

EtherCAT – Der Ethernet-Feldbus

Funktionsweise und Eigenschaften – 1. Teil

Nachdem das Ethernet in der Automatisierung bereits in mehreren Varianten angeboten wird, liegt die Frage nahe: „Wieso noch ein weiterer Ethernet-Feldbus?“ Die Antwort lautet: „EtherCAT geht andere Wege und ist derzeit mit Abstand der leistungsfähigste und der am besten auf automatisierungstechnische Anforderungen zugeschnittene Ethernet-Ansatz!“ Eine fundierte, technisch begründete Antwort gibt dieser zweiteilige Artikel durch die detaillierte Erläuterung des Funktionsprinzips und der Eigenschaften von EtherCAT.

Von Dr.-Ing. Dirk Janssen und Holger Büttner

In der Automatisierungstechnik nimmt der Einsatz von Ethernet als physikalische Kommunikationsschicht weltweit rasant zu. Beginnend mit der Leitebene, über die Fabrikvernetzung bis hin zur Kommunikation in der Steuerung selbst ist Ethernet bereits etabliert und, wegen der weiten Verbreitung in der Office-Welt mit hohen Komponenten-Stückzahlen, vergleichsweise leistungsfähig und kostengünstig. In der Feldebene dominieren allerdings weiterhin die in den 90er Jahren entwickelten Feldbusse. Mit unterschiedlichen Schwerpunkten haben sich diese ihr jeweiliges Marktsegment geschaffen, und mit Datenübertragungsraten von 0,5 bis 16 Mbit/s sind sie auch weiterhin für viele Anwendungen ausreichend. Die, im Vergleich zur Office-Welt, verschwindend

geringen Stückzahlen führen jedoch zu hohen Gesamtkosten, sowohl in der Steuerung als auch in den Feldgeräten und insbesondere bei der Verkabelung.

► Warum EtherCAT?

Während das Mooresche Gesetz – Verdoppelung der Leistungsfähigkeit ca. alle zwei Jahre – zumindest bei den PC-basierten Steuerungen in den nächsten Jahren auch weiterhin gelten wird, gab es bei den Feldbussen keine nennenswerten Weiterentwicklungen. Stattdessen wurde das Ethernet als kommende Ergänzung bzw. Ersatz für die „proprietäre“ Feldbustechnik angesehen und sowohl von Feldbus-Organisationen als auch von „großen“ Automatisierungsfirmen ausgewählt; entsprechende Standards wurden entworfen.

Ethernet hat im Vergleich zu den existierenden Feldbussen nicht nur Vorteile. Die für den Einsatz in der Automatisierungstechnik eher ungünstigen Eigenschaften des Ethernet müssen daher sehr genau betrachtet und möglichst optimal umgangen werden. Darauf beruhen auch die wesentlichen Unterschiede der verschiedenen Ansätze, Ethernet für die Automatisierungstechnik anzupassen. Die Schwachstellen sind:

- Hoher Overhead bei der Kommunikation mit Teilnehmern, die sehr häufig,

aber geringe Datenmengen austauschen müssen.

- Hohe Anschaltkosten pro Teilnehmer im Vergleich zu klassischen Feldbusanschlüssen. Ethernet benötigt einen Übertrager, eine aufwendige physikalische Schnittstelle, zusätzlich einen Media Access Controller (MAC) und eine hohe Prozessorleistung.

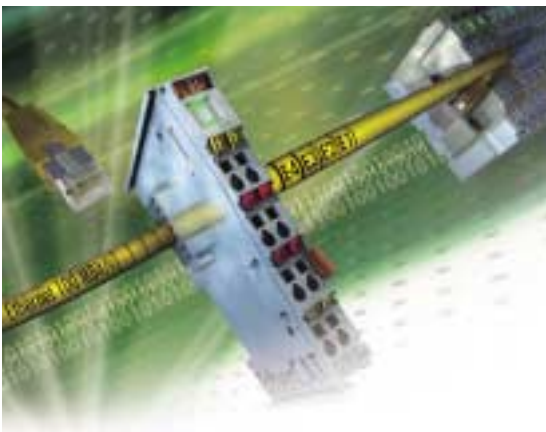
- Mangelnde Echtzeit-Fähigkeit, die bei genauerer Betrachtung aber weniger durch das Übertragungsmedium verursacht wird, sondern auf die ungünstigen Laufzeiten in den Software-Protokollstapeln zurückgeht.

- Ungünstige Topologie: Die inzwischen bei Ethernet übliche sternförmige Topologie ist bei der Anlagenverdrahtung eher ungünstig und führt entweder zu hohem Verkabelungsaufwand oder zu stark kaskadierten Kommunikationsbeziehungen.

Ethernet ist auch nicht gleich Ethernet. Neben den unterschiedlichen Übertragungsgeschwindigkeiten von 10, 100 und 1000 Mbit/s muss auch zwischen Halb- und Vollduplex-Kommunikation unterschieden werden. Halbduplex – also die Datenübertragung, die abwechselnd in die eine und in die andere Richtung geht – hat im Vergleich zu Vollduplex weniger als die Hälfte der Übertragungskapazität, da nach der Übertragung in die eine Richtung Pausenzeiten eingehalten werden müssen, um die Lauf- und Reaktionszeiten zu kompensieren. Bei Halbduplex-Übertragungen können Kollisionen auftreten, die zwar auf den unteren Schichten abgefangen und durch Wiederholungen beseitigt werden, aber für eine deterministische Übertragung nicht akzeptabel sind. Daher müssen die Kollisionen auf den höheren Protokoll-Schichten vermieden werden, was aber wiederum nicht ohne weitere Verschlechterung der Nutzdatenrate möglich ist.

► Entwicklungsziele von EtherCAT

PC-basierte Steuerungstechnik kann aufgrund der Leistungsfähigkeit des PC-Systems einen „eher zentralen“ oder, genauer ausgedrückt, einen hierarchischen Ansatz unterstützen, bei dem alle auf einer Ebene relevanten Informationen gleichzeitig in einer Steue-



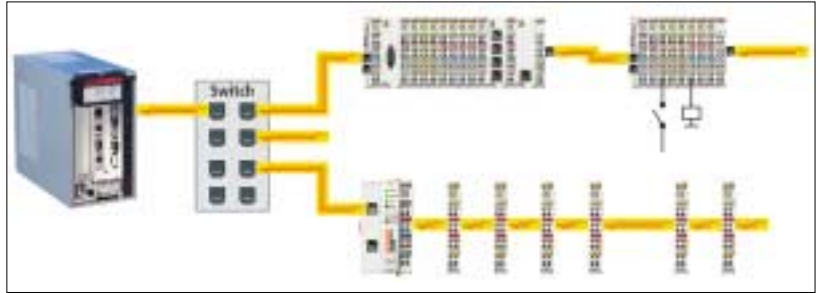


Bild 1. EtherCAT als „großer“ Ethernet-Teilnehmer im normalen Netz. Während im oberen Zweig die Automatisierungsgeräte je einen eigenen Ethernet-Controller benötigen, werden im „EtherCAT-Zweig“ (unten) die EtherCAT-Slaves durch das EtherCAT-Verfahren angekoppelt.

rung verfügbar sind und miteinander kombiniert werden können. Dies erleichtert nicht nur die Konfiguration des Systems, sondern ermöglicht auch den Einsatz intelligenterer Steuerungsalgorithmen. Das Kommunikationssystem ist dabei „nur noch“ für den schnellen Transport der Prozessdaten in und aus der Steuerung verantwortlich. Ein Master-Slave-Kommunikationssystem ist für diesen Zweck die beste Wahl.

Bei der Entwicklung von EtherCAT standen die folgenden Hauptziele im Vordergrund:

- Breite Einsetzbarkeit. Als EtherCAT-Master soll jede Steuerung mit einem handelsüblichen Ethernet-Controller eingesetzt werden können. Angefangen von einem kleinen 16-bit- μ C bis hin zu PC-Systemen mit 3-GHz-Prozessoren soll jeder Rechner ohne spezielle Anschaltung zur EtherCAT-Steuerung werden können.
- Vollständige Konformität zum Ethernet-Standard. EtherCAT soll mit anderen Ethernet-Geräten und -Protokollen am selben Bus koexistieren. Standard-Strukturkomponenten wie Ethernet Switches sollen für EtherCAT einsetzbar sein.
- Kleinstmögliche Teilnehmer-Granularität ohne unterlagerten Sub-Bus. Als EtherCAT-Slave sollen sowohl komplexere Knoten als auch 2-bit-E/As wirtschaftlich eingesetzt werden können.

- Höchstmögliche Effizienz. Die Bandbreite von Ethernet soll möglichst vollständig für Nutzdatentransfers zur Verfügung stehen.

- Kurze Zykluszeiten. Mögliche Zykluszeiten deutlich unter 100 μ s sollen neue Anwendungsgebiete erschließen, wie z.B. das Schließen unterster Regelkreise in der Antriebstechnik.

- Höchste „Deterministik“, auch ohne auf absoluter Telegramm-Sendegenauigkeit zu basieren.

Das Funktionsprinzip

Aus Ethernet-Sicht ist ein EtherCAT-Bus nichts anderes als ein einzelner großer Ethernet-Teilnehmer. Dieser „Teilnehmer“ empfängt und sendet Ethernet-Telegramme (*Bild 1*). Innerhalb des „Teilnehmers“ befindet sich aber kein Ethernet-Controller mit nachgeschaltetem Mikroprozessor, sondern eine Vielzahl von EtherCAT-Slaves. Diese verarbeiten die einlaufenden Telegramme im Durchfluss und nehmen die für sie bestimmten Nutzdaten heraus bzw. blenden sie ein und leiten das Telegramm an den nächsten EtherCAT-Slave weiter. Der letzte EtherCAT-Slave schickt das bereits vollständig verarbeitete Telegramm zurück, so dass es vom ersten Slave – quasi als Antwort-Telegramm – zur Steuerung zurückgeschickt wird. Es wird dabei ausgenutzt, dass Ethernet eine getrennte Übertra-

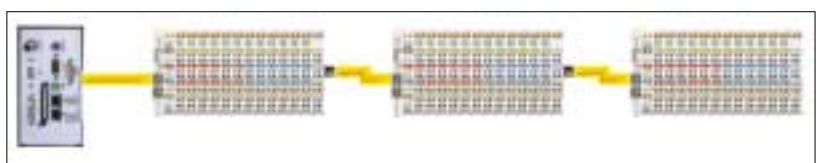


Bild 2. Eine Ankopplung an das Ethernet ist auch ohne Switch möglich. Dabei wird das EtherCAT-Gerät über ein „gedrehtes“ Ethernet-Kabel an die Steuerung (links) angeschlossen.

gung in Hin- und Rück-Richtung (Tx- und Rx-Leitungen) besitzt und im Voll-duplex-Modus arbeitet.

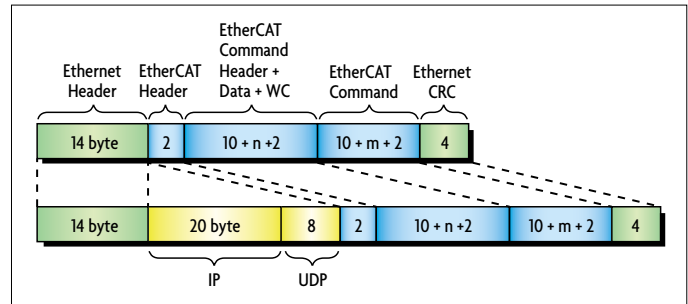
Natürlich kann, wie bei jedem anderen Ethernet-Teilnehmer auch, eine direkte Kommunikation ohne Switch mit einem „gedrehten“ Ethernet-Kabel aufgebaut werden, wodurch ein reines EtherCAT-System entsteht (Bild 2).

Telegramm-Verarbeitung

Die Verarbeitung der Telegramme findet im Durchlauf statt. Während die Telegramme – nur um wenige Bits verzögert – bereits weitergeschickt werden, erkennt der Slave die für ihn bestimmten Kommandos und führt sie entsprechend aus. Die Verarbeitung findet in Hardware statt und ist daher unabhängig von den Reaktionszeiten eventuell angeschlossener Mikroprozessoren. Jeder Teilnehmer besitzt dabei einen adressierbaren Speicherbereich von 64 Kbyte, innerhalb dessen gelesen, geschrieben oder gleichzeitig geschrieben und gelesen werden kann.

In einem Ethernet-Telegramm können mehrere EtherCAT-Kommandos eingebettet werden, die jeweils individuelle Teilnehmer und/oder Speicherbereiche ansprechen. Die EtherCAT-Kommandos werden im Datenbereich eines Ethernet-Telegramms transportiert und können entweder über einen speziellen „Ether Type“ oder per UDP/IP codiert sein (Bild 3).

Die erste Variante mit speziellem Ether Type ist zwar auf ein Ethernet-Subnetz begrenzt, d.h., entsprechende Telegramme werden von Routern nicht weitergeleitet, für Steuerungsaufgaben stellt dies in der Regel jedoch keine Einschränkung dar. Als Adressierung wird die „Ethernet MAC“-Adresse des ersten Teilnehmers genutzt. Hierbei wird ein spezieller erster EtherCAT-Teilnehmer benötigt, der im Falle einer direkten Ansteuerung ohne Switch entfallen kann.



Die zweite Variante per UDP/IP erzeugt einen geringfügig größeren Overhead (IP Header und UDP Header), ermöglicht aber bei weniger zeitkritischen Anwendungen, einerseits das normale IP Routing zu nutzen und andererseits die häufig bereits vorhandenen TCP/IP-Protokollstapel auf der Master-Seite zu verwenden. Jedes EtherCAT-Kommando besteht aus einem EtherCAT-Header, dem Datenbereich und einem anschließenden Zählerfeld – dem „Working Counter“ –, das von all den EtherCAT-Teilnehmern inkrementiert wird, die durch das

Bild 3. Der EtherCAT-Telegrammaufbau (ohne und mit UDP) lässt den Ethernet Header unberührt; für die Internet-Übertragung werden IP Header und UDP Header vor die EtherCAT-Daten eingeschoben.

Übertragungsschicht	Kabel	max. Länge	Kosten (Anschaltung/ lfd. Meter)	Konfektionierbarkeit	Verzögerung pro Anschaltung
Ethernet (100Base-TX)	CAT 5 (Kupfer)	100 m	o/++(+*)	+	ca. 1 µs
Ethernet (100Base-FX)	LWL (Glasfaser)	100 km	-/- -	-	ca. 1 µs
E-Bus (Kupfer)	CAT 5 (Kupfer)	10 m	++/++(+*)	+	0 µs
E-Bus (LWL, in Planung)	LWL (Plastikfaser)	100 m	+/-	++	0 µs

*) industrielle Kabelausführung (schleppkettentauglich, ölfest)

Eigenschaften der verschiedenen Übertragungsmedien des EtherCAT.

EtherCAT-Kommando angesprochen wurden und entsprechende Daten ausgetauscht haben.

Reduzierung des Overhead

Mit Hilfe des oben beschriebenen Telegrammaufbaus können bereits mehrere EtherCAT-Teilnehmer durch ein einzelnes Ethernet-Telegramm mit mehreren EtherCAT-Kommandos angesprochen werden, was zu einer deutlichen Verbesserung der Nutzdatenrate führt. Für den Fall einer 2-bit-Eingangsklemme, die eben genau 2 bit Nutzdaten bereithält, ist der Overhead eines einzelnen EtherCAT-Kommandos aber weiterhin deutlich zu groß.

Die „Fieldbus Memory Management Unit“ (FMMU) beseitigt dieses Problem und ermöglicht eine nahezu hundertprozentige Nutzdatenrate – auch bei Teilnehmern, die, wie beschrieben, nur 2 bit Nutzdaten aufweisen. Ähnlich wie eine Memory Management Unit (MMU) in modernen Prozessoren übersetzt die FMMU eine logische Adresse anhand einer internen Tabelle in eine physikalische. Die FMMU ist im EtherCAT-„Slave ASIC“ integriert und ermöglicht für jeden Teilnehmer individuell ein entsprechendes Adress-Mapping. Im Unterschied zu prozessorinternen MMUs, die komplette Speicherseiten (im Bereich von 4 Kbyte) mappen, unterstützt die FMMU auch bitweises Mappen. Dadurch können die zwei Bits der Eingangsklemme individuell in einen beliebigen Bereich eines logischen Adressraumes eingeblendet werden. Wird nun ein EtherCAT-Kommando verschickt, das, anstatt einen speziellen EtherCAT-Teilnehmer anzusprechen, einen bestimmten logischen Speicherbereich liest oder schreibt, blendet die 2-bit-Eingangsklemme ihre Daten an der richtigen Stelle in den Datenbereich ein. Alle anderen Klemmen, die eine

Übereinstimmung der Adresse (Address Match) in ihrer eigenen FMMU feststellen, blenden ebenfalls ihre Daten ein, so dass mit einem einzelnen EtherCAT-Kommando viele Teilnehmer gleichzeitig angesprochen werden.

Da eine FMMU in jedem Teilnehmer vorhanden ist und dort individuell konfiguriert wird, kann der Master in der Initialisierungsphase bereits komplette Prozessabbilder zusammensetzen und anschließend mit einem einzelnen EtherCAT-Kommando austauschen. Es wird kein zusätzliches Mapping mehr im Master benötigt, so dass die Prozessdaten direkt den unterschiedlichen Steuerungs-Tasks (SPS, NC etc.) zugeordnet werden können. Jede Task kann ihr eigenes Prozessabbild erstellen und in ihrem eigenen zeitlichen Rahmen austauschen. Die physikalische Reihenfolge der EtherCAT-Teilnehmer ist dabei völlig unabhängig und spielt nur in der allerersten Initialisierungsphase eine Rolle.

Serielle Backplane

Der logische Adressraum für die FMMUs beträgt 4 Gbyte. Aus Sicht des Masters kann ein EtherCAT-System daher als großer verteilter Speicher (Distributed Memory) angesehen werden, der wahlfrei beschrieben und gelesen werden kann. EtherCAT ist daher eine Art serieller Backplane für Automatisierungssysteme, die für große, aber auch ganz kleine Automatisierungsgeräte die Anbindung an verteilte Prozessdaten ermöglicht. Über einen Standard-„Ethernet Controller“ und Standard-Ethernet-Kabel (CAT 5) können quasi beliebig viele und beliebig verteilte E/A-Kanäle an Automatisierungsgeräte angeschlossen werden, auf die mit hoher Bandbreite, geringster Verzögerung und nahezu optimaler Nutzdatenrate zugegriffen werden kann.

► EtherCAT-Eigenschaften im Einzelnen

Für die kurze Verbindung: der E-Bus

Für Ethernet existiert eine Vielzahl unterschiedlicher Medien; angefangen vom in die Jahre gekommenem „Yellow Cable“ bis zu Hochgeschwindigkeits-Glasfaser-Strecken. Bei dem von EtherCAT verwendeten 100-Mbit/s-Ethernet gibt es – neben dem CAT-5-Kupferkabel (100Base-TX) – auch Übertragungsprotokolle für Lichtwellenleiter (100Base-FX). Da EtherCAT vollständig kompatibel zu Ethernet ist, können alle entsprechenden Übertragungsschichten genutzt werden. Bei EtherCAT erfolgen häufig Übertragungen auf sehr kurzer Strecke, z.B. zwischen zwei EtherCAT-Klemmen im selben Klemmenblock, so dass zusätzlich eine weitere Übertragungsschicht entwickelt wurde: der „E-Bus“. Der E-Bus basiert auf einer LVDS-Übertragung (Low Voltage Differential Signaling, IEEE-Standard P1596.3-1995) und ist neben der Klemmen-Kommunikation auch für kostengünstige „Fernübertragungen“ bis ca. 10 m geeignet.

Da es sich bei den übertragenen Daten jederzeit um vollständige Ethernet-

Telegramme handelt, kann die physikalische Übertragungsschicht an jeder Stelle und beliebig oft gewechselt werden. Bezogen auf ein Beispiel mit verschiedenen Schaltschränken und Maschinenmodulen kann die jeweils kostengünstigste Übertragungsschicht genutzt werden: Innerhalb eines Schaltschranks reicht der E-Bus; zwischen diesem und den Maschinenmodulen kommt die normale Ethernet-Kupfer-Physik mit bis zu 100 m zum Einsatz. Bei noch größeren Entfernungen oder extremen EMV-Belastungen kann wiederum an jeder Stelle auf Lichtwellenleiter (LWL) umgesetzt werden.

Einzige Voraussetzung an das Medium ist die Vollduplex-Fähigkeit, da EtherCAT so schnell reagiert, dass in der Regel bereits die Antwort zum Master zurückgesendet wird, während der Master noch die letzten Bytes seiner Anfrage absendet. Halbduplex-Übertragungsschichten, wie sie z.B. bei der Funkübertragung genutzt werden, würden Kollisionen erzeugen und können für EtherCAT nicht eingesetzt werden. EtherCAT nutzt in der aktuellen Implementierung das 100-Mbit/s-Ethernet. Das Übertragungsprinzip ist davon unabhängig

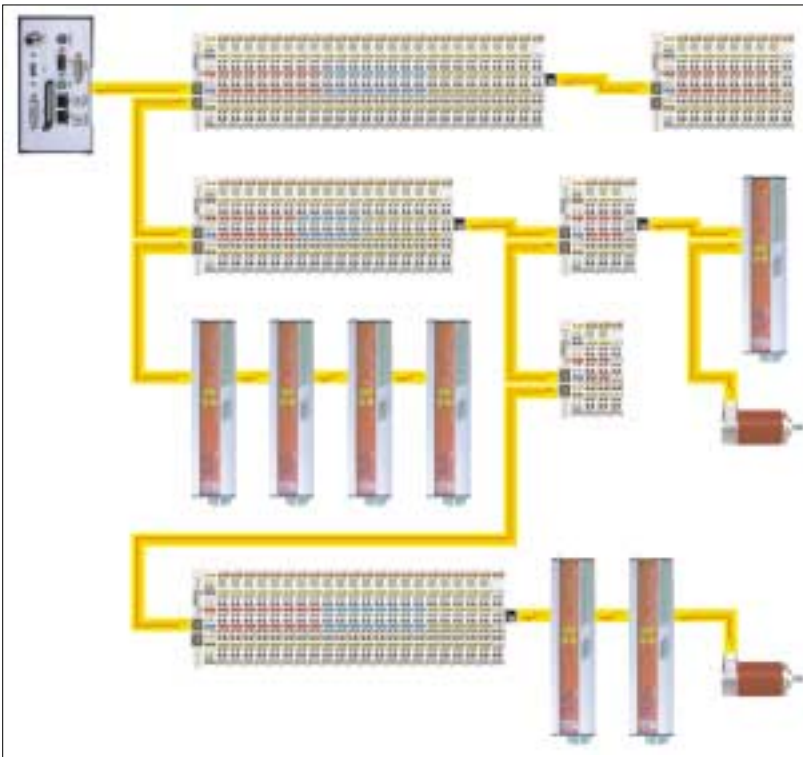


Bild 4. EtherCAT ermöglicht den Aufbau einer flexiblen Baumstruktur. Während der Hauptstrang von einem Modul zum nächsten läuft, können einzelne Äste in Schaltschränke oder Maschinenmodule verzweigen.

und kann zukünftig ohne Änderung auf 1 Gbit/s oder mehr angewendet werden (s. *Tabelle*).

► Topologie beim EtherCAT

Die Topologie eines Kommunikationssystems ist mit ausschlaggebend für den erfolgreichen Einsatz in der Automatisierungsindustrie. Die Topologie hat großen Einfluss auf den Aufwand bei der Verkabelung, für die Diagnose-Eigenschaften, die Möglichkeiten zur Einführung von Redundanzen und die „Hot Plug and Play“-Eigenschaften. Die bei Standard-Ethernet (100Base-TX) übliche Sternverkabelung hat zwar Vorteile bezüglich des „Hot Plug and Play“ und der Redundanz, der Aufwand für die Verkabelung und die Zahl der erforderlichen Switches sind aber in verteilten Anwendungen mit vielen Teilnehmern kaum akzeptabel.

Bei EtherCAT stellen die Slaves logisch gesehen einen offenen Ringbus dar. Am offenen Ende schickt der Master, entweder direkt oder über Standard-„Ethernet Switches“, Telegramme hinein und erhält sie am anderen Ende bearbeitet wieder zurück. Alle Telegramme werden vom ersten Teilnehmer an die nächsten weitergeleitet, vom letzten wird das Telegramm wieder zurück zum Master gesendet. Da ein normales Ethernet-Kabel bidirektional aufgebaut ist (getrennte Tx- und Rx-Leitungen) und auch alle EtherCAT-Slaves in der Rückrichtung übertragen können, ergibt sich ein physikalischer Strang.

Durch Abzweigungen, die prinzipiell an jeder Stelle möglich sind, kann aus der Strangstruktur eine flexible Baumstruktur aufgebaut werden. Eine Baumstruktur ermöglicht einfachste Verkabelungen; so können einzelne Äste z.B. in Schaltschränke oder Maschinenmodule verzweigen, während der Hauptstrang von einem Modul zum nächsten läuft (*Bild 4*).

► Die Telegramm-Übertragung

Jeder EtherCAT-Slave hat zwei Rx- und zwei Tx-Schnittstellen, über die die Telegramme jeweils in der Hin-Richtung und in der Rück-Richtung geleitet werden (*Bild 5a*). Die Auswertung der Telegramme erfolgt immer in Hin-Richtung, die Rück-Richtung dient zum Verstärken und Regenerieren des Signals sowie zum Lokalisieren und Schließen von Leitungsunterbrechungen.

Ein EtherCAT-Slave kann erkennen, ob auf seiner Hin- bzw. Rück-Leitung ein Trägersignal anliegt. Weiterhin besitzt jeder EtherCAT-Slave die Eigenschaft, den Datenfluss so umschalten zu können, dass ein über die Hin-Leitung empfangenes Telegramm sowohl über die Tx-Schnittstelle der Hin-Leitung als auch die der Rück-Leitung gesendet werden kann. Umgekehrt kann ein über die Rück-Leitung empfangenes Telegramm über die Tx-Schnittstelle der Rück- oder der Hin-Leitung geschickt werden (*Bild 5b*).

Wenn der EtherCAT-Slave auf seiner Rückleitung kein Trägersignal mehr erkennt, schaltet er die Tx-Schnittstelle der Hin-Leitung auf die Rx-Schnittstelle der Rück-Leitung kurz, d.h., ein über die Hin-Leitung empfangenes Telegramm wird bearbeitet und dann über die Tx-Schnittstelle der Rückleitung zurückgesendet (*Bild 5c*). Dadurch werden keine zusätzlichen Abschlussmodule benötigt; der EtherCAT-Slave erkennt automatisch, dass nach ihm kein weiterer Slave folgt. Da dieser Zustand auch bei einer kurzzeitigen Unterbrechung auftreten kann, wird auch im Kurzschlussfall versucht, ein Trägersignal bzw. das gerade im Durchlauf befindliche Telegramm über die Tx-Schnittstelle der

Hin-Leitung zu senden, um eine wieder geschlossene Unterbrechung zu erkennen. Über die Rx-Schnittstelle der Rück-Leitung wird dann wieder ein Trägersignal erkannt und der Kurzschluss aufgehoben.

Eine Unterbrechung kann natürlich auch auf der Hin-Leitung auftreten. Erkennt der EtherCAT-Slave an der Rx-Schnittstelle seiner Hin-Leitung kein Trägersignal mehr, so schließt er die Tx-Schnittstelle der Rückleitung mit der Rx-Schnittstelle der Hin-Leitung kurz (*Bild 5d*). Über die Tx-Schnittstelle der Rückleitung wird in diesem Fall aber kein Trägersignal bzw. Telegramm mehr gesendet.

Abzweige in der Übertragungsleitung

Durch das automatische Erkennen eines Trägersignals können auch relativ einfache Abzweige realisiert werden, sofern diese auch über die Send- und Empfangsfunktionen eines EtherCAT-Slaves verfügen. In diesem Fall gibt es jeweils drei Rx- bzw. Tx-Schnittstellen (Hin-Leitung, Rück-Leitung und Abzweig). Die Hin-Leitung ist dabei immer kurzgeschlossen, d.h., ein Telegramm geht ohne Bearbeitung von der Rx-Schnittstelle der Hin-Leitung auf die Tx-Schnittstelle der Hin-Leitung (*Bild 6a*). Die Rx-Schnittstelle der Rück-Leitung und die Tx-Schnittstelle des Abzweigs nehmen jetzt die Stellung

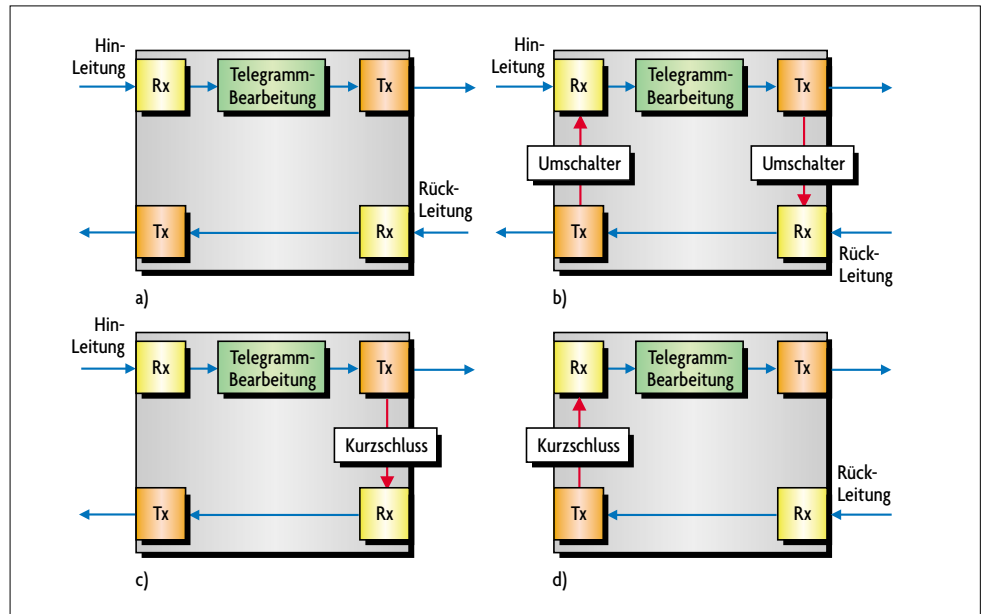


Bild 5. Zustände der EtherCAT-Slaves. Ein an der Hin-Leitung empfangenes Telegramm wird bearbeitet und weitergereicht (a). Umschalter zwischen den Rx- und Tx-Schnittstellen erlauben das Rangieren der Telegramme (b). Bei einer Unterbrechung schließt der letzte „intakte“ EtherCAT-Slave den Kommunikationsring, indem er die Tx-Schnittstelle seiner Hin-Leitung mit der Rx-Schnittstelle der Rück-Leitung kurzschließt (c). Liegt an der Rx-Schnittstelle der Hin-Leitung kein Signal an, wird die Tx-Schnittstelle der Rück-Leitung mit der Rx-Schnittstelle der Hin-Leitung kurzgeschlossen (d).

der Hin-Leitung bei einem „normalen“ EtherCAT-Slave ein, d.h., ein hierüber empfangenes Telegramm wird bearbeitet und über die Tx-Schnittstelle des Abzweigs gesendet. Die Rx-Schnittstelle des Abzweigs und die Tx-Schnittstelle der Rück-Leitung bekommen die Aufgaben der Rück-Leitung bei einem „normalen“ Slave. Solange kein Kabel

an den Abzweig angeschlossen ist, werden Tx- und Rx-Schnittstelle des Abzweigs kurzgeschlossen, d.h., ein bearbeitetes Telegramm wird über die Tx-Schnittstelle der Rück-Leitung gesendet (*Bild 6b*).

Sobald ein Kabel an den Abzweig angeschlossen ist, wird am Abzweig ein Trägersignal erkannt und der Kurz-

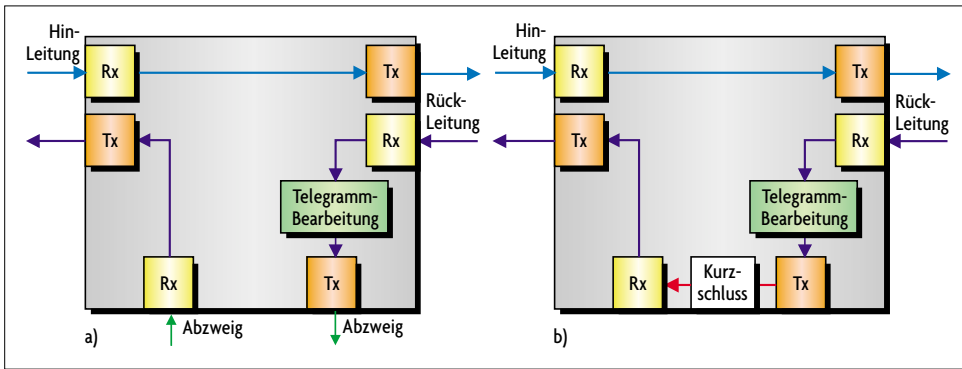


Bild 6. Realisierung eines Abzweigs. Mit der automatischen Erkennung des Trägersignals lässt sich ein Abzweig einfach in die Rück-Leitung des EtherCAT-Slaves einfügen (a). Solange kein Kabel an den Abzweig angeschlossen ist, werden Tx- und Rx-Schnittstelle des Abzweigs kurzgeschlossen (b).

schluss aufgehoben. Da das Stecken oder Ziehen eines Abzweigs während des Betriebs erlaubt ist (im Gegensatz dazu stellt das Unterbrechen einer Linie einen Fehlerzustand dar), muss die Möglichkeit bestehen, das Stecken bzw. Ziehen rückwirkungsfrei durchführen zu können. Dazu gibt es ein Control-Bit im EtherCAT-Slave, mit dem das automatische Erzeugen bzw. Aufheben eines Kurzschlusses einer Übertragungsleitung deaktiviert werden kann. Außerdem verfügt jeder Abzweig über einen Schalter und eine LED, mit denen das Ziehen angefordert und bestätigt wird, so dass der Master die Möglichkeit hat, eine der Durchlaufzeit durch den Abzweig entsprechende zeitliche Lücke zu lassen und den Kurzschluss der Übertragungsleitung definiert durchzuführen.

Diagnose

Die Diagnosemöglichkeiten sind für den praktischen Einsatz eines verteilten Bussystems sicher ebenso entscheidend wie Performance-Daten, Topologie-Eigenschaften oder Verkabelungsaufwendungen. Auch hier erfüllt EtherCAT die Erwartungen an ein modernes Kommunikationssystem. Im Gegensatz zu „Party Line“-Bussystemen – z.B. Profibus und CAN –, bei denen die Teilnehmer alle am selben physikalischen Kabel hängen und die Signale an alle Teilnehmer ohne Auffrischung gesendet werden, kommt bei Ethernet (zumindest ab 100 Mbit/s) und natürlich auch bei EtherCAT eine Punkt-zu-Punkt-Übertragung zum Einsatz. Dadurch lassen sich Fehlerstellen oder auch nur sporadische Schwachstellen exakt lokalisieren, die sich bei „Party Line“-Systemen entweder nur mit speziellen Messaufbauten oder aber gar nicht finden lassen.

der Master die Bruchstelle exakt lokalisieren kann. Aufgrund der verteilten physikalischen Adressen bzw. der logischen Adressierung in den Teilnehmern ist dann weiterhin ein unveränderter Prozessdatenaustausch mit den verbleibenden Teilnehmern möglich.

Sicherungsmaßnahmen

Bei EtherCAT wird mittels Prüfsumme geprüft, ob ein Telegramm korrekt übertragen und von allen adressierten Teilnehmern mittels Working Counter korrekt bearbeitet wurde. Als Prüfsumme wird die normale Ethernet-Prüfsumme benutzt, die sich am Ende des Ethernet-Telegramms befindet. Da ein oder mehrere Slaves das Telegramm während der Übertragung modifizieren, wird die Prüfsumme in jedem Slave neu berechnet. Bei erkanntem Fehler bei der Prüfsumme wird ein Statusbit im EtherCAT-Slave gesetzt und ggf. ein Interrupt zum Master ausgelöst, so dass eine Fehlerstelle genau lokalisiert werden kann.

Bei einer Leseoperation werden die adressierten Daten vom Slave in das dafür vorgesehene Datenfeld eingefügt und beim Schreiben entsprechend entnommen. In beiden Fällen inkrementiert der adressierte Slave einen Working Counter, der sich am Ende eines jeden EtherCAT-Kommandos befindet. Da der Master weiß, wie viele Slaves von dem Telegramm adressiert sind, kann er anhand des Working Counter erkennen, ob alle Slaves ihre Daten korrekt ausgetauscht haben.



Dr.-Ing. Dirk Janssen

studierte Maschinenbau an der Technischen Universität Braunschweig. Nach seiner Promotion im Bereich des Software-Designs für Automatisierungssysteme ist er seit 1996 bei Beckhoff in Verl tätig. Als Leiter der Software-Entwicklung beschäftigt er sich u.a. intensiv mit dem Einsatz und der Konfiguration von Feldbussen.

► E-Mail: info@beckhoff.de



Dipl.-Ing. Holger Büttner

studierte Elektrotechnik an den Technischen Universitäten Hannover und Karlsruhe von 1986 bis 1991. Nach dem Studium entwickelte er bei der Firma TMG in Karlsruhe Profibus-Protokollsoftware. Seit 1996 ist er bei Beckhoff in der Niederlassung Berlin für die Feldbus-Entwicklung zuständig.

Bruchstellendiagnose

Bruchstellen im logischen Kommunikationsring werden automatisch lokalisiert und geschlossen. Jeder Teilnehmer überwacht die Trägersignale sowohl auf der Hinals auch auf der Rück-Richtung und kann entsprechende Störungen feststellen. Werden ein Kabelbruch oder der Ausfall eines Nachbarn erkannt, schließt der Teilnehmer davor den Kommunikationsring, indem er seine Hin-Richtung mit der Rück-Richtung kurzschließt. Dadurch entsteht immer ein kommunikationsfähiger geschlossener Ring, über den

► Fortsetzung folgt

Im zweiten Teil des Artikels werden weitere Eigenschaften von EtherCAT vorgestellt. Insbesondere werden die unterschiedlichen Adressierungsarten und die einzelnen EtherCAT-Kommandos beschrieben. Neben „Distributed Clocks“, Querverkehr, Redundanz und speziellen EtherCAT-Teilnehmern (Hub- bzw. Switch-Klemme, Feldbus-Master-Klemme etc.) werden die unterschiedlichen ASIC-Schnittstellen erläutert, die es den Komponentenherstellern leicht machen, entsprechende Anschaltungen zu entwickeln. *jw*

Weitere Infos:

- [1] www.ethercat.de
- [2] www.beckhoff.de

EtherCAT – Der Ethernet-Feldbus

Adressierung und Aufbau der Protokolle – 2. Teil

Im ersten Teil dieses Artikels in Heft 23, S. 64ff., wurden die Motivation zur Entwicklung des Ethernet-Feldbusses EtherCAT erläutert und die grundsätzlichen Eigenschaften des Systems beschrieben. Der zweite Teil stellt im Einzelnen die technische Realisierung vor: Der Schwerpunkt liegt hier auf der Adressierung der Teilnehmer – bzw. der Speicherbereiche in den Teilnehmern – und dem damit zusammenhängenden Protokollaufbau der Ethernet-Telegramme.

Von Dr. Dirk Janssen und Holger Büttner

An einen Feldbus werden unterschiedliche Anforderungen gestellt, was die Übertragungseigenschaften von Daten betrifft. Parameterdaten werden azyklisch und in großen Mengen übertragen, wobei die zeitlichen Anforderungen relativ unkritisch sind und die Übertragung in der Regel von der Steuerung angestoßen wird. Diagnosedaten werden ebenfalls azyklisch und ereignisgesteuert übertragen; die zeitlichen Anforderungen sind aber höher und die Übertragung wird in der Regel von einem Peripheriegerät angestoßen. Prozessdaten dagegen werden zyklisch mit verschiedenen Zykluszeiten übertragen. Dabei ist es wichtig, dass es nicht zu Zyklusausfällen kommt. Die zeitlichen Anforderungen sind bei der Prozessdaten-Kommunikation am höchsten.

Für die unterschiedlichen Kommunikationsarten gibt es bei EtherCAT entsprechende Adressierungsmöglichkeiten, die den Anforderungen jeweils optimal entsprechen.

Die Adressierungsvarianten

Physikalische Adressierung

Mit der physikalischen Adressierung wird ein Teil des 64 Kbyte großen Adressraums eines Slaves gelesen oder geschrieben. Mit einem Telegramm ist immer genau ein Slave adressiert. Diese Adressierung wird besonders zur

Übertragung von Parameterdaten genutzt. Zur Adressierung des Slaves gibt es zwei Varianten, die „Auto Increment“-Adressierung und die „Fixed Address“-Adressierung.

Auto Increment

Mit der „Auto Increment“-Adressierung kann jeder Slave anhand seiner Position im Kommunikationsring adressiert werden. Dabei inkrementiert jeder Slave das 16-bit-Adressfeld während des Telegrammdurchlaufs; adressiert ist der Slave, der ein Adressfeld mit dem Wert 0 empfängt. Will die Steuerung den zehnten Teilnehmer im Ring ansprechen, sendet sie ein Telegramm mit „Auto Increment“-Adressierung mit dem Startwert 9, der pro Teilnehmer um 1 inkrementiert wird. Die „Auto Increment“-Adressierung wird in der Regel nur in der Startup-Phase benutzt, in der der Master

die Stationsadressen an die Slaves verteilt. Danach können sie, unabhängig von ihrer Position, im Ring adressiert werden.

Diese Vorgehensweise bietet den Vorteil, dass bei den Slaves keine Stationsadressen manuell einzustellen sind. Ein nachträgliches Einfügen von Slaves führt nicht zu neuen Adressen.

Fixed Address

Bei der „Fixed Address“-Adressierung werden die Slaves über die vom Master in der Startup-Phase verteilte Stationsadresse angesprochen. Damit wird gewährleistet, dass auch bei einer Veränderung des Rings während des Betriebs die Slaves über die gleiche Adresse angesprochen werden können.

Logische Adressierung

Bei der logischen Adressierung wird kein Slave einzeln, sondern ein Teil eines 4 Gbyte großen logischen Adressraums angesprochen, der sich über beliebig viele Slaves verteilen kann. Die Zuordnung von physikalischen Adressen eines Slaves zu logischen Adressen im EtherCAT-Bus wird im Master projektiert und in der Startup-Phase an die Feldbus Memory Management Units (FMMU) der Slaves übertragen. Eine FMMU hat die Aufgabe, den physikalischen Adressen eines Teilnehmers eine logische Adresse zuzuordnen. Je FMMU wird eine logische, bitorientierte Startadresse, eine physikalische Startadresse, eine Bit-Länge sowie ein Typ konfiguriert, der angibt, in welcher Richtung gemappt werden soll (Inputs oder Outputs). Damit ist es möglich, beliebige Daten eines EtherCAT-Slaves bitweise auf beliebige logische Adressen zu map-

Byte	Bezeichnung	Bedeutung
1	CMD	Kommando: Identifiziert die im Folgenden beschriebenen Kommandos.
2	IDX	Index: Wird vom Slave unverändert weitergesendet. Damit kann der Master das Telegramm beim Empfang wieder leicht zuordnen.
3, 4	ADP	Address Page: Abhängig vom verwendeten Kommando.
5, 6	ADO	Address Offset: Abhängig vom verwendeten Kommando.
7, 8	LEN	Bit 0 – 10: Länge des Datenfeldes DATA. Bit 11 – 15: Flags.
9, 10	INT	Interrupt-Feld
11 – (n – 2)	DATA	Daten-Feld
(n – 1), n	CNT	Working Counter: Wird von jedem Slave inkrementiert, der adressiert war und das Telegramm erfolgreich bearbeiten konnte.

Tabelle 1. Die Byte-Struktur des EtherCAT-Datenpaketes. Die verfügbaren Kommandos (Byte 1) sind in Tabelle 2 zusammengestellt.

Kommando	Beschreibung	ADP (Address Page)	ADO (Address Offset)
NOP	Keine Bearbeitung	–	–
APRD	Lesen eines phys. Bereichs mit „Auto Increment“-Adressierung	Wird von jedem Slave inkrementiert; adressiert ist der Slave, der beim Empfang den Wert 0 feststellt.	Enthält die phys. Adresse innerhalb des Slaves, ab der sich die übertragenen Daten befinden.
APWR	Schreiben eines phys. Bereichs mit „Auto Increment“-Adressierung.		
APRW	Lesen und Schreiben eines phys. Bereichs mit „Auto Increment“-Adressierung.		
FPRD	Lesen eines phys. Bereichs mit „Fixed“-Adressierung.	Enthält die Stationsadresse des adressierten Slaves.	
FPWR	Schreiben eines phys. Bereichs mit „Fixed“-Adressierung.		
FPRW	Lesen und Schreiben eines phys. Bereichs mit „Fixed“-Adressierung.		
BRD	„Oder“-verknüpftes Lesen eines phys. Bereichs bei allen Slaves.	Jeder Slave ist adressiert und inkrementiert das ADP-Feld.	
BWR	Schreiben eines phys. Bereichs bei allen Slaves.		
BRW	„Oder“-verknüpftes Lesen und Schreiben eines phys. Bereichs.		
LRD	Lesen eines logischen Speicherbereichs.	Enthält das Hi-Wort der logischen 32-bit-Adresse, ab der sich die übertragenen Daten befinden.	Enthält das Lo-Wort der logischen 32-bit-Adresse, ab der sich die übertragenen Daten befinden.
LWR	Schreiben eines logischen Speicherbereichs.		
LRW	Lesen und Schreiben eines logischen Speicherbereichs.		
LRO	„Oder“-verknüpftes Lesen eines logischen Speicherbereichs.		
MRD	Lesen eines Speicherbereichs bei mehreren Slaves.	Enthält die Stationsadresse des ersten adressierten Slaves, der das MRD-Flag setzt. Ein Slave ist adressiert, wenn die Stationsadresse übereinstimmt oder das MRD-Flag gesetzt ist und der mit dem ADO-Feld und den MRD-Längen-Flags adressierte phys. Bereich definiert ist.	Enthält die physikalische Adresse innerhalb des Slaves, ab der sich die übertragenen Daten befinden.

Tabelle 2. Zusammenstellung der EtherCAT-Kommandos

pen. Der Slave überprüft beim Empfang eines Telegramms mit logischer Adressierung, ob eine seiner FMMUs eine Adressübereinstimmung (Match) hat, und fügt ggf. Daten an der entsprechenden Stelle des Datenfeldes in das Telegramm ein (Typ Inputs) bzw. entnimmt Daten von der entsprechenden Stelle aus dem Telegramm (Typ Outputs). Dadurch ist es möglich, Telegramme flexibel zusammenzustellen und an die Bedürfnisse der Steuerung optimal anzupassen. Damit eignet sich diese Adressierungsart besonders für die Übertragung zyklischer Prozessdaten.

Multiple-Adressierung

Mit der Adressierungsart „Multiple“ können physikalische Adressbereiche mehrerer Slaves gelesen werden. Der erste Slave wird dabei per Stationsadresse adressiert und schaltet das „Multiple Read Flag“ im Telegramm ein, über das die im Ring folgenden Slaves erkennen können, dass sie adressiert sind. Jeder Slave, der über den geforderten physikalischen Adressbereich verfügt, trägt seine Stationsadresse und die entsprechenden Daten in das Datenfeld ein, solange noch Platz ist.

Dazu wird in den ersten beiden Bytes des Datenfeldes ein „Working Pointer“ übertragen, der von jedem Slave, der Daten in das Datenfeld ein-

gefügt hat, inkrementiert wird. Damit ist es möglich, spezielle Diagnosedaten im Slave zu definieren, die mit der Multiple-Adressierung in der Regel mit einem Telegramm abgeholt werden können. Diese Adressierungsart eignet sich besonders für die Übertragung von azyklischen Diagnosedaten, da flexibel und schnell ausgewählte Daten von mehreren Teilnehmern gelesen werden können.

Interrupts

In jedem Telegramm befindet sich ein 16 bit breites Interruptfeld, mit dem sehr schnell und direkt Ereignisse gemeldet werden können. Mit diesem Interruptfeld können bis zu 16 Diagnoseadressen in den Slaves verknüpft werden. Wenn bei einem oder auch mehreren Slaves das entsprechende Ereignis auftritt, wird im nächsten empfangenen Telegramm von jedem Slave, bei dem das Ereignis aufgetreten ist, das entsprechende Interrupt-Bit gesetzt. Der Master kann daraufhin durch Senden eines Telegramms mit Multiple-Adressierung direkt die zugehörigen Diagnosedaten der entsprechenden Slaves lesen.

Mit den Interrupts können Ereignisse ohne Zeitverzögerung gemeldet werden. In Kombination mit der Multiple-Adressierung werden Diagnosedaten sehr schnell ereignisgesteuert übertragen, ohne die Übertragung zeitkriti-

scher Prozessdaten zu stören, da Sendezeitpunkt und Größe der zusätzlichen Telegramme vom Master gesteuert werden können.

Broadcast

Mit einem „Broadcast Write“ können physikalische Adressbereiche aller Slaves beschrieben werden; so lassen sich z.B. Zustandsübergänge bei allen Slaves gleichzeitig durchführen. Mit einem „Broadcast Read“ werden physikalische Adressbereiche aller Slaves über die Oder-Funktion verknüpft gelesen. Wenn eine Anwendung nur sinnvoll arbeiten kann, wenn sich alle Slaves im „OK“-Zustand befinden, kann z.B. dieser „OK“-Zustand sehr einfach und schnell mit einem „Broadcast Read“ von allen Teilnehmern abgefragt werden.

Telegrammaufbau

Es wird ein Standard-Ethernet-Telegramm mit einem speziellen Ether-Type benutzt. Innerhalb des Datenfeldes des Ethernet-Telegramms befinden sich ein oder mehrere EtherCAT-Kommandos. Jedes EtherCAT-Kommando hat einen Header mit einer Länge von 10 + 2 byte, der Aufbau ist in *Tabelle 1* dokumentiert.

EtherCAT-Kommandos

Tabelle 2 zeigt die von den EtherCAT-Teilnehmern unterstützten Komman-

EtherCAT Technology Group gegründet

Nachdem die Grundlagenentwicklung der Ethernet-Lösung EtherCAT (Ethernet for Control and Automation Technology), die von der Firma Beckhoff auf Basis des firmeneigenen „Lightbus“ entwickelt und erstmalig zur Hannover-Messe 2003 vorgestellt wurde, weitgehend abgeschlossen ist, soll das System offengelegt und jedem zugänglich gemacht werden. Dazu ist nun eine Anwendervereinigung gegründet worden; sie hat den Namen EtherCAT Technology Group (ETG) und ist innerhalb von nur drei Wochen bereits auf über 30 Mitgliedsfirmen angewachsen. Zum Zeitpunkt der SPS/IPC/Drives in Nürnberg, als die ETG offiziell vorgestellt worden ist, waren es exakt 34 Unternehmen aus dem In- und Ausland. Darunter befinden sich:

1. Anbieter von Automatisierungstechnik (Andrive, Aradex, Baldor, Baumüller, Beckhoff, Danaher, Fraba, Fronius, Hilscher, Kuka, MTS, Schmidhauser, Sigmatek, SND, Stöber, T+R, TAS, Turck, WST),
2. Anwender von Automatisierungstechnik (Applied Materials, Bruderer, Dieffenbacher, DLR, Finnpower, Focke, Heesemann, Husky, IMA, Komax, Müller Weingarten, Schuler) und

3. Endanwender von Automatisierungstechnik (Continental, Imperial Tobacco).

Laut Firmenchef Hans Beckhoff haben die ersten Pilotanwendungen die Vorteile im praktischen Einsatz nachgewiesen. So würden die ersten EtherCAT-Geräte in Applikationen eingesetzt, die nicht mit herkömmlichen Feldbussystemen – und auch mit keinem anderen Echtzeit-Ethernet-Ansatz – realisierbar sind. Beckhoff: „EtherCAT stellt einen Quantensprung in der Performance dar, weil er 10 bis 100-mal schneller ist als traditionelle Feldbusse und gleichzeitig Subbussysteme eliminiert.“ Die ETG hat u.a. die Aufgabe, die EtherCAT-Eigenschaften sowie die weitere Implementierung kritisch zu analysieren. Außerdem soll sie Produkt-, Branchen- und applikationsspezifische Anforderungen einbringen sowie Anwendungs- und Geräteprofile erarbeiten. Ein dazu notwendiges Kommunikations-ASIC liegt bereits als VHDL-Modell vor. Im Februar nächsten Jahres soll die erste technische Konferenz stattfinden. Danach werden die EtherCAT-Spezifikation fertiggestellt und das Protokoll offengelegt (www.ethercat.org). *kla*

dos, die direkt in Hardware im EtherCAT-ASIC implementiert sind. Es ergeben sich damit reine Lese- und Schreibbefehle in den 64-Kbyte-Adressraum der Teilnehmer. Die Kommandos sind mit den beschriebenen Adressierungsarten kombiniert und liefern praxisgerechte, effiziente Zugriffsmechanismen für alle Anforderungen an das Kommunikationssystem. Dabei ermöglichen das ADP-Feld (Address Pointer) und das ADO-Feld (Address Offset) eine einfache Differenzierung der Kommandos.

So liest z.B. das Kommando MRD (Multiple Read) den Speicherbereich bei mehreren Slaves. Das ADP-Feld enthält dabei die Stationsadresse des ersten adressierten Slaves, der das MRD-Flag setzt. Ein Slave ist adressiert, wenn die Stationsadresse übereinstimmt oder das MRD-Flag gesetzt ist. Dann muss zusätzlich der mit dem ADO-Feld und den MRD-

Längen-Flags adressierte physikalische Bereich definiert sein. Das ADO-Feld enthält dann die physikalische Adresse innerhalb des Slaves, ab der sich die übertragenen Daten befinden. Der „Working Pointer“ befindet sich in den ersten beiden Bytes des Daten-Feldes; er zeigt auf das nächste freie Datum innerhalb des Daten-Feldes. Nachdem ein Slave Daten in das Daten-Feld eingefügt hat, inkrementiert er den „Working Pointer“ um die entsprechende Länge. Ein Slave fügt allerdings nur dann Daten in das Telegramm ein, wenn er adressiert ist und die einzufügenden Daten noch in das Telegramm passen (Working-Pointer + MRD-Längen-Flags < Länge des Datenfeldes).

► Distributed Clocks

Mit den „Distributed Clocks“ ist es bei EtherCAT möglich, in allen Busteilnehmern die gleiche Uhrzeit einzustellen. Es gibt einen ausgewählten Teilnehmer, der die „Master Clock“ besitzt, auf die sich die „Slave Clocks“ der anderen Teilnehmer und der Steuerung synchronisieren. Dazu sen-

det die Steuerung in gewissen Abständen, ein spezielles Telegramm, in das der Busteilnehmer mit der „Master Clock“ seine aktuelle Uhrzeit einträgt. Dies erfolgt so häufig, dass die „Slave Clocks“ innerhalb vorgegebener Grenzen nicht auseinanderlaufen. Das Telegramm wird dann von den Busteilnehmern mit „Slave Clock“ aus demselben Telegramm gelesen. Dies ist auf Grund der Ringstruktur von EtherCAT möglich, wenn die „Master Clock“ vor den „Slave Clocks“ angeordnet ist.

Das EtherCAT-ASIC bietet dem angeschlossenen Mikrocontroller darüber hinaus noch die Funktion, mittels einer Capture/Compare-Einheit seinen lokalen Timer mit der „Slave Clock“ abzugleichen (*Bild 1*). Da pro Slave auf dem Hin- und Rückweg eine geringe Verzögerung besteht – sowohl im Teilnehmer als auch in der dazwischen liegenden Übertragungsstrecke – ist die Laufzeit zwischen „Master Clock“ und jeweiliger „Slave Clock“ bei der Synchronisierung der „Slave Clocks“ zu berücksichtigen. Zur Messung der Laufzeit sendet der Master einen „Broadcast Read“ auf eine spezielle Adresse, womit jeder Slave veranlasst wird, den Empfangszeitpunkt des Telegramms bezüglich seiner lokalen Clock auf dem Hin- und auf dem Rückweg zu speichern. Diese gespeicherten Zeitpunkte können vom Master eingelesen und entsprechend verrechnet werden.

Querverkehr

Auch wenn EtherCAT ein klares Master-Slave-Kommunikationsmodell nutzt und damit ideal die hierarchische Steuerungstechnik unterstützt, lässt sich mit den Eigenschaften der Feldbus Memory Management Unit (FMMU) sehr einfach der Querverkehr zwischen EtherCAT-Teilnehmern realisieren. Hierzu werden Speicherbereiche aus dem 4 Gbyte großen logischen Adressraum für Querverkehr reserviert und vom Master zyklisch ausgetauscht. Der Master stellt alternierend eine Leseanfrage und im nächsten Zyklus einen Schreibauftrag für den entsprechenden Speicherbereich. Alle entsprechend konfigurierten Teilnehmer blenden ihre Querverkehrsdaten ein bzw. entnehmen sie im nächsten Zyklus. Für

den Master sind diese Daten transparent und er sorgt nur für den zyklischen Austausch.

Im Vergleich zur Partyline-Kommunikation, bei der alle Teilnehmer am selben Medium hängen, muss ein Zyklus abgewartet werden; dies wird aber durch die überragende Nutzdatenrate und die dadurch möglichen kurzen Zykluszeiten mehr als wettgemacht. Die beschriebene Vorgehensweise hat auch den Vorteil, dass die Querverkehrsdaten von mehreren Quellen eingesammelt werden und dann im nächsten Zyklus bei allen angesprochenen Senken gleichzeitig ankommen. Bei ei-

Prinzipiell gibt es zwei unterschiedliche Ansätze, um azyklische Ethernet-Telegramme im zyklischen Feldbus-Betrieb zu übertragen. Die erste Variante besteht darin, im Zyklus eine entsprechende Zeitscheibe zu reservieren, in der die azyklischen Ethernet-Telegramme eingebettet werden. Diese Zeitscheibe muss entsprechend groß gewählt werden, so dass vollständige Ethernet-Telegramme darin Platz finden. Die übliche MTU (Maximum Transmission Unit) liegt bei 1514 byte, was in etwa 125 µs bei 100 Mbit/s entspricht. Daraus ergibt sich für Systeme, die diese Variante nutzen, eine

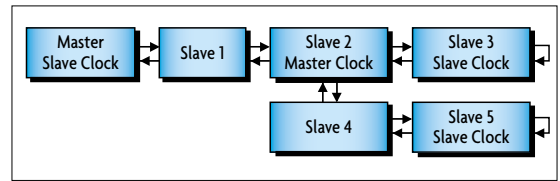


Bild 1. Anordnung von „Master Clock“ und „Slave Clocks“. Die „Distributed Clocks“ bei Slave 1 und 4 sind nicht aktiviert, Slave 2 ist vor Slave 3 und Slave 5 im Ring positioniert und ist daher die „Master Clock“.

Prozessdaten austauscht, ein HTTP-Server integriert werden, der seine eigene Diagnoseschnittstelle in Form von Webseiten mitbringt.

Eine weitere Anwendung der übertragenen Ethernet-Telegramme realisieren die Hub- und Switch-Klemmen. Diese bieten an beliebiger Stelle im EtherCAT-System normale Ethernet-Ports mit entsprechenden RJ45-Buchsen, über die jedes Ethernet-Gerät angeschlossen werden kann. Dies kann z.B. ein Service-Rechner sein, der direkt mit der Steuerung kommuniziert, die Web-Seite eines intelligenten EtherCAT-Teilnehmers abfragt oder einfach über die Steuerung ins Intra- oder Internet „routet“. Die Switch-Klemme integriert zusätzlich zur Hub-Klemme einen vollständigen Switch und bietet dadurch mehrere Ethernet-Anschlüsse in einem Gerät. Im EtherCAT-Master ist ebenfalls ein Switch in Software integriert, der für das Routing der einzelnen Ethernet-Telegramme von und zu den EtherCAT-Teilnehmern und dem IP-Stack des Host-Betriebssystems zuständig ist. Der Funktionsumfang des Switches ent-

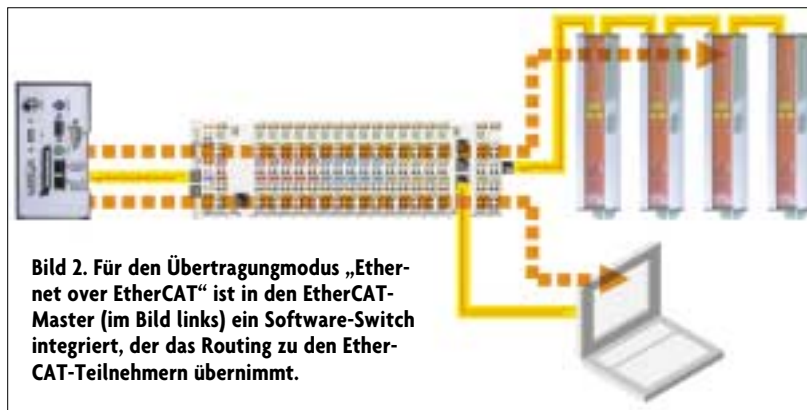


Bild 2. Für den Übertragungsmodus „Ethernet over EtherCAT“ ist in den EtherCAT-Master (im Bild links) ein Software-Switch integriert, der das Routing zu den EtherCAT-Teilnehmern übernimmt.

ner Zykluszeit von z.B. 100 µs können bereits ca. 1000 Bytes von fast beliebig vielen Quellen an ebenso viele Senken gesendet werden. Noch wichtiger ist jedoch, dass die vom Master geplante und von ihm initiierte Querkommunikation absolut deterministisch erfolgt und daher der zyklische Prozessdatenaustausch nicht beeinflusst wird und unnötig Bandbreite reserviert werden muss.

► Ethernet over EtherCAT

Neben den bereits beschriebenen EtherCAT-Adressierungsarten, über die mit den EtherCAT-Teilnehmern kommuniziert wird, wird von einem Ethernet-Feldbus erwartet, dass dieser auch die Standard-IP-Protokolle TCP/IP, UDP/IP und alle darauf basierenden höheren Protokolle (HTTP, FTP, SNMP etc.) beherrscht. Sinnvollerweise sollten dabei einzelne Ethernet-Telegramme transparent übertragen werden, da dadurch keine Einschränkungen bezüglich der zu übertragenden Protokolle entstehen.

minimale Zykluszeit von etwa 200 bis 250 µs. Eine Verringerung der MTU führt häufig zu Problemen: Das IP-Protokoll lässt die Fragmentierung zwar prinzipiell zu, es wird aber davon abgeraten und es wird sie in der nächsten Generation (IPv6) nicht mehr geben. Insbesondere bei UDP/IP-Übertragungen können Datenkonsistenzprobleme auftreten.

EtherCAT nutzt die zweite mögliche Variante, bei der Ethernet-Telegramme getunnelt und im entsprechenden Teilnehmer wieder zusammengestellt werden, bevor sie als vollständige Ethernet-Telegramme weitergeleitet werden. Dieses Verfahren schränkt die erreichbare Zykluszeit nicht ein, da je nach verfügbarer Bandbreite die Fragmente (EtherCAT statt IP-Fragmentierung) optimal eingestellt werden können. EtherCAT definiert dabei einen Standardkanal, der es prinzipiell jedem EtherCAT-Teilnehmer ermöglicht, am normalen Ethernet-Verkehr teilzuhaben. So kann zum Beispiel in einem intelligenten Antriebsregler, der mit 100 µs Zykluszeit seine

■ Die EtherCAT-Entwickler

An der EtherCAT-Entwicklung waren folgende Beckhoff-Mitarbeiter beteiligt:
 Franz-Joseph Kucharski, Thorsten Bunte, Thomas Rettig, Manfred Ullbrock, Michael Schlegel, Holger Büttner, Dirk Janssen sowie der Geschäftsführer Hans Beckhoff.

spricht dem eines üblichen „Layer 2“-Switches, der protokollunabhängig auf die verwendeten Ethernet-Adressen reagiert (Bild 2).

Feldbus einfach anschalten

Dank der Leistungsfähigkeit von EtherCAT können auch komplexe Busteilnehmer, wie z.B. ein Feld-

bus-Mastermodul, realisiert werden. In der Steuerung selbst werden keine zusätzlichen Anschaltungen mehr benötigt. EtherCAT arbeitet quasi als PCI-Ersatz; in der Regel sogar mit einem substantiellen Leistungszuwachs.

Feldbus-Master werden mit einer Prozessabbild-Schnittstelle für den Austausch zyklischer Prozessdaten und einer Mailbox-Schnittstelle für die Übergabe azyklischer Parameter- oder Diagnosedaten an eine Steuerung angeschaltet. Das Prozessabbild kann über EtherCAT in der Regel mit einem ein-

CAT-Modulen reduziert sich also im Vergleich zu herkömmlichen 16-bit-PCI-Anschaltungen auf bis zu 70 %.

Auch eine äquidistante Schnittstelle, wie sie z.B. bei Antriebsregelungen über Sercos, Profibus DPV2 oder CANopen mit TwinCAT üblich ist, lässt sich problemlos realisieren. Über die „Distributed Clocks“ kann der Zyklus des Feldbus-Masters nahezu jitterfrei auf den EtherCAT-Zyklus synchronisiert werden. Auch ohne „Distributed Clocks“ sorgt eine PLL in dem Feldbus-

SSI-Schnittstelle am Slave

Die SSI-Schnittstelle (Synchronous Serial Interface) an den Slaves ist für alle Geräte geeignet, die nur einige wenige Prozessdatenbytes übertragen; hierzu gehören analoge Module, Geber, Encoder oder auch einfache Antriebe. Das ASIC verfügt über einen 4 Kbyte großen Speicher, der über EtherCAT adressiert werden kann. Die Datenübertragung vom ASIC zum Mikrocontroller erfolgt über eine 10 Mbit/s schnelle SSI-Schnittstelle, die vom Buszugriff entkoppelt ist. Die SSI-Übertragung stellt eine kostengünstige Variante für Geräte mit wenigen Prozessdatenbytes dar. Die Menge der azyklischen Parameter- und Diagnosedaten ist nahezu unbeschränkt, sofern deren Übertragung nicht zeitkritisch ist.

Parallel mit 32 bit

Die 32-bit-Parallel-In/Out-Schnittstelle benötigt mehr Pins als die beiden bereits beschriebenen Schnittstellen. Sie eignet sich für den Anschluss von 32 digitalen Eingang- bzw. Ausgangssignalen, aber auch für einfache Aktoren oder Sensoren, die mit 32 parallelen Datenleitungen auskommen. Im Gegensatz zur SSI-Schnittstelle ist der Zugriff auf die Prozessdaten wesentlich schneller, und es ist kein Mikrocontroller erforderlich. Allerdings ist das Datenaufkommen auf 32 bit beschränkt.

Parallele 16-bit-Mikrocontroller-Schnittstelle

Die parallele 16-bit-Mikrocontroller-Schnittstelle entspricht herkömmlichen Schnittstellen bei Feldbus-ASICs mit DPRAM-Schnittstelle (Dual Port RAM). Diese Anschaltung ist etwas aufwendiger, dafür aber auch leistungsfähiger als die anderen ASIC-Schnittstellen. Sie eignet sich daher insbesondere für komplexe Teilnehmer mit größerem Datenaufkommen.

Ethernet in der Feldebene

In der Automatisierungstechnik zeichnet sich der Trend ab, Ethernet auch in der Feldebene einzusetzen. Unterschiedliche Ansätze versprechen hohe Bandbreiten, geringe Kosten, vereinfachte vertikale Integration, Einsatz von Standardkomponenten aus der

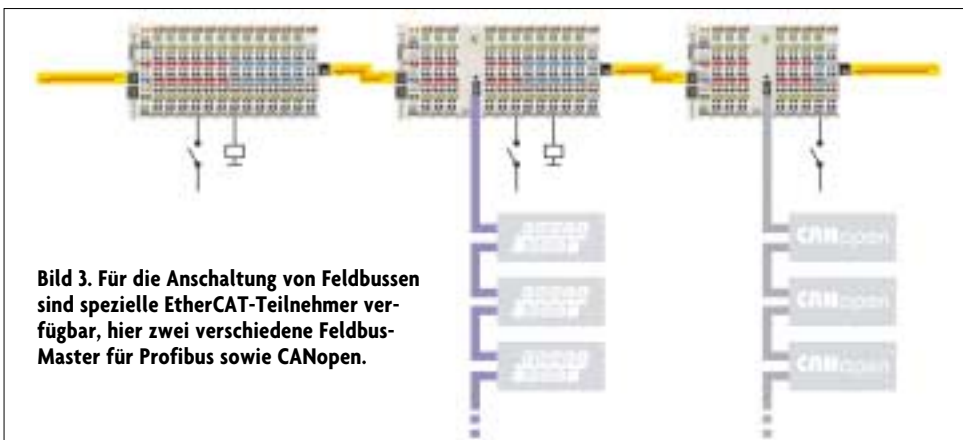


Bild 3. Für die Anschaltung von Feldbussen sind spezielle EtherCAT-Teilnehmer verfügbar, hier zwei verschiedene Feldbus-Master für Profibus sowie CANopen.

zigen Telegramm übertragen werden; wobei der Master das „Output“-Prozessabbild sendet und das „Input“-Prozessabbild wieder empfängt. Die Grenze liegt bei jeweils 1488 byte „Input“- und „Output“-Prozessdaten, die neben dem Ethernet-Header und dem EtherCAT-Header noch im Ethernet-Frame verfügbar sind. Die Übertragung dauert maximal 125 µs; der zusätzliche Aufwand in der Steuerung ist verhältnismäßig gering, da der Ethernet-Chip selbstständig per DMA (Direct Memory Access) auf den Speicher zugreift, ohne den Steuerungsprozessor zu belasten. Herkömmliche Feldbus-Anschaltungen haben in der Regel keine DMA-Schnittstelle zum Speicher: Die Übertragung von je 1488 byte „Input“- und „Output“-Prozessabbild dauert bei einer Feldbus-Anschaltung mit einer 16-bit-Schnittstelle über PCI (Peripheral Computer Interface) ca. 500 µs. Die Übertragungsdauer bei dezentralen Ether-

Mastermodul dafür, dass der Jitter des EtherCAT-Telegramms ausgeregelt wird.

Auf die Mailbox-Schnittstelle eines Feldbus-Mastermoduls kann einfach über physikalische Adressierung zugegriffen werden. Diagnosedaten können über die Interrupt-Bits gemeldet und per „Multiple Read“-Adressierung übertragen werden (Bild 3).

ASIC-Schnittstellen

Um sowohl ganz einfache Teilnehmer, wie z.B. ein digitales Eingangsmodul mit 2 bit Breite, als auch komplexe Teilnehmer, wie ein Feldbus-Mastermodul, optimal hinsichtlich der Kosten und der Leistungsfähigkeit zu implementieren, sind verschiedene ASIC-Schnittstellen erforderlich.

4-bit-I/O-Schnittstelle

Die 4-bit-I/O-Schnittstelle ist für digitale Ein- und Ausgänge bis zu 4 bit Breite gedacht und stellt eine besonders kostengünstige Anschaltung dar. Das ASIC hat vier konfigurierbare I/O-Anschlüsse, ein Mikrocontroller wird nicht benötigt.

Bürowelt und geringe Konfigurations- und Diagnose-Aufwendungen. Das Ganze kombiniert mit der nötigen Echtzeit-Fähigkeit.

Bei genauerem Hinsehen weichen viele der Argumente auf oder verkehren sich ins Gegenteil: Die vergleichsweise hohe Bandbreite von 100-Mbit/s-Ethernet wird bei typischen I/O-Knoten mit wenigen Bytes Prozessdaten zunichte gemacht. Ein Teilnehmer, der z.B. vier Bytes pro Richtung hat, kommt auf eine Nutzdatenrate von 3 bis 4 %. Auch die Kostenseite spricht eher gegen den Einsatz im Feld. Neben den reinen Anschaltungskosten kommt noch ein recht hoher Bedarf an Rechenleistung für die Verarbeitung der Telegramme hinzu. Der Einsatz von Standardkomponenten hört meisten dann auf, wenn ein gewisses Maß an Echtzeit-Fähigkeit gefordert wird. Außerdem ist die typische Sterntopologie mit Switches eher ungünstig für den Einsatz im Feld. Selbst die Konfiguration wird nicht leichter: Die Vergabe der notwendigen IP-Adressen wird nicht selten in Konflikten mit der IT-Abteilung enden.

EtherCAT geht einen anderen Weg und verbindet die Vorteile der Feldbusse mit denen der Ethernet-Welt. Die zur Verfügung stehende Bandbreite wird nahezu vollständig ausgenutzt, die Kosten reduzieren sich auf eine

einfache ASIC-Anschaltung im EtherCAT-Teilnehmer. EtherCAT benötigt keine IP-Adressen, die Konfiguration läuft – vom Master gesteuert – nach einfachen Algorithmen automatisch ab. Die vertikale Integration ist aber trotzdem gegeben. Teilnehmer, die eine IP-Adresse haben möchten, bekommen diese auch und sind dann vollständig transparent im Netzwerk integriert.

Mit EtherCAT lassen sich höchst leistungsfähige Maschinensteuerungen realisieren, bei denen viele verteilte Signale mit Zykluszeiten deutlich unter 100 µs ausgetauscht werden können. Das System eignet sich aber ebenso gut für kostengünstige Steuerungsapplikationen, die mit um drei Größenordnungen längeren Zykluszeiten auskommen, wie z.B. mit 100 ms in der Gebäudeautomatisierung. Jeder handelsübliche PC oder jeder Steuerungscontroller mit integriertem Ethernet-Port kann dabei als Master eingesetzt werden.

EtherCAT bietet daher eine einheitliche, leistungsfähige Kommunikationsbasis für die gesamte Automatisie-

rungstechnik und ist bestens auf deren Anforderungen zugeschnitten. Von der „Klein“-SPS für weniger als 100 Euro bis zur Hochleistungs-CNC kann die gleiche Technik eingesetzt werden. *jw*

Weitere Infos:

- [1] www.ethercat.de
- [2] www.beckhoff.de



Dr.-Ing. Dirk Janssen

studierte Maschinenbau an der Technischen Universität Braunschweig. Nach seiner Promotion im Bereich des Software-Designs für Automatisierungssysteme ist er seit 1996 bei Beckhoff in Verl tätig. Als Leiter der Software-Entwicklung beschäftigt er sich u.a. intensiv mit dem Einsatz und der Konfiguration von Feldbussen.

► E-Mail: info@beckhoff.de



Dipl.-Ing. Holger Büttner

studierte Elektrotechnik an den Technischen Universitäten Hannover und Karlsruhe von 1986 bis 1991. Nach dem Studium entwickelte er bei der Firma TMG in Karlsruhe Profibus-Protokollsoftware. Seit 1996 ist er bei Beckhoff in der Niederlassung Berlin für die Feldbus-Entwicklung zuständig.