

3. Datenübertragung

Gliederung

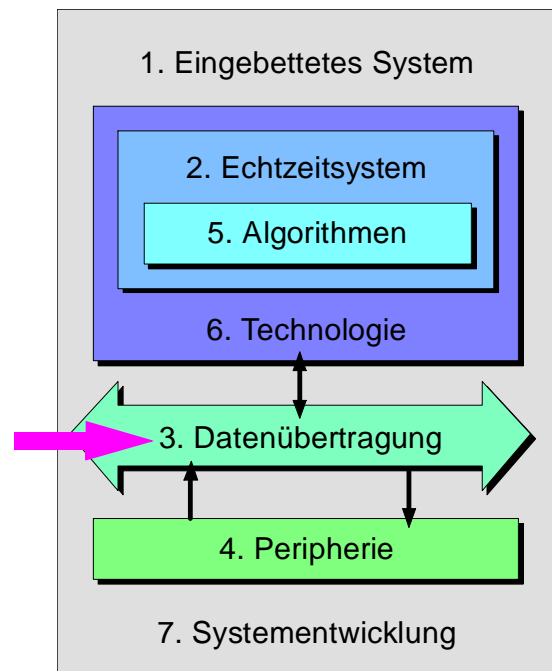
- 3. Datenübertragung
 - 3.1 Feldbusse
 - 3.1.1 Technologie im Überblick
 - 3.1.2 ISO/OSI-Referenzmodell
 - 3.1.3 Bus-Topologien
 - 3.1.4 Physikalisches Medium
 - 3.1.5 Buszuteilungsverfahren
 - 3.1.6 Telegrammaufbau/Dienste
 - 3.1.7 Datensicherung
 - 3.1.8 Application Layer Interface
 - 3.1.9 Kommerzielle Realisierungen
 - 3.2 AD/DA-Wandlung

Anwendungsmöglichkeiten

- Übertragung von Daten zwischen Rechensystem und Peripherie
- Auswahl und Weiterentwicklung von Feldbussen beim Entwurf und Entwicklung von ES

Assoziationen

- Welche Datenübertragungssysteme kennen Sie?



3. Datenübertragung

Kapitel 3: Datenübertragung

3.1 Feldbusse [Reißenweber98]

3.1.1 Technologie im Überblick

Problem: Wie kann das Echtzeitsystem mit seiner Peripherie **kommunizieren**?

Lösung: Einsatz von **Bussystemen** wie z.B. Feldbusse.

Aber: Benötigte Eigenschaften hängen vom **Anwendungsgebiet** ab!

Anwendungsgebiete

- **Fertigungsautomatisierung:**
 - Steuerung von Werkzeugmaschinen in Fabriken...
 - Anzahl von Komponenten, Übertragungsraten und Strecken im mittleren Bereich
- **Prozeßautomatisierung/ Verfahrenstechnik:**
 - Steuerung von technischen insbesondere chemischen Prozessen
 - Komponentenzahl und Übertragungsraten eher gering aber *sichere* Systeme nötig!
- **Gebäudeautomatisierung:**
 - Steuerung von Klimaanlage, Beleuchtung,
 - Viele Komponenten und lange Strecken, aber keine großen Datenmengen
- **Fahrzeugtechnik:**
 - Vielzahl von Komponenten, wie: Motormanagement, ABS, ESP, Beleuchtung, ...
 - Kurze Strecken aber auch kurze Reaktionszeiten, hohe Übertragungssicherheit.

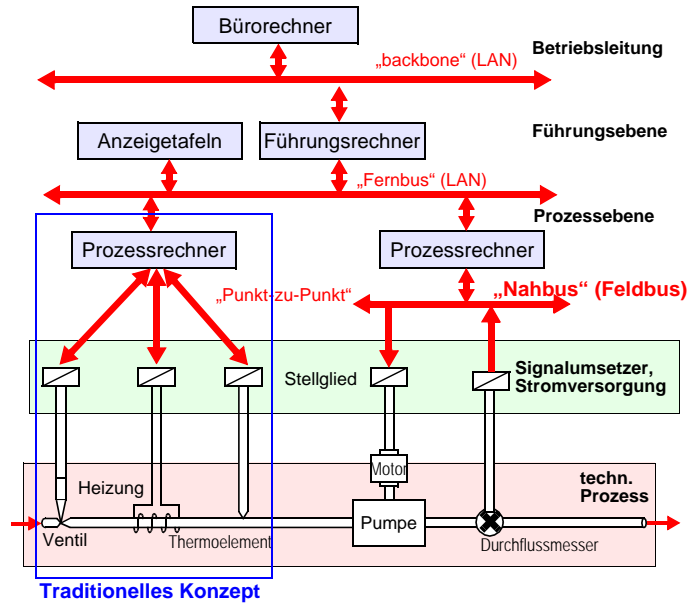


3. Datenübertragung

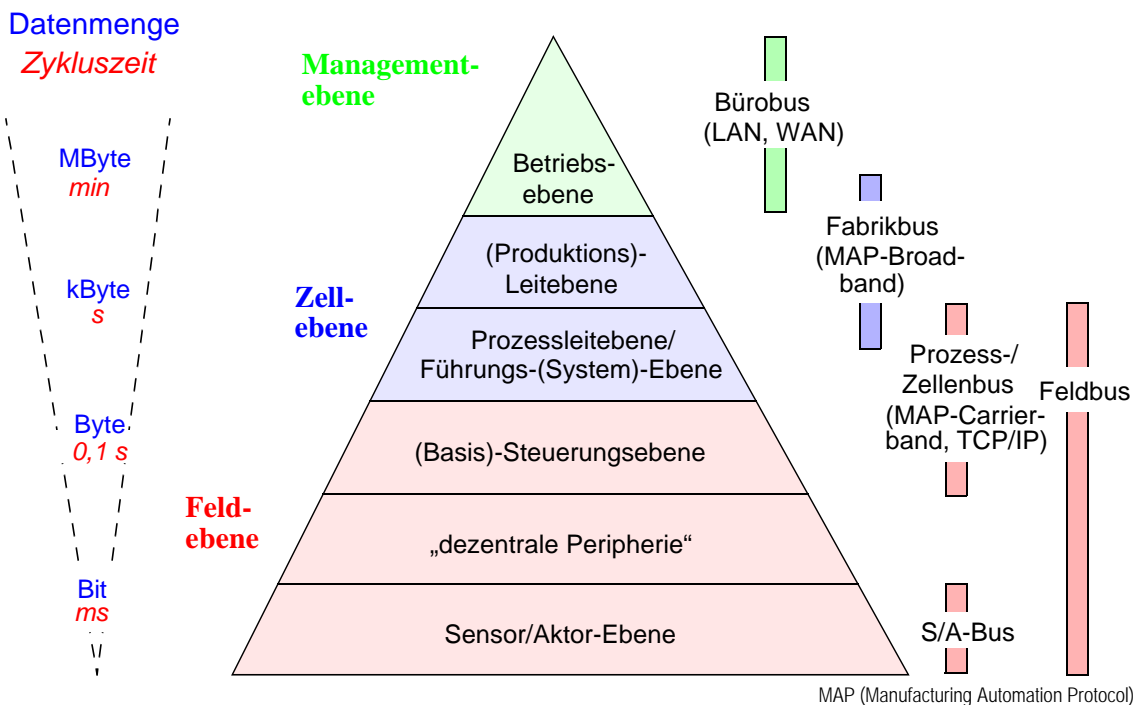
Prinzipielle Struktur digitaler Automatisierungssysteme

Für Automatisierungssysteme ergibt sich i.a. die untenstehende **mehrschichtige Struktur**, die aber je nach Anwendungsbereich (→Fahrzeugtechnik) variieren kann.

- **Offener Bus (Backbone):**
Vernetzung der Fabrikationsanlagen mit der Managementebene bzw. Vernetzung mehrerer Fabriken mittels Ethernet.
- **Systembus (Zellenbus):**
Verbindet Anzeige und Bedienkomponenten mit prozessnahen Komponenten. Spezielle Zellenbusse mittlerweile durch Ethernet verdrängt.
- **Feldbus:**
Verbindet die prozessnahen Komponenten wie Sensoren oder Aktoren mit Steuerrechnern.
- **Aktor-Sensor-Bus:**
Einfache und preisgünstige Variante eines Feldbusses.



3. Datenübertragung

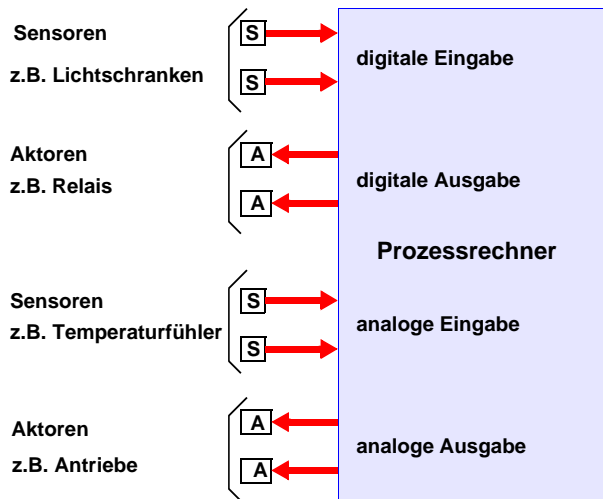


3. Datenübertragung

Traditionelle Lösung

Die folgende Lösung entwickelte sich seit den **sechziger Jahren** des 20. Jhd. Es wurden nur wenige Prozessrechner verwendet, da die Kosten für diese noch im 7-stelligen Bereich lagen. Bei neuen Systemen wird dieses Konzept daher nur noch **selten angewendet**.

- **Zentraler Prozessrechner:**
 - Meist analoge Ein- und Ausgabe.
 - Ein Rechner steuert Teilsystem oder sogar ganze Anlage.
 - Betrieb läuft unter Echtzeitbetriebssystem.
- **Einfache Sensoren:**
 - Liefern meist nur einen (analogen) Messwert.
 - Keine Parameter (z.B. Messbereiche) online einstellbar.
- **Aufwendige Verkabelung:**
 - Vielzahl von Kabelverbindungen.
 - Große Rangierverteiler.
 - Schwer rekonfigurierbar.
- **Einheitliches System:**
 - Internationale Normung.
 - Keine Kompatibilitätsprobleme.



3. Datenübertragung

Anforderungen/Wünsche der Anwender:

- **Funktionalität:**
 - Einhalten von Zeitschranken ⇒ harte Echtzeitanforderungen.
 - Erreichen höherer Messgenauigkeit ⇒ digitale Meßsignale.
 - Anpassung der Anlagen an kürzere Produktzykel ⇒ Flexibilisierung.
- **Fehlertolerantes System:**
 - Übertragungsfehler dürfen nicht zur Katastrophe führen.
 - Ausfall von Komponenten darf komplettes System nicht lahmlegen.
⇒ Dezentralisierte Systeme.
- **Kostenersparnis:**
 - Kleine billige Controller statt großer Prozessrechner.
 - Verbesserte Fehlerdiagnose.
 - Verbesserte Wartung, da Sensoren/ Aktoren z.B. Abnutzungen oder Ausfälle melden.
⇒ Einsatz „intelligenter“ Sensoren/ Aktoren.
- **Zukunftssicheres System:**
 - Komponenten sollten einfach austauschbar sein.
 - Keine Bindung an *einen* Hersteller.
⇒ Standardisierung oder besser: *universelles System*.



3. Datenübertragung

3.1.2 ISO/OSI-Referenzarchitektur (vgl. GS-Vorlesung „Systemsoftware“)

Das OSI-Basisreferenzmodell (OSI = Open System Interconnection) von ISO (Int. Organization of Standardisation) liefert einen **Standard für die Modellierung** von Kommunikationssystemen.

Aufgaben eines Kommunikationssystems:

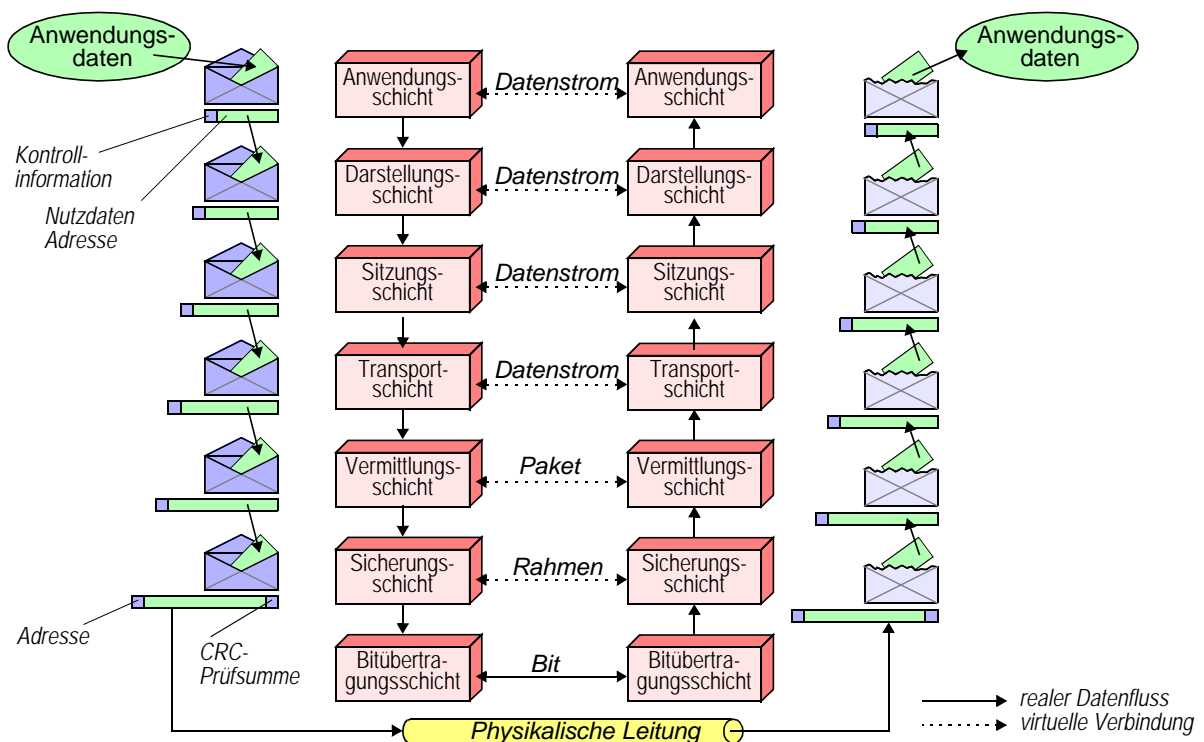
- Festlegung der physikalischen Übertragungsmedien.
- Sicherung der zu übertragenden Daten gegenüber Bitfehlern oder Paketverlusten.
- Adressierungsmechanismen bereithalten und einen Weg zum Empfänger finden.
- Aufbau und Abbau von Verbindungen.
- Bereitstellen von Diensten für die Anwendungen.

Eigenschaften des Referenzmodells:

- Enthält keine technischen Lösungen, sondern dient zur Beherrschung der Komplexität.
- Zuordnung der verschiedenen Aufgaben von Kommunikationssystemen zu 7 Schichten.
- Jede Schicht greift nur auf die direkt unter ihr liegende Schicht zu.
- Jede Schicht bietet nur der direkt über ihr liegenden Schicht Dienste an.
- Die Protokollinformation der Schicht n hat für die Schicht $n-1$ keine spezielle Bedeutung.



3. Datenübertragung



3. Datenübertragung

ISO/OSI Schichtenmodell:

Schicht	Name	Aufgabe
7	Anwendungsschicht (Application Layer)	Festlegung des Dienstes des Kommunikationspartners für das jeweilige Anwendungsprogramm (z.B. Dateiübertragung, E-Mail, ...)
6	Darstellungsschicht (Presentation Layer)	Festlegung der Strukturen der Anwenderdaten (Datenformate) inkl. Formatierung, Verschlüsselung, Zeichenersetzung, ...
5	Sitzungsschicht (Session Layer)	Auf- und Abbau von logischen Kanälen auf dem physikalischen Transportsystem
4	Transportschicht (Transport Layer)	Steuerung des Datenstroms durch Bereitstellen von logischen Kanälen auf dem physikalischen Transportsystem
3	Vermittlungsschicht (Network Layer)	Festlegung von Wegen für einen Datenstrom durch das Netzwerk
2	Sicherungsschicht (Data Link Layer)	Sicherstellung eines korrekten Datenstroms durch Festlegung von - Kanalkodierung - Buszuteilungsverfahren - Prüfmechanismen zur Fehlererkennung
1	Bitübertragungsschicht (Physical Layer)	Festlegung des Übertragungsmediums (→ elektrische und mechanische Eigenschaften)

Anmerkung: Schichten 3 bis 6 sind üblicherweise bei Feldbussen nicht realisiert oder in Schicht 7 modelliert.

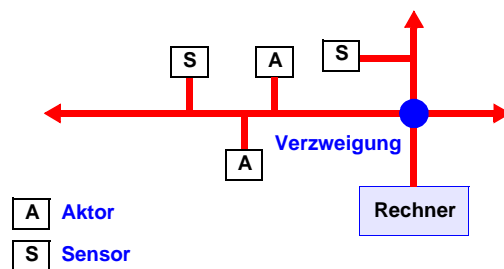


3. Datenübertragung

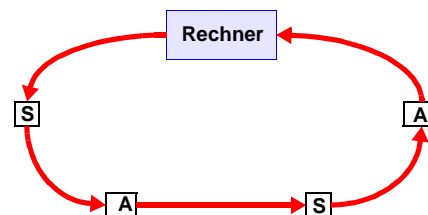
3.1.3 Bus-Topologien

- **Baum/ Linienstruktur:**
 - Einzelne Komponenten hängen über kurze Stichleitungen am Bus.
 - Eine unverzweigte Baumstruktur wird als Linienstruktur bezeichnet.
 - Teilnehmerzahl stark begrenzt, da jeder Teilnehmer als Last wirkt.
 - An Verzweigungspunkten können/ müsse daher Repeater zur Signalverstärkung eingesetzt werden.
- **Ringstruktur:**
 - Bus führt von einem Teilnehmer zum nächsten.
 - Jeder Teilnehmer wirkt als Repeater.
 - Ausfall eines Teilnehmers bewirkt Ausfall des Systems.

Baumstruktur



Ringstruktur



3. Datenübertragung

3.1.4 Schicht 1: Physikalisches Medium

In Abhängigkeit vom Anwendungsbereich können sich spezielle Anforderungen ergeben:

- **Eigensicherheit:**
 - Prinzip: Die verwendeten Spannungen und Ströme müssen so klein sein, dass die zum Entzünden eines umgebenden Gasgemisches nötige Energie nicht erreicht wird.
 - In der Chemische Industrie sehr wichtig.
 - Eigensicherheit muss zertifiziert werden.
 - **Stör- Freiheit/Sicherheit:**
 - Geringe elektromagnetische Abstrahlung.
 - Unempfindlichkeit gegenüber Störfelder.
 - **Einfache Verkabelung:**
 - Gemeinsame Zuführung von Signal- und Hilfsenergie.
 - Kleine Kabeldurchmesser bevorzugt.
- Folgen:**
- Teilweise nur geringe Teilnehmerzahl (z.B. 32 bei RS-485 Norm).
 - Verwendung von geschirmten und ungeschirmten Kabel.
 - Mittelwertfreie Signalcodierungen bevorzugt.



3. Datenübertragung

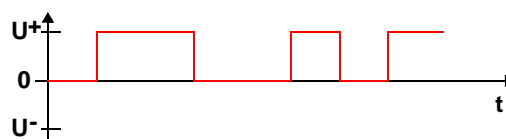
Signalcodierungen

Sendebitfolge

0 1 1 0 0 1 0 1

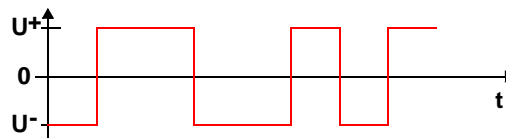
unipolar

- nicht mittelwertfrei
- keine Taktinformation



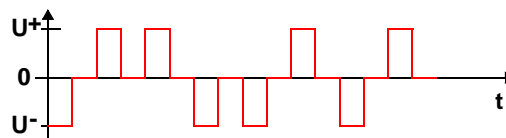
Non-Return-To-Zero (NRZ), bipolar

- mittelwertfrei
- keine Taktinformation



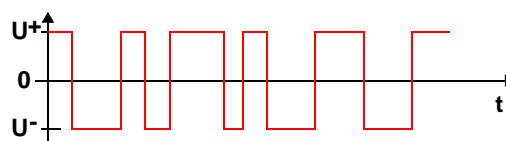
Return-To-Zero (RZ), bipolar

- mittelwertfrei
- volle Taktinformation



Manchester-Code

- mittelwertfrei
- volle Taktinformation
- weniger Flankenwechsel als RZ



3. Datenübertragung

3.1.5 Schicht 2: Buszuteilungsverfahren

Regelung der Busvergabe nötig, da gleichzeitiges Senden mehrerer Teilnehmer zu Datenübertragungsfehlern führt.

Arbitrator-Producer-Consumer-Verfahren

- Busvergabe wird **zentral** von einer Stelle (Arbitrator) gesteuert, der jeweils ein Datum über eine bestimmte Identifikation anfordert. Dieses wird von genau einem Producer geliefert.
- Alle Teilnehmer (Consumer), die dieses Datum brauchen, erhalten dies **gleichzeitig**.
- Die Consumer **quittieren nicht**.
- Die **Reihenfolge** der Übertragungen ist in einer Tabelle des Arbitrators festgelegt.
- Die Daten werden in der Regel **zyklisch** und nicht nach Bedarf übertragen.
- Ein Producer kann aber dem Arbitrator signalisieren, dass er noch zusätzliche Daten senden bzw. empfangen will (**azyklische Übertragung**).

Vorteile:

- **Echtzeitfähigkeit** durch zyklischen Betrieb gesichert.
- **Azyklische Datenübertragung** möglich.

Nachteile:

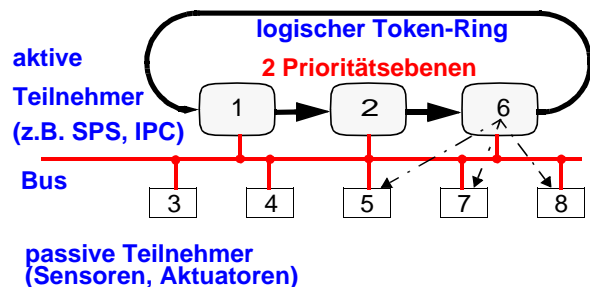
- Ausfall des Arbitrators **legt Bus lahm**.
- **Overhead** durch Arbitrator-Telegramme.
- **Zykluszeit** steigt mit Anzahl der Teilnehmer.



3. Datenübertragung

Token-passing-Verfahren mit Master/Slave-Mechanismus

- Es gibt ein **Token**, das die Sendeberechtigung repräsentiert.
- Der aktive Teilnehmer (Master) mit Token gibt dieses, falls er **keinen Kommunikationsbedarf** hat, oder nach **Ablauf einer Frist** an den nächsten aktiven weiter.
- Der Teilnehmer mit Token kann **Daten** von einem passiven Teilnehmer **anfordern**.
- Passive Teilnehmer (Slaves) senden nur nach **direkter Aufforderung** durch einen Master.
- Es gibt genau einen Master, der beim **Systemstart** das Token generieren darf.



Vorteile:

- **Echtzeitfähigkeit** garantiert, da eine maximale Tokenumlaufzeit existiert.
- Wegen **maximaler Umlaufzeit** kann Verlust des Tokens erkannt werden und defekte Teilnehmer können bei Tokenvergabe übergangen werden.
- Zykluszeit wird durch **Master/ Slave-Mechanismus** verringert.

Nachteile:

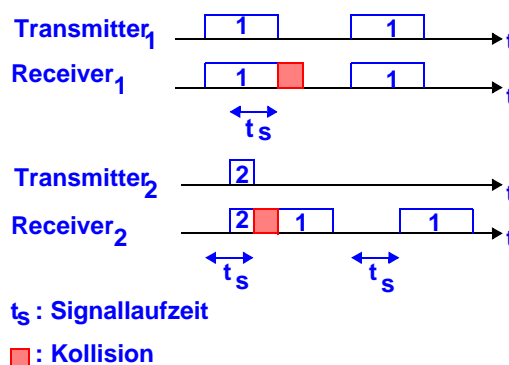
- Tokenumlaufzeit **steigt** mit Anzahl der (aktiven) Teilnehmer.
- Großer **Rekonfigurationsaufwand**, wenn Teilnehmerzahl sich ändert.
- **Passive Teilnehmer** können nicht azyklisch senden (z.B. Alarmmeldung).



3. Datenübertragung

Carrier-Sense-Multiple-Access/Collision-Detection (CSMA/CD)

- Busvergabe erfolgt nicht zyklisch sondern **nach Bedarf**.
- Ein Teilnehmer **hört** zuerst den Bus **ab** und sendet, falls er frei ist.
- Durch (gleichzeitigen) Mehrfachzugriff und wegen der Signallaufzeit sind **Kollisionen** möglich.
- Erkennt ein Sender eine Kollision, so **bricht** er seine Übertragung **ab**
- Nach Abbruch versucht er es nach Ablauf einer **zufälligen Zeitspanne** nochmal.



Vorteile:

- **Kein Overhead**, da Übertragungen nur bei Bedarf stattfinden.
- Hohe bis **sehr hohe Teilnehmerzahlen** möglich.
- Teilnehmer können ohne großen **Rekonfigurationsaufwand** angeschlossen bzw. abgeklemmt werden.

Nachteile:

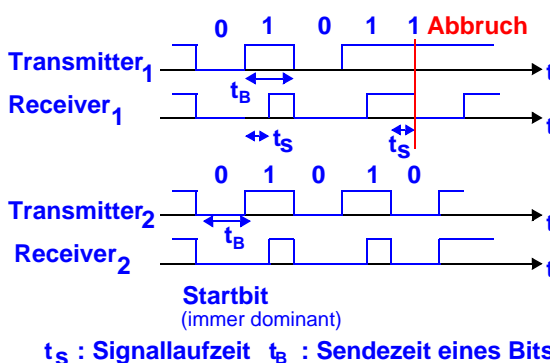
- CSMA/CD ist **nicht echtzeitfähig**, da beliebig lange Verzögerungen möglich.
- **Effizienz sinkt** stark mit steigender Last.



3. Datenübertragung

Carrier-Sense-Multiple-Access/Collision-Avoidance (CSMA/CA)

- Busvergabe erfolgt nicht zyklisch sondern nach **Bedarf**.
- Ein Teilnehmer **hört** zuerst den Bus **ab** und sendet eine Nachricht beginnend mit der Nachrichtenennung, falls der Bus frei ist.
- Mehrfachzugriff wird durch **spezielle Busauslegung** erkannt („0“ dominant gegenüber „1“).
- Erkennt ein Sender das Überschreiben der von ihm gesendeten Kennung, so bricht er ab und wartet.
- **Kollisionen** werden so vermieden.



Vorteile:

- **Kein Overhead**, da Übertragungen nur bei Bedarf stattfinden.
- Bus wird **optimal genutzt**, da eine Nachricht immer durchkommt.
- Nachrichtenennungen wirken als **Priorität**.
- Hochpriorie Nachrichten werden verzögerungsfrei übertragen \Rightarrow **optimale Echtzeiteigenschaften**.

Nachteile:

- Mögliche Kollision nur bei quasi **gleichzeitigem Senden** erkennbar (Signallaufzeit klein gegen Sendezeit pro Bit).
- Signallaufzeit begrenzt das Produkt aus **Leitungslänge** und **Übertragungsrage**
 \Rightarrow Übertragungsrage bricht bei großer Leitungslänge ein.



3. Datenübertragung

3.1.6 Schicht 2: Telegrammaufbau/ Dienste

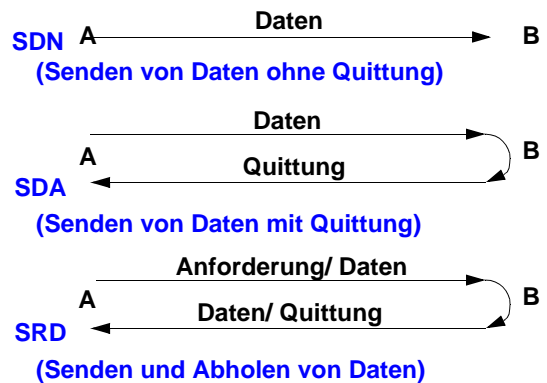
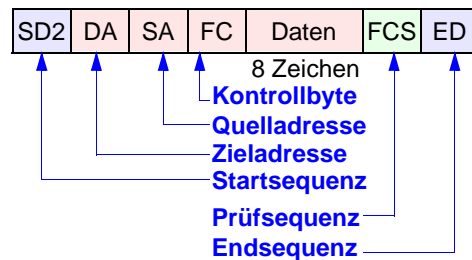
Telegrammaufbau:

- Neben eigentlichen Nutzdaten z.T. umfangreiche **Protollinformation** nötig.
- **Telegrammlänge** kann je nach Dienst fest oder variabel sein.
- Im Feldbusbereich werden meist nur **kleine Telegramme** verwendet.

Dienste:

- Nur **einfache Protokolle**.
- Dienste können **verbindungslos** (Multicast) oder **verbindungsorientiert** sein.
- **Empfangsbestätigung** kann in Schicht 2 Dienste aufgenommen werden.
- Regelt aber nur das Verschicken der einzelnen Telegramme
⇒ Verschicken von **unstrukturierten Daten**.

Beispiel: Profibus



3. Datenübertragung

3.1.7 Schicht 2: Datensicherung

Problem:

Übertragfehler z.B. durch elektromagnetische Felder können zu katastrophalen Fehlverhalten führen.

Lösung:

- Verwendung von physikalischen Schutzmechanismen wie **Schirmung** (trotzdem Fehleraten von 1/10000 möglich).
- Verwendung von **höheren Signalpegeln** (nicht möglich bei eigensicheren Systemen).
- Einführen von **redundanten Prüfbits** zum Erkennen oder Korrigieren von Bitfehlern.

Definition: **Hammingdistanz:**

Die Hammingdistanz ist die Anzahl der Bits, in denen sich die Codes zweier unterschiedlicher Zeichen eines Alphabets mindestens unterscheiden.

Satz: **Fehlererkennung:** (ohne Beweis)

Sei d die Hammingdistanz eines Alphabets, dann gilt für die Anzahl f der sicher detektierbaren Bitfehler: $f = d - 1$.



3. Datenübertragung

Verfahren zur Bitfehlerdetektion

Parity-Bit

- Dem Code wird ein **Bit** (Parity) **hinzugefügt**. Es ist 1, falls die Anzahl der Bits mit Wert 1 im gesamten Wort gerade ist, sonst ist parity 0 (z.B. 1100 → 0 1100).
- Da die **Hammingdistanz** $d = 2$ ist, können 1-Bitfehler sicher erkannt werden. Insgesamt wird aber jede ungerade Zahl von Bitfehlern erkannt.

Prüfsumme

- Dem Code wird eine oder mehrere **1-Byte-Prüfsumme**(n) zugefügt. Diese berechnet sich, indem die Datenbytes modulo 256 aufsummiert werden.
- Wird üblicherweise **mit Parity-Bit** (pro Datenbyte und Prüfbyte) kombiniert.
- Für die **Hammingdistanz** gilt hier $d = 4$, daher können 3-Bitfehler sicher erkannt werden. Insgesamt wird aber jede ungerade Zahl von Bitfehlern erkannt.

Cyclic-Redundary-Check (CRC) *(siehe VL Systemsoftware)*

- **Zahlentheoretisches** Verfahren.
- Die zu schützenden Daten werden als große Binarzahl interpretiert und durch ein sogenanntes **Generatorpolynom** dividiert. Der Rest dieser Division wird als Prüfsumme verwendet.
- Die **Hammingdistanz** hängt von dem verwendeten Generatorpolynom ab. Die im Feldbusbereich verwendeten Polynome liefern $d = 6$.
- CRC ist das **effizienteste** der 3 Sicherungsverfahren.



3. Datenübertragung

3.1.8 Schicht 7: Application Layer Interface (ALI)

Das ALI stellt die **Schnittstelle** zwischen **Kommunikationssystem** und **Anwenderprogramm** her und steht daher eigentlich zwischen Schicht 7 und Schicht 8 (Anwenderprogramm), soll hier aber als Teil von Schicht 7 gelten.

Kommunikationsschnittstelle:

- **Datenorientierte Schnittstelle:**
 - Besonders geeignet für einfache aber schnelle Prozesse.
 - Alle Eingangs- und Ausgangsvariable (Prozessabbild) werden in einem Speicherbereich abgelegt.
 - Kommunikationssystem und Anwendungsprogramm greifen gemeinsam auf diesen zu.
 - Zyklische Übertragung von „reinen“ Daten ⇒ nur Schicht 2 Dienste nötig.
- **Nachrichtenorientierte Schnittstelle:**
 - Unterstützt intelligente Sensoren/Aktoren.
 - Alle Teilnehmer verschicken Nachrichten oder melden Ereignisse (Events).
 - Neben zyklischer auch azyklische Übertragung.
 - Vielzahl unterschiedlicher Nachrichten bzw. Protokolle⇒ Benötigt leistungsfähige Dienste der Schicht 7.



3. Datenübertragung

Kommunikationsobjekte

Zur **Strukturierung** werden den zu übertragenden Daten Kommunikationsobjekte mit festgelegten Datentypen zugeordnet.

- **Definition von Objekten:**
 - Einfache Basisdatentypen wie z.B. BOOLEAN, UNSIGNED8 oder TIME-OF-DAY.
 - Möglichkeit zur Definition komplexerer Typen gegeben (ARRAY, RECORD oder EVENT).
⇒ Daten können einheitlich interpretiert werden.
 - Neben diesen Datentypen können auch Codesegmente verwaltet werden.
- **Übertragung der Objekte:**
 - Objektdefinition muss übermittelt werden.
 - Übermittlung nur in komprimierter Form sinnvoll.
⇒ Verwendung von Objektverzeichnissen.
- **Objektverzeichnis:**
 - Anlegen einer Übersicht aller bekannten Objekte.
 - Strukturdefinition der Objekte in Tabellen.
 - Allen Einträgen wird ein Index zur weiteren Identifikation zugeordnet.
 - Verzeichnisse werden lokal angelegt und können von anderen abgerufen werden.
 - Für die Übermittlung der Verzeichnisse gibt es einen speziellen Dienst.



3. Datenübertragung

Kommunikationsdienste

Diese können prinzipiell immer bestätigt oder unbestätigt sein, bei Übertragung von Daten greifen sie auf die **Objektdefinitionen** zurück. Dabei gibt es meist Dienste für folgende Gruppen:

- **Basisdienste:**
 - Initiierung oder Abbruch eines Verbindungsaufbaus.
 - Auslesen von Objektverzeichnissen.
 - Schreiben von Objektverzeichnissen.
- **Übertragung von Variablen :**
 - Nachrichten zum Lesen und Schreiben von Objekten (Daten).
 - Unterschiedliche Nachrichten für Singlecast und Multicast möglich.
- **Remoteausführung von Programmen:**
 - Herunterladen oder Löschen von Programmcode.
 - Starten, Anhalten oder Abbrechen der Ausführung von Programmcode.
- **Eventhandling:**
 - Senden und Bestätigen von Ereignissen.
 - Ein- Ausschalten des Ereignis-Mechanismus.



3. Datenübertragung

3.1.9 Kommerzielle Realisierungen

Wunsch: Ein einziges **universell** einsetzbares Kommunikationssystem.

Problem: Stark **unterschiedliche Anforderungen** bei den einzelnen Anwendungsgebieten.

Aktuelle Marktsituation:

- **Systemvielfalt:**
 - Universeller Feldbus praktisch nicht realisierbar.
 - Es gibt eine ganze Reihe unterschiedlicher kommerzieller Feldbusse.
 - Jeder Hersteller/ Gruppe versucht sein/ ihr System zu etablieren.
- **Standardisierung(sversuche):**
 - Nationale Standards/ Normen existieren seit einigen Jahren.
 - Einige Systeme haben Marktführerschaft in speziellen Segmenten erreicht
⇒ De-Facto-Standards (z.B. *Profibus*, *CAN*, usw.).
 - Internationale Feldbusnorm ist praktisch gescheitert (Arbeitsbeginn: 1984)
⇒ De-Facto-Standards wurden in Norm zusammengefasst,
wie z.B. *P-Net*, *Profibus* und *World-Fip* in europäischer Norm EN-50170.



3. Datenübertragung

Aktor-Sensor-Interface ASI

Das ASI ist eigentlich ein **Sensor-Aktor-Bus** und kein Feldbus, man kann es aber als einfachen Feldbus ansehen, da es keinen prinzipiellen Unterschied gibt.

- **ISO/OSI-Schicht 1:**
 - Ungeschirmte Leitungen, Kabel haben asymmetrisches Profil (keine Anschlußfehler).
 - Hilfsenergie wird mit Datenleitung zugeführt ⇒ Manchestercodierung.
 - Übertragungsraten von 167kBit/s bei 100m Leitungslänge.
 - System für bis zu 32 Teilnehmer ausgelegt.
- **ISO/OSI-Schicht 2:**
 - Telegramme werden durch Parity-Verfahren gesichert.
 - Einfaches Master/Slave-Verfahren (wie Tokenring mit nur einem aktiven Teilnehmer).
 - Verwendung kurzer Telegramme (14Bit).
- **ISO/OSI-Schicht 7:**
 - Einfache Datenorientierte Schnittstelle keine speziellen Schicht 7 Dienste.
- **Spezielle Eigenschaften:**
 - Einfach und kostengünstig.
 - Ziel: Direkte Ankopplung der Sensoren/ Aktoren an Steuerrechner.



3. Datenübertragung

Profibus (Process Field Bus)

Den Profibus gibt es in drei Varianten (FMS, DP und PA) für **unterschiedliche Anwendungsbereiche**. Dabei ist Profibus-FMS die Standardvariante, DP ist primär für schnelle Kommunikation auf Sensor-Aktorebene und PA für Prozessautomatisierung konzipiert.

- **ISO/OSI-Schicht 1:**
 - Geschirmtes Twisted-Pair Kabel, Verwendung von NRZ-Kodierung.
 - Auch Lichtwellenleiter als Medium möglich.
 - Mindestens 9.6kBit/s bei 1200m Leitungslänge oder bis zu 1.2MBit/s bei 100m.
 - System für bis zu 32 Teilnehmer ausgelegt (bei unkritischen Systemen: 126).
- **ISO/OSI-Schicht 2:**
 - Telegramme werden durch Parity-Verfahren und Prüfsumme gesichert.
 - Tokenring mit Master/ Slave-Mechanismus.
 - Mehrere Dienste mit/ ohne Quittierung und z.T. variabler Telegrammlänge.
- **ISO/OSI-Schicht 7:**
 - Leistungsfähige Dienste der Schicht 7 mit Singlecast/Multicast, Objektverzeichnissen...
- **Spezielle Eigenschaften:**
 - Profibus-PA ist eigensicher.



3. Datenübertragung

CAN-Bus (Controller Area Network)

Speziell für den Einsatz in der **Fahrzeugtechnik** entworfen, mittlerweile ist er einer der verbreitetsten Feldbusse.

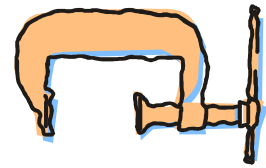
- **ISO/OSI-Schicht 1:**
 - Geschirmtes Twisted-Pair Kabel, Verwendung von NRZ-Kodierung.
 - Möglich sind z.B. 10kBit/s bei 5000m Leitungslänge oder 1 MBit/s bei 25m.
- **ISO/OSI-Schicht 2:**
 - Telegramme werden durch Parity-Verfahren und CRC gesichert.
 - Erkennt ein Teilnehmer einen Übertragungsfehler, so sendet er ein Error-Frame
⇒ alle anderen Teilnehmer erkennen den Fehler.
 - Telegramme für bis zu 2032 Nachrichten ausgelegt (Bei neuerer Norm deutlich mehr).
 - CSMA/CA-Verfahren.
 - Variable Telegrammlänge mit bis zu 8 Byte an Daten.
- **ISO/OSI-Schicht 7:**
 - Leistungsfähige Schicht 7 Dienste nicht in ursprünglicher Definition
⇒ etwa 4 unterschiedliche Systeme für Schicht 7 Dienste verbreitet.
- **Spezielle Eigenschaften:**
 - Nahezu optimale Echtzeitfähigkeit durch Prioritäten und schnelle Übertragung.
 - CAN-Bus ermöglicht sehr sichere Datenübertragungen.



3. Datenübertragung

Zusammenfassung

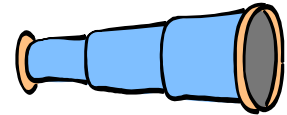
- Einsatz von Feldbussen zur Datenübertragung
- Eingesetzte Techniken sind vom Anwendungsgebiet abhängig
- Nutzung des ISO/OSI-Referenzmodells
- Vielzahl kommerzieller Realisierungen



Ausblick

Vertiefung dieser Vorlesung:

- Vertiefungs-Vorlesung "Bussysteme"



Nächste Vorlesung:

- Wie können die Sensordaten für das Rechensystem geeignet verfügbar gemacht werden?
- Mit welchen Signalen kann das Rechensystem die Aktuatoren geeignet ansteuern?



3. Datenübertragung

Gliederung

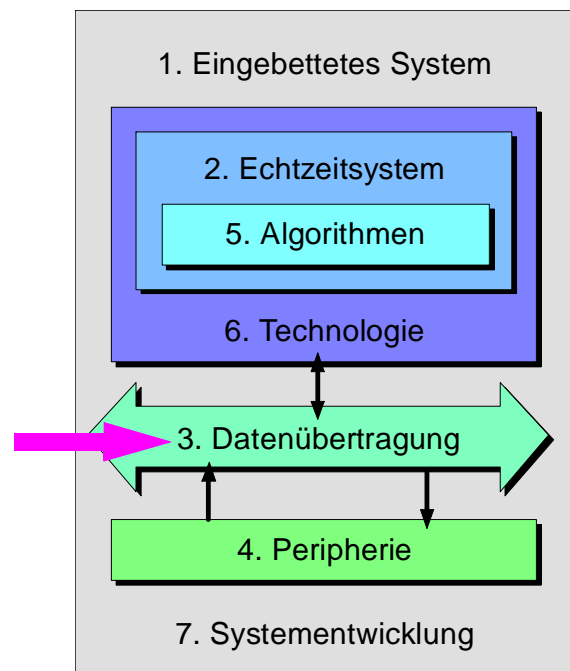
- 3. Datenübertragung
 - 3.1 Feldbusse
 - 3.2 AD/DA-Wandlung
 - 3.2.1 Operationsverstärker (OP)
 - 3.2.2 Analog/Digital-Wandler
 - 3.2.3 Digitale Ausgangsmodulation
 - 3.2.4 Digital/Analog-Wandler

Anwendungsmöglichkeiten

- Einlesen der Messwerte von Sensoren
- Ausgeben der Stellwerte für Aktoren
- Bewertung und Auswahl von Wandler

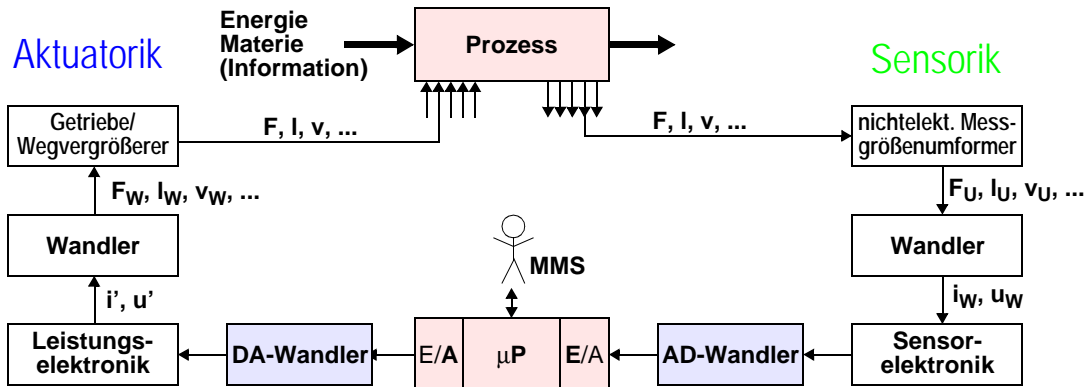
Assoziationen

- Welche Umformung von analogen in digitale Signale kennen Sie aus dem Alltag?
- Welche Probleme können da auftreten?



3. Datenübertragung

3.2 AD/DA-Wandlung [Zander90]

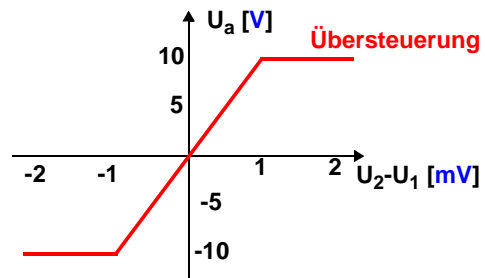
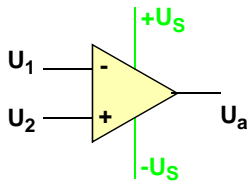


- **PCM-Wandler** (Pulse Code Modulation)
 - Wandler, die als Ergebnis den Analogwert (Eingangsspannung) digital kodieren
- **DM-Wandler** (Delta Modulation)
 - Ausgabe beschreibt Differenz zum Wert des vorangegangenen Abtastzeitpunktes



3. Datenübertragung

3.2.1 Operationsverstärker (OP) [Mechelke89], [Meister94]



wichtige Eigenschaften:

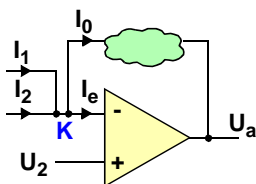
a) Differenzverstärker mit **extrem großer Verstärkung** (Praxis: $10^4 - 10^5$)

→ minimale Differenz zwischen U_1 und U_2 ergibt bereits maximale Ausgangsspannung ($\pm U_S$)

b) **extrem großer Eingangswiderstand**

→ Eingangsströme können als Null betrachtet werden

• bei rückgekoppeltem Ausgang regelt sich der OV folgendermaßen ein:



• **Eingangswiderstand = ∞** : $I_e = 0$

⇒ (Knoten K) $I_0 = I_1 + I_2$

⇒ I_0 „kompensiert“ Eingangsströme

• **Verstärkung = ∞** : U_K regelt sich auf U_2 ein (⇒ $U_K = U_2$)

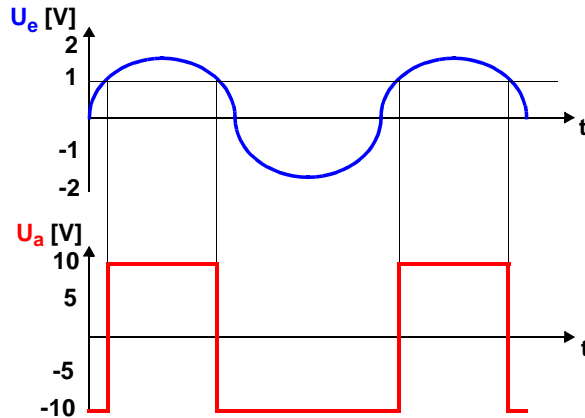
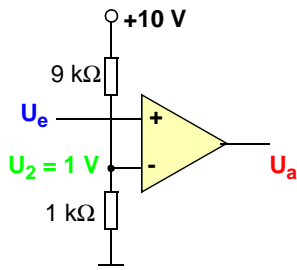
(Sei $U_K < U_2$. ⇒ $U_a \rightarrow +U_S$)

⇒ durch teilweise Rückkopplung von U_a steigt auch U_K [solange, bis $U_K = U_2$]



3. Datenübertragung

OV als Schwellwertschalter / Vergleichler



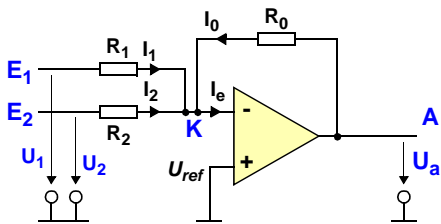
hier:

- durch Spannungsteiler fest eingestellte Referenzspannung $U_2 = 1\text{ V}$
- $|U_a| = \min \{10\text{V}, k \cdot |U_e - U_2|\}$ mit k extrem groß



3. Datenübertragung

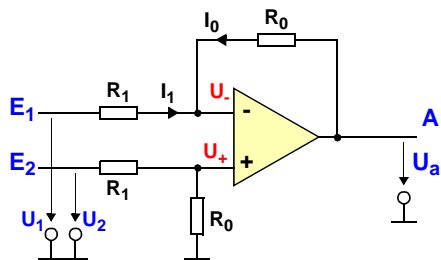
OV als Summierer



- $U_K = U_{ref} = 0\text{V}$: $I_1 = U_1 / R_1$
 $I_2 = U_2 / R_2$
 $I_0 = U_a / R_0$
- $I_e = 0\text{A}$: $I_0 = -(I_1 + I_2)$

$\Rightarrow U_a = -(R_0/R_1 \cdot U_1 + R_0/R_2 \cdot U_2)$
 $\Rightarrow U_a$ ist gewichtete Summe der Eingangsspannungen

OV als Subtrahierer



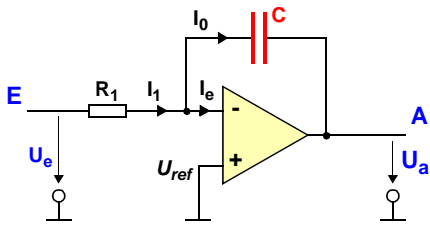
- $I_+ = 0\text{A}$: $U_+ = k \cdot U_2$ mit $k = \frac{R_0}{R_1 + R_0}$
- $U_- = U_+$: $U_- = k \cdot U_2$
 $\Rightarrow U_{R1} = U_1 - U_- = U_1 - k \cdot U_2$
- $I_- = 0\text{A}$: $I_0 = I_1 = \frac{U_{R1}}{R_1} = \frac{U_1 - k \cdot U_2}{R_1}$ und $U_{R0} = I_0 \cdot R_0$
- $U_a = U_- - U_{R0}$: $U_a = k \cdot U_2 - I_0 \cdot R_0 = k \cdot U_2 - \frac{U_1 - k \cdot U_2}{R_1} \cdot R_0$

$$U_a = -\frac{R_0}{R_1} U_1 + k \cdot U_2 \cdot \left(1 + \frac{R_0}{R_1}\right) = -\frac{R_0}{R_1} (U_1 - U_2)$$



3. Datenübertragung

OV als Integrierer



- $I_e = 0A$: $I_0 = I_1$
- $U_- = 0V$: $I_1 = U_e / R_1$
- Sei U_e konst.: $\Delta Q = I_0 \cdot \Delta t = I_1 \cdot \Delta t$
- $\Delta U_C = \Delta Q / C$: $\Delta U_C = \frac{DQ}{C} = \frac{I_1}{C} \times Dt = \frac{U_e}{R_1 \times C} \times Dt$
- $\Delta U_a = -\Delta U_C$: $\Delta U_a = -\frac{U_e}{R_1 \times C} \times Dt$
- Sei U_e variabel: $dU_a = -\frac{U_e}{R_1 \times C} \times dt$
 $\Rightarrow U_a = -\frac{1}{R_1 \times C} \int U_e dt$



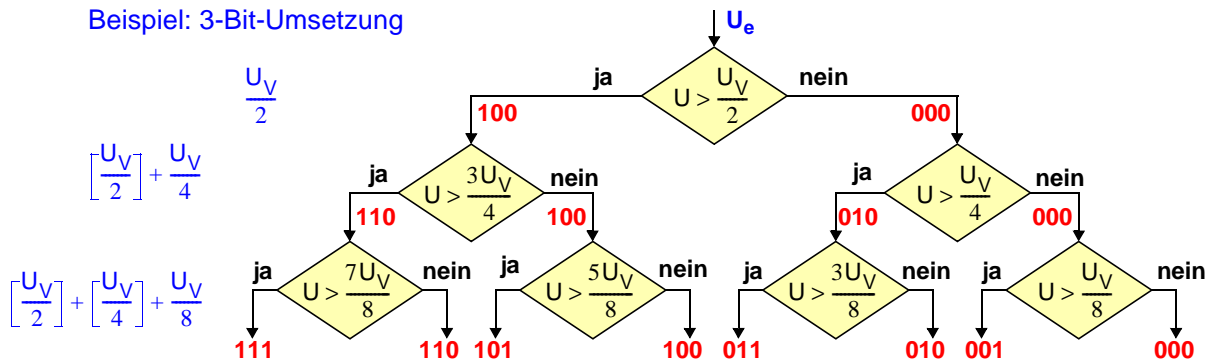
3. Datenübertragung

3.2.2 Analog/Digital-Wandler

Schrittweise Annäherung

- direktes Umsetzverfahren
 → Vergleich der Eingangsgröße mit einer Referenzgröße (ohne Zwischenumsetzung)
- bitserieller Vergleich, beginnend mit MSB
 → analog zum Abwiegen eines unbekanntes Gewichts mittels Balkenwaage
 (→ auch Wägewerfahren oder Successive Approximation Converter genannt)

Beispiel: 3-Bit-Umsetzung



- Genauigkeit bestimmt durch kleinste Referenzspannung $U_V / 2^n$ (Rest: Quantisierungsfehler - s.u.)

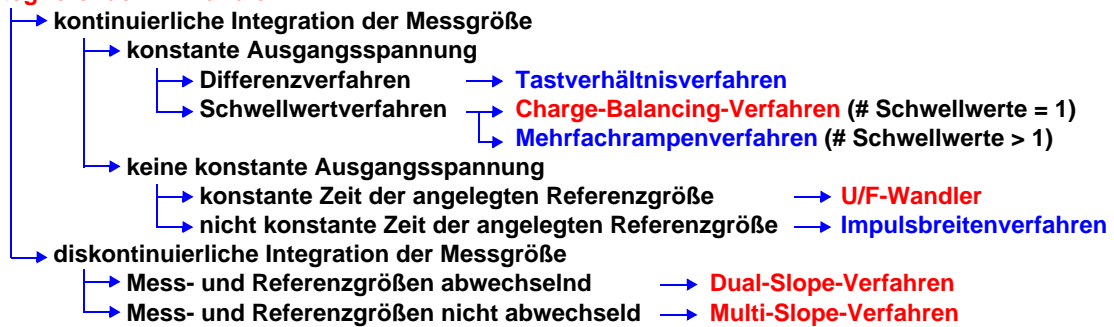


3. Datenübertragung

- Umsetzzeit steigt mit Auflösung (aber unabhängig von Eingangsspannung)
 - Ausgabe kann bitseriell während der Umsetzung schon ausgegeben werden
- Referenzspannungsquelle ist meist ein DA-Wandler mit paralleler Umsetzung (s.u.)
- Vergleich durch als Vergleichler beschalteter OV (s.o.)
- Umsetzzeit weitgehend abhängig von der Geschwindigkeit des digitalen Approximationsregisters
 - bei 400 ns Einstellzeit (→ diskreter Baustein) benötigt ein 10-Bit-Wandler etwa 5 µs
 - heutige integr. Schaltungen kommen auf ca. 50 ns für das Register, d.h. ca. < 1 µs Wandlerzeit bei 10 Bit

Integrierende Umsetzungsverfahren

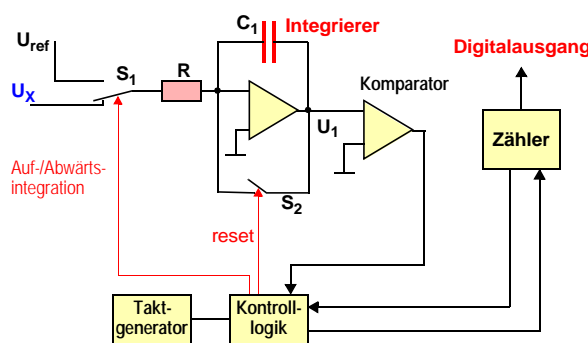
integrierende AD-Wandler



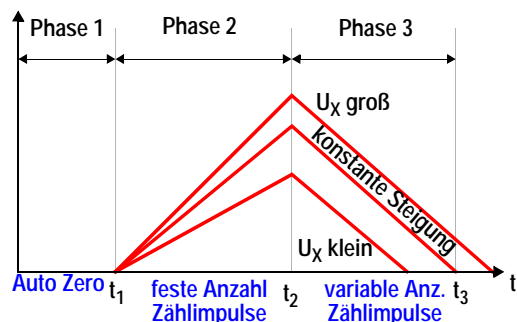
3. Datenübertragung

• Dual-Slope- und Multi-Slope-Verfahren

- Wandlung mittels Umsetzung in eine Zwischengröße (Integrationsspannung)



Zweirampenumsetzung



1. Phase: Nullabgleich (S_2 geschlossen - C_1 entlädt sich)
2. Phase: Aufwärtsintegration über feste Zeit T_2 (N Taktimpulse) mit Eingangsspannung U_x

$$U_{(T_2)} = \frac{1}{RC} \int_{t_1}^{t_2} U_x dt = U_x \frac{t_2 - t_1}{RC} \quad T_2 = t_2 - t_1$$



3. Datenübertragung

- 3. Phase:** Abwärtsintegration mit Referenzspannung (0 V oder entgegengesetztes Vorzeichen zu U_X) solange, bis $U_1 = U_{(T_3)} = 0V$ ($\rightarrow n$ Taktimpulse, variabel):

$$U_{(T_3)} = U_{(T_2)} - \frac{1}{RC} \int_{t_2}^{t_3} U_{ref} dt$$

Für $U_{(T_3)} = 0$ und $T_3 = t_3 - t_2$ gilt: $U_X \frac{T_2}{RC} = U_{ref} \frac{T_3}{RC}$

$$\Rightarrow U_X = \frac{T_3}{T_2} U_{ref} = U_{ref} \frac{nT}{NT} = \frac{U_{ref}}{N} n \quad (T = \text{Periodendauer des Zählers})$$

$$\Rightarrow U_X \sim n$$

- Vorteile:**
- relativ hohe Genauigkeit
 - unkritische Schaltungskomponenten
- Nachteile:**
- begrenzte Umsetzgeschwindigkeit (*i. Wesentl. abhängig von RC - ca. 40 ms bei 13 Bit*)
 - beschränkter Auflösungsbereich
(bei zu hoher Taktfrequenz und kleiner Eingangsspannung kommt es zu Ungenauigkeiten)

Vierrampen-AD-Wandler:

Erweiterung der Zweirampenintegration um eine weitere Doppelrampe mit fester Referenzspannung U_{ref2} zur Fehlerkompensation



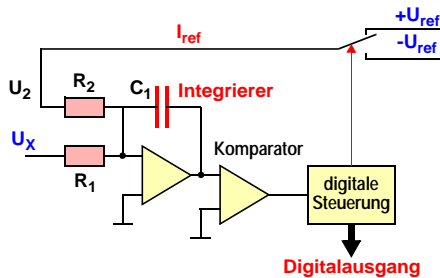
3. Datenübertragung

• U/F-Wandler

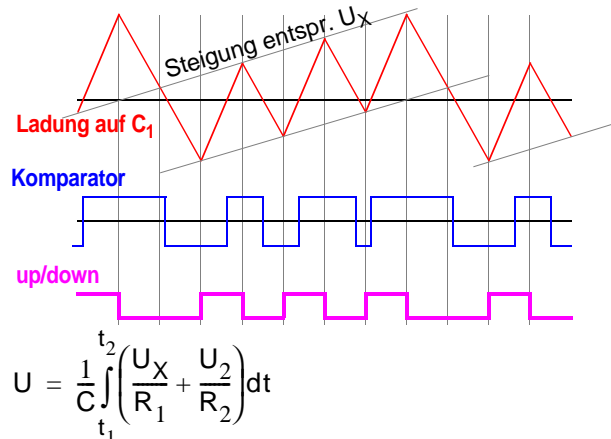
- ebenfalls Messung der Integrationszeit
- ausgegeben wird jedoch ein **Strom digitaler Impulse** der proportional zu U_X ist
(vgl. Delta-Modulation mit Spannungsimpulsen am Ausgang [s.u.])

• Charge-Balancing-Verfahren

- Kondensator eines Integrierers wird durch U_X geladen
 \rightarrow diese Ladung wird durch feste/bekannte Ladungsquanten kompensiert
(Anzahl benötigter Kompensationsladungsquanten entspricht U_X)



Für die Spannung am Integrierer gilt:
mit $U_2 = \pm U_{ref}$ (je nach Vergleicherentscheidung)



3. Datenübertragung

Zur Berechnung von U_x wird die Ladungsmenge gezählt, die notwendig ist, um das Aufladen von C_1 durch U_x auszugleichen:

während eines Zeitintervalls Δt (\rightarrow Zählertakt) wird C_1 durch U_x geladen: $Q_x = \frac{1}{C} U_x \Delta t$

und durch die Referenzspannung ausgeglichen: $Q_{ref} = \frac{1}{C} \frac{\pm U_{ref}}{R_2} \Delta t$

\rightarrow positive/negative Ausgleichladungsquanten werden mit einem Auf-/Abwärtszähler gezählt

\rightarrow 12-Bit-Wandler: vollständiger Messzyklus besteht aus 4096 Integrationsphasen

Die Ladungsmenge, die in N Taktzyklen zum Ausgleich notwendig ist: $\frac{U_x}{R_1} N \Delta t$

Im Gleichgewichtszustand muss gelten: $\Delta Q = 0$

$$\Rightarrow \text{Zählergebnis} \cdot \frac{U_{ref}}{R_2} \Delta t = \frac{U_x}{R_1} N \Delta t \quad \Rightarrow \text{Zählergebnis} = \frac{U_x R_2}{U_{ref} R_1} N = \text{const} \cdot U_x$$

Vorteil gegenüber Dual-Slope-Verfahren: **Wandlerdauer unabhängig von U_x** (aber recht lange)

Vorteil gegenüber schrittweiser Annäherung: **nur zwei Referenzspannungen**

Nachteil: **digitale Schaltungselektronik aufwändiger**

Parallele Wandler

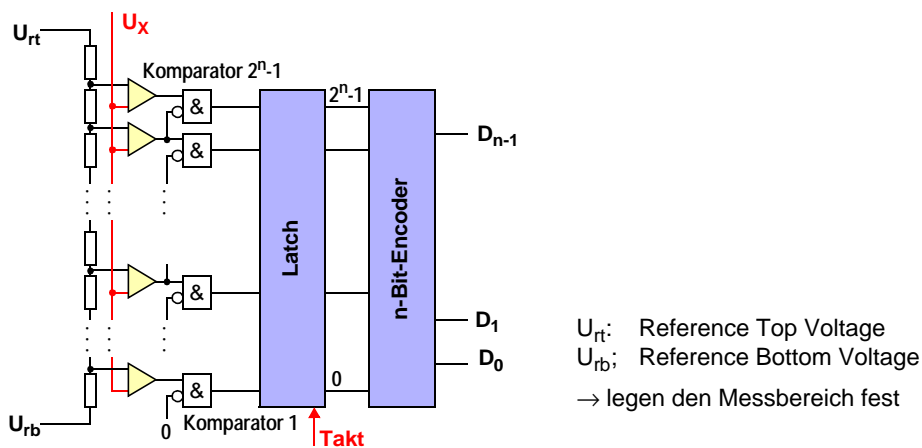
- bisher: Analogsignal durch analogen S&H-Verstärker abtasten und dann wandeln



3. Datenübertragung

\rightarrow alle Ansätze zu langsam für bspw. Videoanwendungen

\rightarrow notwendig: schnelle Wandler, die alle Bits des Ausgangs auf einmal berechnen



- Problem: bei n Bit Auflösung sind $2^n - 1$ Analogvergleicher nötig (typ. 16 Bit \rightarrow 65.535 Vergleicher)
 - \rightarrow heute:
 - integrierte Schaltungen mit sehr einfachen Komparatoren
 - Zusammenschaltung von Wandlern mit kleinerer Auflösung
 - Kombination aus paralleler und serieller Wandlerung

3.2.3 Digitale Ausgangsmodulation



3. Datenübertragung

Pulse-Code-Modulation (PCM)

- Bisher verwendetes Verfahren
- Am Ausgang liegt eine duale Kodierung des Analogwertes an
→ **PCM-Wandler**

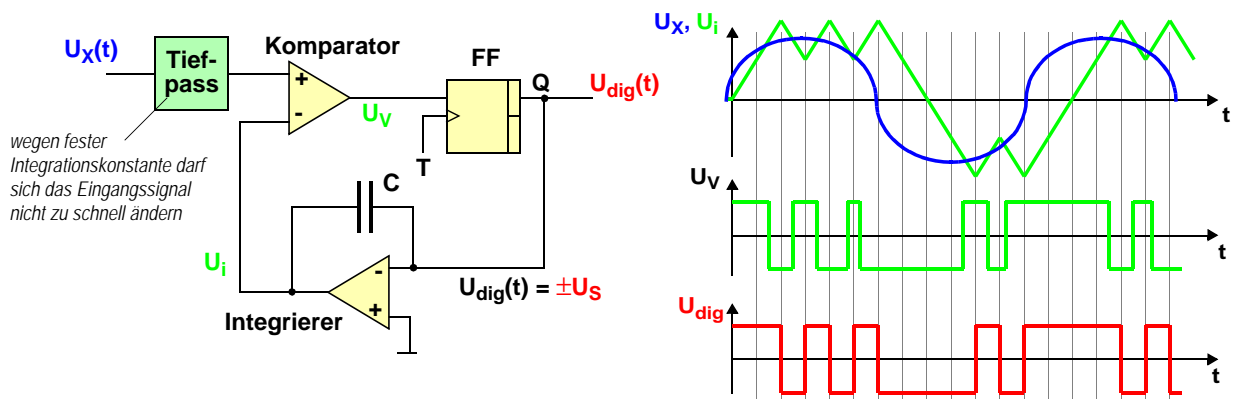
Delta-Modulation

- Wandler-Ausgangssignal kodiert eine **Differenz**, bestimmt aus den Werten des Eingangssignals zwischen zwei Abtastzeitpunkten.
- Vorteil: **Datenreduktion** (wichtig bei höheren Abtastraten - z.B. Audio, Video)
- Grundlegende Idee:
zu wandelnde Signale weisen i.d.R. hohe Datenredundanz auf
(→ *benachbarte Abtastwerte weichen nur geringfügig voneinander ab*)
⇒ redundanter Signalanteil (= Signalanteil, der sich über den wahrscheinlichen Signalverlauf rekonstruieren lässt)
- bestimmt durch einen **Prädiktor** - wird nicht quantisiert
⇒ lediglich der verbleibende Signalanteil (*Differenz zwischen Momentanwert und Vorhersagewert*)
muss kodiert werden (→ *Datenreduktion, Delta-Modulation*)



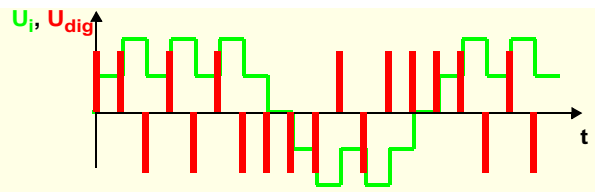
3. Datenübertragung

- einfache Delta-Modulation:



auch möglich:

Ausgabe von kurzen Impulsen der Dauer τ im Abstand T ($\tau \ll T$), die über einen Integrierer mit sehr kleiner RC-Konstante summiert werden. Hierbei ergibt sich dann für U_i eine Treppenkurve, da U_i mit jedem Impuls sehr schnell ansteigt/abfällt und zwischen den Impulsen konstant bleibt.



- **Vorhersagewert**: durch letzten Abtastwert bestimmter Signalwert U_i
- Kodierung: 1-Bit-Kodierung der Differenz von $U_X(t)$ und U_i
(nur „</>-Vergleich der beiden Werte)



3. Datenübertragung

- Eigenschaften:
 - je steiler die Flanke des Eingangssignals, desto mehr Ausgangsimpulse werden mit dem entsprechenden Vorzeichen ausgegeben
 - Amplitude des Digitalsignals bestimmt
 - „erlaubte“ Anstiegsgeschwindigkeit des Eingangssignals
(zu kleiner Wert folgt dem Eingangssignal nur langsam → Steigungsüberlastung, Overload Noise)
 - Quantisierungsrauschen (wächst mit der Amplitude des Digitalsignals)

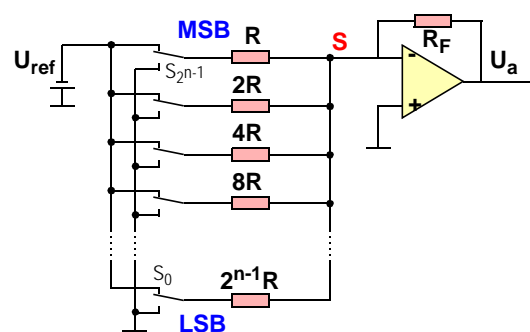
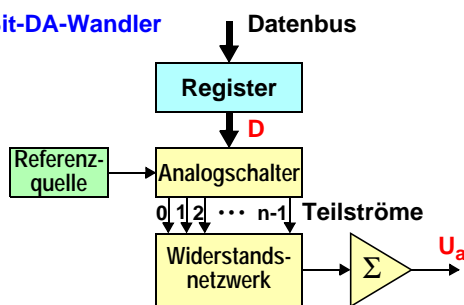


3. Datenübertragung

3.2.4 Digital/Analog-Wandler

Parallele Umsetzung

n-Bit-DA-Wandler



- synchrone Übernahme des digitalen Codes
- Registerausgänge schalten über Analogschalter die Referenzspannung auf die Bewertungswiderstände
 → Verhältnis der Teilströme/-spannungen entspricht der binären Darstellung
- Teilströme/-spannungen addieren sich am Eingang des Analogverstärkers (S)
 → Summe wird ggf. für Anwendung verstärkt

→ U_a ist proportional zur Dualzahl am Eingang

$$I_S = b_{n-1} \cdot U_{ref}/R + b_{n-2} \cdot U_{ref}/(2R) + \dots + b_0 \cdot U_{ref}/(2^{n-1}R) = \frac{U_{ref}}{2^{n-1}R} \sum_{i=0}^{n-1} b_i 2^i$$



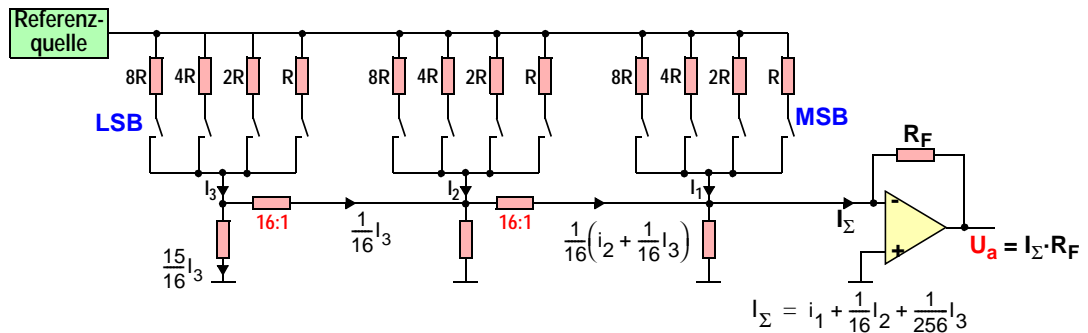
3. Datenübertragung

- Probleme:
 - Genauigkeit der absoluten Widerstandsverhältnisse
 (auch bei Wechsel 011...11 → 100...00 darf der Fehler max. 1 Bit groß sein
 ⇒ bei 16 Bit wird damit eine Widerstandstoleranz von 0,0015% für das MSB gefordert)
 - Driftverhalten aller Widerstände muss gleich sein
 - Schalter besitzen ebenfalls Widerstand R_{ON} , der zu Verfälschungen führt
 - jedes zusätzliche Bit am Eingang verdoppelt das Widerstandsverhältnis

IC-Technologie ermöglicht akzeptable Widerstandsverhältnisse von 20 : 1
 → max. 4 .. 5 Bit möglich

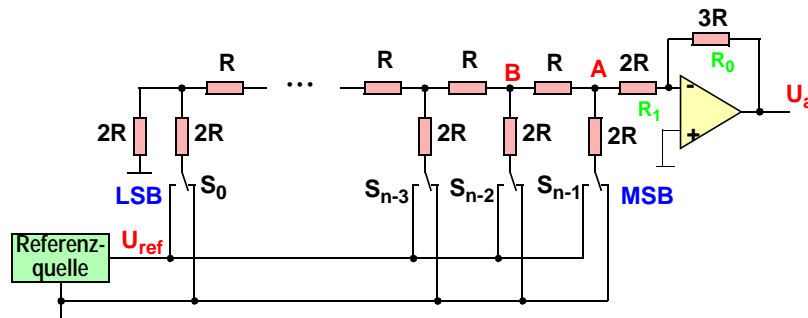
- mögliche Lösungen des Problems:

- a) 12-Bit-Wandler durch drei 4-Bit-Wandler, die über Stromteilerwiderstände im Verhältnis 16:1 gekoppelt sind



3. Datenübertragung

- b) R-2-R-Kettenleiter



→ nur noch zwei Widerstandswerte im Verhältnis 2:1 (lassen sich mit sehr kleinen Toleranzen fertigen)



3. Datenübertragung

- **Schaltungsanalyse:**

bei absolut linearen Kettenleitern können einzelne Kreise/Quellen getrennt betrachtet werden (→ Superpositionsprinzip)
 ⇒ Annahme: Spannungen der versch. Quellen (d.h. Schalter und Widerstände) sind voneinander unabhängig

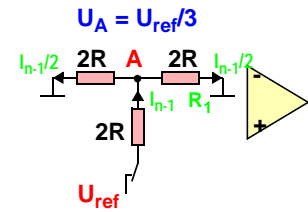
a) Sei **nur das MSB** eingeschaltet:

Widerstand zwischen A und Summenverstärker: $2R$
 Widerstand zwischen A und Referenzquelle: $2R$
 Widerstand der restlichen Schaltung: $2R$

⇒ $U_A = U_{ref}/3$

⇒ (da $I_{R0} = I_{R1}$): $U_a = U_{R0} = I \cdot 3R$; $U_{R1} = U_a = I \cdot 2R = U_{ref}/3$

⇒ (da $I = U_{ref}/3 \cdot 2 \cdot R$): $U_a = U_{ref}/2$

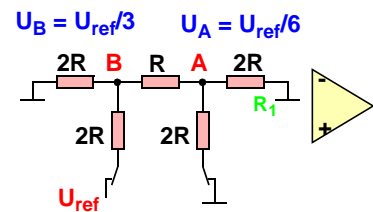


b) Sei **nur das Bit n-2** eingeschaltet:

Widerstand zwischen B und Referenzquelle: $2R$
 Widerstand zwischen B und (virt.) Masse: R

⇒ $U_B = U_{ref}/3$, $U_A = U_{ref}/6$

⇒ $U_a = U_{ref}/4$ (da gleiche Verstärkung)



Für alle anderen Teilspannungen gilt die gleiche **Gesetzmäßigkeit:**

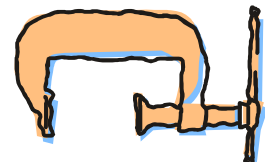
jedes Bit i erhöht den Strom in A (und damit die Ausgangsspannung) um den Anteil $\frac{1}{2^i}$



3. Datenübertragung

Zusammenfassung

- **Operationsverstärker** einsetzbar als analoge Vergleicher, Addierer, Subtrahierer und Integrierer
- **A/D-Wandlung** seriell durch schrittweise Annäherung bzw. Integration oder parallel möglich
- **Digitale Ausgangssignal** ist über Pulse-Code oder über Deltas modulierbar
- Parallele **D/A-Wandler** haben Probleme mit der Widerstandsgenauigkeit



Ausblick

- Welche Peripheriegeräte (insb. Sensorprinzipien) zur Interaktion mit der Umwelt stehen zur Verfügung?

