2.4 Einführung in Controller Area Network (CAN)

Dieser Darstellung von Grundlagen des CAN liegt folgende Literatur zugrunde:

Becker, Norbert Automatisierungstechnik Kamprath-Reihe im Vogelverlag 2006
Zeltwanger, Holger (Hrsg.) CANopen VDE Verlag GmbH Berlin und Offenbach 2001
http://de.wikipedia.org/wiki/Controller_Area_Network
http://de.wikipedia.org/wiki/CANopen

Controller Area Network ist ein asynchrones, serielles **Feldbussystem.** Es arbeitet nach dem **Multi-Master-Prinzip**. Alle Busteilnehmer sind gleichberechtigt. Der Bus wird entweder mit Kupferleitungen oder mit Glasfaser ausgeführt, aber auch drahtlose Lösungen und Powerline sind möglich.

CAN wurde bereits ab 1983 vom Unternehmen Bosch für die Vernetzung von Steuergeräten in Automobilen entwickelt. Primäres Ziel war die schrittweise Einsparung bei Kabelbäumen in Kraftfahrzeugen, die bis zu 2 km Länge pro Fahrzeug annehmen können. Heute erfolgt mit CAN eine zuverlässige und sichere Kommunikation in den Bereichen Automobil-, Verkehrs-, Transport- und Medizintechnik und weiter mit bestimmten Ausprägungen auch in der Automatisierungs- und Robotertechnik.

CAN wurde 1993 international als ISO 11898-1 standardisiert. Die Kommunikation erfüllt hohe Anforderungen an die "Echtzeitfähigkeit". Routinen für Prioritätssteuerung, automatische Fehlermeldungen und für das erneute Übertragen fehlgeschlagener Datenübertragung sind integriert.

Die Organisation **CAN-in-Automation e.V. (CiA**) unterstützt die weitere Verbreitung von CAN sowie der CAN-basierten höheren Protokolle. Anfang 1999 hatte die CiA bereits über 400 Mitglieder weltweit (Internet: www.can-cia.de).

CAL steht für CAN Application Layer und enthält Festlegungen der Organisation CiA zur Gestaltung der Anwendungsschicht 7 des OSI-Referenzmodells (**Bild 2.4-1**). Die Vereinbarungen sind im Standard CiA DS 201-207 von 1993 festgelegt.

CANopen ist ein auf CAN basierendes Kommunikationsprotokoll, welches hauptsächlich in der Automatisierungstechnik und zur Vernetzung innerhalb komplexer Geräte verwendet wird. CANopen kommt vor allem in Europa zum Einsatz. Seit 1995 wird es von der Organisation CiA gepflegt und ist als Europäische Norm EN 50325-4 standardisiert.

In den USA erfolgte eine vergleichbare Entwicklung unter dem Namen **DeviceNet**.

<table>
<tr><th colspan="2">OSI-Schicht</th><th colspan="3">CAN</th></tr>
<tr><td>7</td><td>Anwendung</td><td>CAL</td><td>CANopen</td><td>Device Net</td></tr>
<tr><td>6</td><td>Darstellung</td><td colspan="3" rowspan="4"></td></tr>
<tr><td>5</td><td>Sitzung</td></tr>
<tr><td>4</td><td>Transport</td></tr>
<tr><td>3</td><td>Vermittlung</td></tr>
<tr><td>2</td><td>Sicherung</td><td colspan="3">Frameaufbau, Fehlererkennungsmechanismen</td></tr>
<tr><td>1</td><td>Bitübertragung</td><td colspan="3">modifizierte RS 485</td></tr>
</table>

Bild 2.4-1: CAN und OSI-Modell

2.4.1 Busmedium, Buszugriff und Übertragungsverfahren

Das am häufigsten verwendete Busmedium ist eine modifizierte RS 485 – Zweidraht-Kupferleitung, bestehend aus einem verdrillten Aderpaar und gemeinsamen Rückleiter. Die Adern werden bezeichnet als CAN_High, CAN_Low und Ground(**Bild 2.4-2**). Der Abschlusswiderstand beträgt 120 Ohm (praktisch oft auch 124 Ohm). Der Bus wird als Linie aufgebaut. In Kraftfahrzeugen und einigen

anderen Einsatzfällen kann man CAN auch mit einer Eindrahtlösung gegen Masse ausführen, insbesondere dann, wenn die Übertragungsrate gering bleibt. Dies ist auch eine Möglichkeit für eine Notlösung bei Ausfall einer Ader.

Auf der zweiten CAN-Ader wird das invertierte logische Signal der ersten Ader übertragen. Durch die Arbeit mit Differenzsignalen werden Gleichtaktstörungen unterdrückt.

Die maximale Ausdehnung des CAN-Bus ist begrenzt und direkt der gewünschten Übertragungsrate zugeordnet (Bild 2.4-2). Dies hat seine Ursache im Bitarbitrierungs-Verfahren. Alle CAN-Teilnehmer müssen die Nachrichten (Telegramme) salopp gesagt "gleichzeitig" verarbeiten können.

"CAN-Teilnehmer" sind zumeist Microcontroller mit speziellen CAN-Asics, die über CAN-Transceiver an den Bus angeschaltet werden. "CAN-Controller" selbst erlauben zumeist nur die CAN-spezifischen Operationen und weniger die Abarbeitung von Anwendungsprogrammen (**Bild 2.4-3**).
Man unterscheidet bei den Controllern hinsichtlich der Ausstattung heute je nach Leistungsfähigkeit die Level Basic-CAN, Full-CAN und Extendet-CAN.

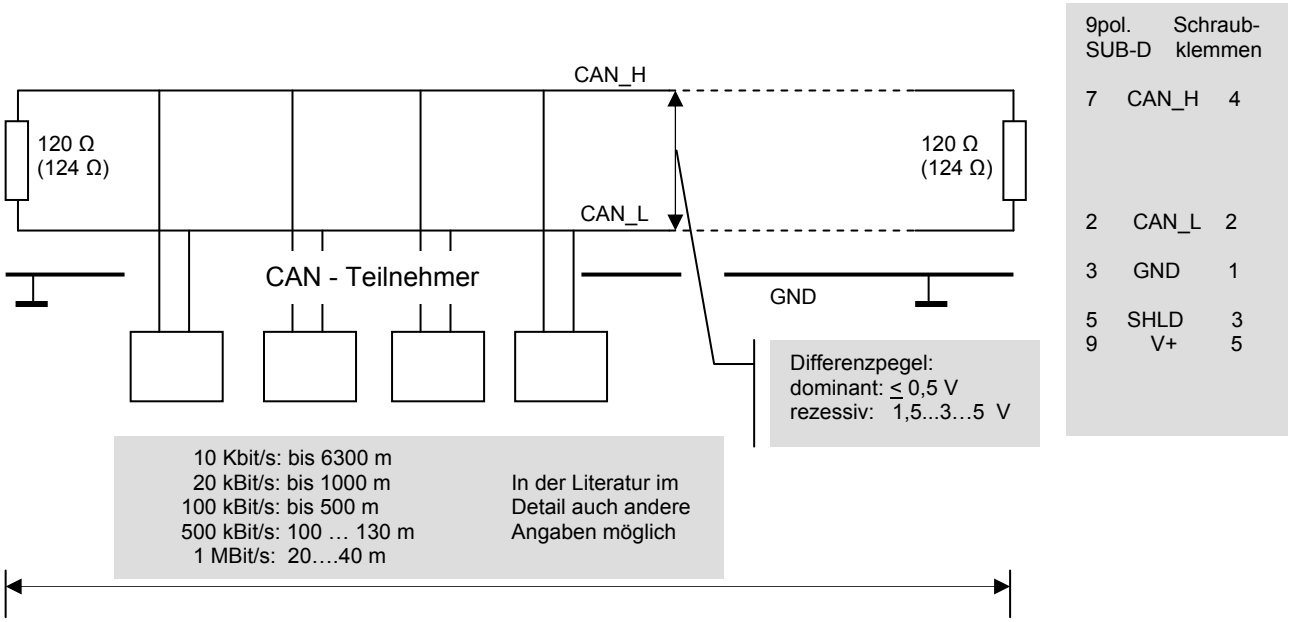

Bild 2.4-2: Das CAN-Bus-Medium

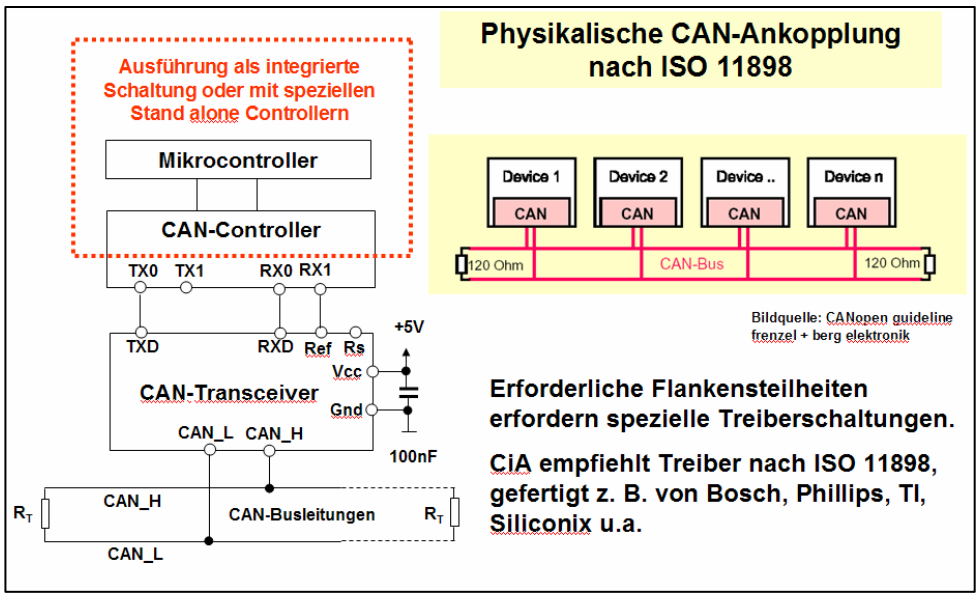

Bild 2.4-3: Anschaltung von Geräten mit CAN-Transceiver und CAN-

Es ist wichtig zu wissen, dass CAN beim Zugriff auf das Übertragungsmedium mit einem anderen Verfahren arbeitet als Ethernet. Für Ethernet ist das Verfahren Carrier Sense Multiple Access / Collision Detection (CSMA/CD) nach IEEE 802.3 standardisiert. Es bedeutet Mehrfachzugriff mit Trägerprüfung und Kollisionserkennung (siehe Abschnitt 2.1).

CAN arbeitet dagegen nach dem Verfahren Carrier Sense Multiple Access / Collision Resolution (CSMA/CR). Es bedeutet Mehrfachzugriff mit Trägerprüfung und Kollisionsvermeidung. Dabei erfolgt der Zugriff ebenfalls zufällig, Kollisionen werden aber durch Bitarbitrierung vermieden.

Bitarbitrierung ist eine Methode zur Ermittlung der Zugriffsrechte. Sie bewirkt, dass am Ende der Arbitrierungsphase lediglich ein Busteilnehmer den Bus belegt. Kollisionen führen nicht zum Abbruch der Kommunikation. Stattdessen erhält die Nachricht mit der höchsten Priorität das Recht auf Busnutzung. Die Prioritäten werden durch Nachrichtenidentifier festgelegt. Werden nicht ständig hoch priorisierte Nachrichten gesendet, so können trotz des zufälligen Buszugriffs Aussagen über die maximale Latenzzeit solcher Nachrichten getroffen werden. Damit ist CAN zwar eingeschränkt, in der Praxis aber häufig ausreichende echtzeitfähig. Mit steigender Busbelastung wird das System allerdings immer weniger deterministisch.

Nicht nur der Buszugriff, sondern auch die Kommunikation zwischen CAN-Teilnehmern unterscheidet sich grundlegend von den wichtigsten anderen Bussystemen: CAN arbeitet bei der Nachrichtenübermittlung nicht mit Ziel- und Quelladresse, sondern nach dem Prinzip des **nachrichtenorientierten Protokolls:**

Ein Teilnehmer sendet seine Nachricht nicht an eine spezielle Busadresse, sondern nach dem Prinzip des Rundfunks (Broadcasting) an alle. Anstelle einer Zieladresse besitzen die Nachrichten eine **Nachrichtennummer (Identifier)** und sind als Objekte charakterisiert. Diese werden in Sendeobjekte und Empfangsobjekte unterschieden und sind zu projektieren. Man bezeichnet diese Arbeitsweise auch als objektorientierte Kommunikation und spricht von einem **Producer-Consumer-Modell** bzw. auch **Pubblisher-Subscriber-Modell**. Ein Producer verfügt über Sendeobjekte und versendet diese an Consumer, die ihrerseits über Sendeobjekte verfügen. Zumeist erfolgt das Versenden bedarfsgesteuert, also nur, wenn sich Werte ändern. Das erlaubt prinzipiell geringere Buslasten als sie im klassischen Master-Slave-Verfahren mit seinen zyklischen Abfragen entstehen.

Unterschieden werden bei CAN

- Data Frame (Datentelegramme) für die Übertragung von bis zu 8 Byte Daten im Broadcasting-Verfahren
- Remote Frame (Datenanforderungstelegramme) für das Anfordern von Nachrichten
- Error Frame (Fehlertelegramme) für die Signalisierung eines von einem Teilnehmer erkannten Fehlers
- Overload Frame für ein zeitweiliges Anhalten der Kommunikation

Für das Verständnis des Datenverkehrs auf dem CAN-Bus ist der grundsätzliche Aufbau der Telegramm-Rahmen (Frame) zu betrachten (Bild **2.4-4**). Zu erkennen ist unter Nr. 2 ein Statusfeld (Arbitrations Field). Hier werden die Identifier eingetragen. Jeder Nachricht wird eindeutig ein Identifier zugeordnet. Nach der Norm EN 50234-4 hat dieser bei Version 2.0A eine Größe von 11 Bit. Daneben gibt es optional (und weniger bedeutend) eine Version 2.0B. mit 29 Bit Identifier (**Bild 2.4-5**).

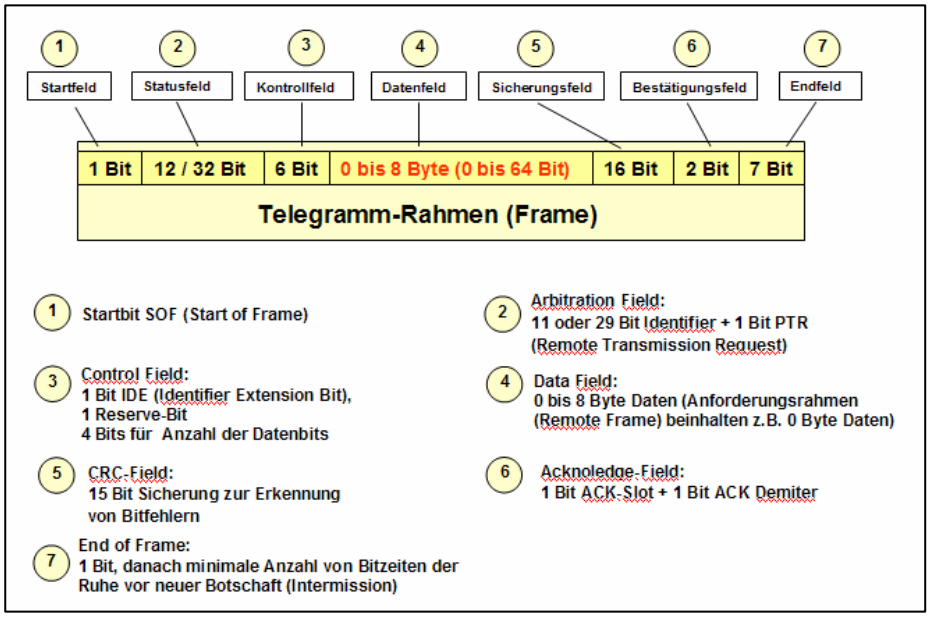

Bild 2.4-4: Elemente des CAN-Telegramms

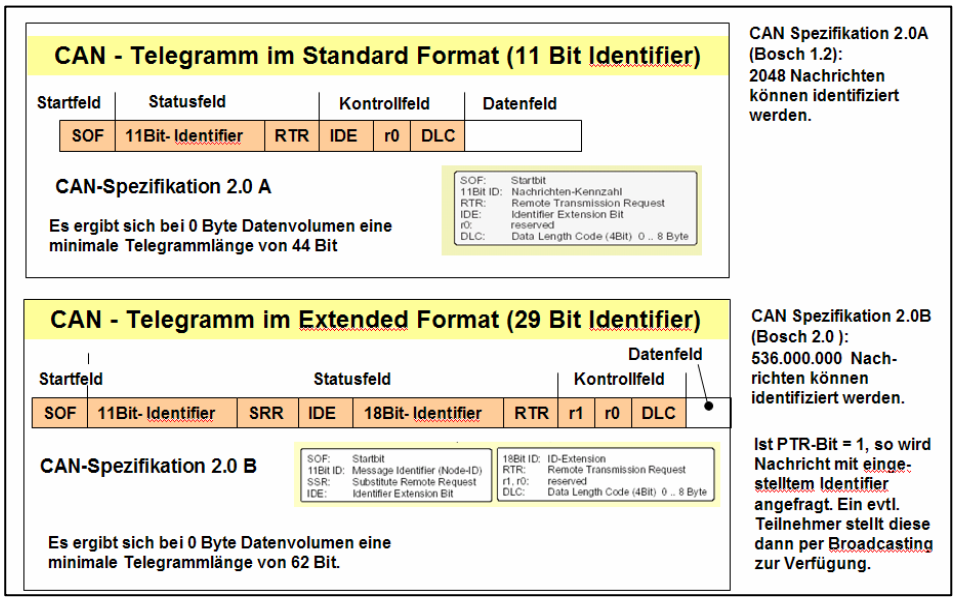

Bild 2.4-5: Identifier im Standard- und im erweiterten Format

Ein CAN-Gerät sendet somit Kommunikationsobjekte. Die Teilnehmer am CAN-Bus besitzen Filter zur Selektion der Nachrichten und filtern die für sie bestimmten Nachrichten heraus. Das erfolgt so, dass ein projektiertes Empfangsobjekt eine zutreffende Nachricht an einem Identifier erkennt, welcher im Sendeobjekt festgelegt wurde. Ein Beispiel für die Funktion eines solchen Filters zeigt Bild 2.4-8.
Ein 11 Bit Identifier kann maximal 2048 Empfangsobjekte adressieren. Nochmals wird betont, dass nicht Teilnehmer adressiert, sondern Objekte gekennzeichnet werden!

Ein weiterer Begriff im CAN sind die Descriptor–Bytes. (**Bild 2.4-6**). Sie beinhalten neben dem Identifier auch ein RTR-Bit sowie die Verschlüsselung der Anzahl der Datenbyte. RTR steht für Remote Transmission Request und dient der Unterscheidung zwischen Daten senden und Daten anfordern.

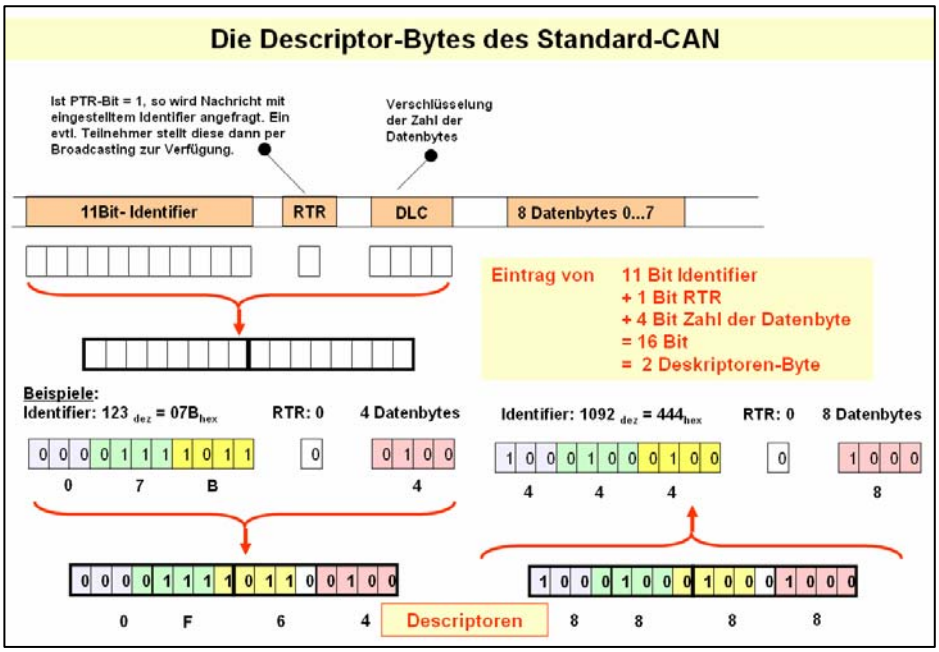

Bild 2.4-6: Beispiele für Descriptoren im CAN-Standardformat

Details der bitweisen Arbitrierung

In der **Arbitrierungsphase** wird entschieden, welcher Busteilnehmer das Senderecht erhält.
Auf dem Bus unterscheidet man dominante Pegel (bis 0,5 V) und rezessive Pegel (1,5 bis 3,5 V). Das Aufschalten von Teilnehmern an den Bus kann am besten mit dem Prinzip des Wireless AND erklärt werden (**Bild 2.4-7**): **Sobald ein Teilnehmer auf den Bus zugreift, liegt dieser auf dominantem Pegel**. **Low-Signale sind dominant**. Ist der Bus dagegen frei, liegt er auf rezessivem Pegel.

Jeder Teilnehmer beobachtet Bit für Bit den Buspegel. Während er ein Bit des Identifier sendet, beobachtet er gleichzeitig den Pegel auf dem Bus. Das vom Teilnehmer zurückgelesene Bussignal wird mit dem gesendeten Signal verglichen. Wird dabei sein gesendetes rezessives Bit dominant überschrieben, dann bedeutet dies, dass bereits ein anderer Teilnehmer eine Nachricht mit höherer Priorität sendet.

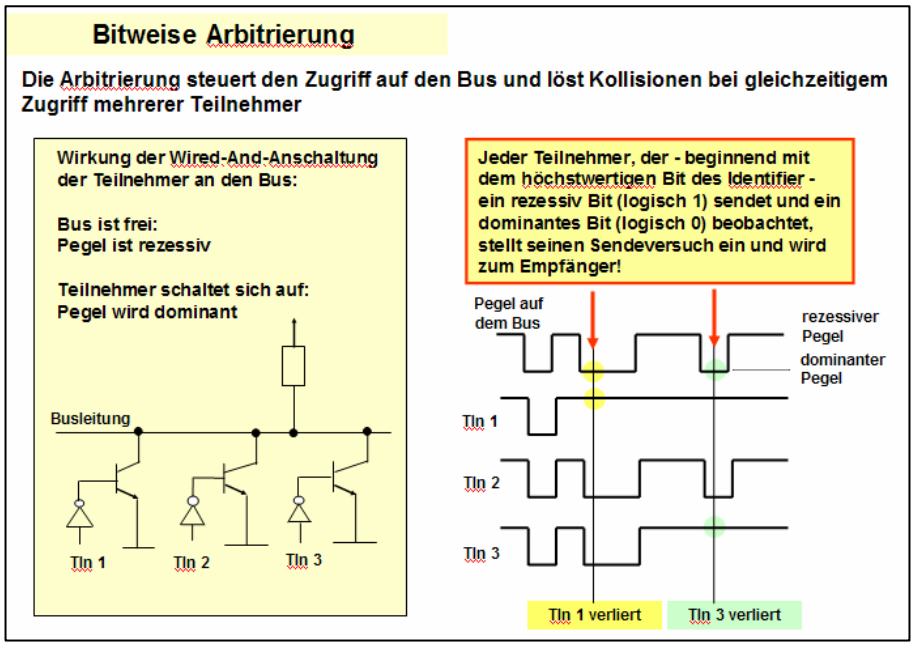

Bild 2.4-7: Die Anschaltung an den CAN nach dem Wired AND Prinzip

Prioritäten werden mit Binärwerten vergeben. Für die Feststellung der Priorität wird nacheinander jedes Bit des Identifiers ausgewertet. Ein Identifier mit niedrigster Binärzahl hat höchste Priorität. Der dominante Zustand „0“ überschreibt den rezessiven Zustand „1“. Jede Station, die rezessiv sendet, aber dominant beobachtet, verliert die Sendemöglichkeit. Die „Verlierer“ werden zu Empfängern der Nachricht mit höherer Priorität. So ergeben sich folgende beispielhaften Situationen:

- Ein Teilnehmer erkennt dominantes Bit auf dem Bus, während er rezessiv sendet bzw. passiv verharrt. Er erwartet das Bit Start of Frame (SOF) und empfängt die Botschaft.
- Ein Teilnehmer will senden, empfängt aber eine Botschaft. Er wartet auf 3 Bit Intermission nach Empfang der Botschaft. Der Bus ist frei! Er sendet seine Priorität und kann evtl. Botschaft senden.
- Bei gleichzeitigem Sendebeginn mehrerer Teilnehmer erfolgt die bitweise Arbitrierung durch Auswertung der Prioritäten im Identifier. Dies ist beispielhaft im **Bild 2.4-8** dargestellt.

Ist der Bus wieder frei, setzen die Stationen das Senden in der Reihenfolge der Prioritäten fort.

Im Bild 2.4-9 sind noch Beispiele für das "Herausfiltern" von relevanten Nachrichten aufgeführt.

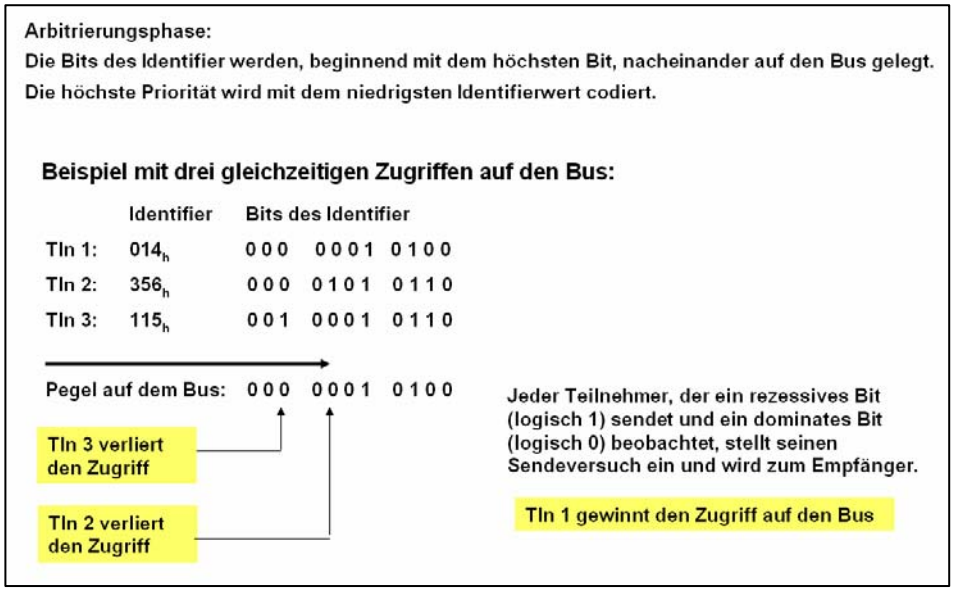

Bild 2.4-8: Beispiel der bitweisen Arbitrierung

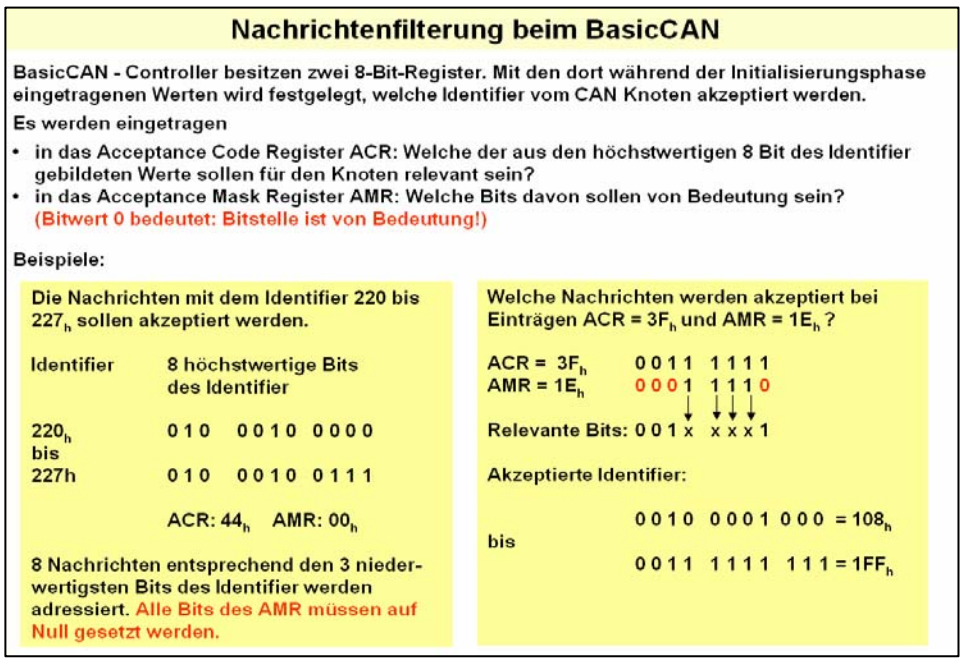

Bild 2.4-9: Beispiel für die Nachrichtenfilterung

2.4.2 Welche Festlegungen trifft CAL?

CAL (CAN Application Layer) ist das allgemeine Kommunikationsprotokoll für Geräte am CAN-Bus. Ohne Einschränkungen stellt es die Objekte, Protokolle und Dienste für die ereignis- oder abfragegesteuerte Übertragung von CAN-Nachrichten bereit, weiter bei Bedarf auch für die Übertragung grösserer Datenmengen. CAL bietet leistungsfähige Verfahren für die automatische Zuordnung von Nachrichten-Identifiern und für die Initialisierung und Überwachung von Netzknoten. Es legt zulässige Datentypen und Regeln für die Reihenfolge der Bitübertragung fest, spezifiziert Dienste und Protokolle für das Netzwerksmanagement.

CAL ist flexibel und allgemein gehalten. Bei der Arbeit nach CAL hat der Anwender die Möglichkeit, die Kommunikation optimal auf seine Bedürfnisse auszulegen. Es ist vorteilhaft anzuwenden, wenn er auf die Konfigurierbarkeit der Systemressourcen verzichten möchte, zum Beispiel in abgeschlossenen Systemen der Medizintechnik oder speziellen Messtechnik, und auch bei komplexen Kommunikationsbeziehungen zwischen Teilnehmern.
Dagegen bieten andere CAN-basierte Protokolle wie CANopen oder DeviceNet "standardisierte" (de facto fertige) Systemlösungen.

CAL beschreibt folgende Grundelemente:

- **CAN Message Specification (CMS)**
  Sie definiert die Kommunikationsdienste zwischen CAL-Teilnehmern und legt dafür notwendige Datentypen wie Bit, Integer, Array, Structure etc. fest.
  Es werden einfache (Basic CMS Objects) und erweiterte (Enhanced CMS Objects) Kommunikationsobjekte unterschieden. CMS basiert auf dem Client-Server-Modell.

  Einfache CMS-Objekte benötigen **keinen zusätzlichen Protokoll-Overhead**. Als "Overhead" werden die zusätzlichen Daten bezeichnet, die zu den eigentlichen Daten beim Durchlaufen von OSI-Schichten addiert werden.
  Zu den einfachen CSM-Objekten gehören z.B. nachfolgende Definitionen:
  - Read-Only-Variablen:

  Diese nutzen einen Server und beliebig viele Clients. Die gewünschte CAN-Nachricht wird von einem Client über ein Datenanforderungstelegramm bei einem Server angefordert.
  - Write-Only-Variablen:

  Diese nutzen mindestens einen Server und höchstens einen Client. Über Write-Only-Variablen senden Clients Daten zu jedem Server dieser Variablen.
  - Uncontrolled Event:

  Diese nutzen einen Server und beliebig viele Clients. Über diese CMS stellt der Server, ohne dass er dazu aufgefordert wird, Daten für die daran interessierten Clients zur Verfügung.

  Bei **erweiterten CMS-Objekten** wird das erste Datenbyte für ein zusätzliches CMS-Protokoll verwendet. Damit werden z.B. Read-Write-Variablen, Multiplexvariablen, Datenbereiche und steuerbare Ereignisse (controlled event) spezifiziert.

- **Distributor**
  Er ermöglicht die dynamische Vergabe der Identifier. Eine manuelle Konfiguration jedes Knotens ist damit nicht notwendig und das System kann einfach erweitert bzw. geändert werden, ohne dass der Anwender die Identifier neu festlegen muss.

- **Network-Management**
  Mit diesem Werkzeug werden globale Dienste der Netzwerküberwachung spezifiziert. Dazu zählen das An- und Abmelden einzelner Teilnehmer, die Überwachung der Teilnehmer während des Betriebes und die Behandlung von Ausnahmezuständen.

- **Layer Management**
  Damit werden Dienste für die netzwerkweite Konfiguration bestimmter Parameter wie z.B. die Vergabe der Identifikationsnummern der einzelnen Knoten verwaltet.

2.4.3 Einführung in CANopen (EN 50325-4)

Das Anwendungsprotokoll CANopen schränkt die Vielfalt von CAL mit dem Ziel ein, für klassische und ausgewählte Anwendungen - insbesondere in der Automatisierungstechnik - den Projektierungsaufwand entschieden zu senken. Salopp gesagt: "Per Automatismus" werden viele notwendige Parameter als Default-Einstellungen vergeben, so dass diese für den Anwender "unsichtbar" bleiben.

CANopen ist damit die auf die Automatisierungstechnik zugeschnittene Anwenderschnittstelle des CAN. Die erforderlichen Festlegungen wurden von der CiA ab 1995 getroffen
(Beispiele: CANopen Device Profile for Generic I/O Modules DS-401 Vers. 2.0 1999
CANopen Device Profile for Drives and Motion Control DSP-402 Vers. 2.0 1998).

Entsprechend der nachrichtenorientierten Kommunikation des CAN werden im CAN-Knoten Objekte eingetragen und im **Objektverzeichnis** verwaltet (**Bild 2.4-10**). Im Objektverzeichnis sind die Daten eines CANopen Gerätes strukturiert abgelegt. Es enthält alle Geräteparameter und bildet die Verknüpfung zwischen Kommunikationsschnittstelle und Anwenderprogramm.

Für den eigentlichen Datentransport sind die **Prozessdatenobjekte (PDO)** verantwortlich. Diese werden wiederum in Sende- und Empfangs-Prozessdatenobjekte (TPDO und RPDO) unterschieden.

Jede PDO besteht aus dem für CAN typischen Identifier sowie Daten. Der Identifier wird hier **Communication Objekt Identifier (COD-ID)** genannt.

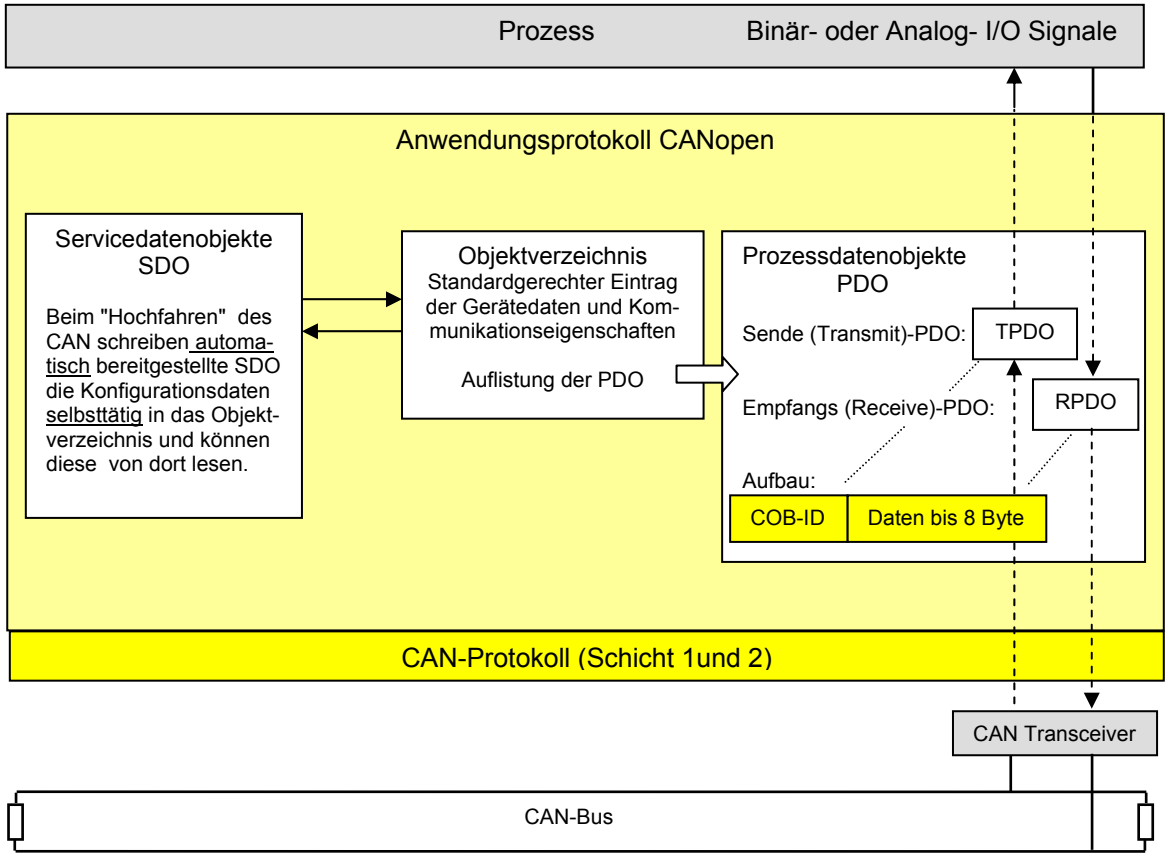

Bild 2.4-10: Schematischer Aufbau eines CAN-Knotens, dargestellt am Beispiel eines Remote I/O

Im gesamten System darf es nur jeweils eine einzige COB-ID TPDO Nr.x als Sendeobjekt geben. Die jeweiligen Empfangsobjekte tragen gleiche COB-ID RPDO - Nummern. Von diesen können im System mehrere verteilt sein.

Der Projektant führt ein vereinfachtes PDO-Mapping durch. Dabei ordnet er in allen CAN-Geräten die zu sendenden bzw. zu empfangenen Daten bestimmten Sende- und Empfangsobjekte zu (**Bild 2.4-11**).

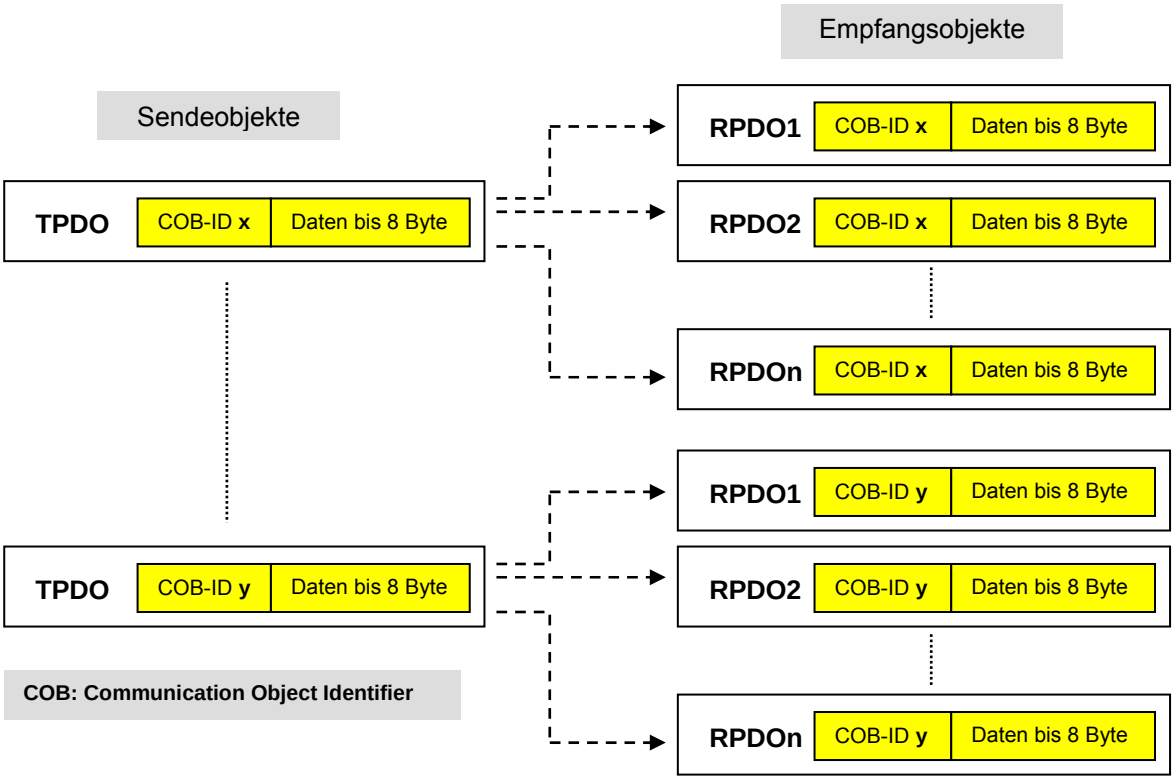

Bild 2.4-11: Zuordnung der Objekte TPDO und RPDO durch PDO Mapping

EDS- und DCF-Dateien:

Parameter und Kommunikationseigenschaften von Profibus-Geräten werden in Gerätestammdateien (GSD) nach Regeln der Profibus-Nutzer-Organisation (PNO) festgeschrieben. Diese Dateien werden in das Engineering-System (zumeist Step7) eingefügt. Dadurch werden die Hardware-Kataloge im Tool Hardware-Konfiguration ergänzt.

Im System CANopen übernehmen Electronic Data Sheets (EDS) die gleiche Aufgabe. Sie beinhalten die Herstellerdaten von CAN-Geräten.
Ein Device Configuration File (DCF) trägt über ein EDS-File hinaus zusätzliche konkrete Projektdaten wie z.B. Baudraten und Adressen.

Die Übertragung von SDOs

SDO (Service Data Objects) kann man mit "Dienstdatenobjekten" übersetzen. Mit ihnen werden überwiegend Parameter zur Gerätekonfiguration ausgetauscht. Es können Daten beliebiger Länge übertragen werden, und die Kommunikation erfolgt ebenfalls über Identifier. SDO haben gegenüber den PDO eine geringere Priorität.

Die Verbindung zwischen einem Client und einem Server im Point-to-Point Verfahren wird auch als SDO-Kanal bezeichnet. Wichtigste Anwendung von SDO-Kanälen ist die Konfiguration über ein Konfigurationstool. Ein CANopen-Gerät muss mindestens ein SDO-Server-Objekt unterstützen und damit für andere Geräte (z.B. ein Konfigurationstool) erreichbar sein.
Mit SDO werden die Einträge im Objektverzeichnis geschrieben oder gelesen.

Automatisches PDO-Mapping mit dem Default Predefined Master-Slave-Connection Set (auch als Master-Slave-Default bezeichnet)

Um dem Projektanten bei Standardanwendungen das Mappen zu erleichtern, ermöglicht CANopen die automatische Zuordnung der PDO durch Default-Einstellungen. Auf diese Weise treten viele Details des Mappens nach außen nicht in Erscheinung bzw. verändern ihr Erscheinungsbild gegenüber dem "Ur-CAN". Insbesondere werden gemäß CiA-301 CANopen Application Layer and Communication Profile Vers. 4.01 vom Jahre 2000 in Geräteprofilen von vornherein vier RPDO- und vier TPDO-Objekte für eine Master-Slave-Kommunikation (!) angelegt.
(Man arbeitet hier abweichend vom Grundgedanken des CAN mit einer (virtuellen) Master-Slave-Struktur, um die Projektierung ähnlich einfach zu machen wie von Profibus-DP gewohnt.)

Zumeist werden ein Automatisierungsrechner als CANopen Master eingeführt und den Geräten ID-Adressen (Node-ID) vergeben. Die ersten vier Slave verfügen wie o.a. über jeweils vier RPDO / TPDO. CANopen vergibt in Abhängigkeit von der Geräte-ID die Default-Identifier (COB_ID). Im Master werden zu jeder vergebenen Geräte-PDO die jeweils korrespondierenden PDO angelegt.

Bild 2.4-12 zeigt schematisch ein Beispiel für das Mappen nach dem Master-Slave-Default. Der Master sowie jedes der drei als Slave projektierte CAN-Geräte verfügen jeweils über vier Sende und Empfangsobjekte, von denen im Bild nicht alle genutzt werden. Das ist u.a. bereits in der begrenzten Anzahl von Objekten im Master begründet.

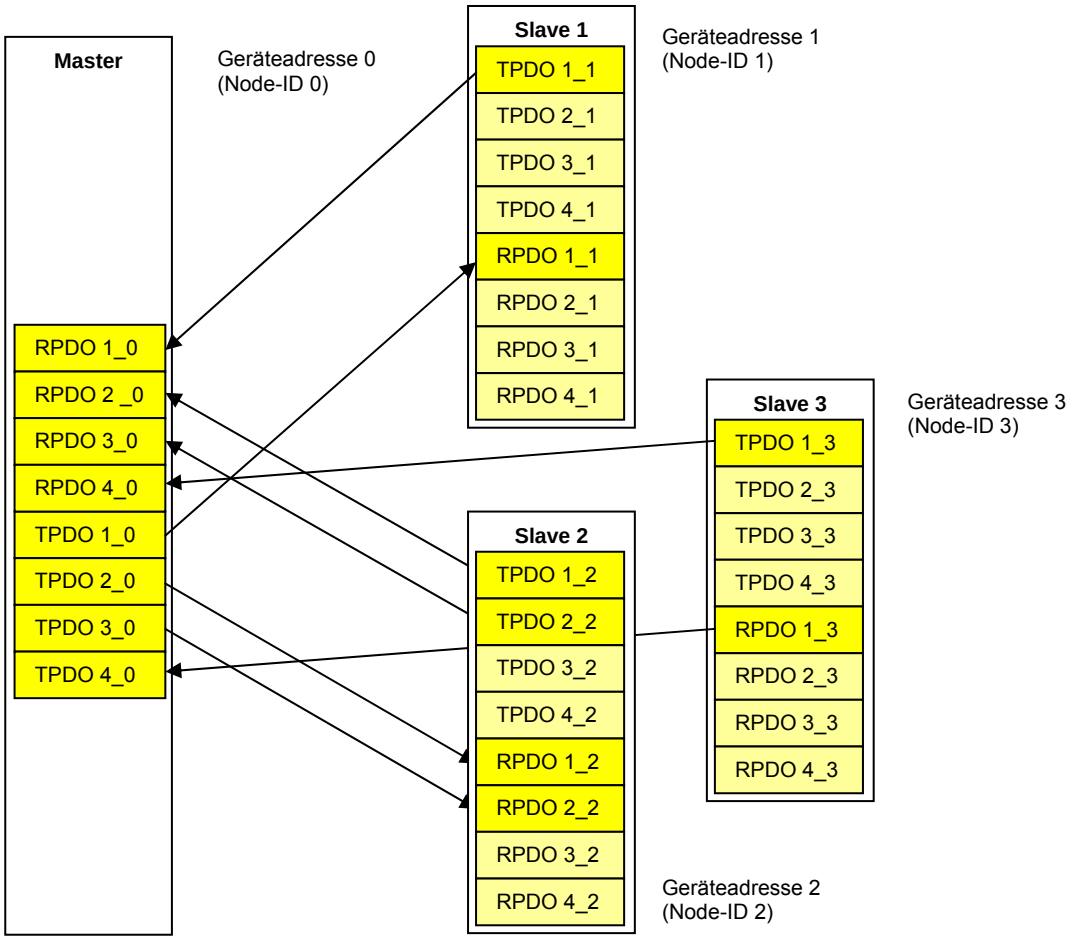

Bild 2.4-12: Beispiel für das Mappen gemäß Default Predefined Master Slave Connection Set (kurz Master-Slave-Default)

Unter **PDO-Linking** versteht man ein ähnliches Vorgehen, um ausgehend von der Master-Slave-Kommunikation gleichfalls auf Basis von Geräteadressen die direkte Kommunikation zwischen Slave's ohne Umweg über einen Master zu parametrieren. Per Master-Slave-Default "hört" zunächst kein Slave auf gesendete Objekte anderer Slave.

Zum automatischen PDO-Mapping gehört die Zuordnung der Daten zu den automatisch angelegten PDO. Die Art dieser Konfiguration soll an einem der meistverwendeten CAN-Geräte - einer I/O-Baugruppe - beispielhaft erläutert werden:

Im Profil DS-401 der CiA mit Namen "CANopen Device Profile für Generic I/O Modules" werden Regeln für die Zuordnung der einzelnen I/O-Signale festgelegt. Danach erfolgt automatisches PDO-Mappen für bis zu 64 binäre I/O-Signale in PDO1 und bis zu 12 analoge Signale in den PDO 2 bis 4. Je PDO mit 8 Byte Daten können bis zu 4 analoge Signale "transportiert" werden, da analoge Signale stets ein Wort umfassen.

Bus- bzw. Geräteadressen, COB_ID, Signale und PDO eines I/O-Gerätes sind wie folgt zugeordnet:

Binäre Eingangssignale 0…63	⟶	TPDO 1 mit COB_ID = Node-ID + 180_H
Analoge Eingangssignale 0…3	⟶	TPDO 2 mit COB_ID = Node-ID + 280_H
Analoge Eingangssignale 3…7	⟶	TPDO 3 mit COB_ID = Node-ID + 380_H
Analoge Eingangssignale 8…11	⟶	TPDO 4 mit COB_ID = Node-ID + 480_H

RPDO 1 mit COB_ID = Node-ID + 200_H	⟶	Binäre Eingangssignale 0…63
RPDO 2 mit COB_ID = Node-ID + 300_H	⟶	Analoge Eingangssignale 0…3
RPDO 3 mit COB_ID = Node-ID + 400_H	⟶	Analoge Eingangssignale 3…7
RPDO 4 mit COB_ID = Node-ID + 500_H	⟶	Analoge Eingangssignale 8…11

Vergibt man zum Beispiel der I/O-Baugruppe die Busadresse (Node-ID) 4_H, so ist
die COB-ID für das Senden der binären Signale mit TPDO 1: 184_H;
die COB-ID für das Empfangen der binären Signale mit RPDO 1: 204_H

Damit steht der Projektant bei Anwendung des Master-Slave-Defaults zumeist nur noch vor der Aufgabe, die Busadressen (Node-ID) zu vergeben und die Baud-Rate des CAN-Bus festzulegen. Allerdings gibt es weiter mit dem **Netzwerkmanagement (NMT)** eine Vielzahl von Einflussmöglichkeiten auf die Arbeitsweise des CAN, von denen nachfolgend nur einige genannt werden.

Das NMT wird als Master-Slave-System ausgelegt, "fliegende Master" sind möglich. Während bei der klassischen Master-Slave-Architektur wie z.B. Profibus die Kommunikation zyklisch erfolgt, bleibt diese bei CAN vom Grundsatz her **ereignisgesteuert.** Eine TPDO wird deshalb nur dann gesendet, wenn sich ein Wert ändert. Diese nichtzyklische Übertragungsart wird auch als asynchrone Datenübertragung bezeichnet. Dennoch gibt es bei CANopen die Möglichkeit, ein **Synchronisationsobjekt (Sync)** zu verwenden, um einen Abgleich aller Prozessabbilder in den Geräten zu erreichen oder aber auch **Daten in bewusst vorgegebenen Zeitabständen abzufragen**, z.B. in Regelungen.

Zur Überwachung aller Geräte am Bus kann die **Heartbeat** (Herzschlag) – Nachricht herangezogen werden. Bei diesem Prinzip melden alle Busteilnehmer ihre ungestörte Funktion in Form eines Lebenszeichens an den Master

Als Alternative kann projektiert werden, dass der Master die ungestörte Funktion der Geräte zyklisch abfragt und die Antworttelegramme auswertet (**Node Guarding**). Gleichzeitig kann durch Prüfung der regelmäßigen Telegrammanfrage auch der Master von den Slave's aus überwacht werden.

Weiter gibt es Möglichkeiten, durch Vorgabe von **Event-Time und Inhibit-Time** auf die zeitliche Abfolge der Kommunikation Einfluß zu nehmen. Inhibit-Time wirkt als Sende-Sperr-Zeit, Event-Time erzwingt ein Senden, auch wenn sich der Wert nicht ändert.

2.4.4 Einfachste Applikation mit CANopen Controllern WAGO-I/O-750-837

Gegenüber dem bereits deutlich vereinfachten Vorgehen beim Master-Slave-Default lassen sich mit bestimmter Firmware und Bibliotheken die erforderlichen Arbeiten zur Parametrierung der Kommunikation von CAN-Geräten noch weiter reduzieren. Ein Beispiel dafür ist die nachfolgende angegebene Verschaltung zweier CAN-Controller 837 aus dem Busklemmensystem WAGO-I/O-750.

Die Darlegungen der hier zu öffnenden Datei CANopen mit Wago sind ein verkürzter und wenig veränderter Auszug aus der Projektarbeit: von Kai Fieber und Ronny Kaufhold mit dem Titel "Darstellung der Datenkommunikation über CANopen mit WAGO-I/O-750, der erforderlichen Parametrierung und Hardwarekonfiguration an einer Beispielapplikation".
Sie wurde durchgeführt im Fachbereich Elektrotechnik der FH Schmalkalden im Zeitraum SS07 bis WS07/08.

Für diese Arbeit standen lediglich zwei CAN-Controller WAGO-I/O-750-837 mit älterer Firmware Vers. 2.1 zur Verfügung. Deshalb musste auf das ältere Softwaretool WAGO-I/O-PRO32 zurückgegriffen werden.
Das Verschalten der CAN-Controller beschränkt sich hier auf den Einsatz der selbsterklärende Software-Bausteine der Bibliothek CiA405.lib
NMT_ADDNODE
NMT_SLAVE_IN_NW_LIST
NMT_CHANGE_STATE

Im üblichen Adressbereich ab Byte 512 sind die Prozessabbilder des jeweils anderen Knotens verfügbar, so dass die Signalzustände leicht in Programme eingebunden werden können.
In der Arbeit wurde festgestellt, dass die elegantere Arbeit mit Netzwerkvariablen nicht möglich war, da die o.a. Bibliothek für die Kommunikation eines Controllers mit Kopplern ausgelegt ist.
Neuere, jedoch in erheblichem Maße kostenintensive CANopen-Bibliotheken des Unternehmens 3S-Smart Software Solutions GmbH erlaubt die Kommunikation mit Netzwerkvariablen über CAN.