start/Lenght
- In Start/Lenght

Standard Configuration

Konfiguration für den regulären Betrieb (Prozessdatenübertragung oder Mailbox-Betrieb)

- Out Start/Lenght
- In Start/Lenght

8.2.6 Memory

Der Memory-Dialog erlaubt dem Anwender, Daten aus dem Speicher (DPRAM) des EtherCAT-Slave-Controllers zu lesen oder dort zu speichern. Die untenstehend Liste zeiget den Speicher des EtherCAT-Slave-Controllers an. Der Start-Offset entspricht dem in der Box *Start Offset* eingetragenem Wert. Jeder Eintrag zeigt ein Register (2 Byte). Wenn der System-Manager eine Beschreibung für ein Register kennt, wird diese neben dem *Offset* angezeigt. Um einen Registerwert zu ändern tragen Sie den gewünschten Wert in die Spalte Dec, Hex oder Char ein. Nach der Änderung eines Werts wird dieser rot dargestellt und die Schaltfläche *Write* wird freigegeben. Nun können Sie die Schaltfläche *Write* drücken um geänderte Werte zum EtherCAT-Slave zu übertragen.

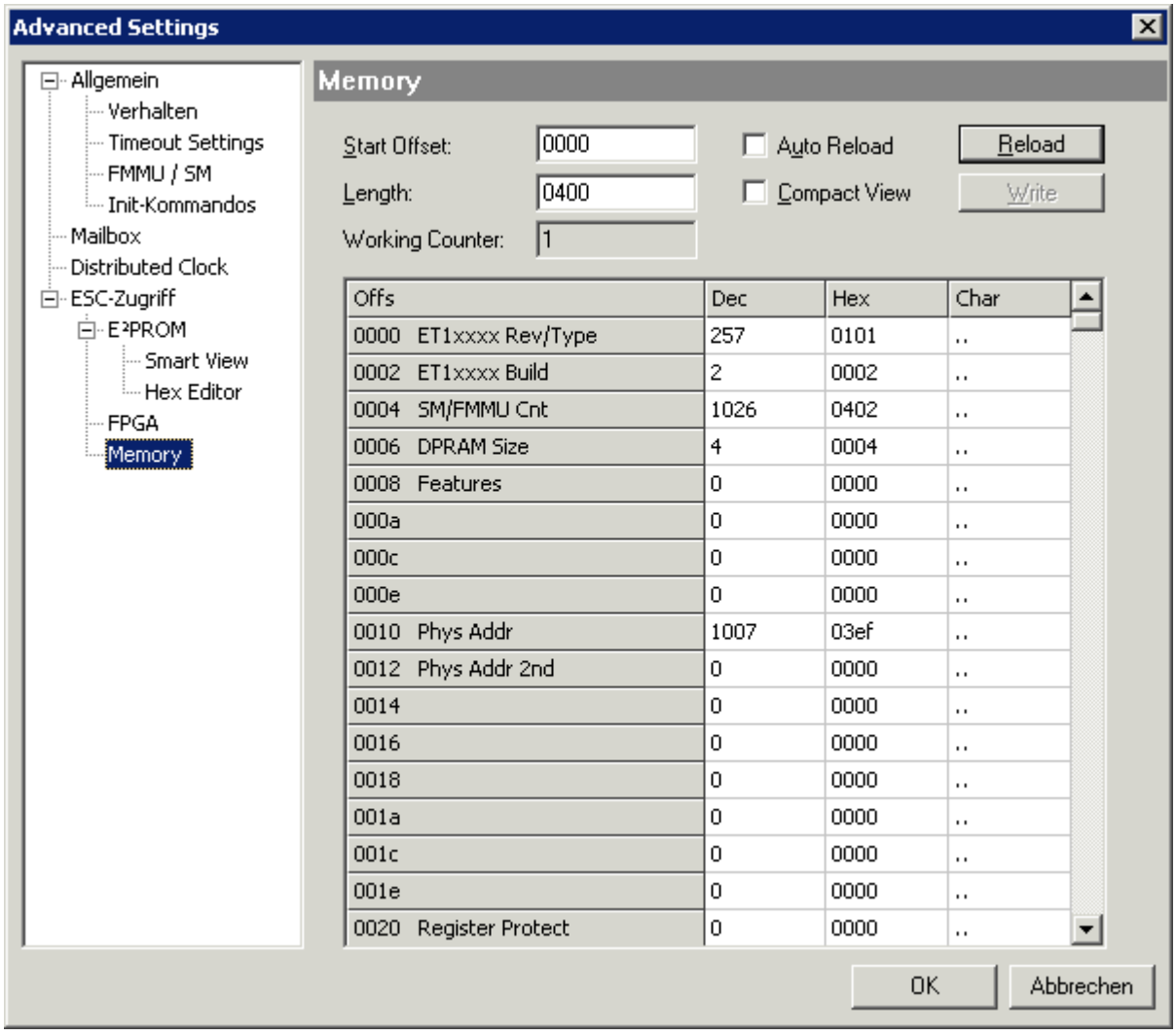


Abb. 333: Advanced Settings: „Memory“-Dialog

Start Offset

Startadresse (hexadezimal) des ersten Registers, das in der Liste angezeigt werden soll.

Lenght

Länge (hexadezimal) der anzuzeigenden Daten in Bytes. Die maximal zulässige Länge ist 0400_{hex} (1024_{dez}).

Working Counter

If the master succeeded in writing to or reading from the slave, the working counter is 1, otherwise the working counter is 0.

Auto Reload

Hier können Sie das zyklische Auslesen des Speichers aktivieren (Default: deaktiviert).

Compact View

Hier können Sie festlegen, dass nur die Parameter angezeigt werden, für die der System-Manager eine Beschreibung hat (Default: nicht aktiv).

Reload

Liest die Werte aktuellen aus dem EtherCAT-Slave.

Write

Speichert geänderte Werte (rot dargestellt) auf dem EtherCAT-Slave.

8.3 Firmware Update EL/ES/ELM/EM/EPxxxx

Dieses Kapitel beschreibt das Geräte-Update für Beckhoff EtherCAT Slaves der Serien EL/ES, ELM, EM, EK und EP. Ein FW-Update sollte nur nach Rücksprache mit dem Beckhoff Support durchgeführt werden.

HINWEIS

Nur TwinCAT 3 Software verwenden!

Ein Firmware-Update von Beckhoff IO Geräten ist ausschließlich mit einer TwinCAT3-Installation durchzuführen. Es empfiehlt sich ein möglichst aktuelles Build, kostenlos zum Download verfügbar auf der Beckhoff-Website https://www.beckhoff.com/de-de/.

Zum Firmware-Update kann TwinCAT im sog. FreeRun-Modus betrieben werden, eine kostenpflichtige Lizenz ist dazu nicht nötig.

Das für das Update vorgesehene Gerät kann in der Regel am Einbauort verbleiben; TwinCAT ist jedoch im FreeRun zu betreiben. Zudem ist auf eine störungsfreie EtherCAT Kommunikation zu achten (keine „LostFrames“ etc.).

Andere EtherCAT-Master-Software wie z.B. der EtherCAT-Konfigurator sind nicht zu verwenden, da sie unter Umständen nicht die komplexen Zusammenhänge beim Update von Firmware, EEPROM und ggf. weiteren Gerätebestandteilen unterstützen.

Speicherorte

In einem EtherCAT-Slave werden an bis zu drei Orten Daten für den Betrieb vorgehalten:

- Je nach Funktionsumfang und Performance besitzen EtherCAT Slaves einen oder mehrere lokale Controller zur Verarbeitung von IO-Daten. Das darauf laufende Programm ist die sog. **Firmware** im Format *.efw.
- In bestimmten EtherCAT Slaves kann auch die EtherCAT Kommunikation in diesen Controller integriert sein. Dann ist der Controller meist ein so genannter **FPGA**-Chip mit der *.rbf-Firmware.
- Darüber hinaus besitzt jeder EtherCAT Slave einen Speicherchip, um seine eigene Gerätebeschreibung (ESI; EtherCAT Slave Information) zu speichern, in einem sog. **ESI-EEPROM**. Beim Einschalten wird diese Beschreibung geladen und u. a. die EtherCAT Kommunikation entsprechend eingerichtet. Die Gerätebeschreibung kann von der Beckhoff Website (http://www.beckhoff.de) im Downloadbereich heruntergeladen werden. Dort sind alle ESI-Dateien als Zip-Datei zugänglich.

Kundenseitig zugänglich sind diese Daten nur über den Feldbus EtherCAT und seine Kommunikationsmechanismen. Beim Update oder Auslesen dieser Daten ist insbesondere die azyklische Mailbox-Kommunikation oder der Registerzugriff auf den ESC in Benutzung.

Der TwinCAT Systemmanager bietet Mechanismen, um alle drei Teile mit neuen Daten programmieren zu können, wenn der Slave dafür vorgesehen ist. Es findet üblicherweise keine Kontrolle durch den Slave statt, ob die neuen Daten für ihn geeignet sind, ggf. ist ein Weiterbetrieb nicht mehr möglich.

Vereinfachtes Update per Bundle-Firmware

Bequemer ist der Update per sog. **Bundle-Firmware**: hier sind die Controller-Firmware und die ESI-Beschreibung in einer *.efw-Datei zusammengefasst, beim Update wird in der Klemme sowohl die Firmware, als auch die ESI verändert. Dazu ist erforderlich

- dass die Firmware in dem gepackten Format vorliegt: erkenntlich an dem Dateinamen der auch die Revisionsnummer enthält, z. B. ELxxxx-xxxx_REV0016_SW01.efw
- dass im Download-Dialog das Passwort=1 angegeben wird. Bei Passwort=0 (default Einstellung) wird nur das Firmware-Update durchgeführt, ohne ESI-Update.
- dass das Gerät diese Funktion unterstützt. Die Funktion kann in der Regel nicht nachgerüstet werden, sie wird Bestandteil vieler Neuentwicklungen ab Baujahr 2016.

Nach dem Update sollte eine Erfolgskontrolle durchgeführt werden

- ESI/Revision: z. B. durch einen Online-Scan im TwinCAT ConfigMode/FreeRun – dadurch wird die Revision bequem ermittelt
- Firmware: z. B. durch einen Blick ins Online-CoE des Gerätes

HINWEIS

Beschädigung des Gerätes möglich!

✓ Beim Herunterladen von neuen Gerätedateien ist zu beachten

a) Das Herunterladen der Firmware auf ein EtherCAT-Gerät darf nicht unterbrochen werden.

b) Eine einwandfreie EtherCAT-Kommunikation muss sichergestellt sein, CRC-Fehler oder LostFrames dürfen nicht auftreten.

c) Die Spannungsversorgung muss ausreichend dimensioniert, die Pegel entsprechend der Vorgabe sein.

⇨ Bei Störungen während des Updatevorgangs kann das EtherCAT-Gerät ggf. nur vom Hersteller wieder in Betrieb genommen werden!

8.3.1 Gerätebeschreibung ESI-File/XML

HINWEIS

ACHTUNG bei Update der ESI-Beschreibung/EEPROM

Manche Slaves haben Abgleich- und Konfigurationsdaten aus der Produktion im EEPROM abgelegt. Diese werden bei einem Update unwiederbringlich überschrieben.

Die Gerätebeschreibung ESI wird auf dem Slave lokal gespeichert und beim Start geladen. Jede Gerätebeschreibung hat eine eindeutige Kennung aus Slave-Name (9-stellig) und Revision-Nummer (4-stellig). Jeder im System Manager konfigurierte Slave zeigt seine Kennung im EtherCAT-Reiter:

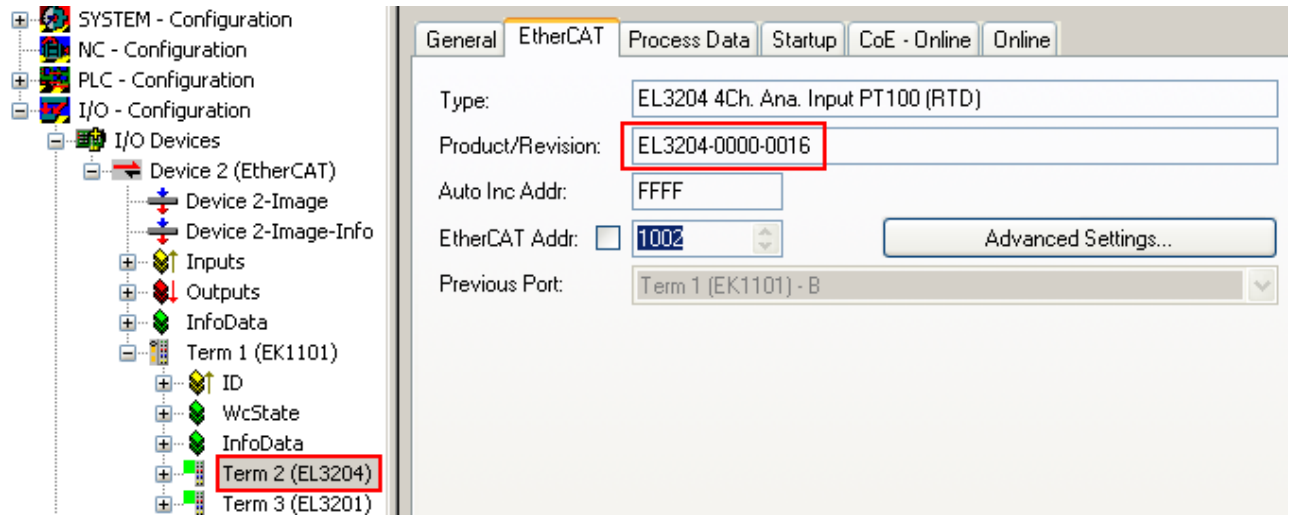


Abb. 334: Gerätekennung aus Name EL3204-0000 und Revision -0016

Die konfigurierte Kennung muss kompatibel sein mit der tatsächlich als Hardware eingesetzten Gerätebeschreibung, d. h. der Beschreibung die der Slave (hier: EL3204) beim Start geladen hat. Üblicherweise muss dazu die konfigurierte Revision gleich oder niedriger der tatsächlich im Klemmenverbund befindlichen sein.

Weitere Hinweise hierzu entnehmen Sie bitte der EtherCAT System-Dokumentation.

Update von XML/ESI-Beschreibung

Die Geräterevision steht in engem Zusammenhang mit der verwendeten Firmware bzw. Hardware. Nicht kompatible Kombinationen führen mindestens zu Fehlfunktionen oder sogar zur endgültigen Außerbetriebsetzung des Gerätes. Ein entsprechendes Update sollte nur in Rücksprache mit dem Beckhoff Support ausgeführt werden.

Anzeige der Slave-Kennung ESI

Der einfachste Weg die Übereinstimmung von konfigurierter und tatsächlicher Gerätebeschreibung festzustellen, ist im TwinCAT-Modus Config/FreeRun das Scannen der EtherCAT-Boxen auszuführen:

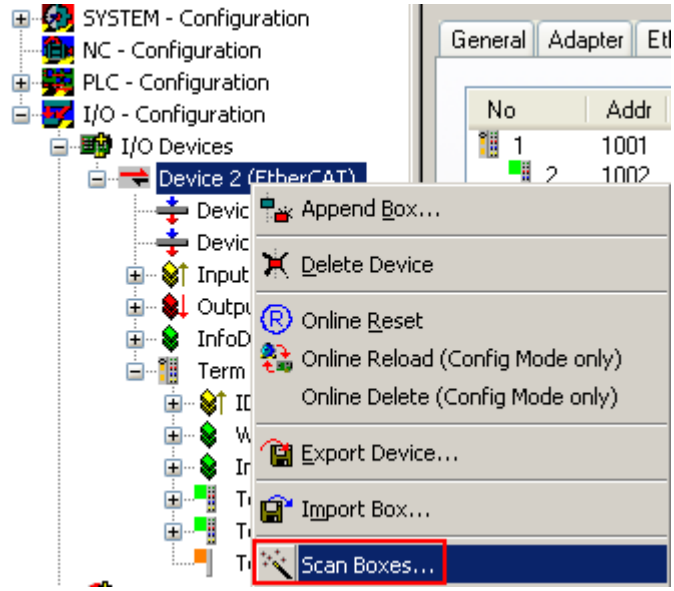


Abb. 335: Rechtsklick auf das EtherCAT Gerät bewirkt das Scannen des unterlagerten Feldes

Wenn das gefundene Feld mit dem konfigurierten übereinstimmt, erscheint

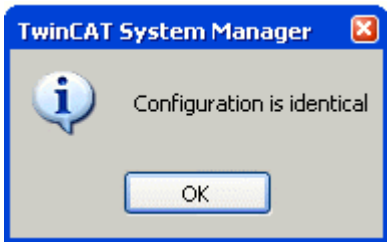


Abb. 336: Konfiguration identisch

ansonsten erscheint ein Änderungsdialog, um die realen Angaben in die Konfiguration zu übernehmen.

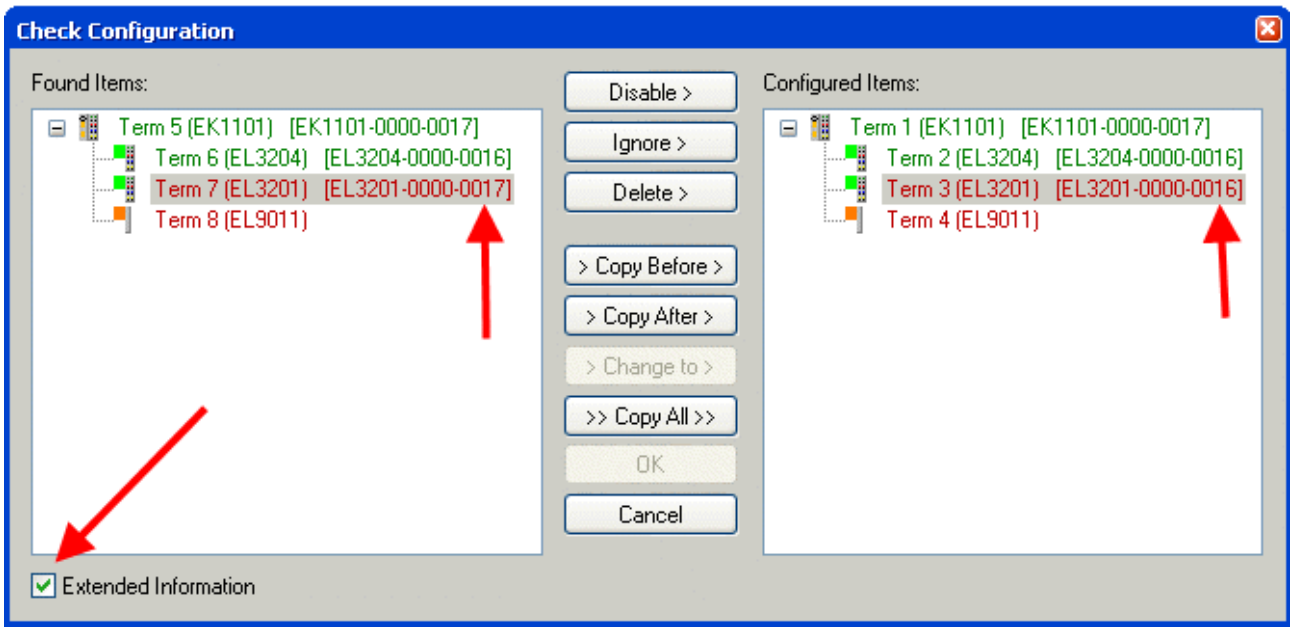


Abb. 337: Änderungsdialog

In diesem Beispiel in Abb. *Änderungsdialog*. wurde eine EL3201-0000-**0017** vorgefunden, während eine EL3201-0000-**0016** konfiguriert wurde. In diesem Fall bietet es sich an, mit dem *Copy Before*-Button die Konfiguration anzupassen. Die Checkbox *Extended Information* muss gesetzt werden, um die Revision angezeigt zu bekommen.

Änderung der Slave-Kennung ESI

Die ESI/EEPROM-Kennung kann unter TwinCAT wie folgt aktualisiert werden:

- Es muss eine einwandfreie EtherCAT-Kommunikation zum Slave hergestellt werden
- Der State des Slave ist unerheblich
- Rechtsklick auf den Slave in der Online-Anzeige führt zum Dialog *EEPROM Update*, Abb. *EEPROM Update*

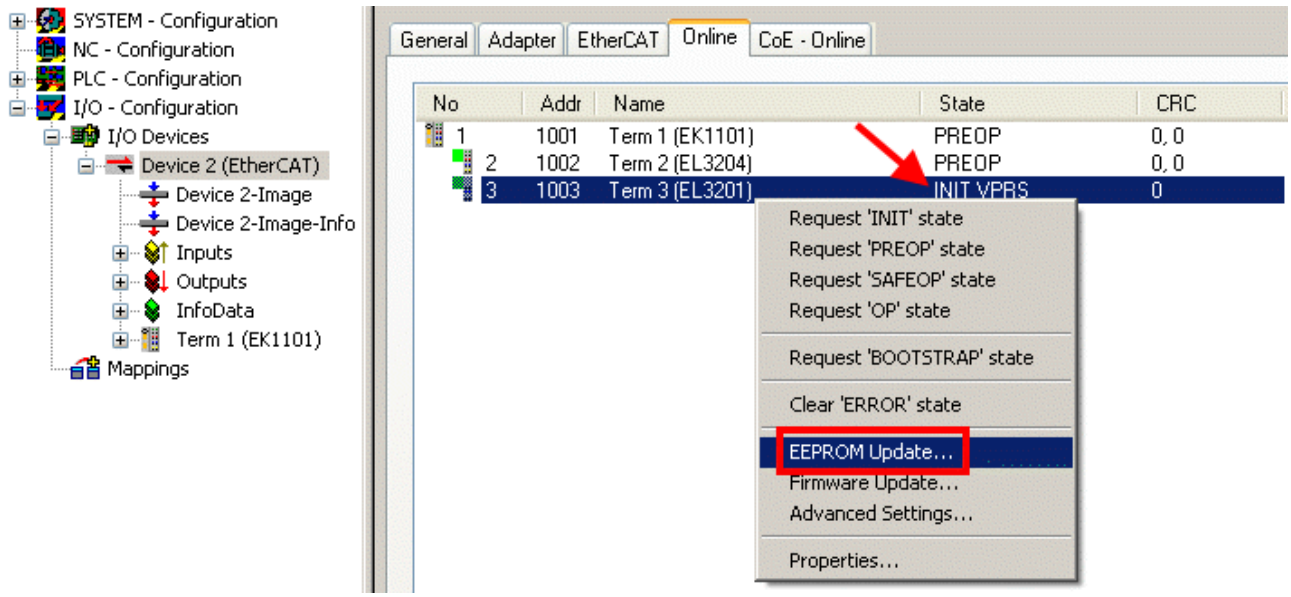


Abb. 338: EEPROM Update

Im folgenden Dialog wird die neue ESI-Beschreibung ausgewählt, s. Abb. *Auswahl des neuen ESI*. Die CheckBox *Show Hidden Devices* zeigt auch ältere, normalerweise ausgeblendete Ausgaben eines Slave.

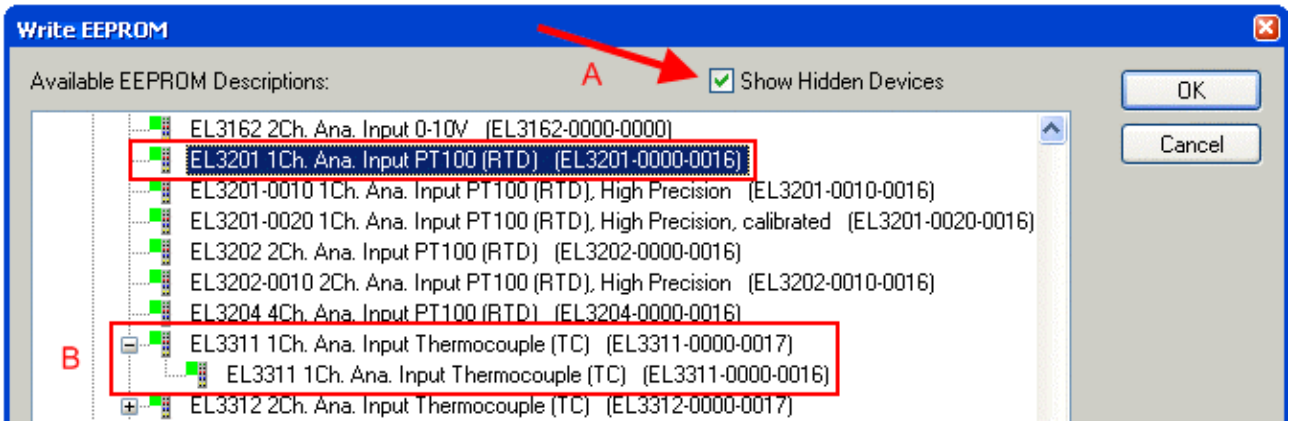


Abb. 339: Auswahl des neuen ESI

Ein Laufbalken im System Manager zeigt den Fortschritt - erst erfolgt das Schreiben, dann das Veryfiing.

Änderung erst nach Neustart wirksam

Die meisten EtherCAT-Geräte lesen eine geänderte ESI-Beschreibung umgehend bzw. nach dem Aufstarten aus dem INIT ein. Einige Kommunikationseinstellungen wie z. B. Distributed Clocks werden jedoch erst bei PowerOn gelesen. Deshalb ist ein kurzes Abschalten des EtherCAT Slave nötig, damit die Änderung wirksam wird.

8.3.2 Erläuterungen zur Firmware

Versionsbestimmung der Firmware

Versionsbestimmung mit dem System-Manager

Der TwinCAT System-Manager zeigt die Version der Controller-Firmware an, wenn der Slave online für den Master zugänglich ist. Klicken Sie hierzu auf die E-Bus-Klemme deren Controller-Firmware Sie überprüfen möchten (im Beispiel Klemme 2 (EL3204) und wählen Sie den Karteireiter *CoE-Online* (CAN over EtherCAT).

CoE-Online und Offline-CoE

Es existieren zwei CoE-Verzeichnisse:
• **online:** es wird im EtherCAT Slave vom Controller angeboten, wenn der EtherCAT Slave dies unterstützt. Dieses CoE-Verzeichnis kann nur bei angeschlossenem und betriebsbereitem Slave angezeigt werden.
• **offline**: in der EtherCAT Slave Information ESI/XML kann der Default-Inhalt des CoE enthalten sein. Dieses CoE-Verzeichnis kann nur angezeigt werden, wenn es in der ESI (z. B. „Beckhoff EL5xxx.xml“) enthalten ist.

Die Umschaltung zwischen beiden Ansichten kann über den Button *Advanced* vorgenommen werden.

In Abb. *Anzeige FW-Stand EL3204* wird der FW-Stand der markierten EL3204 in CoE-Eintrag 0x100A mit 03 angezeigt.

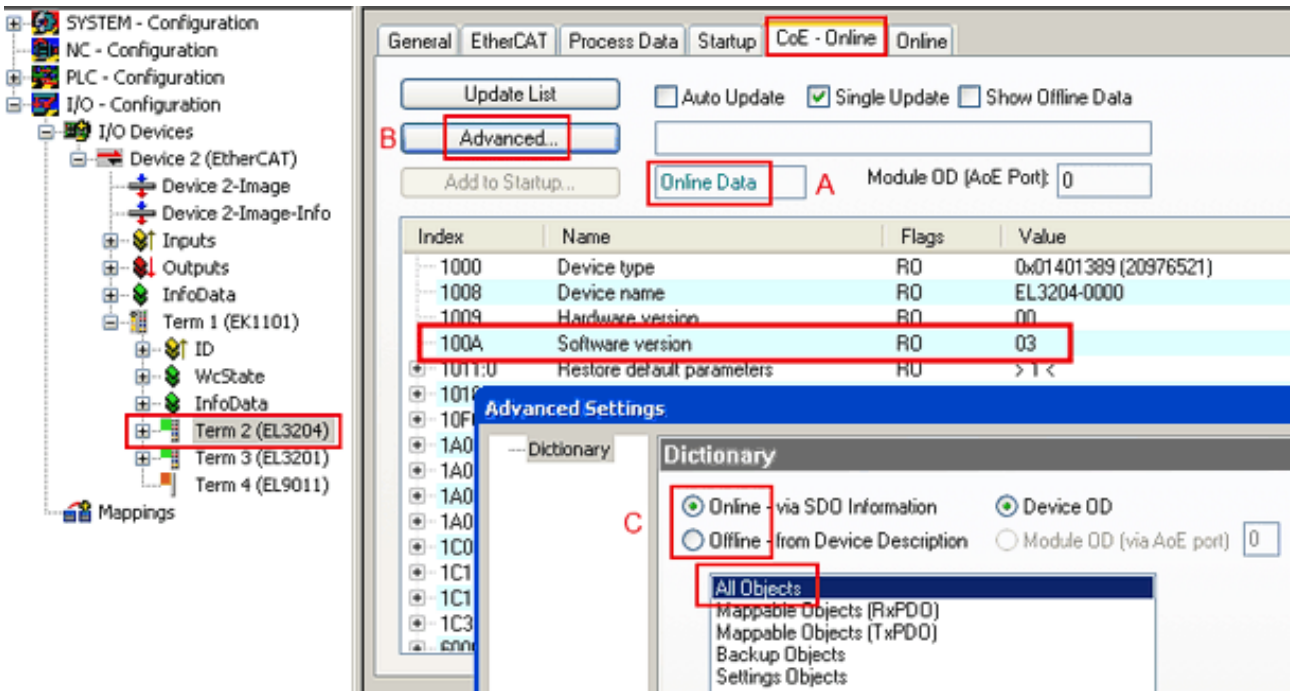


Abb. 340: Anzeige FW-Stand EL3204

TwinCAT 2.11 zeigt in (A) an, dass aktuell das Online-CoE-Verzeichnis angezeigt wird. Ist dies nicht der Fall, kann durch die erweiterten Einstellungen (B) durch *Online* und Doppelklick auf *All Objects* das Online-Verzeichnis geladen werden.

8.3.3 Update Controller-Firmware *.efw

CoE-Verzeichnis

Das Online-CoE-Verzeichnis wird vom Controller verwaltet und in einem eigenen EEPROM gespeichert. Es wird durch ein FW-Update im allgemeinen nicht verändert.

Um die Controller-Firmware eines Slave zu aktualisieren, wechseln Sie zum Karteireiter *Online*, s. Abb. *Firmware Update*.

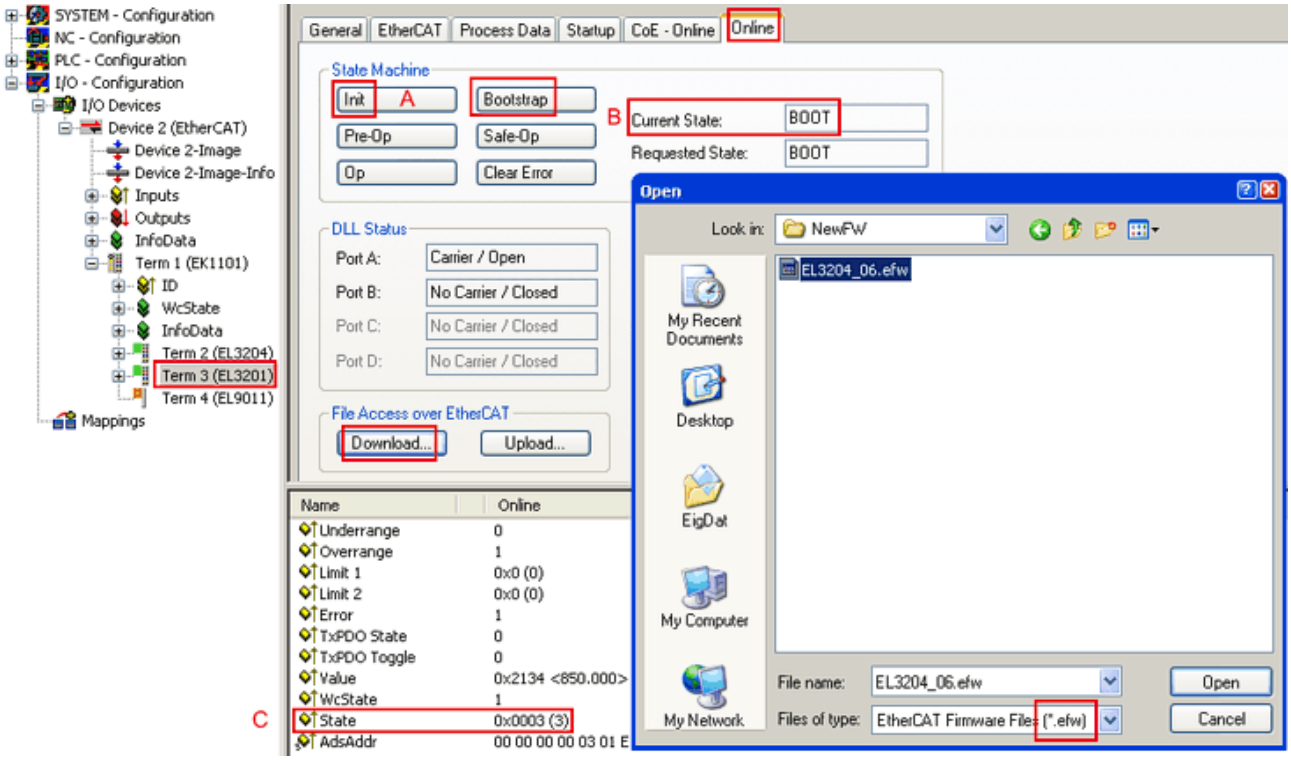
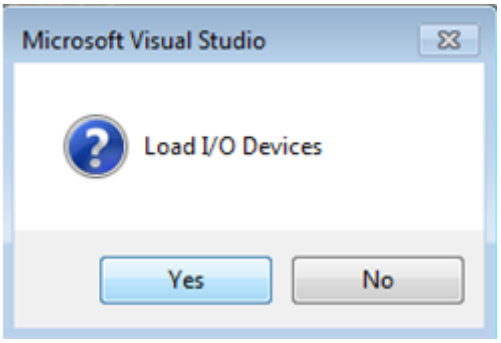
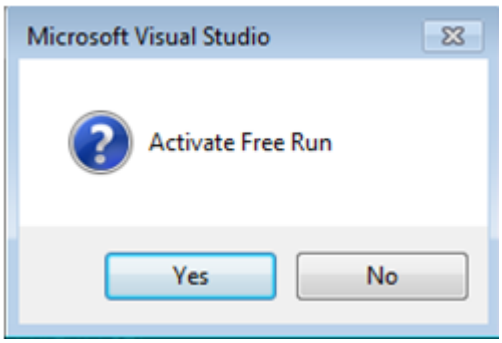


Abb. 341: Firmware Update

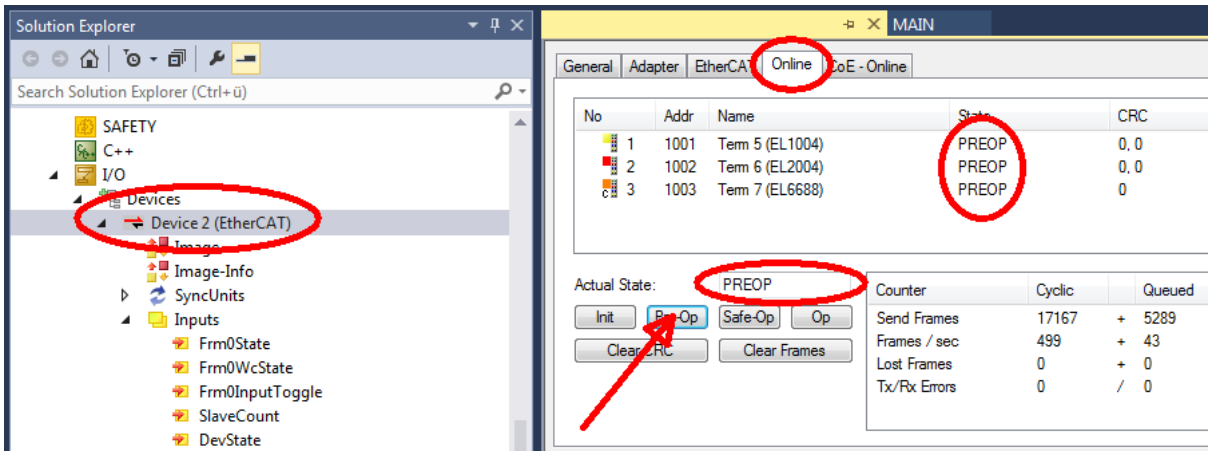
Es ist folgender Ablauf einzuhalten, wenn keine anderen Angaben z. B. durch den Beckhoff Support vorliegen. Gültig für TwinCAT 2 und 3 als EtherCAT Master.

- TwinCAT System in ConfigMode/FreeRun mit Zykluszeit >= 1ms schalten (default sind im ConfigMode 4 ms). Ein FW-Update während Echtzeitbetrieb ist nicht zu empfehlen.



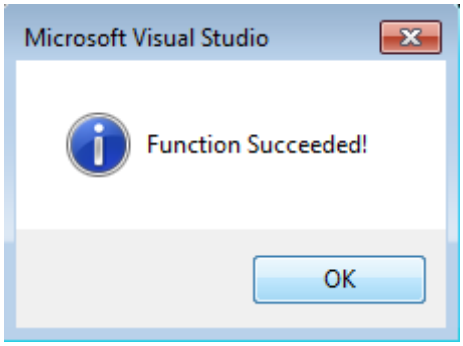


- EtherCAT Master in PreOP schalten



- Slave in INIT schalten (A)
- Slave in BOOTSTRAP schalten

- Kontrolle des aktuellen Status (B, C)
- Download der neuen *efw-Datei, abwarten bis beendet. Ein Passwort wird in der Regel nicht benötigt.



- Nach Beendigung des Download in INIT schalten, dann in PreOP
- Slave kurz stromlos schalten (nicht unter Spannung ziehen!)
- Im CoE 0x100A kontrollieren ob der FW-Stand korrekt übernommen wurde.

8.3.4 FPGA-Firmware *.rbf

Falls ein FPGA-Chip die EtherCAT-Kommunikation übernimmt, kann ggf. mit einer *.rbf-Datei ein Update durchgeführt werden.

- Controller-Firmware für die Aufbereitung der E/A-Signale
- FPGA-Firmware für die EtherCAT-Kommunikation (nur für Klemmen mit FPGA)

Die in der Seriennummer der Klemme enthaltene Firmware-Versionsnummer beinhaltet beide Firmware-Teile. Wenn auch nur eine dieser Firmware-Komponenten verändert wird, dann wird diese Versionsnummer fortgeschrieben.

Versionsbestimmung mit dem System-Manager

Der TwinCAT System-Manager zeigt die Version der FPGA-Firmware an. Klicken Sie hierzu auf die Ethernet-Karte Ihres EtherCAT-Stranges (im Beispiel Gerät 2) und wählen Sie den Karteireiter *Online*.

Die Spalte *Reg:0002* zeigt die Firmware-Version der einzelnen EtherCAT-Geräte in hexadezimaler und dezimaler Darstellung an.

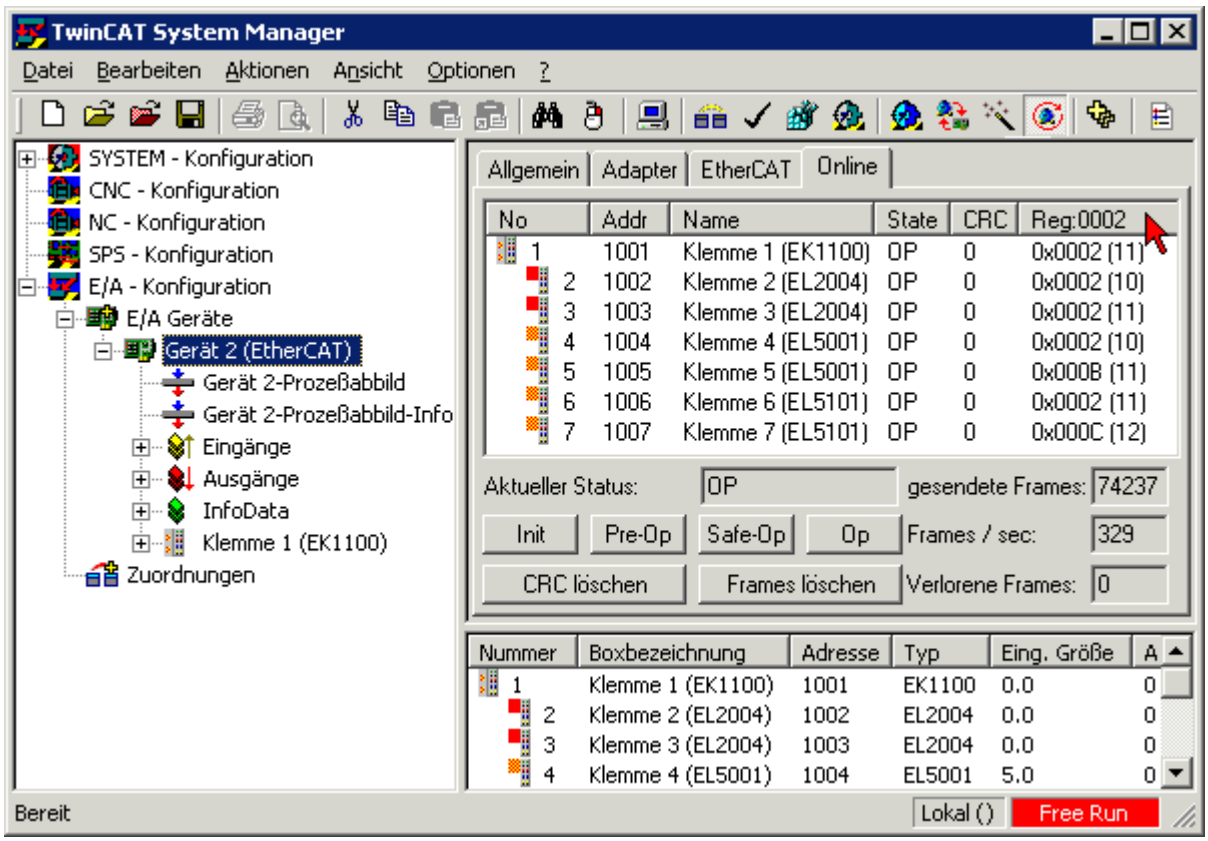


Abb. 342: Versionsbestimmung FPGA-Firmware

Falls die Spalte *Reg:0002* nicht angezeigt wird, klicken sie mit der rechten Maustaste auf den Tabellenkopf und wählen im erscheinenden Kontextmenü, den Menüpunkt *Properties*.

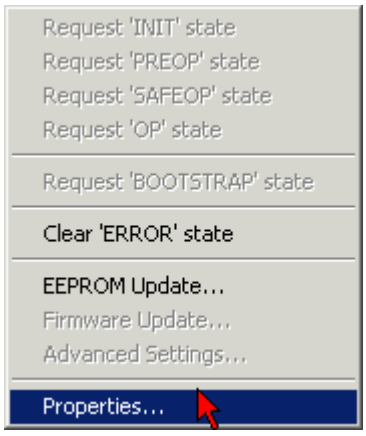


Abb. 343: Kontextmenu *Eigenschaften (Properties)*

In dem folgenden Dialog *Advanced Settings* können Sie festlegen, welche Spalten angezeigt werden sollen. Markieren Sie dort unter *Diagnose/***Online Anzeige** das Kontrollkästchen vor *'0002 ETxxxx Build'* um die Anzeige der FPGA-Firmware-Version zu aktivieren.

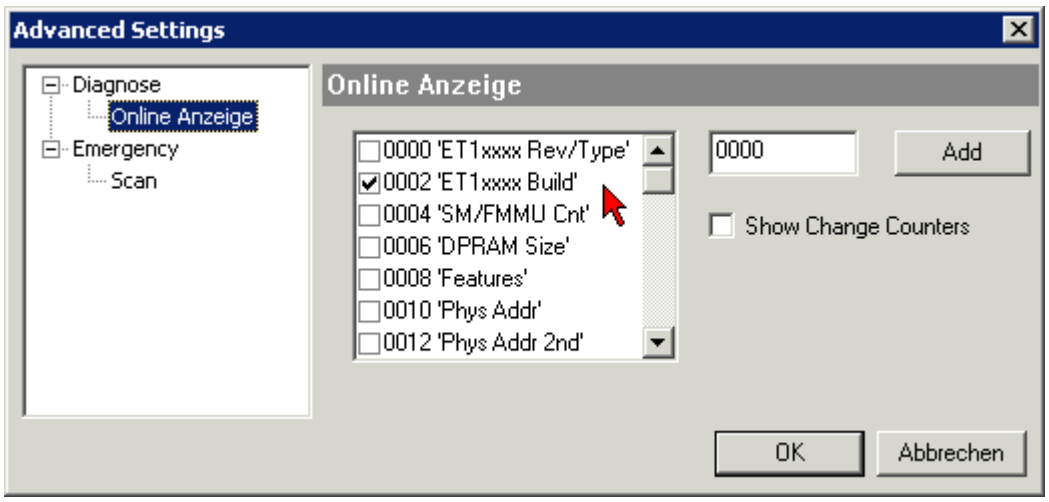


Abb. 344: Dialog *Advanced settings*

Update

Für das Update der FPGA-Firmware

- eines EtherCAT-Kopplers, muss auf diesem Koppler mindestens die FPGA-Firmware-Version 11 vorhanden sein.
- einer E-Bus-Klemme, muss auf dieser Klemme mindestens die FPGA-Firmware-Version 10 vorhanden sein.

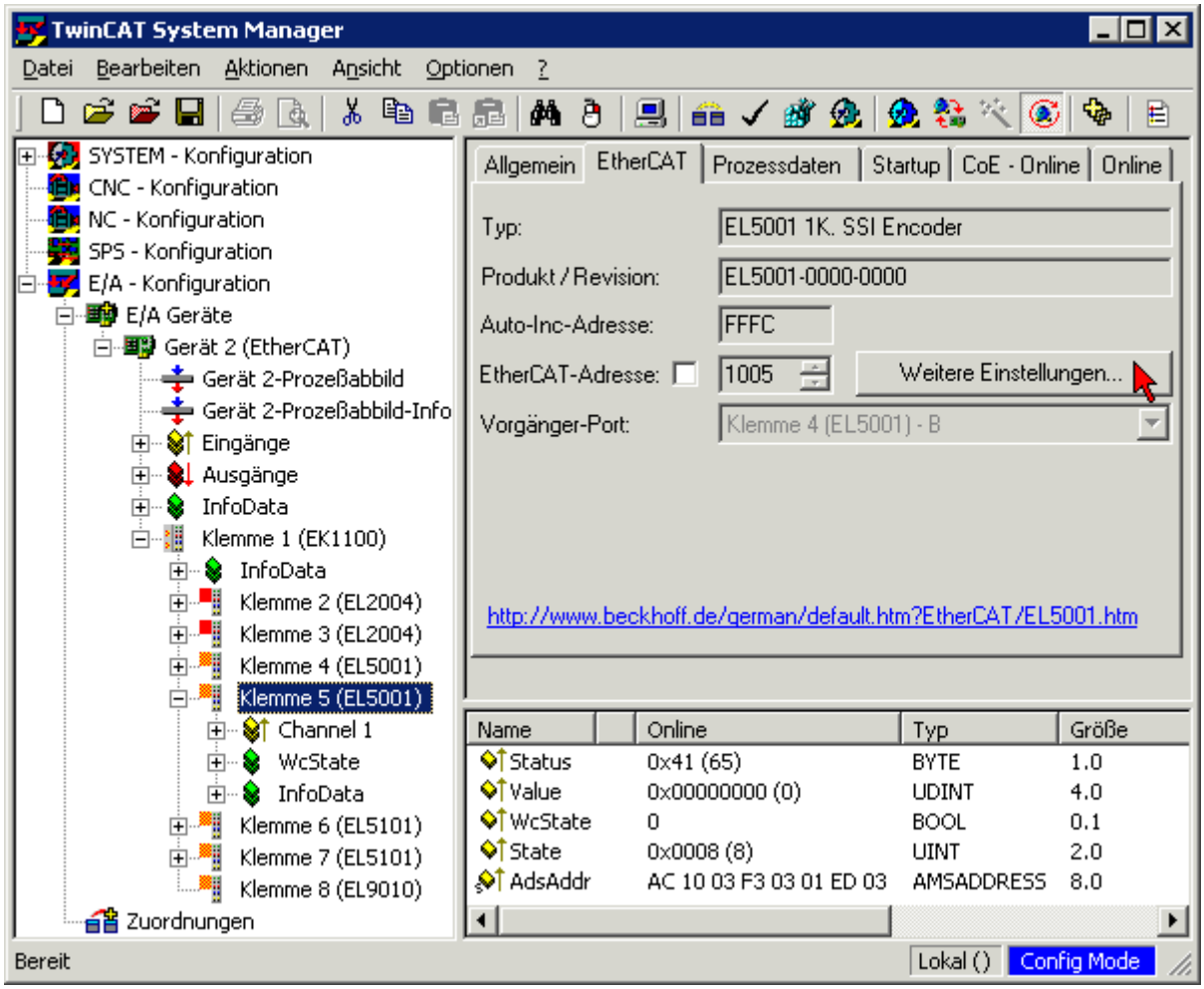
Ältere Firmware-Stände können nur vom Hersteller aktualisiert werden!

Update eines EtherCAT-Geräts

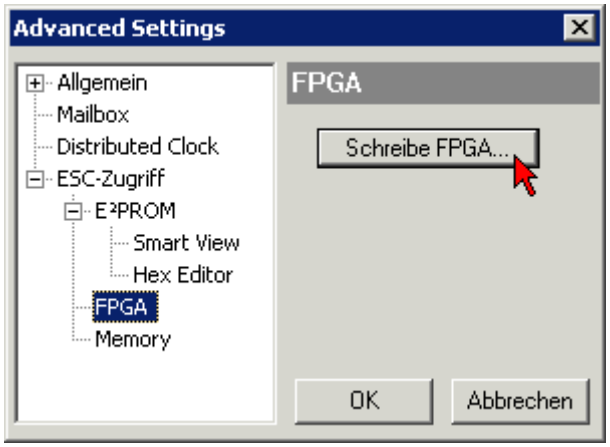
Es ist folgender Ablauf einzuhalten, wenn keine anderen Angaben z. B. durch den Beckhoff Support vorliegen:

- TwinCAT System in ConfigMode/FreeRun mit Zykluszeit >= 1 ms schalten (default sind im ConfigMode 4 ms). Ein FW-Update während Echtzeitbetrieb ist nicht zu empfehlen.

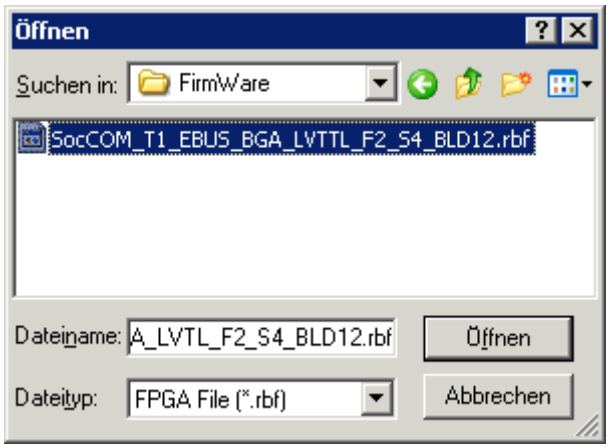
- Wählen Sie im TwinCAT System-Manager die Klemme an, deren FPGA-Firmware Sie aktualisieren möchten (im Beispiel: Klemme 5: EL5001) und klicken Sie auf dem Karteireiter *EtherCAT* auf die Schaltfläche *Weitere Einstellungen*:



- Im folgenden Dialog *Advanced Settings* klicken Sie im Menüpunkt *ESC-Zugriff/E²PROM*/FPGA auf die Schaltfläche *Schreibe FPGA*:



- Wählen Sie die Datei (*.rbf) mit der neuen FPGA-Firmware aus und übertragen Sie diese zum EtherCAT-Gerät:



- Abwarten bis zum Ende des Downloads
- Slave kurz stromlos schalten (nicht unter Spannung ziehen!). Um die neue FPGA-Firmware zu aktivieren ist ein Neustart (Aus- und Wiedereinschalten der Spannungsversorgung) des EtherCAT-Geräts erforderlich
- Kontrolle des neuen FPGA-Standes

HINWEIS

Beschädigung des Gerätes möglich!

Das Herunterladen der Firmware auf ein EtherCAT-Gerät dürfen Sie auf keinen Fall unterbrechen! Wenn Sie diesen Vorgang abbrechen, dabei die Versorgungsspannung ausschalten oder die Ethernet-Verbindung unterbrechen, kann das EtherCAT-Gerät nur vom Hersteller wieder in Betrieb genommen werden!

8.3.5 Gleichzeitiges Update mehrerer EtherCAT-Geräte

Die Firmware von mehreren Geräten kann gleichzeitig aktualisiert werden, ebenso wie die ESI-Beschreibung. Voraussetzung hierfür ist, dass für diese Geräte die gleiche Firmware-Datei/ESI gilt.

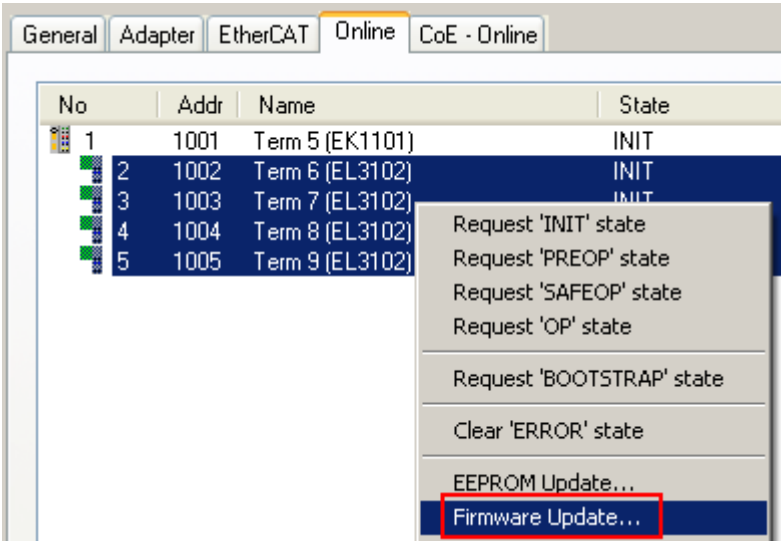


Abb. 345: Mehrfache Selektion und FW-Update

Wählen Sie dazu die betreffenden Slaves aus und führen Sie das Firmware-Update im BOOTSTRAP Modus wie o. a. aus.

9 FAQ

9.1 Windows Memory Dump

Sachverhalt

In seltenen Fällen kann es sinnvoll sein, den internen Software-Zustand des IPC-Systems vor einem Ausfall zu kennen. Zu diesem Zweck kann das Windows Betriebssystem im Moment des Neustartversuchs ein aktuelles Abbild des Arbeitsspeichers auf die Festplatte schreiben, einen so genannten Memory Dump.

Lösung

Das korrekte Schreiben des Memory Dumps muss vor Eintritt des Restarts eingestellt werden.

Windows Betriebssystem

Das Schreiben von Memory Dumps ist nur unter Windows NT/XP/XPe/Vista/7/WES möglich, nicht in den Systemen WEC/CE.

Auslagerungsdatei

Ggf. ist für das erfolgreiche Schreiben eines Dumps die vorherige Aktivierung der Windows Auslagerungsdatei erforderlich (s.u.). Diese Auslagerungsdatei auf der Festplatte erscheint dem Betriebssystem als zusätzlicher Arbeitsspeicher und wird dann auch regelmäßig genutzt. Dies verursacht vermehrte Festplattezugriffe die bei Nutzung einer CompactFlash (CF) Karte als Betriebssystemfestplatte ggf. zu Lasten der Lebensdauer gehen können. Es wird generell die Überwachung der CF Karten Nutzung mit entsprechenden MDP-Diagnosefunktionen aus der Applikation heraus empfohlen (z. B. FB_SiliconDrive_Read). Die Auslagerungsdatei ist dann nur zu Diagnosezwecken vorübergehend zu aktivieren.

Schreibschutz

Ggf. ist für das erfolgreiche Schreiben ein aktiver Schreibschutz der Festplatte aufzuheben. Solche Schreibschutzfilter können sein:

- EWF (Enhanced Write Filter)
- FBWF (File Based Write Filter

Es gibt verschiedene Arten von Memory Dumps, je nach Bereich und Größe des erfassten Speicherbereichs. Die zugehörigen Einstellungen werden deshalb auch in der Windows-Systemverwaltung vorgenommen.

- Small memory dump (64 KB)
- Kernel
- Complete

Das nach einem Restart des Systems auf der Festplatte als Datei vorliegende Speicherabbild kann von Beckhoff ggf. analysiert werden, wenn der Restart z. B. von TwinCAT verursacht wurde.

Die bestmögliche Analyse erlaubt ein "Complete Dump", der allerdings einige MByte betragen kann. Es ist bei den Einstellungen zu beachten

- dass auf der Festplatte/CF-Karte genug Speicherplatz für den Dump vorhanden ist.
- dass der eingestellte Datei-Pfad gültig ist.

Für den Minimal-Dump ist folgende Einstellung vorzunehmen:

1. Im ersten Schritt wird die Auslagerungsdatei aktiviert.
   Windows System Properties --> Advanced --> Performance Settings

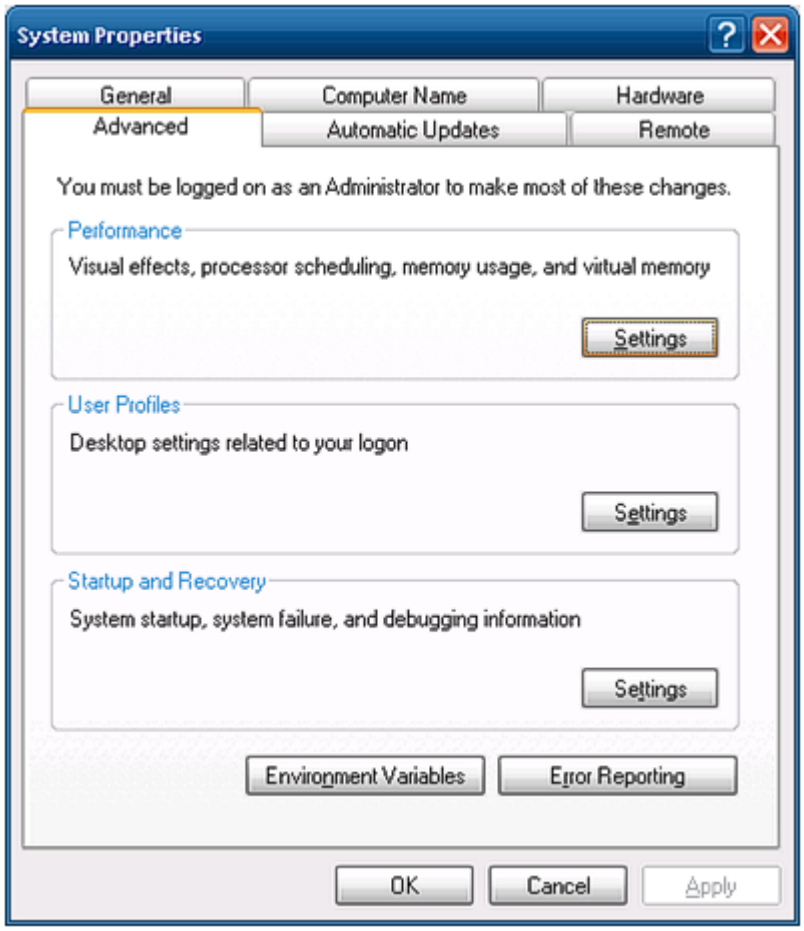


Abb. 346: System Properties

2. Performance Options --> Virtual Memory --> Change

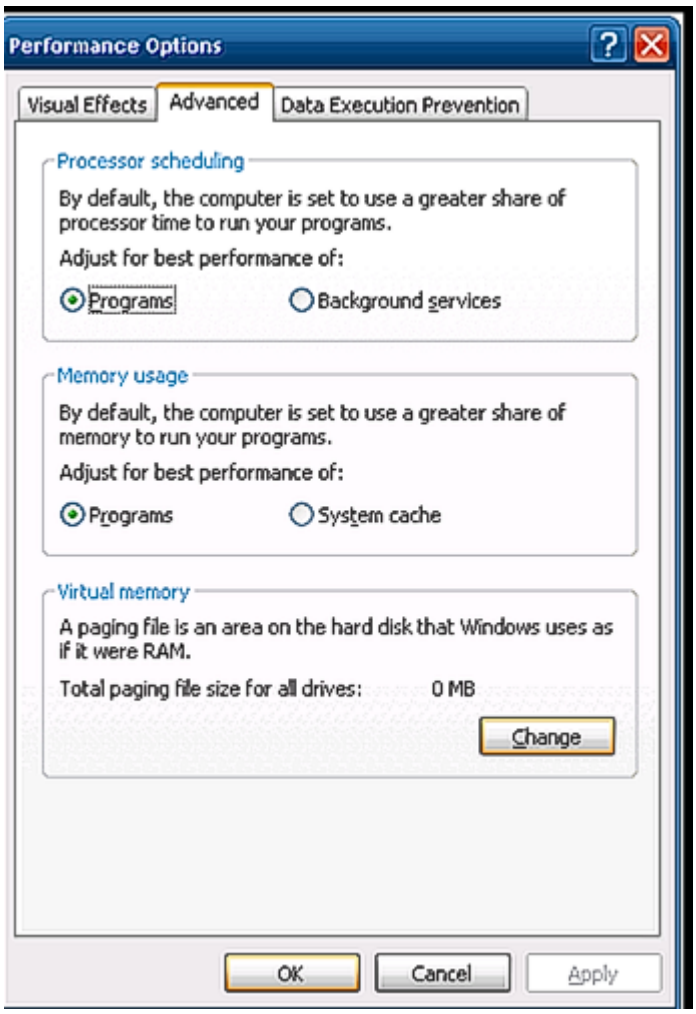


Abb. 347: Einstellung virtueller Speicher

3. Setzen der Auslagerungsdatei

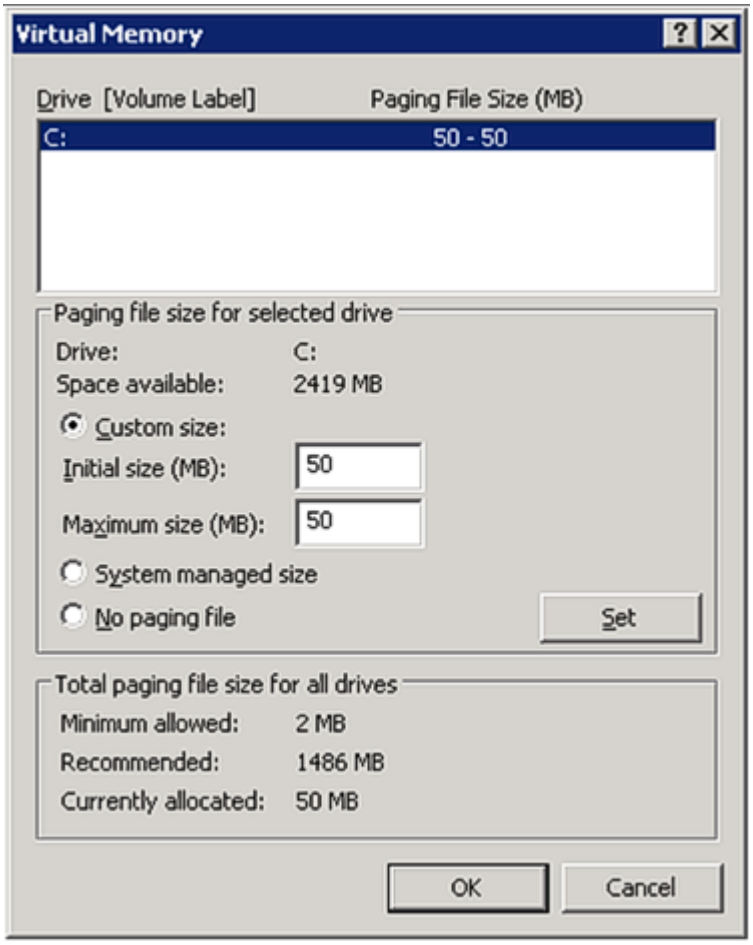


Abb. 348: Einstellung Auslagerungsdatei

"Initial Size" und "Maximum Size" auf mind. 50 MByte setzen und mit SET bestätigen. Ein alleiniger Druck auf OK ist nicht ausreichend!
Mit OK bestätigen.

1. Im zweiten Schritt wird in den Windows Systemeinstellungen (Abb. *System Properties*) das Schreiben des Dumps aktiviert.
   --> Startup and Recovery
   --> Auswahl "Small memory dump" unter "Write debugging information"

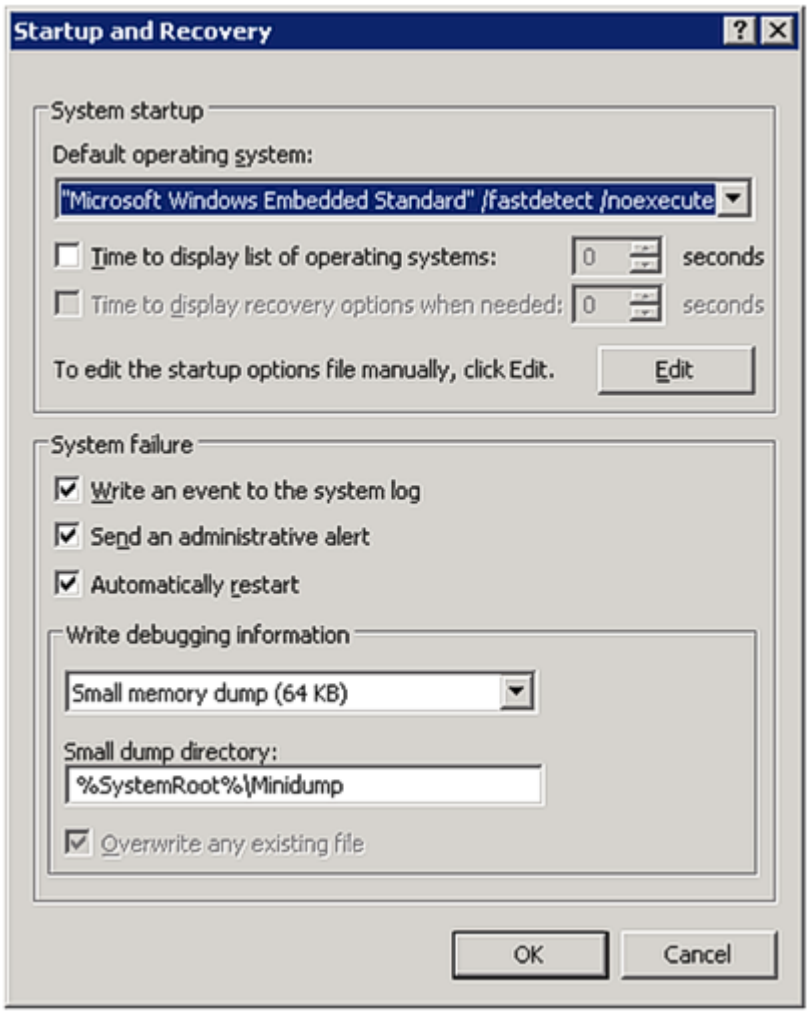


Abb. 349: Einstellung Dump

2. Ein Neustart des IPC ist nun erforderlich.

Nach dem Restart mit erfolgreichem Dump-Schreiben ist im o.a. Pfad z. B. eine Dump-Datei mit aktuellem Zeitstempel zu finden.
Complete-Dumps liegen je nach Pfadeinstellung z. B. als "Memory.dmp" unter C:\Windows\, Small-Dumps als "Mini.dmp" im Unterordner C:Windows\Minidump\.

9.2 MessageBox im Zielsystem

Sachverhalt

Auf dem Zielsystem (mit Betriebssystem MS XP/XPe/7/WES u.ä.) werden im Betrieb TwinCAT MessageBoxen ("PopUp Fenster") angezeigt, die Bestätigung erfordern z. B. durch Druck auf "OK". Das Zielsystem befindet sich im produktiven Betrieb ohne HMI (Visualisierung, Monitor, Tastatur, Maus), die MessageBox kann also nicht bestätigt werden bzw. es ist kein Maschinenführer vor Ort. Dann werden auf dem über ADS verbundenen Programmiergerät/TwinCAT keine Logger-Ausgaben mehr angezeigt.

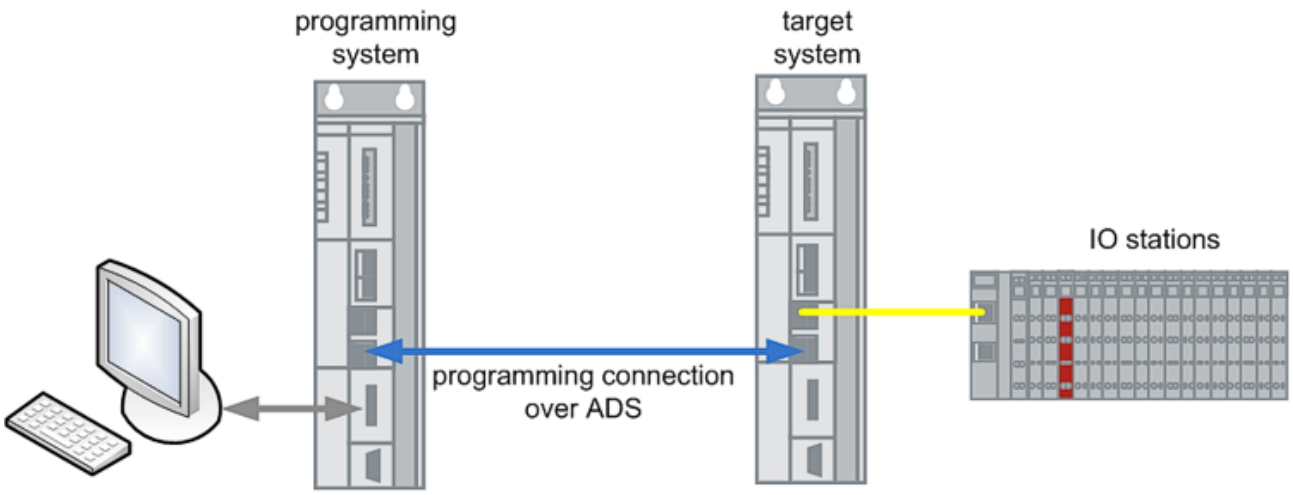


Abb. 350: Topologie Programmiersystem - Zielsystem

Lösung

Es ist auf dem Zielsystem ein RegistryKey zu setzen, der die TwinCAT-Hinweisfenster unterdrückt.

RegistryKey (https://infosys.beckhoff.com/content/1031/ethercatsystem/Resources/zip/2469160331.zip)

Es wird *nicht* empfohlen, generell diesen Key zu setzen, denn dann erscheinen keine TwinCAT-Hinweisfenster mehr.

10 Anhang

10.1 UL-Hinweise

⚠ VORSICHT
Application The modules are intended for use with Beckhoff's UL Listed EtherCAT System only.

⚠ VORSICHT
Examination For cULus examination, the Beckhoff I/O System has only been investigated for risk of fire and electrical shock (in accordance with UL508 and CSA C22.2 No. 142).

⚠ VORSICHT
For devices with Ethernet connectors Not for connection to telecommunication circuits.

Grundlagen

UL-Zertifikation nach UL508. Solcherart zertifizierte Geräte sind gekennzeichnet durch das Zeichen:

10.2 Training und Schulung

Beckhoff

Beckhoff bietet Schulungen u.a. an zu den Themen

- TwinCAT Training für Anwender
- EtherCAT Training für Anwender (TR8020)
  Hier werden Inbetriebnahme und Diagnose von EtherCAT Geräten angesprochen.
- Schulungen für Entwickler auf Basis der Beckhoff Entwicklungsprodukte für EtherCAT Implementation

Kontakt: http://www.beckhoff.de/training

ETG (www.ethercat.org)

www.ethercat.org

Die ETG bietet z. B. an

- Developer Basics
- kundenspezifische Schulungen

10.3 Support und Service

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

Beckhoff Niederlassungen und Vertretungen

Wenden Sie sich bitte an Ihre Beckhoff Niederlassung oder Ihre Vertretung für den lokalen Support und Service zu Beckhoff Produkten!

Die Adressen der weltweiten Beckhoff Niederlassungen und Vertretungen entnehmen Sie bitte unseren Internetseiten: https://www.beckhoff.de

Dort finden Sie auch weitere Dokumentationen zu Beckhoff Komponenten.

Beckhoff Support

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

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

Hotline:	+49(0)5246 963 157
Fax:	+49(0)5246 963 9157
E-Mail:	support@beckhoff.com

Beckhoff Service

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

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

Hotline:	+49(0)5246 963 460
Fax:	+49(0)5246 963 479
E-Mail:	service@beckhoff.com

Beckhoff Firmenzentrale

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20
33415 Verl
Deutschland

Telefon:	+49(0)5246 963 0
Fax:	+49(0)5246 963 198
E-Mail:	info@beckhoff.com
Internet:	https://www.beckhoff.de

Mehr Informationen:
infosys.beckhoff.com/content/1031/ethercatsystem

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