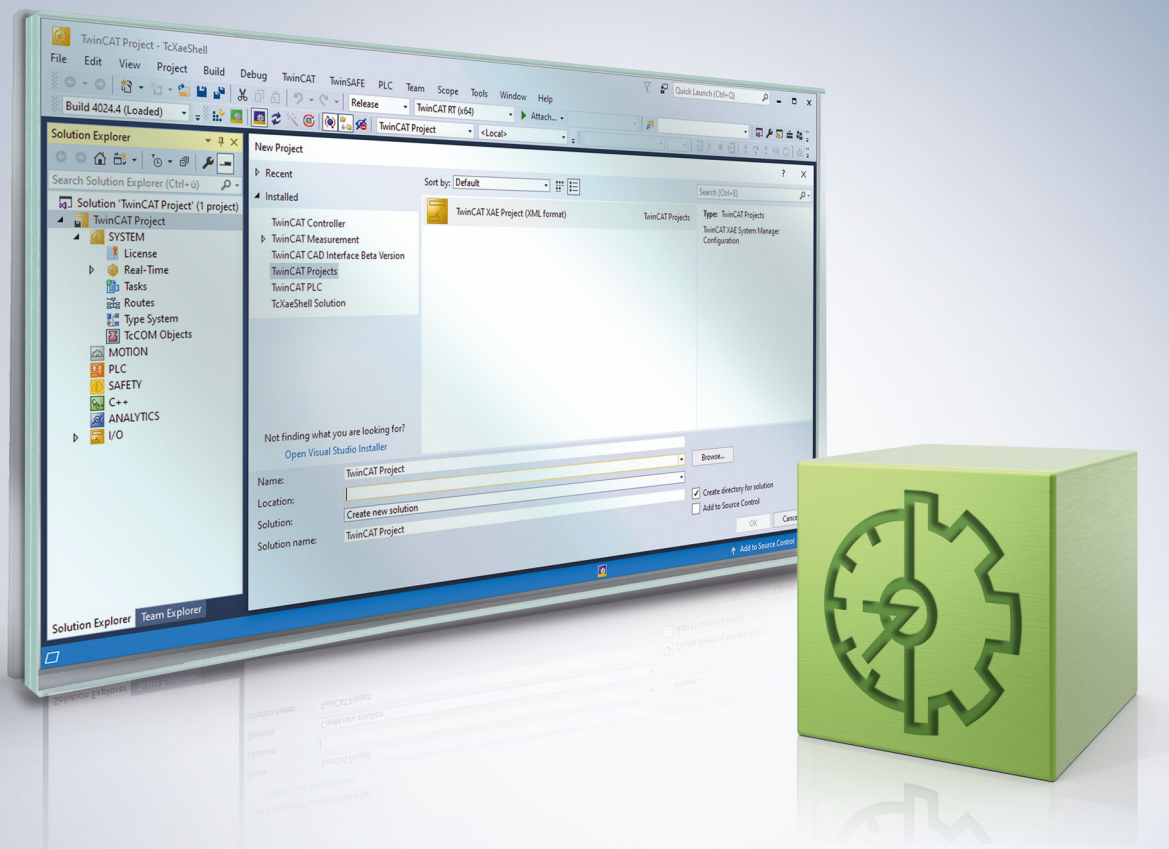
BECKHOFF New Automation Technology

Handbuch | DE

TE1000

TwinCAT 3 | EAP



23.06.2025 | Version: 1.7.1

Inhaltsverzeichnis

1 Vorwort

1.1 Hinweise zur Dokumentation

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

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

Disclaimer

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

Marken

Beckhoff®, ATRO®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar® und XTS® 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 Kennzeichnungen führen.

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

Copyright

Fremdmarken

In dieser Dokumentation können Marken Dritter verwendet werden. Die zugehörigen Markenvermerke finden Sie unter: https://www.beckhoff.com/trademarks.

1.2 Zu Ihrer Sicherheit

Sicherheitsbestimmungen

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

Haftungsausschluss

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

Qualifikation des Personals

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

Signalwörter

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

Warnungen vor Personenschäden

⚠ GEFAHR

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

⚠ WARNUNG

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

⚠ VORSICHT

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

Warnung vor Umwelt- oder Sachschäden

HINWEIS

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

Information zum Umgang mit dem Produkt

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

1.3 Hinweise zur Informationssicherheit

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

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

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

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

2 Produktbeschreibung

Das EtherCAT Automation Protocol (EAP) Gerät ermöglicht den zyklischen, hochdeterministischen Austausch beliebiger Variablen zwischen PCs, die per Ethernet verbunden sind. Die Kommunikation zwischen EAP Geräten erfolgt dabei nach dem Publisher Subscriber Prinzip und ist von der EtherCAT Technology Group (ETG) spezifiziert worden (ETG 1005 – siehe Webpage www.ethercat.org).

Zur hochdeterministischen Kommunikation muss für das TwinCAT EAP Gerät der Realtime Ethernet Treiber für TwinCAT installiert sein.

Vergleich mit TwinCAT 2 Netzwerkvariablen

Das TwinCAT EAP Gerät basiert auf den aus TwinCAT 2 bekannten Netzwerkvariablen (NWV) und beinhaltet einige Erweiterungen. Unter anderem erweitert das EAP Telegramm auch das NWV Telegramm geringfügig. Diese Erweiterung betrifft jedoch nur den Inhalt des Telegramms. Die Struktur des EAP Telegramms ist identisch zum NVW Telegramm geblieben. Aus diesem Grund sind Netzwerkvariablen kompatibel zum EtherCAT Automation Protocol und umgekehrt. Weitere Details zu einer EAP Kommunikation zwischen TwinCAT 2 und 3 sind in Kapitel EAP zwischen TwinCAT 2 und 3 [▶ 51] zu finden.

Voraussetzungen

Der vollständige Funktionsumfang des TwinCAT EAP Gerätes, wie er in dieser Dokumentation beschrieben wird, steht ab der Version TwinCAT 3.1 (Build 4018.13) oder höher zur Verfügung.

2.1 Grundlagen

Das TwinCAT EAP-Gerät ermöglicht die Übertragung von Daten aus beliebigen Variablen von einer TwinCAT Steuerung A zu einer TwinCAT Steuerung B via Netzwerk. Diese Variablen dienen typischerweise dazu, den Verarbeitungsprozess innerhalb einer Maschine zu steuern. Sie werden daher auch Prozessvariablen (*PV*) genannt. Das Senden und Empfangen kann für ein TwinCAT EAP-Gerät über einen normalen Netzwerkadapter erfolgen, der von dem TwinCAT Realtime Ethernet Treiber unterstützt wird.

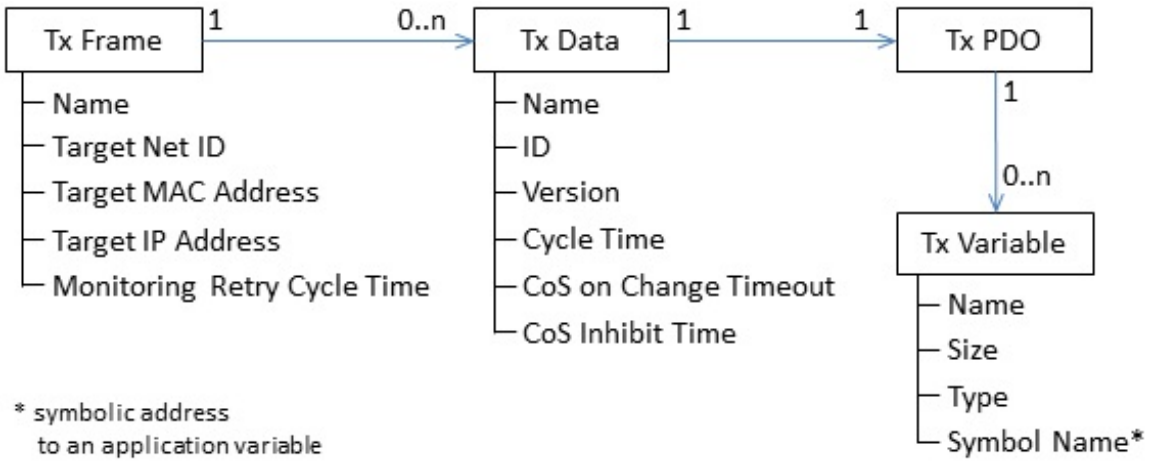
Die Kommunikation zwischen EAP-Geräten erfolgt nach dem Publisher Subscriber Prinzip, bei dem die Sender, Publisher genannt, ihre Nachrichten an alle oder mehrere Netzwerkteilnehmer schicken, wobei ein Publisher grundsätzlich nicht weiß, wer seine Empfänger sind bzw. ob es überhaupt einen Empfänger gibt. Auf der anderen Seite gibt es Empfänger, Subscriber genannt, die sich für bestimmte Nachrichten interessieren und diese empfangen, ohne zu wissen, von welchem Publisher diese kommen bzw. ob es überhaupt einen Publisher gibt, der eine solche Nachricht verschickt.

Aufbau eines EAP-Publishers

Ein EAP-Publisher setzt sich aus einer Summe geschachtelter Elemente zusammen, so wie in der folgenden Abbildung grafisch veranschaulicht. Das Basiselement auf der untersten Ebene ist eine *TxVariable*. Sie definiert eine Ausgangsvariable, die mit einer Prozessvariablen verknüpft wird und beinhaltet ein paar weitere Eigenschaften wie z. B. einen Datentyp. Der Datentyp darf frei gewählt werden; es darf also auch ein komplexer Datentyp sein, der mehrere hundert Bytes groß ist. Es sollte gewährleistet werden, dass die maximale Größe eines EAP-Frames nicht überschritten wird (die Größe eins EAP-Frames entspricht der eines Standard Ethernet Frames). Bei Überschreiten der Größe des Standard Ethernet Frames werden mehrere Pakete versendet, welche beim Empfänger auch in unterschiedlicher Reihenfolge eintreffen können.

In TwinCAT 3.1 Build 4024 wird das Aufteilen auf mehrere Pakete nicht unterstützt.
Beim Engineering erscheint eine Fehlermeldung.

Die Prozessvariable liefert während des Betriebs die Werte, die der Publisher senden soll.



Eine Ebene darüber werden in den *TxPDO* Elementen (TxPDO = TxProcessDataObjects) *TxVariablen* referenziert. Ein *TxPDO* kann mehrere *TxVariablen* referenzieren und fasst sie so zu einem Objekt zusammen. Das *TxPDO* definiert dann eine geordnete Menge von *TxVariablen*. Für ein *TxPDO* muss ebenfalls gewährleistet sein, dass die maximale Größe eines EAP-Frames nicht überschritten wird.

Noch eine Ebene höher befindet sich das *TxData* Element (*TxProcessData* = *TxPD*), welches eine *Publisher Variable* repräsentiert und beim EAP als Kommunikationseinheit des Publishers begriffen wird. Das *TxData* referenziert ein bestimmtes *TxPDO* und definiert einige Eigenschaften, wie z.B. die *ID* der *Publisher Variablen*, deren *Version* und den Zyklustakt, mit dem die *Publisher Variable* überhaupt erst versendet wird. Mithilfe dieser Eigenschaften definiert die *Publisher Variable* ein Objekt auf der Senderseite, zu der eine passende Subscriber Variable auf der Empfängerseite definiert werden muss, so dass ein erfolgreicher Datenaustausch stattfinden kann.

Die Übertragung der Daten erfolgt netzwerkbasiert über das Ethernet Protokoll oder über UDP/IP. Einem *TxFrame* wird dann entsprechend eine Liste von TxData zugeordnet, die an die gleiche Zieladresse gesendet werden sollen. Ein *TxFrame* ist dabei auf eine maximale Datenlänge pro Datenpaket beschränkt. Für das Versenden einer *Publisher Variable* müssen dann wenigstens die folgenden Eigenschaften definiert sein:

Zieladresse:
Als Zieladresse wird in der Regel eine Multicast Adresse eingetragen, so dass die *Publisher Variable* automatisch an eine Gruppe von Empfängern gesendet wird. Es kann allerdings auch die Adresse eines einzelnen Empfängers eingetragen werden.

ID:
Für jede *Publisher Variable* wird eine Nummer definiert, die eindeutig im gesamten Netzwerk sein muss und anhand derer die *Publisher Variable* von einem *Subscriber* identifiziert werden kann.

Zyklustakt:
Anhand des Zyklustaktes wird festgelegt, in welchem zeitlichen Rhythmus die *Publisher Variable* versendet wird. Beim EAP sind im Allgemeinen Zykluszeiten von wenigen Millisekunden bis zu mehreren 100 Millisekunden üblich.

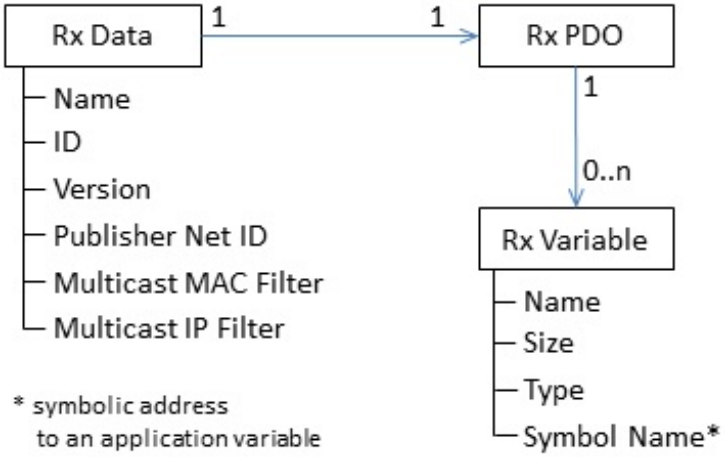
Verknüpfung zu einer Prozessvariablen:
Zu guter Letzt wird noch eine Verknüpfung der *Publisher Variablen* mit einer *Prozessvariablen* benötigt, damit auch tatsächlich Prozessdaten mit Hilfe der *Publisher Variable* verschickt werden. Andernfalls bliebe der Wert der *Publisher Variable* stets null.

Aufbau eines EAP-Subscribers

Die Struktur eines EAP-Subscribers ist analog zu der eines Publishers und wird in der folgenden Abbildung grafisch veranschaulicht. Das Basiselement auf der untersten Ebene eines Subscribers wird dann allerdings *RxVariable* genannt. Die RxVariable definiert eine Eingangsvariable, die ebenfalls mit einer Prozessvariablen verknüpft wird und ein paar Eigenschaften beinhaltet wie z.B. den Datentyp. Die Prozessvariable erhält während des Betriebs die Werte, die der Subscriber empfängt.

Die Elemente der darüber liegenden Ebene werden dann entsprechend *RxPDOs* (RxProcessDataObjects) genannt, die jeweils eine geordnete Menge von *RxVariablen* definieren.

Noch eine Ebene höher befindet sich das *RxData* Element (*RxProcessData* = *RxPD*), welches eine *Subscriber Variable* repräsentiert und beim EAP als Kommunikationseinheit des Subscribers begriffen wird. Das *RxData* referenziert ein bestimmtes *RxPDO* und definiert u.a. die notwendigen Eigenschaften (ID und Version), die mit der *Publisher Variablen* übereinstimmen müssen, die empfangen werden soll. Für einen erfolgreichen Datenaustausch müssen die Datentypen der referenzierten *RxVariable* und deren Ordnung im *RxPDO* identisch zum *TxPDO* der *Publisher Variable* sein. Die *Subscriber Variable* definiert somit ein Objekt auf der Empfangsseite, zu der eine passende *Publisher Variable* auf der Senderseite definiert werden muss, so dass ein Datenaustausch stattfindet.



Dem Design des EtherCAT Automation Protocols nach ist es für einen Subscriber unerheblich, von welchem Absender die empfangenen Daten stammen. Vor allem ist es unerheblich, welche *Publisher Variablen* innerhalb eines Frames versendet werden. Aus diesem Grund gibt es beim Subscriber auch kein Frame Element oder ähnliches, mit Hilfe dessen bestimmte *Subscriber Variablen* zu einer Einheit zusammengefasst und somit nur en bloc empfangen würden. Nichtsdestotrotz bietet das *RxData* die Möglichkeit eine *AMS NetID* als Filteradresse zu definieren, wenn ein Subscriber nur die *Publisher Variablen* eines bestimmten Absenders empfangen soll. Für eine *Subscriber Variable* müssen dann wenigstens die folgenden Eigenschaften definiert sein:

ID:
Die *ID* bei einer *Subscriber Variable* definiert, welche *Publisher Variable* diese empfangen soll. Die *ID* ist eine Nummer, die nach Möglichkeit eindeutig für jede *Publisher Variable* im gesamten Netzwerk ist und anhand derer die *Publisher Variable* beim Empfang identifiziert wird.

Verknüpfung zu einer Prozessvariablen:
Letztlich ist auch die Verwendung einer *Subscriber Variablen* erst dann sinnvoll, wenn diese mit einer Prozessvariablen verknüpft ist. Erst dann werden die empfangenen Daten auch tatsächlich von der Prozessvariablen übernommen und bei der Steuerung der Maschine berücksichtigt.

Außerdem muss sichergestellt sein, dass die Datenlänge der *Subscriber Variable* identisch zur Datenlänge der *Publisher Variable* ist. Andernfalls wird die empfangene *Publisher Variable* verworfen.

2.1.1 Kommunikationsmethoden

Das TwinCAT EAP-Gerät unterstützt zwei Kommunikationsarten. Zum einen die zyklische Prozessdatenkommunikation (EtherCAT Type 4) und zum anderen die azyklische EtherCAT Mailbox-Kommunikation (EtherCAT Type 5). Das TwinCAT EAP-Gerät unterstützt bei der Mailbox-Kommunikation nur das AoE-Protokoll (AoE – ADS over EtherCAT). Die Spezifikation des AoE-Protokolls wird in den EtherCAT Enhancements (ETG 1020) beschrieben. Bei der Prozessdatenkommunikation wird zwischen zwei Kommunikationsmodi unterschieden:

Zum einen den **Pushed Data Exchange** Modus, bei dem ein EAP-Sender zyklisch oder wegen einer Statusänderung seine Prozessinformationen in das Netz sendet und ein EAP-Empfänger entsprechend diese Prozessinformationen erwartet und empfängt. Dieser Modus entspricht dem Publisher Subscriber Prinzip der Netzwerkvariablen (NWV) von TwinCAT 2.
Zum anderen den **Polled Data Exchange** Modus, bei dem ein EAP-Client ein Anfragetelegramm mit seinen Prozessinformationen an einen EAP-Server sendet und dieser wiederum seine Prozessinformationen in einem Antworttelegramm an den EAP-Client zurückschickt.

Des Weiteren unterstützt das TwinCAT EAP-Gerät sowohl unterschiedliche Verbindungsarten als auch unterschiedliche Adressierungsarten bei der Prozessdatenkommunikation. Die unterstützen Verbindungsarten sind:

- **Unicast**: Die EAP-Nachricht wird von einem Endpunkt zu einem anderen Endpunkt geschickt, mit anderen Worten: Die Nachricht ist an genau einen PC adressiert.
- **Multicast**: Die EAP-Nachricht wird von einem Endpunkt zu mehreren anderen Endpunkten geschickt, mit anderen Worten: Die Nachricht ist an eine Gruppe von PCs adressiert.
- **Broadcast**: Die EAP-Nachricht wird von einem Endpunkt zu allen erreichbaren Endpunkten geschickt, mit anderen Worten: Die Nachricht ist an alle Teilnehmer adressiert.

Die Adressierung kann per MAC-Adresse, per AMS NetID oder per IP-Adresse erfolgen. Je nach Konstellation von Verbindungsart und Adressierungsart wird ein bestimmtes Vermittlungsprotokoll bei der EAP-Prozessdatenkommunikation wirksam. Die genaue Zuordnung ist in der folgenden Tabelle festgehalten.

Vermittlungsprotokolle

Adressierungsart Verbindungsart	MAC-Adresse	AMS NetID	IP-Adresse
Unicast	Ethernet Protokoll	Ethernet Protokoll	UDP/IP
Multicast	Ethernet Protokoll	*Nicht möglich*	UDP/IP
Broadcast	Ethernet Protokoll	*Nicht möglich*	UDP/IP

Im Detail werden die Verbindungsarten Unicast, Multicast und Broadcast in Abhängigkeit von den verschiedenen Adressierungsarten (MAC, AMS NetID und IP) wie folgt unterstützt:

MAC-Adressierung:
Die EAP-Nachricht wird per Ethernet Protokoll übertragen. Als Zieladresse wird die MAC-Adresse des Netzwerkadapters konfiguriert, der die Nachricht empfangen soll. Bei dieser Adressierungsart kann die EAP-Nachricht nicht von einem Router in an anderes Subnetz weitergeleitet werden, da dieser auf der Basis von IP-Adressen arbeitet. Daher kann die Nachricht nur innerhalb eines Subnetzes über Switche versendet werden.

Broadcast und Multicast

Für eine Broadcast oder Multicast Nachricht sind spezielle MAC-Adressen reserviert:

Broadcast MAC: FF-FF-FF-FF-FF-FF

Multicast MAC: Eine Multicast MAC-Adresse muss folgende Bedingungen erfüllen.

- Das niederwertigste Bit (Bit 1) des ersten Bytes hat den Wert 1 (Group-Bit).
- Das folgende Bit 2 hat den Wert 0, wenn die MAC-Adresse global eindeutig ist; oder den Wert 1, wenn die Adresse nur lokal eindeutig ist.
- Die ersten 24 Bits (Bit 3 bis 24) entsprechen der Herstellerkennung (engl. Organizationally Unique Identifier - OUI) Die OUI der Firma Beckhoff ist "00-01-05".
- Die verbleibenden 24 Bits (Bit 25 bis 48) können individuell für jede Schnittstelle festgelegt werden. Für das EtherCAT Automation Protocol ist die Folge "04-00-00" definiert.

⇨ Somit ergibt sich die Standard Multicast MAC-Adresse 01:01:05:04:00:00 für TwinCAT EAP-Geräte.

AMS NetID-Adressierung:
Die EAP-Nachricht wird per EtherCAT Protokoll vom Typ 4 (EAP) übertragen. Die notwendige Ziel MAC-Adresse wird mittels Address Resolution Protocol (ARP) unter Verwendung der konfigurierten AMS NetID ermittelt. Wie bei der MAC-Adressierung kann die EAP-Nachricht nur innerhalb des Subnetzes versendet werden.

Kommunikation via AMS NetID

Die Verwendung einer *AMS NetID* als Zieladresse hat den Vorteil, dass es sich dabei um eine logische Adressierung handelt. Die *MAC*-Adresse des Zielgerätes wird mit Hilfe eines speziellen ARP Request unter Verwendung der konfigurierten *AMS NetID* ermittelt.

Ein Anpassen der Konfiguration einer EAP-Verbindung ist auch dann nicht notwendig, wenn beispielsweise ein Steuerungsrechner bzw. ein Netzwerkadapter eines Rechners ausgetauscht wird auf Grund dessen sich die *MAC*-Adresse ändert. Es muss nur sichergestellt sein, dass der neue Steuerungsrechner die ursprüngliche *AMS NetID* erhält.

Wenn die Verbindungsart *Unicast* konfiguriert wird, ist standardmäßig auch der *Subscriber Monitoring* Mechanismus konfiguriert (siehe Gegenstellenüberwachung per ARP [▶ 13]).

IP-Adressierung:
Für die EAP-Nachricht wird das Internet Protokoll (IP) zusammen mit dem User Datagram Protocol (UDP) zur Vermittlung bzw. Adressierung des Empfängers verwendet. Die notwendige Ziel MAC-Adresse wird mittels Address Resolution Protocol (ARP) unter Verwendung der konfigurierten IP-Adresse ermittelt. Mit Hilfe der UDP/IP-Adressierung kann die EAP-Nachricht auch von einem Router in andere Subnetze weitergeleitet werden (z .B. auch ins Internet).

Für eine Broadcast oder Multicast Nachricht sind spezielle IP-Adressen reserviert:

Broadcast IP: Als Broadcast IP-Adresse ist 255.255.255.255 festgelegt. Aus dieser wird direkt die Broadcast MAC Adresse FF-FF-FF-FF-FF-FF abgeleitet.

Multicast IP: Eine Multicast IP Adresse muss in dem Adressbereich 224.0.0.0 bis 239.255.255.255 (IPv4) liegen. TwinCAT generiert beim EAP-Gerät zu jeder konfigurierten *Multicast IP Adresse* eine konforme *Multicast MAC Adresse*, die bei der Inbetriebsetzung von TwinCAT (also beim Schalten in den Run Mode) verwendet wird.

Pushed Data Exchange (n:m Verbindung)

Der Pushed Data Exchange Modus funktioniert nach dem gleichen Modell wie die Übertragung der NWV (Publisher Subscriber Prinzip). Er bietet die Möglichkeit einer n:m-Verbindung in einem Netzwerk. Dabei kann jedes EAP-Gerät ein oder mehrere EAP-Telegramme mit seinen Ausgangsprozessdaten (*TxData*) senden. Ebenso kann jedes EAP-Gerät darauf „lauschen“, ob die in einem empfangenen EAP-Telegramm enthaltenen Prozessdaten zu seinen Eingangsprozessdaten (RxData) passen und diese gegebenenfalls verarbeiten. Ein und dasselbe EAP-Gerät kann also sowohl Prozessdaten senden als auch Prozessdaten empfangen. Auf diese Weise lässt sich eine bidirektionale Kommunikation herstellen.

Beim Pushed Data Exchange kann die Adressierungsart (Unicast, Multicast oder Broadcast) pro konfiguriertes EAP-Telegramm entsprechend der Notwendigkeit frei gewählt werden.

Polled Data Exchange (1:1 Verbindung)

Der Polled Data Exchange Modus unterliegt dem Prinzip der Client/Server Architektur. Er ermöglicht mit Hilfe dieser Architektur eine „weiche“ Synchronisierung. Dabei kann ein EAP-Gerät gleichzeitig sowohl als Client als auch als Server agieren.

Verbindungsart für den Polled Modus

Für den Modus Polled Data Exchange ist ausschließlich die Verbindungsart Unicast eindeutig definiert.

Unicast (1:1-Verbindung)
Ein Client sendet ein EAP-Telegramm mit seinen Ausgangsprozessdaten an einen Server, der daraufhin seine Eingangsprozessdaten in einem eigenen EAP-Telegramm an den Client zurückschickt.

Vermittlungsprotokolle

Ethernet Protocol
Das Ethernet Protocol ist für die Vermittlung der Datenpakete im Netzwerk verantwortlich. Es übernimmt die Aufgaben der OSI-Schichten 1 und 2 (Bitübertragungsschicht und Sicherungsschicht). Im Ethernet Protocol Header wird eine Absender- und Empfängeradresse eingetragen sowie ein Ethernet-Type, der festlegt, welches Protokoll der nächsthöheren OSI-Schicht auf das Ethernet Protocol aufsetzt. Die Absender- und

Empfängeradresse wird in Form einer MAC-Adresse eingetragen. MAC bedeutet *Media Access Control* und steht hier für die eindeutige Hardware-Adresse, die jedes Ethernet Gerät ab Herstellung hat. Der Ethernet-Port eines Beckhoff-PC könnte z. B. die MAC-Adresse 00:01:05:34:05:84 haben - "00:01:05" ist die Beckhoff-Kennung, der Rest wird bei der Herstellung gewählt. Über die Absender- und die Empfänger-MAC-Adresse ist der Weg eines jeden Ethernet-Telegramms zwischen zwei sich im Netzwerk befindlichen PC bestimmt. Ein Ethernet-Telegramm kann über einen Switch, normalerweise aber nicht über einen Router weiterverarbeitet werden.

User Datagram Protocol / Internet Protocol (UDP/IP)
Der Empfänger wird über einen zusätzlichen IP-Header im Ethernet-Telegramm identifiziert und kann auf diese Weise von einem Router weiterverarbeitet werden. Das Telegramm hat den Ether Type 0x0800, wodurch festgelegt ist, dass es sich um ein IP-Telegramm handelt. Im anschließenden UDP-Header wird dann für den Source-Port sowie für den Destination-Port die Portnummer 0x88A4 verwendet. Durch diese Portnummer erkennt das TwinCAT System, dass es sich um ein Echtzeitbasiertes User Datagram handelt.

Ein EAP-Telegramm wird von TwinCAT entweder anhand des Ether Types 0x88A4 (beim Ethernet Protocol) oder anhand des Destination-Ports 0x88A4 (beim UDP/IP) identifiziert. Entsprechend schleust der TwinCAT Ethernet Treiber ein empfangenes EAP-Telegramm am NDIS Stack des Betriebssystems vorbei, so dass es von TwinCAT bevorzugt als Beckhoff Echtzeit Ethernet Telegramm behandelt wird. Beim Versenden eines EAP-Telegramms wird ebenfalls der NDIS Stack des Betriebssystems umgangen.
Nachdem ein EAP-Telegramm von einem TwinCAT PC empfangen und als solches identifiziert wurde, findet in der weiteren Telegrammverarbeitung die Zuordnung der im Telegramm transportierten *Process Data* (*PD*) zu den beim EAP-Gerät konfigurierten RxData statt. Diese Zuordnung erfolgt anhand der *PD*-ID. Das empfangene *PD* wird verworfen, wenn beim Empfänger kein RxData mit der passenden *PD*-ID konfiguriert wurde.

Die Werte der einzelnen *Process Variables* (*PV*) eines *PD* werden erst dann übernommen, wenn auch noch die Datenlänge und die Versionsnummer des empfangenen *PD* mit der erwarteten Datenlänge und Versionsnummer übereinstimmen.

2.1.2 Gegenstellenüberwachung per ARP

Das EAP setzt auf den verbindungslosen Protokollen (Ethernet Protocol sowie UDP/IP) auf. Diese Protokolle liefern keine Empfangsbestätigung für eine Nachricht. Damit der Sender eines EAP-Telegramms mitbekommen kann, dass der Empfänger nicht mehr erreichbar ist, wird vom TwinCAT EAP-Gerät das Address Resolution Protocol (ARP) zur Gegenstellenüberwachung eingesetzt. Mit Hilfe der Eigenschaft *ARP Retry Interval* kann bei einem EAP-Publisher konfiguriert werden, in welchem Zeitraster geprüft werden soll, ob der Empfänger noch erreichbar ist. Die Gegenstellenüberwachung (*Subscriber Monitoring*) kann nur aktiviert werden, wenn eine Unicast-Verbindung konfiguriert ist.

Beim *Subscriber Monitoring* schickt der Publisher entsprechend dem konfigurierten Zeitintervall ein *ARP Request* Telegramm an das konfigurierte Zielgerät. Arbeitet der Empfänger noch wie erwartet, antwortet dieser mit einem *ARP Reply* Telegramm. Andernfalls bleibt die Antwort aus. In der Diagnosevariable FrameState (siehe Publisher) wird im Fehlerfall das 3. Bit gesetzt (0x0004).

Das ARP-Handling

Das ARP-Handling für die Zuordnung von MAC-Adressen zu Netzwerkadressen (IP-Adressen) wird vom Betriebssystem (Windows) übernommen. Das ARP-Handling für die Zuordnung von MAC-Adressen zu AMS NetIDs wird hingegen vom TwinCAT System übernommen.

2.1.3 Der EAP-Sendemechanismus

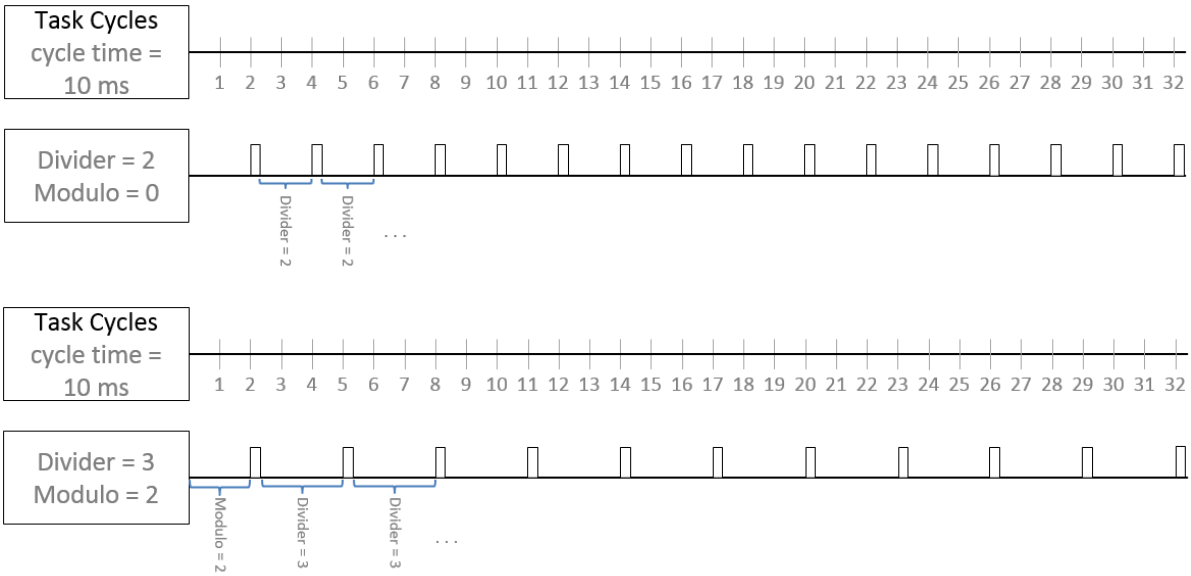
Das Senden eines EAP-Telegramms wird mit Hilfe eines Auslösemechanismus (Auslöser - engl. *Trigger*) angestoßen. Mit Hilfe der Konfiguration eines EAP-Gerätes wird bestimmt, wie dieser *Trigger* Mechanismus arbeiten soll. Dazu wird für jedes *TxData* eine *Trigger* Bedingung definiert. Wenn diese *Trigger* Bedingung erfüllt ist, wird das *TxData* per EAP-Telegramm versendet. Mit anderen Worten: Bei einem EAP-Gerät wird mit Hilfe von *Trigger* Bedingungen für jedes *TxData* konfiguriert, wie der *Trigger* Mechanismus arbeiten soll.

Es gibt fünf verschiedene Arten von Trigger Bedingungen, die hier beschrieben sind.

Überlagerung von Trigger Bedingungen

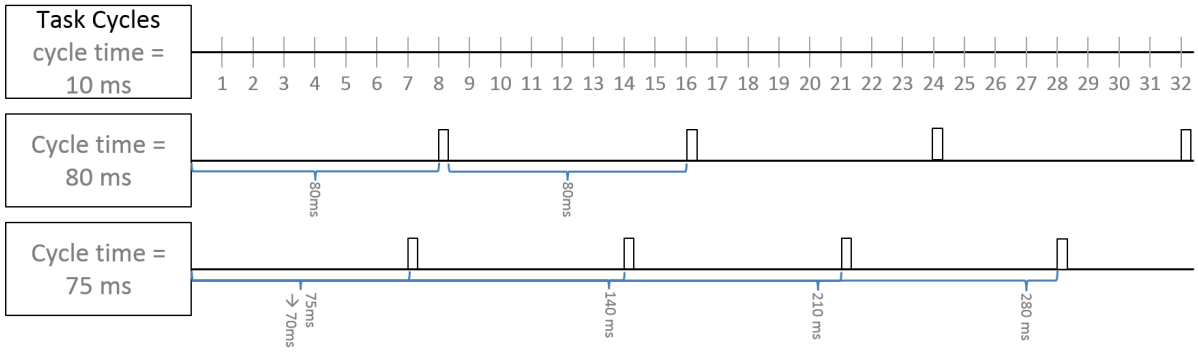
Bei den Erläuterungen zu den einzelnen Trigger Bedingungen wird darauf verwiesen, welche anderen Trigger Bedingungen deaktiviert sein müssen. Welche Bedingungen also nicht in Kombination erlaubt sind. Das Beispiel weiter unten veranschaulicht, dass sich mehrere aktive Trigger Bedingungen gegenseitig überlagern. Wie sie sich überlagern, wird nicht eindeutig definiert. Aus diesem Grund ist es empfehlenswert, alle nicht erlaubten Trigger Bedingungen zu deaktivieren.

1. **Poll Request Rx PD**
   Die Trigger Bedingung *Poll Request Rx PD* steuert das Zurücksenden eines Antwort Telegramms im sogenannten Polled Data Exchange (siehe in Kapitel Kommunikationsmethoden [▶ 10]) Modus. Sobald ein *TxData* über einen gültigen Eintrag für die Trigger Bedingung *Poll Request Rx PD* verfügt, arbeitet das betreffende *TxData* in diesem Modus. Ein gültiger Eintrag liegt dann vor, wenn dieser mit dem Objektindex eines beim EAP-Gerät konfigurierten *RxData* übereinstimmt. Dieses *RxData* definiert dann die erwartete Anfrage (Request), um das *TxData* als Antwort zurückzusenden. Empfängt das EAP-Gerät ein EAP-Telegramm, in dem das erwartete *Process Data* enthalten ist, wird das *TxData* im darauffolgenden Zyklus in einem neuen EAP-Telegramm an den Absender des Request Telegramms zurückgeschickt. Folglich fungiert das EAP-Gerät für dieses *TxData* als *Polled Data Exchange* Server, sobald die Trigger Bedingung *Poll Request RxData* aktiviert ist.
   Alle anderen Bedingungen (2 bis 5) müssen bei aktivierter *Poll Request Rx PD* Bedingung deaktiviert sein.
2. **Divider/Modulo**
   Per *Divider/Modulo* Bedingung wird festgelegt, in welchem Rhythmus, ein *TxFrame* bzw. ein *TxData* versendet wird (siehe folgende Abbildung). Der Rhythmus ist dabei immer ein Vielfaches der Zykluszeit der Task, die das EAP-Gerät treibt. Der *Divider* Wert legt dabei das Vielfache fest. Der *Modulo* Wert legt den Startzyklus fest, wann der *TxFrame* bzw. das *TxData* zum ersten Mal versendet wird. Wenn der *Divider* den Wert 0 hat, ist diese Bedingung deaktiviert.
   Die Bedingungen 3, 4 und 5 haben bei aktivierter *Divider/Modulo* Bedingung keine Relevanz; sie sollten aber dennoch deaktiviert werden. Die Bedingung 1 muss deaktiviert sein.



3. **Cycle Time**
   Das *TxData* wird in einem bestimmten Zeitintervall versendet, welches durch den *Cycle Time* Wert (Einheit in µs) festgelegt wird (siehe folgende Abbildung). Die *Cycle Time* sollte ein ganzzahliges Vielfaches der Taskzykluszeit sein. Wenn ein Wert konfiguriert wird, der kein ganzzahliges Vielfaches der Taskzykluszeit ist, wird automatisch das nächstkleinere Vielfache gesetzt, ggf. auch 0. Wenn der Wert 0 µs beträgt, ist diese Bedingung deaktiviert.

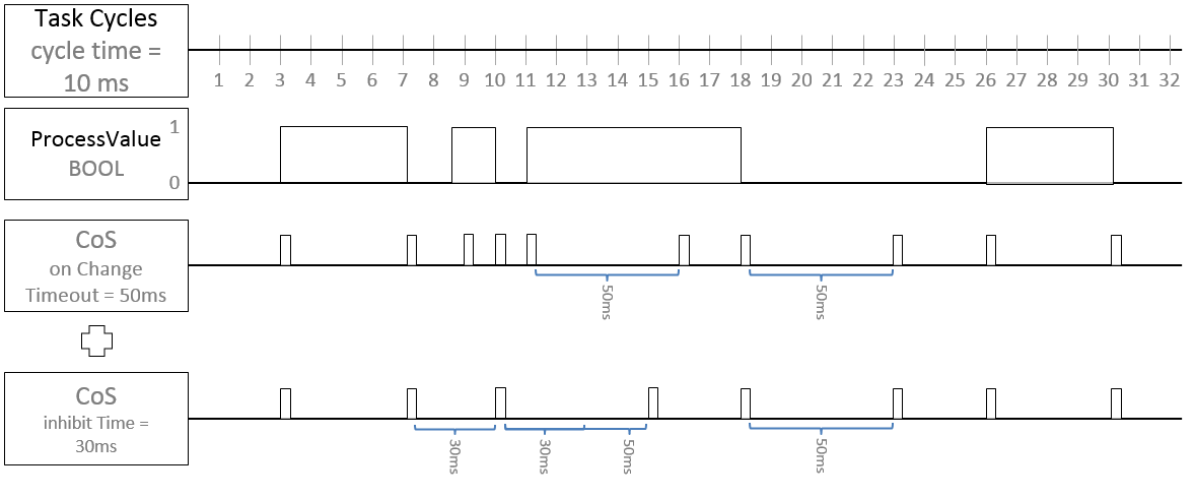
Die Trigger Bedingungen 1, 2, 4 und 5 müssen deaktiviert sein, wenn die Trigger Bedingung *Cycle Time* aktiviert ist.



Der Zusammenhang zwischen der Cycle Time und der Taskzykluszeit

Angenommen die Taskzykluszeit beträgt 5ms (5000µs) und die Cycle Time von *Process Data* A beträgt 10000µs und die von *Process Data* B 20000µs. Nun wird die Taskzykluszeit von 5ms auf 15ms (15000µs) verlangsamt. Weder die Cycle Time von *Process Data* A noch die von *Process Data* B ist ein Vielfaches der Taskzykluszeit; die Cycle Time ist also nicht ohne Rest durch die Taskzykluszeit teilbar.
Die Folge ist, dass *Process Data* A dann nur noch alle 15ms (15000µs) und *Process Data* B nur noch alle 30ms (30000µs) gesendet wird.

4. **Change of State (CoS): On Change Timeout**
   Das *TxData* wird erst dann gesendet, wenn sich der Wert einer seiner Variablen zum vorherigen Wert verändert hat. Dabei wird ein maximales Zeitintervall als sogenannte Timeout Time (Einheit in µs) definiert. Ändert sich der Wert einer Variablen nicht innerhalb dieses Intervalls, wird nach Ablauf des Zeitintervalls das *TxData* dennoch gesendet (siehe folgende Abbildung). Der Wert für das Zeitintervall kann nur ein ganzzahliges Vielfaches der Taskzykluszeit sein. Wenn das Zeitintervall auf 0 µs gestellt wird, ist die Trigger Bedingung *CoS On Change Timeout* deaktiviert. Die Trigger Bedingungen 1, 2 und 3 müssen deaktiviert sein, wenn die Trigger Bedingung *CoS On Change Timeout* aktiviert ist.



5. **Change of State (CoS): Inhibit Time**
   Die *Inhibit Time* legt ein minimales Zeitintervall fest, so dass das *TxData* nicht eher gesendet wird, bis dieses Zeitintervall nach dem letzten Versenden abgelaufen ist.
   Die *Inhibit Time* legt also ein minimales Zeitintervall in µs fest, nach der das *TxData* versendet wird – auch dann, wenn sich ein Wert der enthaltenen *TxVariable* geändert hat (siehe folgende Abbildung). Der Wert für dieses Zeitintervall kann nur ein ganzzahliges Vielfaches der Taskzykluszeit sein, und er muss kleiner als der Wert von *CoS On Change Timeout* sein. Wenn das Zeitintervall auf 0 µs gestellt wird, ist die Trigger Bedingung *Inhibit Time* deaktiviert.
   Die Trigger Bedingungen 1, 2 und 3 müssen deaktiviert sein, wenn die Trigger Bedingung *Inhibit Time* aktiviert ist.

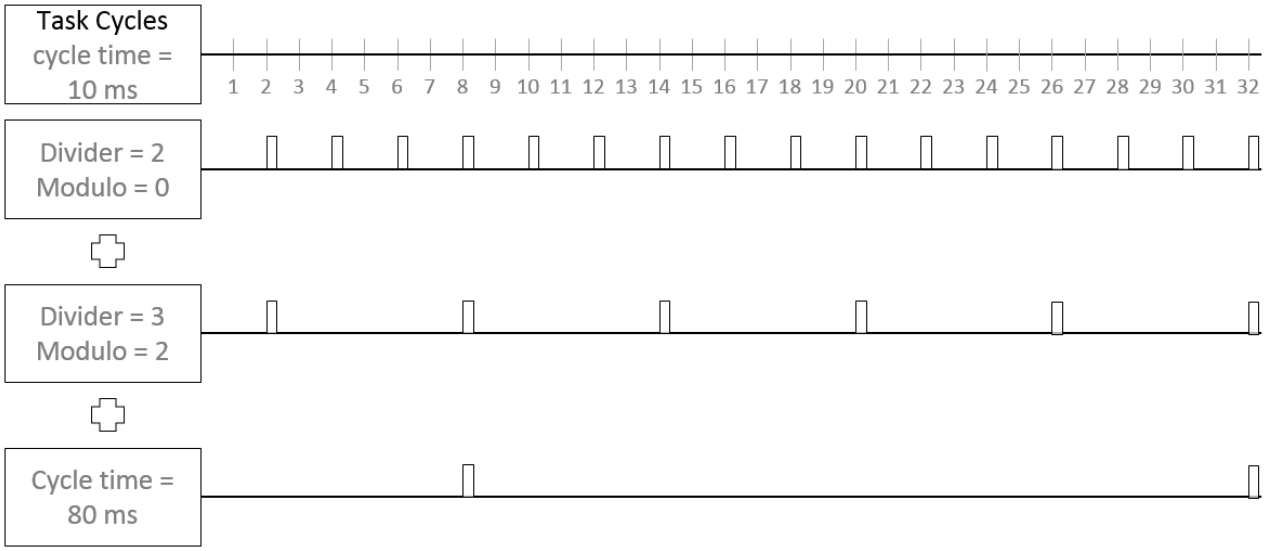
Konfigurationsmöglichkeiten der Trigger Bedingungen

Die Trigger Bedingungen 1, 3 und 5 (Poll Request RxData, Cycle Time und Inhibit Time) können über das EAP-Objektverzeichnis konfiguriert werden (siehe Kapitel CANopen Objektverzeichnis [▶ 53] in der Dokumentation zum TwinCAT EAP-Gerät).

Besonderheiten der Trigger Bedingungen

Bei allen Trigger Bedingungen, die ein Zeitintervall definieren, kann dieses Intervall nicht kleiner sein als die Zykluszeit der Task, die das EAP-Gerät treibt.

Eine Kombination von den Bedingungen ist nicht zu empfehlen, da sie nicht eindeutig definiert sind. Als Beispiel für die Komplexität kann das folgende Beispiel genommen werden:



In der letzten Zeile wird deutlich, dass die Sendung bei 160ms und 240ms nicht stattfindet, da es von den zusätzlichen Divider/Modulo Bedingungen verhindert wird.

2.1.4 Performance des EAP

Bei der Verwendung des TwinCAT EAP-Gerätes sind die zeitlichen Randbedingungen der verwendeten Netzwerkarchitektur zu berücksichtigen:

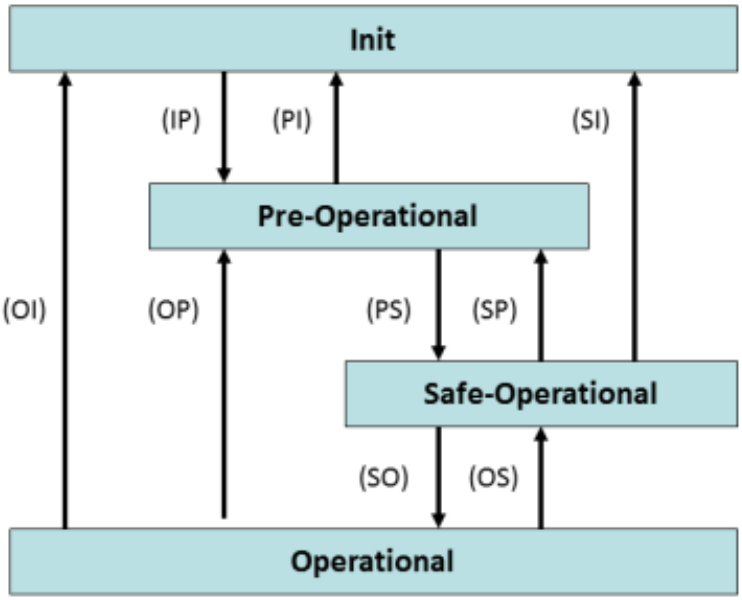
- In einer Netzwerkarchitektur, in der Telegramme ausschließlich über Switches geschickt werden (z. B. per Ethernet Protocol), können rund 10 ms Kommunikationszyklus und auch darunter erreicht werden.
- In einer Netzwerkarchitektur, in der Telegramme auch über einen Router geschickt werden (also per UDP/IP), können rund 100 ms Kommunikationszyklus erreicht werden.

Dabei kann in einem Netzwerk, in dem zusätzlich andere Kommunikation stattfindet, diese die Performanz der EAP-Kommunikation beeinträchtigen.

2.1.5 Die EAP State Machine

Über die EAP State Machine (EAP SM) wird der Zustand des EAP-Gerätes gesteuert. Je nach Zustand (Status) sind unterschiedliche Funktionen im EAP-Gerät zugänglich bzw. ausführbar. Es werden folgende Zustände unterschieden:

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



Der Reguläre Zustand eines jeden EAP-Gerätes nach dessen Start ist der Status **OP**. Bis zum Erreichen des Status **OP** wird das EAP-Gerät der Reihe nach einmal in jeden Zustand geschaltet. Bei jedem Zustandsübergang (Transition) werden bestimmte Aktionen vom EAP-Gerät durchgeführt. Tritt während einer der Transitionen ein Fehler auf, kann das Gerät nicht in den entsprechenden Folgezustand geschaltet werden und verbleibt somit im zuletzt erreichten Zustand. Anhand eines auslesbaren Fehlercodes kann die Ursache für den Fehler diagnostiziert werden.

Init

Der Status **Init** ist beim EAP-Gerät grundsätzlich ein temporärer Zustand. Das heißt, dass das EAP-Gerät nicht explizit in den Status **Init** versetzt werden kann. Dennoch gibt es Fälle, in denen die SM das EAP-Gerät in den Status **Init** zurücksetzt. In diesem Zustand ist weder Mailbox- noch Prozessdatenkommunikation mit dem EAP-Gerät möglich.

Pre-Operational (Pre-Op)

Beim Übergang von **Init** nach **Pre-Op** prüft das EAP-Gerät, ob die Mailbox korrekt initialisiert wurde. Im Status **Pre-Op** ist Mailbox- aber keine Prozessdatenkommunikation möglich.

Safe-Operational (Safe-Op)

Beim Übergang von **Pre-Op** nach **Safe-Op** prüft das EAP-Gerät die internen Objektreferenzen und aktualisiert

- die zykluszeitbasierten Konfigurationsparameter,
- die Referenzzeiger auf die Input- und Output-Variablen des Prozessabbildes sowie
- das Mapping jeder Publisher/Subscriber Variablen zu ihrer Prozessvariablen aus der SPS.

Im Status **Safe-Op** wird Mailboxkommunikation und das Senden von Publisher Variablen betrieben. Es werden jedoch noch keine EAP-Telegramme empfangen.

Operational (Op)

Beim Übergang von **Safe-Op** nach **Op** prüft das EAP-Gerät abschließend noch einmal, ob ein Fehler während des Startens des EAP-Gerätes aufgetreten ist.

Im Status **Op** empfängt das EAP-Gerät ankommende EAP-Telegramme und kopiert ggf. die empfangenen Prozessdaten in die Input-Variablen. Es wird sowohl Mailboxkommunikation betrieben als auch Publisher Variablen gesendet und Subscriber Variablen empfangen.

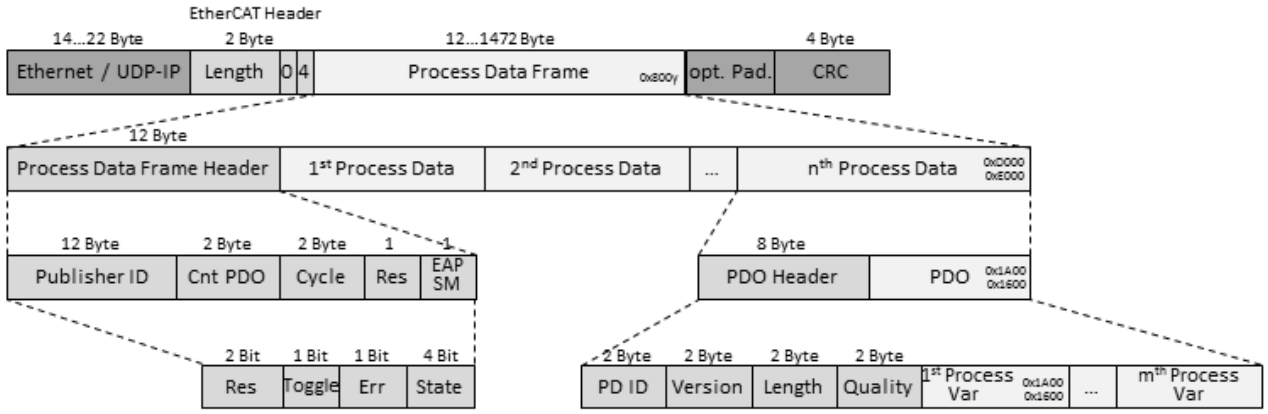
Wenn in einem der Zustandsübergänge ein Fehler auftritt, bleibt das EAP-Gerät entweder im zuletzt erreichten Zustand oder es wird in den Zustand **Safe-Op** zurückgesetzt. Gleichzeitig werden das Fehler-Bit und ein entsprechender Fehler-Code gesetzt (vgl. Kapitel Das CANopen Objektverzeichnis [▶ 53] in der Dokumentation zum TwinCAT EAP-Gerät). Typische Fehler treten beispielsweise wegen Inkonsistenzen im CANopen Objektverzeichnis auf, so dass die Konfiguration ungültig ist.

2.2 Technische Grundlagen

2.2.1 EAP-Telegrammstruktur

Ein EAP-Telegramm kann per Ethernet (Ethertype = 0x88A4 entspricht dem EtherCAT Protocol) oder per UDP/IP (UDP-Port 0x88A4) übertragen werden. Wenn EAP auf dem EtherCAT Protocol basiert, sind die EAP-spezifischen Telegrammteile in den Nutzdaten des EtherCAT Protocols eingebettet.

Das EAP-Telegramm besteht aus einem *Process Data Frame Header* und einem oder mehreren *ProcessData (PD)*. Ein *PD* ist die Haupteinheit beim Datenaustausch via EAP. Ein *PD* setzt sich zusammen aus einem sogenannten *PDO Header* und mindestens einer *Process Variable (PV)*. Die *Process Variables* eines *PD* zusammengenommen bilden das *ProcessDataObject (PDO)*. Vgl. dazu folgende Abbildung.



Der Process Data Frame Header

Der *Process Data Frame Header* besteht aus fünf Feldern (vgl. Zeile 2 und 3 in der Abbildung oben). Das letzte Feld (EAP SM) erweitert das NWV-Telegramm von TwinCAT 2 im Hinblick auf die Inhalte. Dieses Feld beinhaltet unter anderem Informationen zum aktuellen Zustand des EAP-Gerätes.

Publisher ID
Anhand der Publisher ID wird der Sender eines Telegramms identifiziert. Sie enthält die AMS NetID des Absenders. Wenn bei einem Empfänger konfiguriert ist, dass dieser ausschließlich Telegramme von einem bestimmten Absender verarbeiten soll, wird anhand des Feldes Publisher ID geprüft, ob das EAP-Telegramm von diesem Absender stammt.

Cnt PDO
In diesem Feld steht die Anzahl der im Telegramm enthaltenen *ProcessData*, damit der Empfänger das Telegramm vollständig verarbeiten kann.

Cycle
Der Wert dieses Feldes wird bei jedem Taskzyklus des Absenders inkrementiert. Auf der Seite des Empfängers kann der Inhalt dieses Feldes als Sequenznummer verwendet werden, um z. B. zu prüfen, ob ein Telegramm verloren ging. Beim Polled Data Exchange Modus sollte serverseitig dieses Feld überwacht werden.

Res
Reserviert für zukünftige Erweiterungen.

EAP SM
Dieses Feld setzt sich aus 4 Subeinträgen zusammen.

- **Res:** Reserviert für zukünftige Erweiterungen.

- **Toggle:** Dieses Bit wird für jedes neue EAP-Telegramm umgeschaltet.
- **Err:** Dieses Feld zeigt an, ob ein Fehler bei der EAP State Machine aufgetreten ist.
- **State:** Dieses Feld enthält den Wert für den aktuellen Zustand der EAP State Machine.
  1 : Init
  2 : Pre-Operational
  4 : Safe-Operational
  8 : Operational

Andere Werte als diese sind nicht erlaubt.

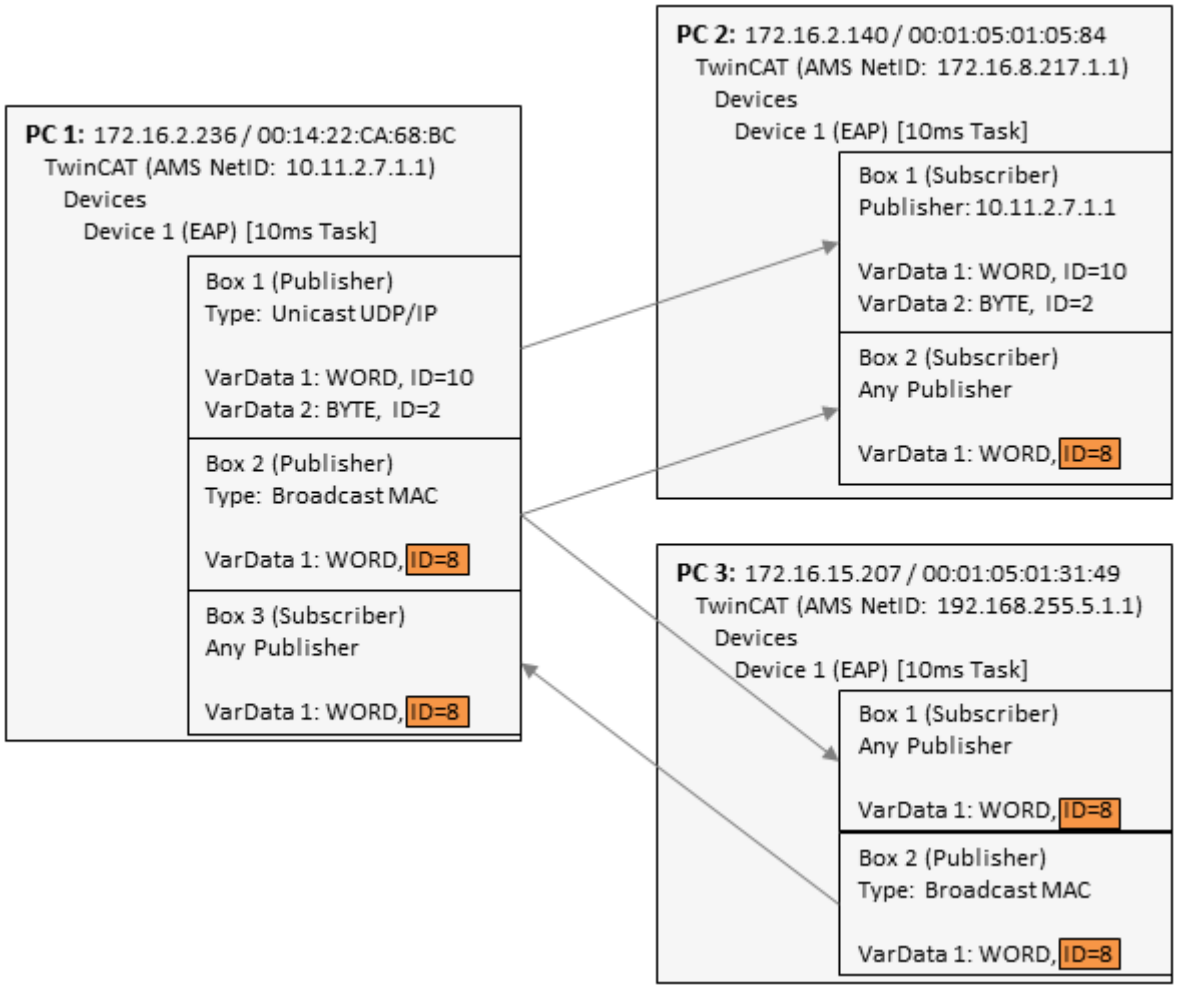
Der PDO Header

Der *PDO Header* besteht aus 4 Feldern, von denen jedes Feld eine Datenlänge von 2 Byte hat (vgl. Zeile 3 und 4 in der Abbildung oben).

PD ID
Die *PD ID* (16 Bit) dient zur globalen Identifizierung der *EAP-Variablen* und sollte eindeutig im Netzwerk sein.

Auswahl der ProcessData Identifier (PD ID)

Um eine eindeutige Zuordnung zu erreichen wird empfohlen, für jede Datenübertragung zwischen zusammenhängenden PCs unterschiedliche IDs zu verwenden.
Begründung: In der folgenden Abbildung erhält *PC 2* in *Box 2 (Subscriber)* nicht nur die vorgesehenen Variable aus *Box 2 (Publisher)* mit der ID = 8 von *PC 1*, sondern, da als Broadcast(!) gesendet, auch die Variable aus *Box 2 (Publisher)* von *PC 3*. Eine Unterscheidung ist in *PC 2* dann nicht mehr möglich!



Version
In dieses Feld kann eine Versionsnummer für das *PD* eingetragen werden. Welchen Einfluss die *Version* bei der EAP-Kommunikation hat, wird im Kapitel Kommunikationsmethoden [▶ 10] im Abschnitt *Vermittlungsprotokolle* erläutert. Die Versionsnummer sollte konsequent erhöht werden, wenn mindestens eine der folgenden Änderungen am *ProcessData* vorgenommen werden:

- Die Datenlänge des *ProcessData* wird verändert
- Der Datentyp einer Variable des *ProcessData* wird verändert
- Die Reihenfolge der Variablen in einem PDO wird verändert

Length
Dieses Feld enthält die Länge des *ProcessData* in Bytes. Die Länge des *Process Data Header* selbst ist nicht darin enthalten.

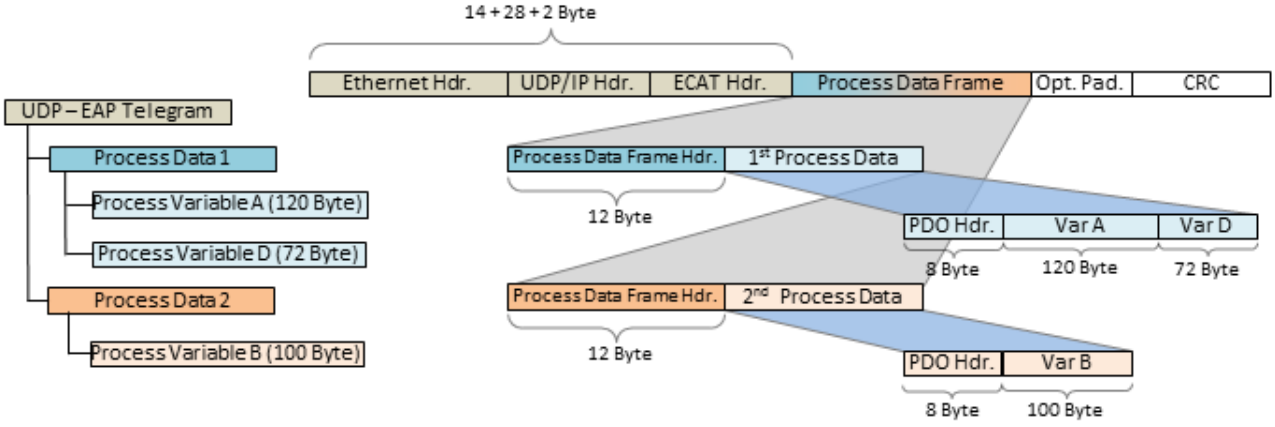
Quality
Der Wert dieses Feldes zeigt das Alter der *ProcessData* in der Einheit [100 µs] an, wenn der Wert zwischen 0x0 und 0xEFFF liegt. Ein Wert größer oder gleich 0xF000 bedeutet, dass das *ProcessData* ungültig ist.

Die Process Variable (PV)

Die *Process Variable* enthält die tatsächlich zu übertragenen Daten (vgl. *Process Var* in Zeile 4 der ersten Abbildung oben). Die Datenlänge einer *PV* darf nur so groß sein, dass das gesamte EAP-Telegramm nicht größer als 1514 Byte wird.

Größenberechnung eines EAP-Telegramms

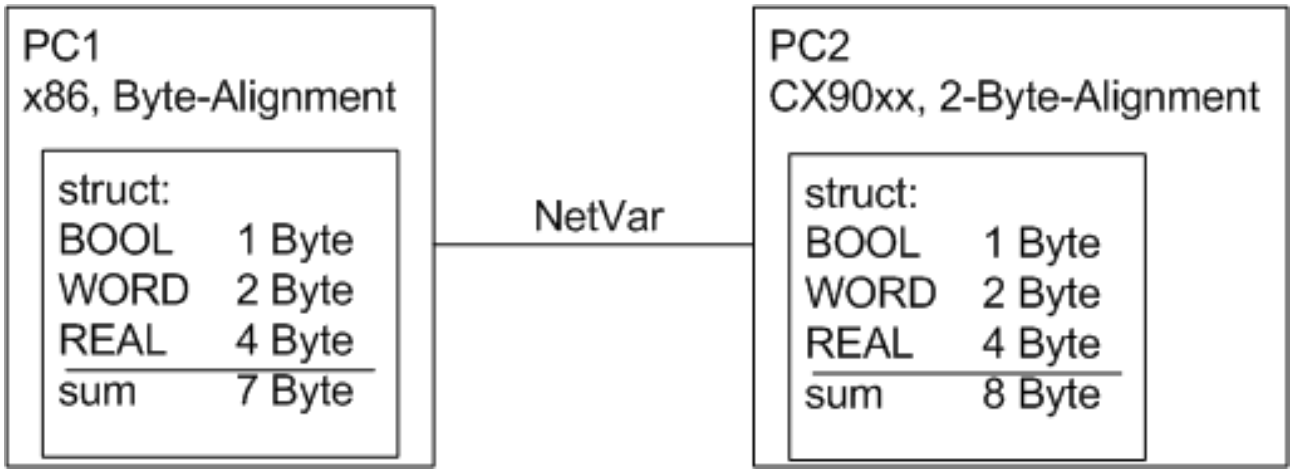
Die Summe der Längen aller Header und aller Publisher Variablen darf die Grenze von 1514 Bytes nicht überschreiten. Das abgebildete EAP-Telegramm (siehe folgende Abbildung) hat eine Gesamtgröße von
14 + 28 + 2 + 12 + 8 +120 + 72 + 12 + 8 + 100 = 376 Bytes.



(Diese Berechnung inkludiert 28 Bytes für den UDP/IP Hdr. Die 28 Bytes stehen bei reiner Ethernet Kommunikation für Daten bereit.)

Datendarstellung auf unterschiedlichen Plattformen

Zu beachten ist, dass einfache wie komplexe Daten (WORD, ARRAYs, REAL, STRING, benutzerdefinierte Strukturen) auf unterschiedlichen Plattformen intern unterschiedlich dargestellt werden. x86-Plattformen arbeiten im Byte-Alignment, andere (Arm®) im 2- oder 4-Byte-Alignment.
Das bedeutet, wenn eine komplexe Struktur jeweils in einem x86/PC-SPS-Projekt und einem Arm®-SPS-Projekt angelegt wird, diese jeweils eine andere effektive Größe und einen anderen inneren Aufbau haben kann.



Im Beispiel (siehe Abbildung oben) ist die Struktur im PC2 größer als im PC1. Außerdem passen die Word- und Real-Variablen nicht zueinander, da im PC1 an jeder Byte-Position eine Variable beginnen kann, im PC2 jedoch nur an jeder geradzahligen Byte-Position.

Es wird empfohlen beim Aufbau von Strukturen folgende Regeln zu beachten, damit keine Größenunterschiede durch das Alignment verursacht werden:

- zuerst alle 4-Byte-Variablen (müssen auf einer durch 4 teilbaren Adresse liegen)
- dann alle 2-Byte-Variablen (müssen auf einer durch 2 teilbaren Adresse liegen)
- dann alle 1-Byte-Variablen

Weitere Empfehlung:

- wenn der Datentyp STRING(x) in einer SPS verwendet wird, dann gehört die Nullterminierung des Strings ebenfalls zum String selbst, so dass gilt: x+1 muss durch 4 teilbar sein. Andernfalls liegt kein 4-Byte-Alignment vor.
- obige Regeln gelten auch für Substrukturen.

Weitere Hinweise finden Sie im Beckhoff Information System in der Dokumentation PLC: Strukturen oder Alignment.

Verwendung von Busklemmen-Controllern (BCxxxx, BXxxxx)

Fließkommazahlen (Datentyp REAL) können auf Busklemmen-Controllern (BCxxxx, BXxxxx) nicht übertragen werden, da die Darstellung einer Fließkommazahl auf einem Busklemmen-Controller von der auf einer x86-Plattform abweicht.
Für vorzeichenbehaftete Werte kann z. B. der Datentyp SINT verwendet werden.

3 Diagnose einer EAP-Verbindung

Das TwinCAT EAP-Gerät stellt verschiedene Variablen bereit, mit Hilfe derer die Qualität oder der Ausfall einer EAP-Kommunikation diagnostiziert werden kann. Solche Diagnosevariablen sind sowohl beim Publisher als auch beim Subscriber zu finden. Sie können programmatisch ausgewertet werden, um auf mögliche Störungen zu reagieren.

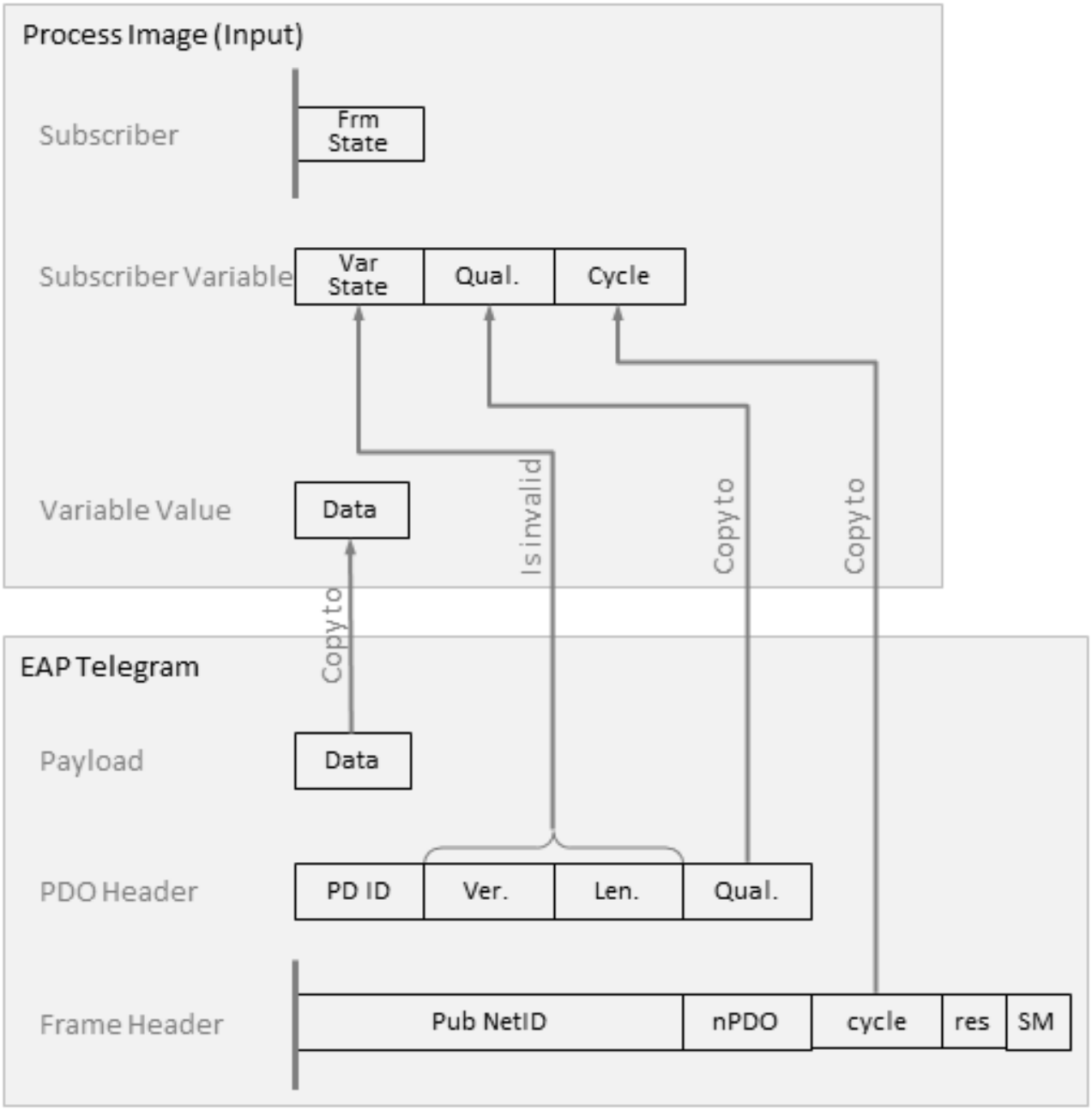
3.1 Subscriber

Beim Subscriber gibt es die Diagnosevariablen *Quality*, *CycleIndex* und *VarState*.

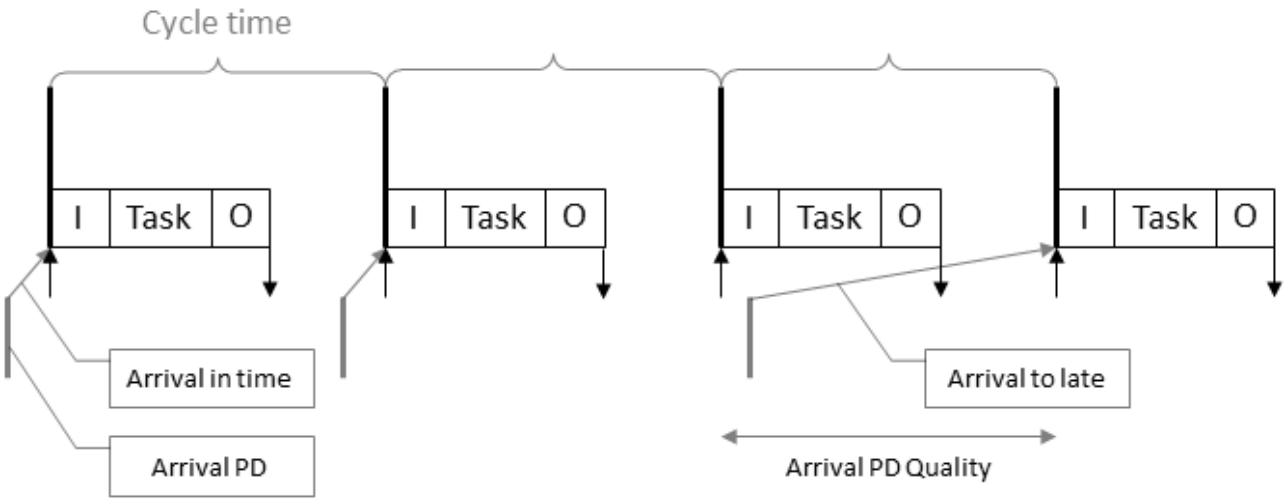
- *Quality*:
  Die *Quality* ist ein Indikator für das Alter der empfangenen Daten.
- *CycleIndex*:
  Mit Hilfe des *CycleIndex* können Übertragungsausfälle detektiert werden.
- *VarState*:
  Mittels der Variable *VarState* wird angezeigt, ob Diskrepanzen zwischen dem konfigurierten *RxData* und den ankommenden *ProcessData* vorliegen, die das Empfangen der Daten verhindern.

Quality

Die *Quality* Variable enthält die Zeit in [100 µs], um die dieses *ProcessData* zu spät beim Empfänger angekommen ist. Sie wird in jedem Zyklus um die Zykluszeit inkrementiert, unmittelbar bevor das Eingangsprozessabbild des TwinCAT Gerätes aktualisiert wird. Die Aktualisierung des Eingangsprozessabbildes erfolgt bei der Verarbeitung der empfangenen EAP-Telegramme. Während dieser Verarbeitung werden einige Werte eines jeden Telegramms den entsprechend zugehörigen Variablen des Eingangsprozessabbildes zugewiesen.



Wie in der oberen Abbildung dargestellt, wird dabei z.B. der empfangene *Quality* Wert aus dem Telegramm der entsprechenden *Quality* Variablen des Eingangsprozessabbildes zugewiesen.
Wenn in einem EAP-Telegramm die *Quality* den Wert 0 hat, wird die *Quality* des Eingangsprozessabbildes also wieder auf den Wert 0 zurückgesetzt, sobald das Telegramm empfangen wurde.



Die vorhergehende Abbildung veranschaulicht, wie ein verspätet ankommendes EAP-Telegramm den *Quality* Wert beeinflusst: Zu Beginn wird das EAP-Telegramm rechtzeitig vor dem Start des 1. Zyklus empfangen. Ebenso im 2. darauffolgenden Zyklus. Danach erreicht das Telegramm den Empfänger erst nachdem der 3. Zyklus bereits begonnen hat. Dies hat zur Folge, dass das Telegramm erst im 4. Zyklus verarbeitet werden kann. Entsprechend wird der Wert der *Quality* Variablen während des 3. Zyklus um die *Cycle time* inkrementiert aber nicht wieder auf den Wert 0 zurückgesetzt. Erst im 4. Zyklus wird die *Quality* Variable wieder auf den Wert 0 zurückgesetzt, indem der *Quality* Wert des verspäteten Telegramms zugewiesen wird.

Diagnosevariable Quality
Angenommen der Taskzyklus des Subscribers ist zehn Mal so schnell wie der des Publishers, dann wird nur jeden zehnten Zyklus ein EAP-Telegramm empfangen. Folglich kommt neun Zyklen lang kein Telegramm beim Subscriber an, so dass auch die *Quality* Variable des Eingangsprozessabbildes neun Zyklen lang nicht zurückgesetzt werden kann. Die *Quality* Variable wird also neun Zyklen lang um die Zykluszeit inkrementiert. Sie steigt also bis auf das Neunfache der Taskzykluszeit.

Fazit: Ein "langsamer" Sender (z.B. 100ms Sendetakt beim Publisher) führt bei einem "schnellen" Empfänger (z.B. 10ms Empfangstakt beim Subscriber) zu einem entsprechend ansteigenden Wert bei der Diagnosevariable *Quality*.

Wichtig ist also eine Berücksichtigung unterschiedlicher Zykluszeiten beim Senden und Empfangen von EAP-Telegrammen. In diesem Zusammenhang ist besonders auf die *Trigger* Bedingungen (siehe Der EAP-Sendemechanismus [▶ 13]) zu achten, die beim Sender konfiguriert werden.

EL6601/EL6614

Bei Verwendung der EL66xx ist der Ankunftszeitpunkt eines *ProcessData* genau dann, wenn die Daten im Eingangsprozessabbild des EAP-Gerätes vorliegen, nicht wenn sie bei der EL66xx oder im Eingangsabbild des EtherCAT-Gerätes ankommen.

CycleIndex
Der *CycleIndex* (Größe 16 Bit) ist ein Zähler, der vom Publisher mit dem *ProcessData* mit gesendet wird. Üblicherweise wird er senderseitig mit jedem neuen Zyklus inkrementiert bevor das EAP-Telegramm versendet wird und lässt so einen Rückschluss auf Übertragungsunterbrechungen zu. Er ist auf der Empfängerseite (beim Subscriber) als *CycleIndex* auslesbar (Vergleiche oberste Abbildung dieses Kapitels).

VarState
Der *VarState* (Größe 16 Bit) gibt Auskunft über den aktuellen Status des *RxData*.
Folgende Werte sind für den *VarState* möglich:

Kurzbeschreibung	Bit	Beschreibung
Invalid Hash/Version	VS.0	Das Bit wird auf 1 gesetzt, wenn ein *ProcessData* nicht empfangen werden konnte, weil die Version des empfangenen ProcessData nicht mit der konfigurierten Version dieses RxProcessData übereinstimmt. Andernfalls ist das Bit auf 0 gesetzt.
Invalid Variable Length Received	VS.1	Das Bit wird auf 1 gesetzt, wenn ein *ProcessData* nicht empfangen werden konnte, weil die Datenlänge der empfangenen Variable nicht mit der konfigurierten RxVariable übereinstimmt. Andernfalls ist das Bit auf 0 gesetzt.

EL6601/EL6614

Der VarState wird nicht bei der Verwendung der EL-Klemmen EL66xx angelegt.

3.2 Publisher

Beim Publisher stehen die Diagnosevariablen *VarState* und *FrameState* zur Verfügung. Mittels dieser Diagnosevariablen werden detaillierte Eigenschaften für das Versenden eines EAP-Telegramms verdeutlicht.

VarState
Der *VarState* (Größe 16 Bit) gibt Auskunft über den aktuellen Status des *TxData*.
Folgende Werte sind für den *VarState* möglich:

Kurzbeschreibung	Bit	Beschreibung
Not Sent (variable skipped)	VS.0	Solange der Wert 0 ist, wird das TxProcessData gesendet. Andernfalls wird das Senden des TxProcessData ausgesetzt.

EL6601/EL6614

Diese Variable wird nicht bei der Verwendung der EL-Klemmen EL66xx angelegt.

FrameState
Der *FrameState* (Größe 16 Bit) gibt Auskunft über den aktuellen Zustand des *TxFrames*. Folgende Werte sind für den *FrameState* möglich:

Kurzbeschreibung	Bit	Beschreibung
Not Sent (frame skipped)	FS.0	Solange der Wert 0 ist, wird der Ethernet Frame gesendet. Andernfalls wird das Senden des Frames ausgesetzt.
Error (frame oversized)	FS.1	Wenn der Wert 1 ist, ist die maximale Größe eines Ethernet Frames überschritten worden. Die verknüpfte Variable muss kleiner gewählt werden.
Subscriber Missing (Unicast only)	FS.2	Dieses Bit wird nur dann gesetzt, wenn das Subscriber Monitoring (siehe unter Kapitel Publisher Box [▶ 45]) aktiviert ist. Sobald mit Hilfe des Subscriber Monitoring Mechanismus festgestellt wird, dass das EAP-Gerät, an den der Ethernet Frame gesendet wird, nicht mehr erreichbar ist, wird der Wert gesetzt. Sobald die Erreichbarkeit wieder hergestellt ist, wird das Bit wieder auf 0 zurückgesetzt.

EL6601/EL6614

Diese Variable wird nicht bei der Verwendung der EL-Klemmen EL66xx angelegt.

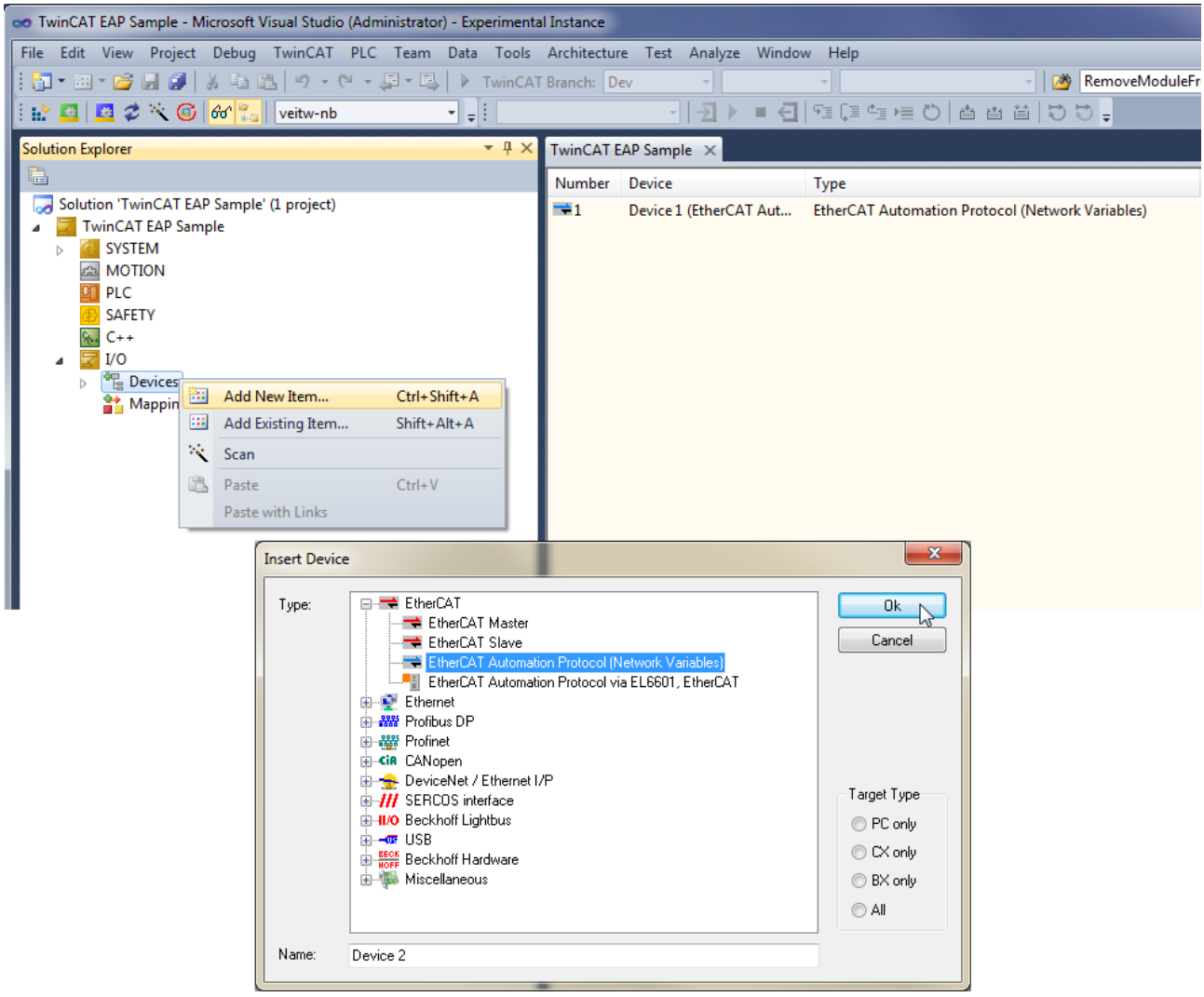
4 Das Anlegen einer EAP-Konfiguration

Eine Kommunikationsverbindung zwischen TwinCAT-Steuerungen via EAP wird mit Hilfe von TwinCAT 3 angelegt.

4.1 Hinzufügen eines EAP-Gerätes

In TwinCAT 3 wird unter dem Pfad [*I/O*] → [*Devices*] ein EtherCAT Automation Protocol Gerät hinzugefügt (vgl. folgende Abbildung).

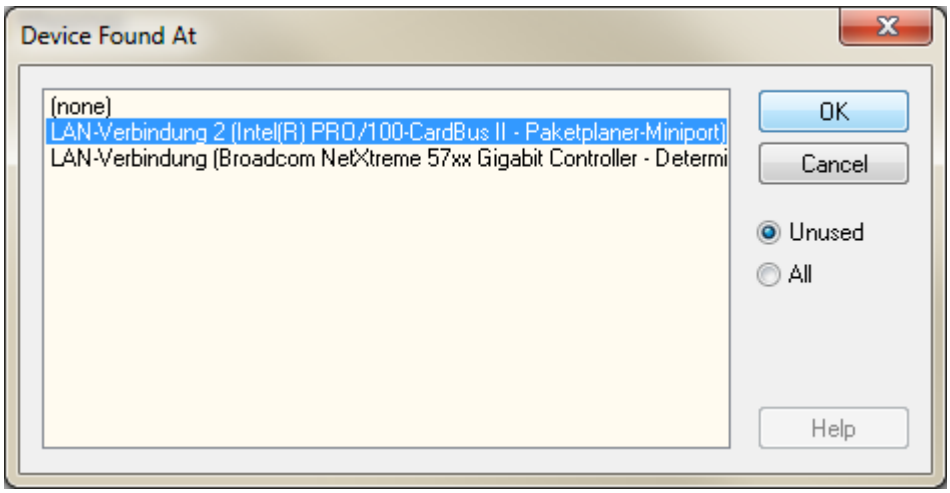
1. Klicken Sie im Kontextmenü des Knoten [*Devices*] auf das Kommando [*Add New Item...*]
   ⇨ Es öffnet sich der Dialog *Insert Device*.
2. Wählen Sie das *EtherCAT Automation Protocol (Network Variables)* unterhalb des Knotens EtherCAT aus und bestätigen Sie Ihre Auswahl mit [*OK*].



Wenn der PC noch mehrere nicht verwendete echtzeitfähige Netzwerkadapter zur Verfügung hat, erscheint ein Dialog, bei dem ein Netzwerkadapter ausgewählt werden kann (siehe folgende Abbildung).

1. Wählen Sie einen gewünschten Adapter aus bestätigen Sie mit [*OK*].

⇨ Anschließend ist das EAP-Gerät mit der Bezeichnung *Device 1 (EtherCAT Automation Protocol)* unterhalb des Knotens *Devices* zu sehen.



Im nächsten Schritt können die *Publisher* bzw. *Subscriber Variablen* konfiguriert werden.

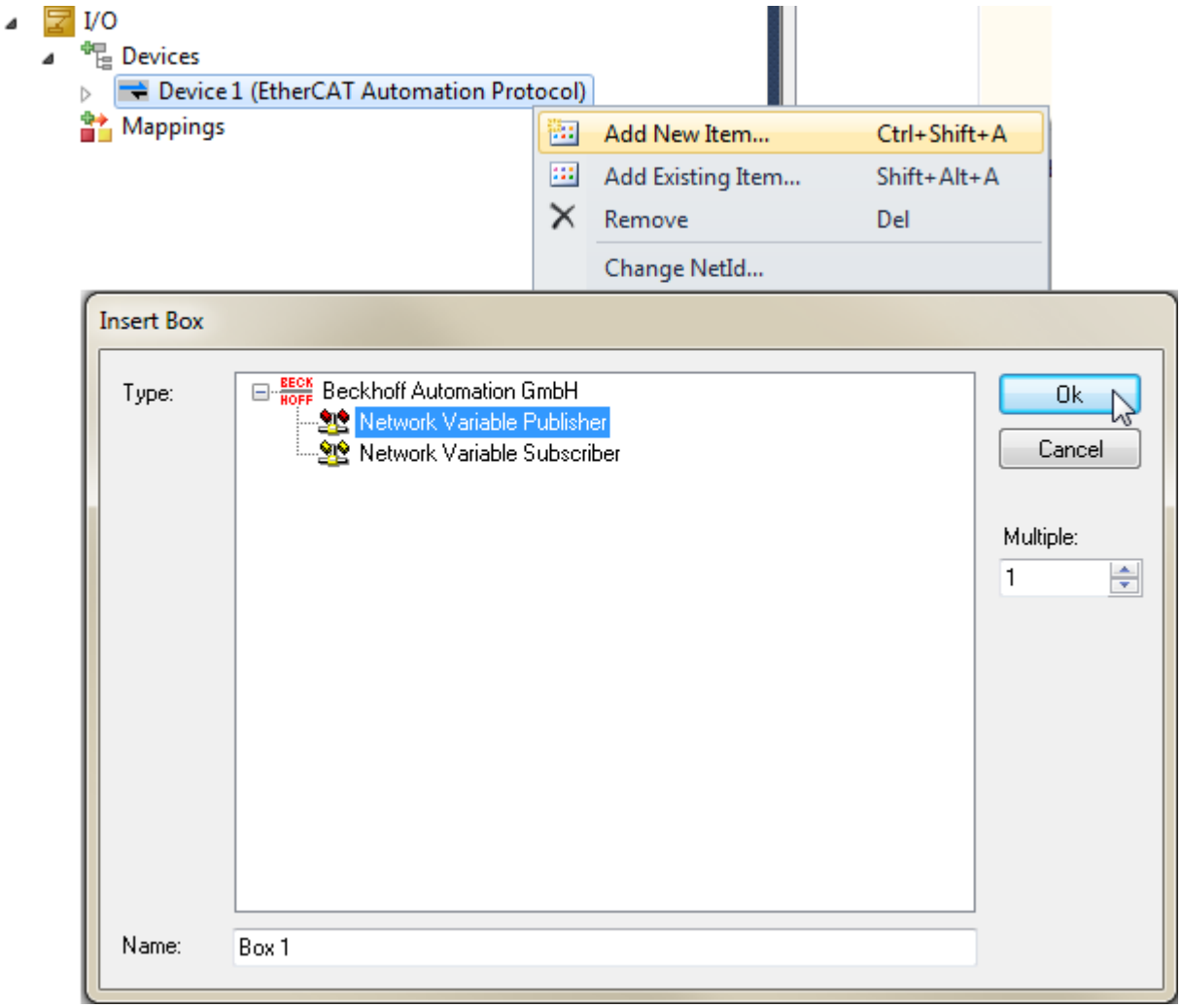
4.2 Hinzufügen von Publisher Variablen

Wenn das EAP-Gerät Variablen versenden soll, müssen zu seiner vollständigen Konfiguration noch *Publisher Variablen* hinzugefügt werden (vgl. folgende Abbildung).

1. Klicken Sie im Kontextmenü des EAP-Geräteknotens (*Device 1*) auf den Eintrag [*Add New Item...*].
   ⇨ Es öffnet sich der Dialog *Insert Box.*
2. Wählen Sie *Network Variable Publisher* aus und klicken Sie auf [*OK*].

Mehrere Publisher hinzufügen

Wenn mehrere *Publisher* angelegt werden sollen, kann die Anzahl mit Hilfe des Eingabefeldes *Multiple* eingestellt werden.



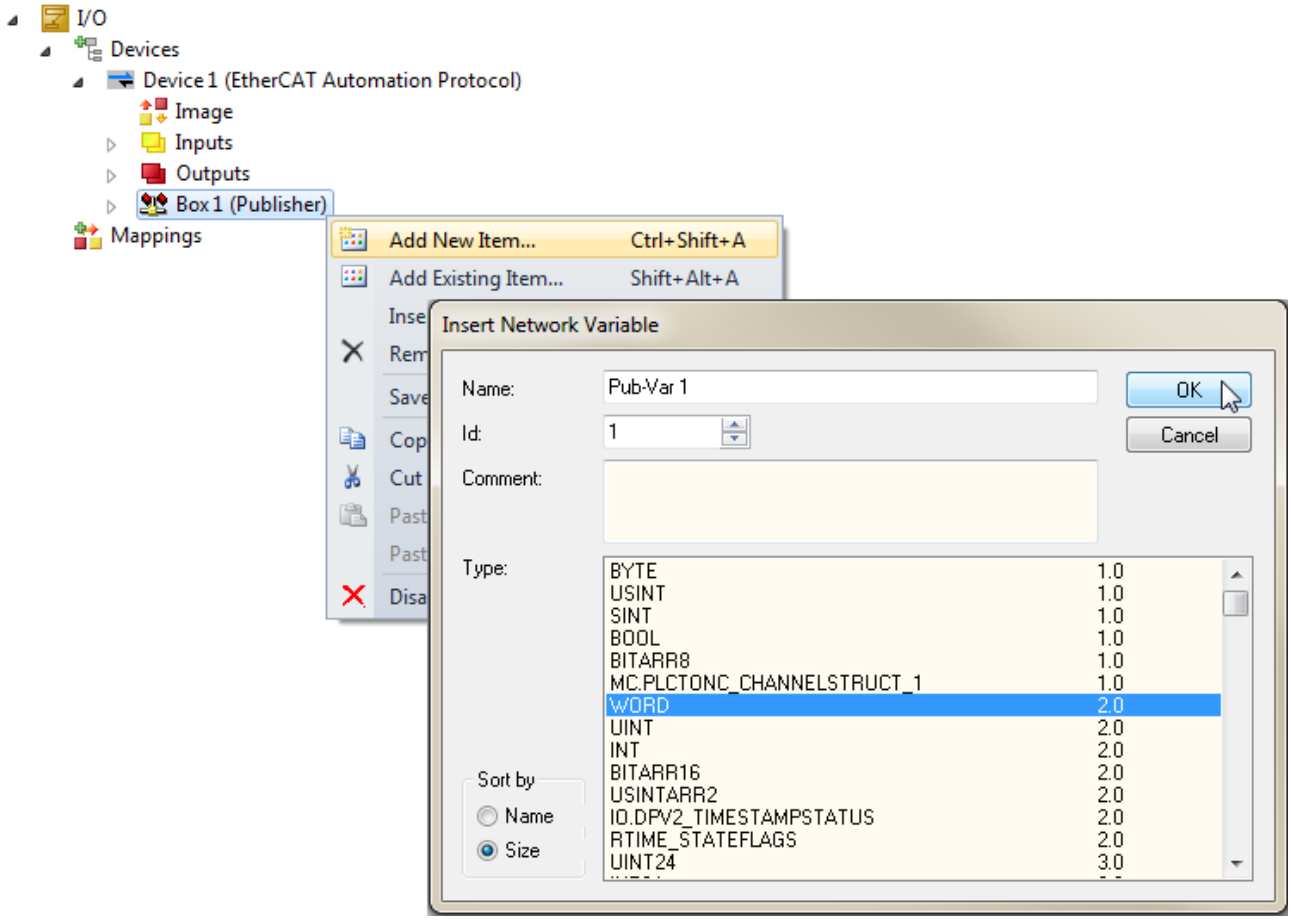
An dem erzeugten *Publisher* werden schließlich die *Publisher Variablen* (*TxData*) angehängt (siehe folgende Abbildung).

1. Klicken Sie im Kontextmenü des *Publishers* auf den Menüeintrag [*Add new Item...*].
   ⇨ Es öffnet sich der Dialog *Insert Network Variable*.
2. Wählen Sie einen Datentypen für die *Publisher Variable* aus, tragen Sie eine Bezeichnung bei *Name* ein, definieren Sie eine *ID* und klicken Sie dann auf [*OK*].

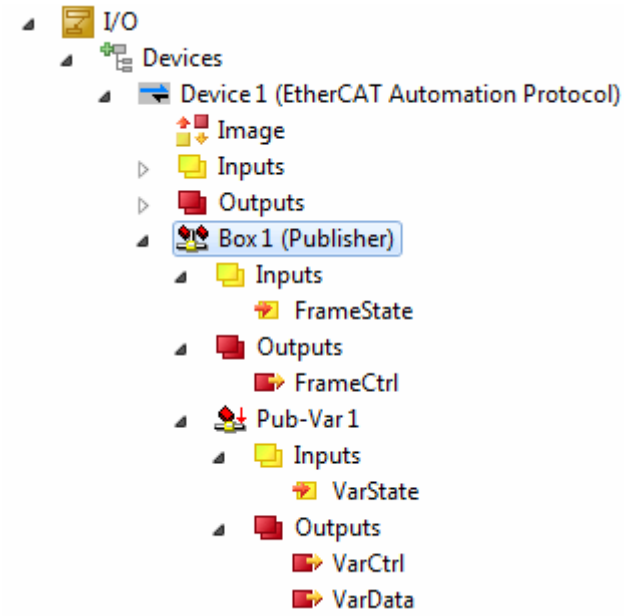
⇨ Es erscheint der Knoten für diese *Publisher Variable* unterhalb des *Publishers* (siehe nächste Abbildung).

Vergabe einer ID

Die gewählte *ID* für eine *Publisher Variable* sollte im Netzwerk eindeutig sein.



Im Zuge der Konfiguration von Publisher Variablen werden Input und Output Variablen erzeugt, die zu den verschiedenen Knoten und Subknoten des EAP-Gerätes gehören (vgl. folgende Abbildung). Dabei gibt es neben den unbedingten Input/Output Variablen noch bedingte, die nur dann erzeugt werden, wenn bestimmte Konfigurationseinstellungen vorgenommen werden.



Alle Input/Output Variablen zusammen bilden schließlich das Prozessabbild des EAP-Gerätes. Wenn Sie mit den Variablen einer Anwendung (z.B. ein SPS Programm) in TwinCAT verknüpft werden, kann diese dann Inhalte von dem Prozessabbild lesen oder Werte in das Prozessabbild schreiben. Dabei dienen die Output Variablen dazu Werte zu schreiben, die z.B. das Verhalten des EAP-Gerätes steuern. Input Variablen dienen dazu, deren Werte zu lesen, um z.B. Statusinformationen in dem SPS Programm auszuwerten und darauf zu reagieren. Auf diese Weise kann von einer SPS aus unmittelbar Einfluss auf das Verhalten des EAP-Gerätes genommen werden.

Erscheinungsbild der Input und Output Variablen

Je nachdem über welchen Port (Netzwerkadapter des PC oder eine Switch Port Klemme EL66xx) das EAP-Gerät kommuniziert, variiert der Aufbau des Prozessabbildes.

Beschreibung der unbedingten Input/Output Variablen

FrameState
Die Input Variable *FrameState* unterhalb des Knotens *Publisher* zeigt den aktuellen Zustand des *TxFrames*. Eine detaillierte Beschreibung ist in Abschnitt Publisher [▶ 25] zu finden.

FrameCtrl
Die Output Variable **FrameCtrl** unterhalb des *Publishers* kann verwendet werden, um das Senden des EAP-Telegramms zu steuern. Folgende Werte sind für das *FrameCtrl* möglich:

Kurzbeschreibung	Bit	Beschreibung
Disable sending	FC.0	Das Senden des Frames wird unterbrochen, wenn das Bit auf den Wert 1 gesetzt wird. Erst wenn der Wert auf 0 zurückfällt, beginnt wieder das Senden des Frames.
Remove destination MAC from ARP cache	FC.1	Dieses Control-Feld hat nur Auswirkungen, wenn das EAP-Telegramm per IP oder *AMSNetID* versendet wird. Die im ARP Cache eingetragene Ziel *MAC*-Adresse wird gelöscht, wenn das Bit gesetzt wird. Erst wenn das Bit wieder auf 0 zurückgesetzt wird, kann ein neuer Eintrag erfolgen, sofern das EAP-Gerät die Ziel *MAC*-Adresse per ARP ermitteln kann. (Siehe auch Gegenstellenüberwachung per ARP [▶ 13])

Erscheinungsbild der Output Variablen FrameCtrl

Bei einer EAP-Kommunikation über die Klemme EL66xx gibt es diese Variable nicht. Stattdessen wird die Output Variable *CycleIdx* (Größe 16 Bit) angelegt, die für eine Diagnose der Kommunikationsverbindung genutzt werden sollte (vgl. Diagnose einer EAP-Verbindung [▶ 22]).

VarState
Die Input Variable *VarState* unterhalb der *Publisher Variable* zeigt den aktuellen Zustand des *TxData.* Eine detaillierte Beschreibung ist in Abschnitt Publisher [▶ 25] zu finden.

VarCtrl
Die Output Variable *VarCtrl* unterhalb der *Publisher Variable* kann verwendet werden, um das Senden der *Publisher Variable* zu steuern. Folgende Werte sind für das *VarCtrl* möglich:

Kurzbeschreibung	Bit	Beschreibung
Disable publishing	VC.0	Das Senden der *Publisher Variable* wird unterbrochen. Erst wenn der Wert des Bits wieder auf 0 ist, beginnt wieder das Senden der *Publisher Variable*.

Erscheinungsbild der Output Variablen VarCtrl

Bei einer EAP-Kommunikation über die Klemme EL66xx gibt es diese Variable nicht!

CycleIndex
Die Output-Variable *CycleIndex* unterhalb des *TxProcessData* sollte mit einer Applikationsvariable verknüpft werden, die zyklisch inkrementiert wird, da der *CycleIndex* auf der Empfängerseite für Diagnosezwecke ausgewertet werden kann (siehe Diagnose einer EAP-Verbindung [▶ 22]).

Erscheinungsbild der Output-Variable CycleIndex

Die Output Variable CycleIndex ist nur dann vorhanden, wenn die EAP-Kommunikation über die Klemme EL66xx erfolgt!
Wenn die EAP-Kommunikation über einen Netzwerkadapter des PCs erfolgt, gibt es diese Variable nicht. Dann wird der *CycleIndex* im EAP-Telegramm automatisch mit jedem Taskzyklus inkrementiert.

VarData
Die Output Variable *VarData* der *Publisher Variable* kann mit jeder beliebigen Variablen des passenden Datentyps verknüpft werden (z.B. mit der Variablen eines SPS Programms).

Beschreibung der bedingten Output Variablen

MAC / NetID / IP
Die Output Variable *MAC / NetID / IP* unterhalb des Knotens *Publisher Box* ist nur vorhanden, wenn die Option *Target Address Online Changeable* aktiviert ist. In diesem Fall kann die Zieladresse des EAP-Telegramms dynamisch (z.B. mit Hilfe eines SPS Programms) geändert werden.
Je nachdem, welche Art der Adressierung beim *TxFrame* konfiguriert wird (MAC Address, AMS NetID oder IP), ist der Datentyp der Variablen eine MAC, eine NetID oder eine IP.

VarId
Die Output Variable *VarId* unterhalb der *Publisher Variable* ist nur vorhanden, wenn die Option *Online Changeable* bei der Variable *ID* aktiviert ist. In diesem Fall kann die *ID* für das TxData dynamisch (z.B. mit Hilfe eines SPS Programms) geändert werden.

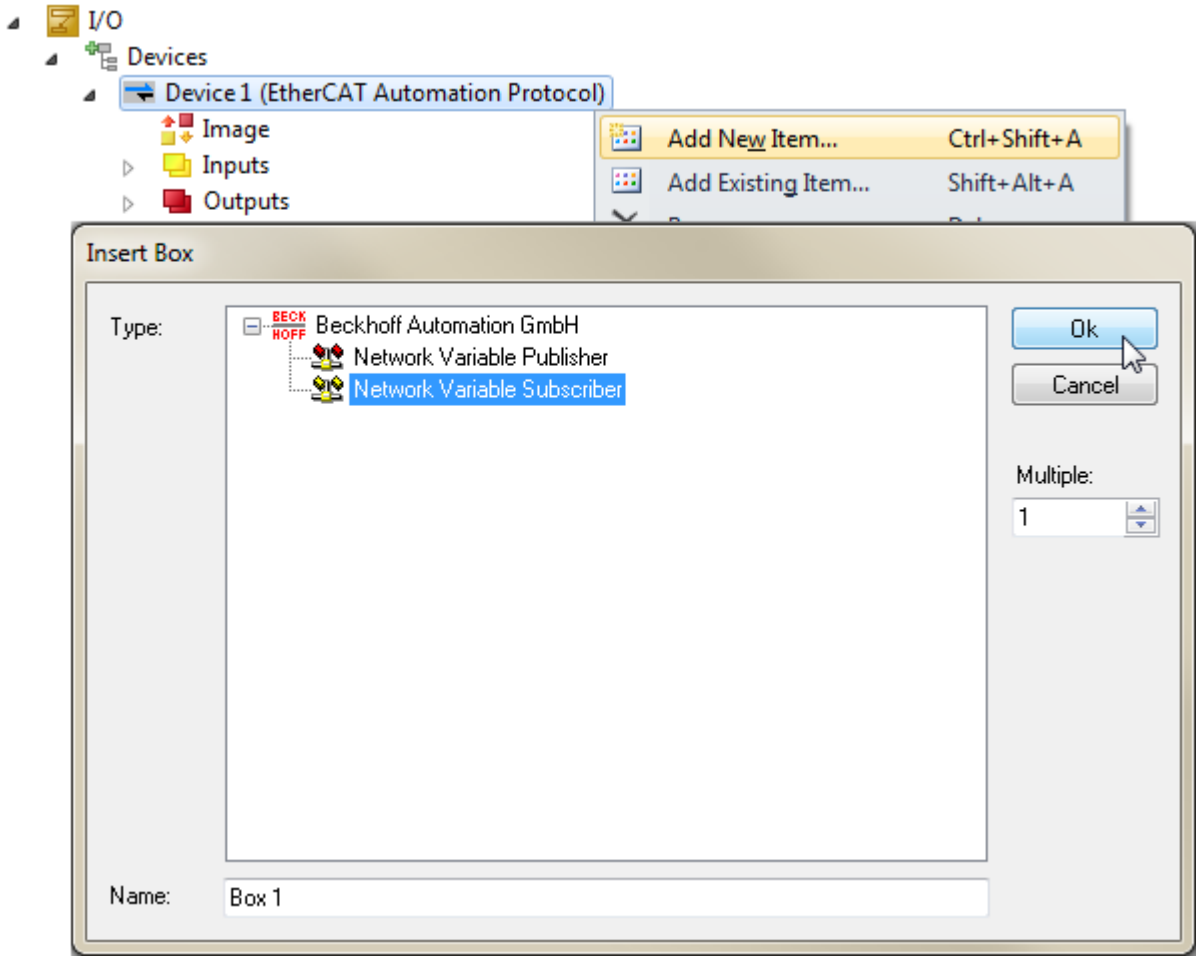
4.3 Hinzufügen von Subscriber Variablen

Wenn das EAP-Geräte Variablen empfangen soll, müssen zu seiner vollständigen Konfiguration noch Subscriber Variablen hinzugefügt werden (vgl. nächste Abbildung).

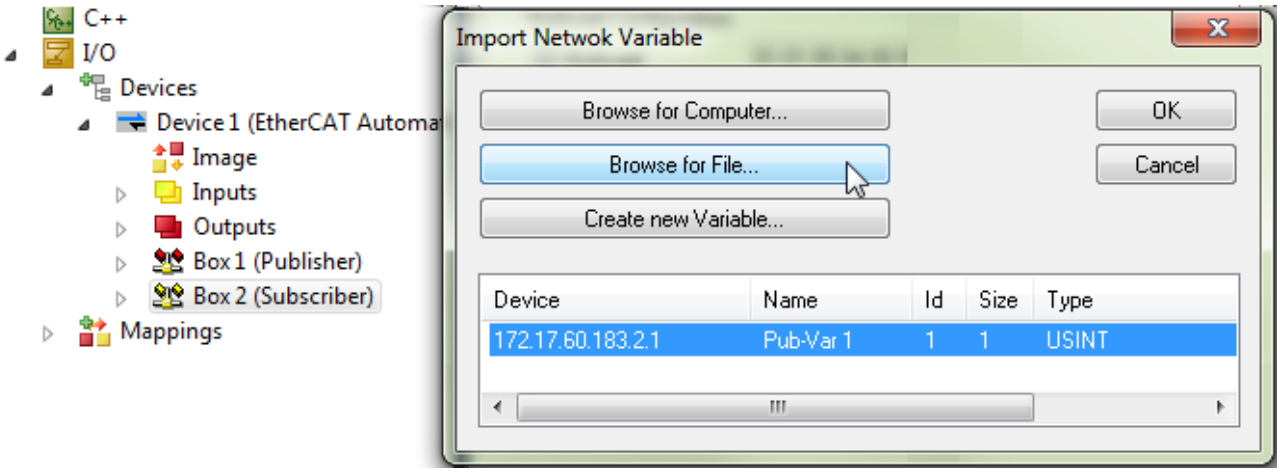
1. Klicken Sie im Kontextmenü des EAP-Geräteknotens (*Device 1*) auf den Eintrag [*Add New Item...].*
   ⇨ Es öffnet sich der Dialog *Insert Box.*
2. Wählen Sie *Network Variable Subscriber* aus und klicken Sie auf [*OK*].

Mehrere Subscriber hinzufügen
Wenn mehrere *Subscriber* angelegt werden sollen, kann die Anzahl mit Hilfe des Eingabefeldes *Multiple* eingestellt werden.

An dem erzeugten *Subscriber* werden schließlich die *Subscriber Variablen* (*RxData*) angehängt (siehe folgende Abbildung).



1. Klicken Sie im Kontextmenü des *Subscribers* auf den Menüeintrag [*Add new Item...*].
   ⇨ Es öffnet sich der Dialog *Import Network Variable* mit Hilfe dessen eine *Subscriber Variable* importiert oder definiert wird. Dafür stehen drei Möglichkeiten zur Auswahl.



Browse for Computer
Die Verbindung zu einer *Publisher Variablen* einer anderen Steuerung im Netzwerk kann automatisch hergestellt werden.

✓ Die andere Steuerung muss in der Liste der bekannten Zielsysteme zu finden sein.

1. Klicken Sie auf [*Browse for Computer…*].
   ⇨ Es öffnet sich der Dialog *Choose Target System*.
2. Wählen Sie den gewünschten Steuerungsrechner aus der Liste aus und klicken Sie auf [*OK*].
   ⇨ Es werden alle *Publisher Variablen* aufgelistet, die dieser Rechner anbietet.
3. Selektieren Sie die gewünschte *Publisher Variable* und bestätigen Sie mit [*OK*].

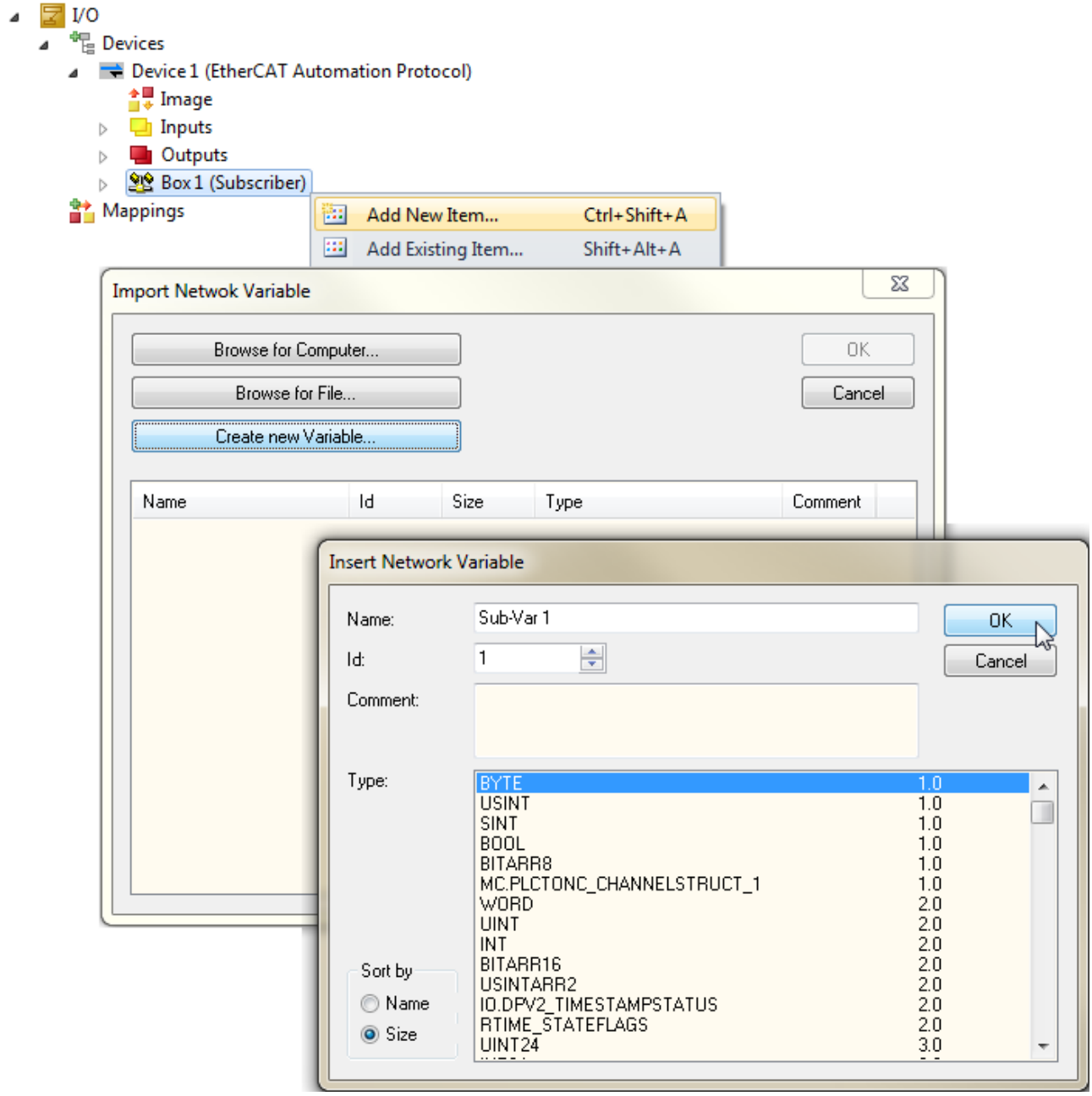
⇨ Die *Subscriber Variable* wird passend zur gewählten *Publisher Variable* angelegt.

Browse for File
Alternativ zu *Browse for Computer* kann die Verbindung zu einer Variablen eines anderen Steuerungsrechner auch automatisch hergestellt werden, indem nach der Projektdatei gesucht wird, die auf dem Steuerungsrechner aktiviert ist oder wird.

1. Klicken Sie auf [*Browse for File…*].
   ⇨ Es öffnet sich der Dialog, um im Dateisystem zu browsen.
2. Suchen Sie im Dateisystem nach der gewünschten Projektdatei und bestätigen Sie Ihre Auswahl mit [*OK*].
   ⇨ Es werden alle *Publisher Variablen* aufgelistet, die dieser Rechner anbietet.
3. Selektieren Sie die gewünschte *Publisher Variable* und bestätigen Sie mit [*OK*].

⇨ Die *Subscriber Variable* wird passend zur gewählten *Publisher Variable* angelegt.

Create new Variable
Als letzte Möglichkeit bleibt die Variante, die *Subscriber Variable* manuell zu konfigurieren (vgl. folgende Abbildung).



1. Klicken Sie auf [*Create new Variable*].

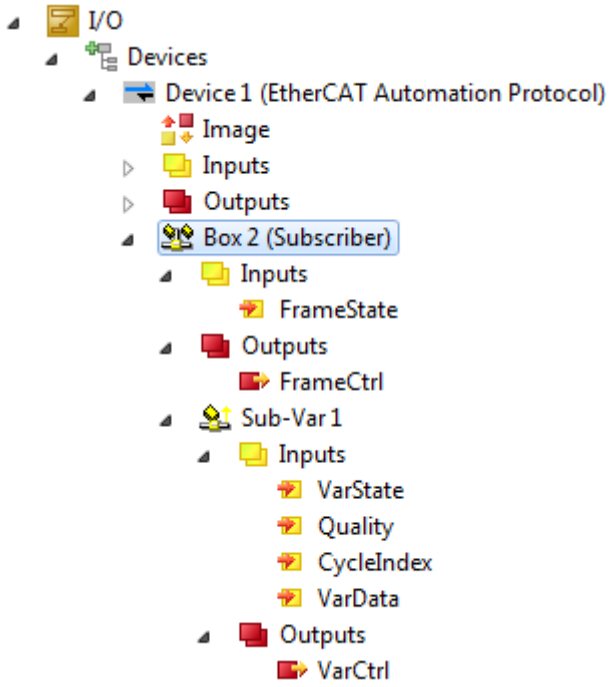
⇨ Es öffnet sich der Dialog *Insert Network Variable*.

2. Wählen Sie einen Datentypen für die *Subscriber Variable* aus, tragen Sie eine Bezeichnung bei *Name* ein, definieren Sie eine *ID* und klicken Sie dann auf [*OK*].

⇨ Es erscheint der Knoten für diese *Subscriber Variable* unterhalb des *Subscribers* (siehe folgende Abbildung).

Vergabe einer ID

Die gewählte *ID* für eine *Subscriber Variable* muss identisch zur *Publisher Variable ID* sein, die empfangen werden soll.



Beschreibung der unbedingten Input und Output Variablen

FrameState/FrameCtrl
Die Input Variable *FrameState* und die Output Variable *FrameCtrl* unterhalb des Knotens *Subscriber* sind reserviert und werden momentan nicht genutzt.

VarState
Die Input Variable *VarState* unterhalb der *Subscriber Variable* zeigt den aktuellen Zustand des *RxData*. Eine detaillierte Beschreibung ist in Abschnitt Subscriber [▶ 22] zu finden.

VarCtrl
Die Output Variable *VarCtrl* unterhalb der *Subscriber Variable* kann verwendet werden, um das Empfangen zu steuern. Folgende Werte sind für das *VarCtrl* möglich:

Kurzbeschreibung	Bit	Beschreibung
Ignore Hash/Version	VC.0	Wenn das Bit auf den Wert 1 gesetzt wird, wird die Überprüfung der Version beim Empfangen eines *Process Data* abgeschaltet.

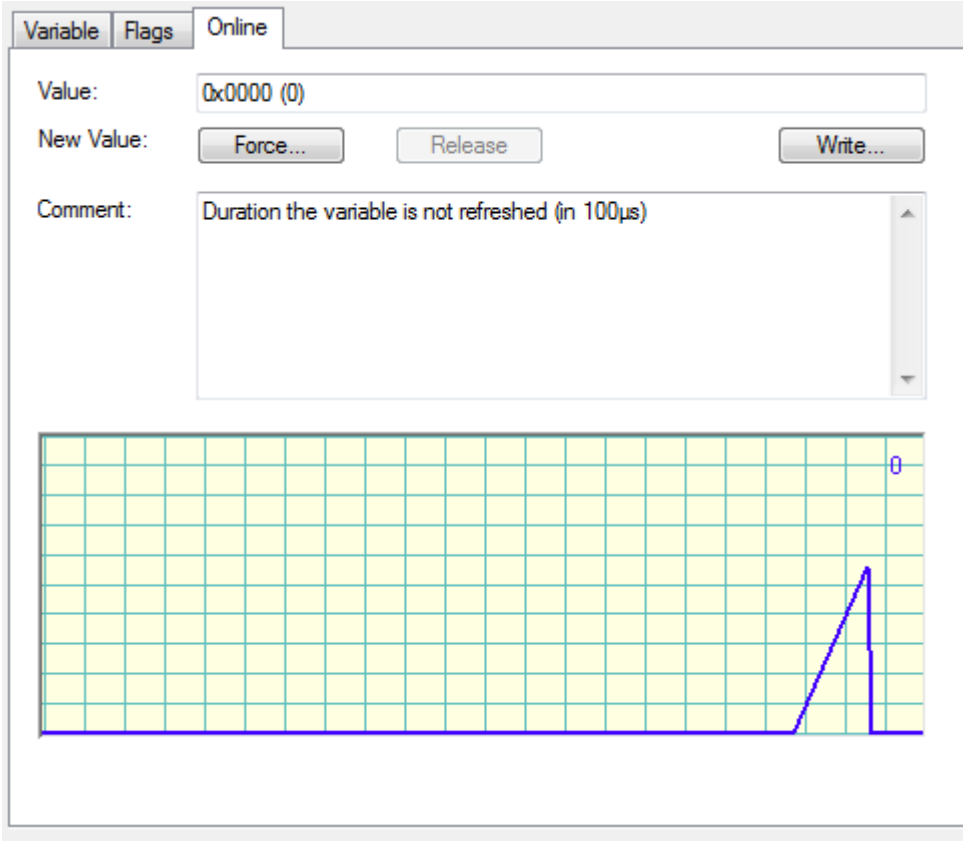
Erscheinungsbild der Output Variablen VarCtrl

Bei einer EAP-Kommunikation über die Klemme EL66xx gibt es diese Variable nicht!

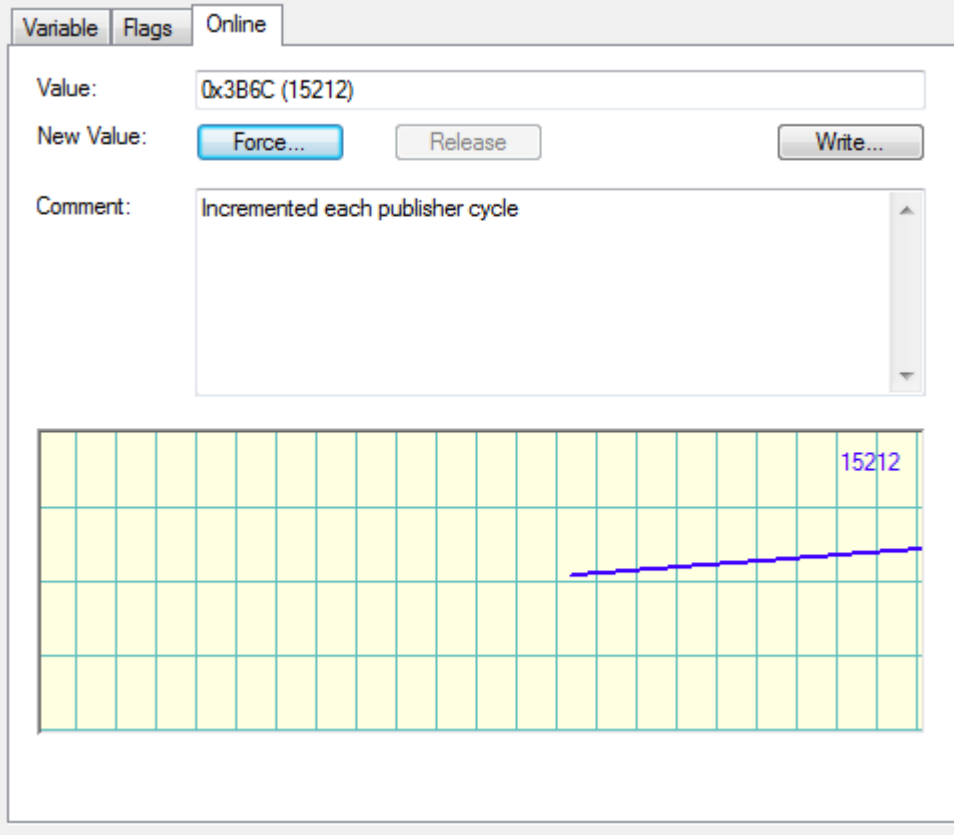
VarData
Die Input Variable *VarData* der *Subscriber Variable* kann mit jeder beliebigen Variablen des passenden Datentyps in TwinCAT verknüpft werden (z.B. mit einer Variablen eines SPS Programms).

Der Empfang einer Variablen wird auf der Empfängerseite diagnostiziert. Hierzu stehen die zwei Input Variablen *Quality* und *CycleIndex* unterhalb der *Subscriber Variablen* zur Verfügung.

Quality
Die Variable *Quality* liefert einen Zähler in 100µs Auflösung. Der Zählerwert zeigt an, wie lange keine Daten mehr für diese *Subscriber Variable* empfangen worden sind. Das Beispiel in der nächsten Abbildung zeigt den online Wert der *Quality* Variablen, nach Ziehen des Netzwerksteckers (Anstieg des Zählers) und erneutem Verbinden (Zählerstand 0).



CycleIndex
Die *CycleIndex* Variable wird bei jedem *Publisher* Zyklus inkrementiert. Das Beispiel in nächsten Abbildung zeigt den typischen Verlauf bei der Ansicht des online Wertes der *CycleIndex* Variablen.



Beschreibung der bedingten Output-Variablen

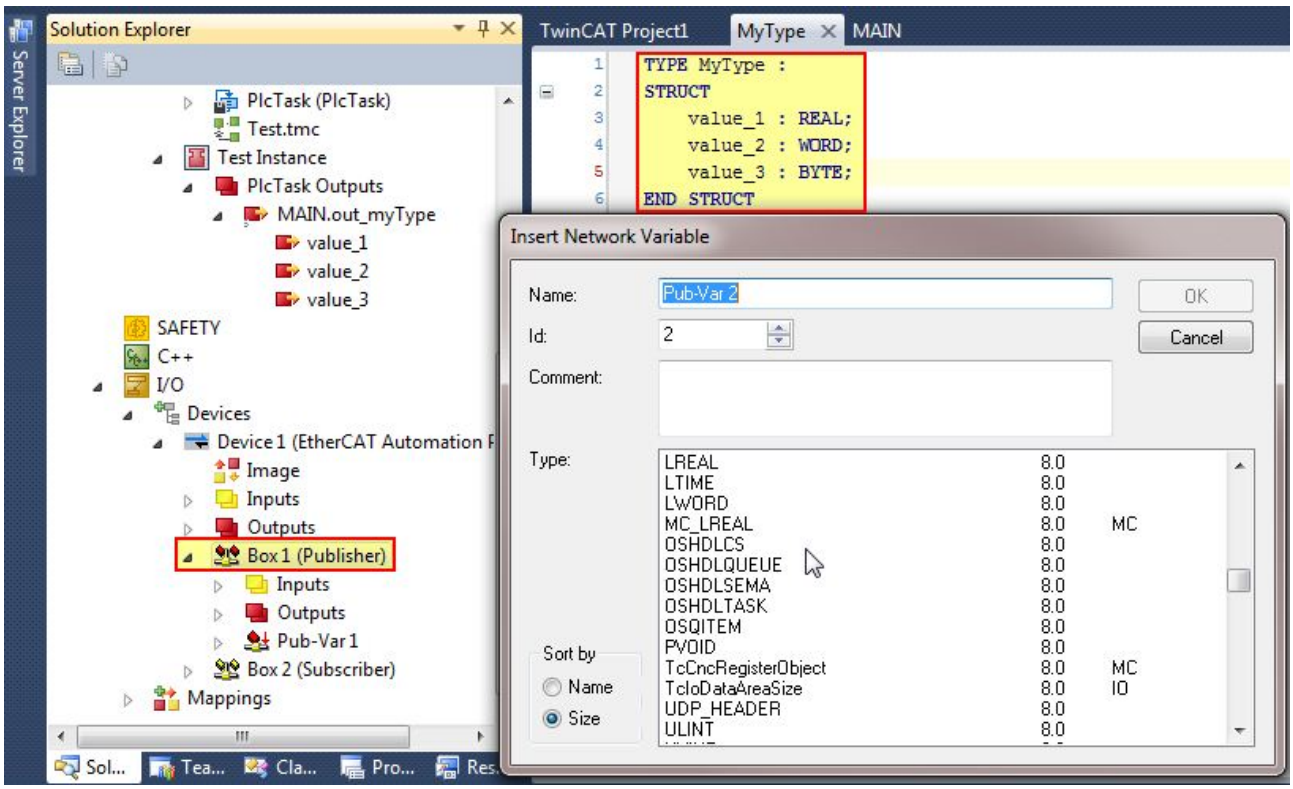
VarId
Die Output Variable **VarId** unterhalb der *Subscriber Variable* ist nur vorhanden, wenn die Option *Online Changeable* bei der *Variable ID* aktiviert ist. In diesem Fall kann die *ID* für die *Subscriber Variable* dynamisch (z.B. mit Hilfe eines SPS Programms) geändert werden.

4.4 Verwendung von benutzerdefinierten Datentypen

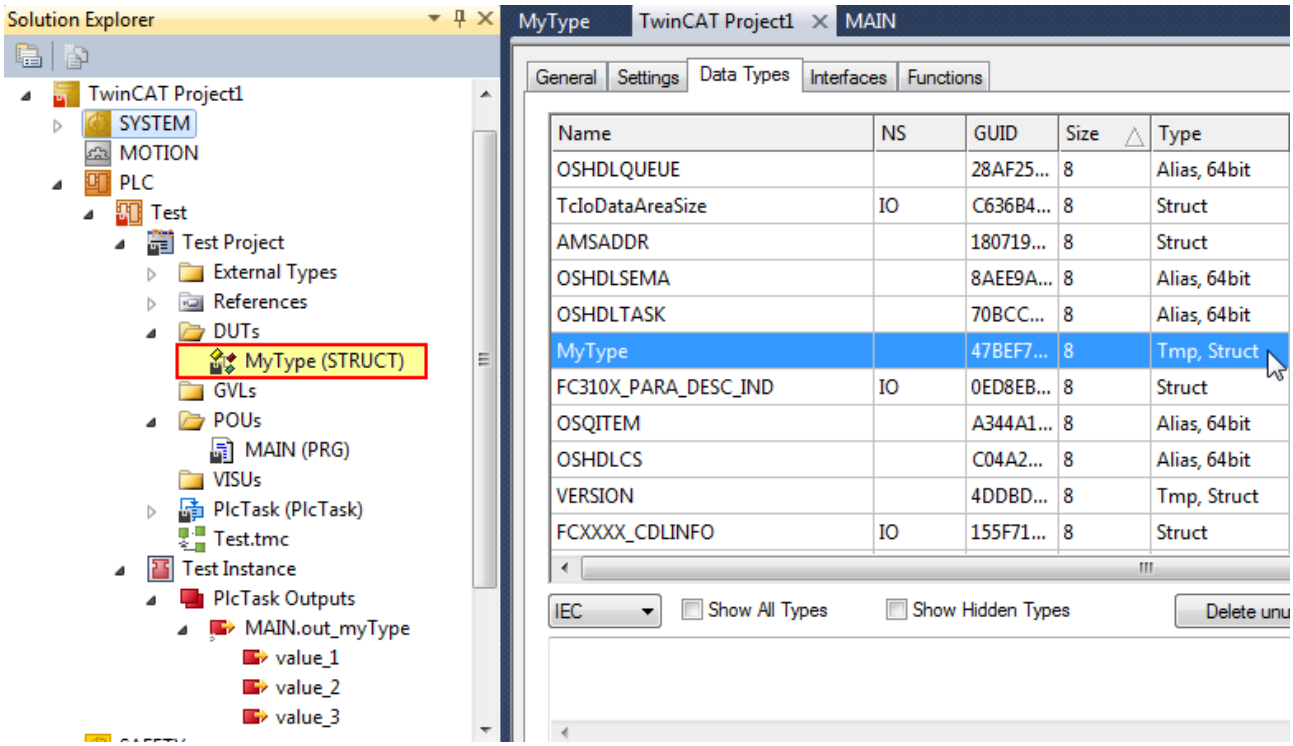
Anlegen einer EAP-Variable

In TwinCAT gibt es zwei gängige Möglichkeiten benutzerdefinierte Datentypen anzulegen. Zum einen kann ein eigener Datentyp über den *System* Knoten des Projektbaums auf dem Karteireiter *Data Types* angelegt werden (vgl. übernächste Abbildung). Ein solcher Datentyp steht dann allen Modulen des TwinCAT-Projektes zur Verfügung. Es handelt sich also um einen globalen Datentypen.
Zum anderen wird häufig ein eigener Datentyp innerhalb eines PLC Projektes angelegt, indem eine DUT (Data Type Unit) definiert wird. Ein solcher Datentyp steht dann zunächst erst mal nur dem PLC Projekt lokal zur Verfügung. Anderen Modulen, wie z.B. der I/O Konfiguration (und damit auch dem EAP-Gerät), bleibt dieser Datentyp verborgen.

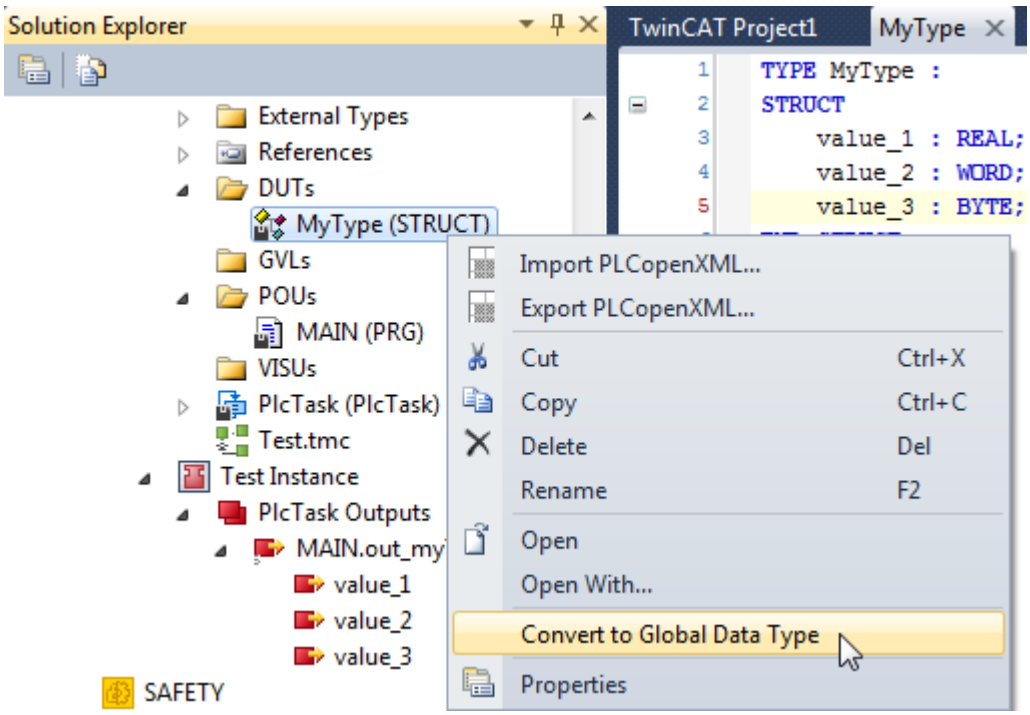
In folgender Abbildung ist zu sehen, dass der Datentyp *MyType* in der PLC definiert worden ist. Dieser Datentyp ist außerdem für eine Output Variable des SPS Programms verwendet worden. Dennoch taucht der benutzerdefinierte Datentyp nicht in der Liste der verfügbaren Datentypen auf, wenn bei der *Publisher Box* des EAP-Gerätes eine Variable von diesem Datentyp angelegt werden soll.



Ein entsprechender Hinweis dafür, dass ein Datentyp nur lokal verwendet wird, lässt sich anhand der Datentypenliste auf dem Karteireiter *Data Types* über den *System* Knoten finden. In folgender Abbildung wird der Datentyp mit dem Namen *MyType* in der Liste aufgeführt. Unter der Eigenschaft *Type* steht jedoch der Vermerk, dass es sich um einen temporären Datentypen handelt *(Tmp)*. Das bedeutet, dass es sich um einen Datentypen handelt, der nicht global ist.



Wenn nun die Notwendigkeit besteht, einen lokalen Datentypen aus der PLC im globalen Kontext zu verwenden, kann die entsprechende *DUT* zu einem globalen Datentypen konvertiert werden (vgl. folgende Abbildung).



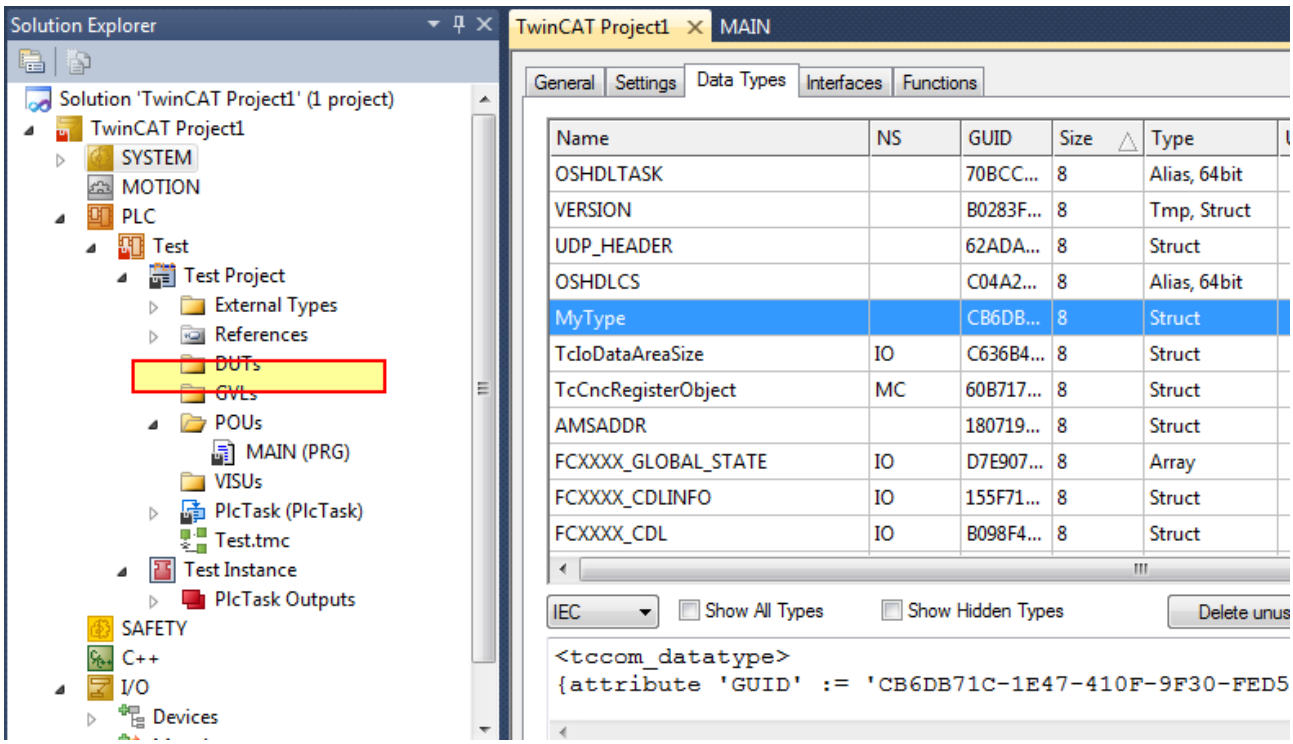
1. Klicken Sie im Kontextmenü des *DUT* auf das Kommando [*Convert to Global Data Type*].

⇨ Der Knoten des *DUT* wird automatisch aus dem PLC Projekt entfernt und eine globale Deklaration des Typen wird angelegt (vgl. folgende Abbildung).

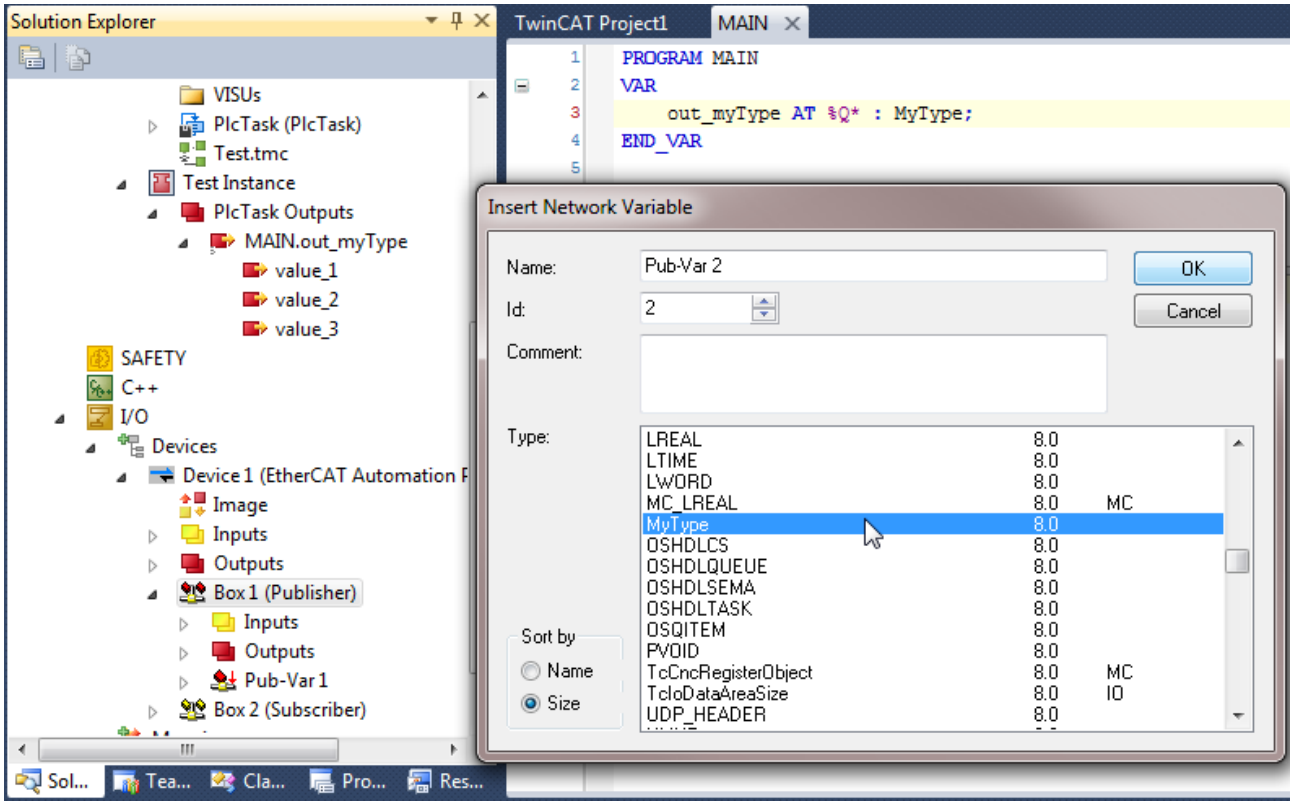
Die Beschreibung für ein und denselben Datentype muss in TwinCAT eindeutig sein. Aus diesem Grund wird die Definition des Datentypen (die ursprüngliche DUT) aus der PLC entfernt. Dafür ist nach der Konvertierung die Definition des Datentypen in der XML-basierten TwinCAT-Projektdatei wiederzufinden, und der Datentyp steht auf diese Weise dem gesamten TwinCAT-Projekt global zur Verfügung.

Ein Datentyp tritt als lokale und globale Variante auf

Nach der Konvertierung eines Datentypen zu einem globalen Datentypen, tritt dieser in der Liste der Datentypen zwei Mal auf. Nämlich sowohl als lokaler als auch als globaler Datentyp. Der lokale Datentyp wird erst dann aus der Liste entfernt, wenn dieser nicht mehr referenziert wird. Dazu ist es in der Regel notwendig, nach der Konvertierung das PLC Projekt erneut zu kompilieren. Dadurch wird die Referenz von dem SPS Programm auf den lokalen Datentypen gelöscht und nur noch der globale Datentyp referenziert. Entsprechend sollte der Datentyp nur noch einmal in der Liste auftauchen wie in der folgenden Abbildung.

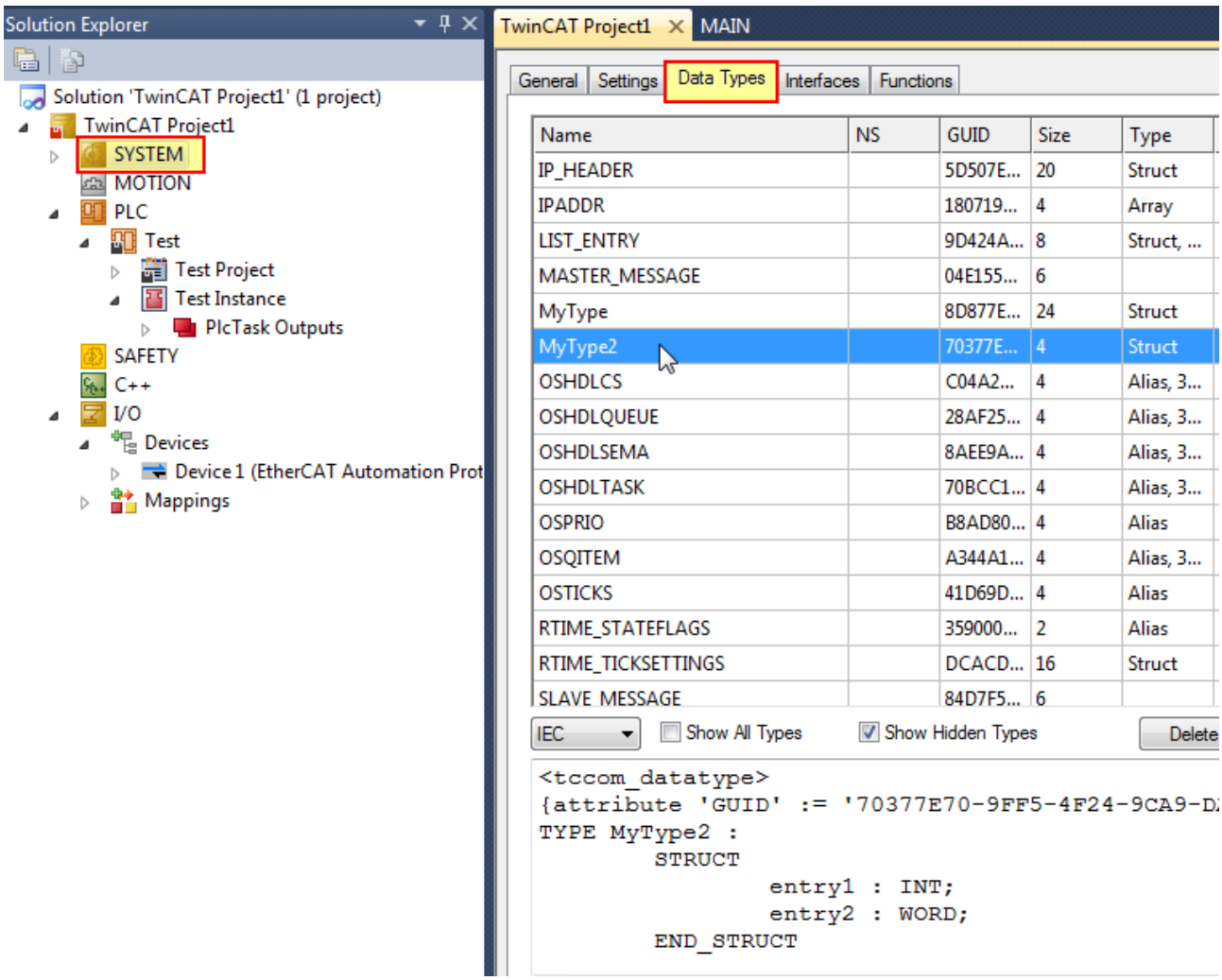


Nachdem der benutzerdefinierte Datentyp zu einem globalen Datentypen konvertiert wurde, kann dieser beim Anlegen einer *Publisher* oder *Subscriber Variable* beim EAP-Gerät aus der Liste der verfügbaren Datentypen ausgewählt werden (vgl. folgende Abbildung).

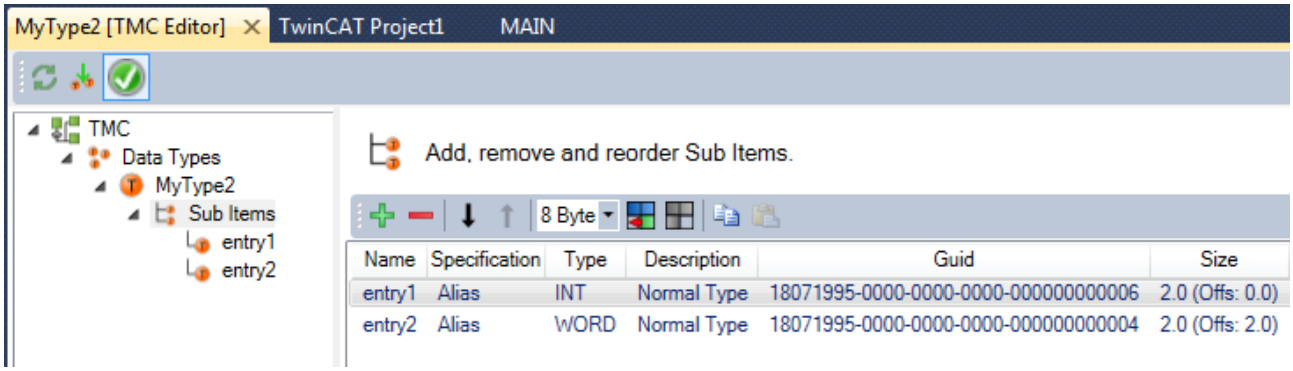


Ändern eines globalen Datentypen

Sobald ein Datentyp zu einem globalen Datentypen konvertiert worden ist, steht seine Definition nicht mehr unter der Kontrolle des PLC Projektes. Die Definition des Datentypen befindet sich in der XML-basierten TwinCAT-Projektdatei. Das Ändern dieses Datentypen muss nun im TwinCAT-Projekt über den Karteireiter *Data Types* des Knoten *System* vorgenommen werden (vgl. folgende Abbildung).



1. Suchen Sie den Datentypen aus der Liste heraus, den Sie ändern möchten.
2. Klicken Sie auf diesen mit der rechten Maustaste und wählen Sie im Kontextmenü das Kommando [Edit].
   ⇨ Es öffnet sich der *TMC Editor* (TMC = TwinCAT Module Configuration).



1. Mit Hilfe des *TMC Editors* können Sie den Datentypen beliebig verändern und anschließend speichern.

⇨ Die Änderung wird im TwinCAT-Projekt übernommen, indem nach dem Speichervorgang eine neue Version des Datentypen angelegt wird und die ursprüngliche Version im TwinCAT-Projekt als *Hidden* (dt. versteckt) markiert wird.

Ein PLC-Programm, welches den betreffenden Datentypen verwendet, benutzt immer automatisch die aktuellste Version des Datentypen. Damit die Änderung des Datentypen auch im Maschinencode übernommen wird, muss das PLC-Projekt nach der Änderung neu kompiliert werden.

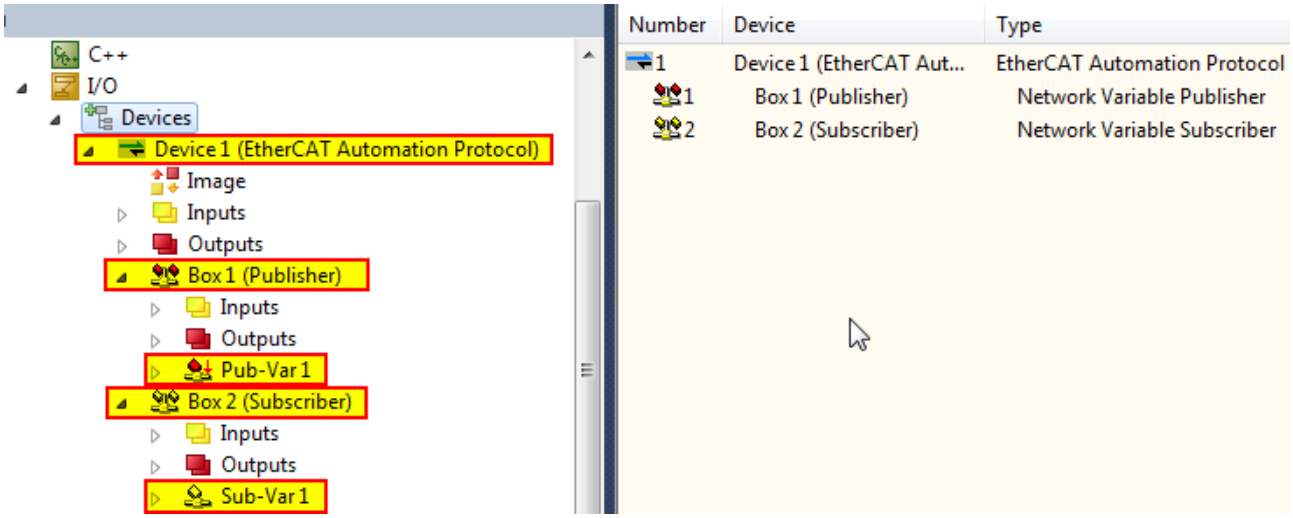
Wenn eine PLC Variable von diesem Datentypen existiert und diese mit einer entsprechenden *Publisher* oder *Subscriber Variablen* eines EAP-Gerätes verknüpft ist, wird auch bei der *Publisher/Subscriber Variablen* die aktuellste Version des Datentypen verwendet, sobald das PLC-Projekt neu kompiliert worden ist.

Existiert keine Verknüpfung von der EAP-Variablen zur PLC Variablen, verwendet das EAP-Gerät weiterhin die ursprüngliche, alte Version des Datentypen für seine Variable. Wenn stattdessen die neue Version verwendet werden soll, muss die betroffene EAP-Variable zunächst gelöscht und eine neue EAP-Variable vom gewünschten Datentypen wieder hinzugefügt werden. Die alte Version des Datentypen bleibt im TwinCAT-Projekt so lange erhalten, bis keine Referenz mehr zu dieser alten Version existiert.

5 Das Konfigurieren eines EAP-Gerätes

Ein per TwinCAT angelegtes EAP-Gerät, wie es in Kapitel Das Anlegen einer EAP-Konfiguration [▶ 26] beschrieben ist, ist zunächst einmal mit Standardeinstellungen konfiguriert. Die Standardeinstellungen sind so gewählt, dass der Anwender nur darauf achten muss, dass die Reihenfolge der Datenvariablen auf der Empfängerseite identisch zu der auf der Senderseite ist und dass auf der Sender- und Empfängerseite jeweils der gleiche Datentyp für die zu übertragene Datenvariable gewählt wird. Eine EAP-Verbindung, die auf diesem Wege konfiguriert worden ist, kommuniziert immer im *Pushed Data Exchange Modus* (siehe Kommunikationsmethoden [▶ 10]).

Mit Hilfe der Konfigurationsmöglichkeiten des EAP-Gerätes und der untergeordneten Boxen (*Publisher/Subscriber*, *Publisher/Subscriber Variable* – siehe folgende Abbildung) kann die EAP-Kommunikation frei konfiguriert werden. Des Weiteren ist es möglich, Informationen zur aktuellen Konfiguration eines aktivierten EAP-Gerätes auszulesen.



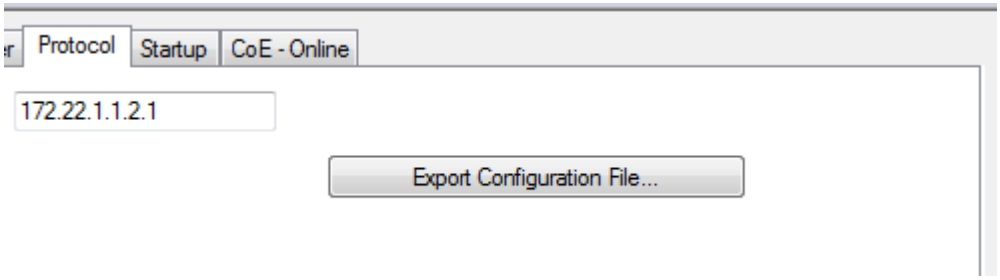
5.1 Das TwinCAT EAP-Gerät

Mit Hilfe der Konfigurationsmöglichkeiten beim EAP-Gerät wird festgelegt, über welchen Netzwerkadapter (engl. Network Interface Card - NIC) die EAP-Telegramme gesendet werden sollen und über welche *AMS NetID* das EAP-Gerät per *ADS/AMS* erreichbar sein soll. Über die Wahl des Netzwerkadapters ist dann auch automatisch festgelegt über welche IP Adresse (im Falle einer UDP/IP Kommunikation) das EAP-Gerät erreicht werden kann. Außerdem besteht die Möglichkeit, auf das Objektverzeichnis des EAP-Gerätes zuzugreifen.

General
Den Standard-Dialog unter dem Karteireiter *General* gibt es für alle TwinCAT-Geräte und Boxen. In diesem Dialog kann ein aussagekräftiger Name und ein nützlicher Kommentar zur Beschreibung des Gerätes oder der Box eingetragen werden.

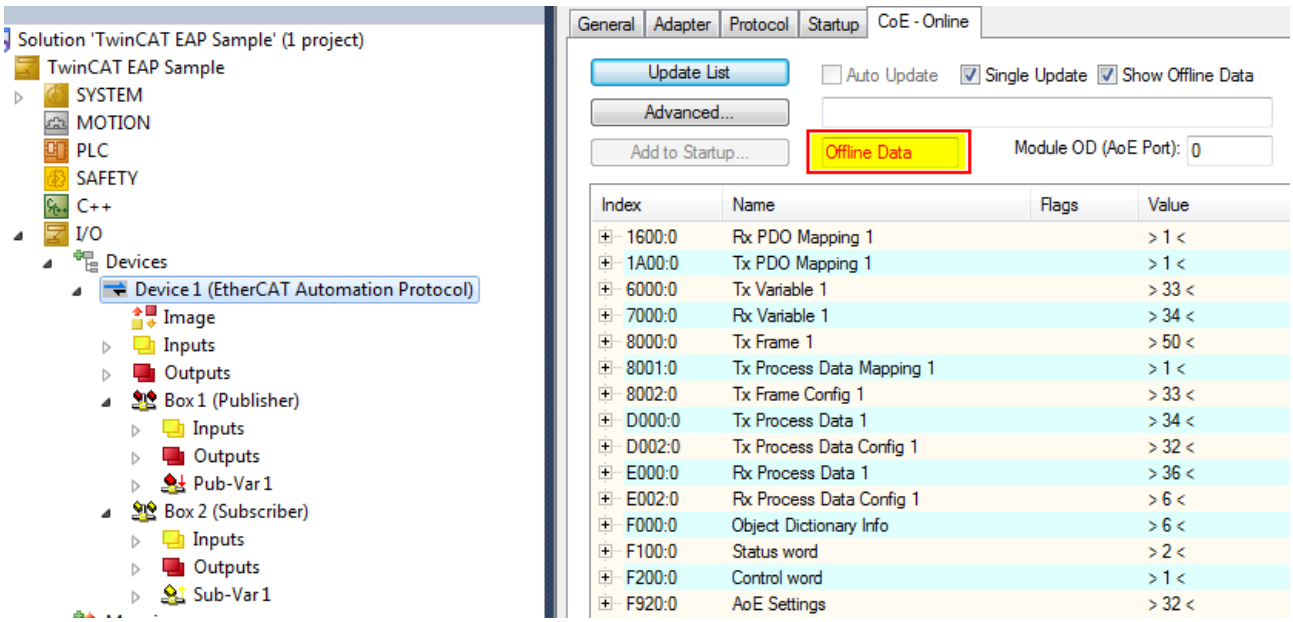
Adapter
Der Dialog unter dem Karteireiter *Adapter* zeigt den ausgewählten Netzwerkadapter an bzw. ermöglicht es einen Adapter zuzuordnen.

Protocol
Der Dialog unter dem Karteireiter *Protocol* (vgl. nächste Abbildung) ermöglicht die Vergabe einer speziellen *AMS NetID*, über die das EAP-Gerät per *ADS/AMS* während des Betriebs angesprochen werden kann. Weiterhin besteht die Möglichkeit mittels der Schaltfläche [*Export Configuration File…*] den aktuellen EAP-Konfigurationsstand des geladenen Projekts in eine XML-Datei zu exportieren. Diese XML-Datei hat ein festgelegtes Schema und wird auch *EAP Device Configuration* (*EDC*) File genannt.



CoE – Online

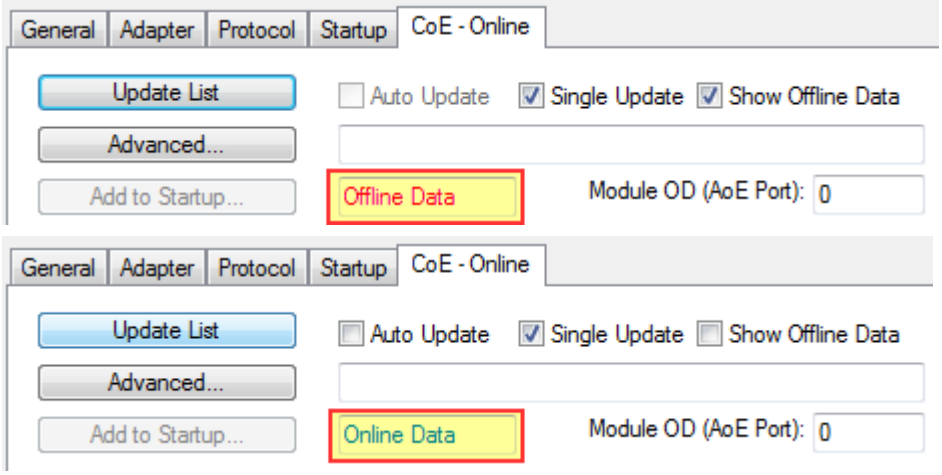
Der Dialog unter dem Karteireiter *CoE - Online* zeigt das EAP-Objektverzeichnis (engl. *Object Dictionary – OD*). Vergleichen Sie dazu auch Kapitel Das CANopen Objektverzeichnis [▶ 53] und die folgende Abbildung. Das Objektverzeichnis wird automatisch angelegt, sobald ein TwinCAT EAP-Gerät mit Hilfe von TwinCAT konfiguriert wird. Es beinhaltet alle Konfigurationsinformationen und wird automatisch um Einträge erweitert oder reduziert, sobald der aktuellen Konfiguration Elemente hinzugefügt oder aus dieser entfernt werden.



Die Kontrollelemente des CoE – Online Dialogs haben folgende Bedeutungen:

Status

Der Status des angezeigten Objektverzeichnisses wird in einem Textfeld ausgegeben (in der Abbildung oberhalb gelb hinterlegt). Der Status *Offline Data* wird immer dann angezeigt, wenn TwinCAT keine Verbindung zu einem aktivierten EAP-Gerät hat. Eine Verbindung kommt zum Beispiel dann nicht zustande, wenn sich die konfigurierte *AMS NetID* von der tatsächlichen *AMS NetID* des aktivierten EAP-Gerätes unterscheidet. Andernfalls wird der Status *Online Data* angezeigt:



Im Onlineverzeichnis	Im Offlineverzeichnis
wird das reale aktuelle Verzeichnis des EAP-Gerätes ausgelesen. Dies kann je nach Größe und Zykluszeit einige Sekunden dauern	wird das Offline-Verzeichnis des EAP-Gerätes angezeigt. Änderungen sind hier nicht sinnvoll bzw. möglich.
ist ein grünes **Online** im TwinCAT-Dialog *CoE-Online* zu sehen	ist ein rotes **Offline** im TwinCAT-Dialog *CoE-Online* zu sehen

Online Daten einer anderen EAP-Geräte Instanz auslesen

Es besteht die Möglichkeit die *AMS NetID* auf dem Karteireiter Protocol auf die *AMS NetID* eines beliebigen EAP-Gerätes innerhalb des Netzwerks zu setzen, um die Online Daten per TwinCAT über den Karteireiter *CoE – Online* auszulesen. Dazu muss das EAP-Gerät aktiviert sein und eine *ADS/AMS Route* zu dem Gerät bestehen.

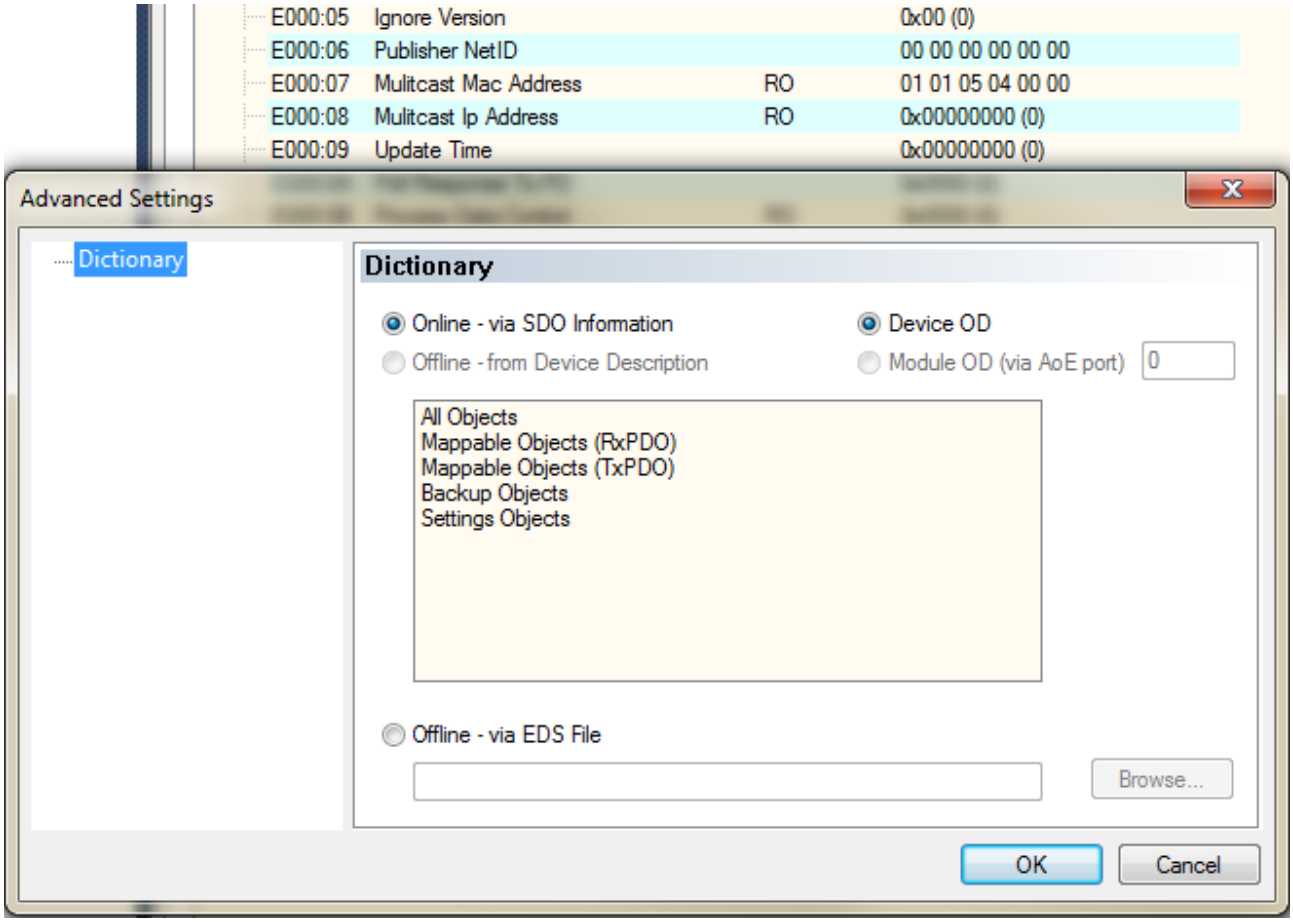
Show Offline Data
Mit Hilfe der Option *Show Offline Data* kann eingestellt werden, ob der Inhalt des Objektverzeichnisses online oder offline angezeigt werden soll. Online bedeutet, dass der *OD* Inhalt aus der aktivierten Konfiguration vom EAP-Gerät gelesen und angezeigt wird. Offline bedeutet, dass der *OD* Inhalt der Konfiguration angezeigt wird, die mit Hilfe von TwinCAT in dem aktuell geladenen TwinCAT-Projekt konfiguriert worden ist.

Single Update
Wenn die Option *Single Update* markiert ist, wird der *OD* Inhalt vom EAP-Gerät immer genau dann gelesen, wenn ein Objekt erweitert wird (Klicken auf das "+"-Symbol), um dessen Subeinträge anzuzeigen, oder wenn innerhalb des Fensters gescrollt wird.

Auto Update
Wenn die Option *Auto Update* markiert ist, wird der *OD* Inhalt für alle sichtbaren Subeinträge zyklisch vom EAP-Gerät gelesen und die Anzeige aktualisiert.

Update List
Die Schaltfläche *Update List* dient dazu, die aktuellen *OD* Inhalte aller sichtbaren Objekteinträge vom EAP-Gerät erneut zu lesen und anzuzeigen.

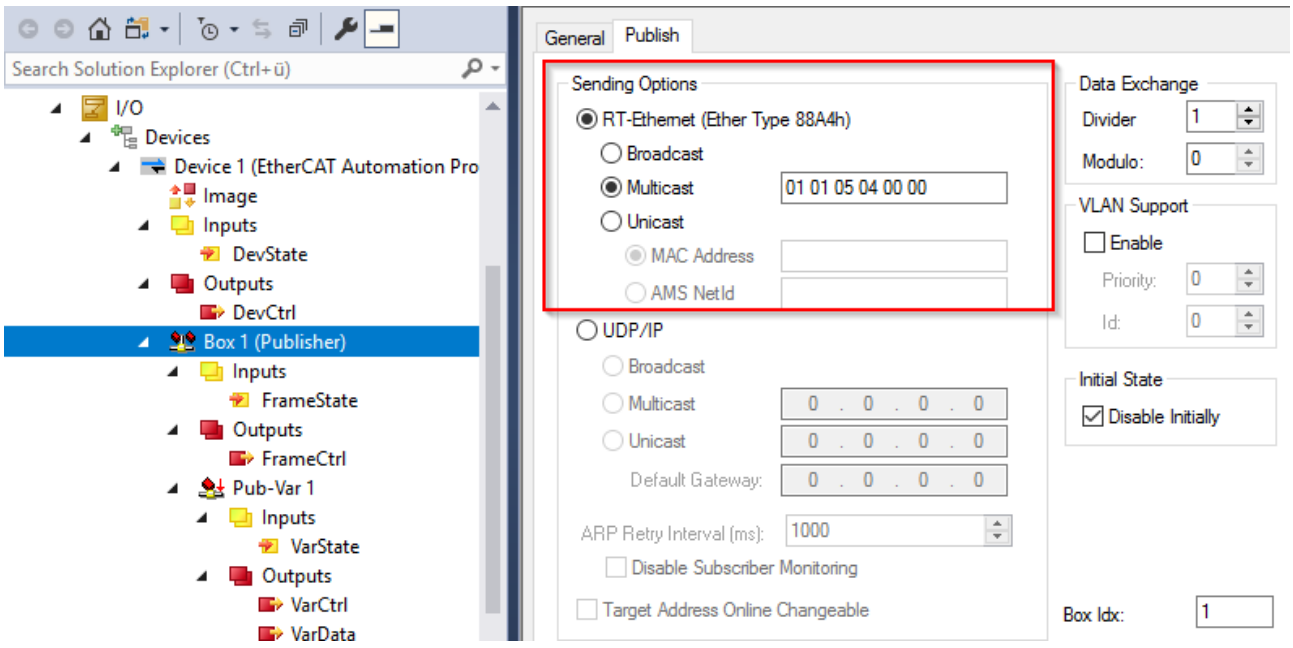
Advanced
Durch einen Klick auf die Schaltfläche *Advanced* öffnet sich der Dialog *Advanced Settings* (siehe folgende Abbildung). Mit Hilfe dieses Dialogs kann die gesamte oder ein Teil der *OD* Beschreibung vom EAP-Gerät gelesen werden. Diese Möglichkeit ist vor allem dann vorteilhaft, wenn während des Betriebs Objekte dem Objektverzeichnis hinzugefügt oder aus dem Objektverzeichnis entfernt worden sind. Denn dann ist die angezeigte *OD* Beschreibung in TwinCAT nicht mehr konsistent zum aktuellen *OD* des EAP-Gerätes.



Startup
TwinCAT generiert aus dem Objektverzeichnis einen Datenstrom, bestehend aus einer Reihe von *Startup*-Kommandos, die an das EAP-Gerät übertragen werden. Die Übertragung erfolgt, sobald die vorliegende Konfiguration aktiviert wird. Die generierten Startup-Kommandos werden auf dem Dialog unter dem Karteireiter *Startup* angezeigt.

5.2 Publisher Box

Auf der Konfigurationsseite der *Publisher Box* (vgl. folgende Abbildung) wird unter den *Sending Options* eingestellt, welches Basisprotokoll zum Versenden des EAP-Telegramms verwendet werden soll. Im Kapitel Grundlagen [▶ 8] werden im Abschnitt Vermittlungsprotokolle des Kapitels Kommunikationsmethoden [▶ 10] die beiden möglichen Basisprotokolle *Ethernet Protocol* und *User Datagramm Protocol* vorgestellt. Bei jedem der beiden Protokolle besteht die Möglichkeit, die drei unterschiedlichen Verbindungsarten *Broadcast*, *Multicast* oder *Unicast* zu konfigurieren. Bei den Verbindungsarten *Multicast* sowie *Unicast* muss zusätzlich noch eine Zieladresse festgelegt werden, mit Hilfe dessen der(die) Adressat(en) im Netzwerk erreicht werden kann(können).



Broadcast
Ein *Broadcast* Telegramm wird von einem Netzwerkteilnehmer an alle anderen Teilnehmer des Netzwerks übertragen. Jeder Empfänger einer *Broadcast* Nachricht entscheidet selbst, ob er die Nachricht verarbeitet oder nicht. Ein *Broadcast* auf der Ebene des *Ethernet Protocol* wird an die Ziel *MAC* Adresse FF:FF:FF:FF:FF:FF gesendet. Auf der Ebene des *UDP/IP* an die *IP* Adresse 255.255.255.255.

Multicast
Ein *Multicast* Telegramm wird von einem Netzwerkteilnehmer an eine ausgewählte Teilnehmergruppe des Netzwerks übertragen. Ein Empfänger einer *Multicast* Nachricht muss die *Multicast* Adresse, an die die Nachricht gesendet wird kennen und bei seinem Netzwerkadapter anmelden. Andernfalls verwirft der Netzwerkadapter die *Multicast* Nachricht.
Je nach verwendetem Basisprotokoll wird als Zieladresse entweder direkt eine *Multicast MAC* Adresse konfiguriert oder eine *Multicast IP* Adresse, die von TwinCAT zu einer *Multicast MAC* Adresse umgerechnet wird. Eine *Multicast IP* Adresse muss in dem Adressbereich 224.0.0.0 bis 239.255.255.255 (IPv4) liegen.

Unicast
Ein *Unicast* Telegramm wird von einem Netzwerkteilnehmer an genau einen anderen Netzwerkteilnehmer übertragen. Erfolgt die Adressierung auf Basis des *Ethernet Protocol*, wird als Zieladresse die *MAC* Adresse des Empfängers konfiguriert. Alternativ kann auch die *AMS NetID* des Empfängers konfiguriert werden. Wird das Telegramm auf Basis von *UDP/IP* versendet, wird als Zieladresse die *IP* Adresse des Empfängers konfiguriert (siehe folgende Abbildung).

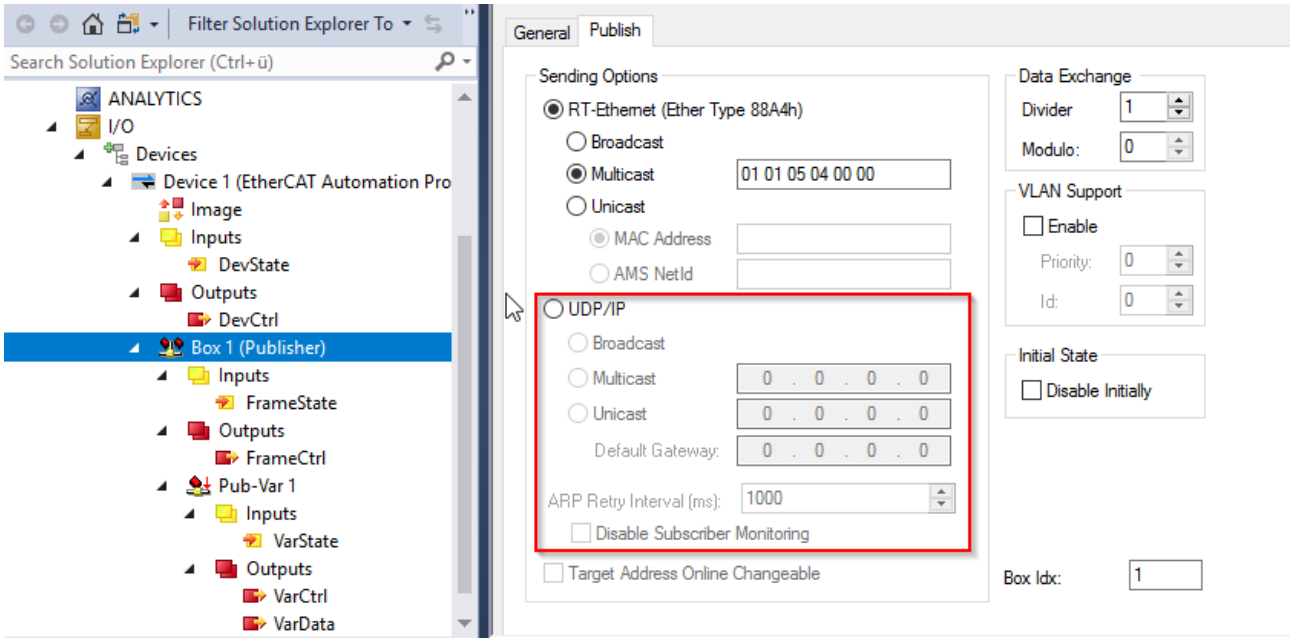
HINWEIS

Verwendung von Broadcast und Multicast

EAP-Telegramme, die als *Broad-* oder *Multicast* auf *MAC-* oder *IP*-Ebene verschickt werden, erzeugen je nach Zykluszeit eine hohe Netzwerklast, da sie an alle Netzwerkteilnehmer gesendet werden! Einfache Netzwerkgeräte wie z.B. Drucker können dann abstürzen, bei kurzen Zykluszeiten kann auch der gesamte Netzwerkverkehr blockiert werden!
Zur Vermeidung einer Netzwerküberlastung bzw. der Überlastung einfacher nicht echtzeitfähiger Netzwerkgeräte wird empfohlen

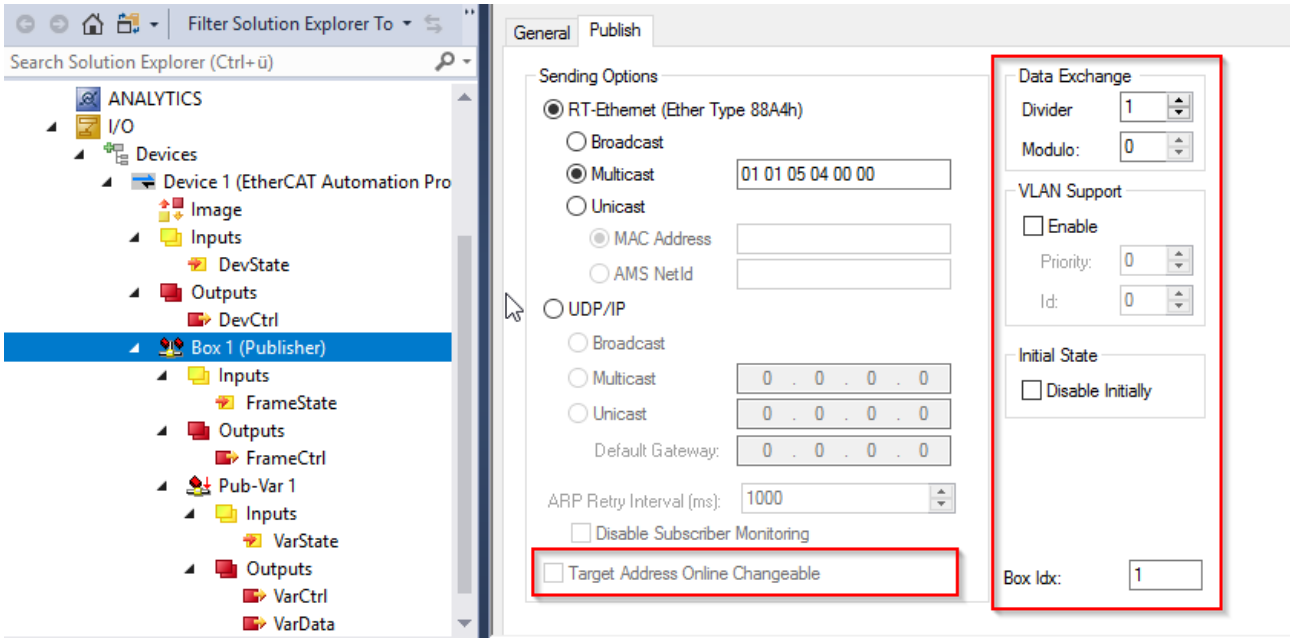
- zum einen die *Unicast* Adressierung zu verwenden,
- zum anderen die Zykluszeit immer nur so klein zu wählen, wie es unbedingt notwendig ist. Erläuterungen zur Einstellung der Zykluszeiten finden Sie weiter unten.

⇨ Wenn die Verbindungsart *Unicast* konfiguriert wird, ist standardmäßig auch der *Subscriber Monitoring* Mechanismus konfiguriert (siehe Gegenstellenüberwachung per ARP [▶ 13]).



Disable Subscriber Monitoring
Der *Subscriber Monitoring* Mechanismus kann mit Hilfe der Option *Disable Subscriber Monitoring* deaktiviert werden.

ARP Retry Interval
Die Zeit, die im Eingabefeld *ARP Retry Interval* eingestellt wird, legt fest in welchem zeitlichen Intervall in Millisekunden *(ms)*, eine Anfrage an den Empfänger gesendet wird, um seine Erreichbarkeit zu überprüfen.



Target Address Online Changeable
Die Option *Target Address Online Changeable* ist ebenfalls nur bei einem *Unicast* verwendbar. Wird diese Funktion aktiviert, existiert eine weitere Output Variable für den *Publisher* im Prozessabbild des EAP Gerätes. Diese Variable definiert je nach konfiguriertem Basisprotokoll eine *IP* Adresse, *MAC* Adresse oder auch eine *AMS NetID*. Die Output Variable kann mit Hilfe eines SPS Programms verändert werden. Auf diese Weise lässt sich die Zieladresse des konfigurierten *Publishers* dynamisch ändern (siehe auch bedingte Ausgänge in Abschnitt Hinzufügen von Publisher Variablen [▶ 27]).

Data Exchange
Mit Hilfe der Eigenschaft *Data Exchange* kann der Rhythmus verändert werden, in dem das EAP Telegramm versendet wird (vgl. Der EAP-Sendemechanismus [▶ 13]).

Data Exchange

Die Eigenschaft *Data Exchange* kann nicht bei der Verwendung einer EL66xx angewendet werden.

VLAN Support

Mit Hilfe der Eigenschaft *VLAN Support* kann in Verbindung mit Managed Switches dem EAP Telegramm durch das *VLAN* (*Virtual Local Area Network*) eine feste Route vorgegeben werden. Bei eingeschaltetem *VLAN* wird die EAP Nachricht mit einem *VLAN Header* versehen. Entsprechend gibt es zwei Eigenschaften, um das gewünschte *VLAN* zu bestimmen und eine *Priorität* für die Verarbeitung der Nachricht innerhalb des virtuellen Netzwerks festzulegen:

- *VLAN Info ID*: Definiert die *ID* des *VLANs* (Wertebereich von 0 bis 4095), in welches die Nachricht geschickt werden soll und
- *VLAN Info Priority*: Definiert eine *Priorität* für die Nachricht im *VLAN* (hoch Prior = 7, nieder Prior = 0).

Initial State

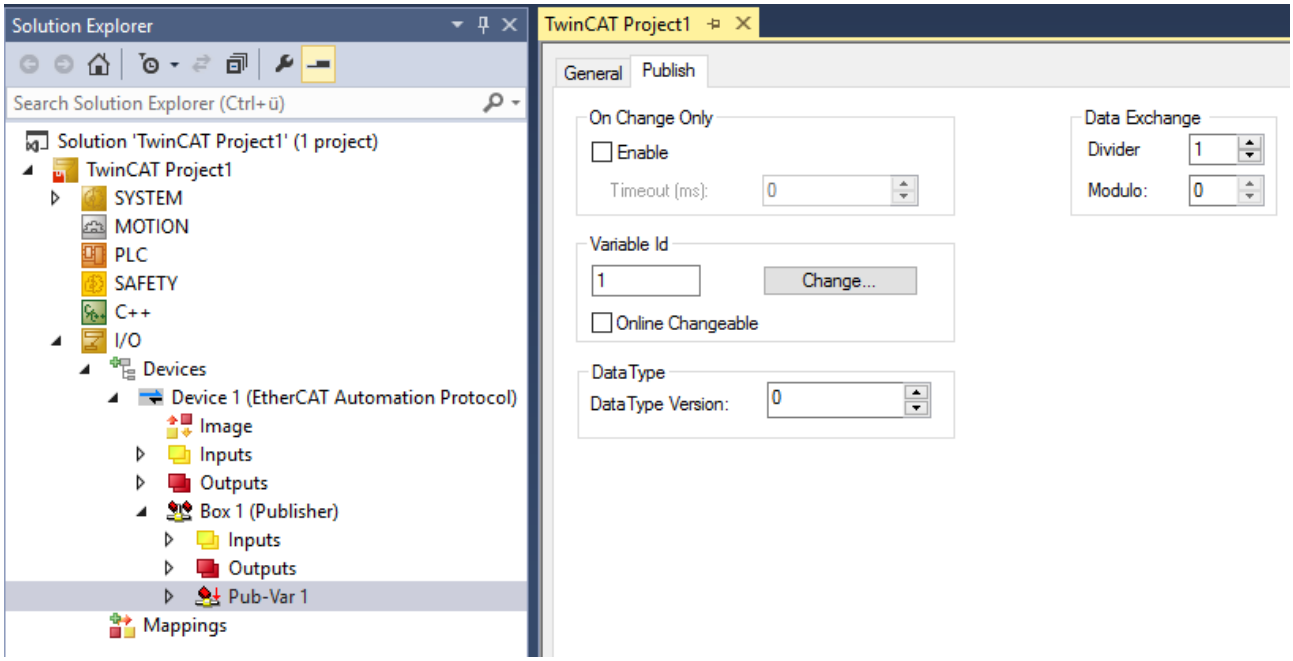
Mit Hilfe der Eigenschaft „Disable Initially“ kann unterbunden werden, dass nach dem Aufstarten des Systems der Publisher Pakete versendet werden. Das Versenden kann nachträglich eingeschaltet werden indem der FrameState auf den Wert 0 gesetzt wird.

Box Idx:

Fortlaufende Nummer für die unterschiedlichen Publisher und Subscriber, nur lesbar.

5.3 Publisher Variable

Jede *Publisher Variable* selbst hat zusätzlich spezielle Eigenschaften, die mit Hilfe einer eigenen Konfigurationsseite parametriert werden (vgl. auch folgende Abbildung).



On Change Only

Durch Aktivieren der Option *On Change Only* wird erreicht, dass das *TxProcessData* nur dann mit dem *TxFrame* versendet wird, wenn sich der Wert der *Publisher Variablen* verändert hat. Der Wert im Feld *Timeout* legt fest, nach wie vielen Millisekunden (ms) eine *Publisher Variable* seit dem letzten Versenden erneut versendet wird, auch wenn sich der Wert der *Publisher Variablen* in der Zwischenzeit nicht geändert hat (Weitere Details unter Der EAP-Sendemechanismus [▶ 13]).

On Change Only

Die Eigenschaft On Change Only kann nicht bei der Verwendung einer EL66xx angewendet werden.

Variable Id
Die *Variable Id (=ProcessData ID)* ist die Identifizierungsnummer der *Publisher Variablen*. Sie kann mit Hilfe eines SPS Programms online verändert werden, wenn die Option *Online Changeable* markiert ist.

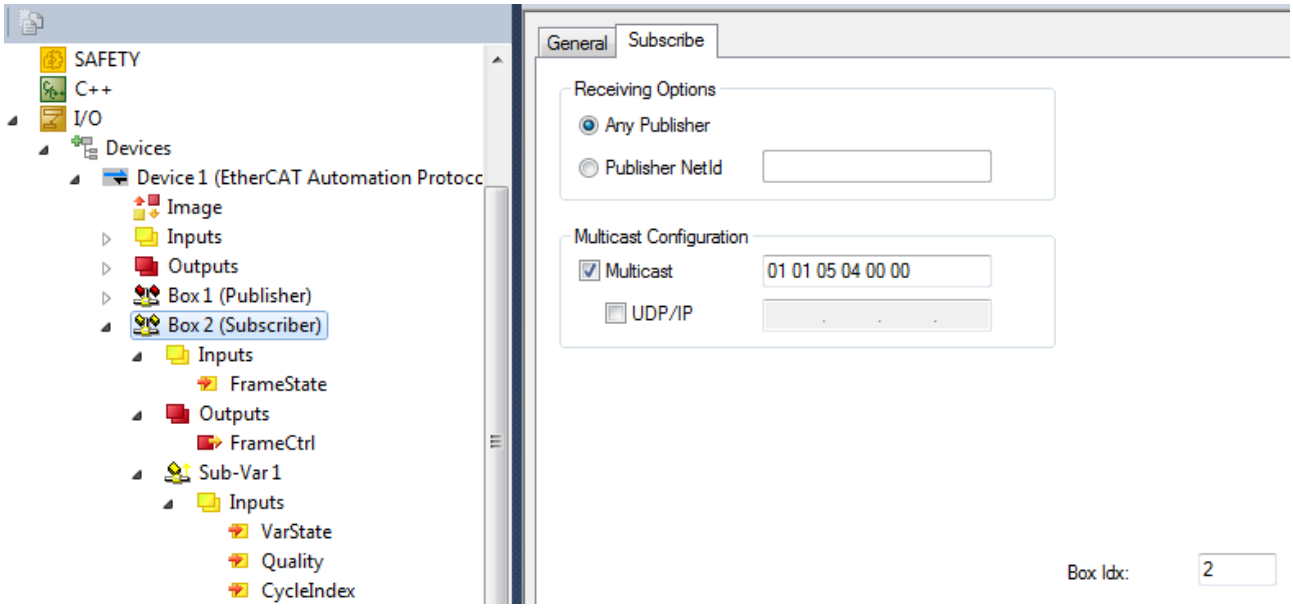
DataType Version
Hier kann eine Versionsnummer angegeben werden. Auf der Subscriberseite muss dieselbe Versionsnummer konfiguriert sein, sofern die beiden Versionsnummern beim Empfangen der Variable auf Gleichheit überprüft werden (standardmäßig findet dieser Vergleich statt). Die Versionsnummer wird verwendet um sicherzustellen, dass der verwendete Datentyp von Publisher Variable und zugehöriger Subscriber Variable identisch ist. Wenn der Datentyp nur auf Publisher- oder Subscriberseite geändert wird, muss die Versionsnummer erhöht werden, um zu verhindern, dass die Variable empfangen wird und dann deren Werte mit dem falschen Datentyp interpretiert werden.

Data Exchange
Mit Hilfe der Eigenschaft *Data Exchange* kann der Rhythmus verändert werden, mit der die *Publisher Variable* versendet wird (Weitere Details unter Der EAP-Sendemechanismus [▶ 13]).

Data Exchange
Die Eigenschaft *Data Exchange* kann nicht bei der Verwendung einer EL66xx angewendet werden.

5.4 Subscriber Box

Auf der Konfigurationsseite eines *Subscribers* werden die Eigenschaften definiert, die sich generell auf den Empfang eines EAP-Telegramms beziehen (vgl. nächste Abbildung). Zu diesen Eigenschaften gehören:



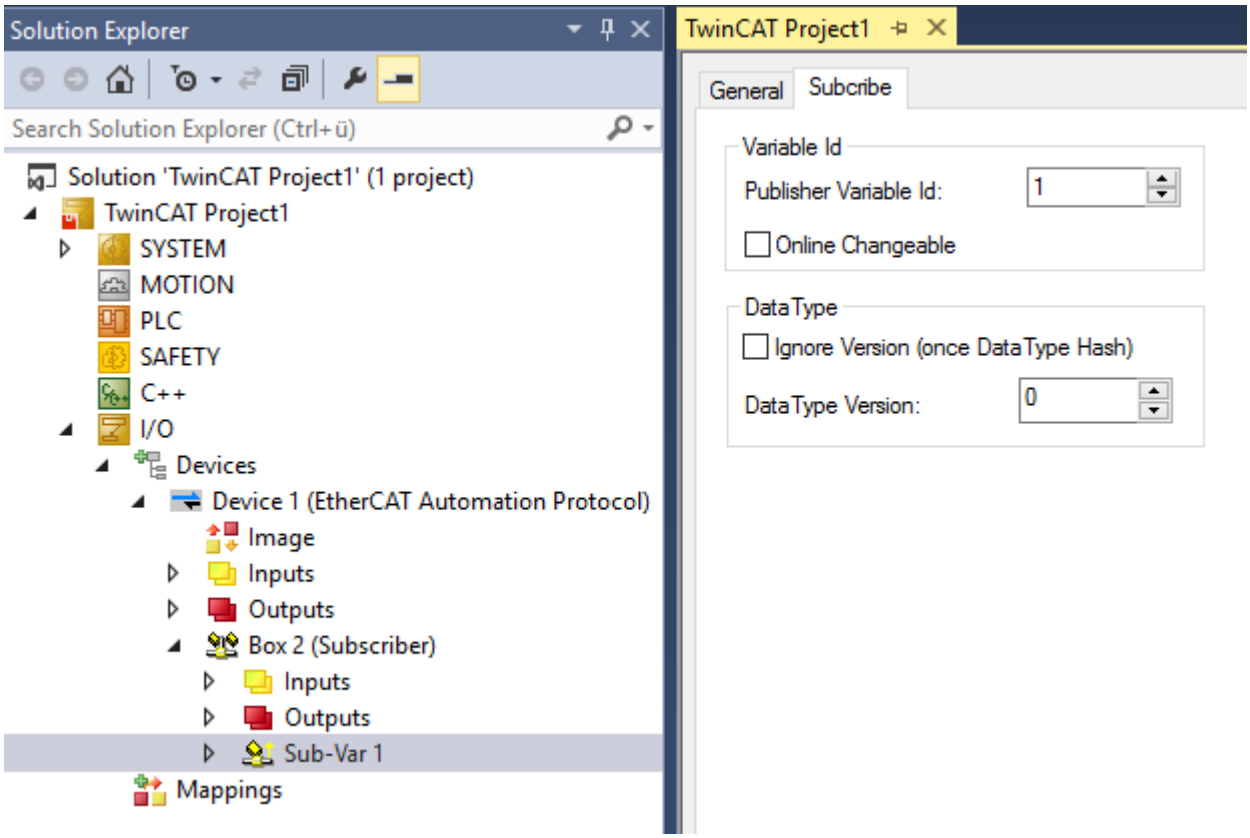
Receiving Options
Mit Hilfe der Auswahl *Receiving Options* wird konfiguriert, ob alle ankommenden EAP-Telegramme, also unabhängig vom Absender, empfangen werden sollen oder ob nur EAP-Telegramme empfangen werden, die von einem bestimmten Absender kommen. Im letzten Fall wird in dem Eingabefeld hinter *Publisher NetId* die *AMS NetID* des gewünschten Absenders eingetragen.

Multicast Configuration
Mit Hilfe der Option *Multicast Configuration* wird festgelegt, ob beim konfigurierten Netzwerkadapter die eingetragene *Multicast* Adresse angemeldet werden soll. Die hier parametrierte *Multicast* Adresse muss identisch zu der beim Sender konfigurierten *Multicast* Adresse sein, wenn dessen EAP-Telegramme empfangen werden sollen (vgl. *Multicast* unter Kapitel Publisher Box [▶ 45]).

Standard Multicast MAC
Die Beckhoff Standard *Multicast* Adresse 01:01:05:04:00:00 wird für ein TwinCAT EAP-Geräte immer beim Netzwerkadapter angemeldet. Auch dann, wenn die Option *Multicast* nicht ausgewählt ist.

5.5 Subscriber Variable

Auf der Konfigurationsseite einer *Subscriber Variable* werden die Eigenschaften definiert, die sich speziell auf die *Subscriber Variable* beziehen (vgl. nächste Abbildung). Zu diesen Eigenschaften gehören:



Variable Id
Die *Variable Id (=ProcessData ID)* ist die Identifizierungsnummer der *Subscriber Variable*. Sie kann mit Hilfe eines SPS Programms online verändert werden, wenn die Option *Online Changeable* markiert ist.

DataType
Jede *Subscriber Variable* hat eine Versionsnummer (vgl. Version im Kapitel EAP-Telegrammstruktur [▶ 18]). Die Versionsnummer kann in „DataType Version“ konfiguriert werden und wird vor dem Empfangen einer *Subscriber Variable* auf Übereinstimmung überprüft (vgl. Abschnitt Vermittlungsprotokolle in Kommunikationsmethoden [▶ 10]). Stimmen die Versionsnummern nicht überein, wird das ankommende *ProcessData* verworfen. Diese Überprüfung wird ausgelassen, wenn die Option *Ignore Version (once DataType Hash)* aktiviert ist.

Konfigurationseinstellungen für einen erfolgreichen Datenaustausch

Um einen Datenaustausch von einer *Publisher Variable* zu einer *Subscriber Variable* zu erwirken, müssen die Konfigurationen der beteiligten Steuerungsrechner zueinander passen. Im Abschnitt Vermittlungsprotokolle des Kapitels Kommunikationsmethoden [▶ 10] wird beschrieben, wie das Empfangen eines EAP-Telegramms bzw. einer *Publisher Variablen* abläuft. In Verbindung mit den folgenden Gesichtspunkten wird deutlich, wie ein Datenaustausch zu gewährleisten ist:

- Die Zieladresse des *Publishers* muss so gewählt werden, dass das EAP-Telegramm den Adressaten erreicht. Telegramme mit *Broadcast* bzw. *Multicast* Adressierung erreichen jeden Netzwerkteilnehmer. Für ein *Unicast* Telegramm muss die genaue Zieladresse des Adressaten konfiguriert werden.
- Sofern beim Empfänger (*Subscriber*) unter den *Receiving Options* die Option *Any Publisher* ausgewählt worden ist, wird jedes ankommende EAP-Telegramm unabhängig von seinem Absender entgegengenommen und weiterverarbeitet.
  Eine Ausnahme liegt nur dann vor, wenn
  - auf der Senderseite eine *Multicast* Adresse konfiguriert wird, die sich von der TwinCAT EAP-*Multicast MAC* (01:01:04:05:00:00) unterscheidet und

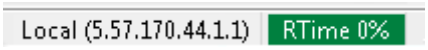
  - die Zieladresse auf der Senderseite (*Publisher*) nicht mit der konfigurierten *Mulitcast MAC* Adresse beim *Subscriber* übereinstimmt.
- Wird beim Empfänger (*Subscriber*) unter den *Receiving Options* eine *Publisher NetId* festgelegt, so werden nur EAP-Telegramme entgegengenommen und weiterverarbeitet, die von dem festgelegten Absender (*Publisher*) kommen.
- Die *ID* der *Publisher* und der *Subscriber Variable* müssen identisch und eindeutig im Netzwerk sein. Eine gesendete *Publisher Variable*, deren *ID* keine Übereinstimmung mit der *ID* einer *Subscriber Variable* hat, wird während des Empfangens verworfen.
- Die Version (der Hash) der *Publisher Variablen* und der zugehörigen *Subscriber Variablen* muss übereinstimmen. Eine gesendete *Publisher Variable*, deren *Version* nicht mit der *Version* der *Subscriber Variable* übereinstimmt, wird während des Empfangens verworfen, es sei denn, es ist beim Empfänger (*Subscriber*) die Option *Ignore Data Type Hash* aktiviert worden.
- Die Rohdatenlänge einer *Publisher Variable* muss mit der erwarteten Rohdatenlänge der *Subscriber Variable* übereinstimmen. Andernfalls wird die *Publisher Variable* während des Empfangens verworfen.

5.6 EAP zwischen TwinCAT 2 und 3

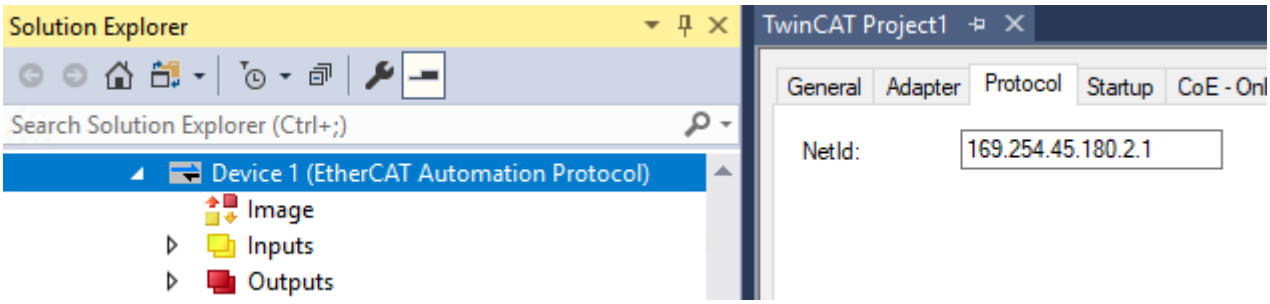
Das EtherCAT Automation Protocol ist kompatibel zu den herkömmlichen Netzwerkvariablen *(NWV)* aus TwinCAT 2. Es gibt allerdings beim EAP unter TwinCAT 3 ein paar Erweiterungen zu den Netzwerkvariablen, die eine erhöhte Aufmerksamkeit erfordern, wenn eine Kommunikation zwischen TwinCAT 2 *NWV* und TwinCAT 3 EAP etabliert werden soll.

Beachtung der Publisher NetID

Wenn TwinCAT 2 einen NWV Publisher nutzt, wird als AmsNetID die AmsNetID des Gesamtsystems verwendet. Diese kann bei dem Empfänger entsprechend eingetragen werden.



Im Gegensatz dazu verwendet TwinCAT 3 bei EAP als Publisher die AmsNetID des EAP-Gerätes.



Die Option „Ignore Data Type Hash“

Ein weiterer zu beachtender Punkt ergibt sich aus der Erweiterung des Datentypsystems zwischen den TwinCAT-Versionen 2 und 3:

TwinCAT 2 berechnet bei der Verwendung eines komplexen (also eines nicht nativen) Datentypen einen 16 Bit Hash-Wert, der diesen Datentypen eindeutig identifiziert. Für native Datentypen ist dieser Data Type Hash stets 0. Beim Versenden einer Publisher Variable der Data Type Hash als Versionsnummer mitgeschickt. Auf der Empfängerseite wird diese Versionsnummer dann mit dem Data Type Hash der konfigurierten Subscriber Variable verglichen. Meldet der Vergleich Übereinstimmung, werden die Daten vom Subscriber angenommen.

In TwinCAT 3 ist jedem Datentypen ein Global Unique Identifier (GUID) zugeordnet. Dieser hat eine Datenlänge von 128 Bit. Entsprechend wird bei der Konfiguration einer Publisher oder Subscriber Variablen als Versionsnummer stets der Wert 0 verwendet.

Wegen dieses Unterschieds zwischen TwinCAT 2 und 3 muss bei der Verwendung von komplexen Datentypen folgendermaßen vorgegangen werden:

✓ Vorausgesetzt, es sollen Variablen zwischen TwinCAT 2 und TwinCAT 3 verschickt werden, die von einem komplexen, als nicht nativen Datentypen sind.

1. Bei der Subscriber Variablen muss in diesem Fall die Option „Ignore Data Type Hash“ aktiviert werden.

⇨ Diese Einstellung unterdrückt die Vergleichsoperation auf der Empfängerseite und die Daten werden ohne Versionsvergleich vom Subscriber angenommen.

Subscriber Variable per „Browse for Computer“ oder „Browse for File“ anlegen

Durch die Änderung des Dateiformates eines TwinCAT 2-Projektes zum TwinCAT 3-Projekt sind die beiden Varianten „Browse for Computer“ oder „Browse for File“, um eine Subscriber Variable passend zu einer bestehenden Publisher Variable anzulegen aktuell nur Abwärtskompatibel. Das heißt, dass es zurzeit nur möglich ist, von einem TwinCAT 3 System aus diese beiden Varianten zu verwenden, um eine Subscriber Variable automatisch anhand einer Publisher Variablen aus einem TwinCAT 2-Projekt erzeugen zu lassen. Im umgekehrten Fall muss eine Subscriber Variable manuell erzeugt werden (Variante „Create new Variable“).

6 Das CANopen Objektverzeichnis

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 *CANopen* Profile definiert, die die veränderlichen und unveränderlichen Parameter eines Gerätes gänzlich beschreiben. Solch ein Parameter umfasst mindestens folgende Eigenschaften:

- Eine **Indexnummer** – zur eindeutigen Identifizierung des Parameters.
  Die Indexnummer unterteilt sich in einen Haupt- und Subindex, um zusammengehörige Parameter zu kennzeichnen und zu ordnen. Der Subindex wird abgesetzt durch einen Doppelpunkt ":".
  Auf diese Weise wird eine Ordnung in zwei Ebenen erreicht (logische Segmente). Der Hauptindex wird grundsätzlich hexadezimal verwendet im Wertebereich 0…65535 (0x0…0xFFFF). Der Subindex wird im Allgemeinen dezimal verwendet im Wertebereich 0…255 (0x0…0xFF).
- Ein **Offizieller Name** - als verständlicher, selbsterklärender Text.
- Die **Zugriffsmöglichkeit** – z.B. ob der Parameter nur gelesen oder auch geschrieben werden kann.
- Ein **Datentyp** – kann je nach Parameter ein Wert vom Typ Text (String), Zahl (Integer, Real), Bool oder Byte-Feld sein.

Die Zuordnung der Indexnummern zu den Parametern wird in einem *CANopen* Profil festgelegt. Auf diese Weise sind alle Parameter hierarchisch wie in einer Tabelle organisiert. In dieser Tabelle ist dann die Gesamtheit der gerätespezifischen Parameter enthalten. Sie wird als *CANopen Object Dictionary* (*OD*) (zu Deutsch: Objektverzeichnis) bezeichnet.

Alle Parameter des TwinCAT EAP-Gerätes sind ebenfalls mit Hilfe eines Objektverzeichnisses organisiert. Es ist konzeptionell genauso aufgebaut wie ein *CANopen OD*. Das Profil für das *OD* eines EAP-Gerätes wurde von der EtherCAT Technology Group (ETG) in der Spezifikation des EtherCAT Automation Protocol festgelegt (ETG 1005, siehe Webpage www.ethercat.org).
Dieses Profil wird identifiziert durch die Profilnummer 5002. Sie legt den Profil-Typen fest (Hauptprofil) und wird im Low-Word (Bits 0-15) des *OD* Parameters *Gerät Type* gespeichert. Das High-Word (Bits 16-31) beinhaltet die Nummer 1000. Sie legt das Modul-Profil fest (Subprofil). Daraus ergibt sich der Wert 0x03e8138a (65541002_{dez}) für den Parameter *Device Type*, der ebenfalls unter dem *Product Code* im *Identity Objekt* hinterlegt ist (Index 0x1018:02).

Beispiel für ein Objekt im OD:
Die eindeutige Identifizierung des Profils eines TwinCAT EAP-Gerätes erfolgt anhand von vier Parametern. Diese werden in einem logischen Segment namens *Identity* zusammengefasst, welches den Hauptindex 4120 (0x1018) hat. Der Parameter *Vendor ID* hat dann die Indexnummer 4120:01 (0x1018:01) und den eingetragenen Wert *2* als Kennzeichnung eines Beckhoff Gerätes.
Das logische Segment *Identity* wird auch als Objekt bezeichnet und aus Anwendersicht so dargestellt:

1018:0	Identity		> 4 <
1018:01	Vendor ID	RO	0x00000002 (2)
1018:02	Product Code	RO	0x03E8138A (65541002)
1018:03	Revision Number	RO	0x00030000 (196608)
1018:04	Serial Number	RO	0x00000000 (0)

Alle Parameter des *Identity* Objekts haben die Eigenschaft *RO* (read only), denn die Parameter sollen vom Anwender nicht verändert werden.

Mit Hilfe der Parameter eines *Object Dictionary* lassen sich ganz verschiedene Eigenschaften beschreiben. Beispiele für solche Parameter sind Herstellerkennung, Versionsnummer, Prozessdateneinstellungen, Gerätename, Abgleichwerte etc. Der Inhalt des *OD* wird zur Inbetriebnahme sowie zur Diagnose des EAP-Gerätes benötigt und kann sehr umfangreich werden.

6.1 Das EAP-Objektverzeichnis (Subprofil 1000)

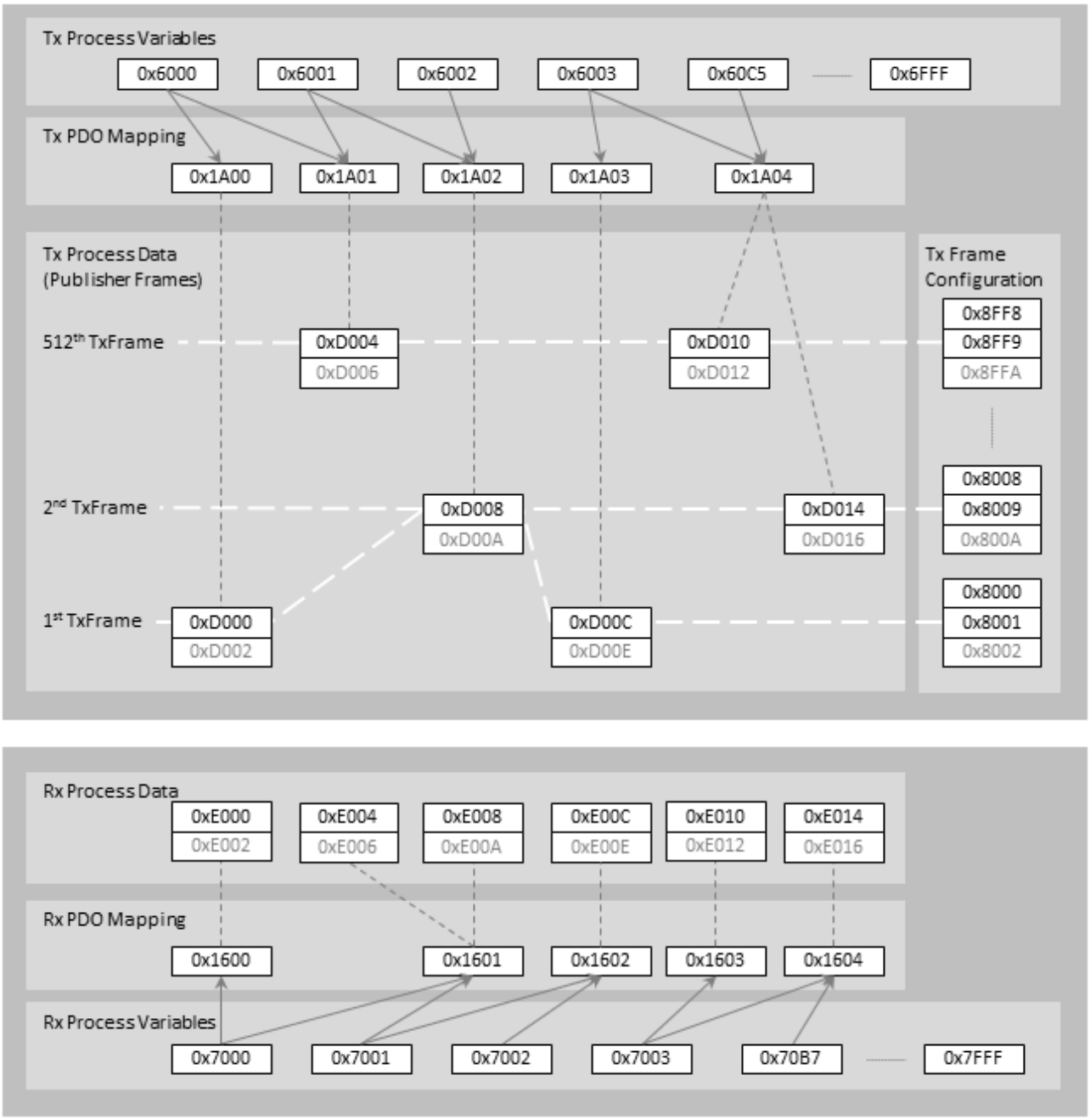
Das EAP-*Object Dictionary* gliedert sich in standardmäßige sowie profilspezifische Objekte. Standard Objekte haben für alle Module die gleiche Bedeutung. Die profilspezifischen Objekte haben für alle Module, die den Profil-Typ 5002 unterstützen, die gleiche Bedeutung. Darüber hinaus können Objekte statisch oder

dynamisch sein. Ein statisches Objekt existiert so lange wie die Instanz eines EAP-Gerätes selbst. Ein dynamisches Objekt kann während der Laufzeit des EAP-Gerätes erzeugt und auch wieder gelöscht werden.

Die Gliederung des Objektverzeichnisses

Das *Object Dictionary* des EAP-Gerätes ist in folgende Bereiche gegliedert:

- **Index 0x1000** – **0x1FFF**: Bereich, der das Kommunikationsprofil beschreibt.
  Hier sind im Bereich 0x1000 – 0x1018 allgemeine Informationen zur Identität des Gerätes hinterlegt wie Name, Hersteller, Seriennummer etc.
  Des Weiteren werden Im Bereich 0x1600 – 0x17FF und 0x1A00 – 0x1BFF *PDO Mapping* Objekte definiert (*PDO = ProcessDataObject*). Ein *PDO Mapping* legt fest, welche Inhalte anderer Objekte des *OD* zu einem *PDO* zusammengefasst werden. Ein *PDO* beschreibt dann den Nutzdateninhalt, der zyklisch in Echtzeit übertragen wird.
- **Index 0x6000** – **0x9FFF**: Bereich, der die funktionsrelevanten Parameter beschreibt.
  Die funktionsrelevanten Parameter sind im ETG Standard 1005 spezifizierte. Unter dem Geräteprofil Nummer 5002, Modulprofil 1000 werden die Parameter inklusive ihrer Struktur definiert. Diese Definition bildet die Grundlage, um einen Datenaustausch per EtherCAT Automation Protocol zu bewerkstelligen. Auf die einzelnen Objekttypen des Profils sowie ihre strukturellen Beziehungen wird im Folgeabschnitt eingegangen.
- **Index 0xF000** – **0xFFFF**: Bereich, der die gerätespezifischen Eigenschaften beschreibt.
  In diesem Bereich liegen Objekte mit Hilfe derer Diagnose- und Kontrollfunktionen beim TwinCAT EAP-Gerät durchgeführt werden können.



Die Objekttypen des standardisierten Profilbereichs und ihre Struktur

Im Folgenden sind die dynamischen Objekte aufgelistet und deren Beziehung zueinander erläutert. Zur Veranschaulichung der Beziehungen dient die Abbildung oberhalb:

Objekte zur Parametrierung eines *Subscribers*:

- *RxVariable* [0x7000+n … 0x7FFF]:
  Eine *RxVariable* definiert eine Variable beliebigen Typs, die mit einer entsprechenden Input Variable einer Steuerungsapplikation (z.B. SPS) verknüpft werden kann.
- RxProcessDataObject (*RxPDO*) [0x1600+n … 0x17FF]:
  Ein *RxPDO* legt eine geordnete Menge von *RxVariables* fest, die ein Prozessdatum als Einheit repräsentiert.
- RxProcessData (*RxPD*) [0xE000+4*n … 0xEFFC]:
  Ein *RxPD* definiert die Eigenschaften für den Empfang eines *PDOs* (vgl. Subscriber Box [▶ 49] und Subscriber Variable [▶ 50]). Das *RxPD* stellt somit die Hauptempfangseinheit der EAP-Kommunikation dar.

- RxProcessDataInfo [0xE002+4*n … 0xEFFE]:
  Ein *RxPDInfo* Objekt erweitert das *RxPD* Objekt um einzelne Eigenschaften, die nicht in der EAP-Spezifikation zu finden sind und speziell zu einem TwinCAT EAP-Gerät gehören.

Objekte zur Parametrierung eines Publishers:

- TxVariable [0x6000+n … 0x6FFF]:
  Eine *TxVariable* Objekt definiert eine Variable beliebigen Typs, die mit einer entsprechenden Output Variable einer Steuerungsapplikation (z.B. SPS) verknüpft werden kann.
- TxProcessDataObject (*TxPDO*) [0x1A00+n … 0x1BFF]:
  Ein *TxPDO* legt eine geordnete Menge von *TxVariables* fest, die ein Prozessdatum als Einheit repräsentiert.
- TxProcessData (*TxPD*) [0xD000+4*n … 0xDFFC]:
  Ein *TxPD* Objekt definiert die Eigenschaften zum Versenden eines *PDOs* (vgl. Publisher Variable [▶ 48]). Das *TxPD* stellt somit die Hauptsendeeinheit der EAP-Kommunikation dar.
- TxProcessDataInfo [0xD002+4*n … 0xDFFE]:
  Ein *TxPDInfo* Objekt erweitert das *TxPD* Objekt um einzelne Eigenschaften, die nicht in der EAP-Spezifikation zu finden sind und speziell zu einem TwinCAT EAP-Gerät gehören.
- TxFrame [0x8000+n*8 ... 0x8FF8]:
  Ein *TxFrame* Objekt definiert die Transporteigenschaften mit denen ein oder mehrere *TxPD* innerhalb des Netzwerks übertragen werden (vgl. Publisher Box [▶ 45]).
- TxPD Assignment [0x8001+n*8 … 0x8FF9]:
  Je einem *TxFrame* Objekt ist ein *TxPDAssignment* Objekt zugeordnet. Das *TxPDAssignment* Objekt besitzt den Index des *TxFrame* Objekts um eins höher. Das Assignment Objekt legt fest, welche *TxPD* gemeinsam in dem entsprechenden *TxFrame* versendet werden.
- TxFrameInfo [0x8002+n*8 … 0x8FFA]:
  Ein *TxFrameInfo* Objekt erweitert das *TxFrame* Objekt um einzelne Eigenschaften, die nicht in der EAP-Spezifikation zu finden sind und speziell zu einem TwinCAT EAP-Gerät gehören.

Die Standard Objekte (0x1000-0x1FFF)

Statische Objekte

Index 1000 Device Type

Index	Name	Bedeutung	Datentyp	Flags	Default
1000:0	Device Type	EAP-Gerätetyp: Das Lo-Word enthält das verwendete CoE Profil (5002). Das Hi-Word enthält das verwendete CoE Sub Profil (1000).	UINT32	RO	0x03E8138A (65541002_{dez})

Index 1008 Device Name

Index	Name	Bedeutung	Datentyp	Flags	Default
1008:0	Device Name	Name des EAP-Gerätes	STRING[256]	RO	EtherCAT Automation Protocol

Index 100A Software Version

Index	Name	Bedeutung	Datentyp	Flags	Default
100A:0	Software Version	Softwareversion des EAP-Gerätes	UINT32	RO	0x00000000 (0_{dez})

Index 1018 Identity

Index	Name	Bedeutung	Datentyp	Flags	Default
1018:0	Identity	Information, um das EAP-Gerät zu identifizieren	UINT8	RO	0x04 (4_{dez})
1018:01	Vendor ID	Herstellerkennung des EAP-Gerätes	UINT32	RO	0x00000002 (2_{dez})
1018:02	Product Code	Produkt-Code des EAP-Gerätes	UINT32	RO	0x03E8138A (65541002_{dez})
1018:03	Product Revision	Änderungsnummer des EAP-Gerätes	UINT32	RO	0x00030000 (196608_{dez})
1018:04	Serial Number	Seriennummer des EAP-Gerätes. 0 bedeutet nicht verwendet	UINT32	RO	0x0 (0_{dez})

Dynamische Objekte

Index 1600-17FF RxPDO Mappings

Index	Name	Bedeutung	Datentyp	Flags	Default
1600+n:0	Number of used Elements	Anzahl der Einträge im RxPDO Mapping Objekt	UINT8	RW	#(Sub-Indizes)
1600+n:01-255	RxVariable m	Bit 0-7: Bit-Länge des eingetragenen Objektes (bei einer Lücke im PDO entsprechend die Bit-Länge der Lücke) Bit 8-15: Subindex des eingetragenen Objektes (bei einer Lücke im PDO den Wert 0) Bit 16-31: Index des eingetragenen Objekts (bei einer Lücke im PDO den Wert 0)	UINT32	RW	-

Index 1A00-1BFF TxPDO Mappings

Index	Name	Bedeutung	Datentyp	Flags	Default
1A00+n:0	Number of used Elements	Anzahl der Einträge im TxPDO Mapping Objekt	UINT8	RW	#(Sub-Indizes)
1A00+n:01-255	TxVariable m	Bit 0-7: Bit-Länge des eingetragenen Objektes (bei einer Lücke im PDO entsprechend die Bit-Länge der Lücke) Bit 8-15: Subindex des eingetragenen Objektes (bei einer Lücke im PDO den Wert 0) Bit 16-31: Index des eingetragenen Objekts (bei einer Lücke im PDO den Wert 0)	UINT32	RW	-

Die profilspezifischen Objekte (0x6000-0xFFFF)

Statische Objekte

Index F100 EAP Status Info

Index	Name	Bedeutung	Datentyp	Flags	Default
F100:0	EAP Status	Statusinformation zum EAP-Gerät	UINT8	RO	0x02 (2_{dez})
F100:01	Status Word	Das Low-Byte kodiert den aktuellen Zustand des EAP-Gerätes: 0 = Invalid 1 = Init 2 = PreOperational 4 = SafeOperational 8 = Operational Das High-Byte kodiert, ob ein Fehler aufgetreten ist: 0 = kein Fehler 1 = Fehler	UINT16	RO	0x0008 (8_{dez})
F100:02	Status Error Code	Eine Fehlernummer die den aufgetretenen Fehler identifiziert. 0 bedeutet, es ist kein Fehler identifiziert.	UINT32	RO	0x03E8138A (65541002_{dez})

Index F200 EAP Control Info

Index	Name	Bedeutung	Datentyp	Flags	Default
F200:0	EAP Control	Parameter zur Kontrolle des EAP-Gerätezustands	UINT8	RO	0x01 (2_{dez})
F200:01	Control Word	Kodiert die Anfrage, um das EAP-Gerät in einen gewünschten Zustand zu setzten: 1 = Init 2 = PreOperational 4 = SafeOperational 8 = Operational	UINT16	RO	0x0008 (8_{dez})

Index F020-F022 Frame List

Index	Name	Bedeutung	Datentyp	Flags	Default
F020+n:0	Number of used Elements	Anzahl der konfigurierten TxFrames	UINT8	RW	#(Sub-Indizes)
F020+n:01-254	Box 1 (Publisher)	Wert 0x0000 0000 = Erstes TxFrame Objekt (Index 8000) existiert nicht Wert 0x0000 03E8 (= 1000): Erstes TxFrame Objekt (Index 8000) existiert Andere Werte sind nicht zulässig. Dieses Objekt kann dazu verwendet werden, um TxFrames zu erzeugen/ löschen	UINT32	RW	0x000003E8 (1000_{dez})

Index F800 EAP Info

Index	Name	Bedeutung	Datentyp	Flags	Default
F800:0	Number of used Elements	Anzahl der Einträge im EAP-Info Objekt	UINT8	RW	0x08 (8_{dez})
F800:01	Available Tx Var	Gibt die maximale Anzahl konfigurierter TxVariable Objekte an (0x6nnn).	UINT16	RW	-
F800:02	Available Rx Var	Gibt die maximale Anzahl konfigurierter RxVariable Objekte an (0x7nnn).	UINT16	RW	-
F800:03	Available Tx Process Data	Gibt die maximale Anzahl konfigurierter Transmit ProcessData Objekte an (0xDnnn).	UINT16	RW	-
F800:04	Available Rx Process Data	Gibt die maximale Anzahl konfigurierter RxProcessData Objekte an (0xEnnn).	UINT16	RW	-
F800:05	Available Tx PDOs	Gibt die maximale Anzahl konfigurierter TxPDO Objekte an (0x1Ann).	UINT16	RW	-
F800:06	Available Rx PDOs	Gibt die maximale Anzahl konfigurierter RxPDO Objekte an (0x16nn).	UINT16	RW	-
F800:07	Available Tx Frames	Gibt die maximale Anzahl konfigurierter TxFrame Objekte an (0x8nnn).	UINT16	RO	-
F800:08	Device Cycle Time	Gibt die Zykluszeit an, mit der das EAP-Gerät getrieben wird. ProcessData Cycle times (z.B. 0xDnnn:07) können nur ganzzahlige vielfache von diesem Wert annehmen.	UINT32	RO	-

Index F801 Bitmap

Index	Name	Bedeutung	Datentyp	Flags	Default
F801:0	Number of used Elements	Anzahl der Einträge im Bitmap Objekt	UINT8	RW	0x06 (6_{dez})
F801:01	Index-Bitmap Tx Var	Bit-kodierte Zuordnung von existierenden TxVariable Objekten. Wenn Bit n gesetzt ist, dann existiert der Index 0x 6000 + n.	OCTETE-STRING [512]	RW	-
F801:02	Index-Bitmap Rx Var	Bit-kodierte Zuordnung von existierenden RxVariable Objekten. Wenn Bit n gesetzt ist, dann existiert der Index 0x 7000 + n.	OCTETE-STRING [512]	RW	-
F801:03	Index-Bitmap Tx Process Data	Bit-kodierte Zuordnung von existierenden TxProcessData Objekten. Wenn Bit n gesetzt ist, dann existiert der Index 0x D000 + 4 *n.	OCTETE-STRING [128]	RW	-
F801:04	Index-Bitmap RxProcess Data	Bit-kodierte Zuordnung von existierenden RxProcessData Objekten. Wenn Bit n gesetzt ist, dann existiert der Index 0x E000 + 4 * n.	OCTETE-STRING [128]	RW	-
F801:05	Index-Bitmap Tx PDOs	Bit-kodierte Zuordnung von existierenden TxPDO Objekten. Wenn Bit n gesetzt ist, dann existiert der Index 0x 1A00 + n.	OCTETE-STRING [64]	RW	-
F801:06	Index-Bitmap Rx PDOs	Bit-kodierte Zuordnung von existierenden RxPDO Objekten. Wenn Bit n gesetzt ist, dann existiert der Index 0x1600 + n.	OCTETE-STRING [64]	RW	-

Index F920 AoE Settings

Index	Name	Bedeutung	Datentyp	Flags	Default
F920:0	Number of used Elements	Anzahl der Einträge im AoE Settings Objekt	UINT8	RW	0x05 (5_{dez})
F920:01	Local AoE NetID	Lokale AoE NetID des EAP-Gerätes	OCTET-STRING [6]	RW	-
F920:02	Router NetID	AoE NetID des zugehörigen AoE-Routers	OCTET-STRING [6]	RO	-
F920:03	Local MAC Address	Lokale MAC Adresse der entsprechenden Netzwerkkarte, die von diesem EAP-Gerät verwendet wird.	OCTET-STRING [6]	RO	-
F920:04	Local IP Address	Lokale IP Adresse der entsprechenden Netzwerkkarte, die von diesem EAP-Gerät verwendet wird.	UINT32	RW	-
F920:05	Local Port Name	Name, unter dem das EAP-Gerät mit seinem AoE Port beim TwinCAT ADS Router angemeldet ist.	STRING [31]	RW	EtherCAT Automation Protocol

Dynamische Objekte

Index 6000-6FFF TxVariables

Index	Name	Bedeutung	Datentyp	Flags	Default
6000+n:0	Number of used Elements	Anzahl der Einträge im TxVariable Objekt	UINT8	RW	0x22 (34_{dez})
6000+n:01	Size	Länge von Data (Subindex 2) in Bit	UINT16	RW	-
6000+n:02	Data	Die aktuellen Daten der Variable	OCTET-STRING [Size/8]	RO	-
6000+n:03	Name	Name der Variablen	STRING [256]	RW	VarData
6000+n:04	Type	Datentyp des Objektes als GUID	GUID	RW	-
6000+n:05	Reserved	-	UINT32	RW	-
6000+n:29	Symbol Name	Symbolname der verknüpften Variable aus der Anwendung (z.B. PLC-Task - PLC-Projektname.MAIN.iCounter	STRING [256]	RW	-
6000+n:30	AoE Address	Octet 7..2: AoE NetID Octet 1..0: AoE Port Des Objektverzeichnisses, welches die aktuelle Prozessvariable enthält	OCTET-STRING [8]	RW	-
6000+n:32	Image Config	Kodierung, welche Input-/Output-Variablen des Prozessabbildes zu diesem Objekt gehören	UINT32	RO	-
6000+n:33	Data Offset	Byte Offset innerhalb des Output-Prozessabbildes	UINT32	RO	-
6000+n:34	Reserved	-	UINT32	RO	-

Index 7000-7FFF Rx Variables

Index	Name	Bedeutung	Datentyp	Flags	Default
7000+n:0	Number of used Elements	Anzahl der Einträge im Rx Variable Objekt	UINT8	RW	0x22 (34_{dez})
7000+n:01	Size	Länge von Data (Subindex 2) in Bit	UINT16	RW	-
7000+n:02	Data	Die aktuellen Daten der Variable	OCTET-STRING [Size/8]	RW	-
7000+n:03	Name	Name der Variablen	STRING [256]	RW	VarData
7000+n:04	Type	Datentyp des Objektes als GUID	GUID	RW	-
7000+n:05	Reserved		UINT32	RW	-
7000+n:29	Symbol Name	Symbolname der verknüpften Variable aus der Anwendung (z.B. PLC-Task - PLC-Projektname.MAIN.iCounter	STRING [256]	RW	-
7000+n:30	AoE Address	Octet 7..2: AoE NetID Octet 1..0: AoE Port Des Objektverzeichnisses, welches die aktuelle Prozessvariable enthält	OCTET-STRING [8]	RW	-
7000+n:32	Image Config	Kodierung, welche Input-/Output-Variablen des Prozessabbildes zu diesem Objekt gehören	UINT32	RO	-
7000+n:33	Reserved	-	UINT32	RO	-
7000+n:34	Data Offset	Byte Offset innerhalb des Output-Prozessabbildes	UINT32	RO	-

Index 8000-8FF8 TxFrame

Index	Name	Bedeutung	Datentyp	Flags	Default
8000+n*8:0	Number of used Elements	Anzahl der Einträge im TxFrame Objekt	UINT8	RW	0x32 (50_{dez})
8000+n*8:03	Name	Name des Frames	STRING [256]	RW	-
8000+n*8:04	Device Type	Sub-Profil Typ (identisch zum entsprechenden Eintrag im Objekt 0xF020-0xF022)	UINT32	RO	0x03E8 (1000_{dez})
8000+n*8:05	Destination Vendor ID	Für eine Peer-to-Peer Kommunikation; 0 = nicht verwendet Polled Connection: Vendor ID des Kommunikationspartners Pushed Connection: Nicht verwendet	UINT32	RW	-
8000+n*8:06	Destination Product Code	Für eine Peer-to-Peer Kommunikation; 0 = nicht verwendet	UINT32	RW	-
8000+n*8:07	Destination Revision Number	Für eine Peer-to-Peer Kommunikation; 0 = nicht verwendet	UINT32	RW	-
8000+n*8:08	Destination Serial Number	Für eine Peer-to-Peer Kommunikation; 0 = nicht verwendet	UINT32	RW	-
8000+n*8:30	Target AMS NetID	AoE NetID (Subscriber Net ID) Wenn der Wert nicht 0 ist, dann müssen die Zieladressen SI 32 und SI 33 den Wert 0 haben.	OCTET-STRING [6]	RW	-
8000+n*8:31	Gateway IP Address	Standard Gateway IP-Adresse muss dann gesetzt werden, wenn dir SI 33 nicht den Wert 0 hat	UINT32	RW	-
8000+n*8:32	Target MAC Address	MAC Adresse Wenn der Wert nicht 0 ist, dann müssen die Zieladressen SI 30 und SI 33 den Wert 0 haben. Die MAC Adresse kann eine Uni-, Multi- oder Broadcast Adresse sein.	OCTET-STRING [6]	RW	01 01 05 04 00 00
8000+n*8:33	Target IP Address	IP Adresse Wenn der Wert nicht 0 ist, dann müssen die Zieladressen SI 30 und SI 32 den Wert 0 haben. Die IP Adresse kann eine Uni-, Multi- oder Broadcast Adresse sein.	UINT32	RW	-
8000+n*8:34	VLAN Info	Die VLAN Info setzt sich aus folgenden Feldern zusammen: Bit 0-15: VLan Type (81 00) Bit 16-18: Priority Bit 19: Reserved Bit 20-31: VLan ID Wenn der Wert 0 ist, wird kein VLAN Header verwendet	UINT32	RW	0x00000000 (0_{dez})

Index	Name	Bedeutung	Datentyp	Flags	Default
8000+n*8:35	Subscriber Monitoring	Wenn der Wert 1 ist, wird regelmäßig ein ARP Request an die konfigurierte Zieladresse gesendet, um sicherzustellen, ob der Adressat noch antwortet. Ist dies nicht der Fall, wird das Senden des TxFrames eingestellt. Das Subscriber Monitoring kann nur bei einer Unicast Kommunikation verwendet werden.	UINT8	RW	0x00 (0_{dez})
8000+n*8:36	Target Changeable	Wenn Target Changeable den Wert 0 hat wird keine Variable für die Zieladresse in das Prozessabbild eingeblendet. Andernfalls gilt folgende Zuordnung bei Einblendung der Zieladresse: 1 : Target MAC Adresse 2 : Target AMS NetID 3 : Target IP Adresse	UINT8	RO	0x00 (0_{dez})
8000+n*8:37	Monitoring Retry Cycles	obsolet	UINT32	RO	-
8000+n*8:38	Monitoring Retry Cycle Time	Wartezeit in µs nach der ein neuer ARP Request gesendet wird sofern SI 35 = 0x01 ist.	UINT32	RW	0xF4240 (1000000_{dez})
8000+n*8:39	Frame Control	Bit 0 = 1 : Das Senden des TxFrames wird eingestellt Bit 1 = 1 : Target MAC Adresse wird aus dem ARP Cache gelöscht	UINT16	RW	0x0000 (0_{dez})
8000+n*8:40	Frame State	Bit 0 = 1: Der TxFrame wurde nicht gesendet Bit 1 = 1: Fehler (der Frame ist zu groß) Bit 2 = 1: Der Subscriber antwortet nicht mehr (nur wenn SI 35 = 0x01 ist)	UINT16	RO	0x0000 (0_{dez})
8000+n*8:48	Control Symbol Name	Symbolname der verknüpften Variable aus der Anwendung (z.B. PLC-Task)	STRING [256]	RW	-
8000+n*8:49	State Symbol Name	Symbolname der verknüpften Variable aus der Anwendung (z.B. PLC-Task)	STRING [256]	RW	-
8000+n*8:50	Target Address Symbol Name	Symbolname der verknüpften Variable aus der Anwendung (z.B. PLC-Task) Wenn ein Symbolname gesetzt wird, wird die Konfigurierte Zieladresse in das Prozessabbild eingeblendet und der SI 36 entsprechend gesetzt.	STRING [256]	RW	-

Index 8001-8FF9 TxProcessData Assignment Objects

Index	Name	Bedeutung	Datentyp	Flags	Default
8001+n*8:0	Number of used Elements	Anzahl der Einträge im TxPD Assignment Objekt	UINT8	RW	#(Sub-Indizes)
8001+n*8 :01-255	Entry n	1. -255. TxProcessData des TxFrames	UINT16	RW	-

Index 8002-8FFA TxFrame Info

Index	Name	Bedeutung	Datentyp	Flags	Default
8002+n*8:0	Number of used Elements	Anzahl der Einträge im TxFrameInfo Objekt	UINT8	RW	0x21 (33_{dez})
8002+n*8:01	Image Config	Kodierung, welche Input /Output Varialen des Prozessabbildes zu diesem Objekt gehören Lo-Word = Input Prozessabbild Bit 0 = 1: State Hi-Word = Output Prozessabbild Bit 0 = 1: Control Bit 1 = 1: Target MAC Address Bit 2 = 1: Target AMS NetID Bit 3 = 1: Target IP Address	UINT32	RO	0x00010001 (65537_{dez})
8002+n*8:02	Control Offset	Byte Offset innerhalb des Output Prozessabbildes	UINT32	RO	-
8002+n*8:03	State Offset	Byte Offset innerhalb des Input Prozessabbildes	UINT32	RO	-
8002+n*8:04	NetID Offset	Byte Offset innerhalb des Output Prozessabbildes	UINT32	RO	-
8002+n*8:32	IoDivMod	Der Divider/Modulo Wert legt fest wie viele Zyklen lang gewartet wird, ehe der nächste TxFrame gesendet wird. Bit 0-7 (Divider): Anzahl der Zyklen, die gewartet wird Bit 8-15 (Modulo): Angabe, ab welchem Start-Zyklus gezählt wird	UINT16	RW	0x0000 (0_{dez})
8002+n*8:33	CoE Index	Für zukünftige Zwecke	UINT16	RW	-

Index D000-DFFC TxProcessData

Index	Name	Bedeutung	Datentyp	Flags	Default
D000+n*4:0	Number of used Elements	Anzahl der Einträge im TxPD Objekt	UINT8	RW	0x22 (34_{dez})
D000+n*4:01	Name	Name des Frames	STRING [256]	RW	-
D000+n*4:02	PDO Number	Die PDO Nummer definiert den Objekt-Index des zugeordneten TxPDO	UINT16	RW	-
D000+n*4:03	Process Data ID	Die PD ID definiert einen Wert im Bereich 0…65535, der eindeutig innerhalb des Kommunikationsnetzwerks sein muss. Die ID ist Teil des Process Data Frame Header.	UINT16	RW	-
D000+n*4:04	Version	Die Version ist ein Wert im Bereich 0… 65535 und sollte konsequent inkrementiert werden, sobald Änderungen an diesem TxPD vorgenommen werden (z.B. der Verweis auf ein anderes TxPDO). Die Version ist Teil des Process Data Frame Headers.	UINT 16	RW	-
D000+n*4:05	CoS On Change Cycles	Obsolet, siehe Subindex 8.	UINT16	RO	-
D000+n*4:06	CoS Inhibit Time	Die Inhibit Time legt die Zeitspanne in µs fest, während der das TxPD nicht erneut gesendet wird; auch dann nicht, wenn sich der Wert einer Prozessvariablen des zugewiesenen PDOs geändert hat. Wenn der Wert gleich 0 ist, dann wird das Senden des TxPD nicht unterbunden. Wenn der Wert > 0 ist, dann muss auch der Wert des Subindex 8 (CoS On Change Timeout) > 0 sein, aber die Werte der Sub-Indizes 7 und 10 müssen gleich 0 sein.	UINT32	RW	-
D000+n*4:07	Cycle Time	Die Cycle Time legt fest, in welchem Zeitintervall in µs das TxPD zyklisch gesendet wird. Wenn der Wert von Cycle Time größer 0 ist, müssen die Sub-Indizes 6, 8 und 10 gleich 0 sein. Wenn der Wert gleich 0 ist, wird das TxPD gar nicht gesendet.	UINT 32	RW	-
D000+n*4:08	CoS On Change Timeout	On Change Timeout legt die maximale Dauer des Zeitintervalls in µs fest, während dessen kein TxPD gesendet wird, es sei denn, es ändert sich währenddessen der Wert einer Prozessvariablen des zugewiesenen PDOs. Wenn der Wert gleich 0 ist, dann werden die Prozessvariablen nicht bei Zustandsänderung gesendet. Wenn der Wert > 0 ist, dann müssen die Werte der Sub-Indizes 7 und 10 gleich 0 sein.	UINT32	RW	-

Index	Name	Bedeutung	Datentyp	Flags	Default
D000+n*4:10	Poll Request Rx PD	Poll Request RxPD definiert den Objekt-Index eines RxProcessData, welches das Senden dieses TxPD auslöst sobald das definierte RxPD einen neuen Wert empfangen hat. Das TxPD arbeitet dann als Server im Polled Data Exchange Modus. Wenn der Wert gleich 0 ist, ist der Polled Data Exchange Modus inaktiv. Wenn der Wert > 0 ist, dann müssen die Sub-Indizes 6, 7 und 8 gleich 0 sein.	UINT16	RW	-
D000+n*4:11	Process Data Control	Bit 0 = 1: Das Senden des TxPD ausschalten	UINT16	RW	0x0000 (0_{dez})
D000+n*4:12	Process Data State	Bit 0 = 1: Das TxPD wurde nicht gesendet	UINT16	RO	-
D000+n*4:32	Control Symbol Name	Symbolname der verknüpften Variable aus einer Anwendung (z.B. PLC-Task)	STRING [256]	RW	-
D000+n*4:33	State Symbol Name	Symbolname der verknüpften Variable aus einer Anwendung (z.B. PLC-Task)	STRING [256]	RW	-
D000+n*4:34	ID Symbol Name	Symbolname der verknüpften Variable aus einer Anwendung (z.B. PLC-Task)	STRING [256]	RW	-

Index D002-DFFE TxProcessData Info

Index	Name	Bedeutung	Datentyp	Flags	Default
D002+n*4:0	Number of used Elements	Anzahl der Einträge im TxPDInfo Objekt	UINT8	RW	0x20 (32_{dez})
D002+n*4:01	Image Config	Kodierung, welche Input/Output Varialen des Prozessabbildes zu diesem Objekt gehören Lo-Word = Input Prozessabbild Bit 0 = 1: State Hi-Word = Output Prozessabbild Bit 0 = 1: Control Bit 1 = 1: ProcessData ID	UINT32	RO	0x00010001 (65537_{dez})
D002+n*4:02	Control Offset	Byte Offset innerhalb des Output Prozessabbildes	UINT32	RO	-
D002+n*4:03	State Offset	Byte Offset innerhalb des Input Prozessabbildes	UINT32	RO	-
D002+n*4:04	ID Offset	Byte Offset innerhalb des Output Prozessabbildes	UINT32	RO	-
D002+n*4:32	IoDivMod	Der Divider/Modulo Wert legt fest wie viele Zyklen lang gewartet wird ehe das TxPD wieder gesendet wird. Bit 0-7 (Divider): Anzahl der Zyklen, die gewartet wird Bit 8-15 (Modulo): Angabe, ab welchem Start-Zyklus gezählt wird	UINT16	RW	0x0000 (0_{dez})

Index E000-EFFC RxProcessData

Index	Name	Bedeutung	Datentyp	Flags	Default
E000+n*4:0	Number of used Elements	Anzahl der Einträge im RxPD Objekt	UINT8	RW	0x25 (37_{dez})
E000+n*4:01	Name	Name des Frames	STRING [256]	RW	-
E000+n*4:02	PDO Number	Die PDO Nummer definiert den Objekt-Index des zugeordneten RxPDO	UINT16	RW	-
E000+n*4:03	Process Data ID	Die PD ID definiert einen Wert im Bereich 0…65535, der mit der ID des empfangenen ProcessData übereinstimmt.	UINT16	RW	-
E000+n*4:04	Version	Die Version ist ein Wert im Bereich 0… 65535 und sollte konsequent inkrementiert werden, sobald Änderungen an diesem RxPD vorgenommen werden (z.B. der Verweis auf ein anderes RxPDO).	UINT 16	RW	-
E000+n*4:05	Ignore Version	Wenn 0, dann wird die Version (Hash-Wert) des empfangenen ProcessData anhand der Version aus Subindex 4 geprüft. Wenn 1, dann ist die Versionsüber-prüfung deaktiviert.	UINT8	RW	0x00 (0_{dez})
E000+n*4:06	Publisher NetID	Definition einer Publisher NetID. Es wird nur ein EAP-Telegramm verarbeitet, welches von einem Sender mit dieser NetID verschickt wurde. Wenn die Publisher NetID den Wert 0 hat, ist dieser Filter deaktiviert.	OCTET-STRING [6]	RW	00 00 00 00 00 00
E000+n*4:07	MAC Address	Es kann eine Multicast MAC Adresse definiert werden, die die Netzwerkkarte (NIC – Network Interface Card) als Filter für den Empfang von Multicast Paketen verwendet. Wenn der Wert 0 ist, ist die Filterfunktion deaktiviert.	OCTET-STRING [6]	RW	01 01 05 04 00 00
E000+n*4:08	IP Address	Es kann eine Multicast IP Adresse definiert werden, die die Netzwerkkarte (NIC – Network Interface Card) als Filter für den Empfang von Multicast Paketen verwendet. Wenn der Wert 0 ist, ist die Filterfunktion deaktiviert.	UINT32	RW	0x00000000 (0_{dez})
E000+n*4:09	Update Time	Mit Hilfe der Update Time wird das Zeitintervall in µs festgelegt, innerhalb dessen ein neues ProcessData empfangen worden sein muss. Wenn der Wert 0 ist, dann ist dieser Mechanismus deaktiviert.	UINT32	RW	0x00000000 (0_{dez})

Index	Name	Bedeutung	Datentyp	Flags	Default
E000+n*4:10	Poll Request TxPD	Poll Request TxPD definiert den Objekt-Index eines TxProcessData, welches als Anfrage gesendet wird, um ein EAP-Telegramm mit dem passenden ProcessData zu erhalten. Das RxPD arbeitet dann als Client im Polled Data Exchange Modus. Wenn der Wert gleich 0 ist, ist der Polled Data Exchange Modus inaktiv.	UINT16	RW	0x0000 (0_{dez})
E000+n*4:11	Process Data Control	Bit 0 = 1: Die Überprüfung der Versionsnummer bzw. des Hash-Wertes ist deaktiviert.	UINT16	RW	0x0000 (0_{dez})
E000+n*4:12	Process Data State	Bit 0 = 1: Ein ProcessData mit ungültiger Versionsnummer (Hash-Wertes) wurde empfangen Bit 1 = 1: Ein ProcessData mit ungültiger Länge wurde empfangen Bit 2 = 1: Der Timeout Poll Response wurde überschritten	UINT16	RO	-
E000+n*4:13	Process Data Quality	Die Quality gibt die Dauer in 100µs an, die dieses RxProcessData nicht mehr aktualisiert wurde (also keine Daten empfangen hat)	UINT16	RO	-
E000+n*4:14	Process Data Cycle Index	Dem Cycle Index wird beim Empfang eines gültigen ProcessData der übermittelte Cycle Index aus dem EAP-Telegramm zugewiesen (Siehe Process Data Frame Header)	UINT16	RO	-
E000+n*4:32	Control Symbol Name	Symbolname der verknüpften Variable aus einer Anwendung (z.B. PLC Task)	STRING [256]	RW	-
E000+n*4:33	State Symbol Name	Symbolname der verknüpften Variable aus einer Anwendung (z.B. PLC Task)	STRING [256]	RW	-
E000+n*4:34	ID Symbol Name	Symbolname der verknüpften Variable aus einer Anwendung (z.B. PLC Task)	STRING [256]	RW	-
E000+n*4:35	Quality Symbol Name	Symbolname der verknüpften Variable aus einer Anwendung (z.B. PLC-Task)	STRING [256]	RW	-
E000+n*4:36	Cycle Index Symbol Name	Symbolname der verknüpften Variable aus einer Anwendung (z.B. PLC-Task)	STRING [256]	RW	-
E000+n*4:37	Timeout Poll Response	Legt die maximale Zeitspanne in µs fest, innerhalb derer die Antwort zum Polled Request empfangen werden muss. Wenn der Wert > 0 ist und der Timeout nach dem Senden des Polled Request abgelaufen ist, wird das Bit 2 in PD State (Subindex 12) gesetzt. Wenn der Wert 0 ist, ist diese Überwachung deaktiviert.	UINT32	RW	0x00000000 (0_{dez})

Index E002-EFFE RxProcessDataInfo

Index	Name	Bedeutung	Datentyp	Flags	Default
E002+n*4:0	Number of used Elements	Anzahl der Einträge im RxPD Info Objekt	UINT8	RW	0x06 (6_{dez})
E002+n*4:01	Image Config	Kodierung, welche Input-/Output-Varialen des Prozessabbildes zu diesem Objekt gehören Lo-Word = Input-Prozessabbild Bit 0 = 1: State Hi-Word = Output-Prozessabbild Bit 0 = 1: Control / Cycle Index Bit 1 = 1: ProcessData ID	UINT32	RO	0x00010001 (65537_{dez})
E002+n*4:02	Control Offset	Byte Offset innerhalb des Output-Prozessabbildes	UINT32	RO	-
E002+n*4:03	State Offset	Byte Offset innerhalb des Input-Prozessabbildes	UINT32	RO	-
E002+n*4:04	ID Offset	Byte Offset innerhalb des Output-Prozessabbildes	UINT32	RO	-
E002+n*4:05	Quality Offset	Byte Offset innerhalb des Input-Prozessabbildes	UINT32	RO	-
E002+n*4:06	Cycle Index Offset	Byte Offset innerhalb des Input-Prozessabbildes	UINT32	RO	-

6.2 Das TwinCAT ADS Interface zum EAP-Gerät

Das TwinCAT EAP-Gerät stellt für andere Kommunikationspartner (z.B. virtuelle Feldgeräte oder Windows-Programme) ein TwinCAT *ADS/AMS* Interface zur Verfügung und agiert als *ADS/AMS*-Server. *ADS* steht für *Automation Device Specification* und beschreibt eine geräte- und feldbusunabhängige Schnittstelle. *AMS* steht für *Automation MessageSpecification* und ermöglicht es, zentrale und dezentrale Systeme zu adressieren wie PCs aber auch Buscontroller. *ADS/AMS* ist von Beckhoff spezifiziert worden und wird vom TwinCAT Router unterstützt. Nachrichten, die über die Rechnergrenze hinaus in einem Netzwerk verschickt werden, werden per *TCP/IP* übertragen.

Über dieses Interface kann auch der *CANopen* Kommunikationskanal *SDO* (*Service Data Object*) genutzt werden. Die *SDOs* dienen in erster Linie zum Lesen sowie Schreiben der Parameter des *CANopen Object Dictionary* (*OD*). Die Übertragung von *SDOs* erfolgt als bestätigter Datentransfer in Form einer Punkt-zu-Punkt-Verbindung zwischen zwei Kommunikationspartnern und ist in ADS eingebettet:

Unter Verwendung der *ADS Read* oder *ADS Write* Kommandos auf den Port 0xFFFF und die NetID des EAP-Gerätes (Tab „Protocol“) werden die Parameter bzw. die *SDO*-Beschreibung des *CANopen OD* gelesen oder geschrieben. Wie in der Tabelle aufgeführt, ist der Kommunikationskanal *CANopen SDO* folgendermaßen in das *ADS* Protokoll eingebettet:

CANopen SDO Kommunikation	ADS Command	Index Group	Index Offset	Bedeutung
SDO Upload	Read	0xF302	Index und Subindex eines SDO. Bit 16-31: Index Bit 8: Complete Access Bit 0-7: Subindex Beispiel: 0x16010001: Index = 0x1601 Complete Access = 0 Subindex = 1 Wenn der Complete Access = 1 ist, dann hat der Subindex keine Bedeutung	Data Type: UINT8[n] SDO Upload Request: Das Objekt wird anhand des Index Offset adressiert und sein Inhalt kann mit Hilfe eines ADS Read gelesen werden.
SDO Download	Write	0xF302	Index und Subindex eines SDO. Bit 16-31: Index Bit 8: Complete Access Bit 0-7: Subindex Beispiel: 0x16010001: Index = 0x1601 Complete Access = 0 Subindex = 1 Wenn der Complete Access = 1 ist, dann hat der Subindex keine Bedeutung	Data Type: UINT8[n] SDO Download Request: Das Objekt wird anhand des Index Offset adressiert und sein Inhalt kann mit Hilfe eines ADS Write geschrieben werden.
SDO Information Get Object List	Read	0xF3FC	Bit 16-31: Listentypen	Liefert die Indizes der Objekte zurück, die zu dem Listentyp gehören, der im Index Offset spezifiziert ist. Mögliche Listentypen sind: ALL_OBJECTS = 1 RXPD_OBJECTS = 2 TXPD_OBJECTS = 3 BACKUP_OBJECTS = 4 SETTING_OBJECTS = 5
			Beispiele: 0x00000000: Liefert zu jedem Listentyp die Anzahl aller vorhandenen Objekte	Data Type: UINT16[6] Wenn Listentyp = 0000: Element = 0 : Anzahl der Listentypen Element > 0 : Anzahl vorhandener Objekte, die zum n-ten Listentypen gehören
			0x00010000: Liefert die Indizes aller Objekte für den spezifizierten Listentyp zurück	Data Type: UINT16[n] Wenn Listentyp > 0000: Element n=0 : Anzahl vorhandener Objekte, die zu diesem Listentypen gehören plus eins. Element n>0: Der n-te Objekteindex, der zu diesem Listentypen gehört
SDO Information Get Object Description	Read	0xF3FD	Bit 16-31: Index	Lesen der SDO Beschreibung des vollständigen Objekts mit dem spezifizierten Index.

CANopen SDO Kommunikation	ADS Command	Index Group	Index Offset	Bedeutung
SDO Information Get Entry Description	Read	0xF3FE	Bit 16-31: Index Bit 0-7: Subindex	Lesen der SDO Beschreibung des einzelnen Eintrags mit dem spezifizierten Subindex, der zu dem Objekt mit dem spezifizierten Index gehört.

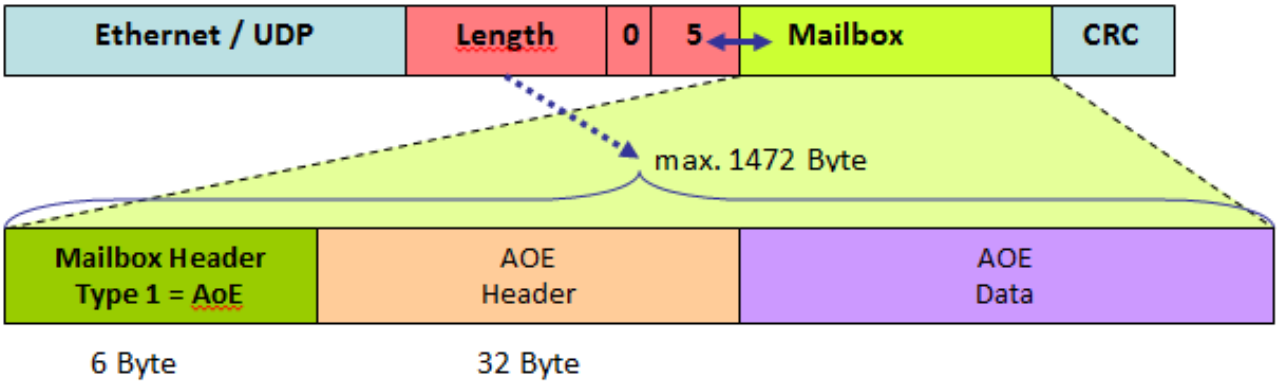
6.3 ADS over EtherCAT (AoE)

Das TwinCAT EAP-Gerät unterstützt des Weiteren das *AoE* Protokoll. Die Spezifikation des AoE Protokolls ist in den EtherCAT Protocol Enhancements (ETG 1020) zu finden. Der Unterschied zwischen der ADS/AMS Kommunikation und AoE Kommunikation ist der, dass bei der AoE Kommunikation im Gegensatz zur ADS/AMS Kommunikation kein TwinCAT-Router notwendig ist. Das AoE Protokoll ist eines der Protokolle, die in TwinCAT unter die Rubrik der Mailbox-Kommunikation eingeordnet werden. Für die Vermittlung von AoE wird ein EtherCAT Telegramm vom Typ 5 (Mailbox-Kommunikation) verwendet. Ein Mailbox Telegramm kann vom bzw. zum TwinCAT EAP-Gerät übermittelt werden:

via Ethernet (EtherType = 0x88A4) oder

via UDP/IP (UDP Port = 0x88A4)

Es besteht die Möglichkeit verschiedene Protokolle per Mailbox Kommunikation zu transportieren (tunneln). Das zu tunnelnde Protokoll wird anhand des Feldes Type im *Mailbox Header* festgelegt. Das AoE Protokoll ist durch den Wert 1 spezifiziert (vgl. *Mailbox Header* in folgender Abbildung). An den *Mailbox Header* schließt sich dann direkt der *AoE Header* gefolgt von den *AoE* Daten an. Deren Aufbau ist identisch zum *ADS/AMS*-Protokoll (vgl. dazu auch die TwinCAT *ADS/AMS*-Spezifikation im Beckhoff Informationssystem). Daraus ergibt sich die Möglichkeit, die *CANopen SDO* Kommunikation auch über das Mailbox Protokoll zu verwenden. Es kann analog zum *ADS/AMS*-Protokoll auf das *CANopen OD* des TwinCAT EAP-Gerätes zugegriffen werden, wie in Kapitel Das TwinCAT ADS Interface zum EAP-Gerät [▶ 70] beschrieben.



Beispiel:
Um die *Vendor ID* des Identity Objekts aus dem EAP-Objektverzeichnis auszulesen, wird im *AoE Header* anhand der *Command ID* das *ADS* Kommando festgelegt:

AoE Kommando	Command ID	Beschreibung
ADS Read	2	ADS Read Kommando, um Daten zu lesen
ADS Write	3	ADS Write Kommando, um Daten zu schreiben

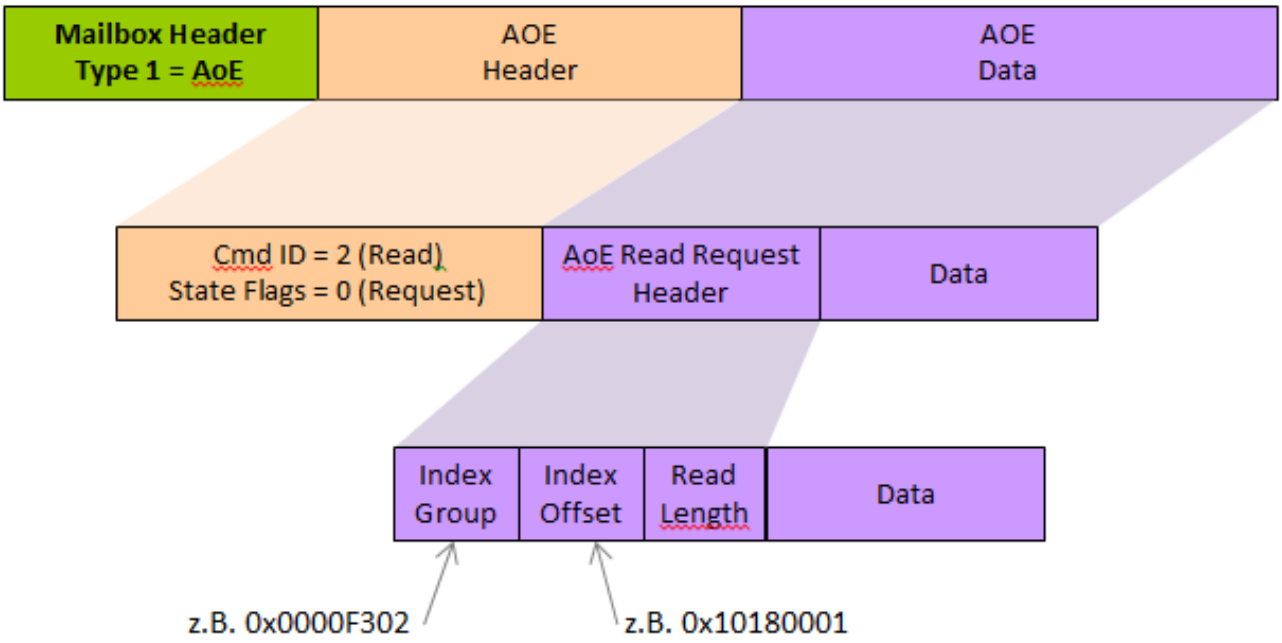
Anhand des 0-ten Bits im *State Flag* des *AoE Headers* wird zwischen *Request = 0* und *Response = 1* unterschieden. Anschließend wird das *AoE Data* Feld mit folgendem Inhalt versehen (vgl. folgende Abbildung):

Zu Beginn steht ein *ADS Read Request Header* mit

- Index Group = 0x0000F302, um den Inhalt des Parameters zu lesen,
- Index Offset = 0x10180001, um den Parameter *Vendor ID* zu adressieren,

- und Read Length = 4, um die Größe des verfügbaren Datenpuffers mitzuteilen.

Anschließend folgt der Datenpuffer für die zu lesenden Daten.



6.4 Das TwinCAT EAP-Gerät online konfigurieren

Das *CANopen OD* bietet den großen Vorteil, ein TwinCAT EAP-Gerät online über eine standardisierte Schnittstelle nahezu uneingeschränkt konfigurieren zu können. Eine Online-Konfiguration vorzunehmen bedeutet, dass das TwinCAT EAP-Gerät bereits aktiv ist und nur in einen sicheren Zustand geschaltet werden muss, ehe eine Änderung der Konfiguration erfolgt. Im sicheren Zustand findet keine Prozessdatenkommunikation statt.

Das Ändern der Konfiguration erfolgt durch Setzen von Parametern und geschieht

- per TwinCAT:
  Die Werte werden dann direkt im online Objektverzeichnis geändert.
  Dies bietet sich bei der Inbetriebnahme der Anlage an. Per Klicken auf die entsprechende Zeile des zu parametrierenden Index und durch Eingabe eines entsprechenden Wertes im Dialog *Set Value*, wird der neue Wert beim entsprechenden Parameter gesetzt.
- oder aus einer Anwendung heraus über ADS oder AoE:
  Dies wird für Änderungen während des Betriebs der Anlage empfohlen oder wenn kein TwinCAT bzw. Bedienpersonal zur Verfügung steht.

Zum einen kann eine Änderung ausschließlich die Einträge bereits existierender Objekte des *OD* betreffen. Zum anderen kann eine Änderung die Erzeugung neuer oder das Löschen bestehender Objekte zur Folge haben.

Existierende Objekte ändern

Eine Änderung, die nur existierende Objekte betrifft, dient der Anpassung des aktuellen Betriebsverhaltens (z.B. Sendezyklus, Zieladresse etc.) oder auch um einzelnen Objekten zweckgebundene Namen für ein besseres Verständnis zuzuweisen. Welche Schritte für solch eine Änderung notwendig sind, wird am besten deutlich durch ein Beispiel:

Das Ändern des Zeitintervalls, mit dem eine Publisher Variable übertragen wird.

Angenommen das EAP-Gerät ist mit einem *TxPD* (0xD000) konfiguriert, bei dem als *Cycle Time* ein Intervall von 10000 µs konfiguriert ist und dieses Intervall soll nun auf 30000 µs vergrößert werden. Dann sind folgende Schritte durchzuführen:

1. Das EAP-Gerät in einen sicheren Zustand versetzen.
   Objekteintrag 0xF200:01 (*Control Word*) auf den Wert 2 (=PreOperational) setzen
   Objekteintrag 0xF100:01 prüfen, ob der Zustand auf Wert 2 (=PreOperational) steht
   Diese Prüfung mehrfach (pollend) durchführen mit einem Timeout von z.B. 200ms
2. Das Zeitintervall des TxPD auf den neuen Wert ändern
   Den Objekteintrag *Cycle Time* des *TxPD* (0xD000:07) auf den Wert 30000 setzen
3. Das EAP-Gerät wieder in seinen operativen Zustand versetzten.
   Objekteintrag 0xF200:01 (*Control Word*) auf den Wert 4 (=SafeOperational) setzen
   Objekteintrag 0xF100:01 prüfen, ob der Zustand auf Wert 4 (=SafeOperational) steht
   Diese Prüfung mehrfach (pollend) durchführen mit einem Timeout von z.B. 200ms
   Objekteintrag 0xF200:01 (*Control Word*) auf den Wert 8 (=Operational) setzen
   Objekteintrag 0xF100:01 prüfen, ob der Zustand auf Wert 8 (=Operational) steht
   Diese Prüfung mehrfach (pollend) durchführen mit einem Timeout von z.B. 200ms

Erzeugen und Löschen von Objekten

Das Erzeugen neuer Objekte dient der Erweiterung bestehender Kommunikationsverbindungen bzw. dem Hinzufügen neuer Kommunikationsverbindungen. Diese Funktion ermöglicht es z.B. eine temporäre Kommunikationsverbindung zwischen zwei Steuerungen zu etablieren. Dabei wird nur das Senden und Empfangen des EAP-Gerätes kurzzeitig während der Konfigurationsphase unterbrochen. Die Prozesse anderer Komponenten und Module der Steuerung bleiben davon unberührt und können durchgehend aktiv bleiben.

Mit Hilfe der Objekte *Frame List* (Index 0xF020 – 0xF022) und *Bitmap* (Index 0xF801) werden Objekte erzeugt oder gelöscht.
In den Objekten *Frame List* gibt es insgesamt 512 Einträge, genauso viele wie es maximal *TxFrame* Objekte geben kann (vgl. Index 0xF020 in Das EAP-Objektverzeichnis (Subprofil 1000) [▶ 53]). Jedem Eintrag ist also genau ein *TxFrame* Objekt zugeordnet. Für jedes existierende *TxFrame* Objekt ist bei dem entsprechenden Eintrag dann der Wert 1000 gespeichert (Sub-Profil Nummer), alle anderen Einträge haben den Wert 0. Angenommen es gibt drei *TxFrames* mit den Indizes 0x8000, 0x8008 und 0x87F0. Dann ist die Zuordnung folgendermaßen zu verstehen:

Nr.	Frame List Einträge	Wert	Zugeordnetes TxFrame Objekt
1	0xF020:01	1000	0x8000
2	0xF020:02	1000	0x8008
…	…	0	…
254	0xF020:254	0	0x87E8
255	0xF021:01	1000	0x87F0
…	…	0	…
512	0xF022:04	0	0x8FF8

Sobald der Wert eines Eintrags auf den Wert 1000 gesetzt wird, wird das entsprechende *TxFrame* Objekte erzeugt. Beim Zurücksetzten auf den Wert 0 wird das Objekt wieder gelöscht.

Im Objekt *Bitmap* ist für jeden dynamischen Objektetyp außer *TxFrames* ein separater Eintrag vorhanden. Entsprechend sind sechs Einträge zu finden (ohne den Eintrag 0 – vgl. Index 0xF801 in Das EAP-Objektverzeichnis (Subprofil 1000) [▶ 53]). Ein Eintrag enthält ein Bit-Feld ausreichender Länge, um die maximale Anzahl von Objekten des entsprechenden Objekttypen in unärer Darstellung speichern zu können.

Angenommen es gibt drei *TxPD* Objekte mit den Indizes 0xD000, 0xD00C und 0xD014. Dann ist im Eintrag 0xF801:03 die Bitfolge 10010100000000…0 mit der Gesamtlänge von 1024 Bits gespeichert. Die Folge ist folgendermaßen zu interpretieren:

Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	Bit 8		Bit 1024
1	0	0	1	0	1	0	0	…	0
0xD000	0x D004	0x D008	0x D00C	0x D010	0x D014	0x D018	0x D01C	…	0x DFFC

Sobald ein Bit auf 1 gesetzt wird, wird das entsprechende Objekt erzeugt. Beim Zurücksetzen wird das Objekt wieder gelöscht.

HINWEIS

Löschen von offline konfigurierten Objekten

Es können auch Objekte gelöscht werden, die vor der Inbetriebnahme mittels TwinCAT konfiguriert worden sind (offline Konfiguration) und dadurch bereits bei der Aktivierung der Steuerung erzeugt werden. Dabei ist jedoch folgendes zu beachten:

Einige Objekte verwalten Variablen (wie z.B. *CycleIndex*, *Quality*, *Status Word*, *Control Word* etc.), die im Prozessabbild des EAP-Gerätes liegen. TwinCAT greift bei der Anzeige von Online-Werten per *ADS* auf das Prozessabbild zu. Dabei berechnen sich die Adressen zu den jeweiligen Variablen im Prozessabbild nach der Konfiguration, die mit Hilfe von TwinCAT erzeugt und aktiviert worden sind.

Wenn nun ein offline konfiguriertes Objekt bei einer online Konfiguration gelöscht wird, bleibt diese Änderung vor TwinCAT verborgen. Er greift also nach wie vor auf die berechneten Adressen zu, um deren Inhalt in der Online-Ansicht anzuzeigen. An dieser Stelle soll deutlich werden, dass dann die angezeigten Online-Werte nicht mehr zu der in TwinCAT erzeugten Konfiguration passen.

Die Online-Ansicht bzw. die Adressen des Prozessabbildes sind in diesem Fall also keine gültige Bezugsquelle mehr, um die Werte des Prozessabbildes zu beobachten oder sogar auszuwerten.

Es wird empfohlen offline konfigurierte Objekte zu deaktivieren, statt zu löschen.

Wenn das Löschen oder Hinzufügen von Objekten mit Hilfe von TwinCAT durchgeführt wird, werden neue Objekte nicht direkt im Objektverzeichnis angezeigt. Gelöschte Objekte verbleiben im *OD* und für deren Einträge wird kein Wert mehr angezeigt. Damit neue Objekte angezeigt und gelöschte Objekte nicht mehr angezeigt werden, muss in TwinCAT die Struktur des *OD* erneut gelesen werden. Dies kann mit Hilfe des Dialogs *Advanced Settings* vorgenommen werden (siehe Abb. *CoEOnline* in Kapitel Das TwinCAT EAP-Gerät [▶ 42]).

Aktivieren/Deaktivieren von Objekten

Für den Fall, dass bestimmte *Prozess Data/PDOs* oder vollständige *Frames* evtl. nur temporär aus der bestehenden EAP-Kommunikation herausgenommen werden sollen, ist es sinnvoll die betreffenden Objekte zu deaktivieren, statt zu löschen.

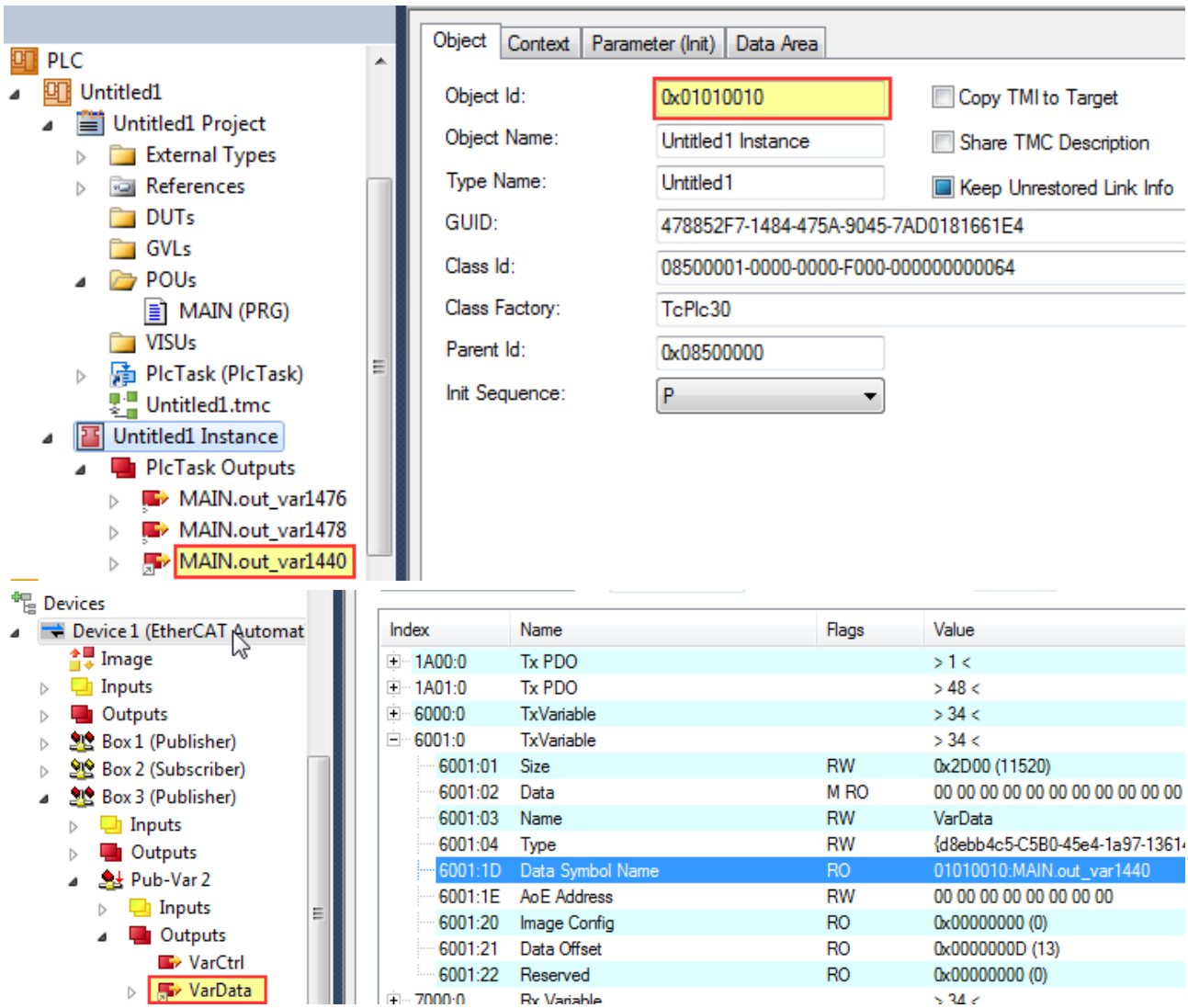
Es können die Objekte *TxFrame* (0x8000), *TxPD Assignment* (0x8001), *TxPD* (0xD000), *TxPDO* (0x1A00), *RxPD* (0xE000) und *RxPDO* (0x1600) deaktiviert und wieder aktiviert werden, indem ihr Subindex 0 auf den Wert 0 gesetzt wird. Das Aktivieren erfolgt dann durch das Setzen des ursprünglichen Wertes, oder einen beliebigen Wert größer 0.

Listenbasierte Objekte

Bei den listenbasierten Objekten *TxPD Assignment* und *Rx/TxPDOs* ist zu beachten, dass deren Einträge stets Referenzen auf andere Objekte enthalten. Das Ändern des Subindex 0 hat daher Einfluss darauf, welche bzw. wie viel andere Objekte tatsächlich referenziert werden.

Verknüpfung einer Variablen aus dem EAP-Prozessabbild mit einer SPS Variable

Die Verknüpfung von Variablen aus der TwinCAT I/O-Ebene mit Variablen eines SPS Programms erfolgt standardgemäß in TwinCAT. Wenn eine *Input/Output Variable* aus einem SPS Programm mit einer *Publisher oder Subscriber Variablen* des EAP-Gerätes verknüpft wird, wird der Symbolname dieser Variablen beim entsprechenden *CANopen* Objekt des EAP-Gerätes eingetragen (siehe nächste Abbildung).



Wie die vorhergehende Abbildung zeigt, setzt sich der Symbolname aus der *Object ID (OID)* der PLC Instance und dem eindeutigen Namen der SPS Programmvariablen zusammen, getrennt durch einen Doppelpunkt. Unter Verwendung der gleichen Syntax kann ein Symbolname auch während der online Konfiguration bei einem EAP-Objekt eingetragen werden. Die SPS Variable wird dann entsprechend mit der *Rx/TxVariable* verknüpft.

Online Konfiguration von offline Verknüpfungen

Bei Verknüpfungen, die bereits während der offline Konfiguration erzeugt worden sind, können diese bei der online Konfiguration weder verändert noch gelöscht werden. Diese Symbolnamen sind daher mit dem *Flag Read-Only (RO)* gekennzeichnet.

Grund:
Die Verknüpfung der Variablen wird mit Hilfe eines Mapping-Objekts beim Anlegen einer Konfiguration von TwinCAT festgelegt. Dieses Mapping-Objekt kann jedoch nicht während des Betriebs verändert werden.

Zugriff auf das CANopen Object Dictionary per Single Access/Complete Access

Das *ADS Interface* mit der Unterstützung des Kommunikationskanals *CANopen SDO* (siehe Das TwinCAT ADS Interface zum EAP-Gerät [▶ 70]) erlaubt es, sowohl die Werte einzelner Objekteinträge als auch die Daten eines kompletten Objektes zu lesen bzw. zu schreiben. Der Zugriff auf einen einzelnen Objekteintrag (*Single Access*) wird im Abschnitt *Existierende Objekte Ändern* in Kapitel Das TwinCAT EAP-Gerät online konfigurieren [▶ 73] erläutert.
Ein *Complete Access* ist der Zugriff auf das komplette Objekt. Dabei werden die Werte aller Objekteinträge aufeinanderfolgend in einem Block binär übertragen. Zur Sicherheit wird bei einem Schreibzugriff die Übereinstimmung der Blocklänge mit der Größe des Objekts geprüft, ehe der Zugriff gewährt wird. Zur Ermittlung der erforderlichen Blocklänge kann die Dokumentation des Objektverzeichnisses herangezogen

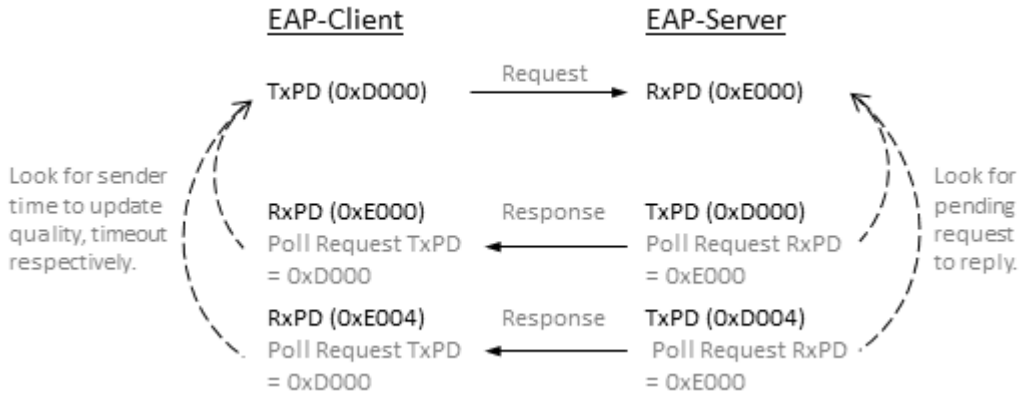
oder auch eine offline Konfiguration in ein *EAP Device Configuration* (EDC) File exportiert werden (vgl. Absatz Protocol in Kapitel Das TwinCAT EAP-Gerät [▶ 42]). Letzteres ist eine Datei im XML-Format, in der u.a. das Objektverzeichnis der aktuellen Konfiguration vollständig enthalten ist (siehe Kapitel Das EAP Device Configuration (EDC) File [▶ 79]).

6.5 Polled Data Exchange konfigurieren

Der Polled Data Exchange Modus wird mit Hilfe des EAP-Objektverzeichnisses konfiguriert. Dazu muss bei einem EAP-Gerät das *RxPD* konfiguriert werden, welches als Client die Antwort des Servers empfangen soll, und bei einem anderen EAP-Gerät muss das *TxPD* konfiguriert werden, welches als Server die Antwort auf die Anfrage des Clients definiert.

Wie in folgender Abbildung dargestellt ist, wird auf der Client-Seite für die Anfrage (eng. Request) ein *TxPD* konfiguriert, das zyklisch gesendet wird wie im Pushed Data Exchange Modus. Für den Empfang der Server-Antwort werden auf der Client-Seite in diesem Beispiel zwei *RxPDs* konfiguriert. Bei beiden *RxPDs* wird bei der Eigenschaft *Poll Request TxPD* der Index des *TxPD* eingetragen, welches als Anfrage dient. Ein RxPD kann dann anhand des Sendezeitpunktes der Anfrage einen Qualitätswert ermitteln, anhand dessen die zeitliche Differenz zwischen Anfrage und Antwort abgelesen werden kann.

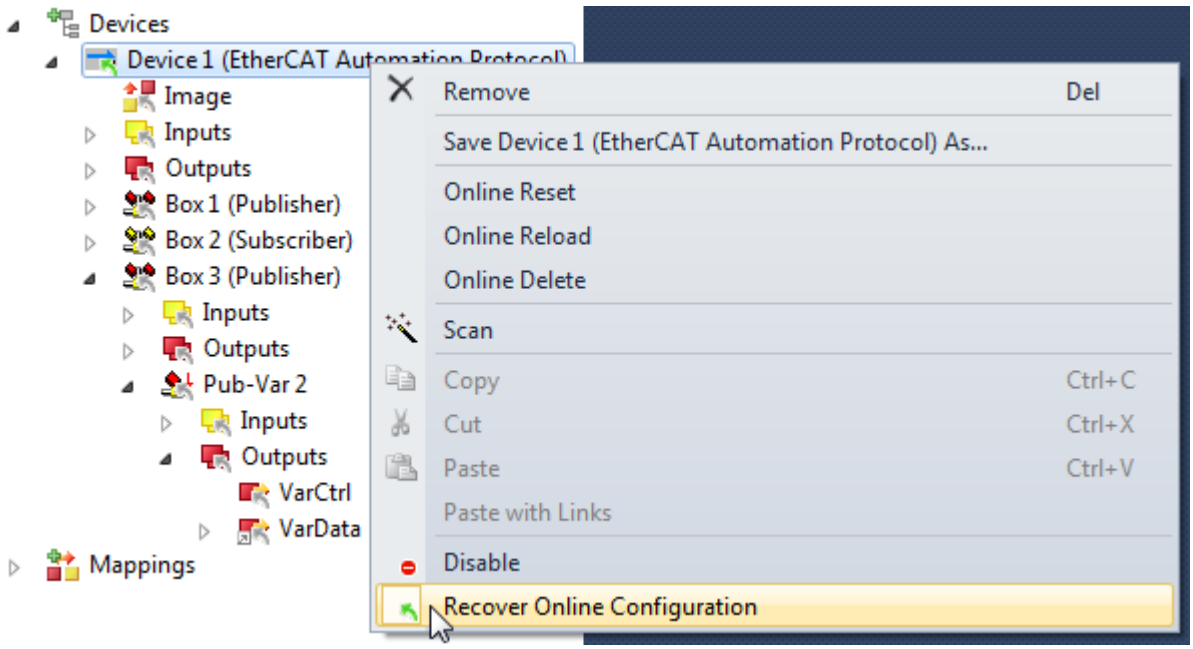
Auf der Server-Seite wird für den Empfang der Anfrage ein *RxPD* konfiguriert, das wie im Pushed Data Exchange Modus arbeitet. Für das Versenden der Antwort auf die Client-Anfrage werden in diesem Beispiel zwei *TxPDs* konfiguriert. Bei beiden *TxPDs* wird bei der Eigenschaft Poll Request *RxPD* der Index des *RxPD* eingetragen, welches als Empfänger für die Anfrage dient. In jedem Verarbeitungszyklus prüft ein *TxPD* bei seinem konfigurierten Poll Request *RxPD*, ob eine Anfrage empfangen worden ist. Wenn ja, dann wird das *TxPD* in diesem Zyklus als Antwort (Response) zurückgesendet.



6.6 Wiederherstellen der Onlinekonfiguration

Wenn beim EAP-Gerät Parameter des *CANopen OD* online verändert werden, wird immer der aktuelle Stand des *CANopen OD* auf der Festplatte archiviert, sobald TwinCAT in den *Stop/Config* Modus zurückgeschaltet wird. Beim erneuten Starten von TwinCAT in den *Run* Modus spielt das angelegte Archiv üblicherweise keine Rolle. Denn es wird einfach die Konfiguration, die offline vorgenommen worden ist, wieder gestartet. Alle Änderungen, die online konfiguriert worden sind, werden verworfen.

Wenn bei einem Neustart von TwinCAT die online Konfiguration wiederhergestellt werden soll, dann muss in TwinCAT bei diesem EAP-Gerät die Option *Recover Online Configuration* eingeschaltet (siehe folgende Abbildung) und anschließend das Projekt neu aktiviert werden. In diesem Fall wird während des Startens der Steuerung zunächst die offline Konfiguration und nachträglich das Archiv von der Festplatte geladen, so dass am Ende des Startvorgangs die online Konfiguration aktiv ist.



Es ist nicht möglich die Konfiguration des TwinCAT EAP-Gerätes in TwinCAT zu verändern, wenn die Option *Recover Online Configuration* gesetzt ist. Dazu muss diese Option erst wieder deaktiviert werden. Dieser Sperrmechanismus dient allein dem Schutz, damit der Anwender nicht die offline Konfiguration ändert, obwohl diese Änderung beim Aktivieren der Steuerung keine Auswirkung haben würde, da ja die online Konfiguration wiederhergestellt wird.

Recover Online Configuration

Das archivierte *CANopen* Object Dictionary ist an die Instanz des EAP-Gerätes in TwinCAT gekoppelt. Sobald diese Instanz in TwinCAT gelöscht und stattdessen eine neue Instanz angelegt wird, kann das zuvor archivierte *OD* für diese Instanz nicht mehr wiederhergestellt werden.

7 Das EAP Device Configuration (EDC) File

Die Konfiguration eines TwinCAT EAP-Gerätes (engl. *EAP Device Configuration* = *EDC*) erfolgt mit Hilfe von TwinCAT. Es muss wenigstens das EAP-Gerät selbst über TwinCAT angelegt werden, so dass beim Starten von TwinCAT entsprechend eine Instanz des EAP-Gerätes erzeugt wird. Das Anlegen von *TxFrames, Tx/RxProcessData, Tx/RxPDOs* und *Tx/RxVariables* kann dann auch im Nachhinein von einer *ADS-Client* Anwendung durchgeführt werden. Dazu muss TwinCAT im *Run*Modus sein. Zumeist wird ein TwinCAT EAP-Gerät jedoch eher vollständig über TwinCAT konfiguriert und gestartet wie in Kapitel Das Anlegen einer EAP-Konfiguration [▶ 26] und Das Konfigurieren eines EAP-Gerätes [▶ 42] beschrieben.

Zur online Konfiguration eines EAP-Gerätes bzw. eines ganzen Netzwerks von EAP-Geräten ist ein Konfigurator mit grafischer Benutzerschnittstelle (GUI) entwickelt worden, der TwinCAT EAP Configurator (Produktbeschreibung TE1610). Dieser Konfigurator ermöglicht die grafische Darstellung aller im Netzwerk befindlichen EAP-Geräte und ihre Kommunikationsverbindungen zueinander. Zudem lässt sich jedes EAP-Gerät über die GUI konfigurieren.
Eine Konfiguration, die in TwinCAT erstellt und zu der eine EDC-Datei exportiert wurde, enthält alle notwendigen Informationen, die beim Import im Konfigurator benötigt werden, um die aktuelle Konfiguration des EAP-Gerätes anzuzeigen. Eine Dokumentation zum TwinCAT EAP Configurator finden Sie im Beckhoff Information System.

8 Support und Service

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

Downloadfinder

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

Die Downloads sind in verschiedenen Formaten erhältlich.

Beckhoff Niederlassungen und Vertretungen

Wenden Sie sich bitte an Ihre Beckhoff Niederlassung oder Ihre Vertretung für den lokalen Support und Service zu Beckhoff Produkten!

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

Dort finden Sie auch weitere Dokumentationen zu Beckhoff Komponenten.

Beckhoff Support

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

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

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

Beckhoff Service

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

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

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

Beckhoff Unternehmenszentrale

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20
33415 Verl
Deutschland

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

Trademark statements

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

Third-party trademark statements

Arm, Arm9 and Cortex are trademarks or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere.

Microsoft, Microsoft Azure, Microsoft Edge, PowerShell, Visual Studio, Windows and Xbox are trademarks of the Microsoft group of companies.

Mehr Informationen:
www.beckhoff.com/te1000

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