

**FI**

**Teil 2**

# **Bussysteme**

klin, 12.09.2021

<b>1</b>	<b>Grundlagen Bussysteme .....</b>	<b>2</b>
1.1	Vorteile von Bussystemen.....	2
1.2	Merkmale und Anwendungsbereiche.....	2
1.3	Automatisierungspyramide .....	4
1.4	Busarchitekturen .....	5
1.5	Das ISO/OSI-Schichtenmodell.....	6
1.6	Fehlererkennung.....	8
1.7	Buszugriffssteuerung.....	11
1.8	Strukturelemente von Netzwerken.....	16
<b>2</b>	<b>Serielle Schnittstellen .....</b>	<b>19</b>
2.1	Serielle Schnittstelle RS-232.....	20
2.2	Serielle Schnittstellen RS-422 und RS-485 .....	22
<b>3</b>	<b>Chip-Bussysteme .....</b>	<b>25</b>
3.1	I <sup>2</sup> C-Bus (TWI).....	25
3.2	SPI - Serial Peripheral Interface .....	30
3.3	1-Wire Bussystem .....	33
3.4	Vergleich I2C, SPI, 1Wire .....	36
<b>4</b>	<b>Bussysteme .....</b>	<b>37</b>
4.1	HART-Protokoll .....	37
4.2	AS-Interface .....	40
4.3	IO-Link.....	46
4.4	KNX-Bus .....	48
4.5	CAN-Bus.....	55
4.6	CANopen Protokoll .....	59
4.7	Profibus.....	64
<b>5</b>	<b>Industrial Ethernet .....</b>	<b>68</b>
5.1	Allgemeines .....	68
5.2	Auswahlkriterien für Industrial-Ethernet Bussysteme .....	71
5.4	Modbus-TCP.....	72
5.5	Profinet .....	75
5.6	EtherCAT .....	79
5.7	OPC Unified Architecture .....	84

# 1 Grundlagen Bussysteme

Ein Bus ist ein **System zur Datenübertragung zwischen mehreren Teilnehmern** über ein gemeinsames Übertragungsmedium. Die an einem Bus angeschlossenen Komponenten werden auch als **Busteilnehmer** oder **Busknoten** bezeichnet.

Ein **Feldbussystem** ist ein **Datennetzwerk auf der industriellen Ebene**. An diesem Netzwerk können I/O Module, Sensoren und Aktoren mit einem Steuerungsrechner verbunden werden. Feldbussysteme sind leistungsfähiger, flexibler und kostengünstiger als konventionelle Verdrahtung und haben diese praktisch vollständig verdrängt.

## 1.1 Vorteile von Bussystemen

- **Projektierungs- und Installationskosten**  
günstige Kabel, einfache Pläne, geringer Platzbedarf, schnelle und fehlersichere Verdrahtung
- **einfache Erweiterbarkeit**  
zusätzliche Busteilnehmer über einfache Busverlängerung und Konfiguration in der Software
- **Parameter- und Diagnosefunktionen**  
Parametrierung und laufende Überwachung aller Busteilnehmer im Betrieb

## 1.2 Merkmale und Anwendungsbereiche

Aktuell ist eine Vielzahl an unterschiedlichen Bussystemen erhältlich. Die Systeme unterscheiden sich je nach Einsatzgebiet in der **Netzarchitektur**, im **Datendurchsatz**, in der **Echtzeitfähigkeit** und der **Übertragungstechnik**.

Bussysteme sind oftmals für einen bestimmten **Anwendungsbereich** konzipiert, was aber nicht bedeutet, dass jeder Feldbustyp nur für eine bestimmte Anwendung verwendbar ist.

	Fertigungsautomation (Roboter)	Prozessautomation (Chemie/Food)	Gebäudeautomation (Heizung/Lüftung)
Zykluszeiten	1 ms	100 ms	100 ms
Anzahl I/O	< 100	> 100	100...1000
Kabellängen	< 100 m	> 1000 m	> 1000 m
Störsicherheit	hoch	hoch	mittel
Aufgaben	Ablauf steuern, regeln, überwachen	Ablauf steuern, regeln, überwachen	Überwachen, para- metrieren, Service
Beispiele	Profibus DP, Interbus, ControlNet, CAN	Profibus PA, Foundation Fieldbus	LON, EIB, Modbus-TCP (Ethernet)

Abbildung 1 Bereiche der Automation

Jede dieser beispielhaft genannten Anwendungen stellt unterschiedliche Anforderungen an die Übertragungstechnik und die Echtzeitfähigkeit. In der Automatisierungstechnik geht aber bei **allen Herstellern** der Trend eindeutig in Richtung **Industrial- Ethernet**.

### Beispiel Bussysteme im Automobil:

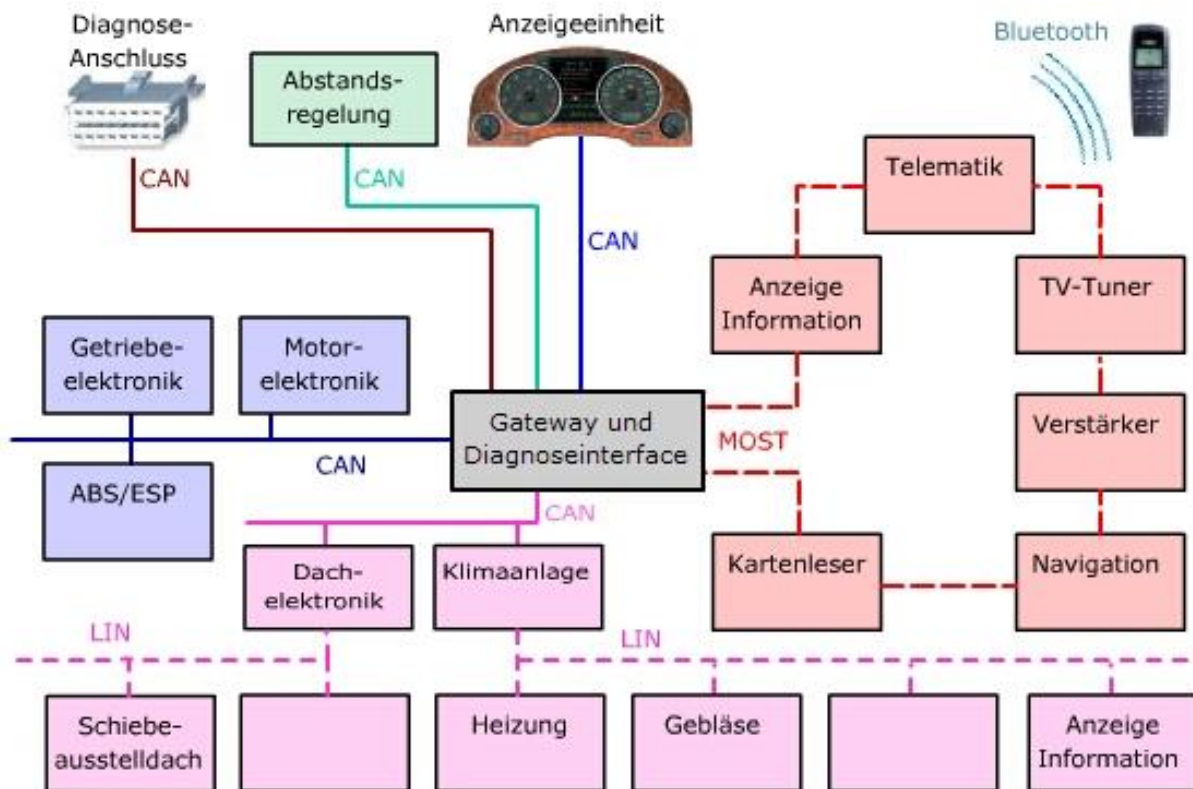


Abbildung 2 Beispiel Bussysteme im Automobil

**CAN-Bus:** existiert in verschiedenen Varianten. In Abhängigkeit von der Geschwindigkeit, mit der er Daten übertragen kann, wird er in den Bereichen Antrieb, Komfort und zur Diagnose eingesetzt. Low Speed bis High Speed, 100 bis 500 kBit/s incl. Diagnose-CAN

**LIN-Bus (Local Interconnect Network):** ist ein preiswertes dem CAN-Bus untergeordnetes Sub-Bussystem für die Anbindung lokaler Komfort-Komponenten und Sensoren - max. 20 kbit/s.

**MOST (Media Oriented Systems Transport)** optisches oder elektrisches Hochgeschwindigkeits-Multimedia-Bussystem, meist ringförmig, sehr hohe Übertragungsrate bis 23 Mbit/s.

**Bluetooth:** Funkstrecke erfordert keine Verkabelung und dient z.B. zur Übertragung von Sprach-Signalen.

### 1.3 Automatisierungspyramide

Die Informationen, die in den **verschiedenen Prozessebenen** zu übertragen sind, können nach Menge und Aufgabe der benötigten Komponenten in einer **Pyramidenstruktur** dargestellt werden.

#### 1.3.1 Prozessebenen

- **Leitebene:** Bedienen, Beobachten, Produktionsplanung und -datenerfassung
- **Steuerungsebene:** SPS, NC, Robotersteuerung, I/O Busknoten, Antriebe, Feldgeräte
- **Feldebene:** Antriebe, I/O-Klemmen, allgemein Sensoren und Aktoren,
- **Sensor / Aktor-Ebene:** Fühler Taster, Schalter



Abbildung 3 Automatisierungspyramide

#### Leitebene:

In der Leitebene werden übergeordnete, den gesamten Prozess betreffende Aufgaben bearbeitet (**Managementfunktionen**). Dazu gehören **Bedienen** und **Beobachten**, Verwaltung von Teilprogrammen und Rezepturen, sowie die **Datenarchivierung**.

In der Leitebene sind in der Regel **große Datenmengen** zu übertragen. Die Kommunikation erfolgt über **Ethernet-TCP/IP**. Die Reaktionszeiten liegen dabei im Bereich **< 10 s**.

#### Steuerungs- oder Zellebene:

In der Steuerungs- oder Zellebene werden alle **Automationsaufgaben** von Automatisierungsgeräten wie **Maschinen- oder Anlagensteuerungen** ausgeführt. Die Automatisierungsgeräte sind in der Regel mit **lokalen Bedienstationen** ausgestattet.

In der Steuerungsebene sind **mittlere Datenmengen** zu übertragen. Die Kommunikation zwischen den Automatisierungsgeräten erfolgt über **Feldbussysteme**. Die Reaktionszeiten liegen im Bereich **< 100 ms**.

**Feldebene:**

Die Feldebene ist das **Bindeglied zwischen den Automatisierungsgeräten** und den zu steuernden **Maschinen- und Anlagenteilen**. Die Feldgeräte messen, melden und geben die Befehle der Steuerungsebene an die Maschinen- und Anlagenteile weiter. Es werden überwiegend kleine Datenmengen übertragen. Die Übertragung erfolgt meist **zyklisch** mit Reaktionszeiten **< 10 ms**.

**Sensor/Aktor-Ebene:**

In dieser Ebene kommuniziert ein Master mit an seinem **Subnetz** angeschlossenen Sensoren und Aktoren. Kennzeichen sind schnelle Reaktionszeiten von ca. **1..10 ms** für wenige Datenbits.

**1.4 Busarchitekturen**

Die räumliche Verbindung der Busteilnehmer wird mit dem Begriff **Topologie** näher spezifiziert. Die Topologie eines Netzes ist entscheidend für seine **Ausfallsicherheit**. Nur wenn alternative Wege zwischen den Knoten existieren, bleibt bei Ausfällen einzelner Verbindungen die Funktionsfähigkeit teilweise oder ganz erhalten. In der Praxis unterscheidet man zwischen **Linien-**, **Stern-** und **Ringtopologie** und Kombinationen davon.

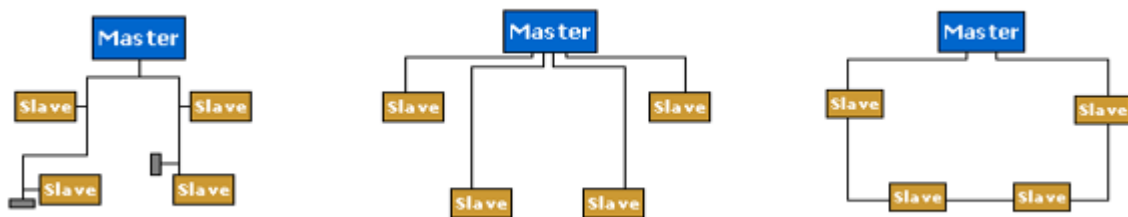


Abbildung 4 Master-Slave Busarchitekturen: Linie(Bus), Stern, Ring

**Bus/Linientopologie:**

Mehrere Teilnehmer werden an eine Busleitung über kurze Stichleitungen angeschlossen. Die Bus-Enden werden zur Vermeidung von Reflexionen terminiert. Eine Störung im Kabel blockiert den gesamten Netzstrang. Es kann immer nur ein Slave Daten senden.

**Sterntopologie:**

Der Master ist der Sternmittelpunkt. Jeder Teilnehmer ist über eine eigene Leitung angekoppelt. Die gesamte Kommunikation wird vom Master gesteuert und über diesen abgewickelt. Fällt der Master aus, so ist keine Kommunikation mehr möglich. Mehrere Slaves können gleichzeitig Daten senden. Der Ausfall eines Teilnehmers hat keine Auswirkung auf den Rest des Bussystems.

**Ringtopologie:**

Jede Station ist sowohl Sender als auch Empfänger. Die Daten umkreisen den Ring genau einmal und werden dabei von Station zu Station weitergereicht. Jede Station prüft, ob die Daten an sie gerichtet sind. Ist dies der Fall, übernimmt sie die Eingangsdaten und überträgt eigene Ausgangsdaten in das Telegramm. Eine Störung im Kabel blockiert den gesamten Ring.

## 1.5 Das ISO/OSI-Schichtenmodell

Im OSI-Schichten-Modell wird beschrieben, welche Voraussetzungen gegeben sein müssen, damit unterschiedliche Netzwerkkomponenten miteinander kommunizieren können. OSI steht für „Open System Interconnection“ und heißt übersetzt „**Offenes System für Kommunikationsverbindungen**“.

Die **International Standards Organization** (ISO) setzte 1977 ein Komitee ein, um das OSI-Modell zu entwickeln. Im Jahre **1983 wurde das Modell zum internationalen Standard erklärt**. Das **ISO/OSI - Referenzmodell** (ISO 7498) dient zur abstrakten Beschreibung der Interprozesskommunikation zwischen räumlich entfernten Kommunikationspartnern.

Das OSI-Modell basiert auf dem **Schichtenansatz**, bei dem jede Schicht der darüber liegenden Schicht eine Reihe **zusammengehöriger Dienste** zur Verfügung stellt und aus der darunter liegenden Schichten einfachere Funktionen anfordert. Die einzelnen Schichten sind so definiert, dass Veränderungen an einer Schicht keine Veränderung an den anderen Schichten erforderlich machen.





	Schicht	Bezeichnung		Funktion
Anwenderorientierte Schichten	7	Anwendungsschicht (Application layer)		Kommunikationsabläufe der Anwendung
	6	Darstellungsschicht (Presentation layer)	Darstellung	Systemunabhängige Datendarstellung
	5	Sitzungsschicht (Session layer)	%% I %%	Verbindung aufbauen, halten und abbauen
Transportorientierte Schichten	4	Transportschicht (Transport layer)		Sichere Verbindung zwischen Prozessen herstellen
	3	Vermittlungsschicht (Network layer)		End-zu-End Verbindung zwischen Rechnern
	2	Sicherungsschicht (Data link layer)	10111111010010110 Start Adresse	Datenübertragung zwischen benachbarten Stationen
	1	Bitübertragungsschicht (Physical layer)		Physikalische Übertragung von Signalen

Abbildung 5 ISO / OSI-Schichtenmodell

**Feldbussystemen** verwenden in der Regel nur die **Schichten 1, 2 und 7**. Einige auf **Ethernet** und dem TCP/IP-Protokoll aufbauende Feldbusse nutzen auch die **Schichten 3 und 4**.

### 1.5.1 Bitübertragungsschicht (Physical layer), Schicht 1

Die Bitübertragungsschicht ist für die **Übertragung der Bitströme über das Übertragungsmedium** (Kabel, Funk) zuständig. Sie legt die elektrische, mechanische und funktionale Schnittstelle zum Übertragungsmedium fest.

Parameter der Bitübertragungsschicht:

- Übertragungsmedium (Kupfer, Glasfaser, Funk)
- Die Funktion der einzelnen Leitungen (Datenleitung, Steuerleitung)
- die Übertragungsrichtung (simplex, halb-duplex, duplex)
- Übertragungsgeschwindigkeit

### 1.5.2 Sicherungsschicht (Data Link Layer), Schicht 2

Die Sicherungsschicht sorgt für den **zuverlässigen Austausch von Datenpaketen** zwischen den Stationen. Sie wird in zwei Unterschichten, die MAC-Schicht (Medium Access Control) und in die LLC-Schicht (Logical Link Control) unterteilt.

Aufgaben der Sicherungsschicht:

- Zugriffssteuerung auf das Übertragungsmedium (MAC)
- Schreiben der physikalischen Sende- und Empfangsadressen in die Datenpakete (MAC)
- Aufteilen des Bitdatenstroms in Datenpakete (LLC)
- Fehlererkennung bei der Datenübertragung und Korrektur (LLC)

### 1.5.3 Netzwerk- oder Vermittlungsschicht (Network Layer), Schicht 3

Die Netzwerkschicht **steuert den Austausch von Datenpaketen**, da diese nicht direkt an das Ziel vermittelt werden können und deshalb mit Zwischenzielen versehen werden müssen. Die Datenpakete werden dann von Knoten zu Knoten übertragen bis sie ihr Ziel erreicht haben.

Aufgaben der Netzwerkschicht:

- Identifikation einzelner benachbarter Netzknoten
- Auf und Abbau von Verbindungskanälen (Routing-Tabellen)
- Wegfindung vom Sender zum Empfänger durch logische Adressierung (Routing)

### 1.5.4 Transportschicht (Transport Layer), Schicht 4

Die Transportschicht ist die oberste Schicht des Transportsystems und damit das **Bindeglied zu den anwendungsorientierten Schichten**. Hier werden die Datenpakete einer Anwendung zugeordnet.

Aufgaben der Transportschicht:

- Auf- und Abbau von Verbindungen
- Zuordnung von Datenpakete zu Applikationen
- Segmentierung von Datenpaketen beim Senden
- Richtige Zusammensetzung der Datenpakete beim Empfangen



### 1.5.5 Sitzungsschicht (Session Layer), Schicht 5

Die Sitzungsschicht ist die unterste Schicht des Anwendungssystems (Schicht 5-7) und baut **logische Verbindungen zwischen Sender und Empfänger** auf, kontrolliert diese und beendet sie wieder.

Es wird beispielsweise festgelegt, wie eine Sitzung zeitlich ablaufen hat: Aufforderung zum Senden eines Passwortes, Senden des Passwortes, Bestätigung des Passwortes.

### 1.5.6 Darstellungsschicht (Presentation Layer), Schicht 6

Die Präsentationsschicht fungiert als Dolmetscher, indem sie die **Datenpakete in das jeweilige Format des Sender- oder Empfängers übersetzt**. Datenkompression und Datenverschlüsselung gehören auch zu ihren Aufgaben. Formate und Codierungen dieser Schicht: ASCII, JPEG, HTML, Unicode

### 1.5.7 Anwendungsschicht (Application Layer), Schicht 7:

Die Anwendungsschicht ist die **Schnittstelle zur eigentlichen Benutzeranwendung**. Hier werden die Netzwerkdaten in vom Benutzer verwendbare Daten umgewandelt.

Beispiele von Benutzeranwendungen: Telnet, FTP, HTTP, SMTP

## 1.6 Fehlererkennung

Bei der Übertragung von Bitfolgen kann es zu Fehlern kommen. Fehler müssen zumindest erkannt werden. Je nach Anwendungsfall kann es auch nötig sein, Fehler zu korrigieren.

### Voraussetzungen zur Fehlererkennung:

- Übertragung redundanter Information in Form von Prüfbits
- Gemeinsame Berechnungsvorschrift für Prüfbits bei Sender und Empfänger
- Übertragung eines Datenstroms in unterteilten Einheiten (Blöcke, Pakete, Rahmen)

### Fehlererkennung – Fehlerkorrektur:

Kodierungen mit Fehlerkorrektur sind da sinnvoll, wo keine Sendewiederholungen möglich sind. Üblicherweise wird bei **Feldbussystemen** die reine **Fehlererkennung** bevorzugt. Wird ein Fehler erkannt, wird das Rahmen einfach verworfen. Ein erneutes Anfordern eines fehlerhaften Rahmens sehen die etablierten Protokolle der Sicherungsschicht nicht vor. Die **Fehlererkennung ist** wegen der weniger erforderlichen Anzahl von Prüfbits **effizienter als die Fehlerkorrektur**.

### 1.6.1 Hamming-Distanz

Die **Hamming-Distanz** eines Codes ist die Anzahl der Stellen, an denen sich die einzelnen Codewörter unterscheiden. Die Hamming-Distanz ist ein Maß für die Störfestigkeit eines Codes.

Beispiel: Code bestehend aus zwei Codewörtern und Hamming-Distanz 3:

1. **101011**
2. **110010**

Erkennbare Fehler:  $D - 1 = 2$

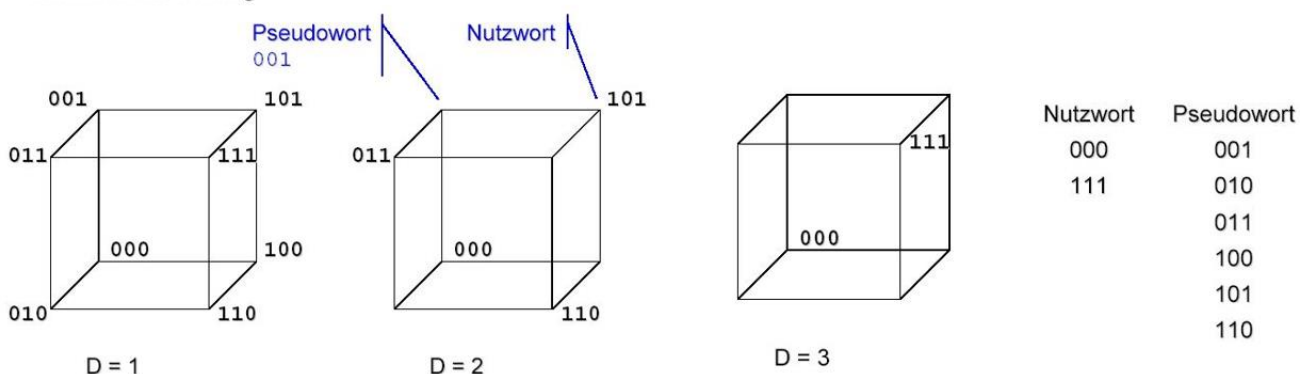
Korrigierbare Fehler:  $(D - 1) / 2 = 1$

Codes, die einen Fehler korrigieren können, werden auch als **Hamming-Codes** bezeichnet. Ein Hamming-Code hat also eine Hamming-Distanz  $\geq 3$ .

Beispiel: Übertragung eines Codes mit 3 Bits:

1. **D = 1:** Für die Codierung von 8 Werten werden alle 3 Bit verwendet. Fällt ein Bit um, ergibt sich wieder ein gültiger Code  $\rightarrow D - 1 = 0 \rightarrow$  keine Fehlererkennung möglich
2. **D = 2:** Fällt ein Bit im **Nutzwort** um, entsteht ein sogenanntes **Pseudowort**, aufgrund der Hamming - Distanz  $D = 2$  kann ein Fehler erkannt, aber nicht korrigiert werden, da jedes der drei Nutzwörter übertragen worden sein kann.
3. **D = 3:** Die beiden Nutzwörter des Codes unterscheiden sich an drei Stellen, damit kann ein 2 Bit - Fehler erkannt und ein 1 Bit - Fehler korrigiert werden. Das Pseudowort 001 kann mit hoher Wahrscheinlichkeit dem Nutzwort 000 zugeordnet werden.

Räumliche Darstellung



### 1.6.2 Paritätsbit

Durch Anhängen eines Paritätsbits wird eine Bitfolge so erweitert, dass die Anzahl der "1" in der Bitfolge immer gerade (even parity) oder ungerade (odd parity) ist.

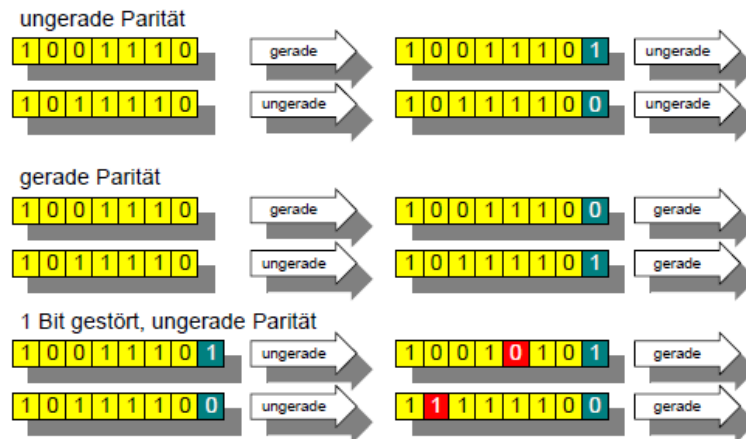


Abbildung 6 Paritätsbit

### 1.6.3 Prüfsumme

Eine einfache Möglichkeit ist die Bildung von einer Prüfsumme über einen zu übertragenden Datenblock. Damit der Aufwand für die zusätzliche Übertragung nicht zu groß wird, überträgt man nur das erste Bytes der Prüfsumme (Modulo 1Byte Rechnung).

Start	Adresse	Funktion	Daten	Prüfsumme	Ende
1 Zeichen (:)	2 Zeichen	2 Zeichen	n Zeichen	1 Zeichen	(CR) (LF)

Abbildung 7 Telegrammaufbau Modbus ASCII

### 1.6.4 Zyklische Redundanzprüfung CRC (Cyclic Redundancy Check)

Ist das meist verwendete Verfahren zur Fehlererkennung und basiert auf der Grundlage der Polynomdivision. Mit dem CRC-Verfahren können Fehler detektiert aber nicht korrigiert werden.

#### Bitfolgen und Polynome:

Eine Bitfolge kann einfach durch eine Polynomdarstellung mit  $x = 2$  beschrieben werden.

Beispiel:

Bitfolge 0010000101 -> Polynom  $1+x^2+x^7$

Bitfolge 1110011000 -> Polynom  $x^3+x^4+x^7+x^8+x^9$

### CRC-Berechnung:

Um eine redundante Information zu einer **Nachrichtenbitfolge N(X)** zu ermitteln, verwendet das CRC-Verfahren den **Rest der Polynomdivision** durch ein so genanntes **Generatorpolynom G(X)**.

- Sender und Empfänger vereinbaren ein Generatorpolynom **G(x)**
- Es wird eine **Polynomdivision** mit dem zu übertragenden Bitfolge **N(X)** durchgeführt
- **G(X)** wird mit dem berechneten **Rest** ergänzt und **übertragen**.
- Der **Empfänger prüft** mit Kenntnis **G(X)** die empfangene Nachricht

### Umsetzung:

- Berechnung erfolgt ohne Berücksichtigung von Überträgen (Modulo-2 Rechnung)
- Berechnung kann einfach in Hardware realisiert werden
- Generatorpolynome werden so konstruiert, dass bestimmte Fehlerklassen detektiert werden

Häufig verwendete Generatorpolynome:

CRC8 (1-Wire)	$X^8+X^5+X^4+1$
CRC16 (CRC-CCITT)	$X^{16}+X^{12}+X^5+1$
CAN-Bus	$X^{15}+X^{14}+X^{10}+X^8+X^7+X^4+X^3+1$
CRC-32 (CSMA/CD)	$X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^8+X^7+X^5+X^4+X^2+X+1$

## 1.7 Buszugriffssteuerung

Befinden sich mehrere Teilnehmern auf einer Leitung, muss ein Zugriffsverfahren entscheiden, welcher Teilnehmer den Bus belegen darf. Bei den Zugriffsverfahren unterscheidet grundsätzlich zwischen **kontrollierten, deterministischen** und **zufälligen, stochastischen** Zugriff.

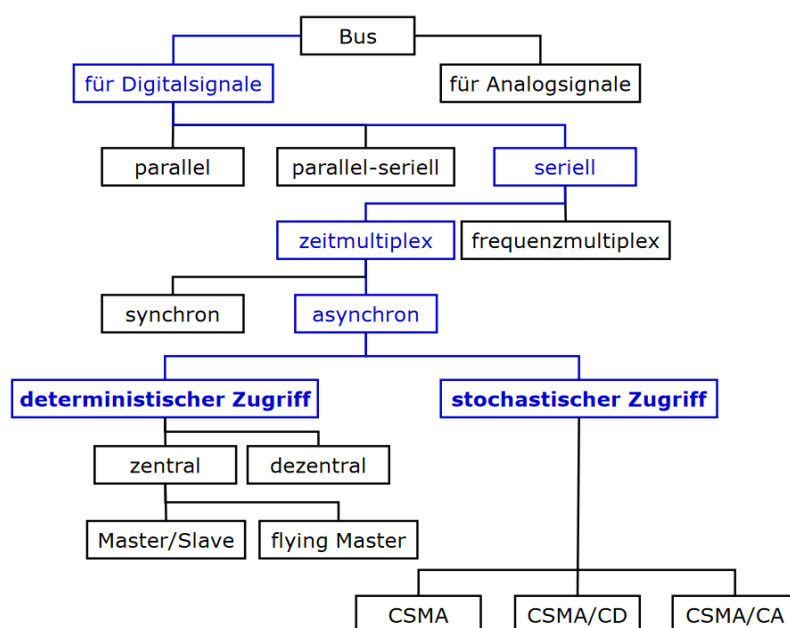


Abbildung 8 Klassifikation von Feldbussystemen

### Zufälliger, stochastischer Buszugriff

Die Teilnehmer können den Bus belegen, sobald dieser frei ist. Da dies jedoch durch mehrere Teilnehmer gleichzeitig erfolgen kann, sind je nach Verfahren Regelungen zur Verhinderung oder Auflösung des gleichzeitigen Zugriffs erforderlich.

- Zufallssteuerung (random control)
  - **CSMA / CD**-Verfahren - Carrier Sense Multiple Access / Collision Detection
  - **CSMA / CA**-Verfahren - Carrier Sense Multiple Access / Collision Avoidance

### Kontrollierter, deterministischer Buszugriff

Bei Verfahren mit kontrolliertem Buszugriff wird darüber hinaus unterschieden, ob die Vergabe des Buszugriffsrechts durch eine **zentrale Steuerung (Master)** oder **dezentral, verteilt** durch Absprache zwischen den Teilnehmern erfolgt.

- Zentrale Steuerung (centralized control)  
Eine Station kontrolliert das ganze Netzwerk. Andere Stationen erhalten von dieser die Sendeberechtigung individuell zugeteilt (
  - **Master / Slave** Verfahren (mit Polling)
- Verteilte Steuerung (distributed control)  
Nur eine einzige Station hat zu einem bestimmten Zeitpunkt das Recht auf dem Kanal zu senden. Die Verfahren mit diesem Konzept heißen:
  - **TDMA** - Verfahren (Time Division Multiple Access)
  - **Token Passing** - Verfahren (token bus, token ring)

#### 1.7.1 CSMA-Verfahren

Beim CSMA Verfahren haben die Station **ohne explizite Zuteilung Zugriff zum Übertragungs-medium**. Die Einschränkung besteht darin, dass eine Station nicht senden darf, wenn das Medium bereits durch eine andere Station belegt ist, weil beim gleichzeitigen Senden zweier Stationen beide Nachrichten zerstört werden.

Kollisionsverfahren eignen sich vor allem für kleinere Netze. Bei größerem Datenverkehr steigt die Anzahl der Kollisionen, wodurch der Datendurchsatz des Netzes erheblich absinken kann. Abhilfe schafft eine Segmentierung des Netzwerks (Switch-Technologie bei Ethernet).

#### 1.7.2 CSMA / CD-Verfahren

Grundprinzip von CSMA / CD (Carrier Sense Multiple Access with **Collision Detection**):

- **Erkennung von Kollisionen durch Datenabgleich** (Vergleich gesendet – auf dem Medium)
- **Sendewiederholung nach teilnehmerspezifischer Wartezeit**

Bevor eine Station zu senden beginnt, überzeugt sie sich durch Abhören der Leitung, dass diese frei ist (**Carrier Sense**). Ist die Leitung frei, darf gesendet werden. Dabei ist nicht auszuschließen, dass eine andere Station ebenfalls zu senden beginnt (**Multiple Access**).

Kollisionen entstehen auf Grund der Signallaufzeiten von einer Station zur anderen. Während der Signallaufzeit wird kein Signal erkannt, obwohl es bereits auf dem Weg ist. Die Signallaufzeit beträgt ungefähr **5nsec / m**.

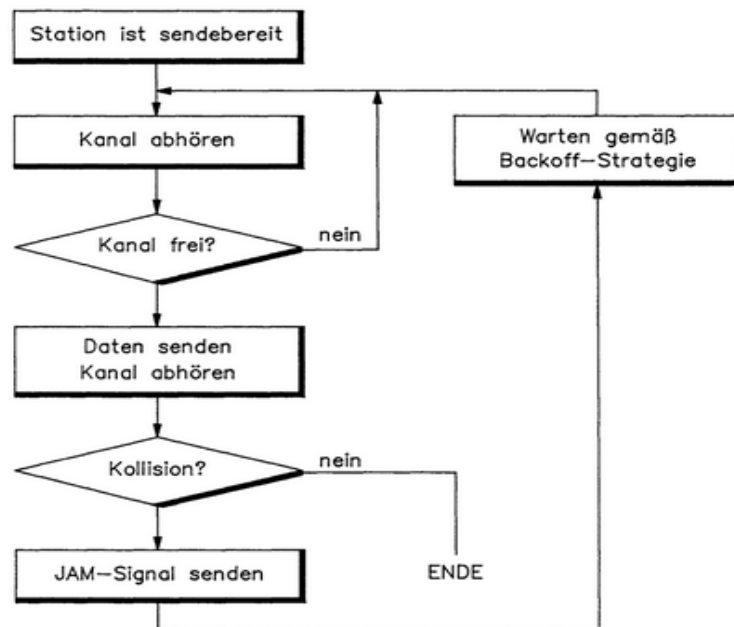


Abbildung 9 CSMA / CD Sendeablauf

Erkennt eine Station durch Abhören der Leitung eine Kollision, wird die Übertragung sofort abgebrochen und ein **Jam-Signal** gesendet. Das Jam-Signal lässt alle weiteren sendenden Stationen die Kollision erkennen. Die Sendung wird daraufhin von jeder Station abgebrochen und durch einen Zufallsgenerator eine neue Wartezeit in jeder Station erzeugt. Nach Ablauf dieser Wartezeit finden neue Sendeveruche statt.

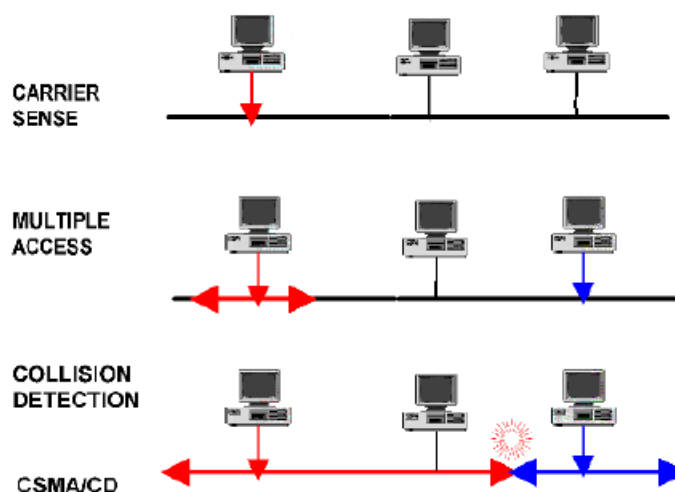


Abbildung 10 Kollision bei CSMA /CD

**Eigenschaften:**

- bei geringer Busbelastung hoher Datendurchsatz
- kurze Latenzzeit im Niederlastbereich
- im Hochlastbereich lange Wartezeiten
- Das Verfahren ist nicht deterministisch, daher ist das Verfahren **nicht echtzeitfähig**

**Anwendungen:**

CSMA/CD ist das **Zugriffsverfahren von Ethernet**. CSMA/CD ist bei einer Linien-Topologie zwingend erforderlich, weil dort alle Teilnehmer über eine Busleitung verbunden sind. Praktisch werden heute Ethernet-Netze vollständig *geswitched*. Switches brechen die Linienstruktur auf und wandeln die Topologie in einen sternförmigen Aufbau, sodass keine Kollisionen vermieden werden.

### 1.7.3 CSMA / CA-Verfahren

Grundprinzip CSMA / CA (Carrier Sense Multiple Access with **Collision Avoidance**):

- **Vermeidung von Kollisionen durch Prioritätsregeln**

Beim gleichzeitigen Zugriff von zwei Teilnehmern auf das Medium setzt sich derjenige mit der höheren Priorität durch, während sich der andere wieder zurückzieht, um es zu einem späteren Zeitpunkt noch einmal zu versuchen.

**Prioritätsregeln:**

- Bei **leitungsbasierter Kommunikation** bestimmt die **Teilnehmer- oder Nachrichtendresse**: Am I2C-, KNX- oder CAN-Bus besitzen z.B. kleinere Adressen aufgrund des elektrischen Busaufbaus höhere Priorität (Open-Drain / Wired AND Ausgänge: LOW-Pegel ist dominant).
- In **Wireless-LAN** Netzwerken bestimmt eine zufällige **Wartezeit einen neuem Sendeversuch**: Nach Beendigung einer Übertragung muss eine teilnehmerspezifische Wartezeit eingehalten werden. Teilnehmer mit niedriger Priorität haben eine längere Wartezeit.

**Anwendungen:**

- CAN, KNX, Wireless-LAN

### 1.7.4 Master / Slave-Verfahren

Im Netz gibt es in diesem System zwei Klassen von Stationen: **einen Master** und **mehrere Slaves**. Der Master tauscht zyklisch mit jedem Slave Daten (z.B. I/O Daten bei Feldbussen) aus. Die Slaves werden der Reihe nach bearbeitet und senden nur nach Aufforderung durch den Master.

**Eigenschaften:**

- einfaches flexibles Verfahren
- Reihenfolge und Häufigkeit der Abfrage vom Master frei wählbar - Priorisierung von Slaves
- Slave zu Slave Datenaustausch muss über Slave geführt werden
- Bei Ausfall des Masters kommt es zu einem kompletten Systemausfall

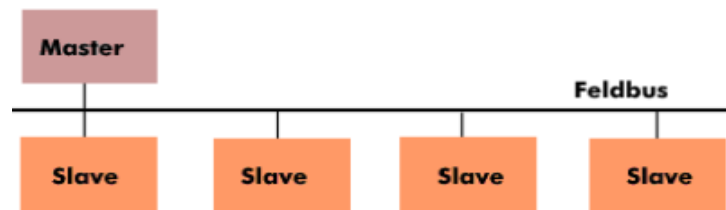


Abbildung 11 Master-Slave Bussystem

**Anwendungen:**

- Profibus, Modbus, ASi

### 1.7.5 TDMA-Verfahren

TDMA ist ein Zeitmultiplexverfahren bei dem die einzelnen Stationen ihre Daten nacheinander in vordefinierten Zeit-Slots übertragen. Jeder Teilnehmer bekommt innerhalb einer Periode (TDMA-Zyklus) einen oder mehrere Zeitschlitze bestimmter Länge.

**Eigenschaften:**

- konstante Zykluszeit
- geringer Protokoll-Overhead
- zeitliche Synchronisierung der Teilnehmer notwendig
- wenig flexibel, keine dynamische Anpassung

**Anwendungen:**

- Profinet IRT (Isochronous Real-Time)

### 1.7.6 Token Passing-Verfahren

Bei mehreren Teilnehmern mit Masterfunktion an einem Bus wird das Zugriffsrecht mittels eines speziellen Telegramms, dem **Token**, reihum weitergegeben. Nur die Station, die den Token hält, darf senden.



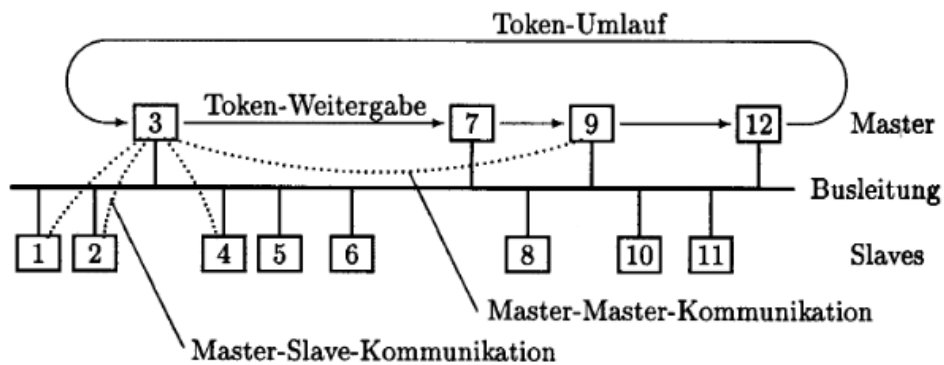


Abbildung 12 Token-Passing und Master-Slave

### Eigenschaften:

- deterministisch / echtzeitfähig - man kann auch Prioritäten vergeben
- komplizierte Technik (Token-Überwachung, Entfernen/Einfügen von Stationen)
- Token-Ring war sehr verbreitet, wird in der Regel heute nicht mehr eingesetzt
- Datenrate 4 bzw. 16 Mbit/s

### Anwendungen:

- Profibus-DP in Verbindung mit Polling der Slaves.

## 1.8 Strukturelemente von Netzwerken

- Repeater, Hub
- Bridge, Switch
- Router
- Gateway

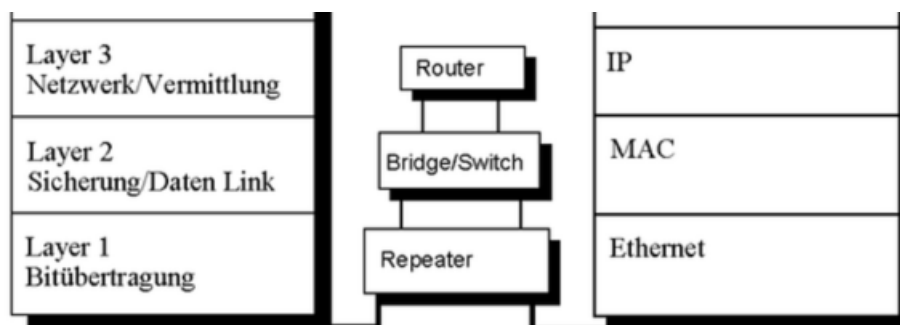
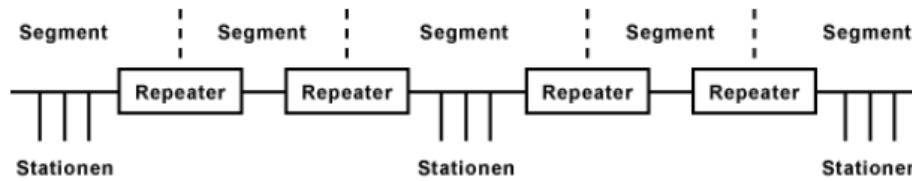


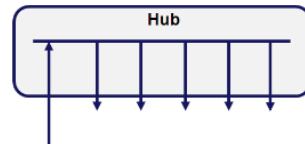
Abbildung 13 OSI-Arbeitsebenen von Repeater, Switch, Router

### 1.8.1 Repeater, Hub

Die Aufgabe eines Repeaters ist die **Regenerierung und Verstärkung von Signalen** und die **Kopplung unterschiedlicher Übertragungsmedien** (Glasfaser/Kupfer) **in Netzwerken**. Ein Repeater verlängert die räumliche Ausdehnung eines Netzwerks. Repeater arbeitet auf **Schicht 1** des ISO/OSI - Modells.

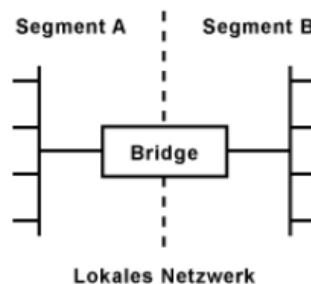


Ein Repeater mit mehreren Ports wird auch als **Hub (Multiport-Repeater)**. Er kann mehrere Netzwerk-Segmente miteinander verbinden.

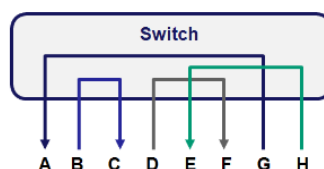


### 1.8.2 Bridge, Switch

Eine **Bridge** ist ein Kopplungselement, das ein lokales Netzwerk in zwei Segmente aufteilt. Eine Bridge arbeitet auf der **Schicht 2** des OSI-Schichtenmodells. Störungen, Kollisionen, fehlerhafte Pakete und der Datenverkehr bleiben innerhalb des Segments und belasten nicht das benachbarte Segment. Eine Bridge teilt ein Netz in **getrennte Kollisionsdomänen**.

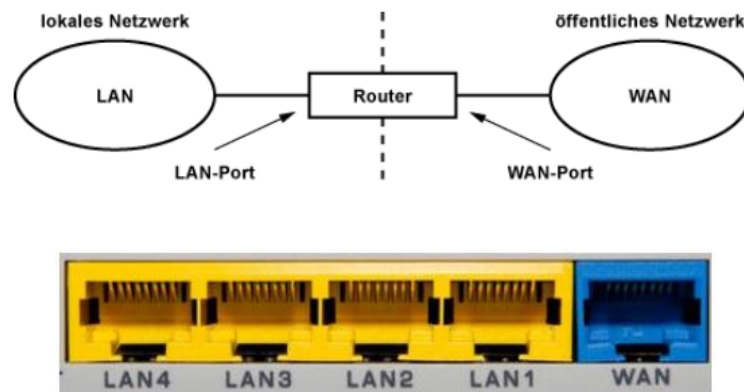


Ein **Switch** ist ein Kopplungselement, das mehrere Teilnehmer (Hosts) in einem lokalen Netzwerk miteinander verbindet. Er arbeitet wie eine Bridge auf **Schicht 2** und besitzt **mehrere Netzwerkanschlüsse**. Ein Switch ist im Prinzip nichts anderes als ein **intelligenter Hub**, der sich merkt, über welchen Port welcher Host erreichbar ist. Auf diese Weise erzeugt jeder Switch-Port eine eigene Kollisionsdomäne.



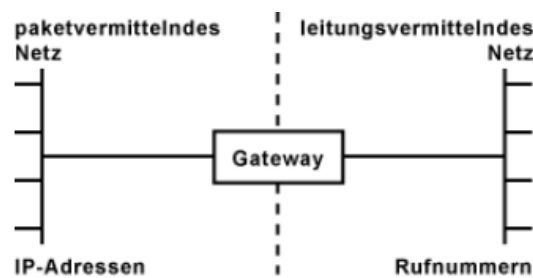
### 1.8.3 Router

Die Aufgabe eines Routers ist die **optimalen Wegsuche** in verzweigten Netzwerken. Wenn in einem Netzwerk mehrere Wege vom Sender zum Empfänger zur Verfügung stehen, ermittelt der Router den optimalen Übertragungsweg. Ein Router besitzt mindestens zwei Netzwerkanschlüsse. Router arbeiten auf der Basis der **Schicht 3** des ISO/OSI - Modells.



### 1.8.4 Gateway

Die Aufgabe des Gateways besteht in der **Verbindung von Netzwerken**, die **physikalisch** zu-  
einander **inkompatibel** sind und/oder eine **unterschiedliche Adressierung** verwenden. Gateways  
transformieren die entsprechenden Kommunikationsfunktionen, teilweise mit erheblichem  
Aufwand. Klassisches Beispiel ist der **ISDN-Router**, der zwischen dem LAN und dem öffentlichen  
Telefonnetz (ISDN) verbinden kann. Gateways arbeiten auf der **Schicht 7** des Referenzmodells.



## 2 Serielle Schnittstellen

Die serielle Schnittstelle ist in verschiedenen Formen weit verbreitet. Da die Datenübertragung bitseriell erfolgt, ist der **Kostenaufwand für das Kabel und die Verdrahtung relativ niedrig**. Der Aufbau ist einfach und es können **größere Entfernungen** überbrückt werden.

Wenn ohne nähere Kennzeichnung von einer „seriellen Schnittstelle“ gesprochen wird, ist damit fast immer die **V.24** oder **RS-232**-Schnittstelle gemeint. Moderne serielle Schnittstellen sind z.B. RS-485, USB, Ethernet, CAN. Allerdings werden diese landläufig nicht als serielle Schnittstellen bezeichnet.

Beispiele serieller Schnittstellen nach Verwendungszweck:

- Ursprüngliche serielle Schnittstellen-Standards:
  - **RS-232** – 1962 eingeführte serielle Schnittstelle
  - **RS-422** – differentielle Transmitter und Receiver
  - **RS-485** – differentielle Transceiver (abschaltbar) und Receiver
- Schnittstellen auf Chip-Ebene:
  - **I<sup>2</sup>C-Bus** – mit Daten- und Taktsignal z. B. bei Telefonkarten
  - **SPI** - Serial Peripheral Interface mit Daten- und Taktsignal z. B. Serial-Flash-Speicher
  - **One-Wire** – Bussystem mit nur einer Datenleitung
- Schnittstellen innerhalb eines Computers
  - **Serial ATA** – für Verbindung Prozessor mit PC-Festplatte
  - **PCI-Express** – für Verbindung Prozessor mit Peripheriebaugruppen (Grafikkarten, ..)
- Schnittstellen in der Größenordnung 1 bis 10 Meter:
  - **USB** - Universal Serial Bus
  - **HDMI** - High Definition Multimedia Interface - Digitale Übertragung von Videodaten
  - **Bluetooth** - Funknetzstandard für kleinere Distanzen (Headsets, Handy, ..)
- Bussysteme > 10m:
  - **AS-Interface** - Feldbus zum Anschluss von Aktoren und Sensoren
  - **CAN-Bus** - Controller Area Network - Steuerungstechnik, Fahrzeugtechnik
  - **Ethernet** - Ethernet basierende Bussysteme (Modbus, Profinet, EtherCAT, ..)
  - **WLAN** - Funknetzstandard für mittlere Distanzen

## 2.1 Serielle Schnittstelle RS-232

**RS-232** (Recommended Standard 232) ist eine inzwischen veraltete, bei früheren Computern vorhandene serielle Schnittstelle, der in den frühen 1960er Jahren von dem US-amerikanischen Standardisierungskomitee Electronic Industries Association (EIA) erarbeitet wurde.

Die RS-232-Schnittstelle definiert ein **asynchrones** Übertragungsprotokoll und wird oft auch als **COM-Schnittstelle** oder **V.24 Schnittstelle** bezeichnet. Die Schnittstelle wurde hauptsächlich zum Anschluss von Textterminals an Großrechnern oder zur Punkt-zu Punkt Verbindung zweier Computer über eine **Modemverbindung** eingesetzt.



Abbildung 14 Verbindung zweier PCs mittels Modem über das Telefonnetz

Bis zum Erscheinen des USB war praktisch jeder PC mit mindestens einer RS-232-Schnittstelle ausgestattet.

### Protokoll:

Die Übertragung eines Datenworts beginnt mit einem vorangestellten **Startbit**. Mit der Signalflanke des Startbits synchronisiert der Empfänger seinen internen Bittakt auf die Empfangsdaten. Sowohl der Sender als auch der Empfänger müssen mit der gleichen **Baudrate** programmiert sein. Anschließend folgen 5 bis 8 **Datenbits**, beginnend mit dem niederwertigsten Bit. Nach dem letzten Datenbit kann zur Erkennung von Übertragungsfehlern ein **Paritätsbit** gesendet werden. Das Ende der Übertragung wird mit einem oder zwei **Stopbits** gekennzeichnet.

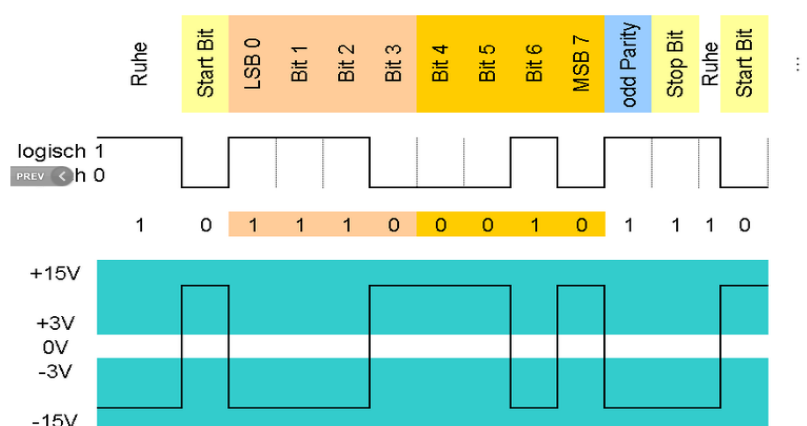


Abbildung 15 RS-232 Zeichenübertragung

### Datenleitungen:

Die RS-232 Schnittstelle ist zur Verbindung von zwei Geräten konzipiert, die beide je eine Datenquelle und eine Datensenke besitzen können. Zur bidirektionalen Datenübertragung werden drei Leitungen benötigt: eine **Sendeleitung** (TxD), eine **Empfangsleitung** (RxD) und eine Masseleitung.

### Handshake:

Zur Vermeidung von Datenverlusten muss der Empfänger die Datenübertragung anhalten können, wenn keine weiteren Daten mehr verarbeitet werden können. Dieser so genannte Handshake kann auf zwei Arten realisiert werden, mit **Software über Steuercodes** oder mit **Hardware über Steuerleitungen**

Beim Software-Handshake sendet der Empfänger zur Steuerung des Datenflusses spezielle Zeichen an den Sender. Beim Xon/Xoff-Protokoll sendet der Empfänger zur Steuerung des Datenflusses spezielle Zeichen an den Sender (**Xon** = 11h und **Xoff** = 13h)

Beim Hardware-Handshake steuert der Empfänger über die Request To Send (**RTS**) - Steuerleitung den Clear To Send (**CTS**) - Handshake Eingang des Senders.

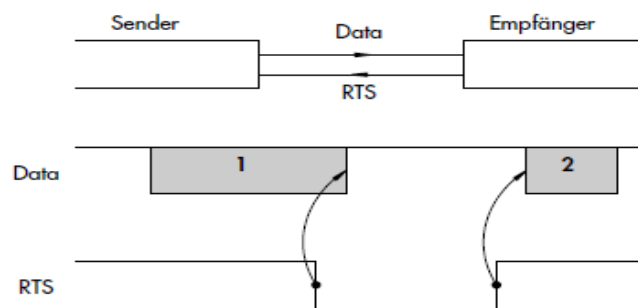


Abbildung 16 RTS fordert eine Unterbrechung der Datenübertragung zwischen Block 1 und 2

### Signalpegel:

Die RS-232 besitzt eine Spannungsschnittstelle. Die Pegel für Daten und Steuerleitungen sind unterschiedlich. Signalpegel zwischen -3V und +3V gelten als undefiniert.

Daten	Steuersignal	Pegel	Spannungsbereich
'0'	'1'	High	+3 bis +15 Volt
'1'	'0'	Low	-3 bis -15 Volt

### Datenraten und Leitungslängen:

Bei der asynchronen Schnittstelle haben sich bestimmte Datenraten historisch entwickelt. Hier wird die Datenrate in der Einheit Bits pro Sekunde (bps) bzw. Baud angegeben. Dabei werden alle Bits (auch Start- und Stoppbit) gezählt und Lücken zwischen den Bytetransfers ignoriert. Typische Datenraten liegen im Bereich **9600 ... 115200 Baud**.

Die maximal zulässige Kabellänge beträgt bei **19200 Baud** maximal **15 Meter**, bzw. ist durch die Kabelkapazität begrenzt. Wegen der hohen Verlustleistung bedingt durch die großen Signalpegel wird auf einen Leitungsabschluss verzichtet. Zur Verminderung von Reflexionen ist die **Flankensteilheit** der Signale lt. Norm **begrenzt**.

**Stecker:**

Die RS-232 Schnittstelle besitzt einen 9-poligen Stecker (früher 25 polig). Beim Verbinden zweier gleichartiger Geräte (Datenübertragungseinrichtungen), müssen die Signalleitungen gekreuzt werden (Tx mit Rx und CTS mit RTS).

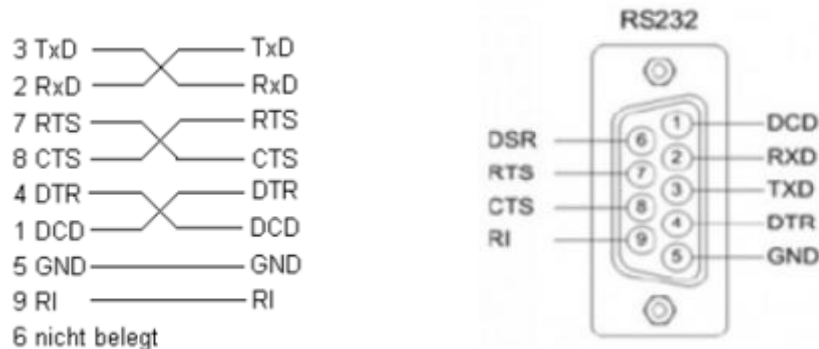


Abbildung 17 RS-232 Steckerbelegung (DB9 Male)

## 2.2 Serielle Schnittstellen RS-422 und RS-485

RS-422 und RS-485 Schnittstellen wurden für die serielle Datenübertragung mit **hoher Geschwindigkeit** über **große Entfernungen** entwickelt. Für beide Schnittstellen sind nur die **elektrischen Eigenschaften** und **kein Protokoll** und **keine Steckerbelegung** spezifiziert.

Die seriellen Daten werden ohne Massebezug als **Spannungsdifferenz** von mindestens  **$\pm 200$  mV** zwischen zwei symmetrischen Leitungen übertragen. Für jedes zu übertragende Signal existiert ein **Aderpaar**, das aus einer **invertierten** und einer **nicht invertierten** Signalleitung besteht. Die invertierte Leitung wird in der Regel durch den Index "**A**" oder "**-**" gekennzeichnet, während die nicht invertierte Leitung mit "**B**" oder "**+**" bezeichnet wird.

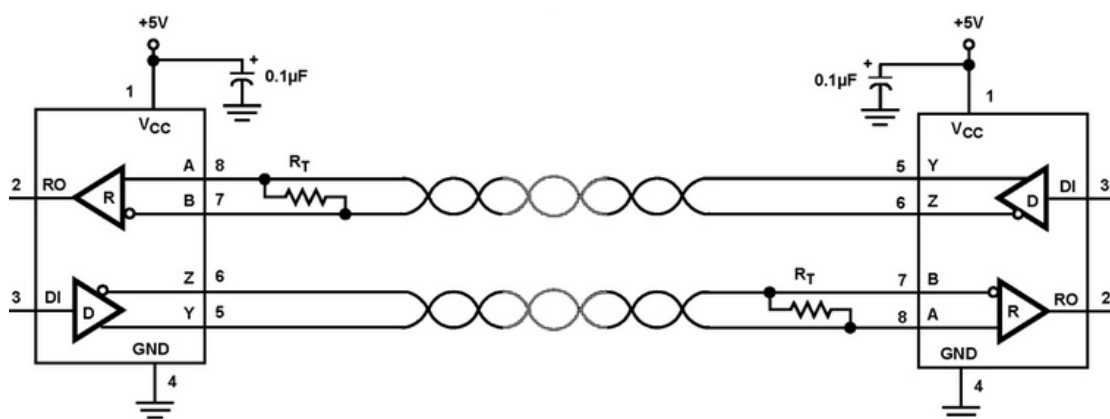


Abbildung 18 Duplex RS-422 Datenverbindung

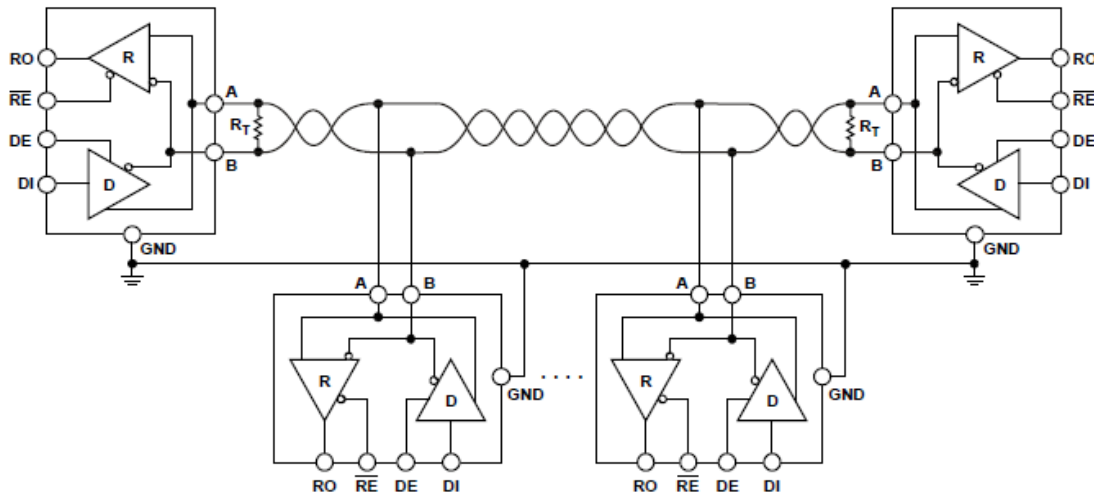


Abbildung 19 Half-Duplex RS-485 Bus

Die Empfänger werten die Differenzspannung zwischen beiden Leitungen aus, so dass **Gleichtakt-Störungen** zu keiner Verfälschung des Nutzsignals führen. Durch die Verwendung von **verdrillten abgeschirmten Leitungen** lassen sich Distanzen von bis zu **1200 Metern** mit einer Datenrate von bis zu **10Mbit/s** realisieren. Die Leitungen müssen **abgeschlossen** werden.

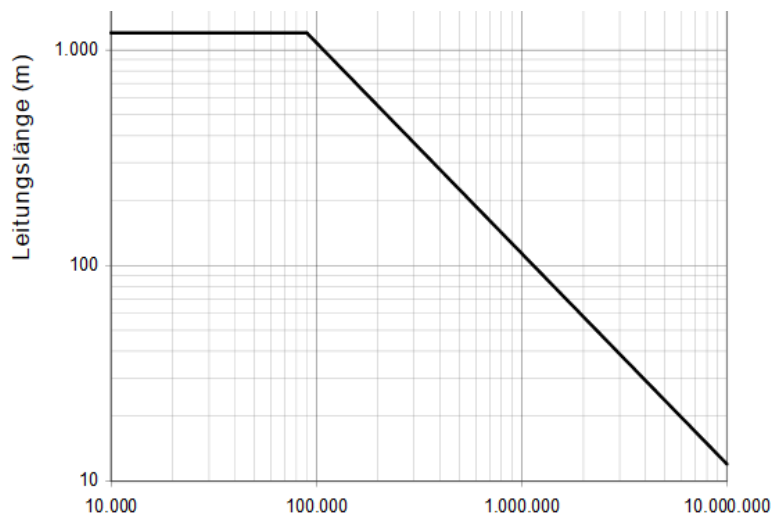


Abbildung 20 Datenrate RS-422 und RS485 in Abhängigkeit von der Leitungslänge

### Unterschiede zwischen RS-422 und RS-485:

An einem **RS-422** Bus können **mehrere Empfänger** und genau **ein Sender** angeschlossen werden. Im Gegensatz dazu können an einem **RS-485** Bus mehrere **gleichwertige Teilnehmer** senden und empfangen. RS-485 Sender besitzen dazu abschaltbare Endstufen. Ein Protokoll stellt sicher, dass jeweils nur ein Sender am Bus aktiv ist. Darüber hinaus gibt es Unterschiede in der elektrischen Spezifikation.



	RS-422	RS-485
Übertragungsart	Differential	Differential
Eingangsempfindlichkeit Empfänger	$\pm 200 \text{ mV}$	$\pm 200 \text{ mV}$
Leitungslänge bis	1200m	1200m
Max. Datenrate	10 Mbit/s	10 Mbit/s
Gleichtakt Eingangsspannungsbereich	$\pm 7 \text{ V}$	-7 V bis +12 V
Eingangswiderstand des Empfängers	4 k $\Omega$	12 k $\Omega$
Anzahl Sender / Empfänger	1 Driver 10 Receivers	32 Drivers 32 Receivers

### Leitungsabschluss bei RS485:

Zur Vermeidung von Reflexionen ist ein Leitungsabschluss an beiden Enden der Leitung erforderlich. Ist kein Sender aktiv, sinkt die Differenzspannung an den Eingängen in einen undefinierten Zustand kleiner  $\pm 200 \text{ mV}$ . Damit der Signalzustand in jedem Fall eindeutig bleibt, verschaltet man Pull-Up und Pull-Down **Failsafe Bias - Widerstände** gegen 5V und Masse.

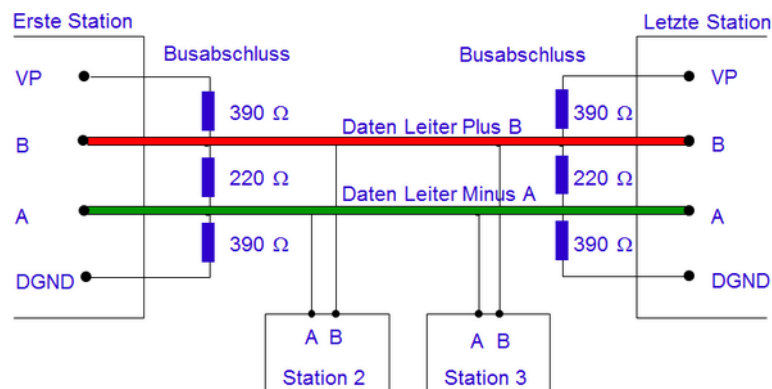


Abbildung 21 Leitungsabschlusspezifikation für RS-485 – Profibus

Die neue aktuelle Generation von RS-485 Bausteinen besitzt sogenannte Failsafe-Empfänger mit einem modifizierten Eingangsspannungsbereich, so dass externe keine Failsafe-Beschaltung mehr erforderlich ist.

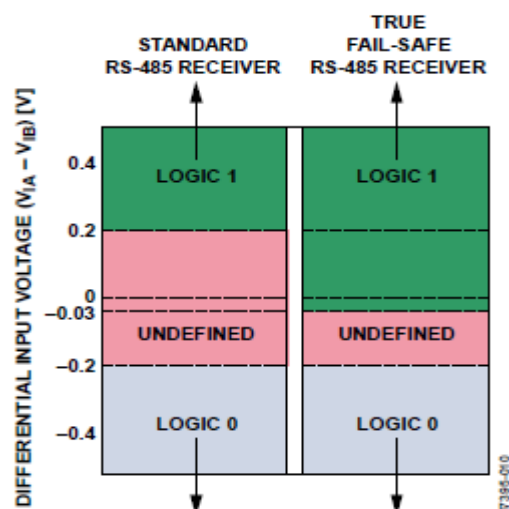


Abbildung 22 Eingangs-Schwellenspannungen von RS-485 Empfängern

### 3 Chip-Bussysteme

Für Datenübertragungen **zwischen Mikroprozessoren** und integrierten **Schaltkreisen** und **Sensoren** werden, wegen der einfachen Anbindung und der damit verbundenen **niedrigen Kosten**, oft **serielle Bussysteme** eingesetzt. Die Datenübertragung erfolgt dabei meist mit geringer Geschwindigkeit und einem definierten Protokoll. Für die Anbindung der Bausteine sind nur **wenige Signalleitungen** erforderlich.

Zu den häufigsten Vertretern zählen:

- **I<sup>2</sup>C-Bus** - Inter-Integrated Circuit Bus (Fa. Phillips)
- **SPI** - Serial Peripheral Interface (Fa. Motorola)
- **1-Wire** - One-Wire mit nur einer Datenleitung (Fa. Dallas Semiconductor)

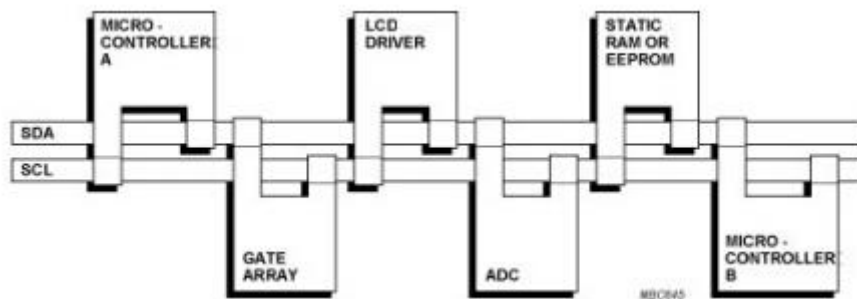


Abbildung 23 I2C Bus Anwendung

#### 3.1 I<sup>2</sup>C-Bus (TWI)

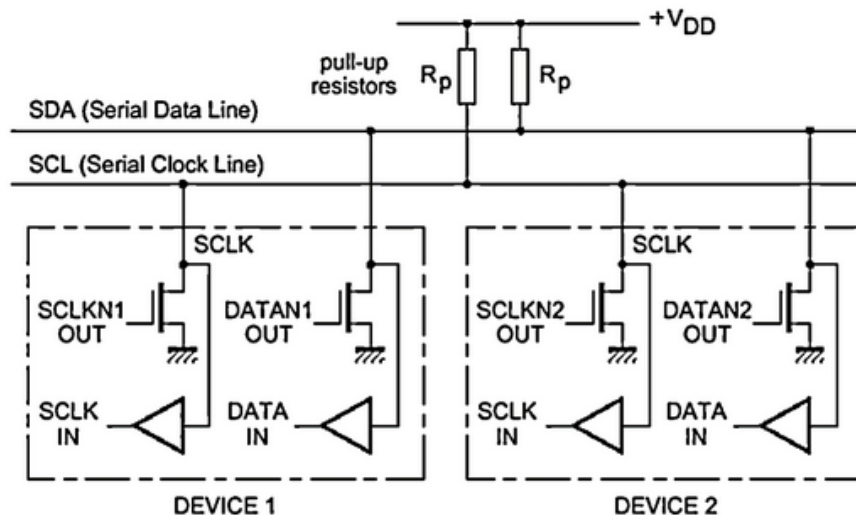
I<sup>2</sup>C steht für **Inter-Integrated Circuit**. Das Bussystem wurde von Fa. **Philips** in den frühen 1980er Jahren entwickelt, um unterschiedliche **Chips in Fernsehgeräten** kostengünstig integrieren zu können. Inzwischen gibt einige tausende verschiedene ICs von über 50 Herstellern.

Eine weitere Bezeichnung für diese Schnittstelle ist **Two-Wire Serial Interface (TWI)**. Viele Hersteller verwenden diese Bezeichnung, da I<sup>2</sup>C eine eingetragene Marke von Philips ist. Technisch sind beide Systeme identisch und können auch miteinander kommunizieren.

##### Aufbau:

I<sup>2</sup>C ist als **Master-Slave-Bus** konzipiert. Es ist aber auch möglich, **mehrere Master** an einen I2C-Bus anzuschließen. Ein Datentransfer wird immer durch einen Master initiiert. Der über eine Adresse angesprochene Slave reagiert auf eine Anforderung. Im Multimaster-Mode können zwei Master-Teilnehmer direkt miteinander kommunizieren, dabei arbeitet ein Master als Slave.

Das Interface besteht aus zwei Signalleitungen **SDA** (Serial Data) und **SCL** (Serial Clock) sowie einer **Masseleitung**. Beide Signalleitungen sind über **Pull-Up-Widerstände** mit der Versorgungsspannung verbunden. Sämtliche an die Signalleitungen angeschlossenen Geräte haben Open-Drain-Ausgänge, was zusammen mit den Pull-Up-Widerständen eine **Wired-AND-Schaltung** ergibt. Die Anzahl der Busteilnehmer wird nur durch die maximale adressierbaren Slaves lt. Protokoll und die Buskapazität begrenzt.

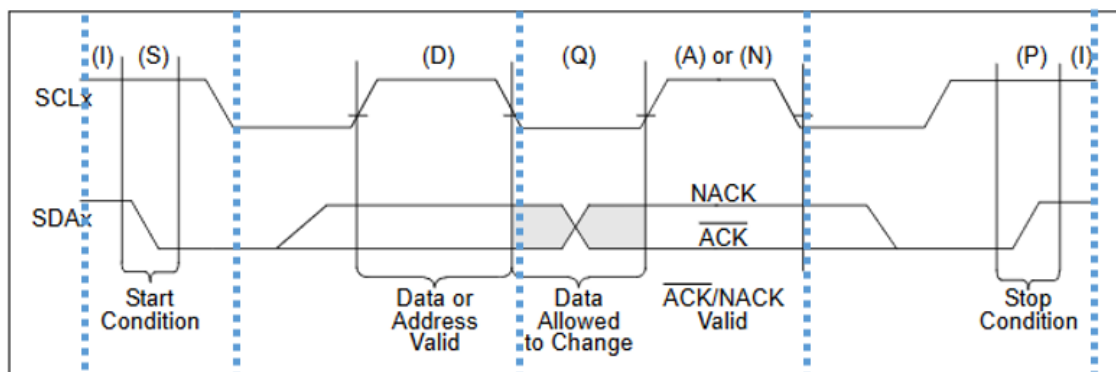
Abbildung 24 I<sup>2</sup>C Bus mit 2 Teilnehmern**Bustakt:**

Der Bustakt wird immer vom Master ausgegeben. Für die verschiedenen Modi ist jeweils ein maximal erlaubter Bustakt vorgegeben. In der Regel können aber auch beliebig langsamere Taktraten verwendet werden, falls diese vom Master-Interface unterstützt werden. Die folgende Tabelle listet die maximal erlaubten Taktraten auf.

Modus	Maximale Übertragungsrate
<b>Standard Mode</b>	<b>100 kbit/s</b>
Fast Mode	400 kbit/s
High Speed Mode	3,4 Mbit/s

**Protokoll:**

Der Beginn einer Übertragung wird mit dem **Startsignal** vom Master angezeigt. Zur Erzeugung eines Startsignals ist eine **fallende Flanke von SDA** während **SCL logisch HIGH** auszuführen.

Abbildung 25 Zeitverhalten am I<sup>2</sup>C Bus

Auf die Startbedingung folgt die Übertragung von Datenworten zu jeweils **8 Datenbits** und einem **ACK-Bestätigungsbit**. Datenbits sind gültig, wenn sich ihr logischer Pegel während einer Clock-High-Phase nicht ändert. Das ACK-Bestätigungsbit wird vom Slave durch einen Low-Pegel auf der Datenleitung während der neunten Takt-High-Phase signalisiert.

Am Ende der Übertragung folgt ein **Stoppsignal**. Für ein Stoppsignal ist eine **steigende Flanke von SDA** während **SCL logisch HIGH** ist herbeizuführen. Laut Spezifikation dürfen Start- und Stoppsignale zu jedem Zeitpunkt eines laufenden Transfers erzeugt werden. Nach dem Stoppsignal wird der **Bus** als **Idle** (leer) bezeichnet.

### Repeated Start Condition:

Hierbei beendet der Master einen Transfer nicht durch eine Stoppbedingung, sondern erzeugt einfach eine neue Startbedingung.

### Clock-Synchronisation (Stretching):

Wenn der Slave mehr Zeit benötigt, als durch den Takt des Masters vorgegeben ist, kann er zwischen der Übertragung einzelner Bytes die Taktleitung **SCL auf logisch LOW halten** und so den Master bremsen. Ab diesem Zeitpunkt fährt der Master nicht einfach fort, sondern liest nun seinerseits die SCL-Leitung, bis diese HIGH wird.

### Multimaster Betrieb:

Der Vorgang der Buszuteilung an einen Master in einem Multimastersystem wird als **Arbitrierung** bezeichnet. Auf dem I<sup>2</sup>C-Protokoll ist ein **CSMA/CA Verfahren** festgelegt. Carrier Sense bedeutet, dass ein transaktionswilliger Master die Leitungen abfragt, um festzustellen, ob der Bus belegt ist oder nicht. Der I<sup>2</sup>C-Bus wird als belegt erkannt, wenn eine von den beiden Leitungen sich im LOW - Zustand befindet.

Hat der Master die Leitung nun als **frei** erkannt, so generiert er eine **Startbedingung** und gibt als erstes die Adresse des gewünschten Slaves heraus. Während der Clock-High Phasen vergleichen die Master nun das jeweils von ihnen ausgegebene SDA-Bit mit dem zurückgelesenen Bit.

Ein Master, der eine 1 ausgibt und eine 0 empfängt (**0 ist dominant**), erkennt daran, dass ein anderes Gerät den Bus bedient, und bricht die Übertragung ab. Dieser Vorgang wird als Kollisionsvermeidung (**Collision Avoidance**) bezeichnet. Der siegreiche Master bekommt vom Zugriffsversuch (der Kollision) des Konkurrenten nichts mit.

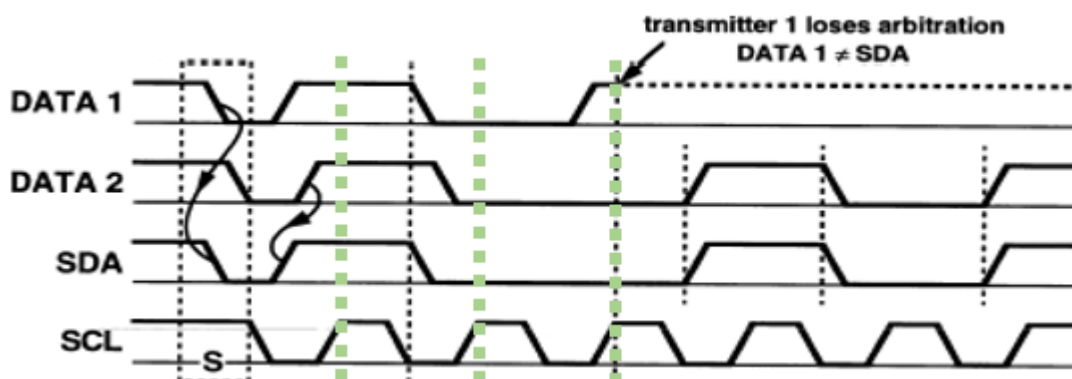


Abbildung 26 I<sup>2</sup>C Multimasterbetrieb

## Adressierung:

Eine **Standard-I<sup>2</sup>C-Adresse** ist das **erste vom Master gesendete Byte**, wobei die ersten sieben Bit die eigentliche Adresse darstellen und das achte Bit (**R/W-Bit**) dem Slave mitteilt, ob er Daten vom Master empfangen soll (LOW) oder Daten an den Master zu übertragen hat (HIGH). I<sup>2</sup>C nutzt daher einen Adressraum von **7 Bit**. 16 der 128 möglichen Adressen sind für Sonderzwecke reserviert.

Adresse	R/W-Bit	Beschreibung
0000000	0	General Call Adresse
0000001	X	CBUS Adresse
0000010	X	Reserviert für ein anderes Busformat
0000011	X	Für zukünftige Erweiterungen reserviert
00001XX	X	Für zukünftige Erweiterungen reserviert
11111XX	X	Für zukünftige Erweiterungen reserviert
<b>11110XX</b>	<b>X</b>	<b>10-Bit Adressierung</b>

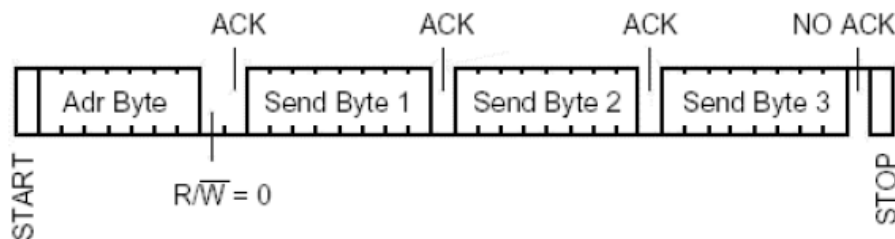


Abbildung 27 I<sup>2</sup>C Datenformat

Wegen Adressknappheit wurde später eine 10-Bit-Adressierung eingeführt. Sie ist abwärtskompatibel zum 7-Bit-Standard durch Nutzung von 4 der 16 reservierten Adressen. Beide Adressierungsarten sind gleichzeitig verwendbar, was bis zu 1136 Knoten auf einem Bus erlaubt.

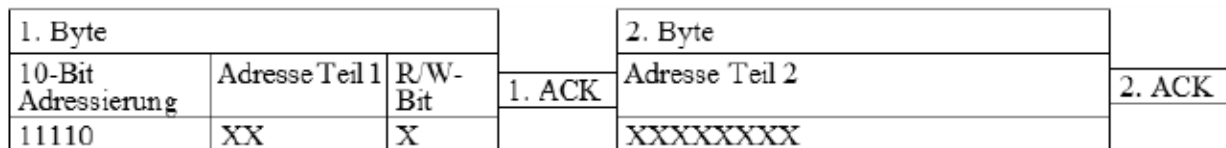


Abbildung 28 I<sup>2</sup>C Bus 10 Bit Adressierung

Jedes I<sup>2</sup>C-fähige Bauteil hat eine vom Hersteller festgelegte Adresse, von der bisweilen die untersten drei Bits, **Subadresse** genannt, über drei Steuerpins festgelegt werden können. In diesem Fall können bis zu **acht gleichartige Bausteine** an einem I<sup>2</sup>C-Bus betrieben werden.

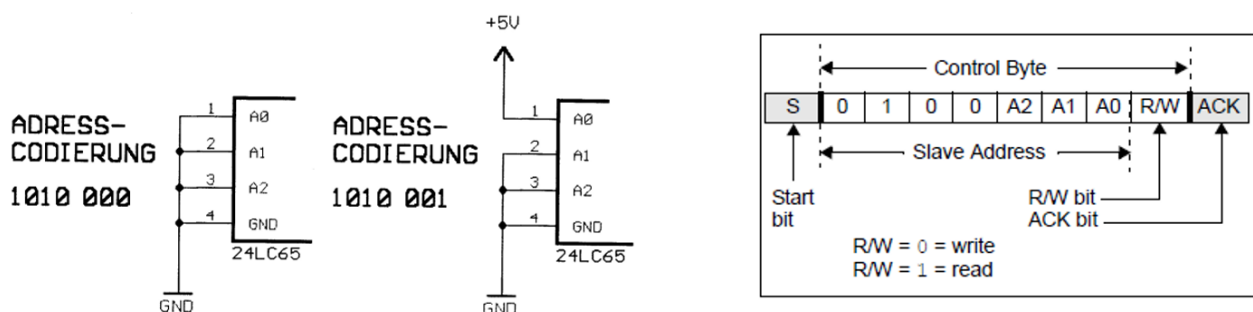


Abbildung 29 I<sup>2</sup>C Bus Subadresseinstellung

**Verwendung:**

Das I<sup>2</sup>C-Bus ist **einfach**, aber **störanfällig**. Diese Tatsache schränkt die Verwendung auf störrarme Umgebungen ein, wo weder mit Übersprechen, Rauschen, EMV-Problemen zu rechnen ist. Auch ist er **ungeeignet zur Überbrückung größerer Entfernungen**, wie es beispielsweise für Feldbusse typisch ist.

Beispiele für I<sup>2</sup>C **Peripheriebausteine** mit sind Port-Expander und AD/DA-Wandler mit **niedriger Abtaste**, Echtzeituhren, nichtflüchtige Speicher (EEPROM), elektronische Sensoren für Temperatur, Druck, Beschleunigung mit integriertem AD-Wandler.

**Anwendungsschicht** (Schicht 7 nach ISO-OSI Modell):

Beispiel der Arduino-IDE: <https://www.arduino.cc/en/Reference/Wire>

- [begin\(\)](#) - Initialisierung der Bibliothek
- [beginTransaction\(\)](#) - Start einer Schreib-Übertragung
- [write\(\)](#) - Bytes zum Schreiben übergeben
- [endTransmission\(\)](#) - Übertragen der zum Schreiben übergebenen Bytes
- [requestFrom\(\)](#) - Anzahl Bytes vom Slave anfordern
- [available\(\)](#) - Anzahl empfangener Bytes
- [read\(\)](#) - Lesen empfangener Bytes

Beispiel Schreiben an Slave:

```
Wire.beginTransaction(device_address); // i2C address + R/W Bit
Wire.write(a); // select register
Wire.write(b); // value to write
Wire.endTransmission(); // start transmission
```

Beispiel Lesen von Slave:

```
Wire.beginTransaction(device_address); // i2C address + R/W Bit
Wire.write(a); // select register
Wire.endTransmission(); // start transmission
Wire.requestFrom(device_address, 2); // receive data
b1 = Wire.read(); // read data from receive buffer
b2 = Wire.read(); // read data from receive buffer
```

### 3.2 SPI - Serial Peripheral Interface

Das **Serial Peripheral Interface (SPI)** ist ein von Fa. **Motorola** entwickeltes Bussystem mit einem sehr lockeren Standard für einen synchronen seriellen Datenbus, zur Verbindung digitaler Schaltkreise auf einer Computerplatine. Ein ähnliches Bussystem existiert von National Semiconductor und nennt sich **Microwire**.

#### Aufbau:

Der Bus besteht aus **zwei Daten- und einer Taktleitung**, an denen alle Teilnehmer parallel angeschlossen sind. Darüber hinaus ist zum Ansteuern der Slaves jeweils eine eigene **Slave Select** Leitung erforderlich.

- **MOSI** (Master out Slave in) oder **SDO** (Serial Data Out)
- **MISO** (Master in Slave out) oder **SDI** (Serial Data In)
- **SCKL** (Serial Clock) - Schiebetakt
- **SS** (Slave Select) oder **CS** (Chip Select)

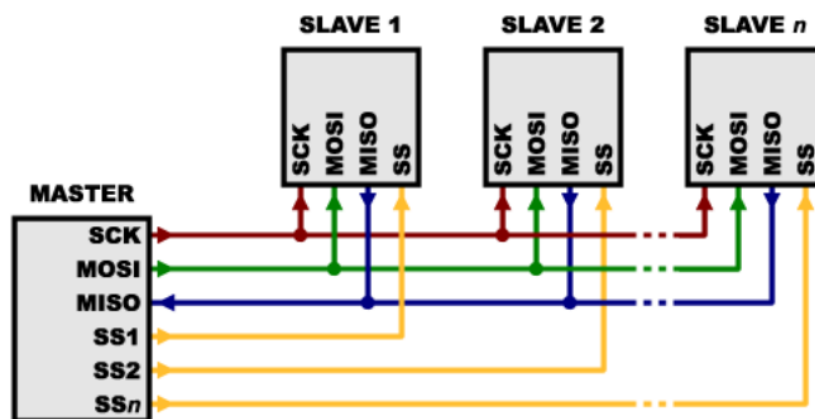


Abbildung 30 SPI-Bus

#### Adressierung:

Mit SS (Slave-Select) kann ein Peripheriebaustein selektiert werden. Im **nicht selektierten Zustand** sind die **SDO-Leitungen hochohmig** und damit vom Datenbus abgekoppelt. Der Master kann somit bestimmen mit welchem Peripheriegerät er kommunizieren möchte. Die Taktleitung SCLK dient wird dem Slave unabhängig vom Selektionszustand zugeführt.

#### Bustakt:

Die Spezifikation des SPI gibt **keine maximale Taktfrequenz** an. Die mögliche Taktfrequenz hängt von den einzelnen Slave-Bausteinen ab und ist dem jeweiligen Datenblatt zu entnehmen. Typische Werte liegen im Bereich **1 - 10Mhz**.

#### Protokoll:

Der Buszugriffsverfahren am SPI-Bus basiert auf dem **Master/Slave-Verfahren**. Die Datenübertragung erfolgt **synchron** zu einem Taktsignal, welches vom SPI-Master vorgegeben wird.

Der Master startet die Datenübertragung, indem er einen Slave über seine **Slave Select** Leitung selektiert. Anschließend gibt der Master auf der Taktleitung je nach Telegramm eine **festgelegte Anzahl von Takten** aus. Mit jedem Takt wird vom Master über SDO ein Bit ausgegeben und gleichzeitig vom Slave über SDI eingelesen.

Umgekehrt empfängt der Master mit jedem Takt über ein Bit vom Slave. Die Übertragung ist eher mit dem Begriff "**Austausch von Bits**" als durch Senden und Empfangen zu beschreiben.

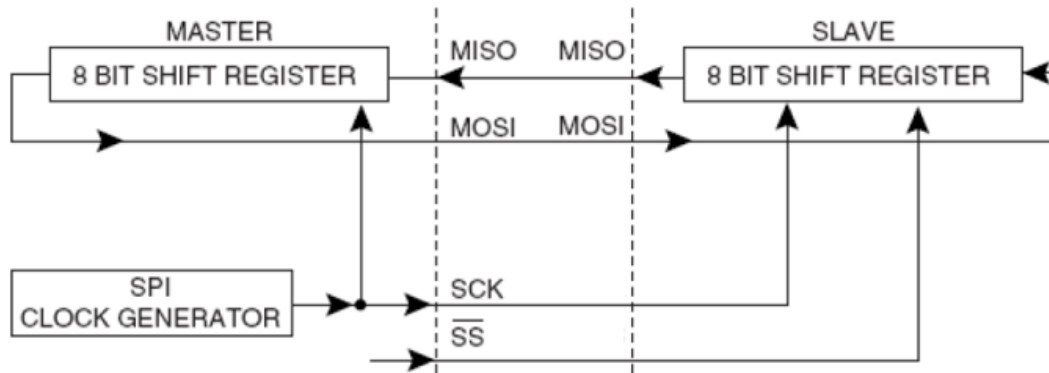


Abbildung 31 SPI-Schiebregister in Master und Slave

### Betriebsarten:

Für die Synchronisierung der Daten mit dem Taktsignal haben sich **vier Betriebsarten** durchgesetzt. Die beiden Parameter **Clock Polarität** (CPOL) und **Clock Phase** (CPHA) spezifizieren eine von vier möglichen Betriebsarten.

### Clock Polarität (CPOL):

- 0: Takt ist in Ruhe LOW, ein Wechsel auf HIGH zählt als steigende Taktflanke
- 1: Takt ist invertiert: in Ruhe HIGH, ein Wechsel auf LOW zählt als steigende Taktflanke

### Clock Phase (CPHA):

- 0: Daten werden bei steigender Taktflanke eingelesen, bei fallender ausgegeben
- 1: Daten werden bei fallender Taktflanke eingelesen, bei steigender ausgegeben

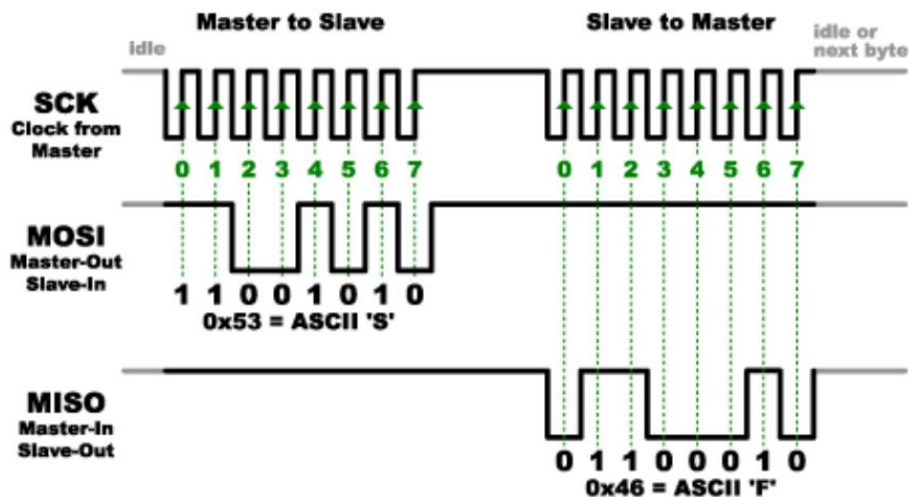


Abbildung 32 SPI-Datenübertragung mit CPOL = 0, CPHA = 0



**Verwendung:**

Der SP-Bus wird zum Anschluss von Peripheriebausteinen an Mikroprozessoren auf Leiterplatten mit **kurzen Leitungslängen** eingesetzt. Beispiele sind **schnelle** AD/DA-Wandler, Sensoren, UARTs, CAN- und Ethernet-Controller.

**Anwendungsschicht** (Schicht 7 nach ISO-OSI Modell):

Beispiel der Arduino-IDE: <https://www.arduino.cc/en/Reference/SPI>

- `begin()` - enables the SPI bus; reserves the SCK, MOSI and MISO pins for the SPI system
- `beginTransaction()` - initialise SPI Settings (Clock, Mode, ...)
- `transfer()` – ein Bytes transferieren (schreiben / lesen)
- `endTransaction()`
- `end()` - disables the SPI bus

Programmauszug: 2 Bytes vom Slave lesen:

```
...
// set up the speed, data order and data mode
SPISettings settings (2000000, MSBFIRST, SPI_MODE1);
pinMode (slaveCS, OUTPUT); // set the Slave Select Pins as outputs
SPI.begin();
...
// read two bytes from device A
SPI.beginTransaction (settings);
digitalWrite (slaveCS, LOW);
val1 = SPI.transfer (0); // reading only, so data sent does not matter
val2 = SPI.transfer (0);
digitalWrite (slaveCS, HIGH);
SPI.endTransaction ();
...
```

### 3.3 1-Wire Bussystem

Das 1-Wire Bussystem kommt nur mit **einer Datenleitung** aus. Gegen das Ende der 80er Jahre begann man bei Fa. **Dallas Semiconductor** (heute Fa. **Maxim**) mit der Entwicklung des 1-Wire Bussystems, zur einfachen kostengünstigen Anschaltung von Peripheriebausteinen an Mikroprozessoren. Entstanden ist ein Bussystem, welche wegen seiner Eigenschaften aktuell auch vermehrt für größere Distanzen als ursprünglich gedacht, eingesetzt wird.

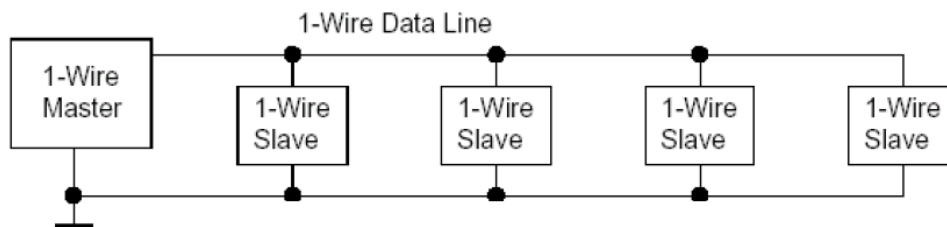


Abbildung 33 -Wire-Bussystem

#### Elektrische Eigenschaften:

Das 1-Wire Protokoll nutzt konventionelle **CMOS/TTL Logiklevel** bei 5V Versorgungsspannung. Die Datenleitung liegt mit einem Pull-Up-Widerstand an der Versorgungsspannung. Alle Teilnehmer haben **Open-Drain-Ausgänge**, was zusammen mit den Pull-Up Widerständen wie beim I2C-Bus eine **Wired-AND-Schaltung** ergibt.

Einige 1-Wire Sensoren mit geringem Stromverbrauch können in der so genannten **parasitären Betriebsart über die Datenleitung versorgt** werden. Ein im Sensor integrierter Kondensator überbrückt die Low-Phasen auf der Datenleitung.

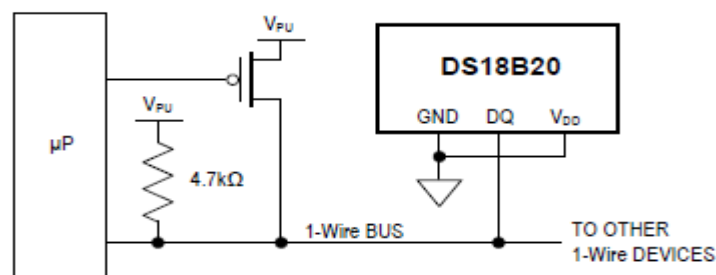


Abbildung 34 Parasitäre Betriebsart

#### Datenrate:

Im regulären Betriebsmodus sind mit den nachfolgend beschriebenen Timing-Bedingungen Datenraten von bis zu **16,3 kbps** möglich.

#### Topologie:

1-Wire Teilnehmer können in Linien-, Baum-, oder Stern-Topologie installiert werden. Dabei sind je nach Busmaster, wegen der geringen Datenrate Leitungslängen bis zu **100m** mit ca. 20 .. 100 Teilnehmern möglich. Die Datenübertragung bei einem längeren Bussystem erfolgt über eine Twisted-Pair Zweidrahtleitung.

### Protokoll:

Vom Prinzip her handelt es sich beim 1-Wire Netzwerk, um eine Master/Slave Architektur. Da der 1-Wire-Bus keine separate Leitung für das Taktsignal besitzt, muss ein **vorgegebenes Timing** eingehalten werden. Die Synchronisation erfolgt bei jedem Bit mit einem vom Master vorgegebenen **Time-Slot**.

Der **Datentransfer** über den 1-Wire-Bus läuft danach immer in folgenden **drei Schritten** ab:

1. **Reset:** Der Master sendet einen Reset und startet damit eine neue Transaktion
2. **ROM-Kommando:** Der Master sendet ein ROM-Kommando an einen oder alle Slaves
3. **Funktionskommando:** Der Master sendet einen spezifischen Befehl an einen Slave

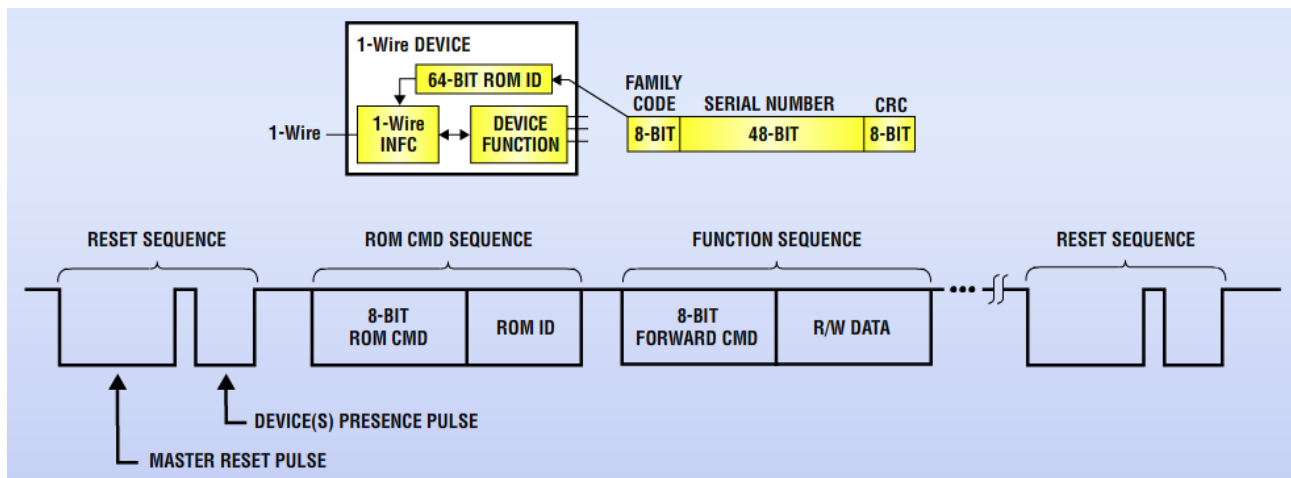


Abbildung 35 Ablauf einer 1-Wire Transaktion

Während die Schritte 1 und 2 (Reset und ROM-Kommandos) bei allen 1-Wire-Devices gleich sind, beinhaltet das Funktionskommando bauteilspezifische Befehle.

### Reset:

Für einen Reset sendet der Master einen Low-Pegel mit einer Dauer von **480µs**. Jeder Slave zeigt mit einem **Presence-Pulse** seine Anwesenheit, indem er danach den Bus für mindestens 60µs auf Low zieht.

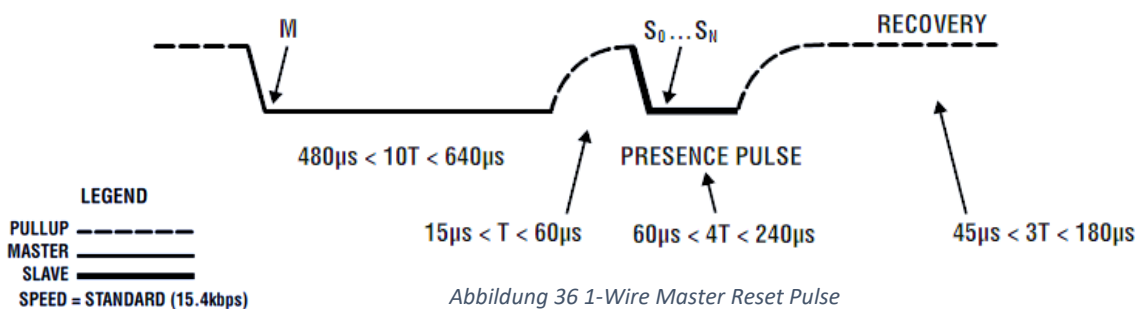
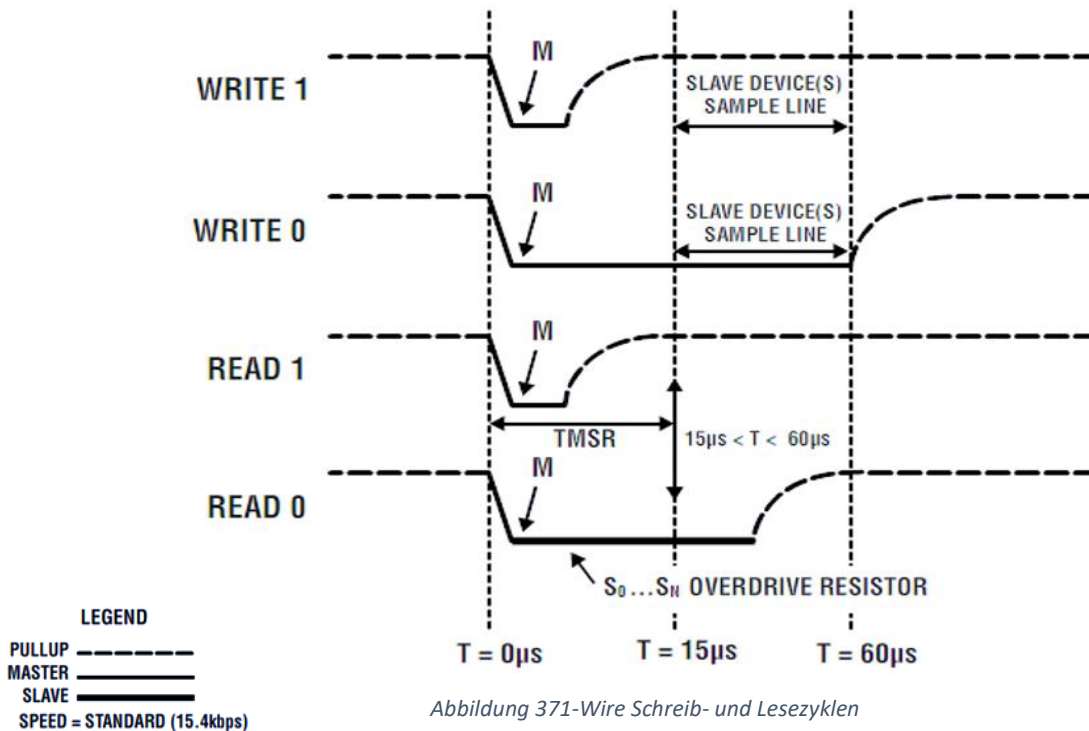


Abbildung 36 1-Wire Master Reset Pulse

### Schreib- und Lesezyklus:

Zum **Schreiben** einer logische 1 bzw. 0, wird die Datenleitung vom Master für 1-15 bzw. 60-120µs auf Low-Pegel gezogen. Zum **Lesen** zieht der Master wie beim Schreiben einer logischen 1 den Bus für 1-15µs auf Low-Pegel und der Slave hält für die Übertragung einer logischen 0 den Bus darüber hinaus auf Low-Pegel. Der Master liest das Signal nach frühestens 15µs vom Bus. Die Reihenfolge der Schreib- oder Lesezyklen ergibt sich aus dem vorangegangenen ROM-Kommando.



### Adressierung:

Alle Teilnehmer haben Ihre **Adressen** (8 Byte Rom-ID) bereits **ab Werk programmiert**. Die Adressen sind **weltweit eindeutig vergeben** und es sind keine Adresseinstellungen erforderlich. Zur Adressierung der Busteilnehmer im Betrieb muss der Master bei jedem Systemstart eine Liste mit den vorhandenen Busteilnehmern und deren eindeutigen Adressen erstellen. Dazu existiert ein vordefiniertes frei verfügbares Protokoll.

Nachteil der weltweit eindeutigen Adressierung: Für die Geräteentwicklung ist immer ein **Schnittstellenbaustein** der Fa. Dallas Semiconductor erforderlich.

### Verwendung:

Der 1-Wire-Bus eignet sich besonders zur Anbindung von **langsamer Sensorik auch über größere Entfernungen** z.B. in der Gebäudeautomation (Temperaturmesser und -logger, Akkuüberwachung, Fensterkontakte, Rauchmelder).

1-Wire Slaves werden wegen ihrer weltweit eindeutigen und nicht veränderlichen 64Bit-Seriennummern auch im Bereich der **Zutrittskontrolle** eingesetzt. Unter dem Markennamen **iButton** werden von Fa. Dallas Semiconductor 1-Wire Teilnehmer nur mit Seriennummer ohne weitere Funktion vertrieben.



Abbildung 38 Zutrittskontrolle mit iButton

## 1-Wire Tutorial Fa. Maxim:

<https://www.maximintegrated.com/en/products/1-wire/flash/overview/index.cfm>

### 3.4 Vergleich I2C, SPI, 1Wire

	I2C	SPI	1WIRE
Übertragungsart	synchron serielle Datenübertragung		
Buszugriffsart	Master-Slave Multimasterbetrieb	Master-Slave	Master-Slave
Datenraten	100 kbit/s (Standard)	1-10Mhz nicht spezifiziert	16 kbit/s
Buslängen	Platinen-Abmessungen	Platinen-Abmessungen	bis zu 100m
Hardware	Open-Drain Daten- und Taktleitung	TTL / CMOS Leitungen	Open-Drain Datenleitungen
Datenleitungen	1x (SDA)	2x (MOSI, MISO) verbundenes Sende- und Empfangsschiebe- register	1x Energieversorgung über Datenleitung möglich
Taktleitung	1x (SCL)	1x (SCK)	Taktrückgewinnung über Daten-Timing
Adressierung	7 Bit über Protokoll, Subadresse über Adress-Pins am	Jeder Slave besitzt einen Slave-Select Pin	64Bit-ID Teilnehmer haben weltweit eindeutige ID
Slave-Anzahl	< 127 Teilnehmer, Begrenzung durch Buslast	begrenzt durch Anzahl der Chip-Select Leitungen	20..100 Teilnehmer, Begrenzung durch Buslast
Protokoll	spezifiziert	sehr "lose" spezifiziert	spezifiziert
Anwendungen	Chip-Anschaltungen	Chip-Anschaltungen	Chip-Anschaltungen, Sensoren in der Hautechnik

## 4 Bussysteme

### 4.1 HART-Protokoll

Das HART- oder **Highway Adressable Remote Transducer** - Protokoll ist ein globaler Standard, mit dem intelligente Feldgeräte und Überwachungssysteme **digitale Informationen über eine analoge Verdrahtung** austauschen. Highway steht für adressierbaren Fernzugriff auf Messwertgeber. Das HART-Protokoll ist kein Feldbussystem im eigentlichen Sinn. Es stellt eine **Übergangsstufe** zwischen einer **analogen und digitalen Signalübertragung** dar.

Die HART-Technologie entstand Ende der 1980-er Jahre, ist in der Anwendung einfach und extrem zuverlässig und in der **Prozessindustrie** eines der am häufigsten eingesetzten Datenprotokolle. Seit 2007 ist die HART-Technologie Teil der Feldbus-Norm IEC 61158.

Merkmale:

- praxiserprobt, einfach im Aufbau, Wartung und Anwendung
- kompatibel zu konventioneller Analoggeräte-Instrumentierung (4-20mA)
- gleichzeitige analoge und digitale Kommunikation
- Punkt-zu-Punkt- oder Multiplex-Betrieb möglich
- Datenzugriff über bis zu zwei Bediengeräte (primärer und sekundärer Master)
- Reaktionszeit von ca. 500 ms

**Aufbau:**

Der Buszugriff erfolgt über ein **Master-Slave** Protokoll. Das HART-Protokoll wird meist als **Punkt zu Punkt Verbindung** aufgebaut. Mit Hilfe eines **Multiplexers** werden Punkt zu Punkt Verbindungen zu mehreren Teilnehmern hergestellt. Die analoge Signalübertragung bleibt davon unbeeinflusst. Weiters können nicht intelligente 4-20mA Feldgeräten an den Bus angeschlossen werden.

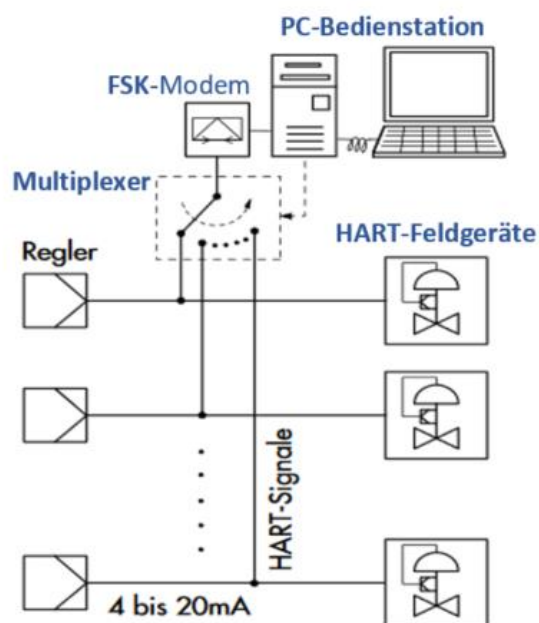


Abbildung 39 HART Kommunikation über Multiplexer

### Datenübertragung:

Das HART-Protokoll verwendet den FSK-Standard (**Frequenzumtastung**), um ein digitales Signal auf ein analoges Messwertsignal von **4-20 mA** zu überlagern. Dies ermöglicht so die Übertragung zusätzlicher digitaler **Parameter- und Überwachungsdaten** von oder zu einem intelligenten Feldgerät (**Smart Transmitter**).

Während bei Feldgeräten und kompakten Handgeräten das **FSK Modem** im Gerät integriert ist, wird es bei PC-Bedienstationen über eine serielle Schnittstelle extern angeschlossen.

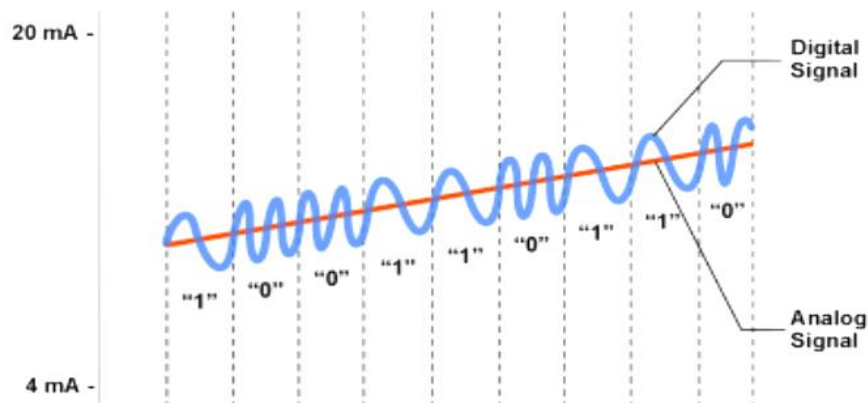


Abbildung 40 HART Frequenzumtastung

Eine **logische 0** wird durch ein sinusförmiges **2200 Hz** – Signal und eine **logische 1** durch ein sinusförmiges **1200 Hz** – Signal dargestellt.

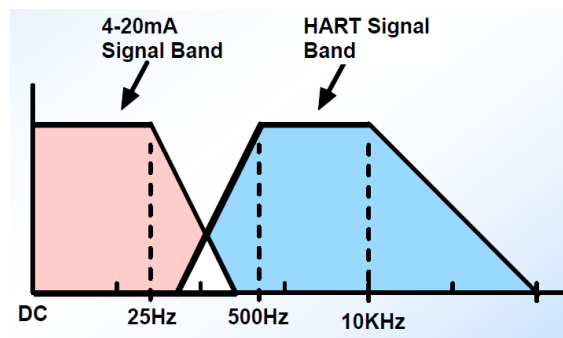


Abbildung 41 HART Frequenzbänder

### Datenraten und Leitungslängen:

Die Bitübertragungsrate beträgt **1200 Bit/sec**. Die Übertragungsrate für einfache Messwerte beträgt **ca. 2 Werte / sec**. Mit geschirmten verdrehten Zweidrahtleitungen sind Leitungslängen bis **3km** möglich.

### Smart Transmitter:

Ein Smart Transmitter ist ein **kommunikationsfähiger Messumformer** mit parametrierbarem Messbereich. Smart Transmitter wandeln nichtelektrische physikalische Größen in ein elektrisches Signal, meist in **4..20 mA** um.

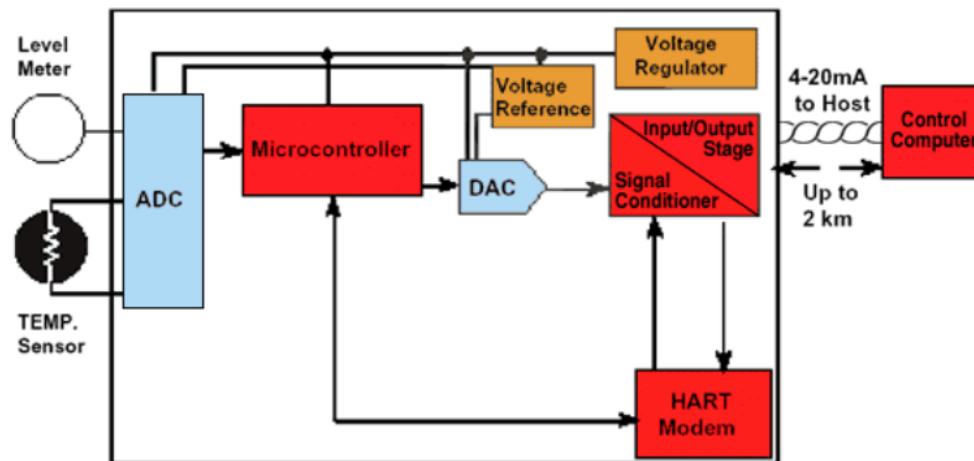


Abbildung 42 Smart Transmitter

Das Eingangssignal wird von einem **A/D-Wandler** digitalisiert und von einem **Mikroprozessor** mit den **Parametern** für Messbereich, Nullpunktverschiebung, usw. aufbereitet. Die Parameter liegen permanent im Transmitter in digitaler Form vor und werden typisch aus der Ferne zu übertragen.

#### Wireless HART:

2007 wurde der Wireless HART-Standard festgelegt und veröffentlicht. Die Funkübertragung verwendet **TDMA** als Übertragungsverfahren. Übertragen werden dieselben Informationen wie beim verdrahteten HART. Die Übertragung erfolgt verschlüsselt in einem Zyklus von mehreren Sekunden.

Wireless HART gewährt eine hohe Zuverlässigkeit und besitzt eine **eingebaute Redundanz**: Fällt ein Teilnehmer als Übertragungsweg aus, wird die Übertragung automatisch über einen anderen Teilnehmer aufgebaut.

#### Anwendungsschicht:

Mit dem Ziel des bestmöglichen Zusammenwirkens von HART-kompatiblen Geräten wurden Konformitätsklassen für Master- und Slave-Geräte festgelegt. Dabei wird zwischen **universellen** Anweisungen (von allen Geräten verstanden), **Standardanweisungen** (von den meisten Geräten verstanden) und **spezifischen Anweisungen** unterschieden.

#### Anwendungsbereich:

Das HART-Protokoll hat sich im Bereich der **chemischen, petrochemischen und verfahrenstechnischen Industrie** als Quasistandard durchgesetzt. Sehr häufig müssen in dieser Industrie Signale in **explosionsgefährdeten Bereichen** erfasst werden. Auch im Fehlerfall, z.B. bei Leitungskurzschluss, darf sich das umgebende **Medium nicht entzünden**. Durch Limitierung der zugeführten Energie kann erreicht werden, dass sich auch im Fehlerfall, z.B. bei Leitungskurzschluss, kein Zündfunke bilden kann (Zündschutzart „**Eigensicherheit**“).

Weitere Infos:

<http://de.hartcomm.org/>



## 4.2 AS-Interface

AS-i ist die Abkürzung für **Aktor-Sensor-Interface**. AS-i ermöglicht eine einfache und kostengünstige Einbindung von Sensoren und Aktoren in die industrielle Kommunikation. AS-i ist ein **offener Standard**. Eine Vielzahl von Herstellern bietet Produkte mit AS-Interface an.

AS-i ersetzt einen aufwendigen Kabelbaum durch eine für mehrere Sensoren und Aktoren gemeinsame **ungeschirmte Zweidrahtleitung**. Aktoren und Sensoren sind überwiegend einfache Komponenten, die vorwiegend **Bitsignale** erfordern oder liefern und die für den Betrieb eines Anlagenprozesses nötig sind. Ab der Version 2.0 ist auch die Übertragung von **Analogsignalen** möglich.

Durch die robuste Aufbautechnik in **Schutzart IP67** ist das AS-Interface auch den, gerade im untersten Feldbereich üblichen, harten Einsatzbedingungen gewachsen.

### Aufbau:

Das AS-Interface ist als **Master-Slave System** mit **zyklischem Polling** konzipiert. Slaves sind als **AS-i Module**, an die konventionelle Sensoren/Aktoren anschließbar sind, oder als **Sensoren/Aktoren mit integriertem AS-i** Anschluss verfügbar.

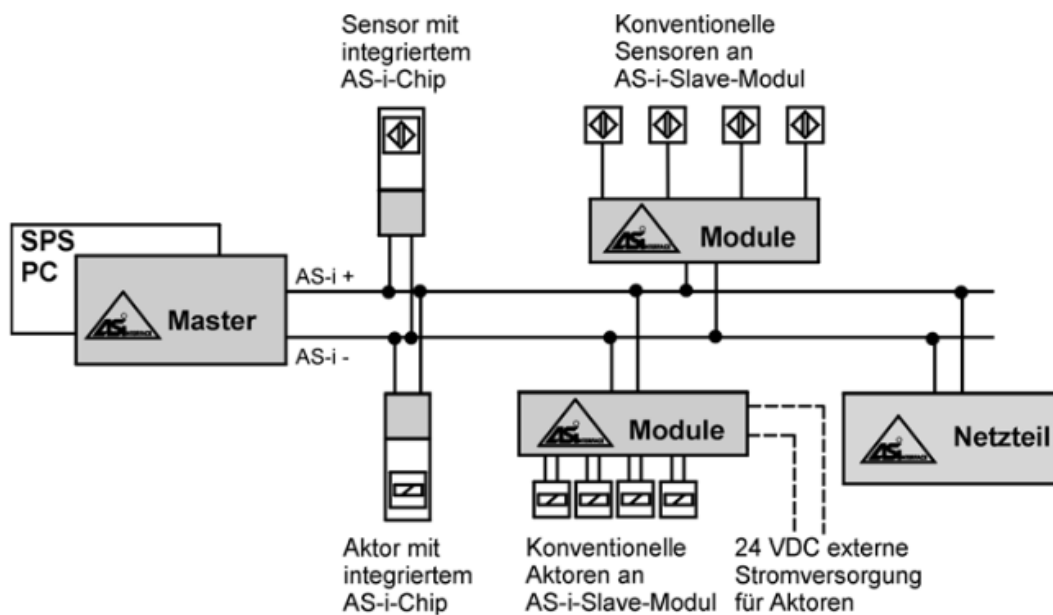


Abbildung 43 AS-i Bussystem

### Datenraten:

Das AS-Interface ist bezüglich Datenmenge und -durchsatz auf die den Anforderungen der **untersten Feldebene** optimiert. Die Datentelegramme haben eine **festgelegte Struktur** und eine **vorgeschriebene Länge**. In einem Zyklus werden insgesamt bis zu **vier nutzbare Datenbits** in **Eingangs- und in Ausgangsrichtung** zwischen jedem Slave und dem Master ausgetauscht.

Jedes AS-Interface Telegramm wird im Empfänger über **Paritätsbits** und mehrerer unabhängiger, weiterer Größen überwacht. Dadurch ist eine hohe Sicherheit bei der Erkennung von Ein- und Mehrfach-Fehlern gewährleistet. Der Einsatz von AS-Interface ist dadurch auch in stark belasteter Umgebung problemlos möglich. Im Fehlerfall werden die **Telegramme wiederholt**.

### Zykluszeiten:

Die maximale Zykluszeit, also die Zeit, die der Master höchstens benötigt, bis der Teilnehmer erneut abgefragt wird, beträgt bei einem mit bis zu **31 Slaves** voll ausgebauten System **5ms**.

Das Abfrageverfahren ist **deterministisch** und damit **echtzeitfähig**, d.h. der Master kann sich darauf verlassen, dass er innerhalb eines bestimmten Zeitintervalls die aktuellen Daten jedes im AS-Interface-Netz angeschlossenen Teilnehmers zur Verfügung gestellt bekommt.

### AS-i Versionen:

	V 2.0	V 2.1	V 3.0
Jahr	1994	1998	2004
Teilnehmer	max. 31 - 4EA	max. 62 - 4E/3A	max. 62 - 4E/4A
Anzahl E / A	124 / 124	248 / 186	248 / 248
Zykluszeit	$\leq 5\text{ms}$	$\leq 10\text{ms}$	$\leq 20\text{ms}$
Analogwerte	mit FB in der SPS	im Master integriert	im Master integriert

### Bustopologie:

Die Topologie eines AS-i-Netzwerkes ist als **Linien-, Stern- oder Baumstruktur** frei wählbar und kann den örtlichen Anforderungen angepasst werden. Es sind **keine Busabschlusswiderstände** erforderlich. Die **max. Leitungslänge beträgt 100m** oder mit **Repeater** kann 2x bis auf **300m** verlängert werden.

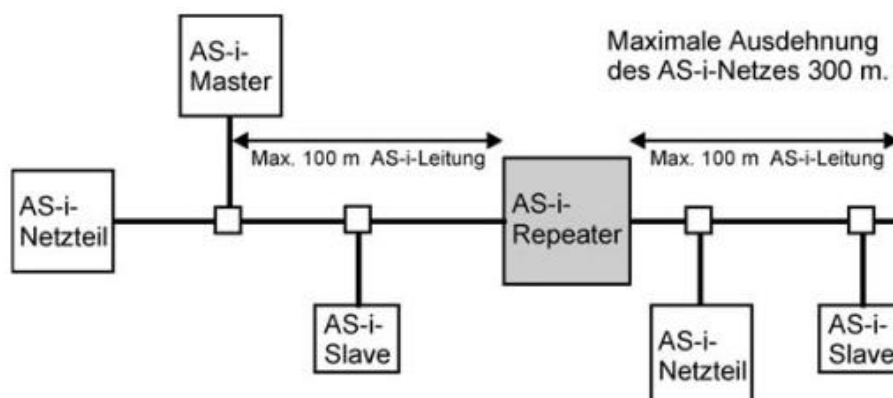


Abbildung 44 Netzwerkerweiterung durch Repeater

### AS-i Leitung:

Die AS-i Leitung ist als gummierte **gelbe ungeschirmte 2-Draht-Leitung** mit **2 x 1,5 mm<sup>2</sup>** Leitungsquerschnitt in **Schutzart IP67** ausgeführt. Das spezielle Profil verhindert eine Verpolung bei der Montage der Slaves.

Der Anschluss erfolgt in **Durchdringungstechnik**. Die AS-i Leitung ist dabei **selbstheilend**. Dies bedeutet, dass sich die durch die Durchdringungsdorne verursachten Löcher in der Gummiummantelung bei Entfernung eines Slaves selbständig schließen und die Schutzart IP67 erhalten bleibt.

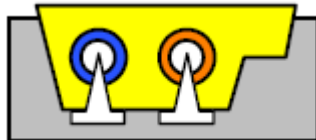


Abbildung 45 AS-i Leitung im Querschnitt mit Teilnehmeranschluss

### AS-i Netzteil:

Das AS-i Netzteil dient primär zur Energieversorgung der Teilnehmer an der AS-i Leitung.

Die Versorgung erfolgt mit ca. **30V** und bis zu **8A** je Busstrang. Jeder AS-i Busstrang benötigt sein eigenes Netzteil. Für Slaves, die einen größeren Leistungsbedarf erfordern, kann eine **externe 24V** Spannungsquelle vorgesehen werden.

Zur **Datenentkopplung** müssen **Widerstands- und Induktivitätsglieder** in die **Versorgungsleitung** eingefügt werden. In jedem AS-i Netzgerät sind diese Glieder integriert.

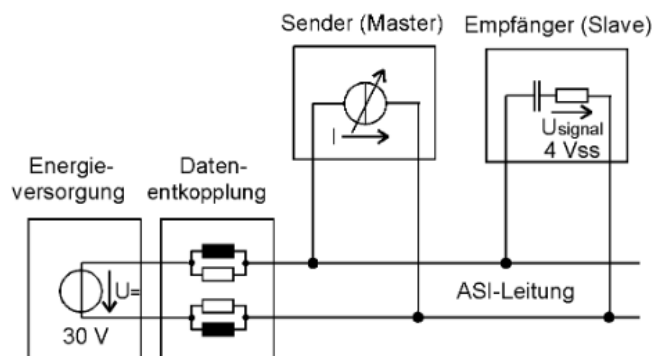


Abbildung 46 AS-i Datenentkopplung am Netzteil

### AS-i Master:

Der AS-i Master-Controller enthält einen Prozessor, dessen Software vom Hersteller mitgeliefert wird, so dass nach der Konfiguration der Slaves die Kommunikation zwischen Master und Slaves völlig selbständig abläuft. Es brauchen **keine Einstellungen** vorgenommen zu werden. Der Master führt automatisch alle Funktionen aus, die für das korrekte Funktionieren des AS-Interface nötig sind. Darüber hinaus ermöglicht er die Selbstdiagnose des Systems.

### AS-i Slaves:

Jeder AS-i Slave enthält einen integrierten Schaltkreis (AS-i Slave IC), der die Ankopplung eines AS-i Gerätes (Sensor/Aktor) an die gemeinsame Busleitung zum AS-i Master realisiert. Der integrierte Schaltkreis besitzt:

- 4 digitale Eingänge/Ausgänge oder alternativ 2 Analogkanäle mit 16Bit
- 4 Parameter

Die Betriebsparameter, Konfigurationsdaten der E/A-Belegung, Identifikationscode und Slave-Adresse sind in einem zusätzlichen EEPROM-Speicher abgelegt.

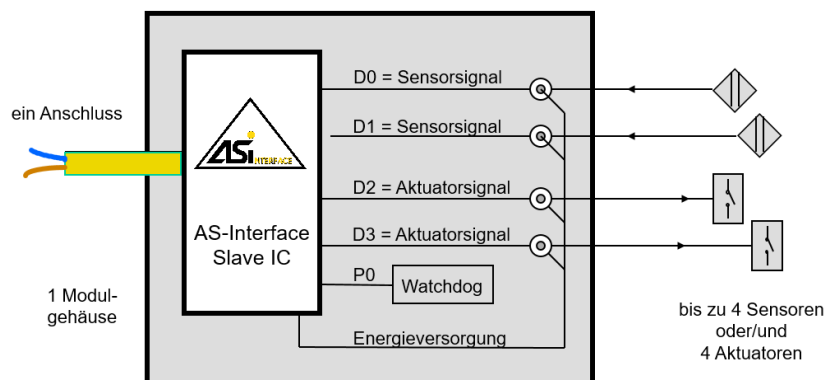


Abbildung 47 AS-i Interface Slave IC

### Erfassung von Analogdaten:

Durch die nur 4 Bit breite Datenübertragung war es anfänglich nicht möglich, direkt Analogdaten mit dem AS-Interface zu verarbeiten. Mit der Spezifikation 2.1 wurde dieses Problem behoben.

Die Übertragung von Analogdaten erfolgt mit einem **Multiplexverfahren** mit 4 Bits pro Zyklus, bei dem der Analogwert in maximal 8 Zyklen zum Master transferiert wird. Pro Zyklus werden 3 Bit und ein Kontrollbit übermittelt. Damit reduziert sich die Zykluszeit für Analogwerte natürlich auf **maximal 8 x 5ms**, also maximal **40ms**. Das ist zwar nicht sehr schnell, aber für sehr viele Anwendungen durchaus genügend. Der Anwender bemerkt von diesem Multiplexverfahren selbst nichts. Der fertig aufbereitete Analogwert steht korrekt im Prozessabbild des Masters.

### Konfiguration des AS-Interface:

Jedem Slave wird bei der Inbetriebnahme eine **eindeutige AS-i Adresse** zugewiesen und im internen Slave-Speicher permanent gespeichert. Die Adressierung erfolgt entweder mit dem **Adressiergerät** oder **über den Master**, in dem jeder Slave einzeln angeschlossen und per Adressiertelegramm beschrieben wird.

### Datenübertragung:

Die AS-i-Leitung steht zur Übertragung von Datentelegrammen in **bitserieller Form** und gleichzeitiger Übertragung eines Gleichspannungspegels für die Elektronik der angeschlossenen Slaves zur Verfügung. Um den Gleichspannungspegel nicht durch die überlagerten Daten-

telegramme zu verändern, müssen diese **gleichstromfrei** sein.

Im Sender werden die zu übertragenden Bits nach der **Manchesterkodierung** kodiert (0 bedeutet eine fallende, 1 eine steigende Flanke in der Bit-Mitte). Bei einer fallenden Flanke des Manchester - kodierten Signals wird vom Sender ein Stromfluss eingeleitet. Dieser Stromfluss überlagert sich dem Gleichstrom der zu Versorgung der Teilnehmer fließt und bewirkt somit eine Änderung des Gesamtstroms. Diese Stromänderung bewirkt an den Widerstands- und Induktivitätsgliedern des ASI- Netzteils die **Selbstinduktion**  $u = L \cdot \frac{di}{dt}$

von Spannungsänderungen die sich der Versorgungsgleichspannung überlagern. Diese überlagerten Spannungsänderungen werden vom Empfänger gefiltert und ausgewertet.

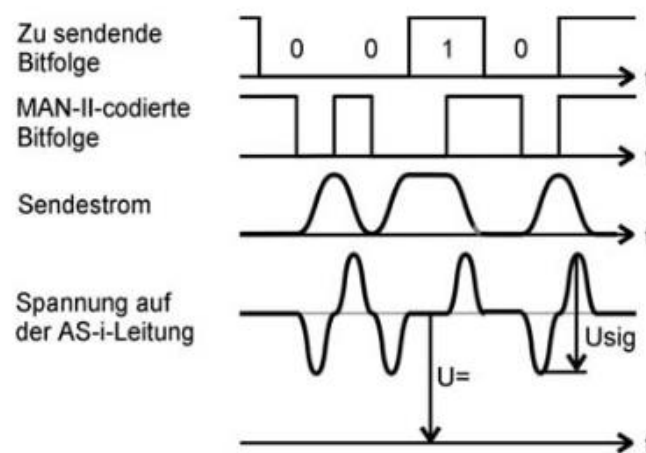


Abbildung 48 AS-i Signale

### AS-i Safety at Work:

Mit *AS-i Safety at Work* wurde ein zertifizierter Standard entwickelt, der den Einsatz von **sicherheitsgerichteten Komponenten am AS-i Bus** ermöglicht. Hierzu zählen Sicherheitsmonitore, Not-Stopp-Betätigungen, Türverriegelungen, Lichtgitter, optische Scanner usw.

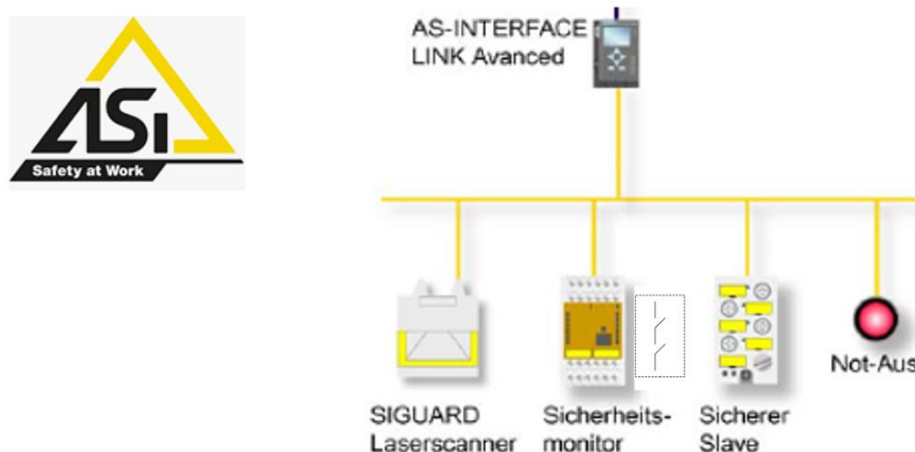


Abbildung 49 AS-i Safety at Work

Der AS-i-Sicherheitsmonitor überwacht innerhalb des AS-i-Systems, entsprechend der vom Anwender per Konfigurationssoftware angegebenen Konfiguration, die ihm zugeordneten sicherheitsgerichteten Slaves. Im Fall einer Stopp-Anforderung oder eines Defektes schaltet der AS-i-Sicherheitsmonitor mit einer Reaktionszeit von maximal **40ms** sicher ab.

Basis der sicheren Datenübertragung ist ein **dynamisiertes sicheres Übertragungsprotokoll** zwischen dem Sicherheitsmonitor den sicheren AS-i Slaves. Pro Zyklus erwartet der Sicherheitsmonitor von jedem AS-i Slave ein **spezifisches Telegramm**, das sich nach einem definierten Algorithmus kontinuierlich ändert und die Erkennung von Übertragungsfehlern ermöglicht.

#### **Zusammenfassung:**

- Master-Slave Zugriffssteuerung
- im Standard bis zu 31 Slaves an einer Leitung mit 5ms Zykluszeit
- kleine Datenpakete, niedrige Datenrate, hohe Störfestigkeit
- 4 digitale Eingänge und 4 digitale Ausgänge pro Slave
- analoge E/A möglich durch Übertragung in mehreren Buszyklen möglich
- ungeschirmte verpolungssichere 2 Draht Leitung, Schutzart IP67
- Information- und Energieübertragung mit einer Leitung
- 100m Leitungslänge, 300m mit Repeater
- freie Netztopologie und keine Abschlusswiderstände erforderlich
- AS-i Safety Sicherheitsmodule in das Bussystem integrierbar

Link Produktinfo ASi-Bus Fa. Bihl & Wiedemann:

<https://www.bihl-wiedemann.de/de/produkte.html>

### 4.3 IO-Link

IO-Link ist eine **standardisierte IO-Technologie** (IEC 61131-9) um mit Sensoren und Aktoren zu kommunizieren. Die leistungsfähige **Punkt-zu-Punkt Kommunikation** basiert dabei auf einem **3-Leiter Sensor/Aktor-Anschluss**. IO-Link ist somit kein echter Feldbus, sondern eine Weiterentwicklung einer vorhandenen Anschlusstechnik für Sensoren und Aktoren.

#### Komponenten:

Ein IO-Link System besteht aus folgenden Komponenten:

- **IO-Link Master**
- **IO-Link Device** - Sensoren, RFID-Reader, Ventile, Motorstarter, I/O-Module
- **Ungeschirmte 3- oder 5-polige Anschlussleitung**
- **Engineering-Tool** zur Projektierung und Parametrierung von IO-Link

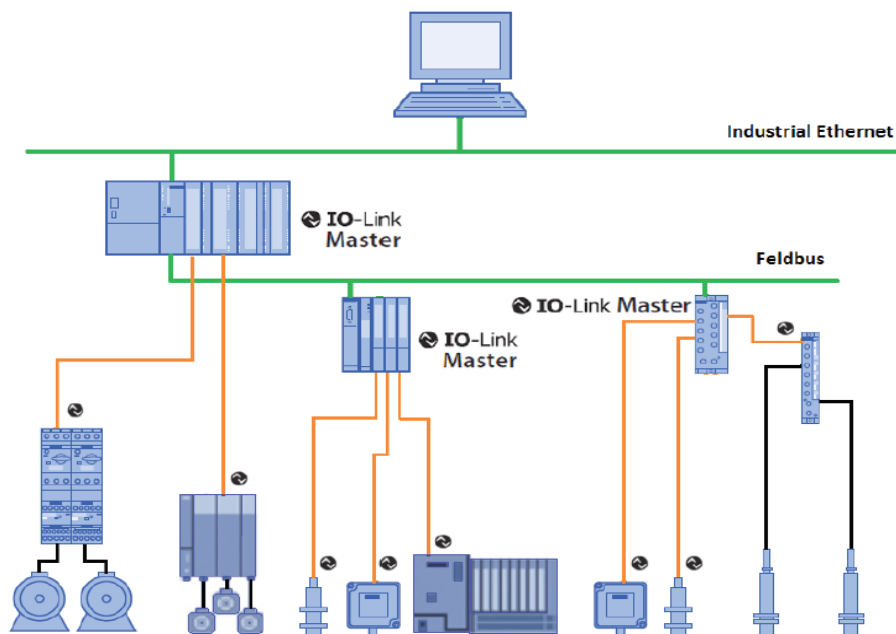


Abbildung 50 Anlagenarchitektur mit IO-Link

Bei dem Konzept handelt es sich um ein **Sensor-Aktor-Netzwerk**, das aus den Feldgeräten, den Sensoren und Aktoren und dem **IO-Link-Master** besteht. Ein IO-Link-Master besitzt typisch **mehrere Portanschlüsse**, zum Anschluss jeweils **eines IO-Link Device**. Üblicherweise werden die IO-Link-Geräte über M12-Stecker an den Master angeschlossen.

Der **IO-Link-Master** bildet die **Schnittstelle zur übergeordneten Steuerung**. IO-Link-Geräte übertragen ihre Kenndaten und Parameter über das IO-Link-Protokoll. Der Master kann die **Parameter der IO-Link-Geräte vorgeben und im Betrieb ändern**.

Die IO-Link-Spezifikation unterscheidet für IO-Link-Master zwischen **2 Typen von Ports** mit unterschiedlicher Stromversorgung. bei **Typ A** die Funktion des Kontaktes von Pin 2 des 5-poligen

M12-Steckers vom Hersteller frei definiert werden kann, werden an **Typ B** nur Geräte mit spezieller Spannungsversorgung für Geräte mit erhöhtem Strombedarf angeschlossen.

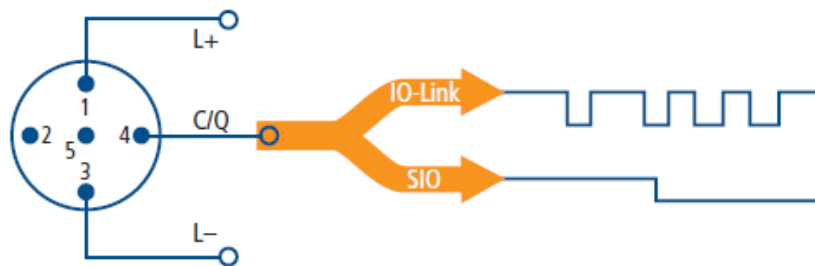


Abbildung 51 Anschlussbelegung IO-Link Device (Typ A)

### Datenübertragung:

Bei dem Anschluss von **normalen binären Sensoren** und **Aktoren** wird über die C/Q-Leitung der **Schaltzustand eingelesen** bzw. **ausgegeben**.

In der **IO-Link Betriebsart** werden die Daten über ein **Kommunikationsprotokoll** übertragen. IO-Link arbeitet mit einem **24V-Pegel** über eine maximal **20m** lange, ungeschirmte, **3-adrige Leitung**. Die Datenübertragung erfolgt **seriell** im **halbduplex-Modus**.

Folgende drei Datenübertragungsraten sind spezifiziert:

- COM 1 = 4,8 kBaud
- COM 2 = 38,4 kBaud
- COM 3 = 230,4 kBaud (optional nach Spezifikation V1.0)

Standardmäßig stehen **2 Byte Prozessdaten pro Zyklus** zur Verfügung. Nach einer fehlerhaften Übertragung wird das Telegramm noch 2-mal wiederholt.

Grundsätzlich stehen vier Datenarten zur Verfügung:

- Prozessdaten mit Status → zyklische Daten und deren Gültigkeit
- Gerätedaten → azyklische Daten - z.B. Parameter, Diagnoseinformationen
- Ereignisse → azyklische Daten - z.B. Warnungen, Fehler

### Gerätebeschreibung – IODD:

Jedes IO-Link Device besitzt eine IODD (IO-Device Description). Das ist eine **Gerätebeschreibungsdatei**, in der Herstellerdaten, Artikelnummer, Funktionalität usw. enthalten sind. Der Aufbau der IODD ist für alle Devices aller Hersteller gleich. Damit ist die gleiche Handhabung für alle IO-Link-Devices **herstellernunabhängig** garantiert.

Link IO-Link Infos Fa. Siemens:

<https://new.siemens.com/global/de/produkte/automatisierung/industrielle-kommunikation/io-link.html>



#### 4.4 KNX-Bus

**KNX** ist ein Feldbussystem zur **Gebäudeautomation**. Auf dem Markt der Gebäudeautomation ist KNX der Nachfolger der Feldbusse **EIB** (European Installation Bus). Der KNX-Standard ist ein offener Standard, dem sich mittlerweile an die 400 Firmen weltweit angeschlossen haben.

Intelligente Gebäudesysteme werden eingesetzt, um die Eigenschaften von Gebäuden in den Bereichen **Betriebskosten**, **Sicherheit** und **Flexibilität** der Nutzung sowie die **Energieeffizienz** zu verbessern.

In herkömmlichen Elektroinstallationen sind die Steuerfunktionen fest mit der Energieversorgung verbunden und erfolgen mittels Parallel- oder Reihenschaltung. Nachträgliche Schaltungsänderungen sind schwierig umzusetzen. Die Realisierung übergeordneter Steuerfunktionen wie z.B. zentrales Schalten mehrerer Beleuchtungskreise ist nur mit hohem Aufwand möglich.

##### Aufbau:

KNX ist ein **dezentrales ereignisgesteuertes** Feldbussystem. Es gibt **kein Zentralgerät**, die Funktionalität des Systems ist auf alle Busgeräte verteilt. Bei Ausfall eines Gerätes bleibt die Funktion des verbleibenden Systems erhalten.

KNX trennt die **Gerätesteuerung** und die **Energieversorgung** in zwei Netze. Das Steuerungsnetz zeichnet sich durch eine hierarchische Struktur mit **Bereichen** und darunterliegenden **Linien** aus. Jeder Bereich enthält eine eigene Energieversorgung. Eine KNX-Anlage kann praktisch bis zu ca. 10.000 Teilnehmer umfassen.

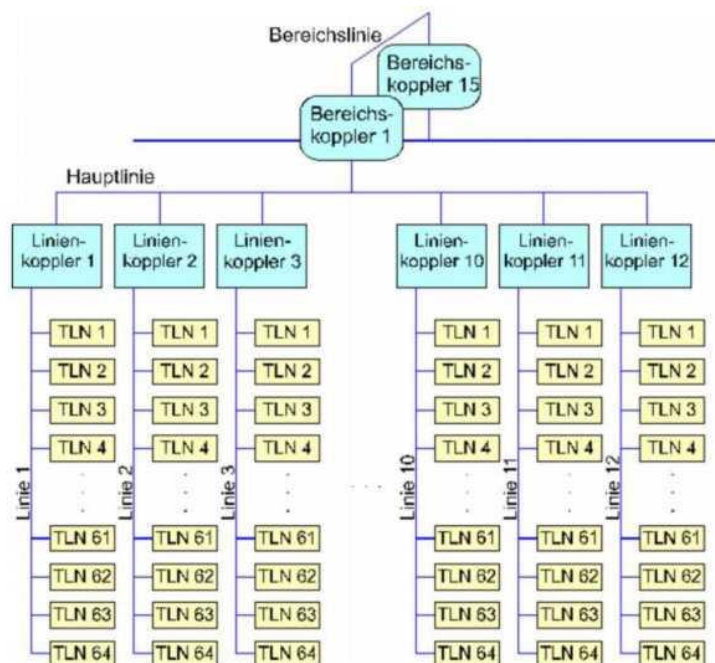


Abbildung 52 KNX Systemaufbau

**Linien-** bzw. **Bereichskoppler** dienen nicht nur zur logischen Strukturierung der Anlage, sondern auch zur Senkung des Kommunikationsaufkommens in den einzelnen Teilabschnitten. So werden Telegramme nur dann zu anderen Linien oder Bereichen weitergegeben, wenn dort über die

Adresse entsprechende Teilnehmer angesprochen werden sollen. Die Koppler übernehmen somit zugleich eine **Filterfunktion**.

### Beispiele von Busgeräten:

Systemgeräte:	Sensoren:	Aktoren:	Sonstige:
Spannungsversorgung Linien- und Bereichskoppler Linienverstärker Buskoppler (IP)	Tastsensor Bewegungssensor Glasbruchsensor	Schaltaktor Jalousienaktor Rolladenaktor	Logikmodul Bedienpanel



Abbildung 53 Schaltaktor 6-fach, Tastsensor 4-fach

### Übertragungsmedien:

Für die Übertragung der Daten zwischen den Teilnehmern können verschiedene Übertragungsmedien eingesetzt werden:

- **Twisted Pair - KNX TP** (am häufigsten eingesetzt)
- Power Line - KNX PL
- Funk - KNX RF
- Ethernet - KNX IP (für Bereichs- und Linienebene)

#### 4.4.1 KNX Twisted Pair (KNX TP)

Bei KNX TP überträgt eine Twisted-Pair Busleitung die Betriebsspannung und die Daten für die angeschlossenen Teilnehmer, d.h. die Information wird auf die Betriebsspannung moduliert. Die Nennbetriebsspannung des Bussystems beträgt **24 V**. Das System arbeitet zwischen 21V und 29V.

Die Datenübertragung auf der Busleitung erfolgt **asynchron** mit **9600 Bits/s** und einem **symmetrisch modulierten Datensignal** auf den Busleitungen. Wird eine logische 0 gesendet, so nimmt die Busspannung kurzzeitig ab, steigt dann wieder an und pendelt sich danach auf die Spannung von 28 V ein. Dies ist auf die Spulenwirkung der Netzdrossel zurückzuführen. Wird eine logische 1 vom Bus gesendet, so liegen 28 V an. Der Ruhezustand des Busses entspricht also dem permanenten Übertragen von Einsen.

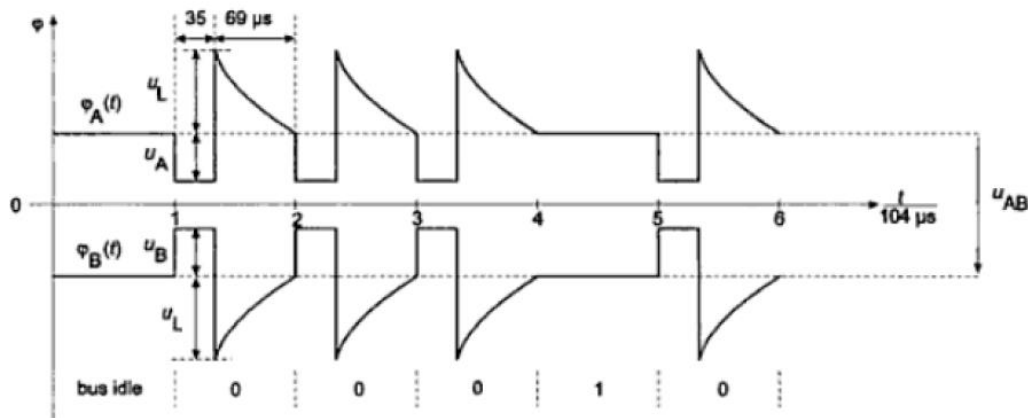


Abbildung 54 Spannungsverläufe der A- und B-Leitung bei KNX.TN (Twisted Pair)

### Installation:

Als Buskabel kommt ein spezielles grünes KNX-Kabel mit zwei verdrehten Leitungsparen  $2 \times 2 \times 0,8 \text{ mm}^2$  zum Einsatz. Signalleitungen sind die **rote(+)** und **schwarze(-) Ader**. Die gelbe und weiße Ader sind Reserve oder zur zusätzlichen Spannungsversorgung von Teilnehmern. Der Schirm wird nicht aufgelegt. Es sind **keine Abschlusswiderstände** erforderlich.

Bei der Installation der Busleitung muss bezüglich der Berührungssicherheit nichts beachtet zu werden, da die Busspannung in den Bereich der **Schutzkleinspannung** fällt.

Die Busteilnehmer werden über eine **Busklemme** angeschlossen. Die Busklemme ist eine Steckklemme, die bis zu vier KNX Kabel verbinden kann. Die Busklemme sorgt dafür, dass der Teilnehmer vom Bus genommen werden kann, ohne dass die Busleitung unterbrochen wird.

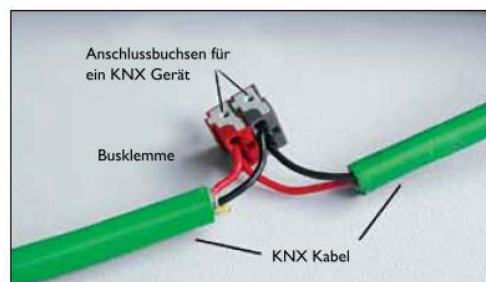


Abbildung 55 KNX Busklemme

### Topologie:

Außer einer Ringstruktur sind beliebige Topologien zulässig. Bei der Leitungsführung sind aufgrund der physikalischen Übertragungseigenschaften folgende maximalen Leitungslängen zulässig:

- Abstand zwischen Spannungsversorgung und Teilnehmer: max. 350m
- Abstand zwischen zwei Teilnehmern: max. 700m
- Gesamtleitungslänge einer Linie mit allen Verzweigungen: max. 1000m

### Buszugriffsverfahren:

Der Buszugriff wird durch ein **CSMA/CA - Buszugriffsverfahren** geregelt. Beim gleichzeitigen Zugriff von zwei Teilnehmern auf den Bus setzt sich derjenige mit der höheren Priorität durch, während sich der andere wieder zurückzieht, um es zu einem späteren Zeitpunkt noch einmal zu versuchen. Dabei hat eine wiederholte Nachricht die höchste Priorität. Besitzen beide Teilnehmer die gleiche Priorität, so bestimmt die Quelladresse der Teilnehmer.

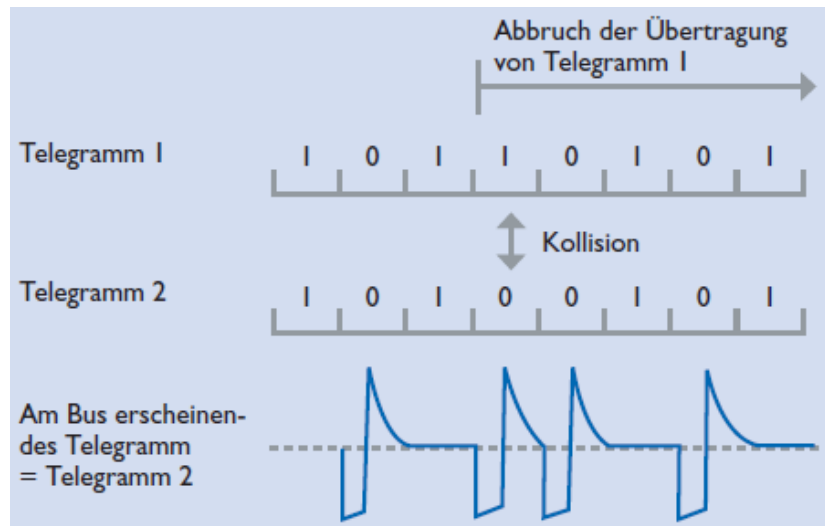


Abbildung 56 Kollisionsvermeidung bei KNX

### Protokoll und Adressierung:

Ein KNX-TP Telegramm besteht aus einer **Folge von UART-Zeichen** mit Startbit, 8 Datenbits, Paritätsbit und Stopbit.

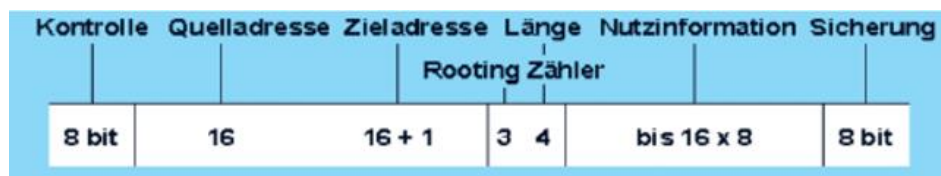


Abbildung 57 KNX Telegrammaufbau

Im **Kontrollfeld** sind die **Prioritäten** der Teilnehmer festgelegt. Bei gleichen Prioritäten zweier Teilnehmer wird zusätzlich die Quelladresse zur Prioritätsbestimmung miteinbezogen.

Die **Quelladresse** enthält die **physikalische Adresse** des sendenden Teilnehmers. Im Servicefall ist somit ein Sender leicht identifizierbar. Diese Adresse wird bei der Programmierung vergeben und identifiziert jeden Teilnehmer eindeutig.

Die **Zieladresse** dient zur Teilnehmeradressierung und kann eine **physikalische** oder **logische Adresse** beinhalten. Die Unterscheidung erfolgt mit einem zusätzlichen Adress-Bit.

Ist das Adress-Bit Null, so handelt es sich um eine physikalische Adresse, anderenfalls um eine logische **Gruppenadresse**. Bei der logischen Adressierung sind alle Teilnehmer mit derselben Gruppenadresse die Empfänger.

Das **Datenfeld** enthält die Nutzdaten und kann bis zu 16 Byte lang sein. Die Nutzdatenlänge wird im vorangestellten Zeichen angegeben, so dass der Empfänger frühzeitig die Länge der folgenden Nutzinformation erkennen kann.

### Sicherungsfeld:

Zur Erkennung von Übertragungsfehlern werden die Daten durch Prüfzeichen ergänzt. Diese Prüfzeichen werden vom Sender gebildet und in die Nachricht eingebaut.

### Physikalische Adressierung:

Sie kommt jeweils nur **einmal** im Gesamtsystem vor, kennzeichnet somit **ein Gerät** eindeutig und beschreibt, an welcher Position am Bus sich der Teilnehmer befindet. Die physikalische Adresse wird bei der Inbetriebnahme vergeben und am jeweiligen Gerät sichtbar angebracht.

Aufbau der physikalischen Adresse: **Bereich.Linie.Teilnehmer** (B.L.T)

theoretische Teilnehmeranzahl:  $16 \times 16 \times 256 = 65.536$  Teilnehmer

- Bereiche 4 Bits -> 16
- Linien 4 Bits -> 16
- Teilnehmer 8 Bits -> 256

### Logische Adressierung:

Die logische Adresse ist eine Gruppenadresse, die unabhängig von Bereichen und Linien bei der Projektierung vergeben wird. Somit können innerhalb einer Gruppe die Teilnehmer beliebiger Linien und Bereiche zusammengefasst werden.

Um die klare Struktur des Gesamtsystems zu erhalten, wird zwischen **Hauptgruppen** und **Untergruppen** unterschieden. Im Allgemeinen werden **Aufgabenbereiche** wie Beleuchtung, Jalousie, Heizung, Alarm etc. den Hauptgruppen zugeordnet und den Untergruppen entsprechende Teilbereiche daraus.

## 4.4.2 KNX Powerline (KNX PL)

Bei KNX Power Line (KNX PL) wird keine separate Busleitung verlegt, sondern die vorhandene **230V-Leitung auch als Übertragungsmedium** verwendet. Alle Geräte befinden sich **elektrisch an einer Linie**, da es keine Beschränkungen auf 64 Teilnehmer pro Segment gibt. Trotzdem erhalten die Geräte aus Kompatibilitätsgründen eine physikalische Adresse mit Bereichs-, Linien- und Teilnehmernummer. Die Busteilnehmer werden über das 230V-Netz versorgt.

Die Datensignale werden durch **Frequenzumtastung** auf die Netzspannung moduliert. Es werden 105,6 kHz für die Übertragung einer 0 und 115,2 kHz für die Übertragung einer 1 verwendet. Die **Mittenfrequenz** dieser beiden Schwingungen beträgt **110 kHz**, weshalb man das KNX PL System auch **PL110** nennt. Die Übertragungsgeschwindigkeit beträgt **1200 bit/s**.

**Buszugriffsverfahren:**

Bei KNX-PL ist das Zugriffsverfahren von KNX-TP aus technischen Gründen nicht möglich. Ein **Zeitschlitzverfahren** soll stattdessen Kollisionen schon vor ihrem Auftreten verhindern.

Mit Zeitschlitz ist ein Abstand zwischen möglichem Beginn eines Telegramms nach Abschluss des vorhergehenden Telegramms gemeint. Jeder Teilnehmer ermittelt über ein **Zufallsprinzip** für sich eine von mehreren möglichen Zeiten. Dadurch reduziert sich die Wahrscheinlichkeit einer Kollision drastisch.

#### 4.4.3 KNX Funk (KNX PF)

Die Funkversion von KNX kann sowohl mit einem bestehenden Bussystem gekoppelt, als auch alleine betrieben werden. Auf diese Weise sind **Nachrüstungen** von weiteren Geräten für die Hausautomation ebenso problemlos zu realisieren, wie die Smart Home Ausstattung von Gebäuden im Bestand. Zur Anbindung an ein bestehendes Bussystem KNX TP ist allerdings ein **Linienkoppler erforderlich**. Bei einem reinen Funksystem wird dieser nicht benötigt.

#### 4.4.4 KNX over Ethernet (KNX IP)

KNX IP bietet folgende Möglichkeiten:

- KNX-Anlagen aus der Ferne per Browser kontrollieren und steuern
- KNX-Anlagen aus der Ferne projektieren und umprogrammieren
- Kommunikation zwischen mehreren KNX-Anlagen

Mehrere KNX-Anlagen lassen sich über IP-Koppler (KNX-Gateways) zu einer Anlage zusammenfassen. IP-Koppler verfügen auf ihrer Oberseite über eine Ethernet-Schnittstelle und leiten KNX Telegramme in die darunterliegende KNX-Anlage. IP-Koppler lassen sich sowohl als **Linien-** als auch als **Bereichskoppler** einsetzen. Sie sind wie KNX TP Linien- und Bereichskoppler in der Lage, Telegramme zu filtern. KNX IP kann ausschließlich als **Haupt-** oder **Bereichslinie** ausgeführt werden.

**IP-Kommunikationsarten:**

Tunneling:

Ziel ist der Zugriff auf den Bus über ein Netzwerk zur Inbetriebnahme, zum Test oder zur Fehlersuche. Vereinfacht kann man sich das benutzte Netzwerk als verlängertes Programmierkabel zwischen dem Inbetriebnahme-PC und der KNX Anlage vorstellen.

Routing:

Ziel ist die Weiterleitung von KNX Telegrammen im laufenden Betrieb über ein Netzwerk um z.B. zwei Anlagenteile über eine vorhandene Ethernet-Strecke zu verbinden.

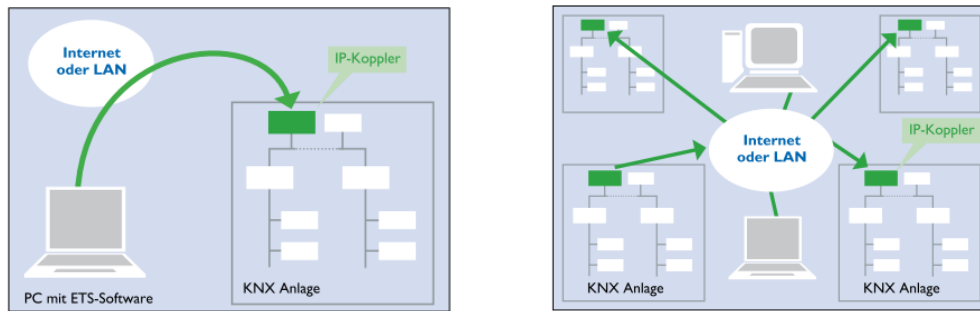


Abbildung 58 KNX Tunneling und KNX Routing

#### 4.4.5 Die ETS - Engineering-Tool-Software

**Jeder KNX Teilnehmer** muss bei der Inbetriebnahme **programmiert** werden. Um diese Programme erstellen und übertragen zu können, wird die KNX Programmiersoftware Engineering-Tool-Software (ETS) benötigt. ETS wird von der KNX-Association kostenfrei bereitgestellt.

Die ETS enthält neben den Projektierungs- und Inbetriebnahme-Werkzeugen auch umfangreiche Diagnosemöglichkeiten. Alle mit der ETS durchgeführten Arbeiten bei Projektierung und Inbetriebnahme können mit der ETS kommentiert werden.

Schritte bei der Programmierung:

- KNX Projekt anlegen
- Festlegen der Gebäudestruktur und der Bustopologie
- Die benutzten Geräte aus einer Datenbank filtern und physikalischen Adressen vergeben
- Einstellen der Parameter der KNX-Produkte entsprechend den Erfordernissen
- Festlegen der Funktionen in der Anlage und Anlegen der Gruppenadressen
- Verbinden der Kommunikationsobjekte der KNX Produkte durch Gruppenadressen

## 4.5 CAN-Bus

Das **Controller Area Network (CAN)** wurde 1983 von der Robert **Bosch GmbH** gemeinsam mit der **Fa. Intel** für den Einsatz im **Automobil** entwickelt. Aber auch in anderen Bereichen, wie z.B. in der Medizin- und Automatisierungstechnik hat sich der CAN-Bus wegen seiner Eigenschaften etabliert.

In der Automobilindustrie konnte die Fehleranfälligkeit durch den stetig steigenden Einsatz von Elektronikmodulen und deren Vernetzung wesentlich reduziert werden. Die Einführung von CAN im Automobil ermöglichte eine deutliche **Reduktion von Kabeln und Steckern** und damit **von Preis und Gewicht**.

Der CAN-Bus ist als **ISO 11898** international **standardisiert** und definiert die **Schicht 1 und 2** im ISO/OSI-Referenzmodell. In der **Automatisierungstechnik** wird häufig das **CANopen Protokoll** eingesetzt, welches den CAN-Bus um eine **OSI-Schicht 7** (Anwendungsschicht) erweitert.

### Entwicklungsgeschichte:

- 1983 - Start der Entwicklung des CAN Bus durch die Robert Bosch GmbH
- 1989 - CAN-Controller Chips verfügbar
- 1991 - CAN-Protokoll Version 2.0 Spezifikationen 1994 - erste ISO-Norm
- 1994 - Erstes Serienfahrzeug mit CAN-Bus (Mercedes S-Klasse)
- 2008 - Geschätzte 60 Millionen produzierte Fahrzeuge mit durchschnittlich 15 CAN-Knoten

### Eigenschaften:

Der CAN-Bus unterscheidet sich von anderen Bussystemen durch eine ausgeprägte Fehlerbehandlung, die Möglichkeit zur Priorisierung von Nachrichten (Echtzeitfähigkeit), eine hohe Datenübertragungsrate und das Multimaster Prinzip.

- **Linienstruktur** mit abgeschlossener verdrehter Zweidrahtleitung
- Ausdehnung abhängig von der Übertragungsrate: **40m bei 1 MBit/s** , 1000m bei 50 kBit/s
- Teilnehmeranzahl nur durch Leistungsfähigkeit der Treiberbausteine begrenzt
- **Objektorientierte Nachrichten**, Broadcasting mit Akzeptanzprüfung
- **Echtzeitfähig** für hochpriorie Nachrichten
- **Multimasterfähig**, Buszugriffsverfahren CSMA/CA
- Sehr **hohe Datensicherheit** (Hamming-Distanz 6)
- **Offenes System** (ISO 11898 und CiA DS 301)

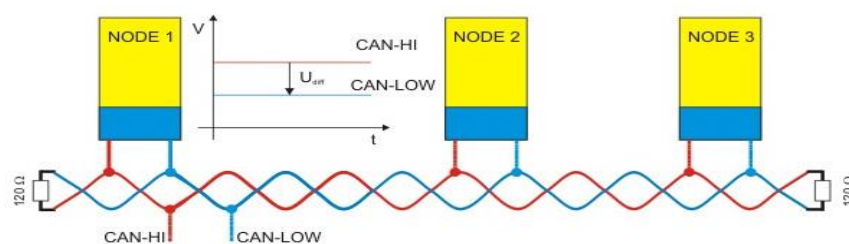


Abbildung 59 CAN-Bus mit Busteilnehmern (Nodes)



### Elektrische Eigenschaften:

Beim CAN wird eine **Linienstruktur** verwendet. Eine Nachricht kann somit alle Teilnehmer (abgesehen von der Signallaufzeit) gleichzeitig erreichen. Als Übertragungsmedium wird eine mit **verdrillte Zweidrahtleitung** eingesetzt. Die Übertragung kann über eine **RS485-Schnittstelle (Differenzspannung)** oder nach ISO 11898 erfolgen.

### Bustopologie:

Wird die Ausdehnung des CAN auf **40m** begrenzt, so ist eine maximale Übertragungsrate von **1Mbit/s** zulässig. Soll die Ausdehnung vergrößert werden, ist die Baudrate zu reduzieren. Ursache ist die in einer Bitzeit zu erkennenden Pegelüberschreitung (dominant/rezessiv), welche für das verwendete CSMA/CA - Verfahren wichtig ist. Der Bus muss mit einem 120 Ohm **Abschlusswiderstand** terminiert werden. Eine Terminierung ist auch bei kurzen Leitungen mit niedrigen Baudraten erforderlich.

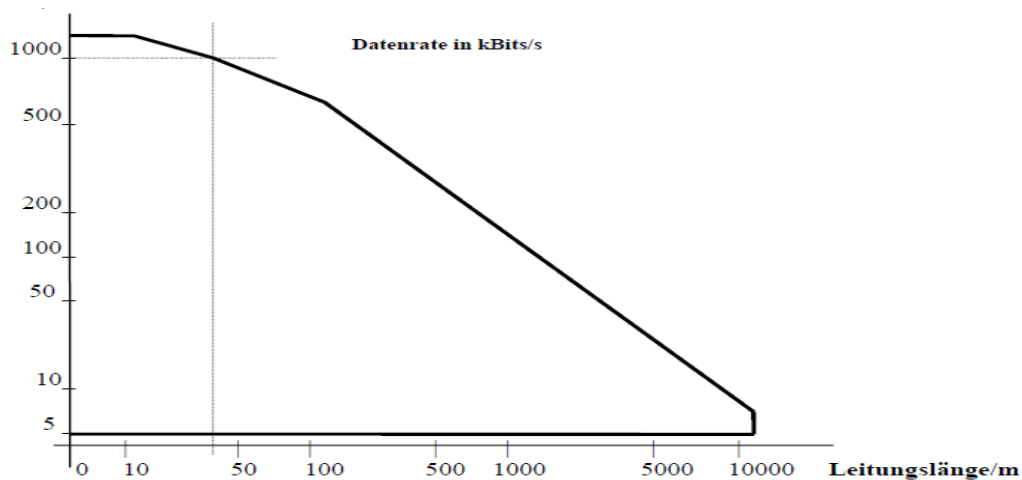


Abbildung 60 CAN Leitungslängen

### Nachrichtenorientiertes Protokoll:

Ein Hauptmerkmal des CAN-Protokolls ist die **objekt- oder nachrichtenorientierte Datenübertragung**. Es werden keine Busteilnehmer adressiert, sondern jedes Telegramm wird durch einen eindeutigen **Identifizier** gekennzeichnet. Jeder Busteilnehmer besitzen für die Selektion der für ihn interessanten Nachrichten eine **Filtereinrichtung**.

### Buszugriffssteuerung:

Am CAN-Bus sind alle Teilnehmer gleichberechtigt. Der direkte Austausch von Nachrichten im unter allen Teilnehmern ist somit möglich (**Multimaster-Betrieb**). Als Zugriffsverfahren kommt **CSMA/CA** zum Einsatz. Im Konfliktfall wird der Zugriff durch eine **bitweise Arbitration** aufgelöst. Mit dem **Identifizier** wird die **Priorität** des Telegramms festgelegt. Identifizier mit kleinerem Wert haben höhere Priorität (**0-Pegel ist dominant**).

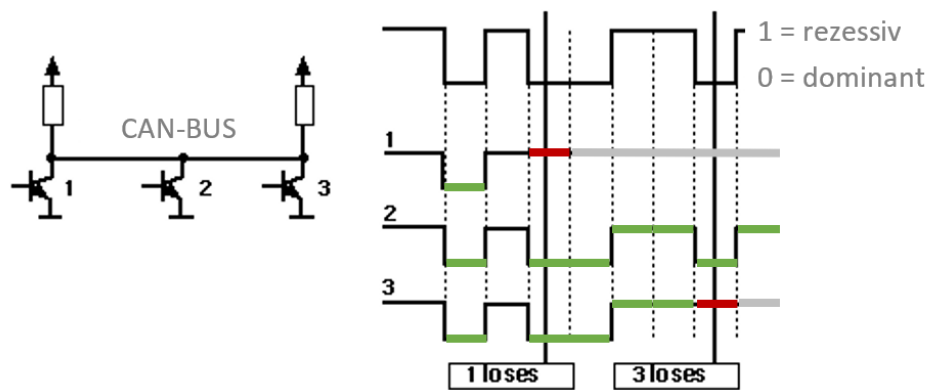


Abbildung 61 CAN Bus-Arbitrierung

**Echtzeitverhalten:**

Stellt man in der Applikation sicher, dass ein Teilnehmer mit hoch priorisierter Nachricht nicht ständig den Bus belegt, so ist es trotz des zufälligen Buszugriffes möglich, die maximale **Latenzzeit** (Wartezeit bis Bus frei ist) einer Nachricht zu bestimmen. Somit ist die allgemeine Bedingung für Echtzeitfähigkeit erfüllt.

**Aufbau einer CAN-Nachricht:**

Eine CAN-Nachricht setzt sich zusammen aus:

- **Nachrichtenennung** (Ident-Feld) = Arbitrierungsfeld
- **Remote Transmission Request Bit** (RTR)
- **Data Length Code** (DLC)
- **max. 8 Datenbytes**
- **weitere Bits:** Start (SOF), Basic/Extended (IDE), CRC, ACK, Ende (EOF), Inter Frame Space(IFS)

Zusätzlich werden ein Startbit, Ende-Bits,

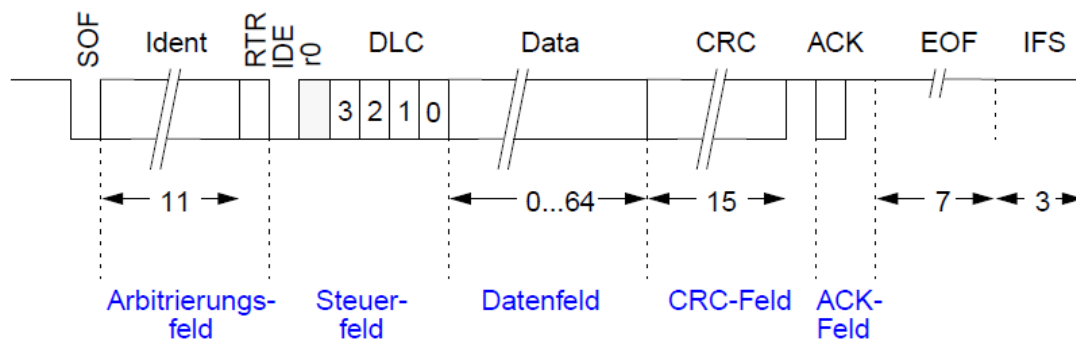


Abbildung 62 Aufbau einer CAN Nachricht

Das ergibt eine Nachrichtenlänge von **ca. 100 Bits** oder eine Sendzeit von ca. 100µs / Telegramm (bei 2 Byte Nutzdaten).

**Nachrichtenennung (Ident-Feld):**

Das Ident-Feld besteht beim Standard-CAN aus 11 Bits. Somit können bis zu **2048** unterschiedliche Nachrichten spezifiziert werden. Die **niedrigste Nummer** trägt dabei die **höchste Priorität**.

**Remote Transmission Request Bit (RTR):**

Unterscheidet zwischen **Daten-** und **Datenanforderungstelegramm**. Ist das RTR-Bit gesetzt, so wird die Nachricht mit dem eingestellten Identifier im System angefragt. Besitzt ein Teilnehmer diese Nachricht, stellt er diese im Broadcasting allen Teilnehmern zur Verfügung.

**Data Length Code (DLC) und Datenbytes:**

Die Anzahl der Datenbytes wird im DLC-Feld eingetragen. Das Feld ist 4 Bit lang. Es sind allerdings nur max. 8 Datenbytes zugelassen.

**Nachrichtenfilterung:**

Nicht alle Nachrichten sind für jeden Knoten interessant. Eine im CAN-Controller vorhandene Nachrichtenfilterung leitet nur gewünschte Nachrichten in den Empfangspuffer weiter.

CAN-Controller werden je nach Filterrealisierung und Speichergröße in **Full-CAN** und **Basic-CAN** Controller unterteilt. Bei Basic-CAN kann nur auf eine Gruppenadresse gefiltert werden (Filter auf z.B. 8 höherwertige Nachrichtenbits). Der Mikrocontroller übernimmt die weitere Auswahl. Bei **Full-CAN ist eine vollständige Filterung** möglich.

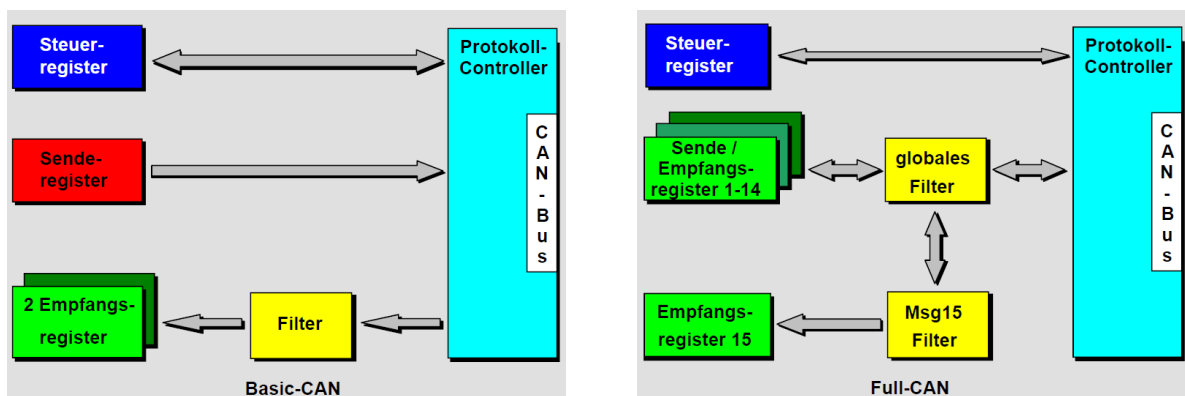


Abbildung 63 typischer Aufbau eines BasicCAN und FullCAN Controllers

**Mechanismen zur Fehlererkennung:**

- **Monitoring:** Rücklesen und vergleichen der gesendeten Bits (für CSMA-CA Buszugriff)
- **Frame-Check:** Überwachung fest vorgeschriebene Bits im Telegramm
- **Blockprüfung:** mit 15 Bit CRC
- **ACK-Bit:** jedes Telegramm wird von jedem Empfänger mit einem dominanten ACK-Bit bestätigt
- **Bitstuffing:** nach 5 gleichen Bits muss ein verschiedenartiges Bit eingeschoben werden

## 4.6 CANopen Protokoll

Das CANopen Protokoll ist ein standardisiertes **Schicht-7 Protokoll** für den CAN Bus. CANopen ist ein von der Vereinigung **CAN in Automation** (CiA) erarbeiteter Standard. In diesem Verbund ist eine Vielzahl von Geräteherstellern organisiert. CANopen hat alle bisherigen herstellerspezifischen CAN-Protokolle weitgehend ersetzt.

CiA Homepage: <https://www.can-cia.org>

### Geräteprofile:

In den CANopen-Profilen ist festgelegt, mit **welchen Telegrammen** (Identifier) und **Parametern** CANopen-Geräte angesprochen werden können. Das CANopen-**Profil CiA301**, legt grundlegende Telegramme und Parameter **für alle CANopen-Geräte** fest. Zusätzlich existieren **Profile** für eine **Reihe von Geräten und Anwendungen**.

Link CiA Profile: <https://cia-productguides.org/canopen/profiles>

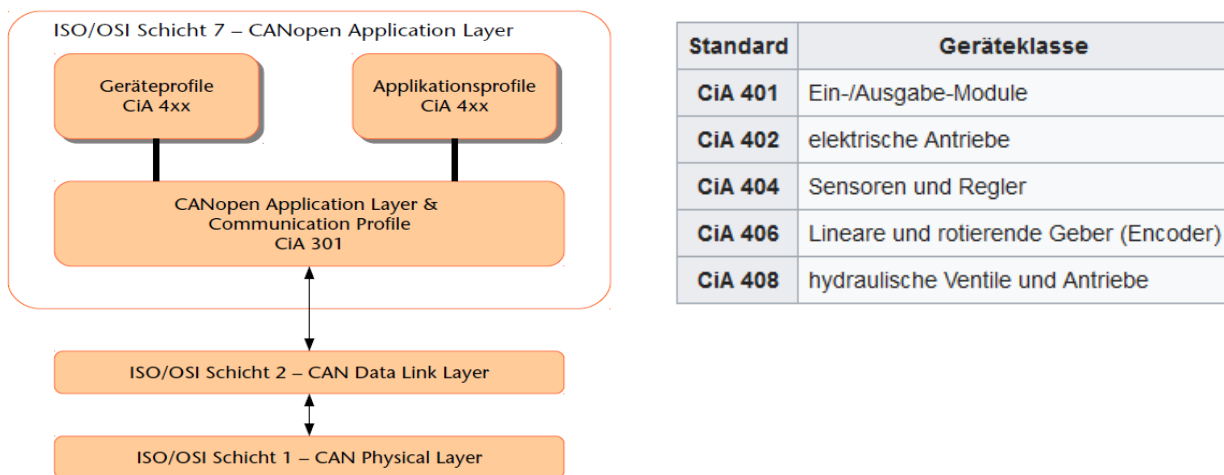


Abbildung 64 CANopen Profile

### Objektverzeichnis:

CANopen-Profile sind in Tabellenform in einem **Objektverzeichnis** organisiert. Das Objektverzeichnis beschreibt den **Funktionsumfang** eines CANopen-Gerätes. Im Objektverzeichnis sind **standardisierte** und **herstellerspezifische** Objekte enthalten. Die Adressierung eines Eintrags erfolgt mit einem **16 Bit Index** und **8 Bit Sub-Index**.

Index (hex)	Objekt
0000	nicht verwendet
0001 - 001F	statische Datentypen
0020 - 003F	komplexe Datentypen
0040 - 005F	herstellerspezifische Datentypen
0060 - 007F	profilspezifische statische Datentypen
0080 - 009F	profilspezifische komplexe Datentypen
00A0 - 0FFF	reserviert
1000 - 1FFF	Kommunikationsprofil (DS-301)
2000 - 5FFF	herstellerspezifische Parameter
6000 - 9FFF	Parameter aus den standardisierten Geräteprofilen
A000 - FFFF	reserviert

Abbildung 65 Aufbau CANopen Objektverzeichnis

**Beispiel: CANopen-Profil eines digitalen Eingangsmoduls (aus Profil CiA401):**

Index	Object Code (OC)	Name	Data Type	Category
6000h	Array	Read Input 8-bit	Unsigned8	C: DI
6001h	-	Reserved	-	-
6002h	Array	Polarity Input 8-bit	Unsigned8	O
6003h	Array	Filter Constant Input 8-bit	Unsigned8	O
6004h	-	Reserved	-	-
6005h	Var	Global Interrupt Enable Digital	Boolean	O
6006h	Array	Interrupt Mask Any Change 8-bit	Unsigned8	O
6007h	Array	Interrupt Mask Low-to-High 8-bit	Unsigned8	O
6008h	Array	Interrupt Mask High-to-Low 8-bit	Unsigned8	O
6009h	-	Reserved	-	-
to				
601Eh	-	Reserved	-	-

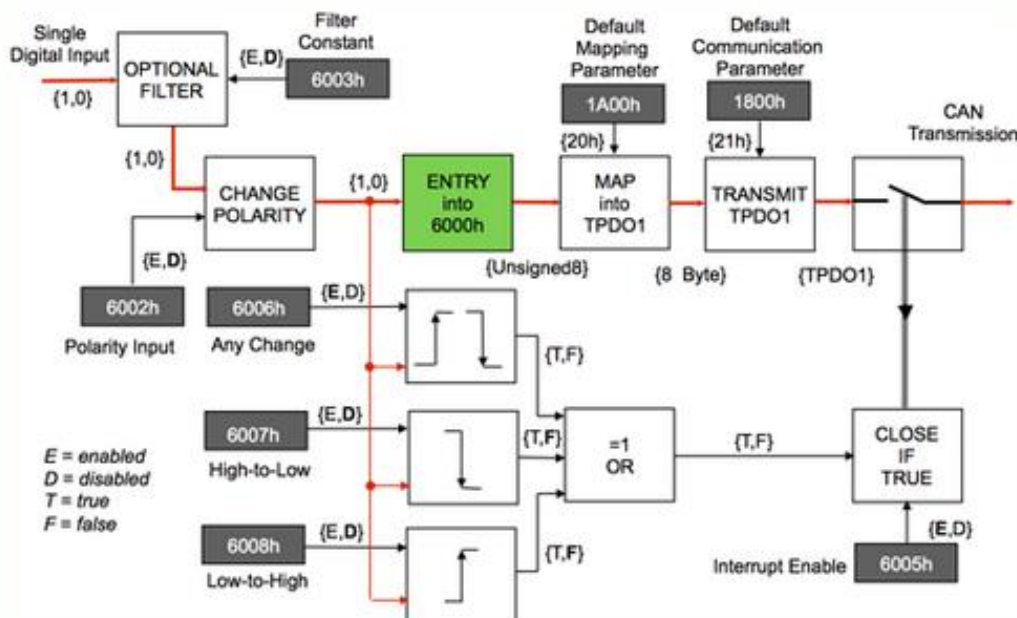


Abbildung 66 CiA401 - Geräteprofil für allgemeine IO-Module

**Kommunikations-Mechanismen:**

CANopen spezifiziert Mechanismen zum Austausch von **Prozessdaten in Echtzeit** (PDOs) und zum **zeitunkritischen Übertragen großer Datenmengen** (SDOs).

**4.6.1 Service Data Objects (SDO)**

Service Data Objects werden für den **Zugriff auf das Objektverzeichnis** und für die **Übertragung größerer Datenmengen** wie z.B. bei einem Dateitransfer eingesetzt. Jedes CANopen Gerät verfügt über **mindestens einen SDO Kanal**. Mit SDOs lassen sich **Daten beliebiger Länge** übertragen, wobei die Daten gegebenenfalls auf mehrere CAN-Telegramme aufgeteilt werden.

**SDOs werden immer bestätigt**, d.h. der Empfang jedes einzelnen Telegramms wird vom Empfänger mit einem Antworttelegramm quittiert.

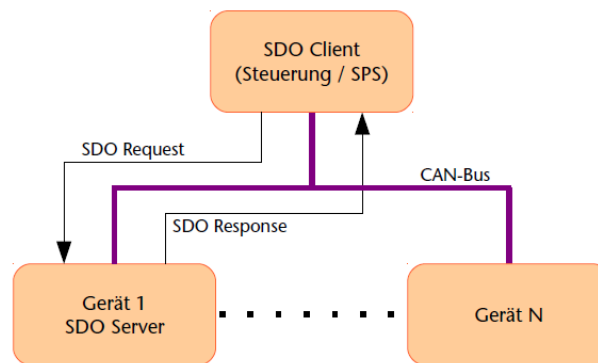


Abbildung 67 SDO Client-Server Struktur

#### 4.6.2 Process Data Objects (PDO)

Für die Übertragung von **Prozessdaten** steht der Mechanismus des PDO zur Verfügung. Jedes Prozessdaten produzierende oder konsumierende CANopen-Gerät verfügt über mindestens ein PDO. Dem Anwender stehen **8 Nutzdatenbytes** im PDO zur Verfügung.

Die **Übertragung** eines PDO **erfolgt unbestätigt ohne Protokoll-Overhead**. Die CAN-Verbindungsschicht stellt die fehlerfreie Übertragung des PDO sicher. Bestätigte Dienste sind in zeitkritischen Applikationen nicht erwünscht, weil sie die Busbandbreite signifikant reduzieren.

##### PDO Datenübertragung:

Die Übertragung der Prozessdaten kann auf unterschiedliche Arten durchgeführt werden.

- **ereignisgesteuert**

Die Aussendung des PDO wird durch ein internes Ereignis gesteuert

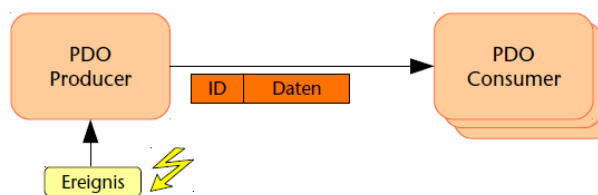


Abbildung 68 Ereignisgesteuerte PDO-Übertragung

- **synchron**

Bei der synchronen Übertragung werden von einem Busteilnehmer Synchronisationstelegramme gesendet, auf deren Empfang hin ein PDO-Producer seine Prozessdaten überträgt

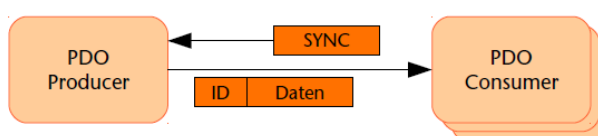


Abbildung 69 synchrone PDO-Übertragung

### PDO-Mapping:

Zur Übertragung von Prozessdaten stehen **8 Datenbytes** zur Verfügung. Da keine zusätzliche Protokollinformation übertragen wird, muss das **Format zwischen Sender und Empfänger** vereinbart werden. Dies erfolgt durch das sogenannte PDO-Mapping.

Beim **variablen Mapping** können die Prozessdaten eines PDOs wahlfrei angeordnet werden. Dazu werden aus dem Objektverzeichnis die Adresse (Index und Sub-Index) sowie die Größe in die Mapping-Tabelle des PDOs eingetragen.

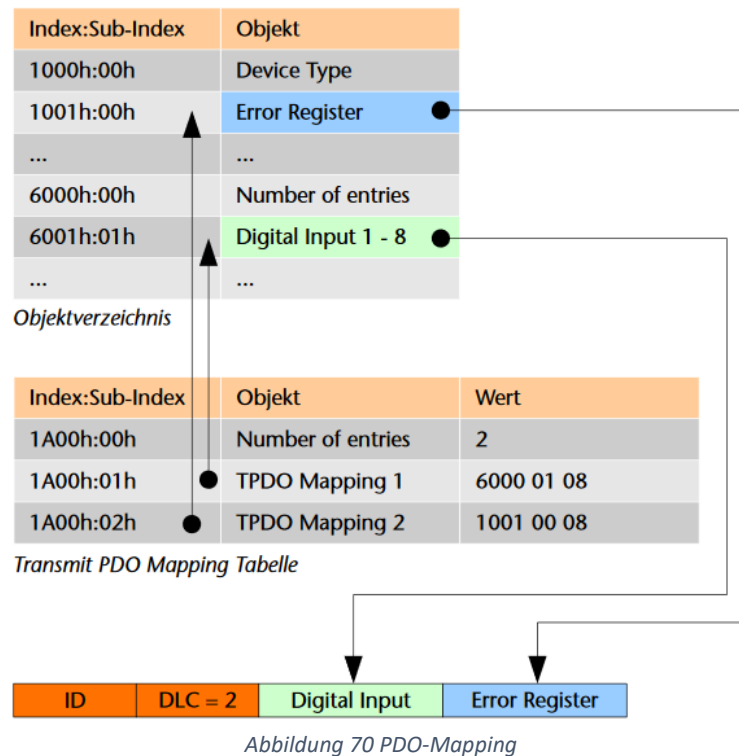


Abbildung 70 PDO-Mapping

### Network Management (NMT):

In einem CANopen-Netzwerk gibt es genau einen **NMT-Master**. Der NMT-Master besitzt die Kontrolle über alle am Bus angeschlossenen CANopen-Geräte. Er **initiiert** und **überwacht** den **Hochlauf** und prüft die Funktion nicht dauernd sendender Slaves durch **Node-Guarding**. Beim Node-Guarding sendet der Master Telegramme an die vorhandenen CANopen-Slaves, die auf diese innerhalb einer bestimmten Zeit antworten müssen.

Hochlaufzustände eines CANopen-Slaves:

- **Initialization**  
Initialisierung der Gerätefunktion, danach selbstständig in den Zustand Pre-Operational
- **Pre-Operational**  
In diesem Zustand kann mit dem Knoten über SDOs kommuniziert werden.
- **Operational**  
In diesem Zustand hat der CANopen-Knoten die volle Betriebsbereitschaft.
- **Stopped**  
Es können nur NMT-Kommandos empfangen werden. Ausgänge gehen in den Fehlerzustand.

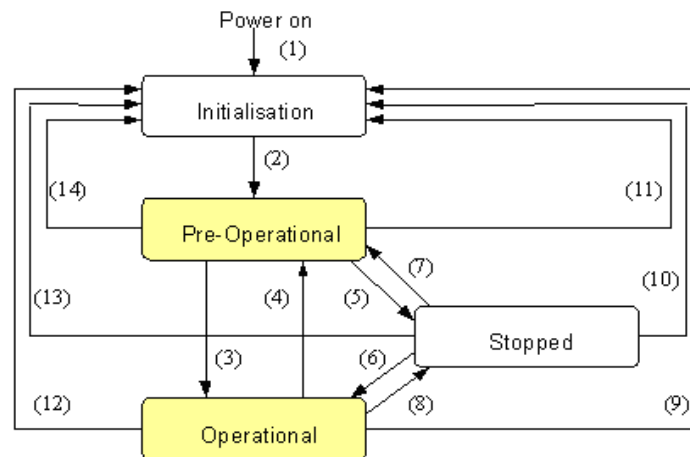


Abbildung 71 CANopen Slave-Zustände

### Emergency-Botschaften:

Botschaften vom Typ "Emergency" werden verwendet, um Fehler eines Gerätes zu melden. Im Emergency-Telegramm wird ein Code übertragen, der den Fehler eindeutig identifiziert. Die Codes sind im Kommunikationsprofil **CiA-301** sowie in den jeweiligen Geräte- und Anwendungsprofilen festgelegt.

### Anzahl Busteilnehmer und Identifier-Verteilung:

Die zur Verfügung stehende Menge von möglichen **2048 Identifier** ist in **festgelegte Bereiche** für jeweils **128 CANopen Teilnehmer** aufgeteilt: einem **NMT-Master** und **127 CANopen-Slaves**. Die CANopen-Slaves besitzen die Node-IDs (Adressen) 1 bis 127 (1-0x80 hexadezimal).

Identifier	Service	Richtung	COB-ID Berechnung	Hinweis
000 <sub>h</sub>	NMT	Receive	-	fest
080 <sub>h</sub>	SYNC	Rcv. / Trm	-	variabel, Index 1005 <sub>h</sub>
081 <sub>h</sub> .. 0FF <sub>h</sub>	EMCY	Transmit	080 <sub>h</sub> + Node-ID	variabel, Index 1014 <sub>h</sub>
100 <sub>h</sub>	TIME	Rcv. / Trm.	-	variabel, Index 1012 <sub>h</sub>
181 <sub>h</sub> .. 1FF <sub>h</sub>	TPDO1	Transmit	180 <sub>h</sub> + Node-ID	variabel, Index 1800 <sub>h</sub>
201 <sub>h</sub> .. 27F <sub>h</sub>	RPDO1	Receive	200 <sub>h</sub> + Node-ID	variabel, Index 1400 <sub>h</sub>
...				
581 <sub>h</sub> .. 5FF <sub>h</sub>	SDO	Transmit	580 <sub>h</sub> + Node-ID	fest
601 <sub>h</sub> .. 67F <sub>h</sub>	SDO	Receive	600 <sub>h</sub> + Node-ID	fest

Abbildung 72 Auszug der vordefinierten Identifier

### Electronic Data Sheet (EDS):

Das Electronic Data Sheet beschreibt die Funktionalität eines CANopen-Gerätes in maschinenlesbarer Form. EDS-Dateien werden in die Projektierungswerkzeuge der Hersteller importiert und ermöglichen so die **interaktive Konfiguration der CANopen-Geräte**.



## 4.7 Profibus

Die Geschichte von Profibus geht auf ein 1987 in Deutschland öffentlich gefördertes **Projekt „Feldbus“** zurück, an dem 21 Firmen und Hochschulinstitute mitgewirkt haben. Aus der Arbeit resultierte Anfang **1991** eine **DIN-Norm**, bekannt unter dem Namen **Profibus (PROcess Field BUS)**.

Ziel des Projektes war die Realisierung und Verbreitung eines **bitseriellen Feldbusses**, der die **Vernetzung von Automatisierungsgeräten** der unteren Feldebene von Sensoren und Aktoren bis hin zu Prozesssteuerungen in der Zellenebene ermöglicht. Mit Profibus wurde ein **Feldbus-standard** geschaffen, der **offen und firmenneutral** ist.

### 4.7.1 Einsatzbereiche (Kommunikationsprofile)

Historisch gewachsen sind **drei Varianten** des Profibus:

#### **Profibus-DP (Dezentrale Peripherie):**

Profibus-DP wird hauptsächlich in der **Fertigungsautomatisierung** eingesetzt und ist das **am häufigsten eingesetzte Profil**. Es ist auf **Geschwindigkeit, Effizienz und geringe Anschlusskosten** hin optimiert und speziell für die **Kommunikation zwischen Automatisierungssystemen und dezentralen Peripheriegeräten**, wie z.B. **E/A-Modulen und Antrieben**, zugeschnitten.

#### **Profibus-PA (Prozess Automation):**

Profibus PA ist eine Profibus-Variante mit einer für den **explosionsgefährdeten Bereichen** geeigneter **Übertragungsschicht** (ISO/OSI-Schicht 1). Profibus-PA Prozessgeräte können über Buskoppler mit dem Profibus-DP System verbunden werden.

#### **Profibus-FMS (Fieldbus Message Specification):**

Profibus-FMS war vor allem für den Einsatz in komplexen Maschinen und Anlagen auf Zellebene gedacht. Diese Protokollvariante wurde von Profibus-DP abgelöst und ist heute nicht mehr Bestandteil der Internationalen Feldbusnorm.

### 4.7.2 Aufbau der Profibus-DP

Profibus-DP basiert elektrisch auf der 2-Draht **RS485-Schnittstelle**. Der Vorteil der 2-Draht-Technik liegt hauptsächlich in der **Multimaster-Fähigkeit**. Jeder Teilnehmer kann prinzipiell mit jedem anderen Teilnehmer Daten austauschen. Durch ein **Protokoll** wird sichergestellt, dass zu jedem Zeitpunkt nur ein Sender aktiv ist.

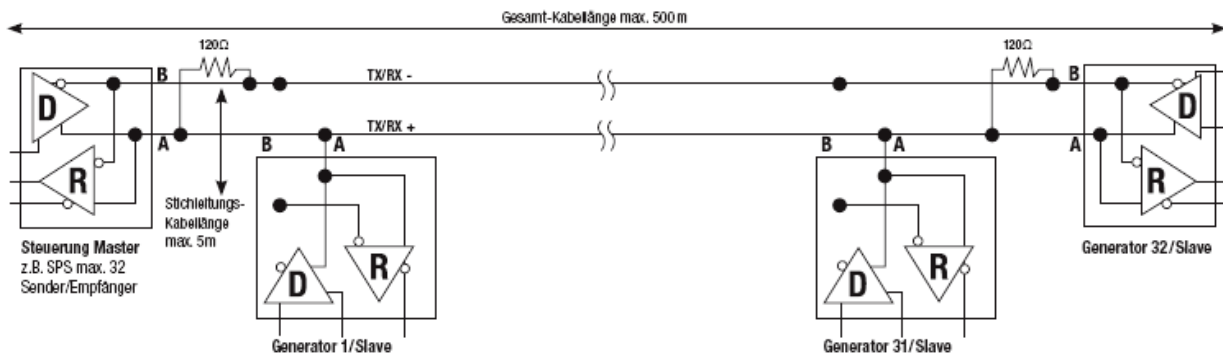


Abbildung 73 Profibus mit RS485 Teilnehmern

### Allgemeine Eigenschaften des Profibus-DP:

- Multimasterfähiges Bussystem mit **Token-Passing** Zugriffsverfahren unter den Masters und unterlagerter **Master-Slave** Kommunikation mit typische Zykluszeiten im Bereich **5 -10ms**.

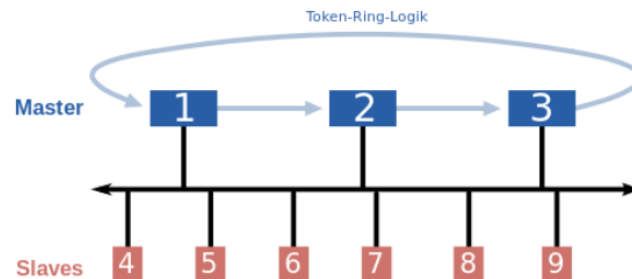


Abbildung 74 Profibus Zugriffsverfahren

- Die Datenübertragung erfolgt über eine **abgeschlossene RS485-Busleitung** oder über **Lichtwellenleiter**. Die Busteilnehmer sind während des **Betriebs an- und abkoppelbar**.
- Maximal **127 Teilnehmer am Bus** auf **mehrere Bussegmente** aufgeteilt (max. 10) mit bis zu **32 Teilnehmern je Segment**. Die Topologie der einzelnen Bussegmente ist die **Linienstruktur**.

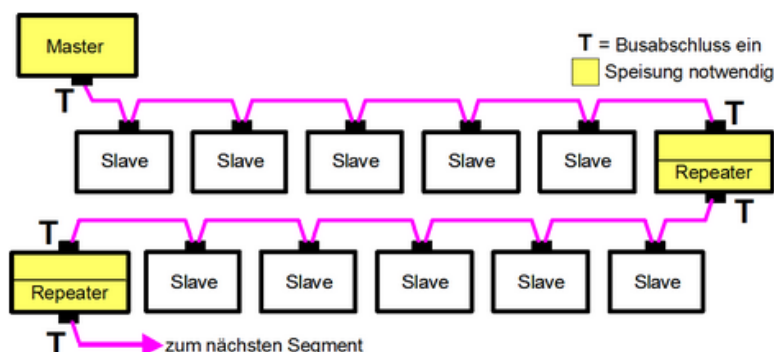


Abbildung 75 Profibus Bussegmente

- Maximal **12km Buslänge bei RS485 über alle Bussegmente** (max. 1200m je Segment) abhängig von der Übertragungsrate. Übertragungsgeschwindigkeiten von **9,6 Kbaud bis 12 Mbaud**.

Übertragungsrate in KBAud	9,6	19,2	93,75	187,5	500	1500	3000	6000	12000
Länge pro Segment in m	1200	1200	1200	1000	400	200	100	100	100
max. Länge in m	12000	12000	12000	10000	4000	2000	400	400	400
bei Anzahl Bus-segmente:	10	10	10	10	10	10	4	4	4

Abbildung 76 Profibus Buslängen abhängig von der Übertragungsrate

### 4.7.3 Gerätetypen bei Profibus-DP

#### DP- Master Klasse 1:

Hierbei handelt es sich um eine **zentrale Steuerung**, die in einem festgelegten Nachrichtenzyklus Daten mit den dezentralen Stationen (DP-Slaves) austauscht. Konkret werden folgende Master-Slave Anwendungsfunktionen unterstützt:

- Parametrierung und Konfigurierung der DP-Slaves
- Zyklischer Nutzdatenaustausch (Lesen der Eingangs- und Schreiben der Ausgangswerte)
- Erfassen von Diagnoseinformationen der DP-Slaves

Diese Funktionen werden von der Anwendungsschicht (ISO/OSI-Schicht 7) des DP-Master selbständig abgewickelt. Typische Geräte sind speicherprogrammierbare Steuerungen (SPS), Numerische- (NC) und Robotersteuerungen (RC).

#### DP-Master Klasse 2:

Geräte dieses Typs sind **Programmier-, Projektierungs- oder Diagnosegeräte**. Sie werden bei der Inbetriebnahme eingesetzt, um die Konfiguration des DP-Systems vorzunehmen.

Ein DP-Master Klasse 2 beherrscht folgende Funktionen zusätzlich zu den Klasse 1 Funktionen:

- Zuordnung der Teilnehmeradressen der DP-Slaves am Bus (0..126 für Slaves)
- Zuordnung der E/A Adressen im Prozessabbild der Steuerung
- Rücklesen der DP-Slave Konfiguration

#### DP-Slave:

Als DP-Slave wird ein Peripheriegerät, das E/A Daten verarbeitet bezeichnet. Typische DP-Slaves sind **Busklemmen** mit E/A-Modulen, **Antriebsregler** und auch **Steuerungen**, die als Slave betrieben werden.

### 4.7.4 Systemkonfiguration

Mit Profibus-DP lassen sich Single- und Multi-Master-Systeme realisieren. **Bei Single-Master-Systemen** ist in der Betriebsphase des Bussystems nur ein Master am Bus aktiv. **Die SPS ist die**

**zentrale Steuerungskomponente.** Es liegt ein reines **Master-Slave Zugriffsverfahren** vor. Mit dieser Systemkonfiguration wird die **kürzeste Buszykluszeit** erreicht.

Im **Multi-Master-Betrieb** befinden sich an einem Bus mehrere Master. Sie können entweder voneinander unabhängige Subsysteme - bestehend aus je einem Master und den zugehörigen Slaves - bilden oder als zusätzliche Projektierungs- und Diagnosegeräte fungieren. **Die Eingangs- und Ausgangsabbilder der Slaves können von allen Mastern gelesen werden.** Das Beschreiben der Ausgänge ist jedoch jeweils nur von einem Master möglich.

#### 4.7.5 Datenübertragung und Übertragungssicherheit

##### **Zeitverhalten von Profibus-DP:**

Die Architektur des Profibus-DP und die daraus resultierenden niedrigen Buszykluszeiten erlauben den Einsatz für **zeitkritische Anwendungen**. Bei Profibus-DP mit **12 MBit/s** ist die Verzögerungszeit des Feldbusses auch bei einer größeren Anzahl von Slaves und E/A-Daten klein.

##### **Schutzmechanismen:**

Der **Master** führt je zugeordnetem Slave eine **Zeitüberwachung des Nutzdatentransfers** durch. Dabei wird überprüft, ob innerhalb einer bestimmten festgelegten Zeitspanne, **mindestens einmal ein ordnungsgemäßer Nutzdatentransfer** mit dem Slave stattfand. Im Fehlerfall wird das Anwenderprogramm informiert. Ist die automatische Fehlerbehandlung freigegeben, verlässt der Master den Normalbetrieb und **schaltet die Ausgänge aller Slaves in den sicheren Zustand**.

Der **Slave** führt zur Erkennung von Fehlern des Masters oder der Übertragungsstrecke die **Ansprechüberwachung** durch. Findet innerhalb eines festgelegten Intervalls kein Datenverkehr mit dem zugeordneten DP-Master statt, dann schaltet der DP-Slave die Ausgänge selbständig in den sicheren Zustand.

Zusätzlich ist bei Multi-Master-Betrieb ein **Zugriffsschutz für die Ausgänge** der Slaves vorhanden, der sicherstellt, dass der **direkte Zugriff nur vom berechtigten Master** ausgeht. Die Slaves stellen für alle anderen Master ein Abbild der Eingänge und Ausgänge zur Verfügung, das von jedem Master, auch ohne Zugriffsberechtigung, gelesen werden kann.

#### 4.7.6 Inbetriebnahme des Profibus DP

Die **Adressen** der einzelnen Teilnehmer können zwischen **0 und 126** frei vergeben werden. Jedem Teilnehmer muss bei der Projektierung eine eindeutige Adresse zugewiesen werden.

Die Eigenschaften eines Profibus-Teilnehmers sind in so genannten **GSD-Dateien** (Geräte spezifische Daten) mit fix vorgegebenem Aufbau beschrieben. Die Dateien enthalten alle Funktionen und Parameter und ermöglichen eine **steuerungsunabhängige Konfiguration**. Die GSD-Dateien aller Hersteller werden von der **Profibus Nutzer Organisation PNO** verwaltet.

## 5 Industrial Ethernet

### 5.1 Allgemeines

Industrial Ethernet ist der Überbegriff für alle Bestrebungen, den **Ethernet-Standard für die Vernetzung von Geräten der industriellen Fertigungstechnik** nutzbar zu machen. Bei Ethernet handelt es sich um die am weitesten verbreitete Technologie für Local Area Networks (LANs). Ethernet ermöglicht eine **einheitliche Netzwerkinfrastruktur über alle Ebenen der Automatisierungspyramide** von der Sensor/Aktor- bis zur Zellebene. Ethernet-basierte Bussysteme bieten im Vergleich zu konventionellen Feldbussystemen mehr Flexibilität im Aufbau und der Erweiterung.

Vorteile der Ethernet-Technologie:

- anerkannte und **zukunftsichere Technologie**
- 10- bis 100-fach höherer **Datendurchsatz** als bei seriellen Feldbussen
- **großer Adressbereich** mit einer fast unbegrenzten Teilnehmeranzahl
- **durchgängige Übertragungsmedien** und **-protokolle** von der Office- bis zur Feldebene
- Kombination **verschiedener Übertragungsmedien** möglich (Kabel, Lichtwellenleiter, Funk)
- Möglichkeit zur **weltweiten Vernetzung** für Diagnose und Wartung

#### 5.1.1 Verbindung der Office-Welt mit der Steuerungstechnik

Ein wichtiges Argument für den Einsatz von Industrial Ethernet ist die Möglichkeit zur **Nutzung von IT-Standards** in den Automatisierungsgeräten. Als Beispiel ist die **Integration eines Webserver**s in die Feldgeräte zu nennen. Ein integrierter Webserver ermöglicht es, sich mittels eines Standard-Webrowsers einen schnellen Überblick über Status und Diagnosemeldungen eines Feldgerätes zu verschaffen.

Feldgeräte mit integrierten IT-Funktionen bieten häufig auch die Möglichkeit **e-Mails** oder **SMS** im Fehlerfall direkt an das zuständige Wartungspersonal zu versenden.

#### 5.1.2 Verkabelungstechnik

Die gemeinsame Basis aller Industrial-Ethernet-Systeme bildet die **Ethernet-Übertragungstechnik**. Fast alle industriellen Systeme nutzen die **100-Mbit Fast-Ethernet-Technologie**.

Bei den meisten Systemen können die aus der Office-Welt bekannten **RJ45-Stecker** in Kombination mit **CAT5-Ethernet Kabeln** eingesetzt werden. Für optimale Industrietauglichkeit steht darüber hinaus systemspezifisch eine Vielzahl verschiedener Spezialstecker und Kabel zur Verfügung.

Während sich in der Office-Welt die sternförmige Verkabelung auf der Basis von externen Switches durchgesetzt hat, tendiert man in der Automatisierungstechnik dazu, die Industrial-Ethernet-Netzwerke in der von den Feldbussen her bekannten **Linien- oder Bustopologie** aufzubauen. Dies erfordert **Feldgeräte mit integrierten 2-Port-Echtzeit-Switches** und ermöglicht damit die Installation optimal an die Anlagentopologie anzupassen.

Werden die Netzwerke in Linientopologie aufgebaut, so ist es aus Gründen der Ausfallsicherheit wichtig, Redundanzfunktionen zu integrieren. Ohne **Redundanzfunktionen** würde bereits das einfache Ausschalten eines Teilnehmers in der Linie zu einer Unterbrechung des Datenverkehrs im Netzwerk führen.

### 5.1.3 Echtzeitverhalten

Ein echtzeitfähiges Bussystem muss Prozessdaten **verlässlich** und mit **kalkulierbaren Übertragungszeiten** transportieren. Dabei ist nicht nur die reine Übertragungsgeschwindigkeit wichtig, vielmehr sind die Verzögerungen durch parallelen zeitunkritischen Datenverkehr (**Jitter**) sowie die Reproduzierbarkeit der Übertragungszeiten (**Taktsynchronität**) zu berücksichtigen.

Als **Jitter** (engl. „Fluktuation“ oder „Schwankung“) bezeichnet man ein Taktzittern bei der Übertragung von Digitalsignalen, eine leichte Genauigkeitsschwankung im Übertragungstakt (Clock). In der Netzwerktechnik wird mit Jitter außerdem die Streuung der Laufzeit von Datenpaketen bezeichnet.

Im Gegensatz zu den recht einheitlichen Kommunikationsanforderungen im IT-Bereich, sind die Anforderungen an die Echtzeitfähigkeit stark vom Anwendungsbereich abhängig. Üblicherweise unterscheidet man heute **drei Anforderungskategorien**, die durch typische **Aktualisierungszeiten der Prozessdaten** beschrieben werden können:

	Kategorie A	Kategorie B	Kategorie C
Echtzeitanforderungen	gering	mittel	hoch
Aktualisierungsraten	<b>100ms</b>	<b>10ms</b>	<b>1ms</b>
Anwendungen	Steuerungsebene	Feldebene	Motion-Control

Abbildung 77 Echtzeit-Anforderungskategorien

**Standard-Ethernet** ist nicht echtzeitfähig. Der fehlende Determinismus ist darin begründet, dass beim traditionellen **CSMA/CD-Verfahren** die **Kollisionen** von Nachrichten zwar erkannt, nicht aber vermieden werden können. Dies resultiert in erheblichen Zeitschwankungen bei der Nachrichtenübertragung.

**Switched-Ethernet** vermeidet Kollisionen durch die im Switch aufgebauten Punkt-zu-Punkt-Verbindungen und ermöglicht Duplex-Übertragung. Allerdings entstehen beim Einsatz von „normalen“ Switches zusätzliche **Latenzzeiten** und nichtdeterministische Verzögerungen in Überlastsituationen.

#### Maßnahmen zur Erreichung der Echtzeitfähigkeit:

Es gibt verschiedene, sehr unterschiedliche Maßnahmen, um Ethernet echtzeitfähig zu machen. Je nach Anforderungsklasse ist die Echtzeitfähigkeit mit Software und bei höheren Anforderungen mit Hardware-Unterstützung erreichbar.

### 1. Vermeidung von Kollisionen am Bus

- Einsatz von **Switched-Ethernet** mit Punkt-zu-Punkt Verbindungen
- **Master / Slave** Buszugriffsverfahrens
- Zuweisen vom **Zeitfenster** für jeden Busteilnehmer (TDMA Buszugriffsverfahren)

### 2. Beschleunigung der Verarbeitung

- Aufwändige **Protokollschichten überspringen** (z.B. Schicht 3 und 4)
- Verbesserung der Protokolleffizienz durch **Summenteleggramme** für mehrere Teilnehmer
- Protokollauswertung mit **Hardware-Unterstützung** (spezielle Ethernet-Controller)

### 3. Vermeiden von Verzögerungen bei hohem Datenverkehr

- Priorisierung von Telegrammen in Switches (Vorreihung priorisierter Telegramme)
- Einsatz von Switches die Telegramme im Durchlauf verarbeiten

### Beispiel Übertragung in Zeitschlitzten:

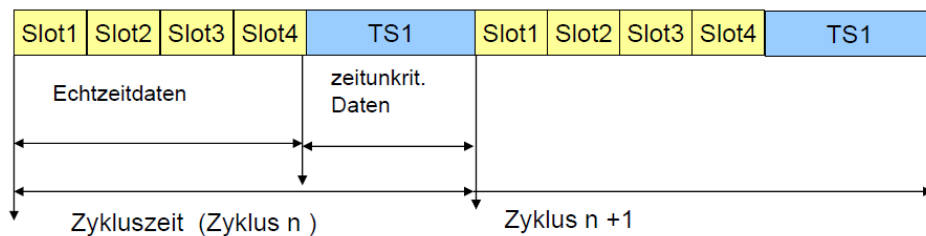


Abbildung 78 Echtzeitdatenübertragung mit Zeitschlitzten

### Lösungskonzepte:

Am Markt befindliche Real-Time Ethernet Lösungskonzepte unterscheiden sich in ihrer Leistungsfähigkeit und dem Grad der Konformität zum Standard-Ethernet.

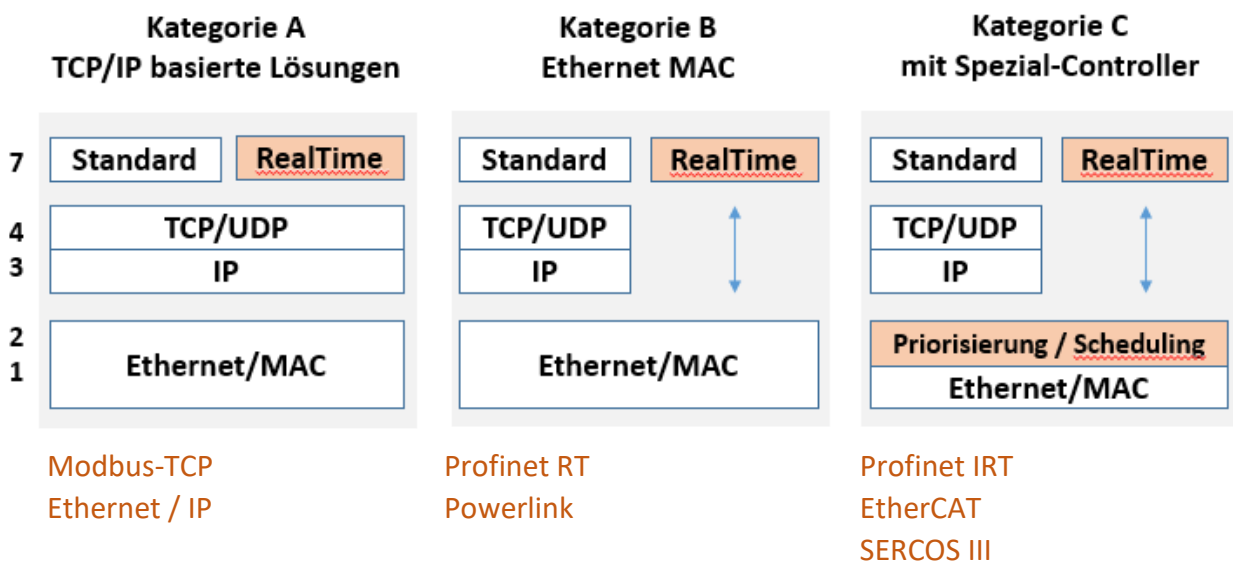


Abbildung 79 Real-Time Ethernet Lösungskonzepte

## 5.2 Auswahlkriterien für Industrial-Ethernet Bussysteme

Die Diskussion bei der Bewertung von Ethernet-Bussystemen dreht sich häufig um technische Leistungsparameter. Neben den technischen Eigenschaften, gibt es aber weitere Kriterien, die bei der Auswahl unbedingt berücksichtigt werden sollten und in folgender Übersicht dargestellt sind:

### Auswahlkriterien:

- **Engineering** - z.B. werden Geräte unterschiedlicher Hersteller unterstützt?
- **Service und Diagnose** - Gerätetausch und Diagnosewerkzeuge?
- **Feldbusintegration** - Verfügbarkeit von Gateways?
- **Marktakzeptanz** - Verbreitung?
- **Preis** - Preis / Leistungsverhältnis

### Engineering:

Das Engineering umfasst die **Projektierung der Geräte** und deren **Kommunikationsbeziehungen**. Engineering-Tools unterscheiden sich vor allem dadurch, wie nahtlos die Projektierungsaufgaben durchgeführt werden können.

### Service und Diagnose:

Aus Sicht der Instandhaltung ist ein **einfacher Gerätetausch** ein wichtiges Auswahlkriterium. Die Teilnehmer im Ethernet-Netz werden eindeutig durch ihre MAC-Adressen gekennzeichnet. Dies gilt uneingeschränkt für alle Echtzeit-Ethernet-Lösungen.

Während bei den Feldbussystemen häufig die Prüfung der Kabel und Signale auf physikalischer Ebene (ISO/OSI-Schicht 1) im Vordergrund steht, dominiert bei Ethernet-Systemen die Diagnose **auf funktionaler Ebene** (Schicht 2 und 7). Entsprechende **Diagnose-Tools** sind erforderlich.

### Feldbusintegration:

In vielen Anwendungsfällen stellt sich die Anforderung nach einer **schrittweisen Migration** von bestehenden Feldbus-Lösungen zu Ethernet-basierten Lösungen. Bei der Auswahl ist daher einerseits die **Verfügbarkeit von Gateways**, sowie deren möglichst nahtlose Integration in das Engineering zu bewerten.

### Marktakzeptanz:

Ethernet-basierte Bussysteme können daher nur dann am Markt bestehen, wenn auch das Geräteangebot ausreichend ist. Dies wird in der Regel durch unterschiedliche Hersteller abgedeckt. Die **Herstellervielfalt** ist dabei auch ein Indikator für die Verbreitung und Marktakzeptanz einer Lösung.



## 5.4 Modbus-TCP

Das Modbus-Protokoll ist seit seiner Entwicklung im Jahr **1979 ein De-facto-Standard** für die industrielle Kommunikation. Es wurde von der Firma Modicon (heute **Schneider Electric**) für den Datenverkehr mit ihren Steuerungen entwickelt.

Das Protokoll ist inzwischen mit drei verschiedenen Übertragungsschichten (ISO/OSI-Schicht 1 und 2) verfügbar:

- **Modbus-ASCII** - Übertragung mit ASCII Zeichen über RS485 oder RS232
- **Modbus-RTU** - Übertragung binär über RS485 oder RS232, Zeit-Synchronisation
- **Modbus-TCP** - Übertragung über Ethernet TCP/IP, *Well Known Port 502*

### 5.4.1 Einsatzbereiche

Modbus-TCP hat sich als ein **Standard** für die industrielle Kommunikation über Ethernet etabliert. Basis hierfür bildet eine stabile Spezifikation, verfügbare Basistechnologie und eine Vielzahl industrieller Seriengeräte.

Der praktische Vorteil des Modbus ist, dass sämtliche Modbus-Protokolle ein **gemeinsames, einfaches und gut dokumentiertes Anwendungsprotokoll** nutzen. Das Anwendungsprotokoll ist vollkommen in Software realisierbar und **hardware-unabhängig**.

### 5.4.2 Protokollaufbau

Das Modbus TCP/IP Protokoll ist ein **Client/Server-Protokoll** für den verbindungsorientierten gesicherten Austausch von Prozessdaten. Die Rollen des Clients und des Servers sind nicht fest zugeordnet. Jedes Gerät im Netzwerk kann beide Rollen spielen. Somit können Zugriffe auf Daten lesend oder schreibend flexibel den jeweiligen Aufgabenstellungen angepasst werden.

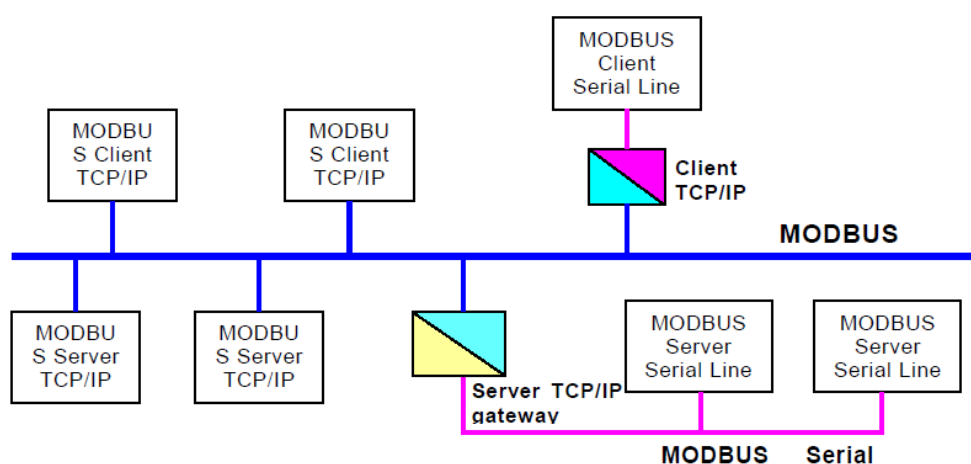


Abbildung 80 Modbus Beispiel

**Buszugriffsverfahren:**

Als Buszugriffsverfahren wird das Master/Slave Zugriffsverfahren eingesetzt. Die Adressierung der Slaves erfolgt mit dem TCP-Protokoll über IP-Adressen.

**Echtzeitverhalten:**

Modbus-TCP bietet „**weiche**“ **Echtzeit** (Echtzeitkategorie A). Durch das Master/Slave Zugriffsverfahren und bei Verwendung eines privaten IP-Netzes für die Slaves, lassen sich Kollisionen am Bus vermeiden und Zykluszeiten bis unter 10ms erreichen.

**Allgemeiner Telegrammaufbau:**

Das Modbus-Protokoll legt unabhängig von der Übertragungsschicht ein **einheitliches Prozessdatenpaket** fest (ISO/OSI-Schicht 7, **Protocol Data Unit = PDU**). Zusätzlich werden je nach Übertragungsschicht zusätzliche Adresse- und Sicherheitsanteile hinzugefügt (**Applikation Data Unit = ADU**).

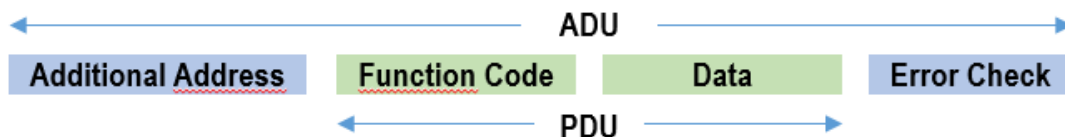


Abbildung 81 Allgemeine Modbus ADU

**Modbus-TCP Application Data Unit:**

Modbus-TCP/IP definiert einen speziellen **Modbus Application Protokoll Header MBAP**. Die Datensicherung wird vom Ethernet-Standard-Frame erledigt. Es sind keine spezifischen Modbus-Sicherungsmaßnahmen erforderlich.

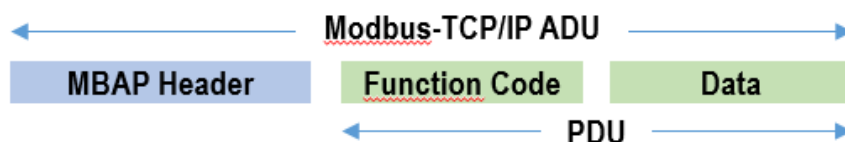


Abbildung 6 Modbus-TCP/IP ADU

**Aufbau der PDU:**

- Function Code:
- Lesen / Schreiben von digitalen E/A Signalen (Bits)
  - Lesen / Schreiben von 16 Bit Registerwerten (beliebige Variablen)
- Data:
- Startadresse der Daten von E/A Bits oder Registern
  - Anzahl der zu übertragenden Bits/Bytes
  - zu übertragene Daten

**PDU Beispiel:** Lesen der digitalen Eingänge 197 - 218:

Request		Response	
Field Name	(Hex)	Field Name	(Hex)
Function	02	Function	02
Starting Address Hi	00	Byte Count	03
Starting Address Lo	C4	Inputs Status 204-197	AC
Quantity of Inputs Hi	00	Inputs Status 212-205	DB
Quantity of Inputs Lo	16	Inputs Status 218-213	35

Die Nummerierung der E/A Bits und Register beginnt bei 1, die Adressierung bei 0!

### Datenmodell:

Das Datenmodell ist einfach strukturiert und unterscheidet 4 Grundtypen:

- **Discrete Inputs** (Bit-Eingänge)
- **Coils** (Bit-Ausgänge)
- **Input Register** (Wort-Eingangsdaten)
- **Holding Register** (Wort-Ausgangsdaten)

Aus der Namensgebung sind auf die Ursprünge des Modbus-Protokolls, den Anfängen der speicherprogrammierbaren Steuerungen, erkennbar.

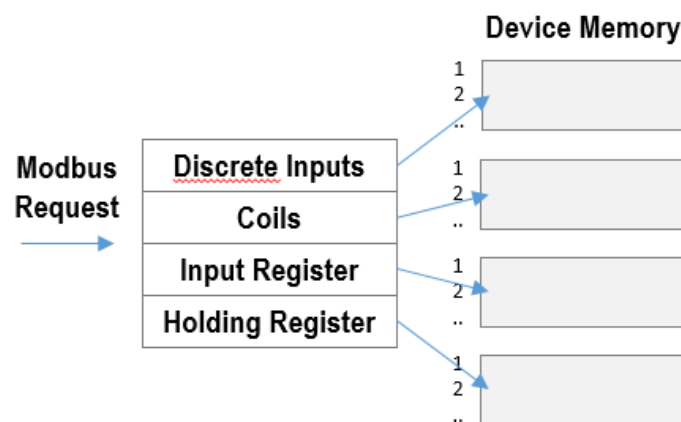


Abbildung 82 Modbus Datenmodell mit separaten Blöcken

## 5.5 Profinet

Die Profibus Nutzerorganisation hat den **Profibus-Standard auf Ethernet** portiert. Es wurde ein völlig neuer Standard mit dem Namen **PROcess Field EtherNET** geschaffen.

### 5.5.1 Einsatzbereiche (Kommunikationsprofile)

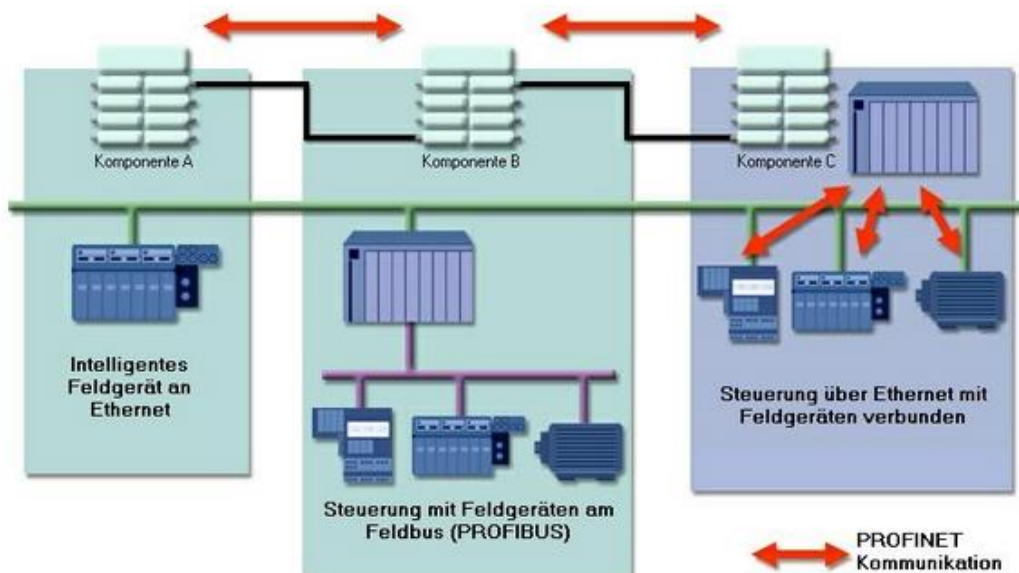


Abbildung 83 Profinet Kommunikation

#### Profinet-IO - Dezentrale Feldgeräte :

Profinet-IO beschreibt ein Gerätemodell für den **zyklischen Datenaustausch** zwischen Steuerungen und E/A-Geräten, das sich am Profibus orientiert und aus Steckplätzen (Slots) und Gruppen von E/A-Kanälen (Subslots) besteht.

#### Profinet-CBA - Verteilte Automatisierung:

Profinet-CBA (Component Based Automation) kommt in verteilten Automatisierungsanlagen für zyklischen und azyklischen Datenaustausch zwischen Steuerungen zur Anwendung. Das Komponenten-Modell beschreibt autonom agierende Teileinheiten von Maschinen und Anlagen als technologische Module.

### 5.5.2 Gerätetypen

Profinet-IO unterscheidet, wie Profibus-DP auch, zwischen verschiedenen Gerätetypen, wobei ein Gerät auch mehrere Rollen beinhalten kann.

#### IO-Controller:

Der IO-Controller hat die Kontrolle über den auf ein oder mehrere Feldgeräte verteilten Prozess. Es führt ein Steuerungsprogramm aus und verarbeitet dabei die Prozessdaten.

**IO-Supervisor:**

Der IO-Supervisor ist eine Engineering-Station in einer Anlage, die für Inbetriebnahmezwecke einen temporären Zugriff auf die Feldgeräte besitzen kann.

**IO-Device:**

Ein IO-Device ist ein dezentral angeschlossenes, prozessnahes Feldgerät. Es erwartet die Konfiguration von einem IO-Controller/Supervisor und überträgt seine Prozessdaten zyklisch an den IO-Controller.

Profibus	Profinet
DP Master System	Profinet-IO-System
DP Master Klasse 1	Profinet-IO-Controller
DP Master Klasse 2	Profinet-IO-Supervisor
DP Slave	Profinet-IO-Device

Abbildung 84 Geräteklassen Vergleich mit Profibus

### 5.5.3 Systemkonfiguration

Ein minimales Profinet-IO-System besteht aus **mindestens einem IO-Controller** der ein oder mehrere **IO-Devices** kontrolliert. Zusätzlich können optional ein oder mehrere **IO-Supervisoren für das Engineering** der IO-Devices temporär zugeschaltet werden.

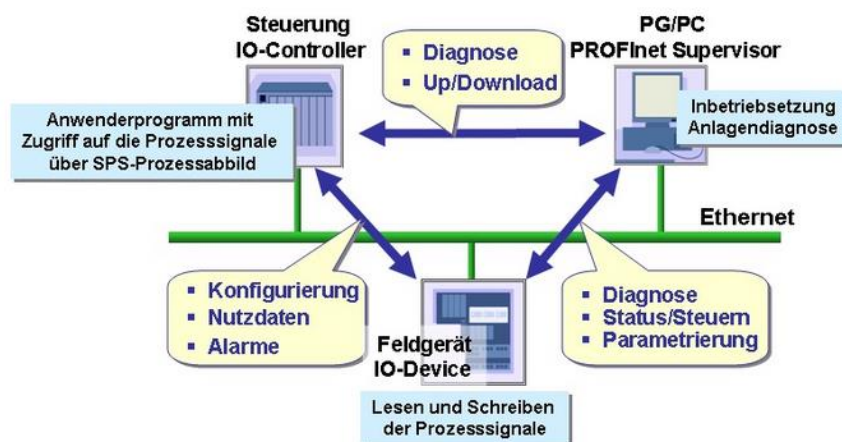


Abbildung 85 Profinet Systemkonfiguration

Sind zwei IO-Controller im selben Netzwerk, können sie sich Eingangssignale eines IO-Devices als **Shared-Inputs** teilen, in dem sie auf das Submodule des IO-Device **lesend** zugreifen. Ebenso kann ein ganzes IO-Device als **Shared-Device** geteilt werden, in dem Submodule eines IO-Device unterschiedlichen IO-Controllern zugeordnet werden.

Jeder **IO-Controller** kann neben der Aufgabe seine IO-Devices zu kontrollieren, **selbst als IO-Device** für eine Partnersteuerung konfiguriert werden.

### 5.5.4 Kommunikationsprotokolle

Profinet beinhaltet **mehrere Kommunikationsprotokolle** für **unterschiedliche Leistungsstufen**:

- **IP-Protokoll** für zeitunkritische Datenübertragung
- **Real Time - RT Protokoll** für weiche Echtzeit
- **Isochrones Real Time IRT-Protokoll** für harte Echtzeit

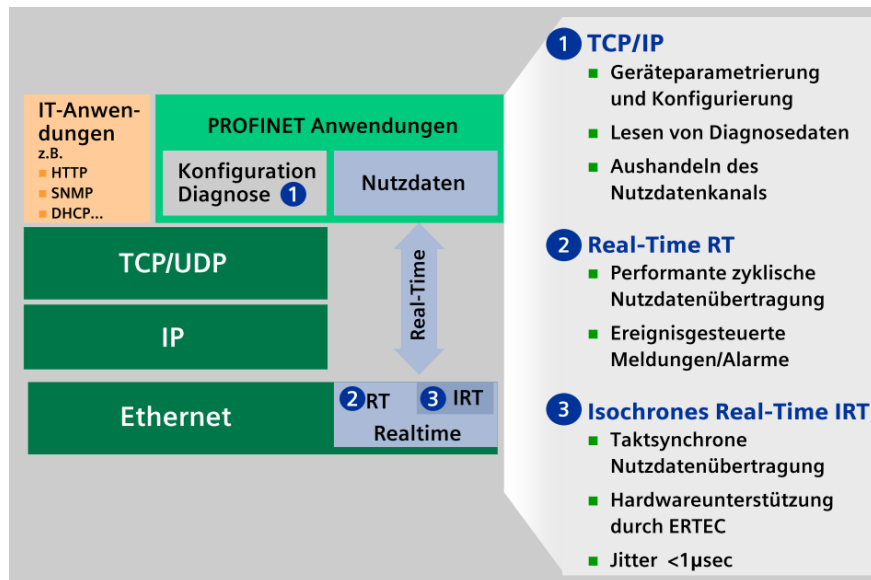


Abbildung 86 Profinet Kommunikationsverfahren

#### IP-Protokoll:

Die **nicht zeitkritische Übertragung von Parametern und Konfigurationsdaten** und erfolgt bei Profinet über den Standardkanal auf Basis des IP-Protokolls. Damit sind die Voraussetzungen für die Anbindung der Automatisierungsebene zu übergeordneten Netzen geschaffen.

#### Profinet-RT:

Profinet-RT ermöglicht vergleichbar mit Profibus die Übertragung von **zeitkritischen Prozessdaten**. Bei dieser auf **Duchsatz** ausgelegten Kommunikation wird der RT-Kanal **ohne Verwendung des IP-Protokolls** (Schicht 3, 4) betrieben. Zusätzlich wird mit Hilfe von speziellen Switches das **Vorbeischieben der Nutzdaten** an den Standard IP-Paketen ermöglicht. Profinet-RT ist kompatibel zum Ethernet-Standard.

#### Profinet\_IRT:

Profinet-IRT wird bei anspruchsvollen Anwendungen mit **kleinen Zykluszeiten** und **sehr kurzen Reaktionszeiten**, wie z.B. im Motion-Control Bereich gefordert, eingesetzt. Damit der notwendige Determinismus erreicht werden kann, sind folgende Maßnahmen erforderlich:

- **Hardwareunterstützung** zur Telegrammauswertung (spezieller Ethernet-Controller)
- Buszugriff mit **Zeitschlitzverfahren**
- **Spezielle Switch-Technologie**

Jeder Busteilnehmer überträgt seine Echtzeitdaten in einem ihm **zugeordnetem Zeitschlitz**. Dazu ist in der Hochlaufphase eine exakte **Laufzeitbestimmung und Zeitsynchronisation aller Teilnehmer und Switches** erforderlich.

Dabei wird auch für jeden Switch festgelegt, wann welches Telegramm empfangen und zu welchem Teilnehmer weitergeleitet werden muss. Mit dieser Information kann das **Cut-Through Verfahren** angewandt werden, das einem Switch erlaubt seine "Schalter" bereits vor Empfangen des gesamten Telegramms auf den richtigen Port umlegen und das Telegramm beinahe ohne Verzögerung durchzuleiten.

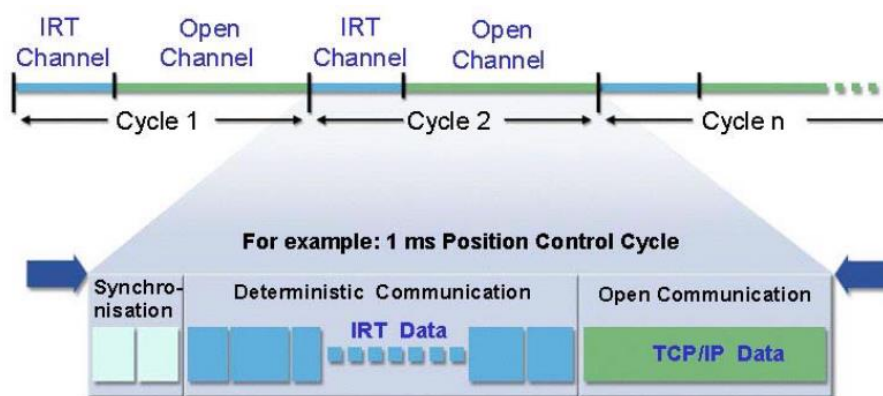


Abbildung 87 Profinet Scheduling

### Adressierung der E/A-Geräte:

Für den **Echtzeitbetrieb** werden **MAC-Adressen**, für den Verbindungsaufbau und die azyklischen Dienste wird das IP-Protokoll verwendet. Jedes Profinet-Gerät erhält bei der Projektierung einen **symbolischen Namen** und eine **IP-Adresse**.

### Konfiguration eines Profinet-Netzwerkes:

Die technischen Eigenschaften der Feldgeräte werden mit **GSDML General Station Description**-Dateien auf XML-Basis beschrieben. Jedes Feldgerät besitzt eine individuelle GSDML-Datei, die vom jeweiligen Gerätehersteller zur Verfügung gestellt werden muss.

### PROFIsafe Protokoll:

PROFIsafe ist ein **zertifiziertes Sicherheitsprofil** für Profibus und Profinet. Es legt fest, wie sicherheitsgerichtete Geräte miteinander kommunizieren. Das PROFIsafe-Protokoll realisiert die sichere Kommunikation über das unsichere Profinet-Bussystem.

### Zertifizierung:

Zur Gewährleistung einer fehlerfreien Kommunikation zwischen Profinet-Geräten von unterschiedlichen Herstellern, ist eine normkonforme Implementierung der Kommunikationsprotokolle erforderlich. Vor Vertrieb von Profinet-Produkten ist ein Zertifizierungsverfahren erforderlich.

Link zur Profinet- und Profibus-Nutzerorganisation: <https://www.profibus.com/>

## 5.6 EtherCAT

EtherCAT steht für **E**thernet for **C**ontrol **A**utomation **T**echnology und wurde ursprünglich von der Fa. Beckhoff entwickelt. Die Schwerpunkte der Entwicklung von EtherCAT lagen auf kurzen **Zykluszeiten  $\leq 100 \mu\text{s}$**  und **niedrigem Jitter** für exakte Synchronisierung  $\leq 1 \mu\text{s}$  und **niedrigen Hardwarekosten**.

2003 wurde die EtherCAT Technology Group gegründet. Diese Vereinigung bestehend aus Endanwendern, Maschinen- und Steuerungsherstellern ist bezüglich der Anzahl ihrer Mitglieder heute die größte Industrial Ethernet Nutzerorganisation weltweit.

EtherCAT ist seit 2005 IEC-Norm.

### 5.6.1 Funktionsprinzip

Was unterscheidet EtherCAT von anderen Echtzeit-Ethernet-Ansätzen? Bei bisherigen Lösungen wird z. B. das Zugriffsverfahren CSMA/CD durch überlagerte Protokollschichten außer Kraft gesetzt und durch ein Zeitscheibenverfahren oder durch Polling ersetzt; andere Vorschläge sehen spezielle Switches vor, die Ethernet-Pakete zeitlich präzise kontrolliert verteilen.

Der vom EtherCAT-Master versendete Standard Ethernet Frame gemäß IEEE 802.3 wird nicht wie bei anderen Industrial Ethernet Lösungen in jeder Anschaltung zunächst empfangen, interpretiert und bearbeitet. Die EtherCAT-Anschaltungen bearbeiten die für sie bestimmten Daten, **während das Telegramm das Gerät durchläuft**. Dabei wird ein Rahmen nicht vollständig empfangen, bevor er verarbeitet wird, sondern die Bearbeitung wird so früh wie möglich begonnen. Das Versenden erfolgt ebenso mit einem minimalen Versatz von wenigen Bitzeiten.

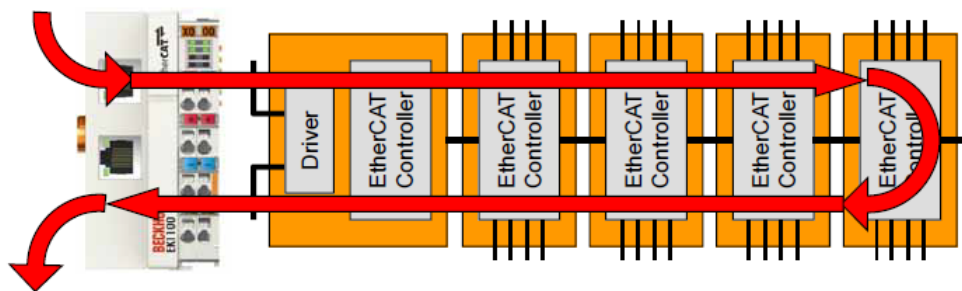


Abbildung 88 Telegrammdurchlauf bei EtherCAT-Klemmen

Das **Ethernet-Protokoll gemäß IEEE 802.3** bleibt auch bei modularen Baugruppen bis ins Endgerät - z. B. die elektronische Reihenklemme - erhalten, ein unterlagerter Bus entfällt.

Da ein Ethernet-Frame sowohl in Sende- als auch in Empfangsrichtung die Daten vieler Teilnehmer enthält, steigt die **Nutzdatenrate auf über 90 %** an. Dabei werden die Voll-Duplex Eigenschaften von **Fast-Ethernet** vollständig ausgenutzt, so dass effektive Datenraten von über 100 MBit/s ( $> 90\%$  von  $2 \times 100 \text{ MBit/s}$ ) erreichbar sind.



### Prozessdatenbearbeitung im Slave:

Die Daten werden von jedem Slave im Durchlauf entnommen und eingefügt. Dabei können die Daten im Telegramm schon so angeordnet werden, dass kein zusätzliches Umsortieren von z.B. PLC-, NC-Daten mehr erforderlich ist (Mapping).

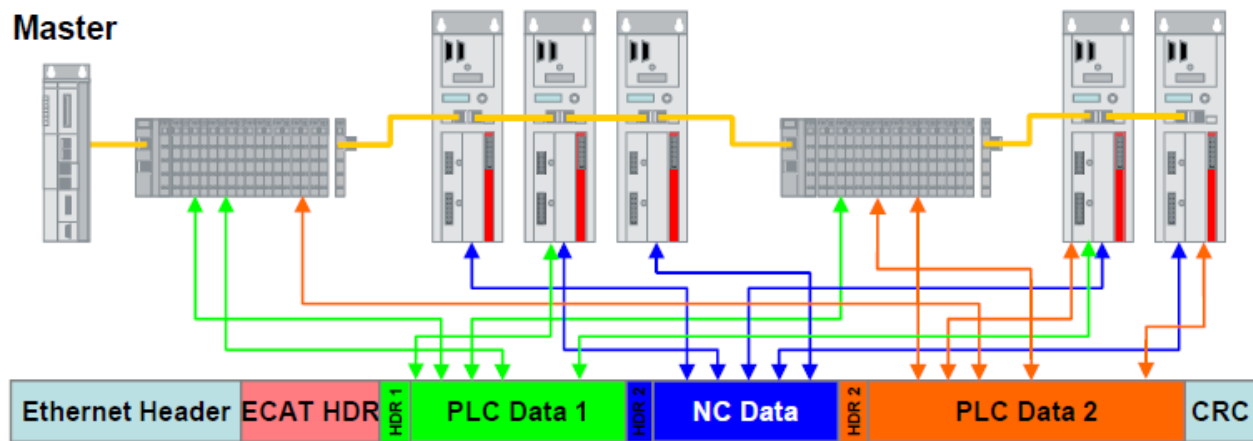


Abbildung 89 Prozessdatenbearbeitung bei EtherCAT

### EtherCAT Slave-Controller:

Möglich wird die beschriebene Art der Bearbeitung des durchlaufenden Telegramms, durch den Einsatz **hochintegrierter EtherCAT-Slave Controller** mit oder ohne integriertem Mikroprozessor, die die Ethernet-Telegramme in Hardware bearbeiten und weiterleiten.

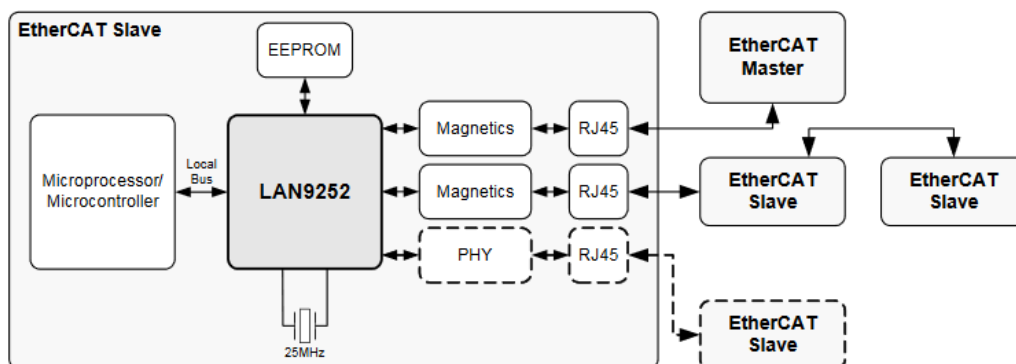


Abbildung 90 Beispiel EtherCAT Slave Controller

### Performance:

EtherCAT erreicht eine hohe Netzwerk-Performance. Dank Hardware-Integration im Slave und DMA-Zugriff der Netzwerkkarte im Master erfolgt fast die gesamte Protokollbearbeitung in Hardware und ist damit unabhängig von einer Laufzeit zur Bearbeitung eines Protokoll-Stacks.

Beispiele möglicher Update-Zeiten:

- 30µs für 1000 Digital-I/Os
- 100µs für 100 Servoachsen

## 5.6.2 Topologie

Mit EtherCAT werden nahezu alle Topologien wie **Stern**, **Baum**, **Ring**, **Linie** unterstützt. Die Übertragung erfolgt mit Fast-Ethernet **100BASE-TX** Standard bis 100m zwischen 2 Geräten oder **100BASE-FX** mit Lichtwellenleiter für größere Entfernungen.

Der Zugriff auf die Slaves mit normalen Ethernet-Frames, z.B. zum Zugriff auf einen Webserver, kann von außen über den Master oder spezielle **EtherCAT-Switch-Ports** erfolgen. Die Ethernet-Frames werden durch das EtherCAT-Protokoll getunnelt, so wie es auch bei anderen Internet Protokollen üblich ist.

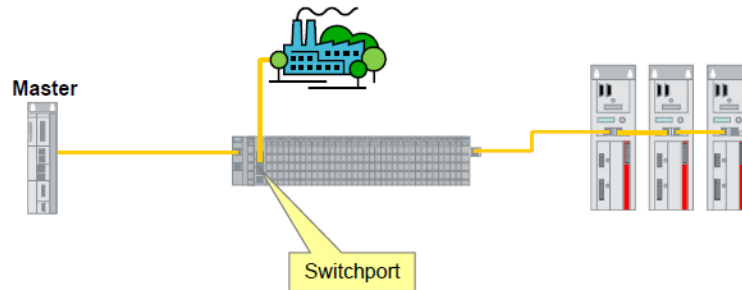


Abbildung 91 Vertikale Integration mittels Switch-Port

### Redundanter Betrieb:

Jedes Slave Gerät mit mindestens 2 Ports unterstützt automatisch den redundanten Betrieb. Da jeder Slave einen offenen Port automatisch schließt, läuft die Kommunikation beim Abziehen eines Teilsegmentes für die verbleibenden Slaves weiter.

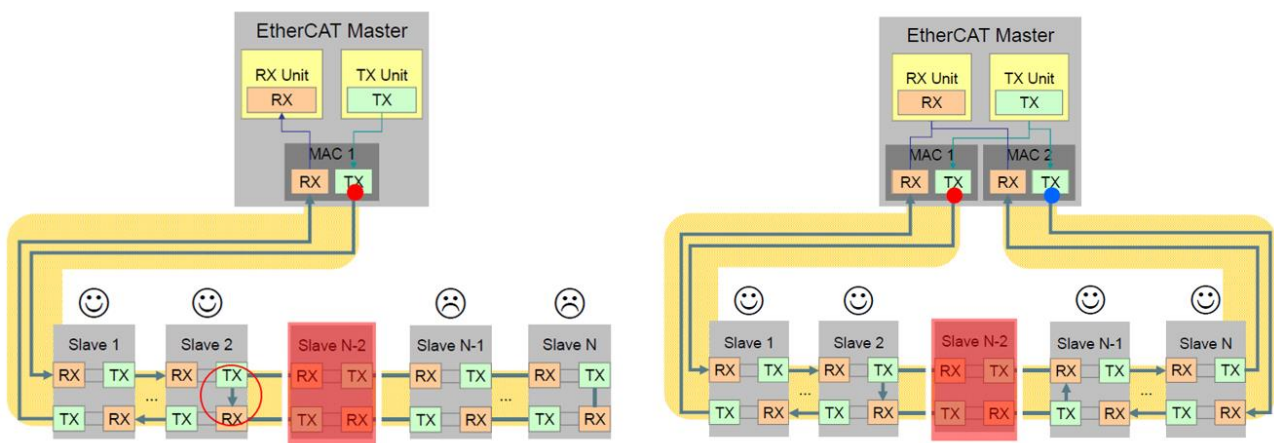


Abbildung 92 EtherCAT Teilnehmerausfall

Mit einem **zweiten Port am Master** und einem zusätzlichem Ethernet-Kabel vom letzten Slave an diesen Port kann eine **Ringstruktur** aufgebaut werden. Bei einer Unterbrechung dieses Rings durch ein defektes oder abgezogenes Kabel oder durch einen defekten Teilnehmer, können beide Teilsegmente weiterhin vom Master erreicht werden.

### 5.6.3 Kommunikationsprotokolle

#### Zyklische Prozessdaten:

Die zyklischen Prozessdaten für die Echtzeitanwendung werden direkt im Datenbereich eines Ethernet-Frames transportiert. Dadurch entfallen Laufzeiten in Software-Stacks. Die Telegramme werden nicht quittiert.

#### Azyklische Konfigurations- und Diagnosedaten:

Die Übertragung dieser Daten erfolgt mit einem azyklischen Mailbox-Protokoll. Ähnlich wie bei den CANopen-SDOs wird der Empfang jedes Telegramms quittiert.

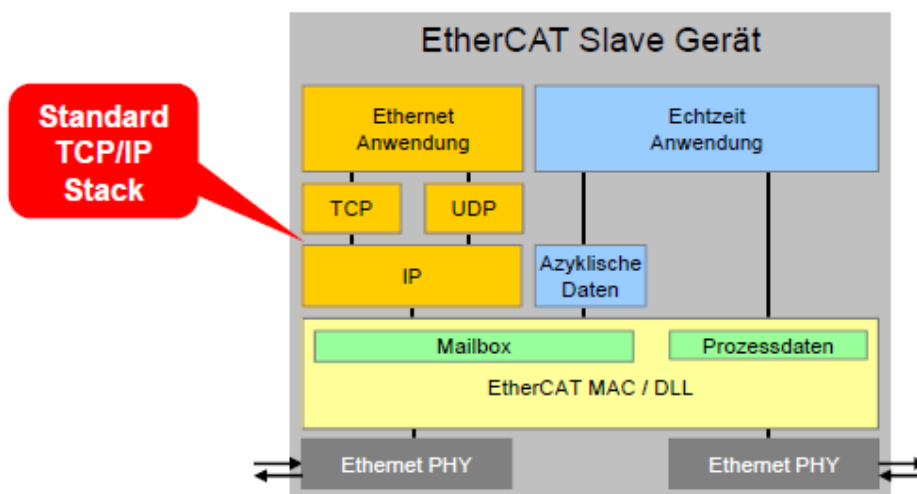


Abbildung 93 EtherCAT Telegrammverarbeitung

### 5.6.4 Geräteprofile

EtherCAT definiert keine eigenen Geräteprofile zur Konfiguration der Teilnehmer. Stattdessen werden **Schnittstellen für bestehende Geräteprofile** angeboten.

#### Protokoll CAN over EtherCAT (COE-Protokoll):

CANopen Geräte- und Applikationsprofile sind für eine Vielfalt von Geräteklassen festgelegt. Um diese nutzen zu können, stellt EtherCAT mit dem Protokoll COE mit CANopen vergleichbare Kommunikationsmechanismen bereit: **Objektverzeichnis, PDOs, SDOs, Netzwerkmanagement**. Mit COE kann EtherCAT auf Geräten, die zuvor mit CANopen Schnittstellen ausgestattet waren, mit überschaubarem Aufwand implementiert werden.

### Sicherheitsprotokoll Safety over EtherCAT (FSOE-Protokoll):

FSOE ist ein international standardisiertes Protokoll zur **Übertragung von sicherheitsrelevanten Daten**.

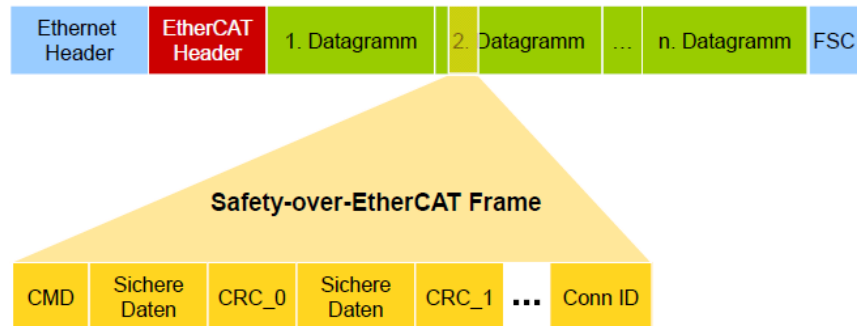


Abbildung 94 SoE Frame

Das Transportmedium wird bei FSOE als „**Black Channel**“ betrachtet und daher nicht in die Sicherheitsbetrachtung einbezogen. Die Safety-Frames enthalten die sicheren Prozessdaten und die erforderliche Datensicherung. Zur Verarbeitung der Safety-Frames sind sicherheitsgerichtete Controller und Ein- und Ausgabegeräte (Sicherheitsklemmen) erforderlich.

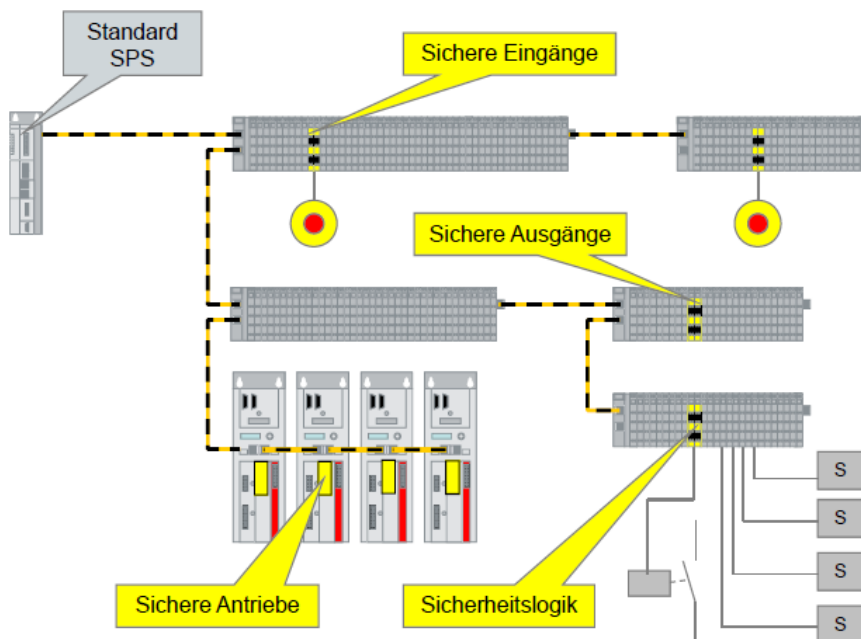


Abbildung 95 Dezentrale EtherCAT Sicherheitslogik

Link zur EtherCAT Nutzerorganisation: <http://www.ethercat.de>

## 5.7 OPC Unified Architecture

**OPC Unified Architecture**, kurz OPC UA, ist ein **industrielles Kommunikationsprotokoll**. Die Spezifikation des Protokolls wird von der **OPC-Foundation** gepflegt. Mittlerweile sind über 450 Unternehmen Mitglieder der **OPC-Foundation**. Homepage: <https://opcfoundation.org>

OPC UA schlägt die **Brücke zwischen der IT-Welt und der Produktion**. OPC UA basiert auf dem Client/Server-Prinzip und ermöglicht die **durchgängige Kommunikation über alle Schichten** der Automatisierungspyramide von der Sensor/Aktor- bis zur Leitebene.

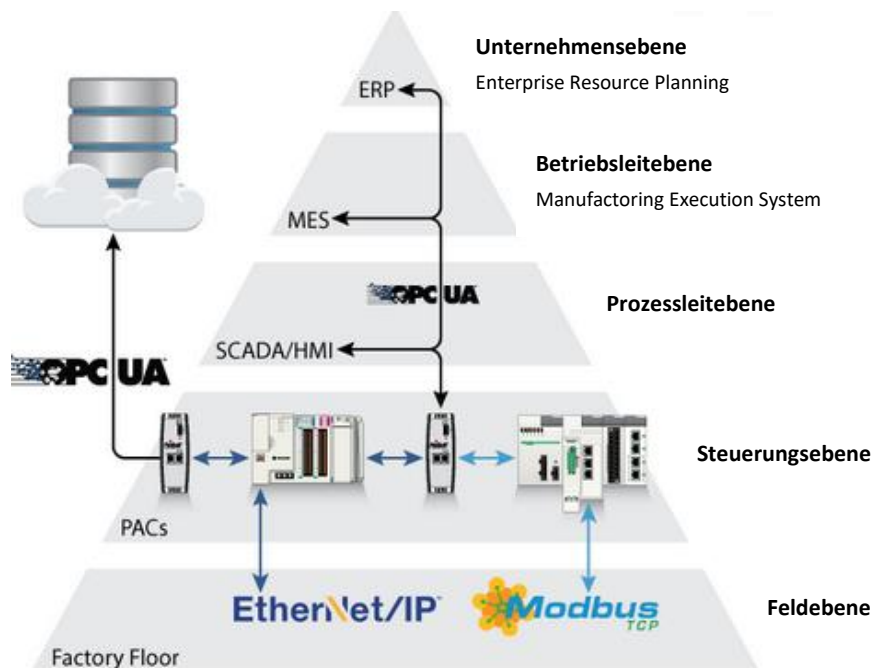


Abbildung 96 OPCUA in der Automatisierungspyramide

Das Protokoll ist **plattformunabhängig** und verfügt über integrierte Sicherheitsmechanismen. Da OPC UA flexibel und unabhängig ist, wird es heute auch als ideales **Kommunikationsprotokoll** für die Umsetzung **von Industrie 4.0** angesehen.

### 5.7.1 Entwicklungsgeschichte

Prozessdaten werden über verschiedene Ebenen hinweg erfasst, gesammelt und weiterverarbeitet. Der Datenaustausch wird erheblich erschwert, wenn **kein einheitlicher Schnittstellenstandard zwischen Feld- und Leitebene** genutzt werden kann.

Nachteile von proprietären Schnittstellen:

- jede Steuerung erfordert eine Entwicklung einer passenden Treiber-Software
- die angebotenen Kommunikations- und Schnittstellenfunktionen sind uneinheitlich

### Object Linking and Embedding (OLE):

Seit dem Einsatz von PC's in der Automatisierung wurde versucht, monolithische Software durch **wiederverwendbare Softwarekomponenten** zu ersetzen. Erste Lösungen zum standardisierten Datenaustausch zwischen Softwarekomponenten wurden von **Microsoft** mit Object Linking and Embedding ermöglicht.

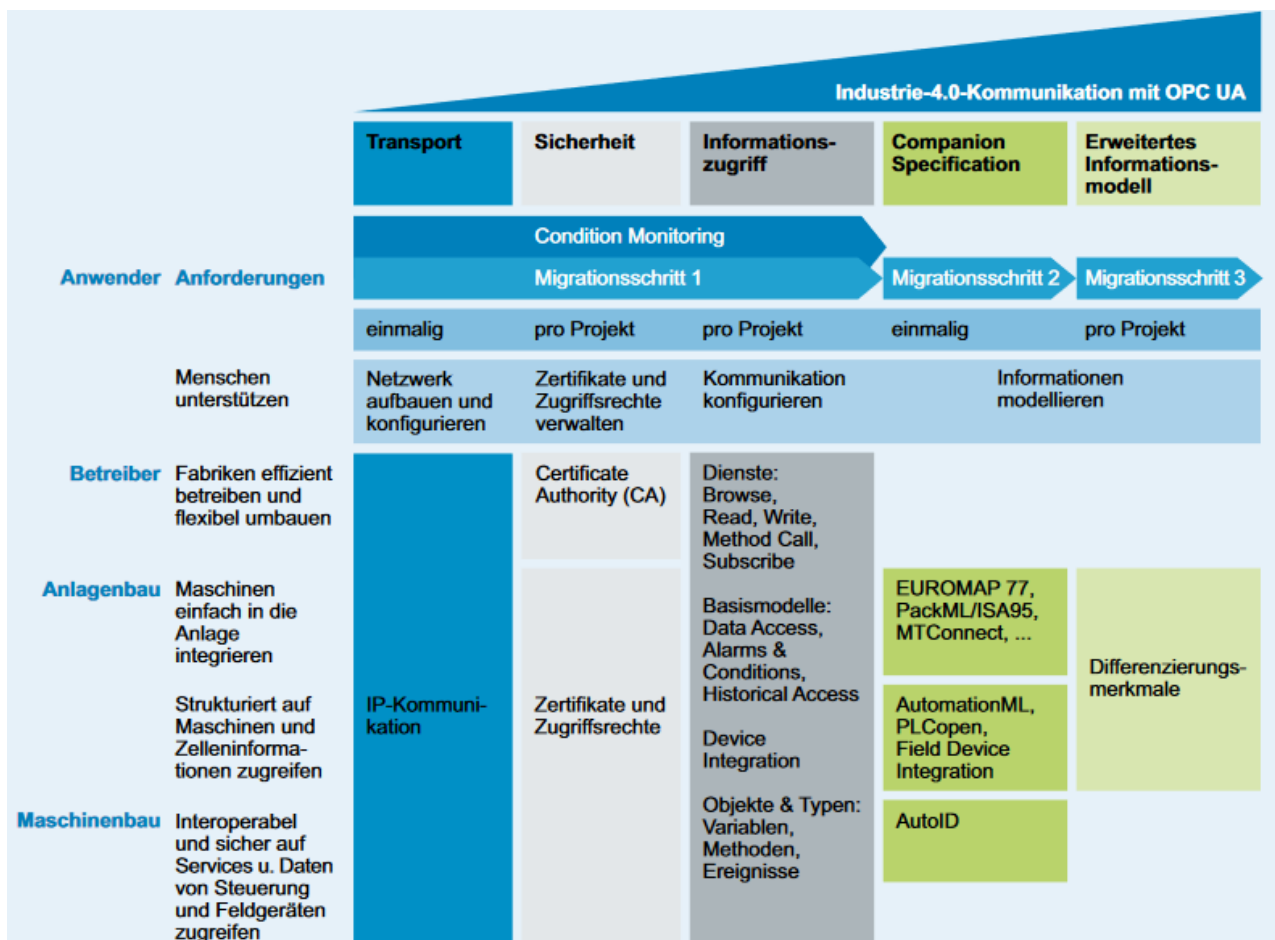
### OLE for Process Control (OPC):

1995 hatten einige **Software-Hersteller von Prozessvisualisierungen** die Idee, die Kommunikation zu den unterschiedlichen Automatisierungsgeräte auf Basis OLE zu standardisieren. Microsoft lieferte dabei die technische Unterstützung. Es entstand die erste Spezifikation von **OPC**, was zum damaligen Zeitpunkt als Abkürzung für **OLE for Process Control** stand.

### **OPC Unified Architecture:**

Die **größte Einschränkung** der ersten Generation war die **Bindung an Microsoft und Windows**. Im Jahr 2003 begann die Entwicklung einer neuen plattformunabhängigen Architektur mit der Bezeichnung **OPC Unified Architecture**. OPC UA ist Plattformunabhängig und unterscheidet sich in der Spezifikation und Umsetzung grundlegend von der ersten OPC-Versionen. Die Abkürzung OPC steht heute für **Open Platform Communication**.

## 5.7.2 OPC UA als einheitliche Kommunikationsschnittstelle



In einem ersten Schritt (Migrationsschritt 1) wird OPC UA als **einheitliche Kommunikations-schnittstelle für den Informationszugriff** verwendet. Informationszugriff bedeutet, dass von Maschinen und Anlagen zur Verfügung gestellte Daten manuell gefunden und ausgewertet werden können.

### 5.7.3 OPC UA Kommunikations-Stack

**OPC UA ist vollständig in Software realisiert.** OPC UA ist üblicherweise Bestandteil des Steuerungsbetriebssystems und liegt unterhalb des Anwendungsprogramms. OPC UA konkurriert nicht mit Bussystemen wie z.B. PROFINET, sondern **stellt dem Anwendungsprogramme nur standardisierte Dienste zum Datenaustausch** auf Basis einer IP-Kommunikation zur Verfügung.

Der Aufbau eines OPC UA-Interface, egal ob für Server oder Client, gliedert sich in mehrere Software-Schichten. **OPC UA Realisierungen** sind von **verschiedenen Firmen** und in **mehreren Programmiersprachen** erhältlich (z.B. C, C#, JAVA).

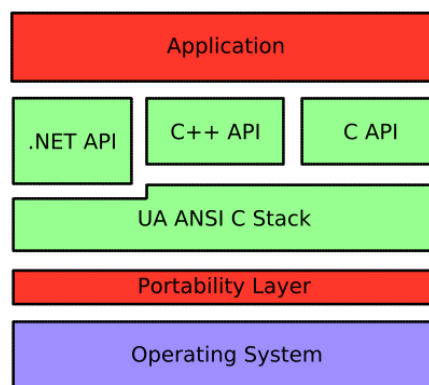


Abbildung 97 OPC UA Kommunikationsstack

### 5.7.4 Vorteile der OPC UA Technik

- **Herstellerunabhängigkeit bei Hard- und Software:**

Ein Hardware-Hersteller muss nur noch ein **OPC UA Server-Interface**, ein Hersteller von Softwareprodukten für Visualisierungen, Messsysteme etc. nur noch ein **OPC UA Client-Interface** in sein System integrieren.

Der **Endkunde** wiederum **kann frei** zwischen den verschiedenen Anbietern von Hard- und Software-Komponenten **wählen** und sich auf funktionale Auswahlkriterien konzentrieren.

- **Multi-Client Fähigkeit:**

Jeder OPC Server ist in der Lage Anfragen von **mehreren Clients** zu bearbeiten. Sollen Daten mehrfach genutzt werden, zum Beispiel von einer Visualisierung, als auch von einer Datenbank, können verschiedene Anwendungen auf die Daten des OPC Servers zugreifen.

- **Netzwerkfähigkeit und Internet/Intranet:**

OPC UA Klienten kommunizieren über ein gesichertes oder ungesichertes IP-Protokoll mit einem OPC UA Server im lokalen oder verteilten Netzwerk. OPC UA Klienten bemerken nicht einmal, ob es sich um eine lokale oder remote Datenquelle handelt (transparenter Datenzugriff).

### Beispiel: OPC Server für KNX

Der OPC Server ist die Sammelstelle für KNX-Datenpunkte

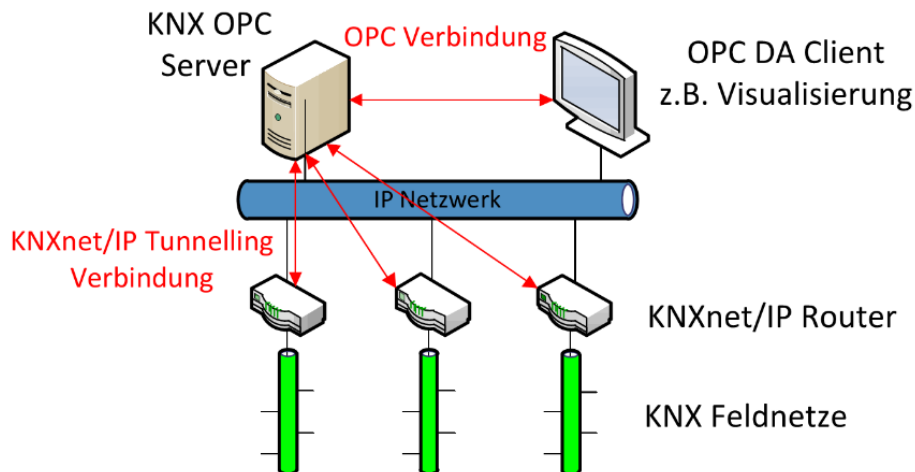


Abbildung 98 OPC Server für KNX

### 5.7.5 OPC UA-Spezifikation

Die OPC UA Spezifikation besteht aus mehreren Teilen, die laufend schrittweise erweitert wird. Folgende Teile sind bereits verfügbar. Die in den meisten Automatisierungsgeräte verfügbaren Teile sind die Schnittstellen für den **Zugriff auf Daten (Variablen)** und **Alarmer**.

- Teil 1: Übersicht und Konzepte
- Teil 2: Modell für die IT-Sicherheit
- Teil 3: Adressraummodell
- Teil 4: Dienste
- Teil 5: Informationsmodell
- Teil 6: Protokollabbildungen
- Teil 7: Profile
- Teil 8: **Daten (Variablen)**
- Teil 9: **Alarmer**
- Teil 10: Programme
- Teil 11: Historischer Zugriff

...

Frei verfügbare OPC UA Clients zum Testen:

<https://www.unified-automation.com/products/development-tools/uaexpert.html>

<https://www.prosysopc.com/products/opc-ua-client-for-android/>