

# SIMULATIONSGESTÜTZTE STEUERUNG VERNETZTER SYSTEME – VOM SIMULATIONSMODELLE ZUR SPS

Simon Goy, Johannes Wagner

Wintersemester 2014/15



Teile der Vorlesungsinhalte und –unterlagen basieren auf Kooperationsprojekten zwischen dem Fachgebiet upp, dem IdE Institut dezentrale Energietechnologien und der Limón GmbH.



U N I K A S S E L  
V E R S I T Ä T



### **Trennende und fügende Fertigungsverfahren**

Prof. Dr.-Ing. Markus Böhm

### **Umformtechnik**

Prof. Dr.-Ing. habil. Kurt Steinhoff

### **Gießereitechnik**

Prof. Dr.-Ing. Martin Fehlbier

### **Umweltgerechte Produkte und Prozesse**

Prof. Dr.-Ing. Jens Hesselbach



### **Produktionsorganisation und Fabrikplanung**

Prof. Dr.-Ing. Sigrid Wenzel

# Fachgebiet umweltgerechte Produkte und Prozesse

## Lehrstuhlinhaber Prof. Dr.-Ing. Jens Hesselbach



### Forschungsschwerpunkte

1. Klima-, energie- und ressourceneffiziente Produktion
2. Modellierung, Simulation und Steuerung von Produktion und Umfeld
3. Dezentrale Energieversorgung und Erneuerbare Energien
4. Life Cycle Engineering

### Universitäre Lehre

- Life Cycle Engineering
- Fabrikbetriebslehre
- Messen von Stoff- und Energieströmen
- Energieeffiziente Produktion
- Unternehmensgründung
- Energiemanagementsysteme
- Dezentrale Energieversorgung in der Industrie
- Simulation und Steuerung von Produktions- und Energiesystemen



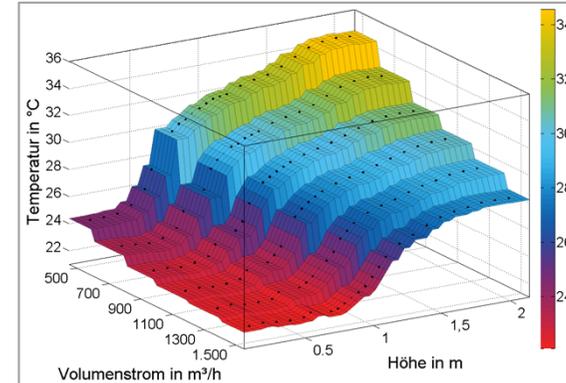
### Labore

- TGA und Klimalabor
- Kunststoff- und Thermolabor
- Druckluftlabor
- Dezentrale Energieversorgung



### Mobile Messtechnik

- Elektrische Energie
- Thermische Energie
- Wärmebildkameras
- Druckluft- und Leckage
- Luftströmungen
- uvw.



### Computer-aided engineering

- CAD (Pro Engineer)
- Dynamische Simulation (MATLAB)
- Messen, Steuern und Regeln (LabView)
- Ökobilanzierung (GaBi)

Weitere Informationen finden Sie im Internet unter:

<http://www.upp-kassel.de>

# Ansprechpartner

Simon Goy

R. 2112, Kurt-Wolters-Str. 3

+49.561.804.3944

[goy@upp-kassel.de](mailto:goy@upp-kassel.de)

Johannes Wagner

R. 2111, Kurt-Wolters-Str. 3

+49.561.804.3649

[wagner@upp-kassel.de](mailto:wagner@upp-kassel.de)

Weitere Informationen zur Lehrveranstaltung finden Sie im Internet unter:

<http://www.upp-kassel.de/studentyp/bachelor>

bzw.

<http://www.upp-kassel.de/studentyp/master>

Einleitung

# ALLGEMEINES UND ORGANISATORISCHES

# Struktur der Veranstaltung

Die Veranstaltung „Simulationsgestützte Steuerung vernetzter Systeme“ umfasst insgesamt 6 Präsenztermine:

- Laborpraktikum (16.- 20.02.2015): Aufbau einer Steuerung, Programmierung, Inbetriebnahme. 09:00 Uhr bis 17:00 Uhr.
- Präsentationen (noch bekanntzugeben): Vorstellung der Schwerpunktthemen durch die Studierenden. (Prüfungstermin!)
- Abgabe des Seminarberichts am 31.03.2015

# Prüfungsleistung

Vollständige Teilnahme an den Präsenzterminen ist verpflichtend!

Jeder Teilnehmer soll im Rahmen der Veranstaltung einen Projektbericht verfassen und eine 45 minütige Lehrinheit zu einem festgelegten Thema durchführen.

Die Projektberichte werden anschließen und einen Seminarheft zusammengefügt und allen Teilnehmern zur Verfügung gestellt.

Die Lehrinheit haben das Ziel der Lerngruppe Grundlagenwissen zu vermitteln und die Anwendung auf die programmierte Anlage zu verstehen.



# Seminarbericht

- Inhalt des Berichts ist eine Aufarbeitung des gewählten Themas. Es soll eine möglichst vollständige Darstellung des Themas auf ca. 25 Seiten erfolgen.
- Die Formatvorlage ist der Fachgebietswebsite „upp“ unter folgendem Link zu entnehmen: <http://www.upp-kassel.de/studium/vorlagen-fuer-wissenschaftliche-arbeiten>
- Es wird dabei auf eine wissenschaftliche Arbeitsweise Wert gelegt! Hinweise zur Struktur und Zitierweise sind dem Dokument „Richtlinien für die Gestaltung von wissenschaftlichen Arbeiten“ zu entnehmen.
- Die Abgabe des Berichts erfolgt sowohl in gedruckter als auch digitaler Form.

# Präsentation

Ziel der Präsentation ist es, das Thema der Lerngruppe in Form einer Unterrichtseinheit zu vermitteln. Die Dauer der Unterrichtseinheit sollte auf ca. **45 Minuten** ausgelegt werden. Es können dabei, neben Frontalunterrichtselementen, auch Übungsaufgaben oder Demonstrationen eingebunden werden.



# Zusammensetzung der Note

- 50 % zählt das Engagement während der praktischen Arbeit. Dabei wird die Motivation und Vorgehensweise bewertet. Zum Abschluss jedes Praktikumstags berichten die Teilnehmer über die Ergebnisse, Probleme und Erfolge.
- 25 % zählt der Seminarbericht.
- 25 % wird die Unterrichtseinheit bewertet.

Regelung und Steuerung

# GRUNDLAGEN STEUERN UND REGELN

# Einleitung

- Technische Anlagen haben mit der Zeit eine Komplexität angenommen, die eine Steuerung der Anlage per Hand unmöglich macht.
- Diese Entwicklung erfordert eine automatische Überwachung, Steuerung und Regelung der Anlage
- Die Funktion des Steuerns und Regelns werden unter dem Begriff „Automatisierung“ zusammengefasst
- Automatisierung ermöglicht den effizienten Betrieb von verfahrens- und fertigungstechnischen Produktionsprozessen, Energiesystemen, Transportanlagen, etc.

# Begriffe und Benennungen

- Ein **System** bedeutet eine abgegrenzte Anordnung von aufeinander einwirkenden Gebilden. Diese Anordnungen kann man sich durch Hüllflächen von ihrer Umgebung abgegrenzt denken.
- Das Verhalten des Systems wird über die Beziehung von **Größen** untereinander beschrieben, diese sind mittels Verbindungen übertragene Eigenschaften und Zustände.
- Die Darstellung von Informationen geschieht durch **Signale**. Das entspricht dem Wert oder dem Werteverlauf einer physikalischen Größe. Der **Signalparameter** ist diejenige Kenngröße des Signals, welche die Information trägt.
- In Systemen ist stets ein **Wirkungsweg** und eine **Wirkungsrichtung** vorhanden. Die Wirkungsweg ist dabei der Weg, längs dessen die einzelnen Wirkungen (Regeln oder Steuern) übertragen werden. Die Wirkungsrichtung ist die Richtung der Übertragung dieser Wirkung. Es ist dabei nicht notwendig, dass Wirkungsweg und –richtung mit denen der zugehörigen Massenströme (z.B. Wasser oder Luft) übereinstimmen.
  - So ist z.B. die Wirkungsrichtung der Störgröße häufig der Strömungsrichtung der Flüssigkeit/Gas entgegen gerichtet.

# Begriffe und Benennungen

- **Wirkungsplan:** beschreibt sinnbildlich die Struktur, in der die Teile eines Systems durch veränderliche Größen aufeinander einwirken. Seine wesentlichen Elemente sind: Wirkungslinien, Blöcke, Additionsstellen und Verzweigungen.
- Die **Wirkungslinien** symbolisieren die veränderlichen Größen, mit denen die Teilsysteme aufeinander einwirken oder sie stehen für Einwirkungen der Umgebung.
- Die Wirkungsrichtung geht vom verursachenden zu beeinflussten Teil und wird durch eine Pfeilspitze an der Wirkungslinie gekennzeichnet. In diesem Sinne unterscheiden man zwischen verursachenden Größen (**Eingangsgrößen**) und beeinflussten Größen (**Ausgangsgrößen**).
  - Die Wirkungsrichtung kann sich von der Richtung der Massen- und Energieflüsse unterscheiden.

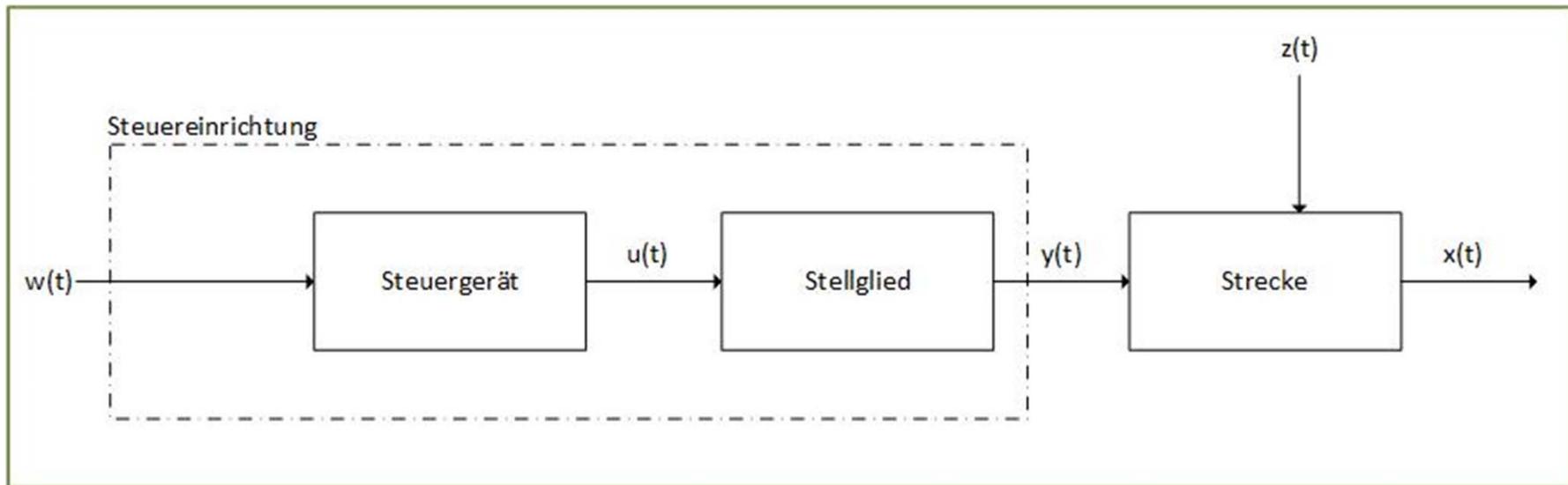
# Begriffe und Benennungen

- **Block:** symbolisieren (Teil-)Systeme oder auch (Teil-)Prozesse. Sie werden auch als **Übertragungsglieder** bezeichnet und als Rechtecke dargestellt. Wesentliches Merkmal ist ihr **Übertragungsverhalten**, welches die Verknüpfung zwischen Ein- und Ausgangsgrößen beschreibt.
  - Übertragungsglieder sind nicht unbedingt mit den Baugliedern identisch

# Steuerung gemäß DIN 19226 Teil 1

- Das Steuern, die Steuerung ist ein Vorgang in einem System, bei dem eine oder mehrere Größen als Eingangsgrößen andere Größen als Ausgangsgrößen aufgrund der dem System eigentümlichen Gesetzmäßigkeiten beeinflussen.
- Kennzeichen für das Steuern ist der offene Wirkungsweg oder ein geschlossener Wirkungsweg, bei dem die durch die Eingangsgrößen beeinflussten Ausgangsgrößen nicht fortlaufend und nicht wieder über dieselben Eingangsgrößen auf sich selbst wirken.

# Steuerung



# Steuerung – Beispiel

- Die Temperatur in einem Raum soll unabhängig von der Außentemperatur auf einem konstanten vorgegeben Wert gehalten werden.

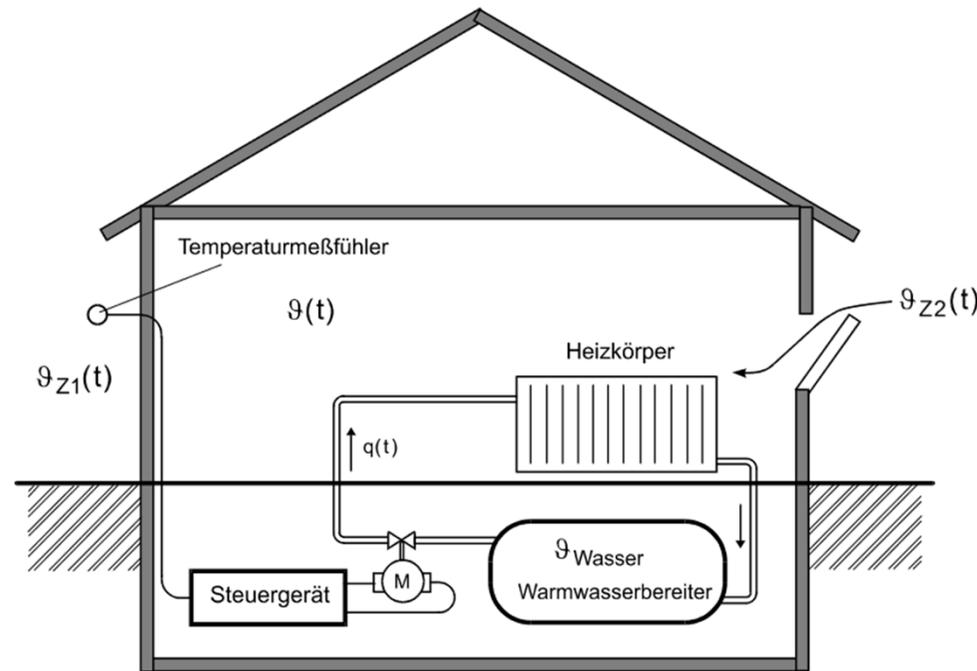
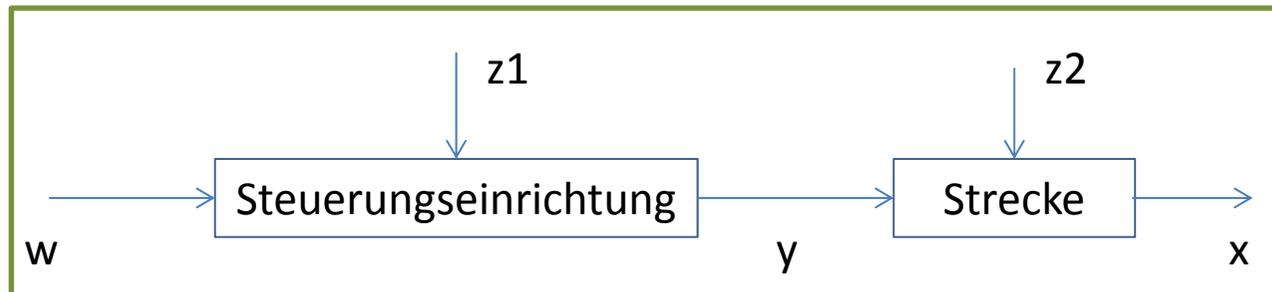


Bild 1.1.1: Außentemperaturgesteuerte Raumheizung

# Steuerung – Beispiel

- Bei der Steuerung der Raumtemperatur  $x$  beeinflusst die Außentemperatur  $w$  über einen Messfühler und ein geeignetes Stellglied zunächst die Heizwassertemperatur  $y$ , die dann die Raumtemperatur  $x$  ändert. Außerdem wirken noch Störgrößen  $z_1$  und  $z_2$  ein.



- Man erkennt deutlich die offene, rückwirkungsfreie Steuerkette. Rückwirkungsfrei bedeutet dabei, dass die Eingangsgröße  $w$  die Stellgröße  $y$ ,  $y$  beeinflusst aber nicht  $w$ .

Regelung und Steuerung

# REGELUNG

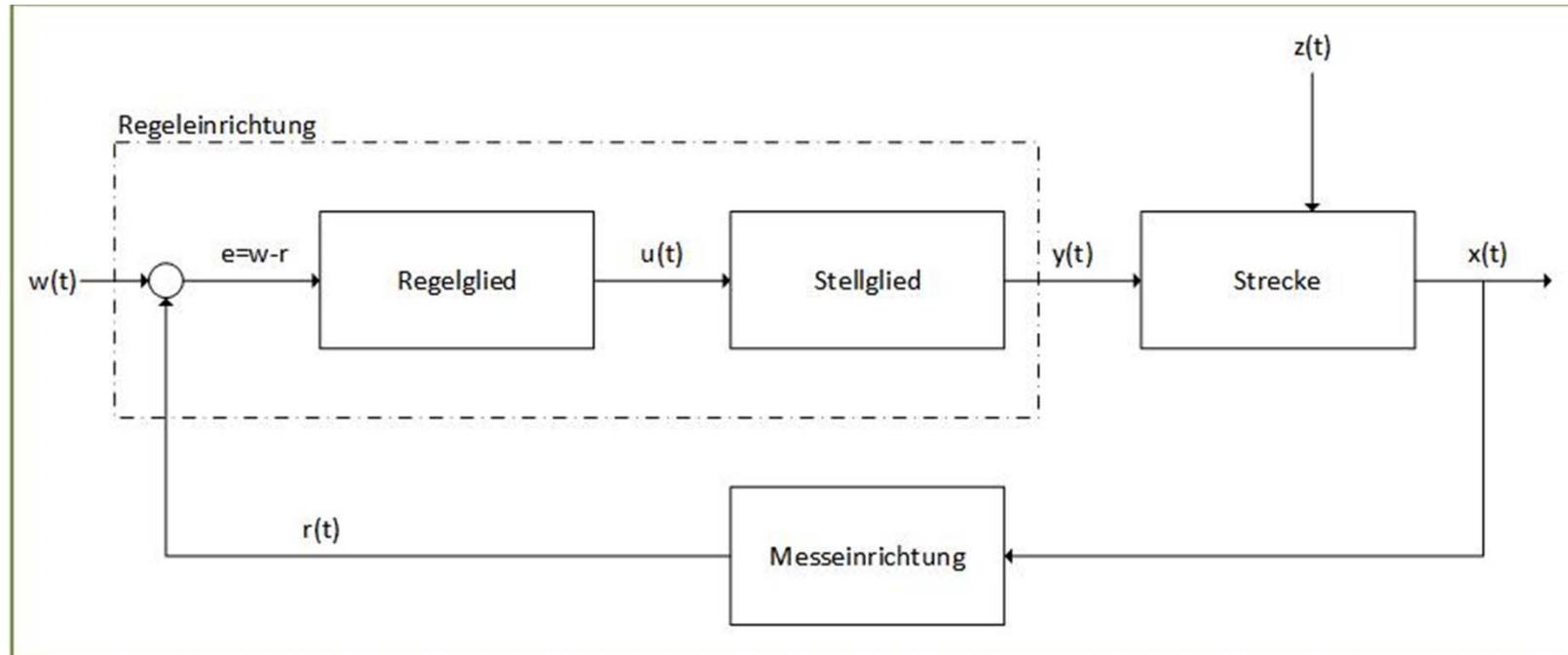
# Regelung

- Die Rückwirkung bei Steuerungen kann aufgrund externer Störungen bei zahlreichen Prozessen zu Problemen führen.
- Um die Ausgangsgröße  $x$  trotz vorliegenden Störungen an einen vorgegebenen Wert anzupassen, werden Regelungen verwendet.

# Regelung gemäß DIN 19226 Teil 1

- Das Regeln, die Regelung ist ein Vorgang, bei dem fortlaufend eine Größe, die Regelgröße (die zu regelnde Größe), erfasst, mit einer anderen Größe, der Führungsgröße, verglichen und im Sinne einer Angleichung an die Führungsgröße beeinflusst wird.
- Kennzeichen für das Regeln ist der **geschlossene Wirkungsablauf**, bei dem die Regelgröße im Wirkungsweg des Regelkreises fortlaufend sich selbst beeinflusst.

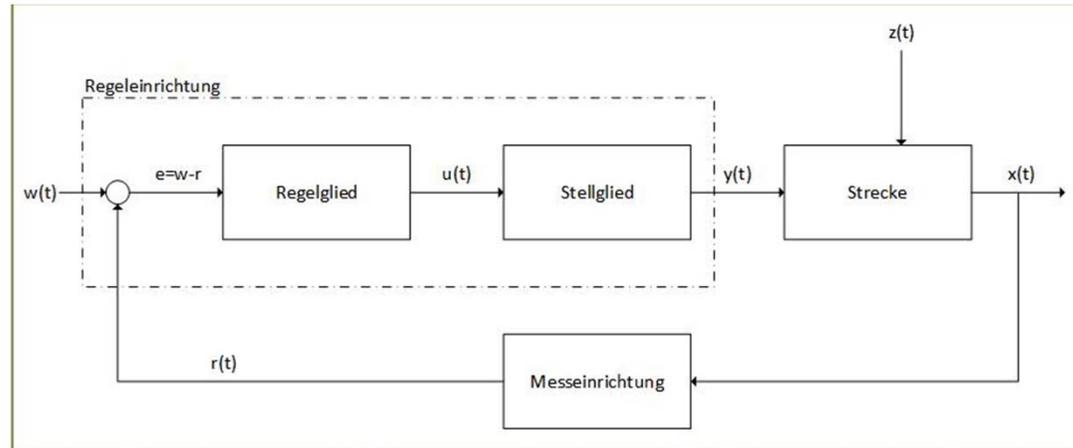
# Regelung



$w(t)$ : Führungsgröße  
 $x(t)$ : Ausgangs-/Regelgröße  
 $y(t)$ : Stellgröße  
 $z(t)$ : Störgröße

$e(t)$ : Regeldifferenz  
 $u(t)$ : Regler-/Steuergerätausgangsgröße  
 $r(t)$ : Rückführgröße

# Regelung



- Der dargestellte Wirkungsweg im geschlossenen Kreis, dem **Regelkreis**, ist deutlich zu erkennen. Darin besteht der wesentlichen Unterschied zu einer Steuerung.
- Durch die Regelung soll der Wert einer Größe, der Regelgröße  $x$ , unabhängig von unvermeidlichen Störgrößen, deren Zeitverläufe gewöhnlich nicht bekannt sind und oft nicht direkt messbar sind, einen vorbestimmten Verlauf annehmen. Die Regelgröße  $x$  soll zu allen Zeitpunkten möglichst gleich der Führungsgröße  $w$  sein.

# Regelungstechnik - Begriffe

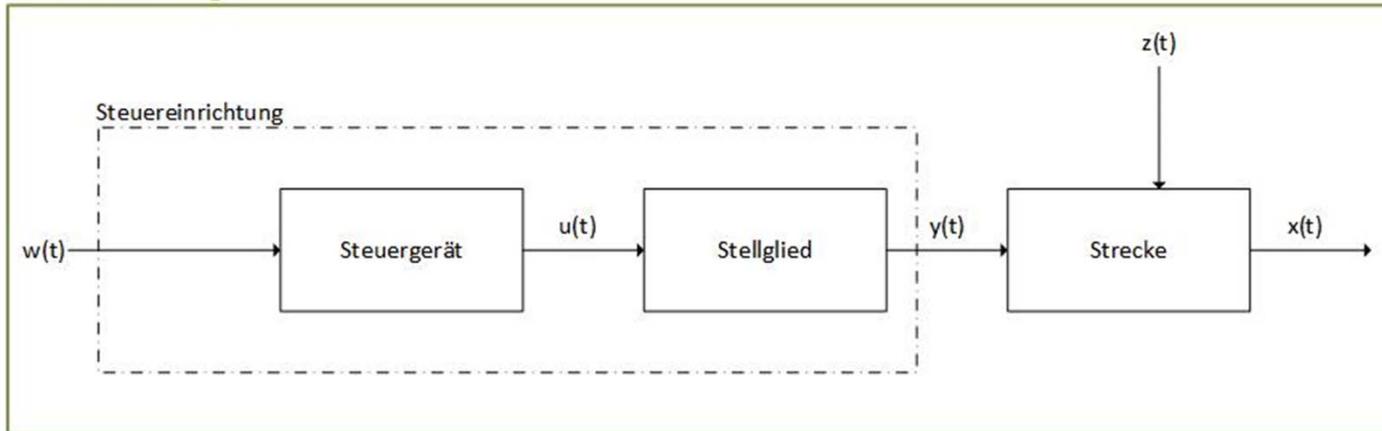
- Die **Strecke (Steuerstrecke, Regelstrecke)** ist der Teil des Systems, der geregelt bzw. gesteuert werden soll.
- Das **Stellglied** ist am Eingang der Strecke angeordnet und zur Regelstrecke gehörende Funktionseinheiten, welche dort zum Zwecke der Steuerung oder Regelung in einem Massenstrom oder Energiefluss eingreifen.
- Die **Regeleinrichtung, Steuereinrichtung** ist derjenige Teil, welcher die aufgabengemäße Beeinflussung der Strecke über das Stellglied bewirkt.
- Die **Messeinrichtung** ist die Gesamtheit aller zum Aufnehmen, Weitergeben, Anpassen und Ausgeben von Größen bestimmten Funktionseinheiten.

Regelung und Steuerung

# STEUERUNG VS. REGELUNG

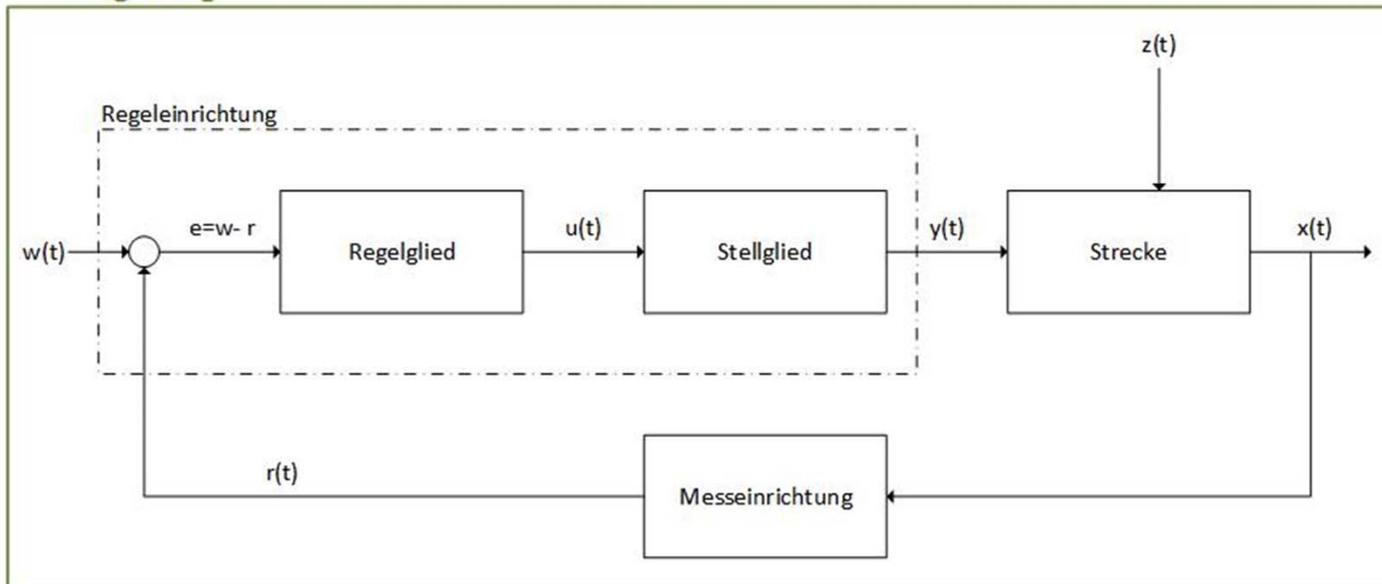
# Steuerung vs. Regelung

Steuerung



$w(t)$ : Führungsgröße  
 $x(t)$ : Ausgangs-/Regelgröße  
 $y(t)$ : Stellgröße  
 $z(t)$ : Störgröße

Regelung

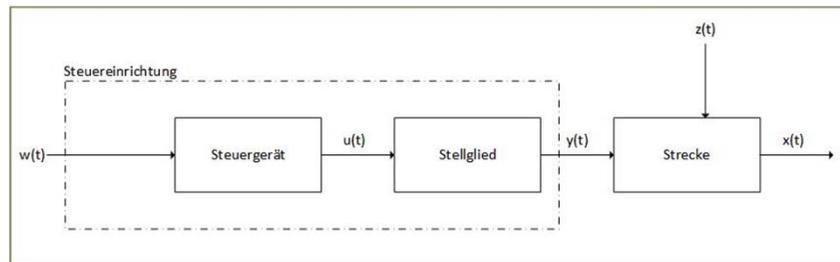


$e(t)$ : Regeldifferenz  
 $u(t)$ : Regler-/  
 Steuergerätausgangsgröße  
 $r(t)$ : Rückführgröße

# Steuerung vs. Regelung

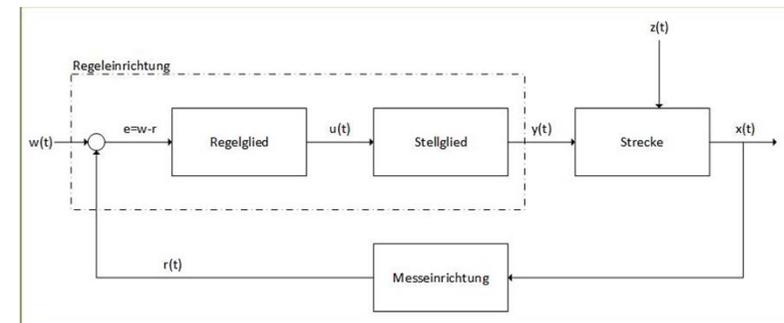
- Steuerung

- Offener Wirkungskreis
- Ausgangsgröße wirkt nicht auf Eingangsgröße zurück
- Störgrößen können nicht ausgeglichen werden



- Regelung

- Geschlossener Wirkungskreis
- Ausgangsgröße wirkt auf Eingangsgröße zurück
- Störgrößen können ausgeglichen werden



# Steuerung vs. Regelung

## Gegenüberstellung Regelung – Steuerung:

<b>Regelung:</b>	<b>Steuerung:</b>
<u>Merkmal:</u> laufende Rückführung der Ausgangsgröße der Strecke  (geschlossener Wirkungsweg)	<u>Merkmal:</u> bezüglich der Ausgangsgröße werden höchstens einzelne Ereignisse, z.B. "Endwert erreicht", zurückgemeldet (offener Wirkungsweg).
<u>Vorteil:</u> Es kann ein genauer, vorgegebener Wert oder Zeitverlauf der Regelgröße eingehalten werden. Störgrößen werden indirekt über ihre Wirkung auf die Regelgröße erfasst und ausgeglichen.	<u>Vorteil:</u> Meist geringerer Aufwand, eine Messung ist nur erforderlich, wenn die Störgröße erfasst und ausgeglichen werden soll. Weniger Stabilitätsprobleme.
<u>Nachteil:</u> Es ist gerätetechnisch ein höherer Aufwand, in jedem Fall eine Messung erforderlich. Ein Regelkreis kann schwingen oder sogar instabil werden (Beispiel Dusche)	<u>Nachteil:</u> Störungen werden nicht automatisch erfasst, oder es ist für jede Störung eine Messung erforderlich. Sollen beliebige Zwischenzustände angesteuert werden, so muss das Übertragungsverhalten der zu steuernden Strecke genau bekannt sein und in der Steuereinrichtung entsprechend nachgebildet werden Instabile Systeme können mittels einer Steuerung nicht beherrscht werden

Die Unterscheidung zwischen Regelung und Steuerung kann oft etwas spitzfindig werden, z.B. enthalten speicherprogrammierbare Steuerungen (SPS) oft auch Reglerbausteine. In Englisch kennt man für Steuerung und Regelung nur den einen Begriff "control". Zur Verdeutlichung spricht man im Englischen von der Regelung als „closed loop control“

Regelung und Steuerung

# REGELKREIS

# Regelkreis

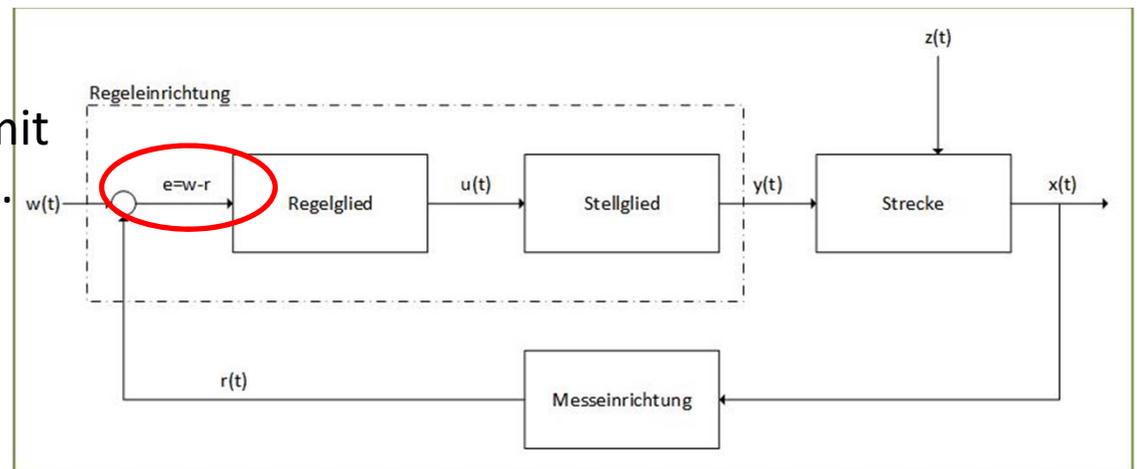
- Im systemtheoretischen Sinne ist die beschriebene Regelungsstruktur eine Kreis-Anordnung miteinander gekoppelter Übertragungssysteme.

# Regelkreis - Fachbegriffe

- **Regelgröße  $x$  (Istwert):** In einem Regelkreis wird diejenige Prozessgröße mit  $x$  bezeichnet, deren Zustand geregelt werden soll.
  - In der Verfahrenstechnik ist dies zumeist ein physikalischer (z.B. Temperatur, Druck, Durchfluss) oder chemischer Zustand (z.B. pH-Wert, Härte usw.)
- **Führungsgröße  $w$  (Sollwert):** Diese Größe gibt den Wert vor, den die zu regelnde Prozessgröße einnehmen soll (Sollwert). Ihr physikalischer Wert –in Form einer mechanischen oder elektrischen Größe (Kraft, Druck, Strom, Spannung etc.)- wird im geschlossenen Regelkreis mit der Regelgröße verglichen.
  - Die Führungsgröße muss nicht dieselbe Dimension wie die Regelgröße haben, sondern ist eine Messgröße!
- **Rückführgröße  $r$ :** Die aus der Messung der Regelgröße hervorgegangene Größe, die zum Regeleingang auf das Vergleichsglied zurückgeführt wird.

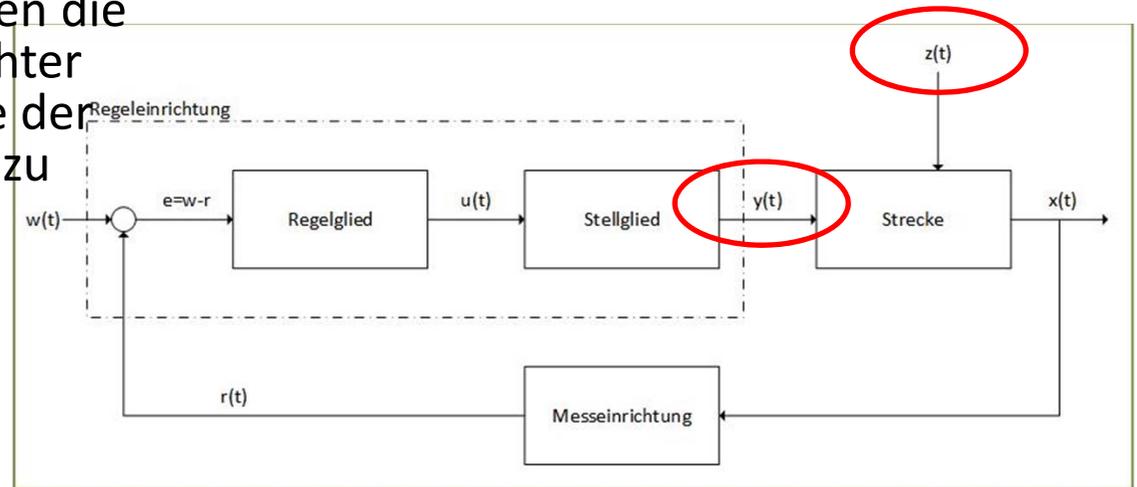
# Regelkreis – Fachbegriffe

- **Regeldifferenz  $e = w - x$ :**
  - Die Eingangsgröße  $e$  des Regelgliedes ist die vom Vergleichsglied errechnete Differenz aus Führungsgröße und Regelgröße. Wird die Wirkung der Messeinrichtung mit berücksichtigt, muss mit  $e = w - r$  gerechnet werden.
- **Regelabweichung  $x_w = x - w$ :**
  - Per Definitionsgleichung hat die Regelabweichung denselben Betrag wie die Regeldifferenz  $e$ , jedoch mit umgekehrtem Vorzeichen.



# Regelkreis – Fachbegriffe

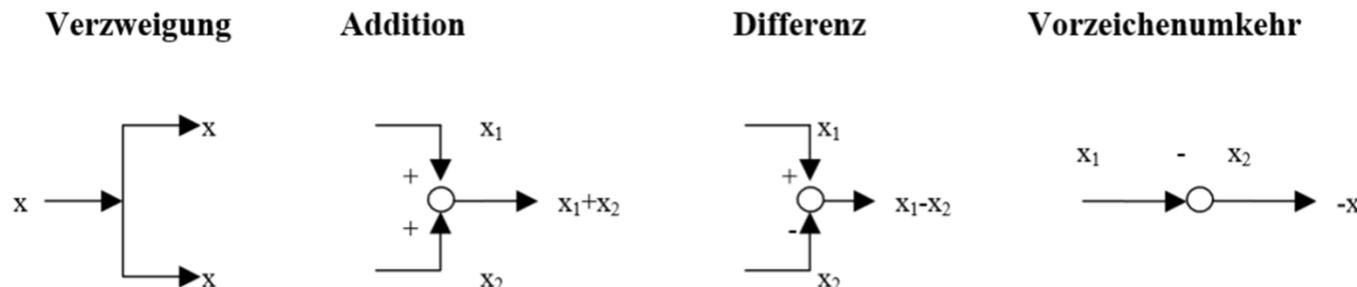
- **Stellgröße  $y$ :**
  - Ist die Ausgangsgröße der Regeleinrichtung und Eingangsgröße der Regelstrecke. Sie wird vom Regler bzw. vom Steller generiert. Sie ist abhängig von der Einstellung der Regelparameter sowie vom Wert der Regelabweichung.
- **Störgröße  $z$ :**
  - Störgrößen wirken auf den Regelkreis und beeinflussen die Regelgröße in unerwünschter Weise. Es ist eine Aufgabe der Regelung, diesen Einfluss zu kompensieren.



# Regelkreis – Komponenten

- **Vergleichsglied:**

- Es werden der aktuelle Messwert und der Wert der Führungsgröße miteinander verglichen. Die Differenz der beiden Größen wird als die *Regeldifferenz*  $e$  bezeichnet. In den meisten Fällen handelt es sich bei diesen Größen um eine Messspannung.



# Regelkreis – Komponenten

- **Regelglied:**
  - Ist das Herzstück einer Regelung, im wesentlichen unterscheidet man zwischen stetigen und unstetigen Reglern.
  - Der Regler wertet die Regeldifferenz, also die Information, ob und wie weit die Regelgröße vom aktuellen Sollwert abweicht aus. Daraus wird die Reglerausgangsgröße abgeleitet, durch die in letzter Konsequenz die Regelgröße beeinflusst wird.
- **Steller:**
  - Hier werden die Informationen der Reglerausgangsgröße, wie die Regelgröße beeinflusst werden soll, umgesetzt und so eine Änderung der Stellgröße bewirkt.
- **Stellgerät/Stellglied:**
  - Dieses Glied des Regelkreises, das in Abhängigkeit der Stellgröße die Regelgröße (mehr oder weniger stark) beeinflusst.
- **Regelstrecke/Prozess:**
  - Ist der Teil des Systems, in dem sich die zu regelnde Größe befindet.

# Informationsfluss im Regelkreis

- Die Regelgröße wird fortlaufend (oder auch in bestimmten Zeitintervallen) von einer Messeinrichtung gemessen. Die so gewonnene Rückführgröße wird im Vergleichsglied mit der Führungsgröße verglichen und daraus die Regeldifferenz gebildet. Aus dieser bestimmt das Regelglied auf unterschiedliche Weise die Regelausgangsgröße, die über Steller, Stellgröße und Stellglied auf die Regelstrecke übertragen wird bzw. einwirkt und sich so auf die Regelgröße auswirkt.

Regelung und Steuerung

# REGELSTRECKEN

# Regelstrecken

- In der Regelungstechnik wird eine zu regelnde Strecke in erster Linie durch ihr Zeitverhalten charakterisiert
- Dieses bestimmt, mit welchem Aufwand und mit welcher Güte sich eine regelungstechnische Aufgabe lösen lässt.
- Um dieses Zeitverhalten – die Streckendynamik – darzustellen, verwendet man häufig die so genannte Sprungantwort der Regelstrecke.

# Sprungantwort

- Das dynamische Verhalten einzelner Regler und auch Regelstrecken wird anhand von Sprungantworten dargestellt.
- Sprungantworten werden erzeugt, indem ein Reglertyp mit einer sprunghaften Veränderung des Eingangssignals beaufschlagt wird.
- Jeder Reglertyp reagiert anders auf diesen Sprung und erzeugt damit eine charakteristische Sprungantwort

# Klassifizierung von Regelstrecken

- Regelstrecken werden nach dem zeitlichen Verlauf Sprungantwort charakterisiert:
  - P-Regelstrecken (proportionales Verhalten)
  - I-Regelstrecken (integrales Verhalten)
  - Strecken mit Totzeit
  - Strecken mit Energiespeichern

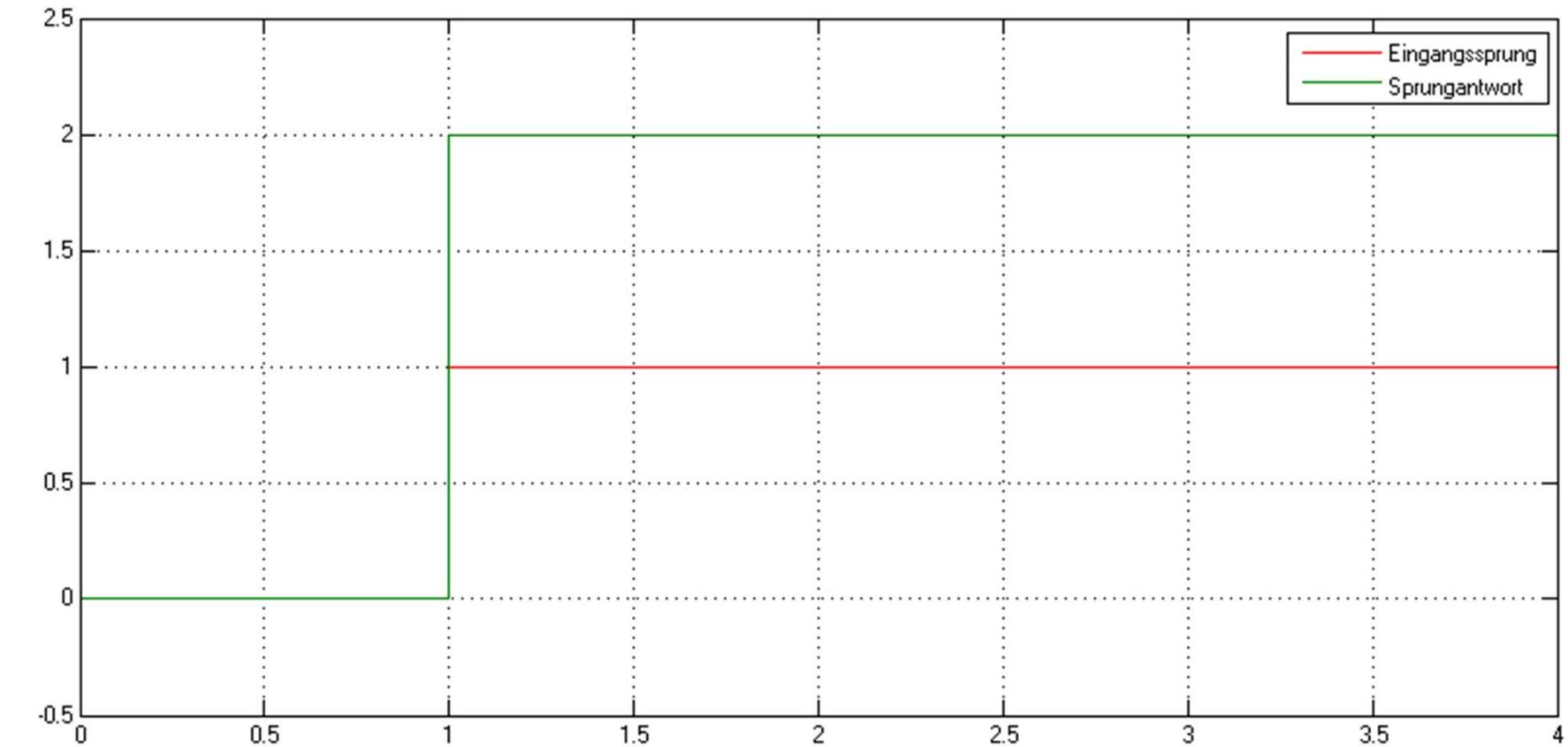
# P-Regelstrecken

- Bei Regelstrecken mit Proportionalverhalten ändert sich die Regelgröße  $x$  proportional mit der Stellgröße  $y$
- Hierbei erfolgt die Änderung der Regelgröße ohne zeitliche Verzögerung.
- Beispiel: Durchflussbegrenzendes Ventil (Der Durchfluss ändert sich (fast) sofort mit der Änderung der Ventilstellung)

# P-Regelstrecken

- P-Regelstrecken sind Strecken mit Ausgleich: Es stellt sich ein neuer **Gleichgewichtszustand** ein
- P-Regelstrecken sind **ohne Verzögerung** in dem neuen Gleichgewichtszustand
- Nur theoretisch möglich, in der Realität werden immer endliche Zeiten für die Übertragung benötigt. Sind die Übertragungszeiten aber so gering, dass sie sich regelungstechnisch nicht auswirken, spricht man von P-Regelstrecken

# Sprungantwort einer P-Regelstrecken



Time offset: 0

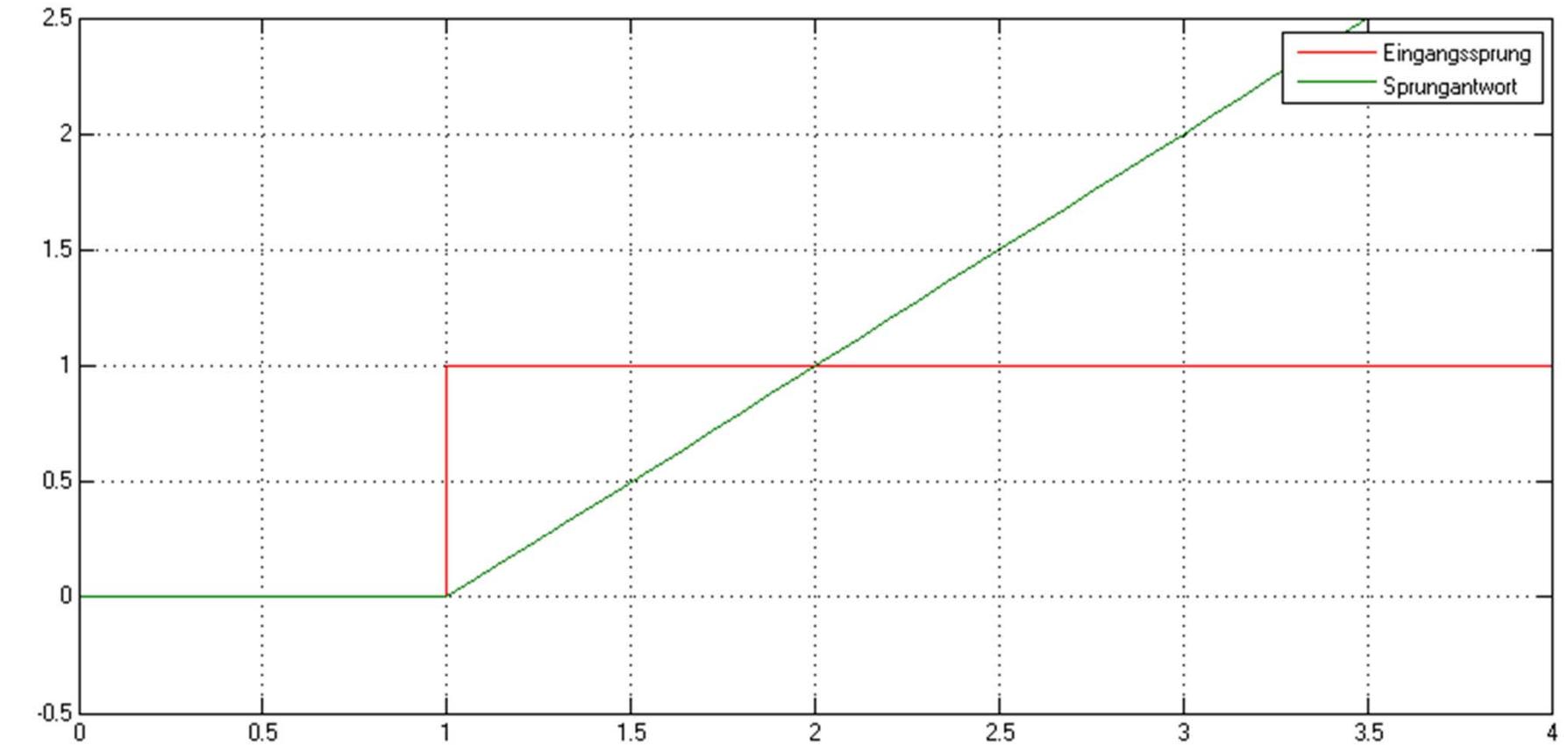
# I-Regelstrecken

- Die Regelgröße  $x$  verhält sich bei I-Regelstrecken Integral zu der Stellgröße  $y$
- Ist die Stellgröße  $y$  ungleich 0, verändert sich die Regelgröße kontinuierlich
- Beispiel: Füllstandsregelung
  - Bei einem Behälter mit Zu- und Ablauf stellt sich eine konstante Füllhöhe ein
  - Verändert sich der Zu- oder Abfluss, fällt oder steigt der Flüssigkeitsspiegel
  - Der Pegel ändert sich umso schneller, je größer die Differenz zwischen Zu- und Ablauf ist
  - Der Pegel ändert sich so lange, bis die systemischen Randbedingungen erreicht sind (Behälter leer/Behälter voll)

# I-Regelstrecken

- I-Regelstrecken sind Strecken **ohne Ausgleich**
- Ist die Stellgröße ungleich 0, stellt sich **kein Gleichgewichtszustand** ein
- Die Regelgröße wird sich so lange ändern, bis Randbedingungen das integrale Verhalten begrenzen (z.B. min./max. Druck)

# Sprungantwort einer I-Regelstrecke

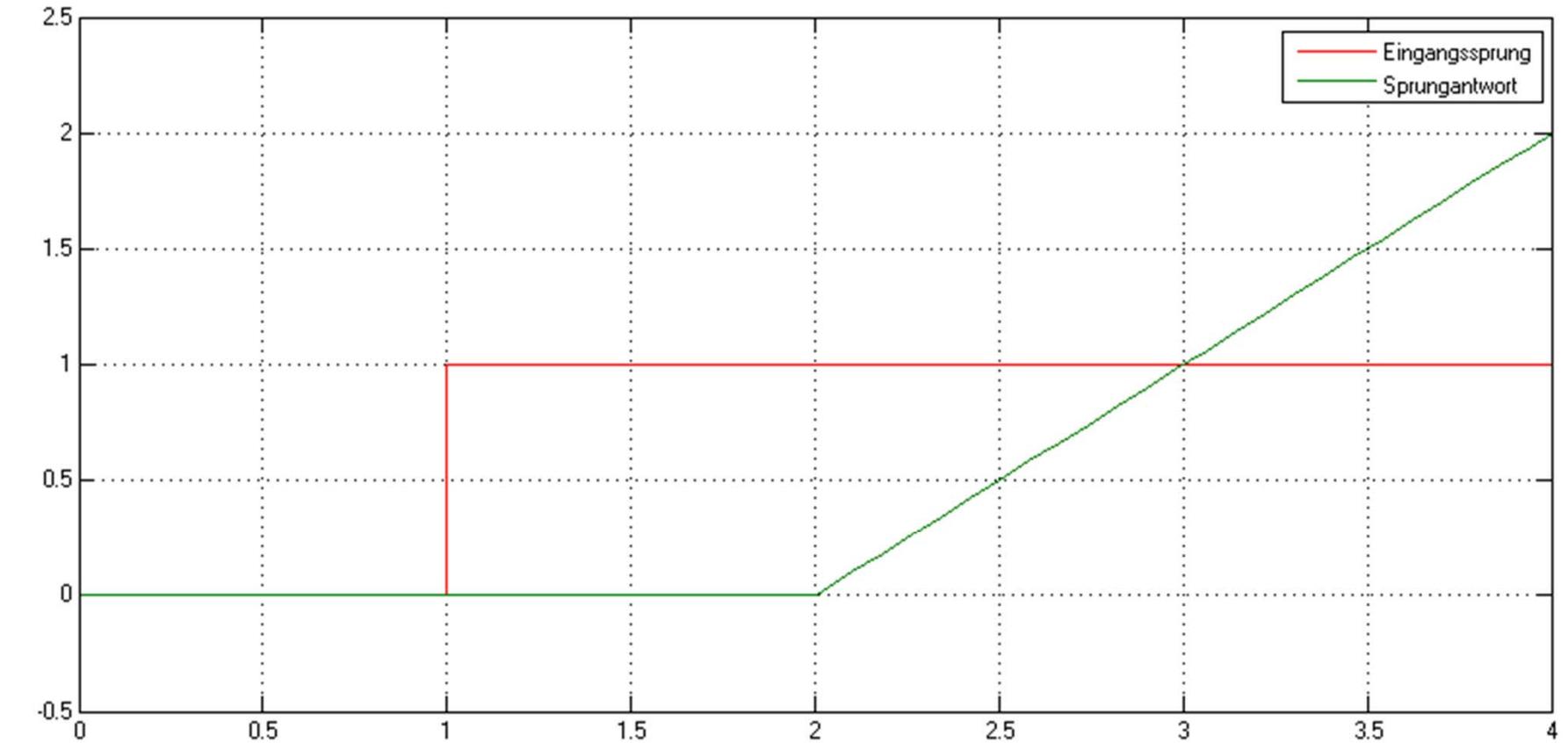


Time offset: 0

# Regelstrecken mit Totzeit

- Regelstrecken mit Totzeiten weisen eine zeitliche Verzögerung der Sprungantwort auf
- Die Änderung der Stellgröße wirkt sich erst nach einer Totzeit  $T_t$  auf die Regelgröße aus
- Diese Regelstrecken sind in der Regelungstechnik schwierig zu behandeln, da es durch die Verzögerungen schnell zu Schwingungen des Regelkreises führen kann
- Totzeiten werden oft durch verschiedene Stellglieder im Regelkreis verursacht (z.B. Spiel in einer Getriebeübersetzung; Schaltzeiten von Schützen)

# Sprungantwort einer I-Regelstrecke mit Totzeit: 1s



Time offset: 0

# Regelstrecken mit Totzeit

- Regelstrecken mit Totzeiten weisen eine zeitliche Verzögerung der Sprungantwort auf
- Die Änderung der Stellgröße wirkt sich erst nach einer Totzeit  $T_t$  auf die Regelgröße aus
- Diese Regelstrecken sind in der Regelungstechnik schwierig zu behandeln, da es durch die Verzögerungen schnell zu Schwingungen des Regelkreises führen kann
- Totzeiten werden oft durch verschiedene Stellglieder im Regelkreis verursacht (z.B. Spiel in einer Getriebeübersetzung; Schaltzeiten von Schützen)

# Regelstrecken mit Energiespeichern

- Bei Regelstrecken mit Energiespeichern stellt sich die Regelgröße nach einer Änderung der Stellgröße nur allmählich ein
- Die Regelstrecke reagiert also verzögert (träge) auf eine Stellgrößenänderung
- Beispiel: Temperaturregelung eines Raumes
  - Stellgröße (Heizung) ändert sich sprungartig
  - Regelgröße (Temperatur) ändert sich nur allmählich
  - Oft sind in einem System mehrere Energiespeicher vorhanden (Heizkessel, Wasser, Heizkörper, Isolierungen, Raumluft, Wände, etc.)

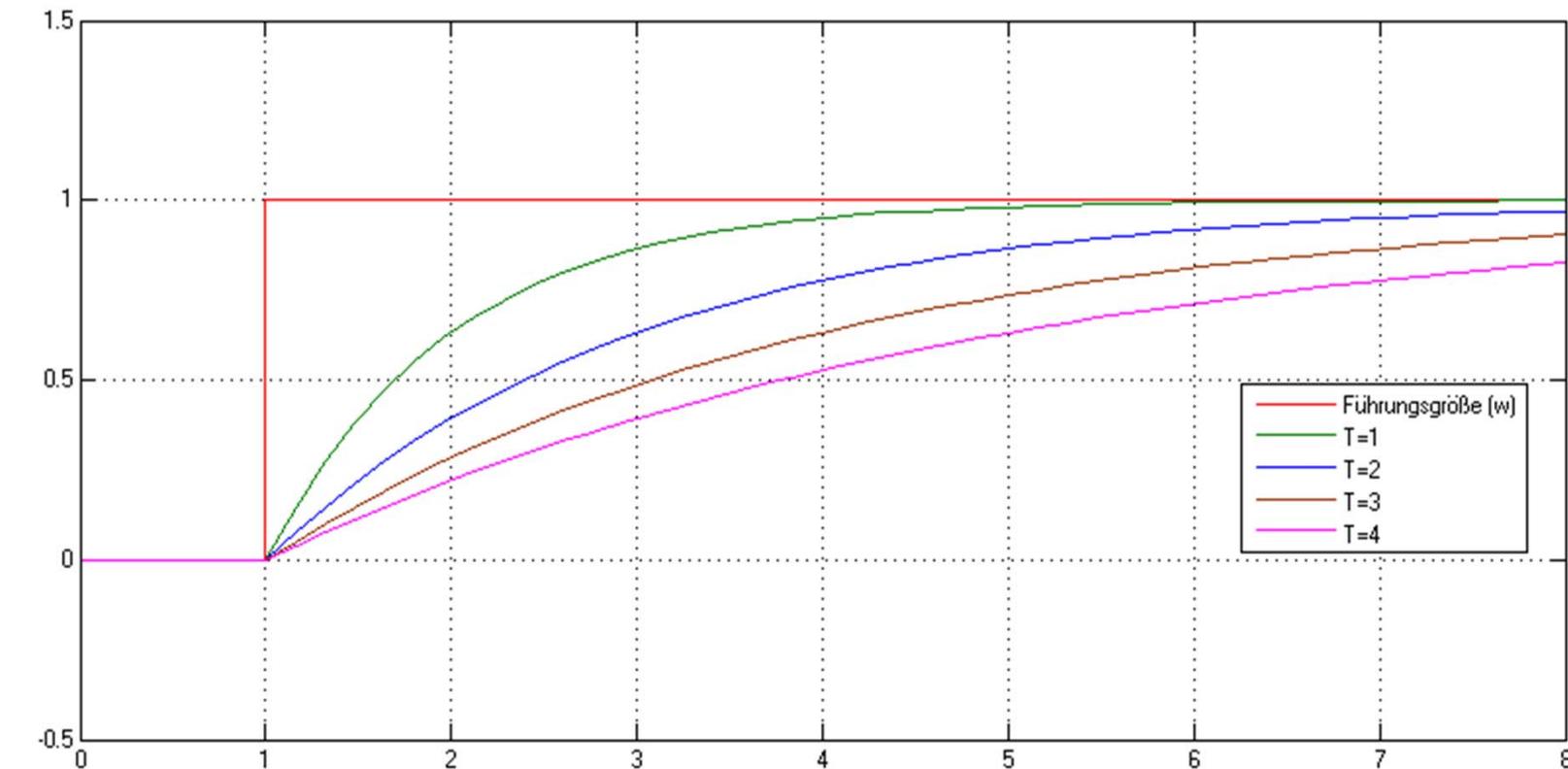
# Regelstrecken mit Energiespeichern

- Regelstrecken mit Energiespeichern werden nach der Anzahl ihrer Energiespeicher charakterisiert
- Eine Strecke 1. Ordnung enthält einen Energiespeicher, eine Strecke 2. Ordnung zwei Energiespeicher, usw.
- Dieses Verhalten wird auch als  $PT_1$ -,  $PT_2$ -Verhalten bezeichnet
- Die Konstante  $T_i$  gibt dabei das Maß der Verzögerungswirkung an
- Mathematisch wird das Verhalten nach dieser Gleichung beschrieben:

$$x + T \frac{dy}{dt} = k * y$$

# Regelstrecken mit Energiespeichern: $PT_1$ -Glieder

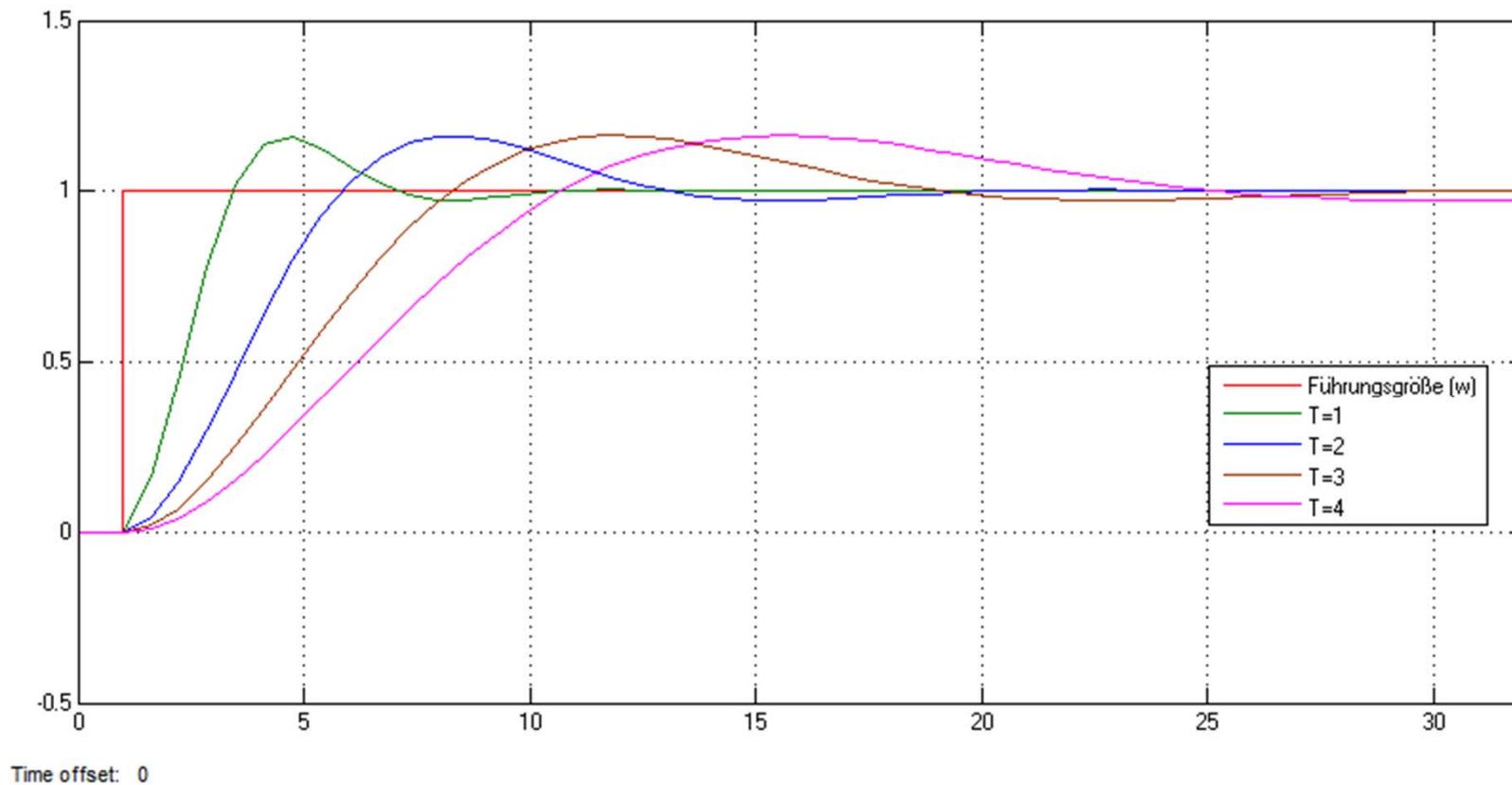
- $PT_1$ -Verhalten einer Sprungantwort mit verschiedenen Zeitkonstanten



Time offset: 0

# Regelstrecken mit Energiespeichern: $PT_2$ -Glieder

- $PT_2$ -Verhalten einer Sprungantwort mit verschiedenen Zeitkonstanten



Regelung und Steuerung

# AUFGABEN DER REGELUNG

# Aufgaben der Regelung

- Die Regelgröße  $x$  (z.B. Temperatur, Drehzahl) soll einen vorgegebenen Wert annehmen (Vorgabe über Führungsgröße  $w$ ) und diesen entgegen allen Störeinflüssen halten.
- Hier ist ferner zu unterscheiden zwischen einer **Festwert- und einer Folgeregelung:**
  - Festwertregelung: Führungsgröße  $w$  ist konstant (stimmt im wesentlichen für Tempomat)
  - Folgeregelung: Führungsgröße  $w$  ändert sich laufend (z.B. Vorschubantrieb, Werkzeugmaschine, Industrieroboter)

# Aufgaben der Regelung

- Genauer betrachtet ergeben sich folgende Forderungen an die Regelung (ggf. je nach Regelungsaufgabe unterschiedlich wichtig):
- dynamisches Verhalten: Regelgröße folgt Änderungen der Führungsgröße schnell, regelt Störungen schnell und ohne große Abweichungen aus. Es treten hierbei keine oder gut gedämpfte Schwingungen auf. Sonderfall: Schwingungsdämpfungsregelung. Die Regelung hat hier die Aufgabe, die Dämpfung eines Systems zu vergrößern (z.B. "aktiver Dämpfer" bei Fahrwerk von PKW, ICE)
- stationäre Genauigkeit: Nach Ablauf des Regelvorganges soll Regelgröße möglichst genau mit der Führungsgröße übereinstimmen.
- Begrenzungen, Belastungen der Stellglieder: Regelvorgang erfordert oft schnelle und große Änderungen der Stellgröße, die das Stellglied unter Umständen gar nicht leisten kann (z.B. Ventil ist vorher am Anschlag) oder es ist ggf. stark belasten.

# Aufgaben der Regelung

- Schnittstellen zur Umgebung: Hier ist zum einen die Sollwert Vorgabe (Einstellung der Führungsgröße) zu sehen. Diese kann entweder von Hand erfolgen oder durch ein übergeordnetes System, z.B. einen Leitreechner. Ein anderer Gesichtspunkt ist die Installation, Einstellung, Änderung, Wartung und Überwachung der Regelung. Dies erfordert einen übersichtlichen, flexiblen Aufbau und übersichtliche Anzeigen (Benutzeroberfläche)
- Anpassung an verändertes Streckenverhalten: Das Verhalten des Reglers muss dem Streckenverhalten angepasst sein. Bei neueren Regelungskonzepten erfolgt eine automatische Anpassung des Reglers an die Strecke (**adaptive Regelung**).
- Sicherheit, Zuverlässigkeit, Robustheit: Auch bei extremen Betriebs- und Umweltbedingungen und Bedienungsfehlern keine Katastrophen.
- Wirtschaftlichkeit: Können die Anforderungen auch mit einer weniger aufwendigen Regelung erfüllt werden, genügt ggf. auch eine Steuerung?

Regelung und Steuerung

# TYPISCHE REGLERTYPEN – UNSTETIGE REGLER

# Unstetige Regler

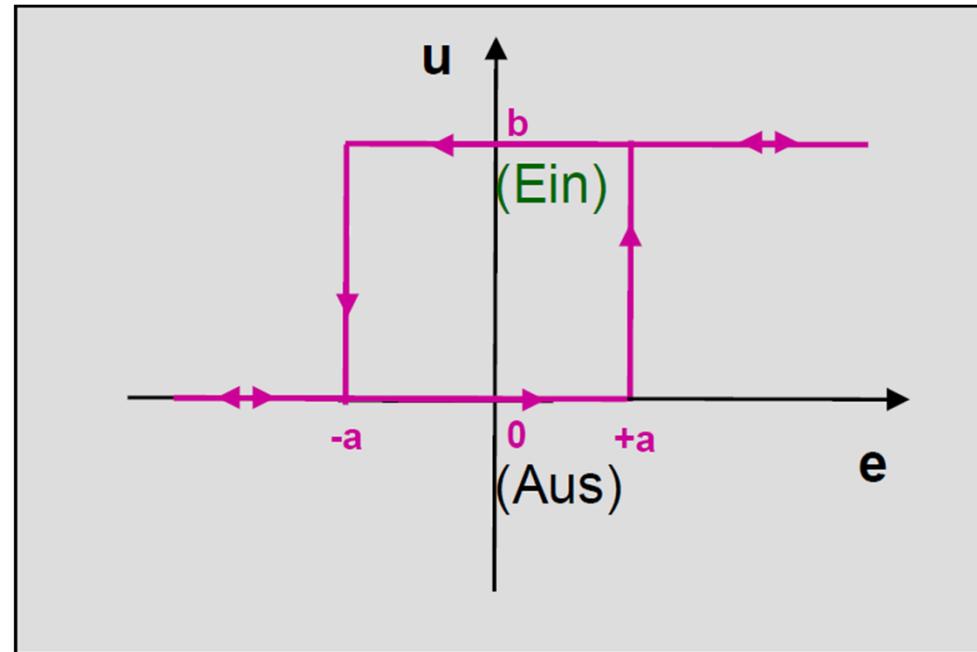
- Die Stellgröße  $y$  springt zwischen wenigen diskreten Werten, so dass die Energie- oder Massenzufuhr zur Strecke nur schrittweise verändert werden kann
- Je nach Anzahl der verschiedenen Zustände der Stellgröße, unterscheidet man zwischen Zwei-, Drei-, Mehrpunktreglern
- Unstetige Regler können mit sehr einfachen, schaltenden Stelleinrichtungen arbeiten
- Werden oft auch als schaltende Regler bezeichnet

# Zweipunktregler

- Bildet die einfachste Form der un stetigen Regler, die wesentliche Eigenschaft besteht darin, dass sie nur zwei Zustände kennen: „An“ und „Aus“
- Werden häufig eingesetzt wenn darum geht die Regelung möglichst einfach zu halten und die absolut präzise Einhaltung des Sollwertes nicht zwingen erforderlich ist.
  - z.B. Wassertemperatur im Heizkessel, darf durchaus um einige Grad schwanken und es ist einfacher den Kessel ein- bzw. auszuschalten als die Brennstoffzufuhr zu regeln
  - Temperaturregelung mit Hilfe eines Bimetall z.B. Bügeleisen
    - Bei geschlossenem Kontakt –Bimetall und Sollwertesteller berühren einander- fließt Strom, die Heizplatte erwärmt sich. Der Bimetallstreifen in der Nähe der Heizplatte erwärmt und dehnt sich mit aus, bis die Energiezufuhr schließlich unterbrochen wird

# Zweipunktregler

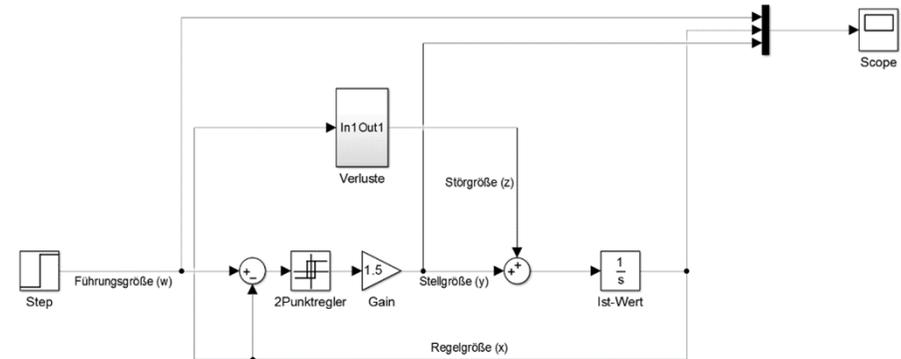
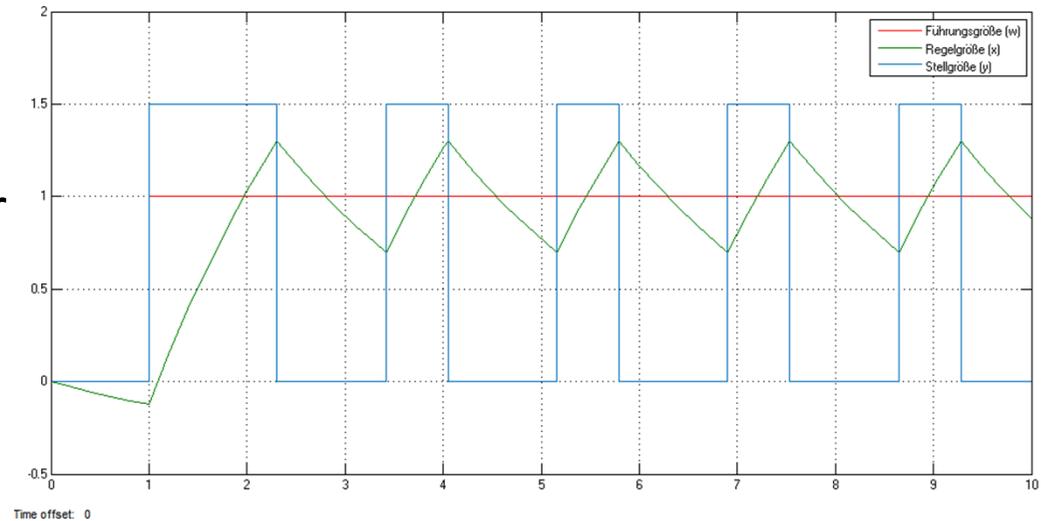
- Um den Sollwert exakt einzuhalten, muss theoretisch unendlich schnell hintereinander ein- und ausgeschaltet werden, deshalb besitzt der Zweipunktregler eine *Hysterese*.
- Die Hysterese stellt eine Art Umgebung um den Sollwert dar, innerhalb der der Istwert schwanken darf, begrenzt durch eine Minimal- und Maximalwert.
- Erst wenn der Istwert einen der beiden Werte unter- bzw. überschreitet, reagiert die Regelung.



Quelle: Claudi, A. „Regelungstechnik RE<sup>2</sup>“ (2010)

# Zweipunktregler

- MATLAB-Beispiel:
  - Führungsgröße  $w = 1$
  - Stellgröße  $y=1,5$ , wenn der Regler eingeschaltet ist
  - Regler schaltet ein bei  $e > 0,3$
  - Regler schaltet aus bei  $e \leq 0,3$  ( $y=0$ )
  - Eine veränderliche Störgröße  $z$  wirkt auf die Regelstrecke
- Die Hysterese ist damit der Bereich zwischen 0,7 und 1,3



# Zweipunktregler - Vorteile

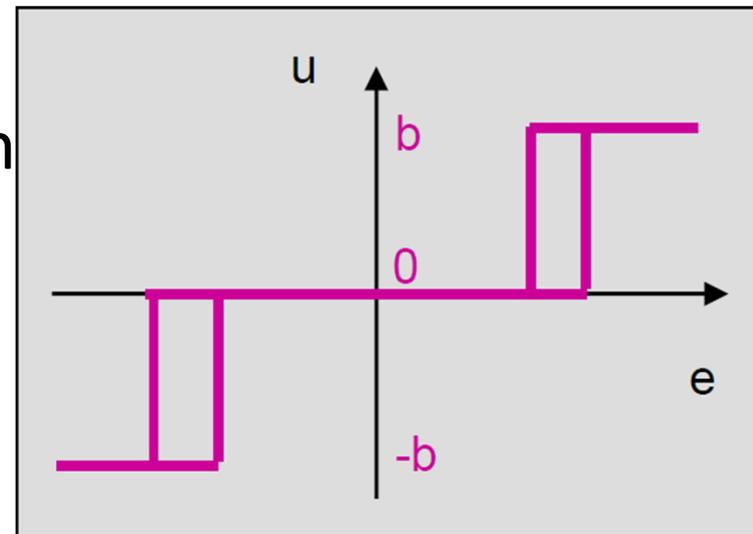
- Wegen des einfachen Aufbaus ist der Regler billig
- In mechanischer Ausführung braucht der Regler keine Hilfsenergie
  - z.B. Bimetallschalter (Thermostat)
- Anwendung:
  - Temperaturregelung bei trägen Strecken
    - (Raumheizung, Kühlschrank, Boiler, LötKolben, Kochherd, Bügeleisen,....)

# Dreipunktregler

- Stellt die zweite wichtige Klasse der diskreten Reglertypen dar
- Die Stellgröße kann drei verschiedene Werte zur Beeinflussung der Regelgröße annehmen:
  - Positive Beeinflussung, keine Beeinflussung und negative Beeinflussung
- Bei der Temperaturregelung lassen sich damit nicht nur –wie bei einem Zweipunktregler– die Zustände „Aus“ und „Heizen“ verwirklichen, sondern es besteht zusätzlich die Möglichkeit auf „Kühlen“ zu schalten
- Der Dreipunktregler übernimmt demnach die Funktion von zwei miteinander gekoppelten Zweipunktreglern, die bei unterschiedlichen Zuständen schalten.

# Dreipunktregler

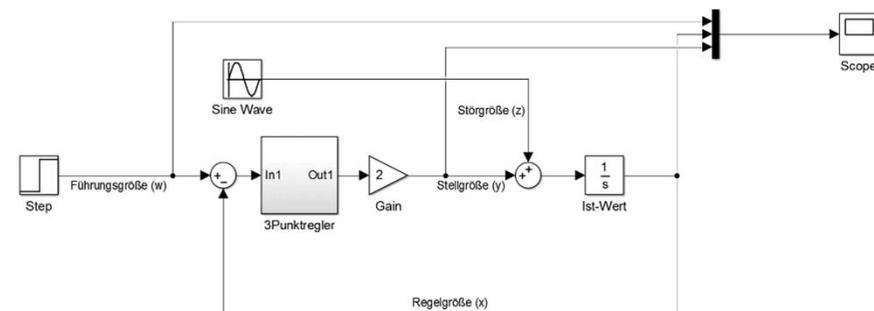
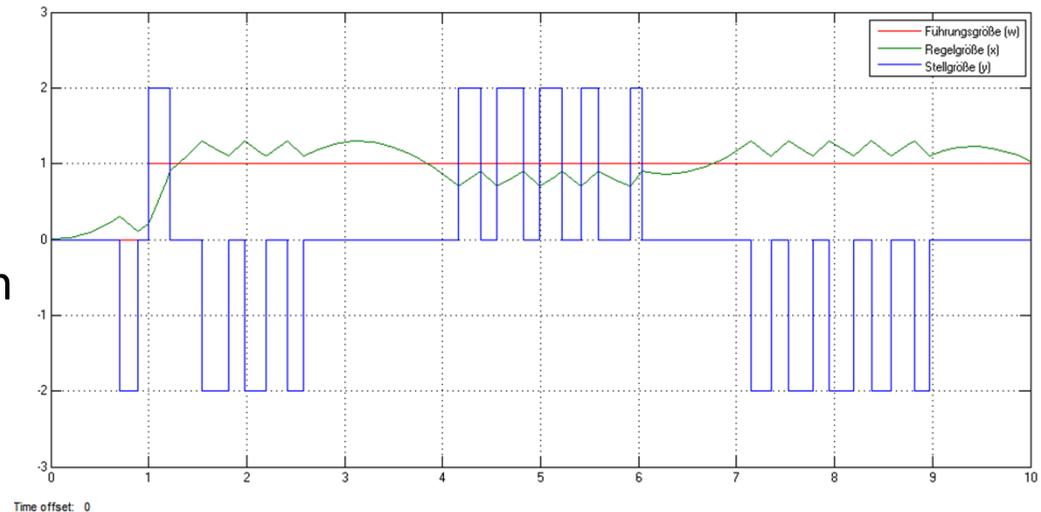
- Dreipunktregler werden häufig in Verbindung mit elektrischen Stellantrieben verwendet
- Motor mit den Zuständen
  - Linkslauf (z.B. Öffnen), Rechtslauf(z.B. Schließen) und Aus kann ein Stellmotor ein Ventil in jede beliebige Hubstellung fahren
- Klimaanlage mit den Zuständen
  - Heizen, Kühlen und Aus



Quelle: Claudi, A. „Regelungstechnik RE<sup>2</sup>“ (2010)

# Dreipunktregler

- MATLAB-Beispiel :
  - Führungsgröße  $w = 1$
  - Stellgröße  $y$  kann die Werte  $-2,5/0/2,5$  einnehmen
  - Die erste Hysterese deckt den Bereich  $-0,3 < e < -0,1$  ab (positives  $y$ )
  - Die zweite Hysterese deckt den Bereich  $0,1 < e < 0,3$  ab (positives  $y$ )
  - Eine veränderliche Störgröße (Sinuskurve)  $z$  wirkt auf die Regelstrecke
- Die Regelung kann also in beide Richtungen wirken und lässt Abweichungen von  $e \pm 0,3$  zu



Regelung und Steuerung

# TYPISCHE REGLERTYPEN – STETIGE REGLER

# Stetige Regler

- Die Stellgröße kann jeden beliebigen Wert innerhalb eines Stellbereiches annehmen.
- Jeder Regler und jedes Stellglied verrichtet Arbeit und benötigt dafür Energie. Wird diese Energie in pneumatischer, elektrischer oder hydraulischer Form von außen zugeführt, spricht man von Regler mit Hilfsenergie.

# P-Regler (Proportionalregler)

- Die Änderung der Stellgröße  $\Delta y$  verhält sich immer proportional zu der Änderung der Regeldifferenz  $e$  ( $y \sim e$ )
- Daraus ergibt sich, dass der P-Regler
  - Ohne eine Verzögerung auf eine Regelabweichung reagiert
  - Nur dann eine Stellgröße erzeugt, wenn eine Regelabweichung  $e$  vorliegt
- P-Regler werden bei einfach zu regelnden Strecken eingesetzt, bei denen eine bleibende Regelabweichung nicht stört. Mit geringem Aufwand erreicht man hier ein stabiles und dynamisches Regelverhalten.

$$y = K_p \cdot e$$

großes  $K_p$  bewirkt  
großen Stelleingriff

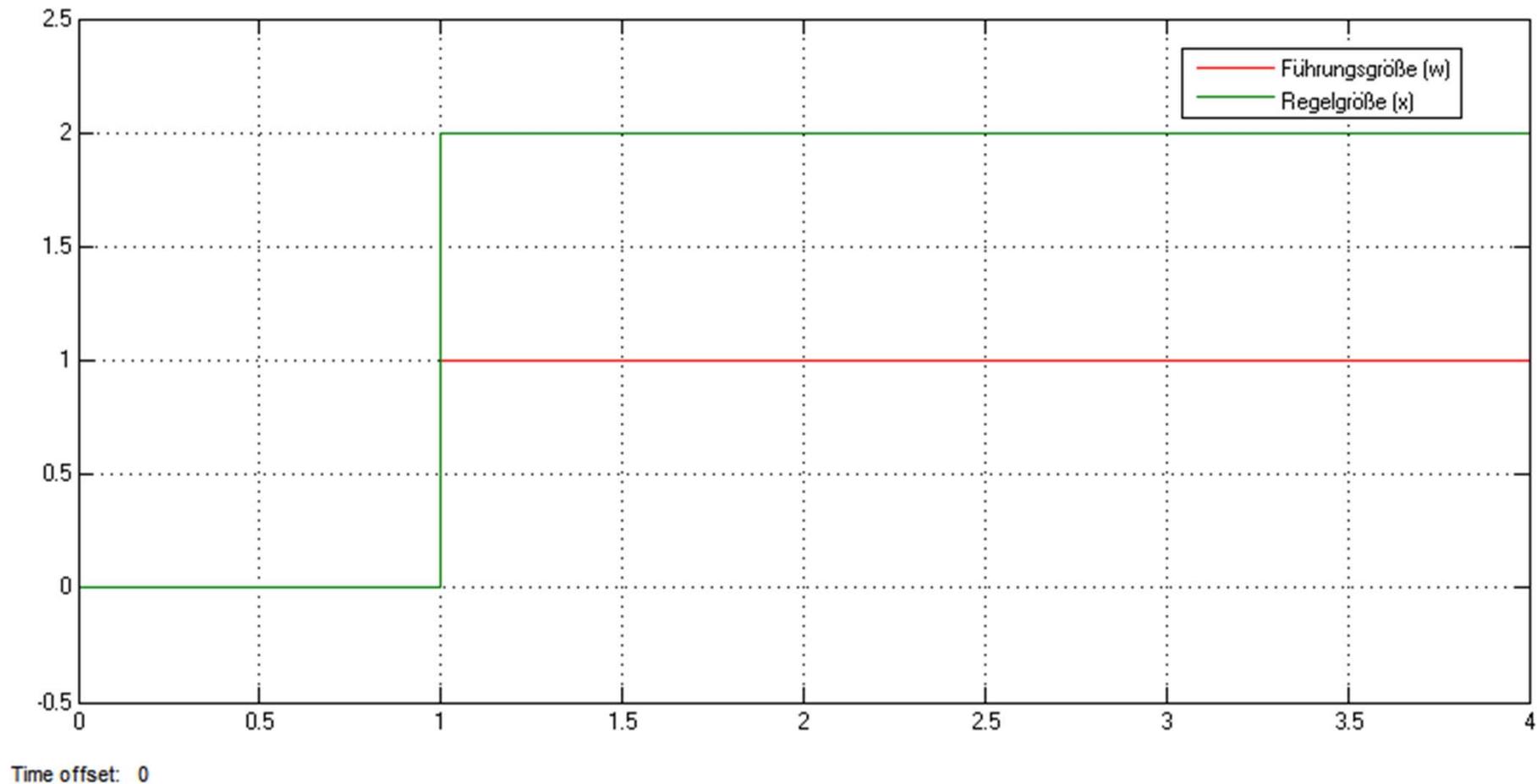
Dieser Ausdruck entspricht einer Geradengleichung, deren Steigung durch  $K_p$  bestimmt wird. Bild 24 verdeutlicht, dass ein großes  $K_p$  für einen starken Anstieg steht, so dass schon kleine Regelabweichungen zu großen Stellvorgängen führen.

Proportionalbeiwert  
oder -bereich

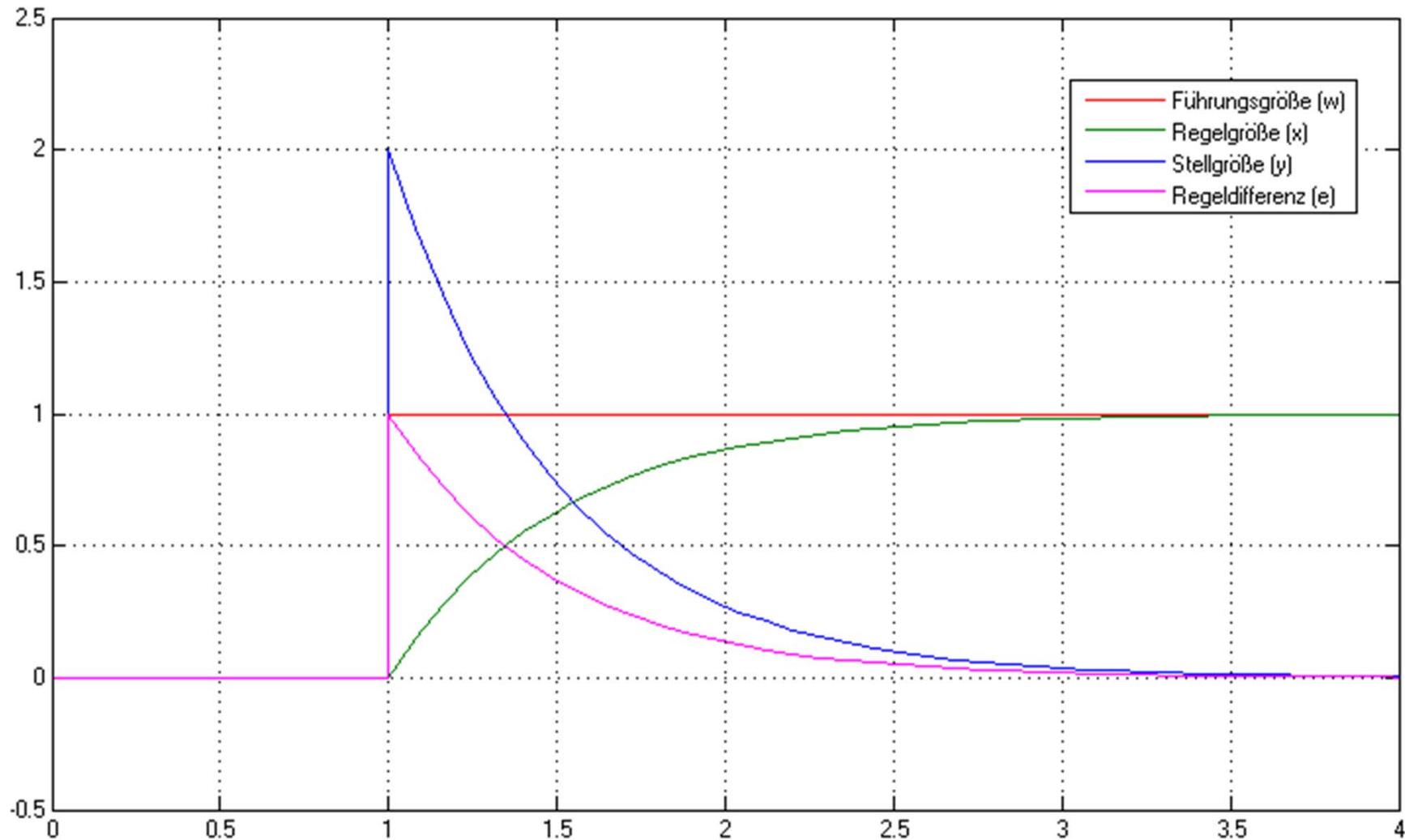
Anmerkung: Statt des Proportionalbeiwertes  $K_p$  findet man in der Literatur häufig noch den alten Begriff 'Proportionalbereich' bzw. 'P-Bereich', der mit den Formelzeichen  $X_p$  [%] abgekürzt wird. Die Umrechnung erfolgt mit:

$$X_p = \frac{100[\%]}{K_p} \quad \text{bzw.} \quad K_p = \frac{100[\%]}{X_p}$$

# Sprungantwort eines P-Reglers ( $K_p=2$ )



# Regelverhalten eines P-Reglers ( $K_p=2$ )



Time offset: 0

# P-Regler

- Vorteile
  - Schnelles Eingreifen in den Regelprozess
  - Sehr stabile Regelung, wenn  $K_p$  richtig dimensioniert ist
- Nachteile
  - Bleibende Regeldifferenz beim Auftreten einer Störung, da erst die Regelabweichung eine Veränderung der Stellgröße bewirkt
- P-Regler eignet sich für unkritische Regelungen, bei denen bleibende Regelabweichungen beim Auftreten von Störungen akzeptiert werden können

# I-Regler

- I-Regler werden eingesetzt, wenn sehr geringe Anforderungen bezüglich der Regeldynamik bestehen und um Regelabweichungen in jedem Betriebspunkt vollständig auszuregeln. Solange die Regelabweichung ungleich null ist, ändert sich der Betrag der Stellgröße. Erst wenn Führungs- und Regelgröße gleich groß sind ist die Regelung eingeschwungen.

Wie schnell die Stellgröße ansteigt (oder abfällt), hängt von der Regelabweichung und der Integrierzeit  $T_I$  ab (Kehrwert vom Integrierbeiwert  $K_I$ ). Bei einer kleinen Integrierzeit steigt das Stellsignal schneller als bei einer großen Integrierzeit (kleiner Integrierbeiwert).

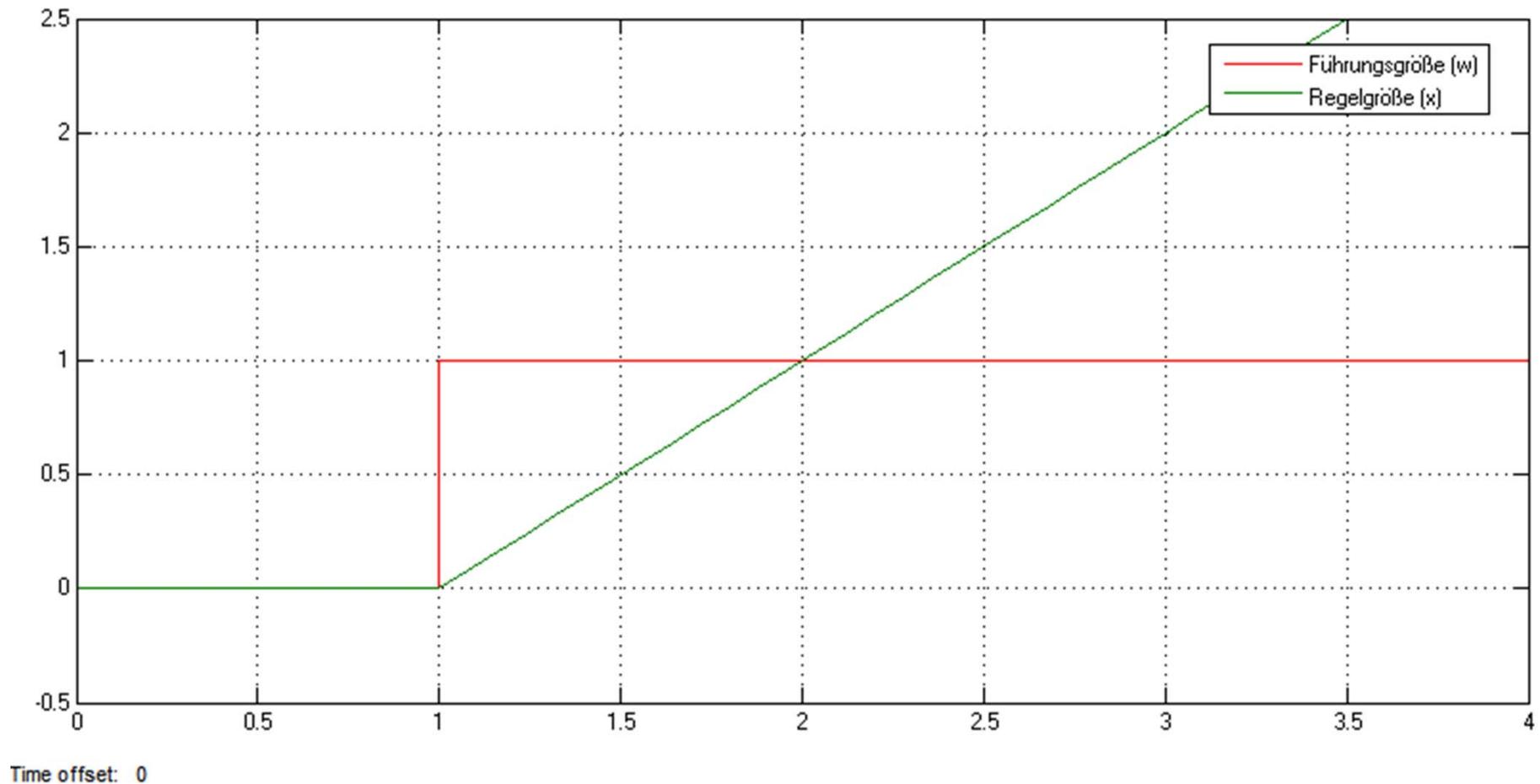
Merke: Die Integralwirkung des I-Reglers verringert sich bei steigender Integrierzeit  $T_I$  – bzw. steigt mit dem Integrierbeiwert  $K_I$ .

geschwungen. Die mathematische Formulierung dieses integralen Verhaltens lautet: Die Stellgröße ist dem Zeitintegral der Regeldifferenz  $e$  proportional:

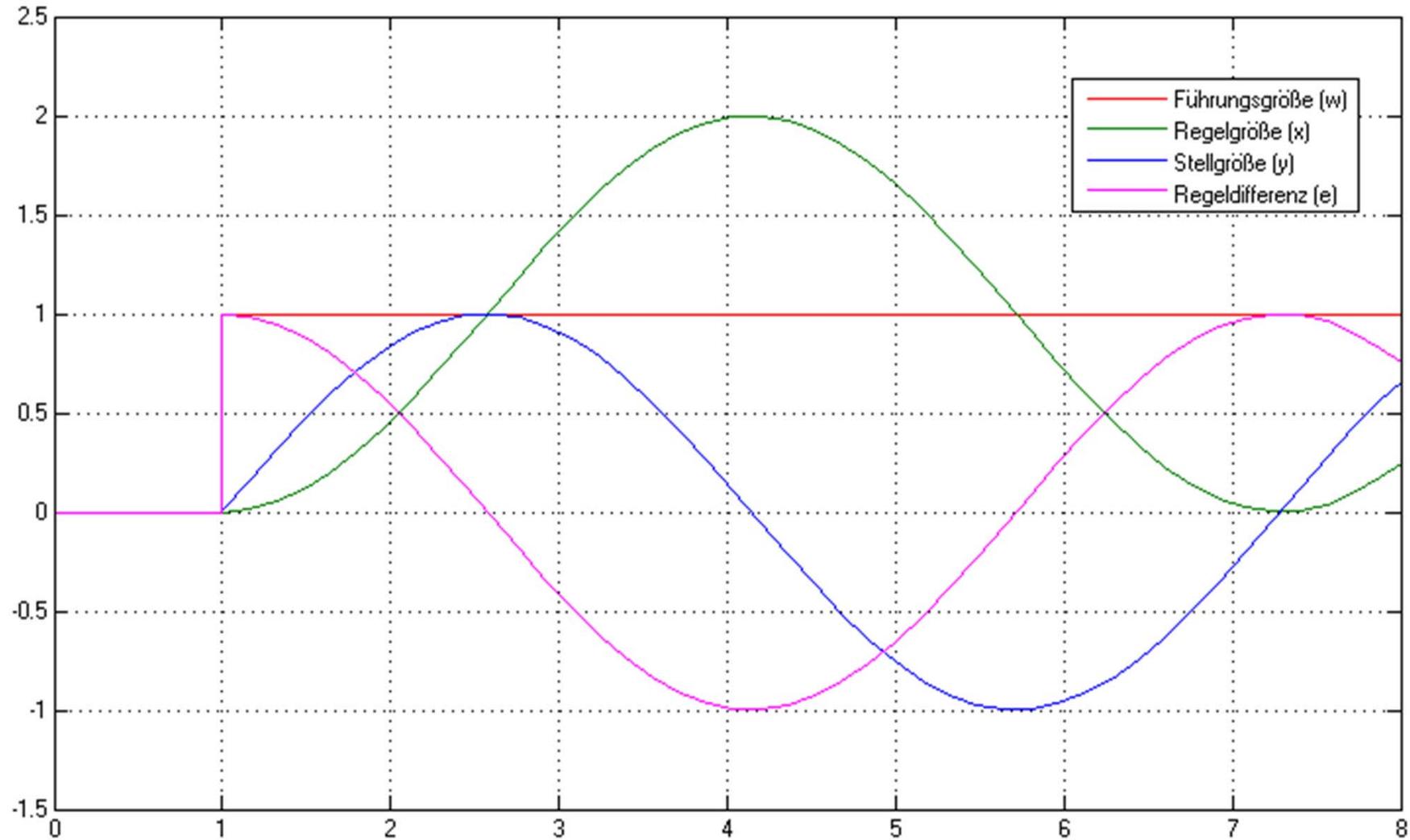
$$y = K_I \int e \, dt \quad \text{mit: } K_I = \frac{1}{T_I}$$

**großes  $T_I \Rightarrow$  langsamer  
Stelleingriff**

# Sprungantwort eines I-Reglers ( $K_I=1$ )



# Regelverhalten eines I-Reglers ( $K_I=1$ )

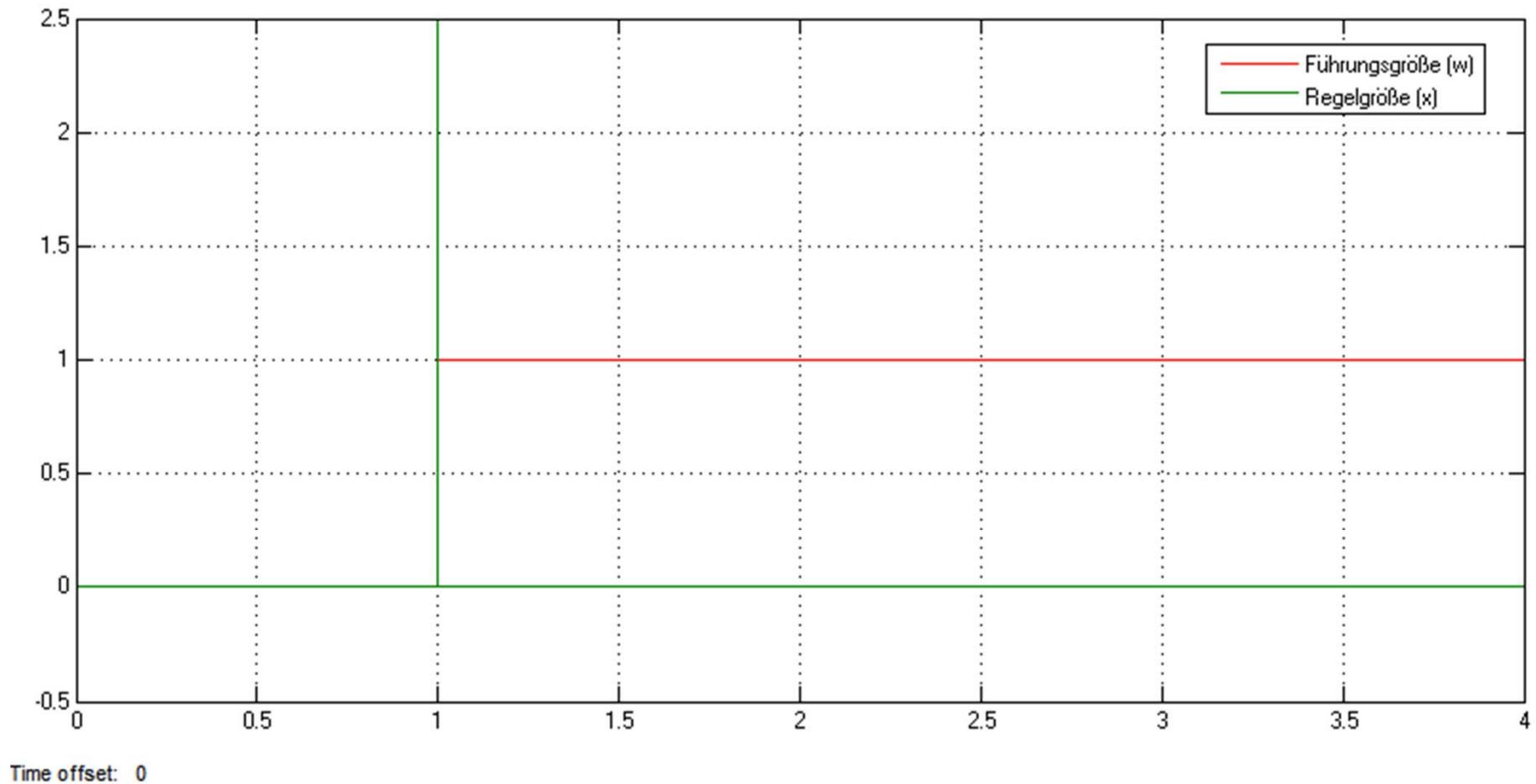


Time offset: 0

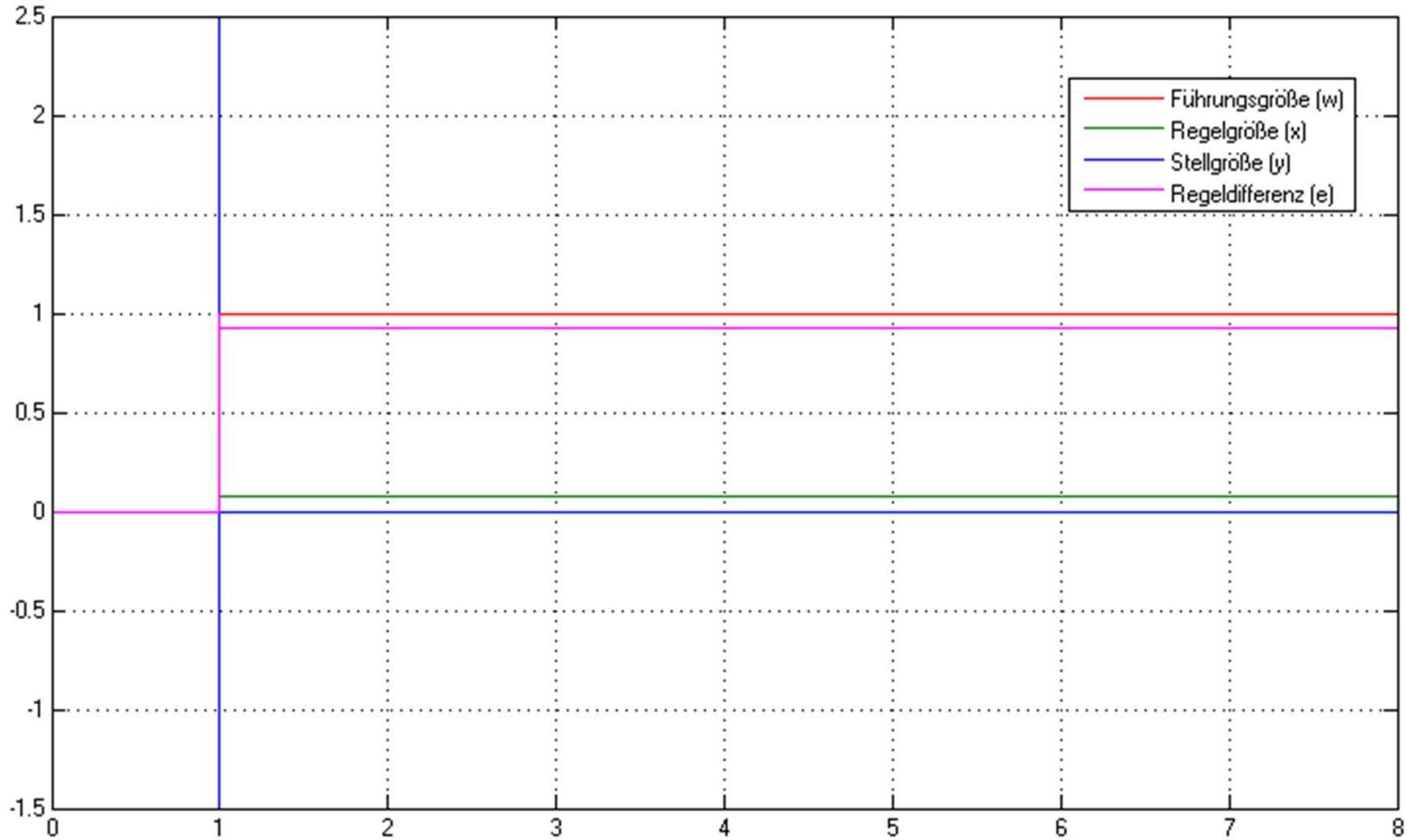
# D-Regler

- D-Regler bildet seine Stellgröße aus der Änderung der Regeldifferenz und nicht –wie der P-Regler- aus deren Amplitude. Er reagiert deshalb wesentlich schneller als der P-Regler:
  - Selbst bei kleiner Regeldifferenz erzeugt er - quasi vorausschauend - große Stellamplituden, sobald eine Amplitudenänderung auftritt.
- Eine bleibende Regelabweichung erkennt der D-Regler hingegen nicht, denn ganz unabhängig wie groß sie ist, ihre Änderungsgeschwindigkeit ist gleich null.
- In der Praxis wird dieser Regler deshalb selten alleine verwendet. Vielmehr kommt er in Verbindung mit anderen Regelementen, meist einem Proportionalanteil, zum Einsatz.

# Sprungantwort eines D-Reglers ( $K_D=1$ )



# Regelverhalten eines D-Reglers ( $K_D=1$ )

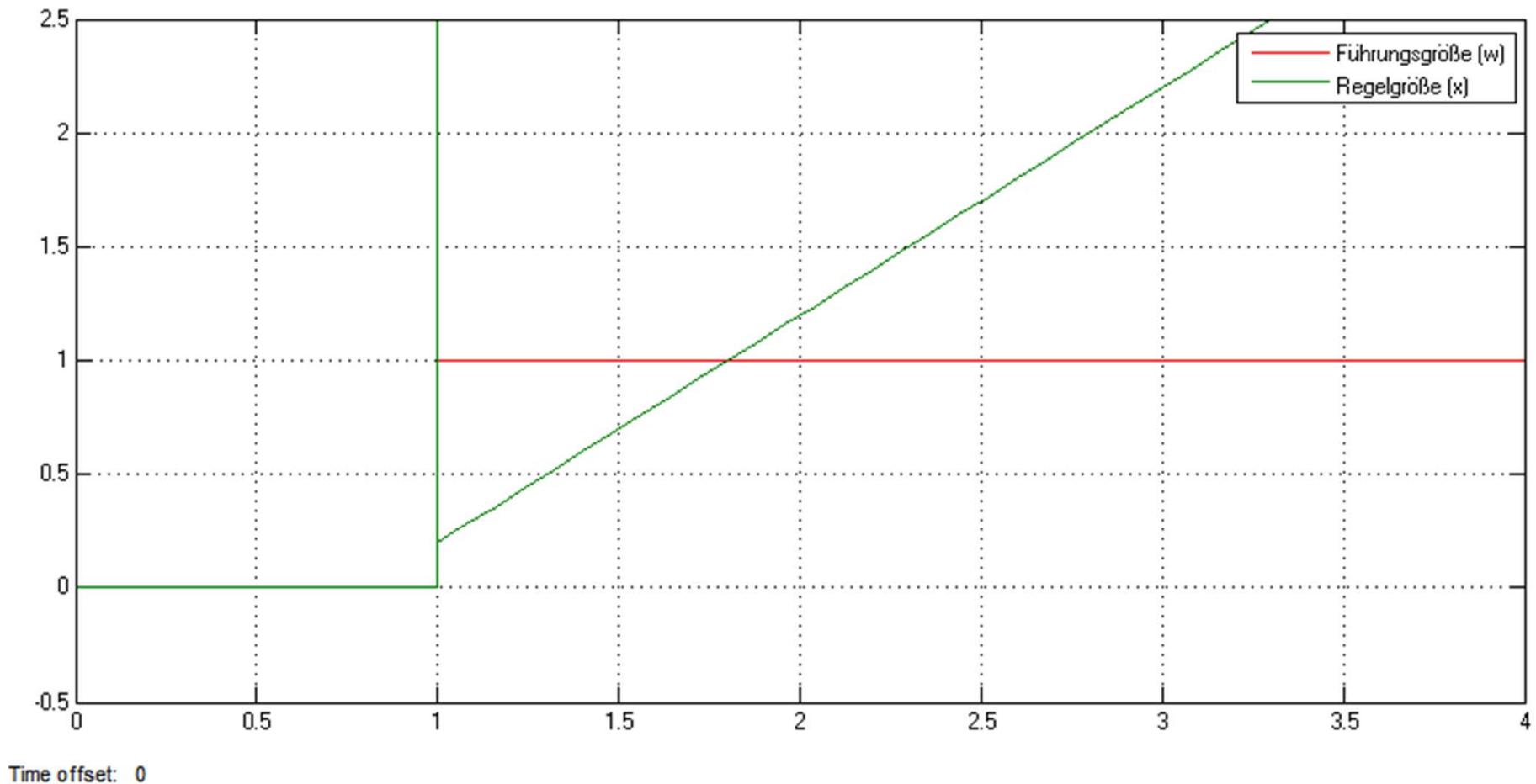


Time offset: 0

# PID-Regler

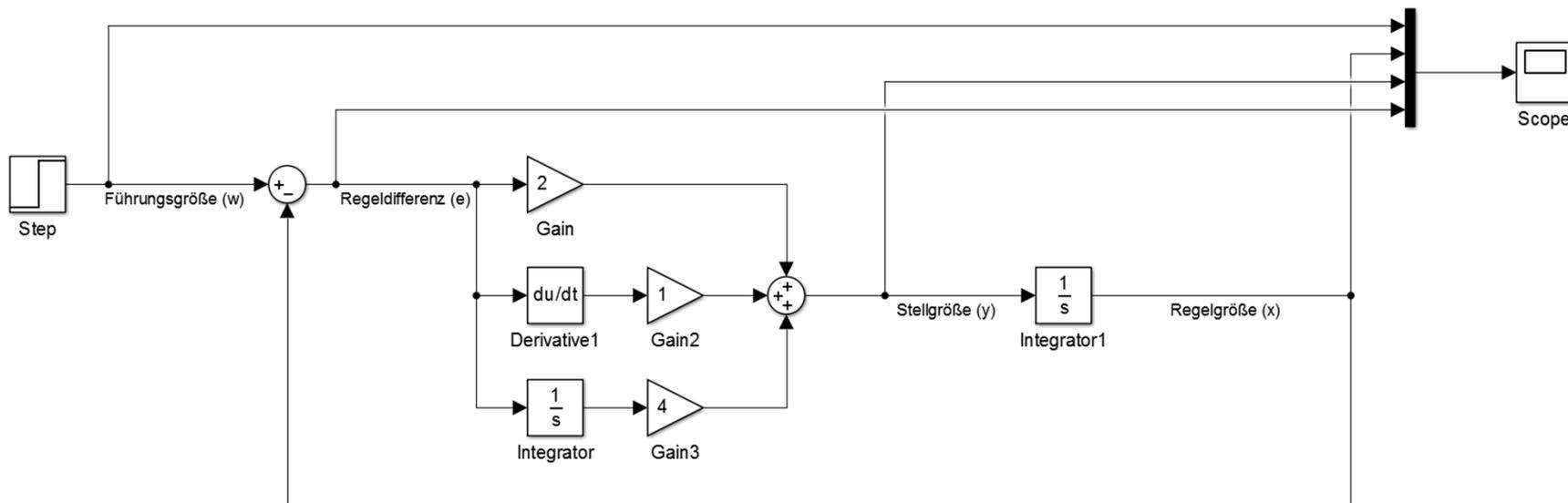
- Setzt man alle bisherigen stetigen Regleranteile in einer Parallelschaltung zusammen, erhält man den universellen PID-Regler.
- Durch die P- und I-Anteile vereinen sich die Vorteile der beiden Regler, sodass die jeweiligen Nachteile kompensiert werden (stabil, schnell und keine bleibende Regelabweichung)
- Fügt man noch den D-Anteil hinzu, bewirkt die Ergänzung, dass –bei richtiger Auslegung- die Regelgröße früher ihren Sollwert erreicht und schneller einschwingt

# Sprungantwort eines PID-Reglers ( $K_p=0,2; K_i=1; K_D=0,02$ )

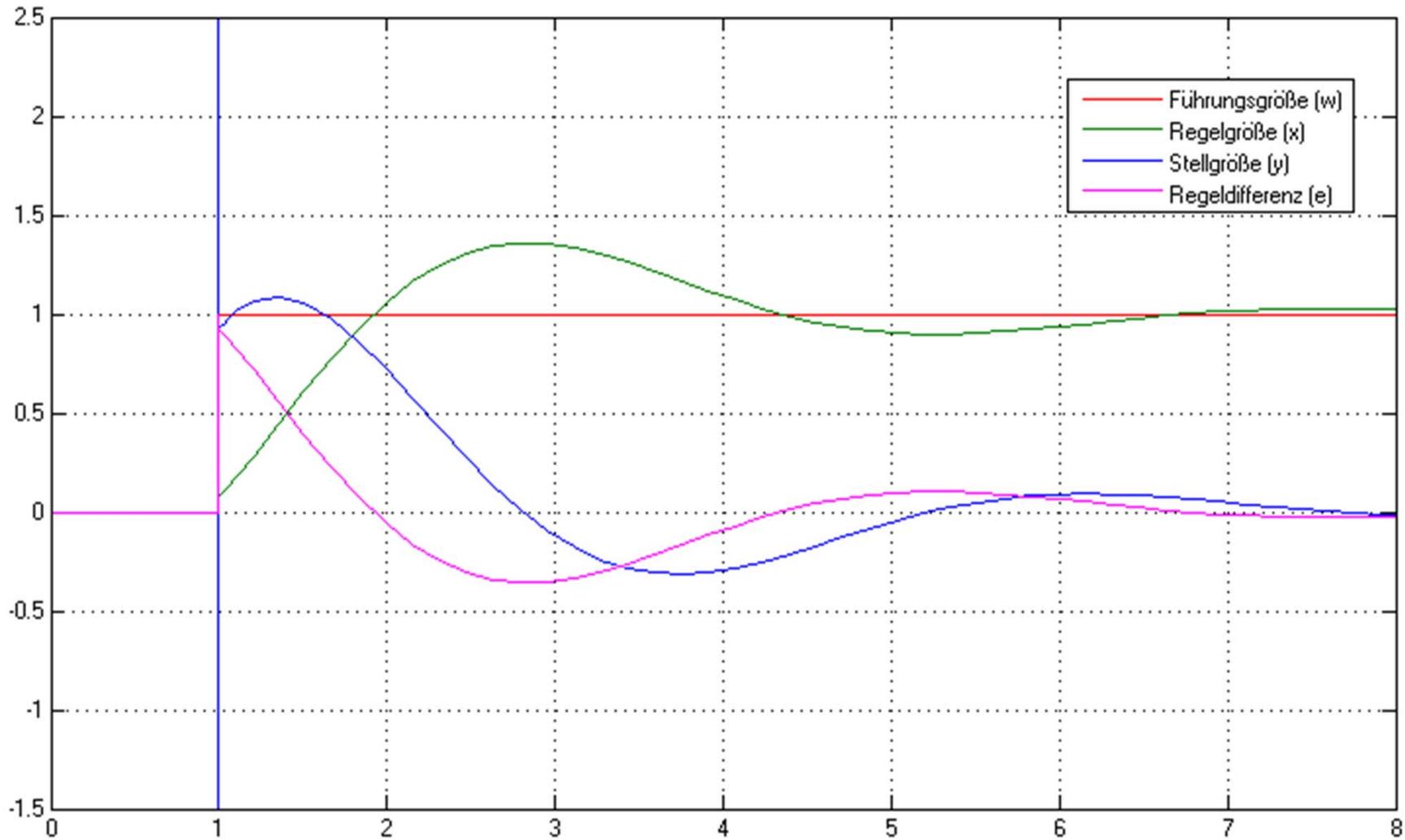


# Regelverhalten eines PID-Reglers ( $K_p=2;K_i=4;K_D=1$ )

- Da die P-,I- und D-Anteile des PID-Reglers parallel geschaltet werden, müssen ihre Reglerausgangsgrößen ( $u$ ) addiert werden
- In der Abbildung erkennt man gut die Beiwerte  $K_p$ ,  $K_i$  und  $K_D$



# Regelverhalten eines PID-Reglers ( $K_p=2;K_i=4;K_D=1$ )



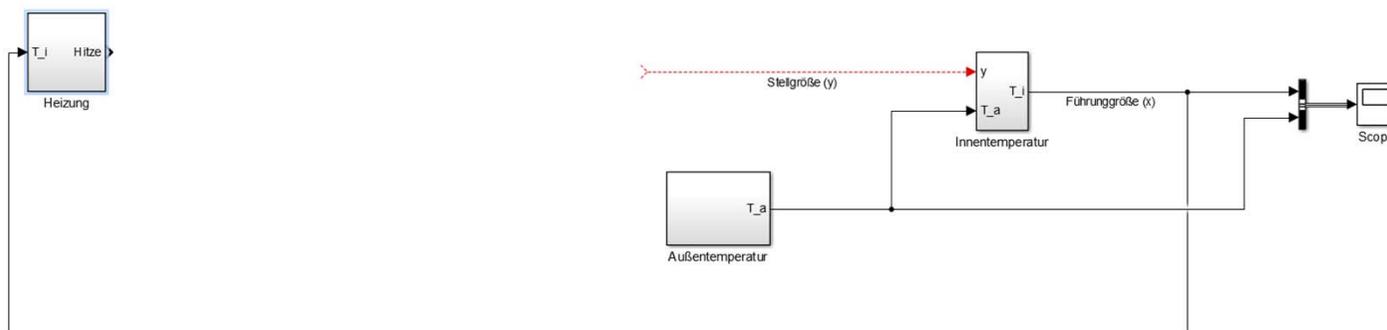
Time offset: 0

Regelung und Steuerung

# AUFGABE: REGELUNG EINER HAUSHEIZUNG

# Aufgabe: Regelung einer Hausheizung

- Vorgegeben in MATLAB: Außentemperatur, Heizungsanlage, Verhalten der Innentemperatur
- Die Soll-Innentemperatur soll maximal 3 °C unter den Sollwert fallen
- Der Sollwert liegt bei 21 °C



Regelung und Steuerung

# REGLERAUSLEGUNG

# Schritte bei der Bearbeitung einer Regelungsaufgabe

1. Abklärung, Festlegung der Aufgabenstellung (siehe auch 2.1.1, 2.1.2)
2. Überprüfen, welche Störgrößen auf die Regelstrecke wirken, und welche Auswirkungen sie haben, sind einige davon direkt oder indirekt messbar (für Störgrößenaufschaltung)?
3. Stellbereiche der Stellgrößen bestimmen
4. Auswahl der erforderlichen Stellglieder
5. Messorte und Messglieder festlegen
6. dynamisches Verhalten der Regelstrecke einschließlich Stell und Messglied ermitteln, d.h. meist: mathematisches Modell erstellen
7. den zugehörigen Regler ermitteln und seine Parameterwerte abschätzen
8. ggf. Verhalten des Regelkreises am Rechner simulieren und optimieren
9. Installieren und optimieren des Reglers an der Strecke

Anmerkung: die Schritte 4 - 8 sind ggf. iterativ zu wiederholen, da die Regelgüte auch ganz wesentlich vom Verhalten des Stellgliedes und des bzw. der Messglieder

# Regelkreisoptimierung (Auswahl und Parametrierung des Reglers)

- Randbedingungen:
  - Der Regelkreis soll stationär genau arbeiten oder eine vorher definierte stationäre Ungenauigkeit (bleibende Regelabweichung) nicht überschreiten
  - Da ein Regelkreis wegen seiner Kreisstruktur instabil werden kann, d.h. die Regelgröße nicht gegen einen stationären Endwert läuft, sondern Dauerschwingungen oder aufklingende Schwingungen erzeugt, ist das Hauptziel des Reglerentwurfs Stabilität des Regelkreises (s. Schwingungsverhalten)

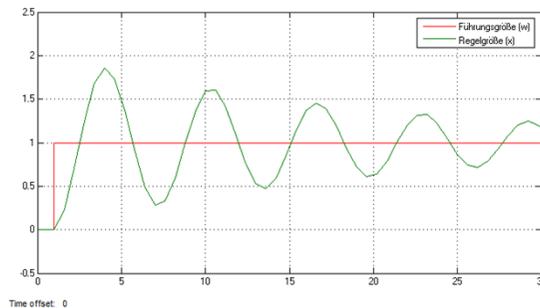
# Schwingungsverhalten

- Regelstrecke und Regler bilden zusammen den Regelkreis.
- Der Regler greift bei einer Änderung der Störgröße  $z$  oder der Führungsgröße  $w$  (Sollwert) in den Regelkreis ein.
- Dabei reagiert der Regler unterschiedlich, je nachdem ob eine Störung oder eine Führungsgrößenänderung stattfindet.

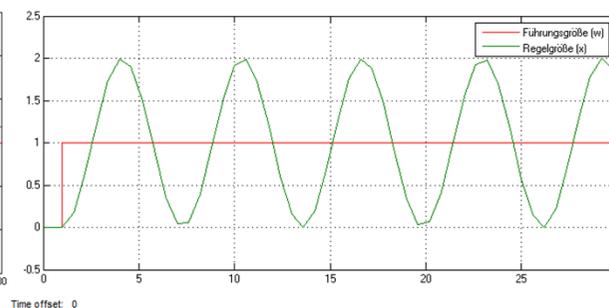
# Schwingungsverhalten

- Ist der Regler gut an eine Regelstrecke angepasst, kommt es zu rasch abklingenden Schwingungen der Regelgröße  $x$ .

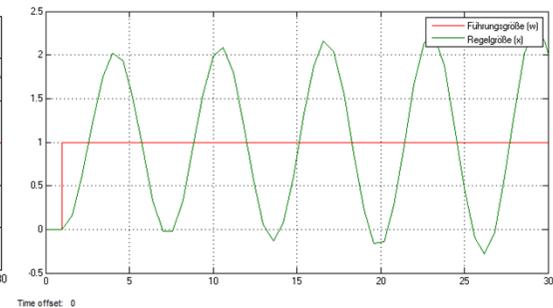
a) Stabil



b) Grenzstabil



c) Instabil



# Kennzeichen guter Regelungen

- Allgemein verbindliche Qualitätsmerkmale für eine gute Regelung sind schwer zu definieren. Die Güte einer Regelung ist von der speziellen Regelaufgabe abhängig.
  - Bei einer Lageregelung einer Drehmaschine darf es zum Beispiel keine Überschwinger geben, weil sonst zu viel Material abgedreht würde. Dadurch wird aber zwangsläufig die An- und Ausregelzeit größer.

# Kennzeichen guter Regelungen

- Es können aber folgende Anforderungen definiert werden:
  - **Überschwingweite  $x_m$  klein**
  - **Anregelzeit  $t_{an}$  kurz**
  - **Ausregelzeit  $t_{aus}$  kurz**

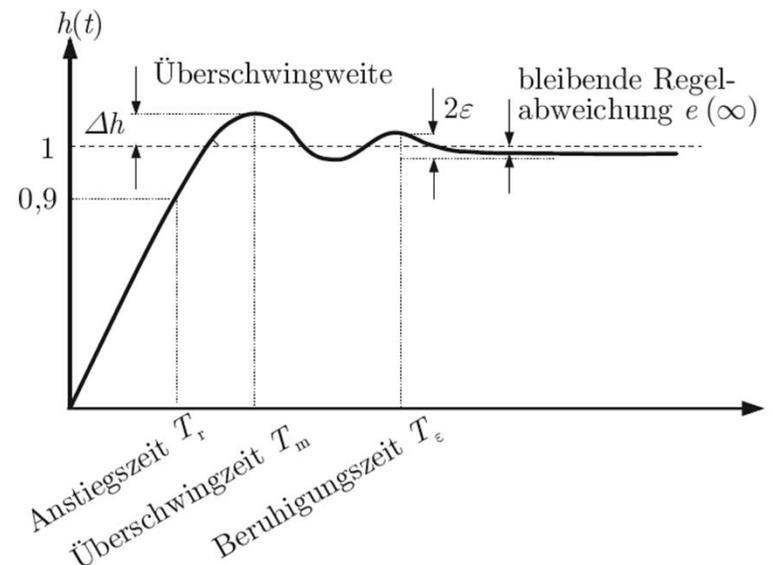


Abb. 7.1: Führungsübergangsfunktion des Regelkreises mit Kennzeichnung wichtiger Kennwerte

Quelle: Lunze, J.: „Regelungstechnik 1“ (2008)

Regelung und Steuerung

# ANALYSE DER REGELSTRECKE

# Analyse der Regelstrecke

- Beim Regeln ist es wichtig, das Verhalten der Regelstrecke, Man unterscheidet
  - Statisches Verhalten: Auf welchen neuen Wert stellt sich die Regelgröße  $x$  nach Änderung der Stellgröße  $y$  ein?
  - Dynamisches Verhalten: Wie ist der zeitliche Verlauf der Regelgröße  $x$  nach Änderung der Stellgröße  $y$ ?

# Anpassung des Reglers an die Strecke

- In der Regelungstechnik wird eine zu regelnde Strecke in erster Linie durch ihr Zeitverhalten charakterisiert. Dieses
- Damit ein Regelkreis seine Aufgabe möglichst gut erfüllt, muss die Reglerstruktur mit den zugehörigen Parametern so an die Strecke angepasst werden, dass das geschlossene System ein „optimales“ Verhalten annimmt.
- Der Regelkreis kann sprungförmigen Änderungen der Führungsgröße nur mehr oder minder verzögert folgen, Störungen werden deshalb erst nach einer gewissen Zeit ausgeregelt.

# Hilfestellungen bei der Reglerauswahl

- Ist abhängig von:
  - Ob eine integral- oder proportionalwirkende Strecke vorliegt
  - Wie groß die Verzögerungen (Zeitkonstante und /oder Totzeiten) der Strecke sind
  - Wie schnell eine Regelabweichung ausgeregelt werden muss
  - Ob bleibende Regelabweichungen akzeptiert werden können
- Die wichtigsten Merkmale der zumeist verwendeten Regelglieder in der folgenden Tabelle zusammengefasst.

Regelglied	Regel-abweichung	Stell-geschwindigkeit
P	bleibend	hoch
PD	bleibend	sehr hoch
I	keine	niedrig
PI	keine	hoch
PID	keine	sehr hoch

Regelung und Steuerung

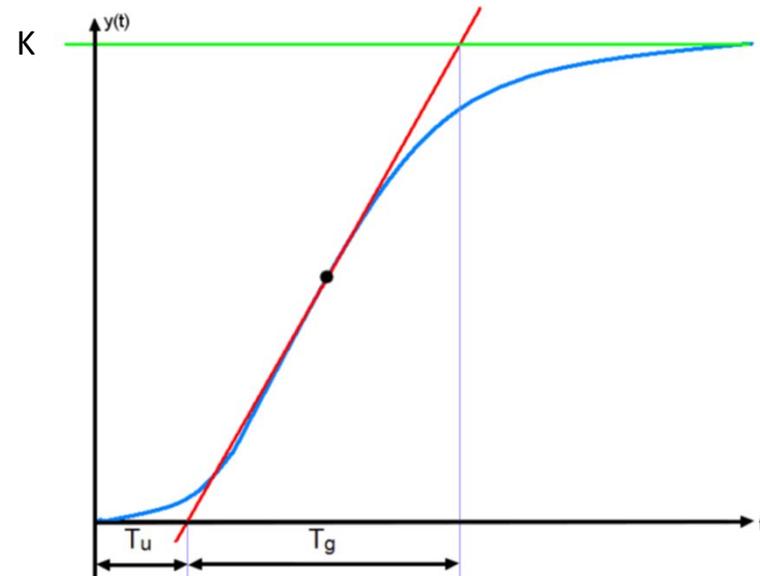
# REGLEREINSTELLUNG

# Reglereinstellung

- Ein Regelkreis arbeitet stabil, wenn alle Glieder (Regelstrecke, Messeinrichtung, Stellglied und Regler) aufeinander abgestimmt sind.
- Die Eigenschaften der Regelstrecke, der Messeinrichtung und des Stellgliedes liegen fest und sind meist bekannt.
- Einstellungen können also nur am Regler erfolgen, eine brauchbare Reglereinstellung erweist sich aber als schwierig und langwierig.
- In der Praxis gibt es durch Versuche ermittelte Einstellkriterien, die zu guten Regelergebnissen führen.

# 1. Verfahren nach Ziegler/Nichols

- Besitzt eine Strecke eine stabile annähernd S-förmige Sprungantwort, kann die Wendetangenten-Methode nach Ziegler und Nichols angewendet werden um die Regelparameter abzuschätzen.
- Der erste Wert kann abgelesen werden, die:
  - Verstärkung  $K$
- Die Tangente am Wendepunkt der Sprungantwort liefert 2 weitere Parameter zur Dimensionierung, die:
  - Verzugszeit  $T_u$
  - Ausgleichszeit  $T_g$



# 1. Verfahren nach Ziegler/Nichols

- Die Reglerdimensionierung erfolgt auf, der Grundlage von Erfahrungswerten.
- Diese Werte sind in der nachstehenden Tabelle zusammengefasst:

Regler	$K_p$	$K_I(\triangleq T_n)$	$K_D(\triangleq T_v)$
P	$\frac{T_g}{K * T_u}$	---	---
PI	$\frac{0,9 * T_g}{K * T_u}$	$3,33 * T_u$	---
PID	$\frac{1,2 * T_g}{K * T_u}$	$2 * T_u$	$0,5 * T_u$

# 1. Verfahren nach Ziegler/Nichols

- Vorteile:
  - Sehr schneller Entwurfsvorgang
  - Ausreichend robustes Regelverhalten (Stabilität)
- Nachteile:
  - Keine hohe Regelgüte erreichbar
  - Nicht anwendbar bei Verzugszeit  $>$  Ausgleichszeit
  - Sehr sensitiv für Verzugszeit  $\approx$  Ausgleichszeit
  - Nur für stabile Strecken mit oben dargestellter Sprungantwort

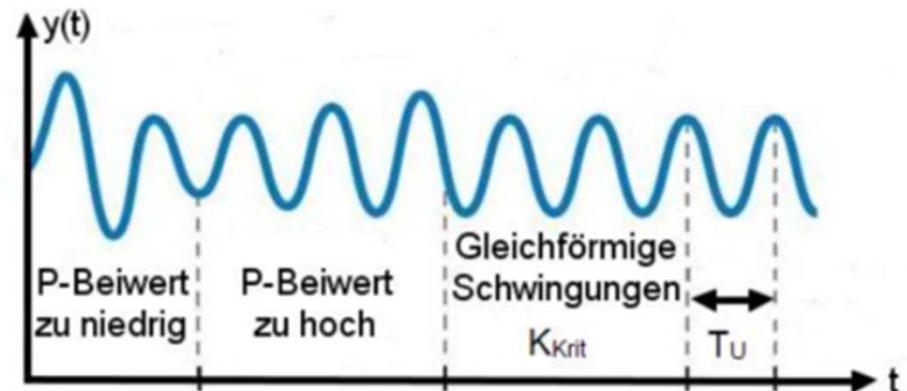
## 2. Verfahren nach Ziegler/Nichols

- Für diese sogenannte Einstellung am Stabilitätsrand sind keine Annahmen bzgl. des Übertragungsverhalten der Strecke nötig.
- Zunächst wird der Regler als reiner P-Regler betrieben. Dazu wird der I-Beiwert des Reglers „sehr groß“ und der D-Beiwert „sehr klein“ eingestellt.
  - Die entspricht einem Deaktivieren der Anteile!
- Dann variiert man den P-Beiwert  $K_p$  des Reglers so lange, bis die Regelgröße ungedämpft schwingt.

Diese Einstellung von  $K_p$   
nennt man

**kritischen Proportionalbeiwert  $K_{krit}$**

Außerdem kann nun die  
**kritische Periodendauer  $T_{krit}$**   
ermitteln



Quelle: „Reglereinstellung nach Ziegler und Nichols“, TU Dresden(2011)

## 2. Verfahren nach Ziegler/Nichols

- Die Reglerdimensionierung erfolgt mittels der beiden vorher bestimmten Werte:
  - kritischen Proportionalbeiwert  $K_{Krit}$
  - kritische Periodendauer  $T_{Krit}$

Regler	$K_p$	$K_I(\triangleq T_n)$	$K_D(\triangleq T_v)$
P	$0,5 * K_{Krit}$	---	---
PI	$0,45 * K_{Krit}$	$0,85 * T_{Krit}$	---
PID	$0,6 * K_{Krit}$	$0,5 * T_{Krit}$	$0,12 * T_{Krit}$

## 2. Verfahren nach Ziegler/Nichols

- Vorteile:
  - Keine Kenntnisse über die Regelstrecke notwendig
- Nachteile:
  - Ein Erzielen einer Schwingung an der Stabilitätsgrenze wie oben beschrieben kann jedoch nur dort durchgeführt werden, wo ein Ausscheren des realen Systems in den instabilen Bereich keine schädlichen Folgen hat.
  - Ein instabiler Tempomat am Auto würde abwechselnd Vollgas geben und kein Gas geben, was vielleicht noch in geeigneter Umgebung durchführbar wäre, bei einem Autopiloten eines Passagierflugzeugs wären die Folgen sicherlich nicht tragbar.

Sensorapplikationen

# MESSEN VON STOFF- UND ENERGIESTRÖMEN

# Ohmsches Gesetz

$$U = R \cdot I$$

Elektrische Leistung:

$$P = U \cdot I$$

$$P = I^2 \cdot R$$

$$P = \frac{U^2}{R}$$

$U$  : Spannung [V]

$R$  : Widerstand [ $\Omega$ ]

$I$  : Stromstärke [A]

$P$  : Leistung [W]

# Shunt

Messsignal: 4 - 20 mA

Karte: 0 - 10 V

$$R = \frac{U}{I} = \frac{10 \text{ V}}{0.02 \text{ A}} = 500 \ \Omega$$

Shunt = Messwiderstand:

Problem bei Shunt:

Temperaturabhängigkeit -> R(T)  
kein Standardwiderstand!

# Erdung (I)

## Was bedeutet „Erden“?

Als Erden bezeichnet man im allgemeinen Sprachgebrauch jeden Anschluss an ein Bezugspotenzial, das über sogenannte „Erder“ mit dem leitfähigen Erdreich verbunden ist. Erder sind leitfähige Teile, die in einem guten elektrischen Kontakt mit dem Erdreich stehen und deren Anschlusspunkte meist aus dem Erdreich herausgeführt sind.

Erder könne, je nach Funktion oder örtlichen Gegebenheiten, verschieden ausgeführt sein z.B. als Staberder, Banderder oder Plattenerder.

# Erdung (II)

## Warum „Erden“?

**Das Verbinden mit Erdpotenzial verfolgt unterschiedliche Ziele:**

**Das Erden als Schutzmaßnahme im Sinne der geltenden VDE-Vorschriften (DIN VDE 0100) und der Niederspannungsrichtlinien, wobei diverse nationale Unterschiede zu beachten sind.**

**Das Verhindern von Störeinstrahlungen und Störausstrahlungen im Sinne der EMV-Richtlinien.**

**Das für den Betrieb elektrischer Geräte und Einrichtungen erforderliche Festlegen eines gemeinsamen Bezugspotenzials.**

# Erdung (III)

## Bezeichnungen für Bezugs- und Erdungsleiter

Für Bezugs und Erdungspunkte sind die folgenden Bezeichnungen üblich:



- ◆ **0 V/Masse** - allgemein für das Bezugspotential elektrischer Schaltungen



- ◆ **0 VA** - für Bezugsleiter innerhalb analoger Schaltkreise eines Geräts



- ◆ **0 VD** - für Bezugsleiter innerhalb digitaler Schaltkreise eines Geräts



- ◆ **TE** - für den Sternpunkt (meist Sammelschiene), an dem 0 VA und 0 VD zusammengeführt sind. TE wird oft auch als "Elektronikerde" oder "Messerde" bezeichnet.



- ◆ **SE** - Sternpunkt (meist Sammelschiene), an dem alle Kabelschirme z.B. in einem Gehäuse aufgelegt werden. SE ist oder wird meist mit PE verbunden.



- ◆ **PE** - für den Anschluss des grün-gelben Schutzleiters der Netzversorgung oder allgemeiner Schutzerdungspunkt.

TE und SE können mit PE verbunden werden, wenn PE störungsarm ist. Ist das nicht der Fall, muss zumindest TE an eine getrennte, saubere (störungs-armen) Erde angeschlossen werden.



- ◆ **FE** - Betriebsstätten mit konsequentem Erdungskonzept stellen solche sogenannten Funktionserden **FE** zur Verfügung (z.B. in Schaltwarten).

# Erdung (IV)

## Elektromagnetische Verträglichkeit (EMV)

EMV relevante Störsignale werden unterschieden in:

Leitungsgebundene, wie sie z.B. von Leuchtstoffröhren, EDV-Anlagen, Schützen, Schaltheandlungen und vor allem Frequenzumrichter verursacht werden und in strahlungsgebundene, die z.B. von HF-Sendern (Funk, Fernsehen, Mobiltelefon usw.) erzeugt werden.

Die eindeutige Zuordnung der Störquellen ist dabei nicht immer möglich, da viele in beide Kategorien eingeordnet werden können, wie z.B. Schütze.

# Erdung (V)

**Abschirmen gegen Störungen (EMV):**

**Kabelabschirmungen möglichst aus gut leitendem Material (z.B. Aluminiumfolie)**

**Größtmöglicher Schutz durch doppelt abgeschirmte Kabel, bestehend aus: Gesamtschirm (1) und Einzelschirmen (2)**

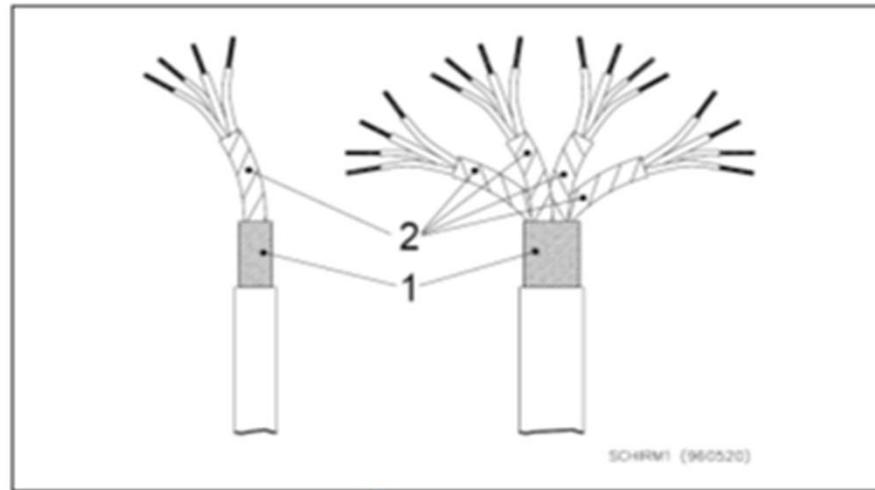


Abb.: Doppelt abgeschirmte Kabel

# Erdung (VI)

Nach Möglichkeit sollten Kabelschirme beidseitig aufgelegt werden.

Zu beachten sind dabei eventuelle Potenzialunterschiede, die Ausgleichsströme verursachen.

Ausgleichsströme können Schirm, Kabel und Elektronik zerstören.

**Folgendes ist zu beachten:**

- Kabellänge **unter 25m**: Keine Maßnahmen notwendig, da keine erheblichen Potenzialunterschiede zu erwarten sind.
- Kabellänge **über 25m**: Potenzialunterschiede prüfen und wenn notwendig, Maßnahmen ergreifen (z.B. stromtragfähige Ausgleichsleitung verlegen).

# Erdung (VII)

## Praktische Hinweise zum Auflegen von Schirmen

Schirme sollten nach dem Einführen in ein Gehäuse auf dem kürzesten Weg aufgelegt werden.

Die vom Schirm nicht überdeckten Einzeladern an den Kabelenden sollten so kurz wie irgend möglich gehalten werden.

Für doppelt geschirmte Kabel bedeutet das, dass der außenliegende Schirm direkt beim Eintritt in das Gehäuse und der oder die innenliegenden Schirme möglichst bis zu der Auflagestelle der Kabeladern mitgeführt wird.

Sie sollten möglichst großflächig und impedanzarm auf geerdeten Sammelschienen oder Montageplatten aufgelegt werden.

# Erdung (VIII)

**Richtige Kabelwahl und –verlegung:**

**Signal- und Datenkabel müssen immer getrennt von Energie- und Steuerleitungen oder in ausreichend Abstand davon verlegt werden. Unvermeidbare Kreuzungen zwischen diesen müssen im rechten Winkel verlaufen.**

**Alle nicht benutzte Adern eines Kabels sind einseitig zu erden.**

**Das Verlegen aller Kabel sollte auf dem kürzesten Weg unter Vermeidung von Schleifenbildung erfolgen**

**Leiter gleichen Potentials sollten möglichst sternförmig, also an einem gemeinsamen Punkt miteinander verbunden werden.**

# Erdung (IX)

## Verbinden von 0V-Bezugspotenzialen

Im Gegensatz zum vorherigen Abschnitt geht es hier nicht um die EMV-Festigkeit, sondern um das Vermeiden von Schäden an Kabeln und Schaltkreisen.

Die 0V-Bezugspotenziale elektrischer Geräte, die Mess-, Steuer- oder Datensignale austauschen, müssen funktionsbedingt miteinander verbunden werden. Dabei können örtliche Potenzialunterschiede unerwünschte Ausgleichsströme verursachen, die zu Signalverfälschung oder im Extremfall zur Zerstörung von Kabeln und Schaltkreisen führen können.

# Erdung (X)

## Verbinden von 0V-Bezugspotenzialen – potenzialfreie Verbindungen

Die Signalein- bzw. ausgänge der zu verbindenden Geräte sind potenzialfrei, also galvanisch von festen Potenzialen getrennt (z.B. durch Optokoppler, Übertrager):

- Hier sind keine besonderen Maßnahmen erforderlich. Dies gilt auch dann, wenn eines der beteiligten Geräte nicht potenzialfrei ist.

# Erdung (XI)

## Verbinden von 0V-Bezugspotenzialen – nicht potenzialfreie Verbindungen

Die Signalein- bzw. ausgänge sind nicht potenzialfrei, d.h. die Bezugspotenziale der Geräte sind mit dem Erdpotenzial ihres jeweiligen Montageortes oder der örtlichen Spannungsversorgung verbunden. Hier gilt:

bei Kabellängen unter 25m: Keine besonderen Maßnahmen zu ergreifen, da keine schädigenden Potenzialunterschiede zu erwarten sind. Sicherheitshaber sollten trotzdem die Erd- und Potenzialverhältnisse überprüft werden.

# Erdung (XII)

**Bei Kabellängen über 25m sind die örtlichen Erd- und Potenzialverhältnisse zu prüfen. Wenn verfälschende bzw. schädigende Ausgleichsströme verursacht werden können, sind folgende Maßnahmen abzuwägen:**

- Zwischen den unterschiedlichen Potenzialen muss eine Potenzialausgleichsschiene (PAS) oder –leitung von ausreichender Stromtragfähigkeit verlegt werden.
- Die Schaltkreise müssen mittels Trennverstärker o.ä. entkoppelt werden.

**Manche Geräte haben Bezugspotenziale, die angehoben, also nicht auf das lokale Erdpotenzial bezogen sind. Sie dürfen nicht mit Erdpotenzial verbunden werden. In diesem Fall muss mittels Trennverstärker entkoppelt werden.**

# Grundbegriffe (XII) – Fachbegriffe der Mess- und Prüftechnik

- **Eichen:** Von staatlicher Eichbehörde vorgenommene Prüfung u. Bestätigung
- **Einmessen:** Bestimmen der Kenngrößen des Messsystem
- **Graduieren:** Teilen einer Skala (Aufbringen von Teilungsmarken auf dem Skalenträger)
- **Justieren:** Einstellen eines Messsystems um Messabweichungen auf Werte zu bringen, die den Forderungen entsprechen
- **Kalibrieren:** Feststellen des für ein Messsystem gültigen Zusammenhangs zwischen Eingangs- und Ausgangsgröße
- **Lehren:** Maßlicher Vergleich auf vorgegebene Grenzwerte mit Hilfe nicht anzeigendem Prüfmittel

# Messfehler (I)

- Alle gemessenen Größen sind mit Fehlern behaftet. Als Fehler ist dabei die Differenz zwischen einem Messwert und dem zugehörigen wahren Wert zu verstehen.
- Für den in der Regel nicht bestimmbar wahren Wert wird von der Annahme ausgegangen, er sei durch den Vergleich mit einem Referenzwert gesichert oder er sei als mathematischer Erwartungswert existent.
- Die mathematische Behandlung von Messfehlern beruht auf der Wahrscheinlichkeitstheorie von C.F. Gauß.
- Dabei wird eine Wiederholbarkeit der Messungen unter bestimmten Bedingungen vorausgesetzt, was jedoch nicht immer möglich ist (z.B. Prozessmessung, die zur Prozessüberwachung und -optimierung eingesetzt wird).

# Messfehler (III) - Fehlerdefinition

Ein ideales Messsystem folgt der Übertragungsfunktion  $Y = X$ .  
Jede Abweichung davon ist als Fehler zu betrachten:

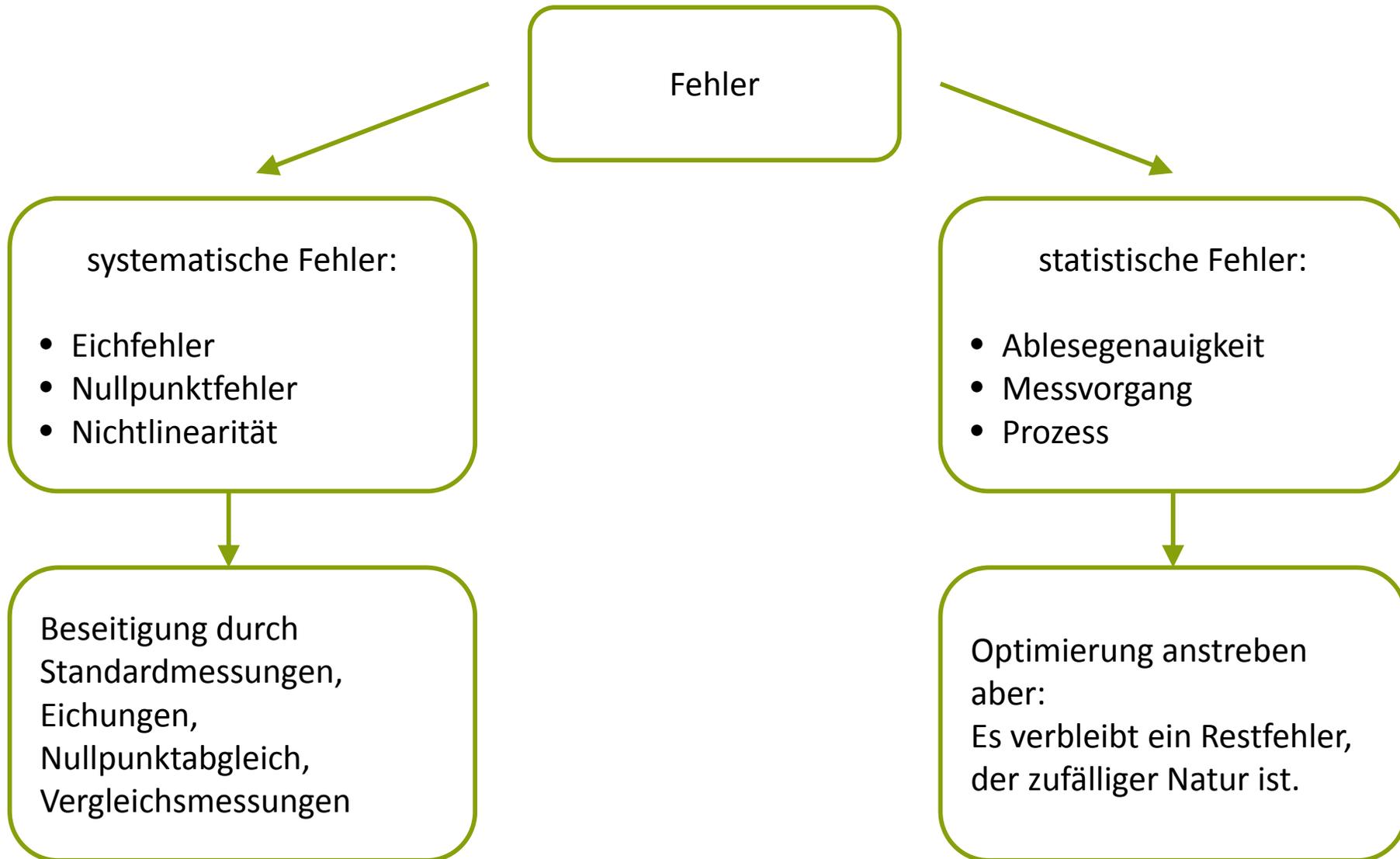
$$e = Y - X$$

Ein kleineres Messergebnis als die Messgröße führt zu einem Fehler mit negativem Vorzeichen, ein größeres Messergebnis als die Messgröße führt zu einem Fehler mit positiven Vorzeichen.

absoluter Fehler  $e$  wird mit der Einheit der Messgröße angegeben.

relativer Fehler  $e/X$  wird auf den wahren Wert bezogen und oft in % angegeben.

# Messfehler (IV) - Fehlerarten



# Messfehler (VI) – Ursachen für Messabweichungen

- **Unvollständige und falsche Formulierung der Messaufgabe**
- **Messobjekt**
- **Rückwirkung von Messobjekt auf Messmittel**
- **Messverfahren**
- **Messmittel**
- **Bediener**
- **Umwelt**

# Messfehler (VII) – Dynamisches Verhalten

- Jedes Messsystem besitzt eine bestimmte *Trägheit*, so dass das Ausgangssignal dem Eingangssignal nicht in einer beliebig kurzen Zeit folgen kann.
- Das *dynamische Verhalten* eines Messsystems beschreibt die Abhängigkeit der Ausgangsgröße von der Eingangsgröße bis zum Erreichen des stationären Zustands.

# Messfehler (VIII) – Dynamisches Verhalten

Folgende Fragen sind dabei in Bezug auf das dynamische Verhalten zu beantworten:

*Wie lange* dauert es nach Anlegen oder sprungförmiger Änderung der Messgröße, bis ein Messergebnis mit vertretbaren Fehlern abgelesen werden kann?

*Wie schnell* darf sich eine Größe selbst verändern, um bei vertretbarem Fehler den Verlauf verfolgen zu können?

# Messfehler (IX) – Zeitverhalten

**Totzeit (Verzögerungszeit):** Die Zeit, die zwischen dem Start des Input-Signals und dem beobachteten Start des Output-Signals vergeht.

**Einstellzeit (Anstiegszeit):** Die Zeit zwischen Auftreten des ersten signifikanten Messsignals und dem Erreichen von 95% des zu erwartenden Messsignals.

**Ansprechzeit:** Summe aus Tot- und Einstellzeit

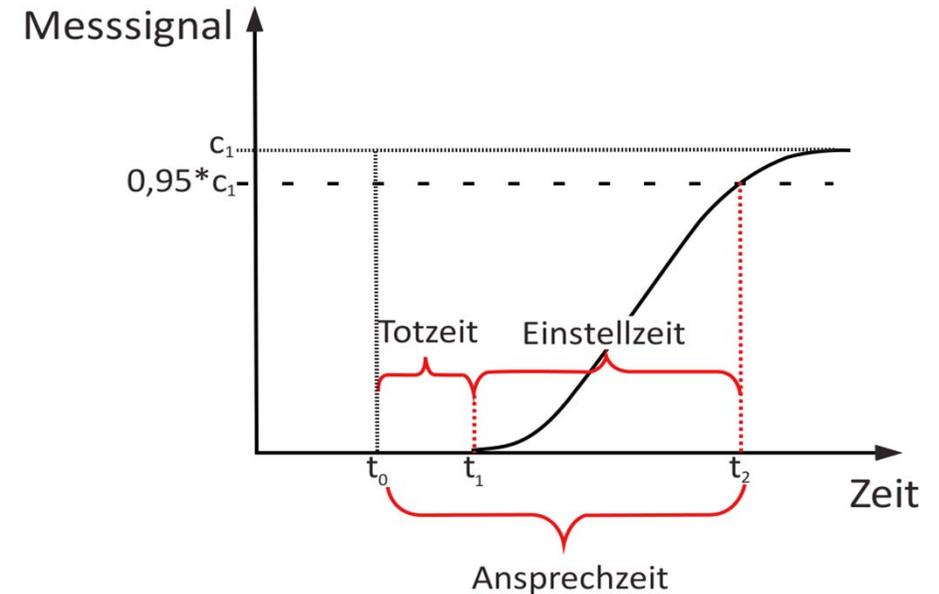
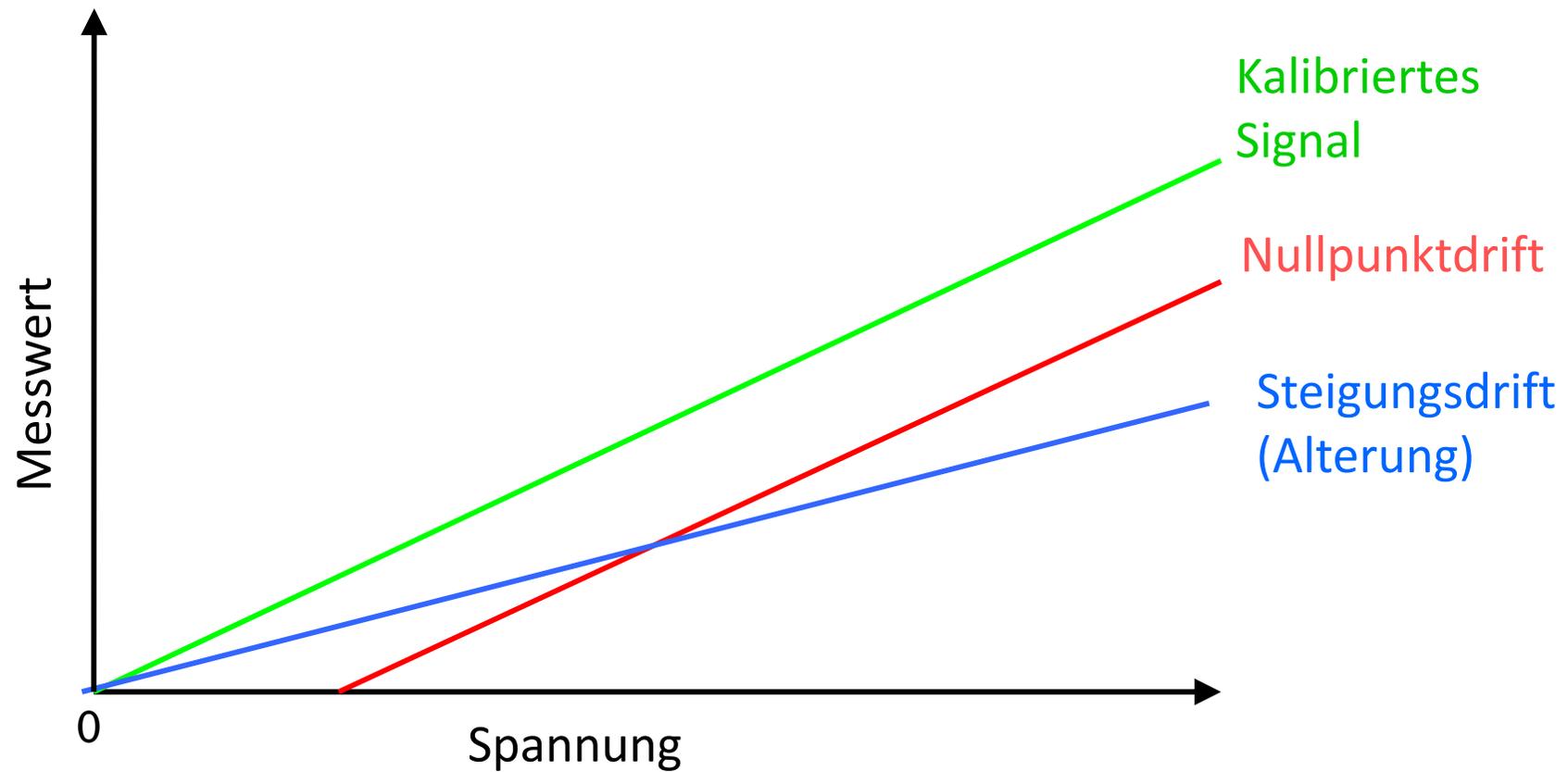
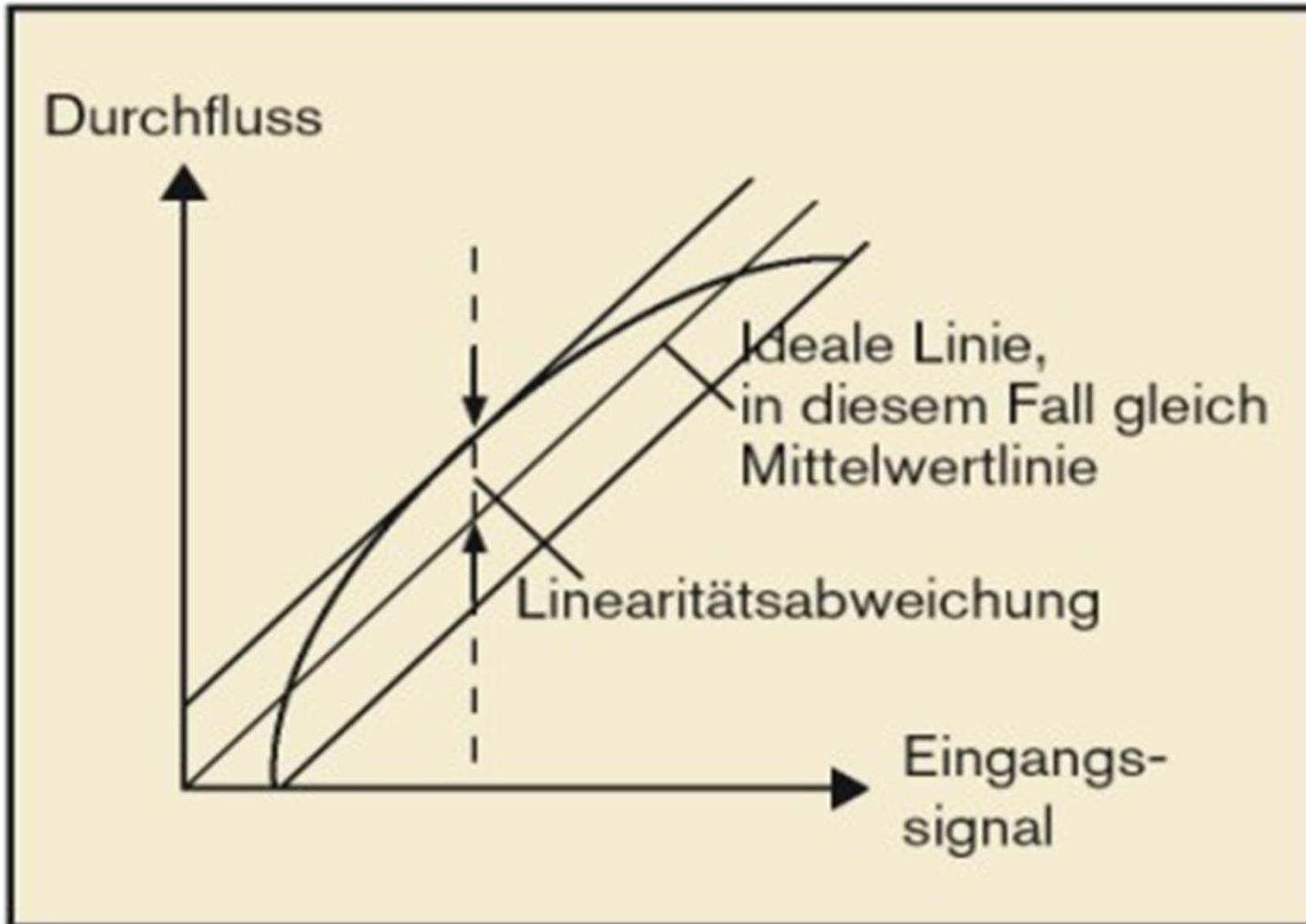


Abb.: Zeitverhalten eines Messsignals

# Messfehler (XII) - Driftarten



# Messfehler (XIII) - Linearität



# Messfehler (XIV) – Dynamisches Verhalten

- Jedes Messsystem besitzt eine bestimmte *Trägheit*, so dass das Ausgangssignal dem Eingangssignal nicht in einer beliebig kurzen Zeit folgen kann.
- Das *dynamische Verhalten* eines Messsystems beschreibt die Abhängigkeit der Ausgangsgröße von der Eingangsgröße bis zum Erreichen des stationären Zustands.

# Messgenauigkeit (II)

## Die Genauigkeit bestimmt sich aus der Abweichung der Anzeige

### Referenzbedingungen

Fehlergrenzen in Anlehnung an ISO/DIS 11631:

- 20...30 °C; 2...4 bar
- Kalibrieranlagen rückgeführt auf nationale Normale
- Nullpunkt unter Betriebsbedingungen abgeglichen
- Dichteabgleich durchgeführt

### Max. Messabweichung

Die angegebenen Werte beziehen sich jeweils auf den Impuls-/Frequenzausgang.  
Die Messabweichung beim Stromausgang beträgt zusätzlich typ.  $\pm 5 \mu\text{A}$ .

#### **Massedurchfluss (Flüssigkeit)**

$\pm 0,5\% \pm [(\text{Nullpunktstabilität} / \text{Messwert}) \times 100]\% \text{ v.M.}$

#### **Massedurchfluss (Gas)**

$\pm 1,0\% \pm [(\text{Nullpunktstabilität} / \text{Messwert}) \times 100]\% \text{ v.M.}$

#### **Volumendurchfluss (Flüssigkeit)**

$\pm 0,7\% \pm [(\text{Nullpunktstabilität} / \text{Messwert}) \times 100]\% \text{ v.M.}$

v.M. = vom momentanen Messwert

Steuerungsaufbau

# FUNKTIONSABLAUFPLAN

# Funktionsablaufplan

## Darstellungsmittel

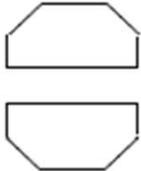
- Bei dem **Entwurf eines Programms** existieren/entstehen sehr umfangreiche und komplizierte Problemstellungen.
- **Folgen:**
  - Notwendigkeit einer **systematischen Vorgehensweise** für die Beschreibung eines Lösungsweges.
  - Verwendung **graphischer Darstellungsmittel** für die Logik eines Programmablaufs.
- **Vorteil:** Kontrollstrukturen und Verzweigungen lassen sich **anschaulich darstellen**.

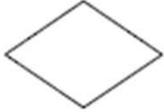
# Programmablaufplan

- Programmablaufplan ist genormt in DIN 66001.
- **Synonyme:** PAP; Ablauf-, Fluss- oder Blockdiagramm.
- PAP sind **Hilfsmittel** bei der Programmentwicklung auf Basis **graphischer Darstellungen** mit **genormten Symbolen**.
- Die **Programmstruktur** wird bildhaft **als Ablauf** dargestellt, wobei die Pfeile die Richtung der Verarbeitung anzeigen.
- PAP werden - je nach Komplexität des Programms - in mehreren Stufen erstellt.

# Programmablaufplan Grundelemente

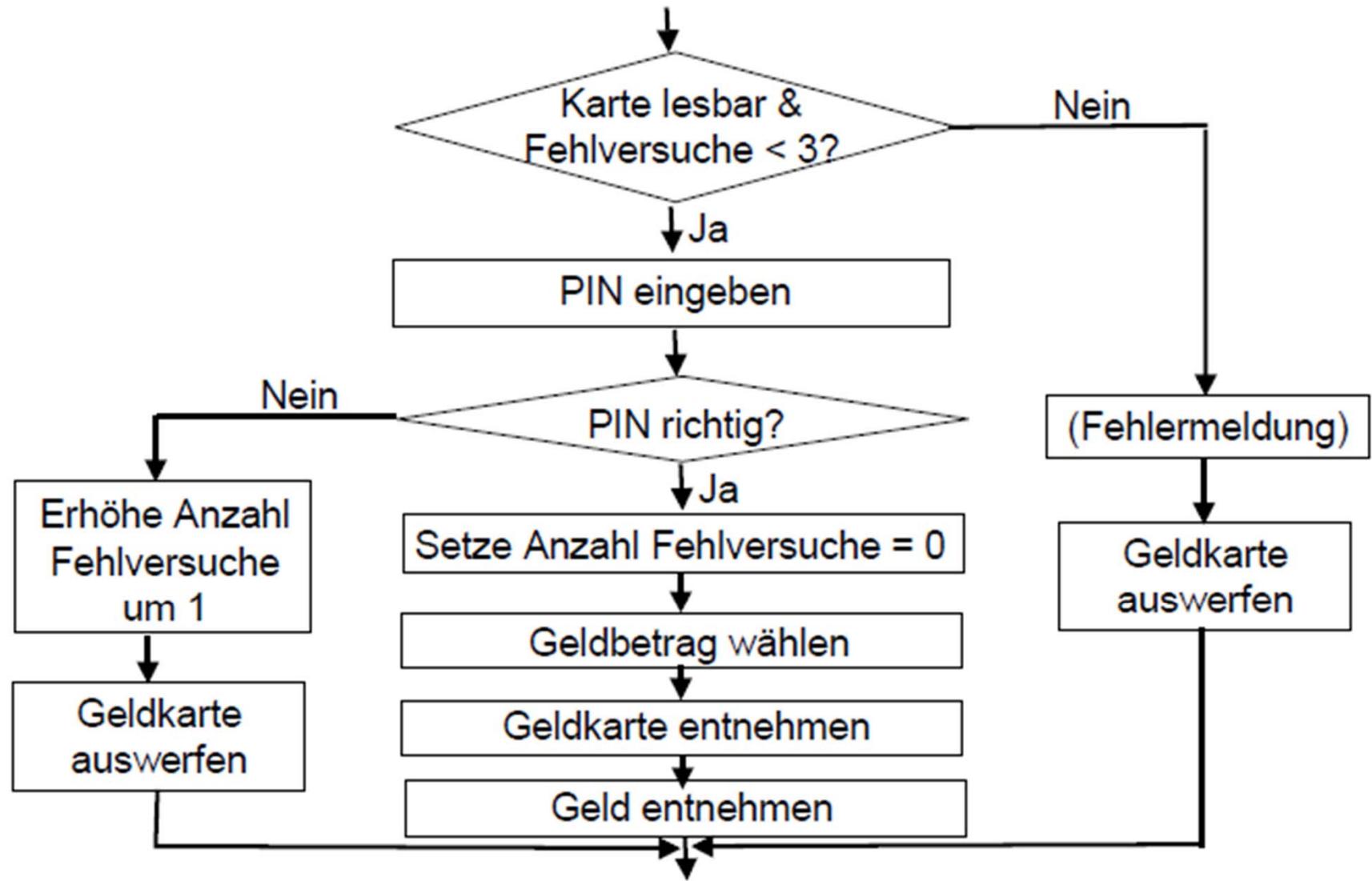
Symbol	Bedeutung	Erläuterung
	Allgemeine Verarbeitung	Die allgemeine Verarbeitung ist für <b>Zuweisungen</b> oder <b>Ein-/Ausgabeoperationen</b> vorgesehen.
	Verbindung	Zur <b>Verdeutlichung der Ablaufrichtung</b> werden Linien mit einer Pfeilspitze benutzt.

Symbol	Bedeutung	Erläuterung
	Grenzstelle	Eine Grenzstelle kennzeichnet den <b>Anfang</b> bzw. das <b>Ende</b> eines <b>PAP</b> .
	Schleifenbegrenzer	Darstellung von <b>Programmwiederholungen</b> ; kennzeichnen den <b>Anfang</b> bzw. das <b>Ende</b> einer <b>Schleife</b> .

Symbol	Bedeutung	Erläuterung
	Verzweigung	<p><b>Abfragen von Bedingungen</b> über Vergleichsoperatoren.</p> <p>I. d. R. führt eine Linie hinein und zwei führen heraus (true/false).</p> <p><b>Je nach Wahrheitswert</b> der Bedingungen wird der Ablauf in die jeweilige Richtung fortgeführt.</p>

Symbol	Bedeutung	Erläuterung
	Vordefinierter Prozess	Es wird auf einen <b>anderen PAP</b> hingewiesen, der z. B. ein Unterprogramm darstellt.
	Verbinder	<p>Teilt sich ein PAP und wird dann wieder zusammengeführt, wird ein sog. <b>Konnektor</b> eingesetzt.</p> <p>Somit wird vermieden, dass sich Ablauflinien kreuzen und die Übersichtlichkeit wird erhöht.</p>

# Beispiel Geldautomat



# Vorteile PAP

- Anweisungsteile der Algorithmen sind **übersichtlich und gut lesbar.**
- **Überprüfbarkeit der Terminierung und der Korrektheit.**
- **Methodik der schrittweisen Verfeinerung wird unterstützt,** da jede Kontrollstruktur als Blackbox betrachtet werden kann.

## Nachteile PAP

- Für **grundlegende Kontrollstrukturen existieren keine Symbole**, z. B. für Fallauswahlen, Vereinbarungsteile.
- **Schachtelstrukturen sind im PAP nicht gut zu erkennen.**
- **Strukturierte Programmierung** wird durch den PAP **nicht sichergestellt** (nur lineare Kontrollstrukturen zugelassen).

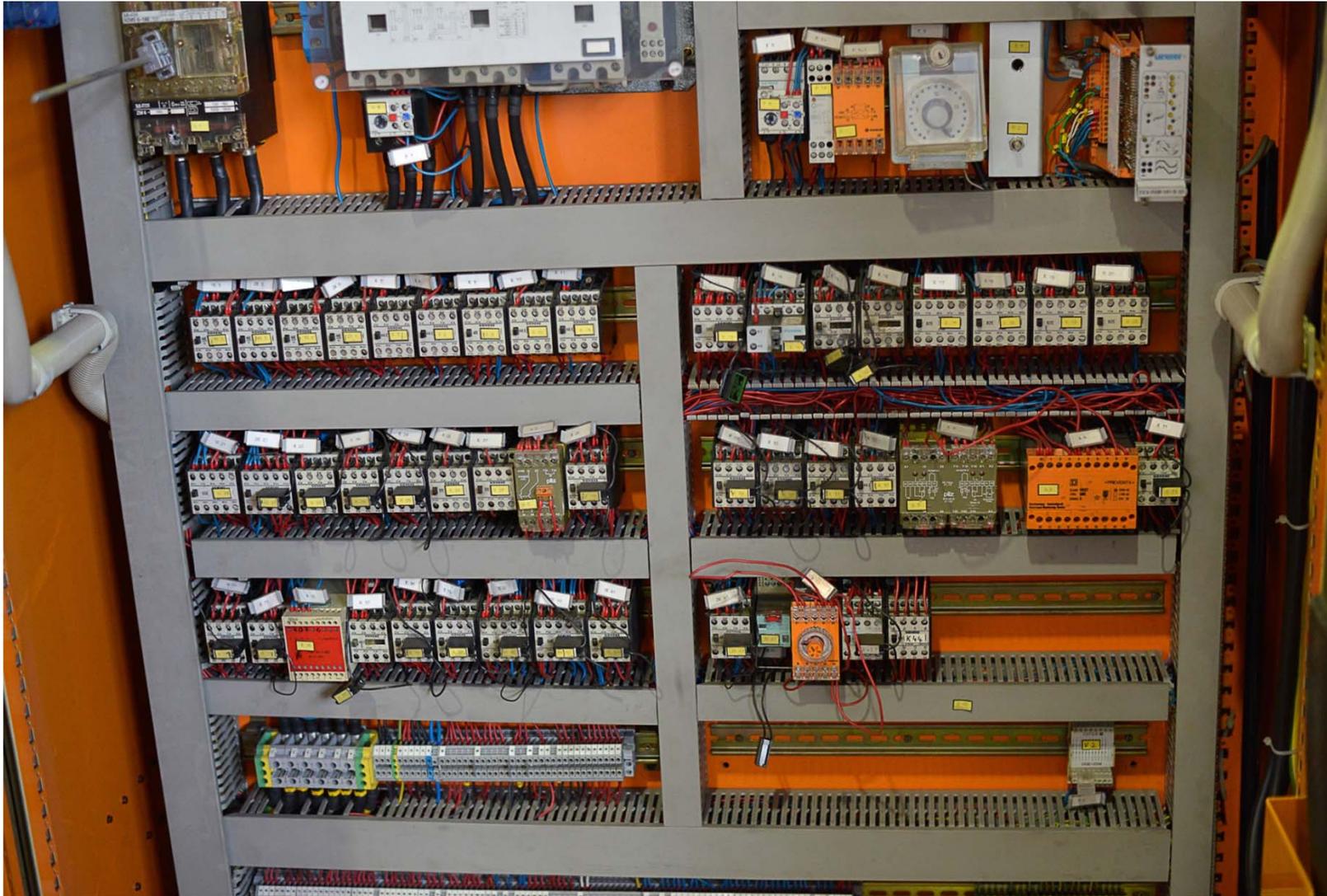
SPS-Ebene

# VPS vs. SPS

# VPS

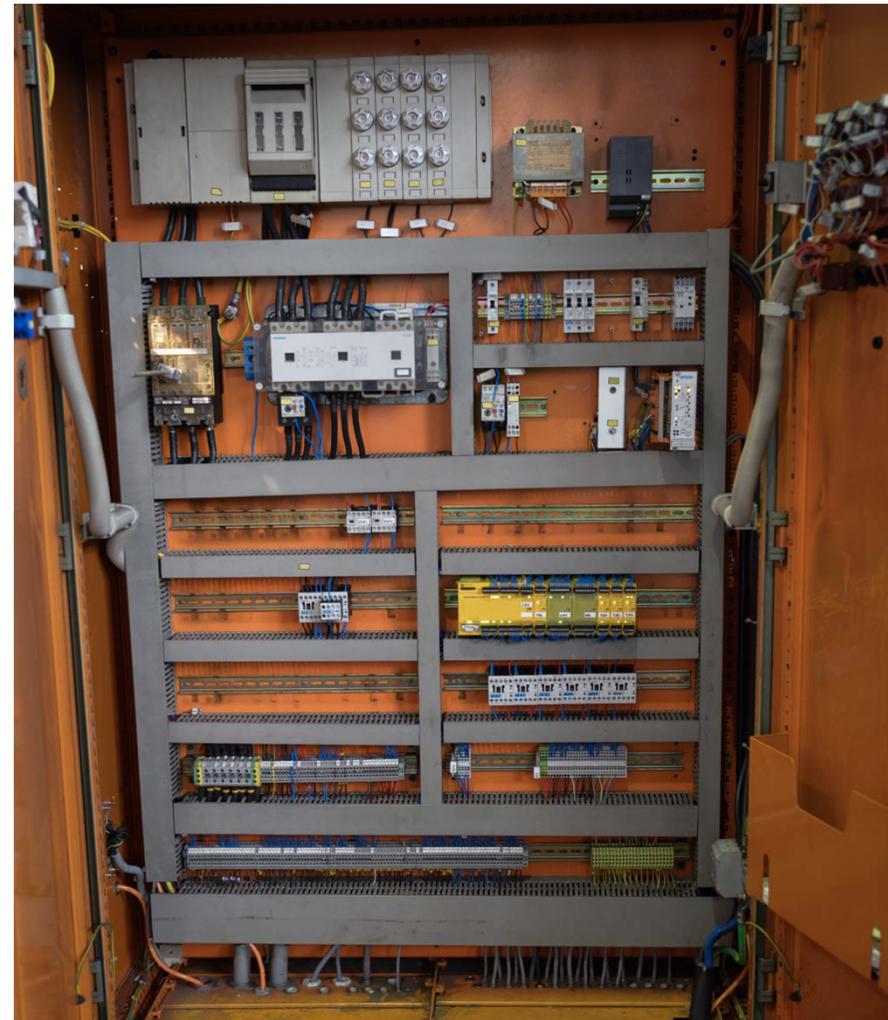
- Ursprünglich wurden Steuerungen mit Relais realisiert. Da einzelne Relais fest verdrahtet werden mussten, nennt man diese Art von Steuerung auch Verdrahtungsprogrammierte Steuerung.
- Ein Relais ist also ein Schalter der mit Hilfe von geringen Spannungen und Strömen hohe Spannungen und Ströme schaltet. Es gibt einen Steuerstromkreis und einen Arbeitsstromkreis. Im Steuerstromkreis befindet sich ein Elektromagnet. Sobald der Steuerstromkreis über einen Schalter geschlossen wird, zieht der Elektromagnet an und der Arbeitsstromkreis wird geschlossen. Wird der Schalter geöffnet, lässt der Elektromagnet los, man sagt auch er fällt ab und der Arbeitsstromkreis wird geöffnet. So ist es mit einem Relais möglich mit relativ kleinen Strömen und Spannungen Stromkreise zu steuern, die selbst hohe Spannungen und Ströme haben. Wenn man nun mit vielen Relais eine Steuerschaltung realisiert, nennt man das verdrahtungsprogrammierte oder verbindungsprogrammierte Steuerung (VPS). Eine VPS ist ein fest verdrahtetes Programm.

# VPS



# SPS

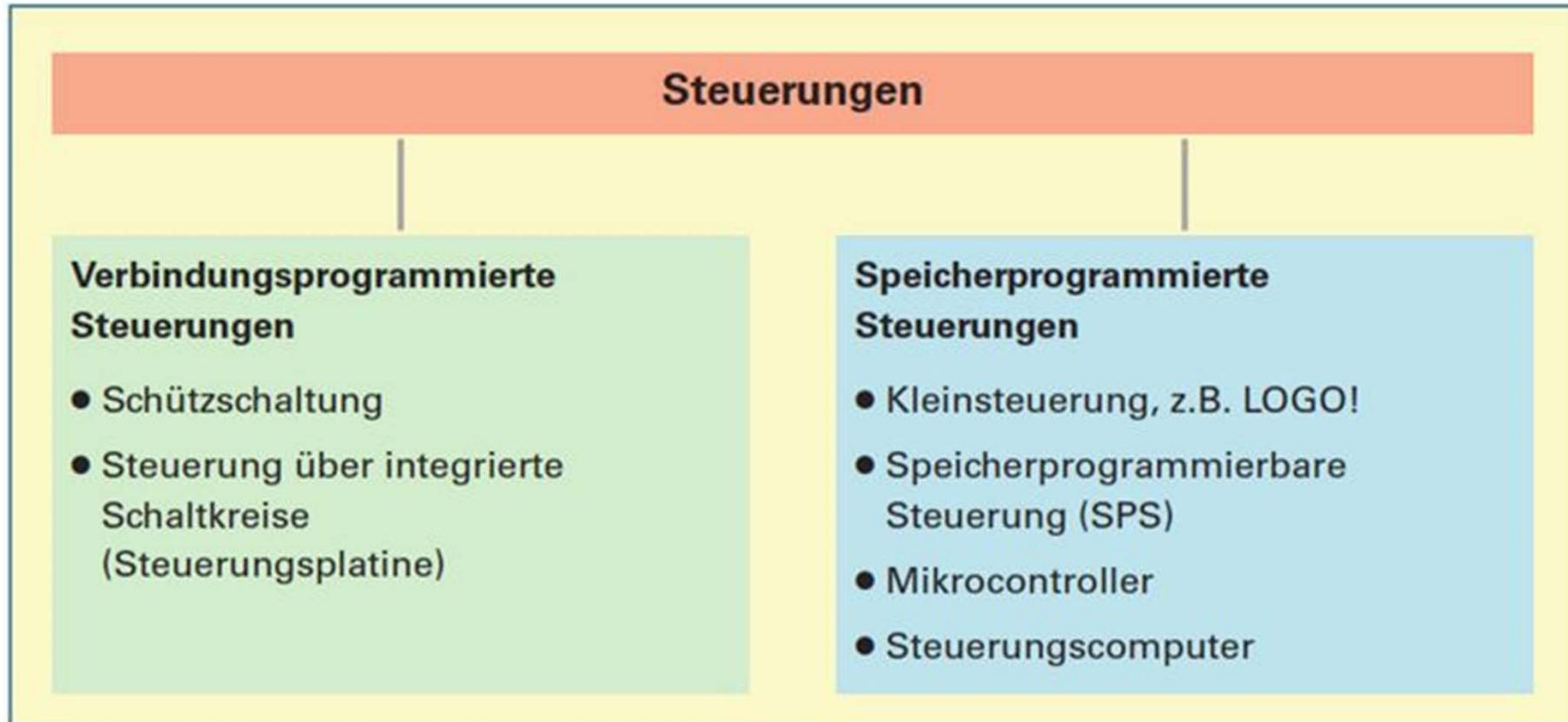
Bei einer speicherprogrammierten Steuerung wird die Steuerschaltung mit einer Software realisiert. Man schreibt also ein Programm, um die Steuerung von Arbeitsstromkreisen bzw. Laststromkreisen zu übernehmen. Durch diese Art der Steuerung ergeben sich viele Vorteile einer SPS gegenüber dem VPS.



# Vor- und Nachteile einer SPS

- Vorteile:
  - Anpassungsfähigkeit
  - Wartungsarmut
  - Platzersparnis
  - Automatische Programmdokumentation
  - Visualisierung ist möglich
  - Erhöhte Kommunikationsfähigkeit (Bussysteme)
  - Fernwartung ist möglich
- Nachteile:
  - Spezifische Fachkenntnisse erforderlich
  - Zusätzliche Kosten

# Arten von Steuerungen



# Definition SPS nach DIN EN 61131 Teil 1

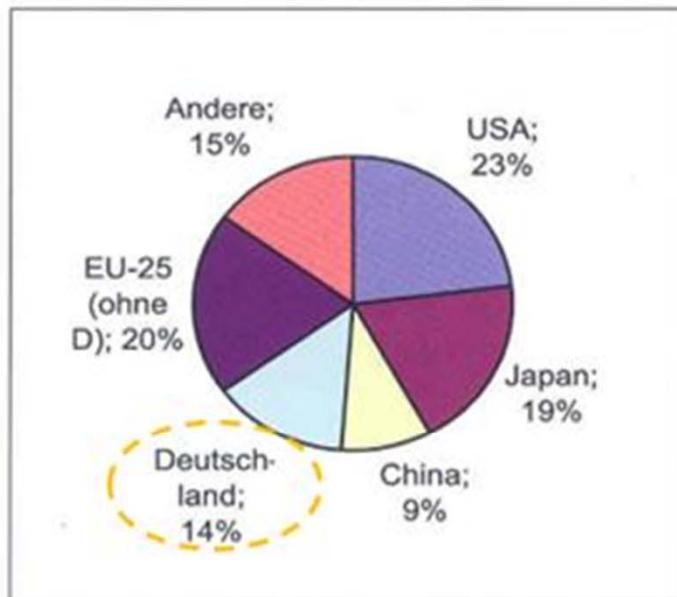
- „[...] Ein digital arbeitendes elektronisches System für den Einsatz in **industriellen Umgebungen** mit einem **programmierbaren Speicher** zur internen Speicherung der **anwenderorientierten Steuerungsanweisungen** zur Implementierung spezifischer Funktionen wie z.B.
  - Verknüpfungssteuerung,
  - Ablaufsteuerung,
  - Zeit-, Zähl- und arithmetische Funktionen,um durch **digitale oder analoge Eingangs- und Ausgangssignale** verschiedene Arten von Maschinen und Prozessen zu **steuern**. [...]“

Quelle:  TECHNISCHE  
UNIVERSITÄT  
DRESDEN

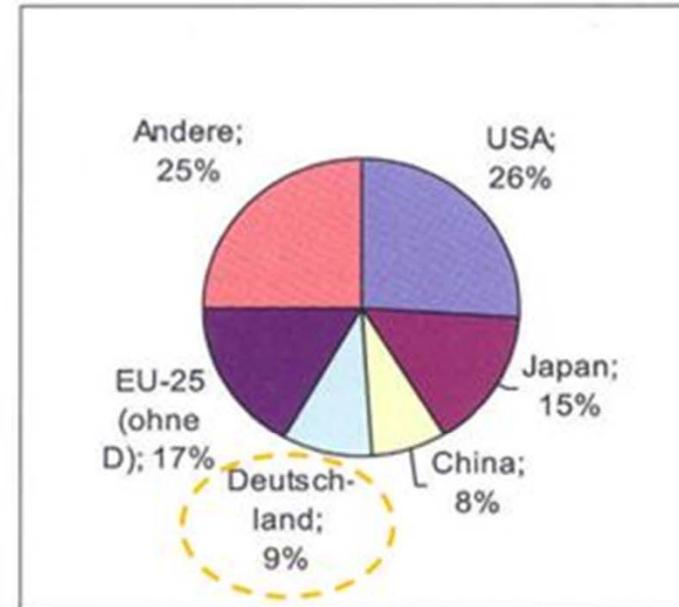
# Wirtschaftliche Bedeutung der Automatisierung

228 Mrd. EUR Gesamtvolumen (2006) für die elektrische Automation in Energie-, Verfahrens- und Fertigungstechnik

**Herstellung Automatisierung**  
(2006)



**Anwendung Automatisierung**  
(2006)



- Deutschland hat den größten Anteil am Export von Automatisierungsprodukten
- 9% des Weltmarktes der Automation kommt in Deutschland zum Einsatz
- Exportquote beträgt 77%

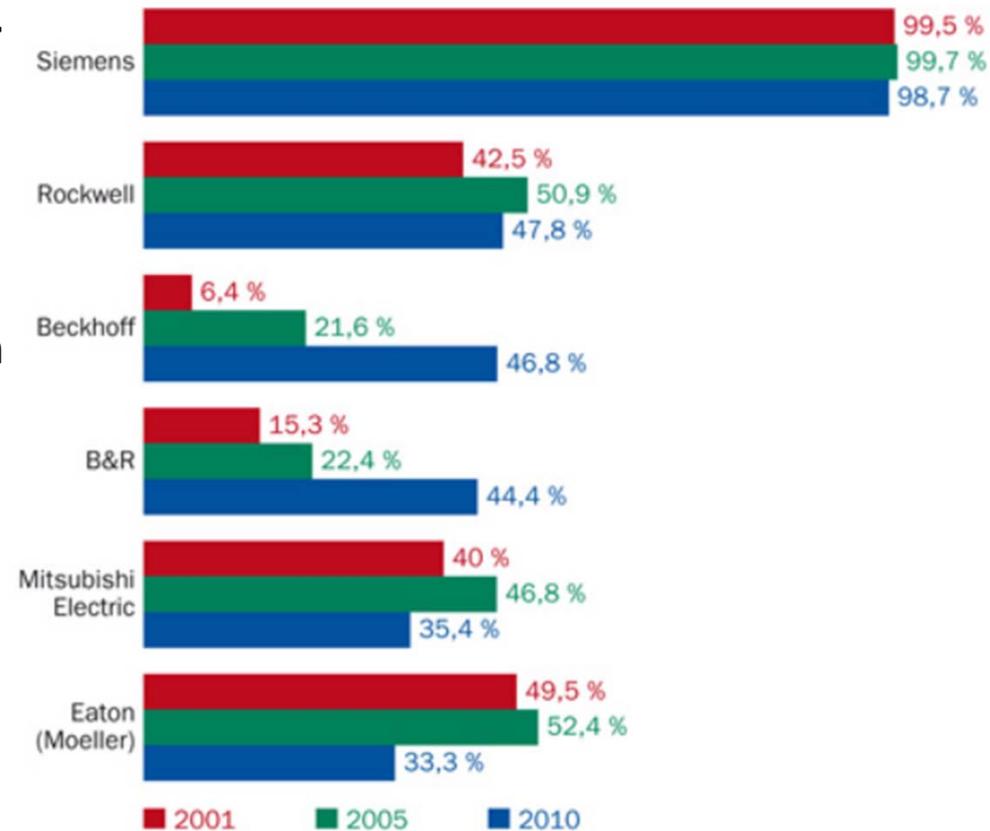
Quelle: IAS Universität Stuttgart

# Hersteller von SPS (1)

- Mehr als 100 verschiedene Hersteller
- Hauptauswahlkriterien
  - Preis (Anschaffung, Projektierung, Wartung, Ersatz),
  - Funktionalität (Leistung, Baugruppen, Erweiterbarkeit, Ausfallsicherheit),
  - Integration (Programmiersystem, Kommunikation).

## Hersteller von SPS (2)

- **Unabhängige Marktstudie der Fachhochschule Südwestfalen von 2009**
- Kaufverhalten bei SPS-Systemen
- Vergleich mit früheren Studien von 2001 und 2005
- Für aktuelle Studie deutschlandweit 486 Unternehmen aus den Bereichen Maschinenbau, Steuerungsbau und Ingenieurbüros anonym schriftlich befragt.



Auswertung der Antworten auf die Frage: Welche Hersteller von SPS-Systemen kennen Sie?

Quelle: Bernecker + Rainer Industrie Elektronik

# Hersteller von SPS (3)

## Herstellerübersicht: Industrie PC als SPS



**BECK**

[www.beck-ipc.com](http://www.beck-ipc.com)

**FESTO**

[www.festo.de](http://www.festo.de)



**ferrocontrol**

[www.ferrocontrol.de](http://www.ferrocontrol.de)

Quelle:  TECHNISCHE  
UNIVERSITÄT  
DRESDEN

# Hersteller von SPS (4)

## Herstellerübersicht 2



**KEYENCE**

[www.keyence.com](http://www.keyence.com)



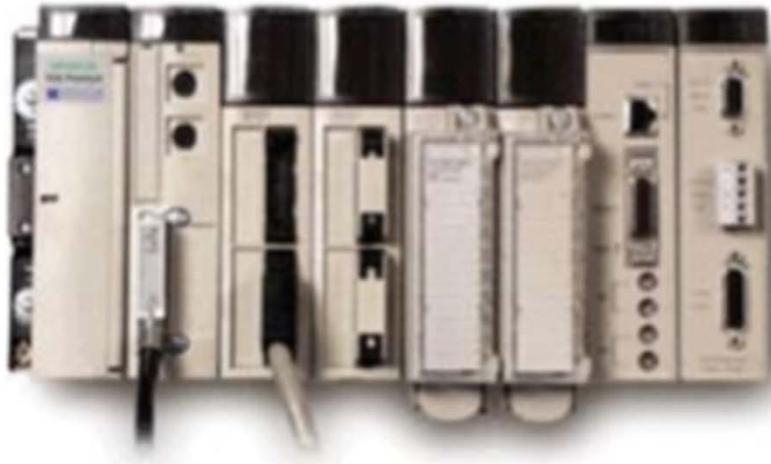
**KUHNKE**

[www.kuhnke.de](http://www.kuhnke.de)

Quelle:  TECHNISCHE  
UNIVERSITÄT  
DRESDEN

# Hersteller von SPS (5)

## Herstellerübersicht 3



**Schneider**  
 **Electric**  
[www.schneiderelectric.com](http://www.schneiderelectric.com)



[www.phoenix-contact.de](http://www.phoenix-contact.de)

Quelle:  TECHNISCHE  
UNIVERSITÄT  
DRESDEN

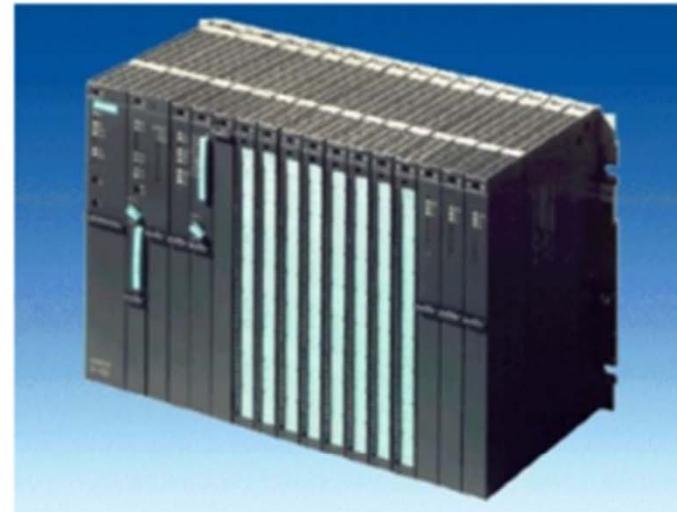
# Hersteller von SPS (6)

## Herstellerübersicht 4



**MOELLER** 

[www.moeller.net](http://www.moeller.net)



**SIEMENS**

[www.siemens.de](http://www.siemens.de)

Quelle:  TECHNISCHE  
UNIVERSITÄT  
DRESDEN

SPS-Ebene

# SOFTWARE/PROGRAMMIERUNG IEC 61131-3

# Kompatibilitätsproblem

- Anwender müssen mit SPS-Systemen verschiedener Hersteller umgehen können
  - Unterschiedliche Standards
  - Jeweils spezielle Vor- und Nachteile
  - Schulungen, Fehler
- Kommunikation und Kompatibilität verschiedener SPS-Systeme untereinander nur bedingt möglich

# Ziele der Norm IEC 61131-3

- Schnelle und einfache, herstellerunabhängige Erstellung des SPS-Programms.
  - Zeit- und Kosteneinsparungen
- Durch Standardisierung können Programme teilweise auf anderen Systemen wiederverwendet werden
  - Senkung Erstellungskosten

# Lösungsansatz IEC 61131

- DIN EN 61131 (IEC 61131) legt eine weltweite einheitliche Basis der SPS-Technik fest.
- Die Norm besteht aus 5 Teilen:
  - Begriffsbestimmungen und Funktionsmerkmale
  - Elektrische, mechanische und funktionelle Anforderungen
  - Fünf Programmiersprachen
  - Anwenderrichtlinien für alle Projektphasen
  - Kommunikation von SPS unterschiedlicher Hersteller
- Die Umsetzung der Norm ist noch nicht abgeschlossen!

# Programmbausteine (1)

- Ein SPS-Programm muss in verschiedenartige Bausteine unterteilt werden, die unter verschiedenen Bedingungen, aber in der Praxis meistens seriell-zyklisch bearbeitet werden.
- POE  
**Programm-Organisationseinheiten** / Von der IEC genormt: Oberbegriff für die Bausteintypen, die zur Programmstrukturierung zur Verfügung stehen.

## Programmbausteine (2)

- Bei der SPS-Programmierung kann der Anwender sein Programm in den OB-, FC- oder FB-Programmbausteinen einstellen.
- Bei der Aufteilung des SPS-Programmcodes gibt es viele unterschiedliche Ansichten der Umsetzung, wobei die Gliederung des Anwenderprogramms in einer einfachen, übersichtlichen Programmstruktur gehalten werden sollte. Die ist sinnvoll für die schnellere eigene Einarbeitung in das SPS-Programm und für die Einarbeitung anderer SPS-Techniker.

# Organisationsbausteine - OBs

- Diese Organisationsbausteine (OB-Bausteine) werden vom System zur Verfügung gestellt, sie dienen hauptsächlich der Organisation der anderen Bausteine. Die Anzahl der möglichen OB-Bausteine wird von der SPS-CPU bestimmt, wobei der OB1 immer vorhanden ist.
- Dieser Baustein wird zyklisch aufgerufen. In diesem OB1 werden häufig die grundlegenden Signale/Daten für das weitere SPS-Anwenderprogramm erstellt und in die einzelnen Anwenderprogrammteile aufgerufen.
- Es sind für unterschiedliche Situationen oder für die Anwendungsfälle verschieden OB's vorgesehen. Dabei spielt die Fehlerauswertung eine wichtige Rolle. Sollte das SPS-System einen Fehler erkennen, wird versucht einen Fehler-OB aufzurufen. Sollte dieser Fehler-OB nicht im Anwenderprogramm vorhanden sein, geht die SPS-Steuerung in Stop.

# Organisationsbausteine in der EN 61131-3

- In der Norm EN 61131-3 sind Tasks vorgesehen. Diese werden in Step7 in Form von Organisationsbausteinen bereitgestellt und sind die Schnittstelle zwischen dem Betriebssystem und dem Anwenderprogramm. Sie werden ausschließlich bei bestimmten Ereignissen vom Betriebssystem gestartet.
- Im OB1 befindet sich das Anwenderprogramm bzw. wird hier der Aufruf weiterer Funktionen oder Funktionsbausteine veranlasst. Solange keine besonderen Ereignisse oder Störungen auftreten, wird der Organisationsbaustein OB1 zyklisch abgearbeitet. Damit der OB1 bei Störungen die Abarbeitung unterbricht, sind Organisationsbausteine nach Prioritäten kategorisiert.
- Die OB's mit höherer Priorität stoppen die OB's mit der niedrigeren Priorität und arbeiten ihr eigenes Programm ab. Sobald ein OB mit der höheren Priorität das Programm abgearbeitet hat, wird mit der Abarbeitung des OB mit der niedrigeren Priorität fortgesetzt. Die Prioritäten sind voreingestellt und bei einigen OB's können sie sich innerhalb bestimmter Grenzen ändern. Die OB's können entweder aus der Bibliothek genommen oder selbst erstellt werden.

# Programmbausteine

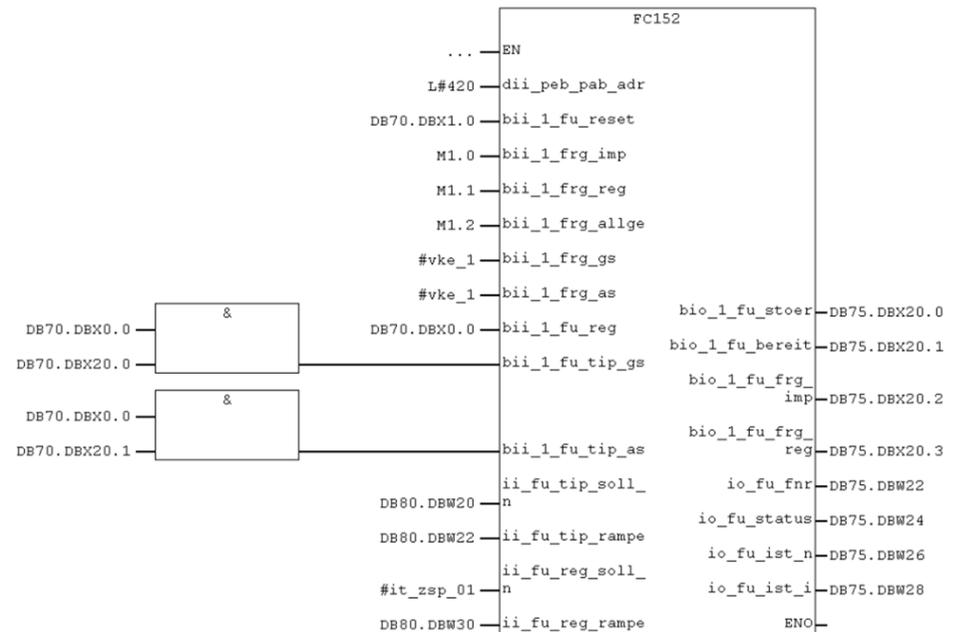
- Bei der **SPS-Programmierung** sind für das Anwenderprogramm die **FC- oder FB-Programmbausteine** vorgesehen. Neben der gewohnten Möglichkeit der Programmierung, die Signale direkt über Anweisungen zu verarbeiten, können diese Bausteine sehr flexibel eingesetzt werden, indem man ihre Möglichkeiten zur Parametrierung nutzt.
- **Folgende Übergabebereiche stehen beiden Bausteinarten zur Verfügung:**
  - INPUT
  - OUTPUT

# Funktionen - FCs

- Unterprogramm, dass aus Organisationsbausteinen oder anderen Funktionen/Funktionsbausteinen heraus aufgerufen werden kann. Eine Funktion kann einen Funktionswert zurückgeben oder mehrere Ausgabeparameter besitzen.
- Die FC-Bausteine werden gerne für kleine parametrierbare Programmfunktionen verwendet. Die FC-Bausteine haben keine permanenten Speicherbereiche (eigenes Gedächtnis), die vom System genutzt werden können, das heißt die lokalen Daten gehen nach der Bearbeitung der Funktion verloren.

Netzwerk 33: Titel:

Kommentar:



# Zusammenfassung Funktionen – FCs

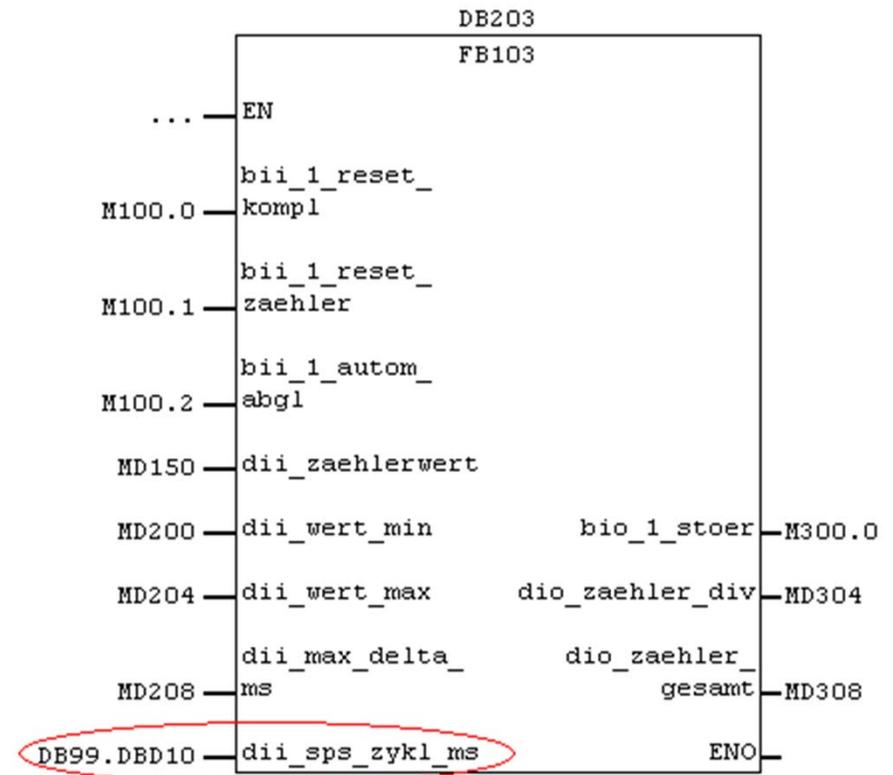
- Funktionen werden oft mit dem Kürzel FC bezeichnet.
- SFC sind Systemfunktionen, die im Betriebssystem der CPU integriert sind.
- Funktionen sind Unterprogramme, die in strukturierten Programmen eine Teilaufgabe übernehmen und für immer wiederkehrende Aufgaben eingesetzt werden können.
- Funktionen werden entweder von Organisationsbausteinen (z.B. OB1) oder von anderen Funktionen oder Funktionsbausteinen aufgerufen.
- Funktionen besitzen im Gegensatz zu Funktionsbausteinen keinen Speicherbereich, die über den Bausteinaufruf hinaus gültig sind. Nach der Bearbeitung der Funktion gehen die lokalen Daten (z.B. Temperatur) verloren.
- In Funktionen können z.B. Operanden natürlich direkt eingesetzt werden, jedoch ist diese Methode nicht empfehlenswert und so auch nicht vorgesehen. Im Idealfall werden Funktionen mit Formalparametern parametrisiert. Erst beim Aufruf werden sie mit Aktualparametern versorgt.

# Funktionsbausteine – FBs (1)

- Funktionsbausteine (FB) stellen eine Erweiterung von Funktionen (FC) dar.
- Im Gegensatz zur Funktion kann ein Funktionsbaustein Daten speichern. Die Daten selbst werden nicht im Funktionsbaustein gespeichert, sondern in einem diesem zugeordneten Datenbaustein (DB). Datenbausteine, die als Datenspeicher von Funktionsbausteinen dienen, werden als Instanz-Datenbausteine bezeichnet. Ein Instanz-Datenbaustein besitzt immer die gleiche Datenstruktur wie der ihm zugeordnete Funktionsbaustein. Im Funktionsbaustein werden die Variablen deklariert und das eigentliche Programm geschrieben, wohingegen im Datenbaustein nicht programmiert wird. Im Instanz-Datenbaustein werden statische Daten gespeichert oder im Funktionsbaustein berechnete Werte.

# Funktionsbausteine – FBs (2)

- Der Vorteil eines FB-Bausteins ist sein Gedächtnis.
- Für wiederkehrende, umfangreiche Standardfunktion werden ebenfalls die FB's verwendet, um dort die Signale/Daten zu speichern, die nur von der entsprechenden Funktion benötigt werden (z.B. Regelungen, umfangreiche Ansteuerungen, Schrittketten).



# Datenbausteine – DBs (1)

- In den Datenbausteinen können Variable als einzelne Elemente (Bool, Byte, INT, DINT, REAL, usw.) angelegt werden. Mit der Aufteilung der Datenbausteine in verschiedene Gruppen (Einstellungen, Bedienung, Störungen, Funktionen, Anlagengruppen, usw.) und der weiteren Unterteilung in Strukturen und Datenfeldern kann eine übersichtliche Aufgliederung der Signale/Daten erstellt werden.
- Ein weiterer **Vorteil der Datenbausteine** ist, dass diese den letzten Signalzustand behalten, der vor dem Stop der CPU vorhanden war. Aufgrund dieser Speicherfunktionalität der **DB's** ist der Programmierer dazu gezwungen, sich bewusst zu machen, welche Signale/Daten beim Anlaufen zurückgesetzt werden müssen.
- **Achtung!** Das Übertragen von Datenbausteinen während der SPS-CPU im Run Zustand, kann zu unerwarteten Reaktion der Anlage führen.

## Datenbausteine – DBs (2)

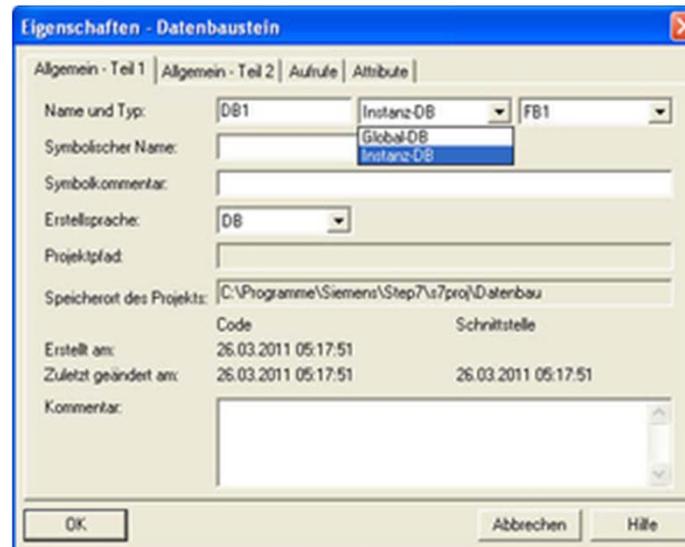
- In einem Datenbaustein wird kein Code abgelegt, sondern nur Daten. Als Daten kämen z.B. Stückzahlen, Uhrzeiten, Namen der Fertigungsmitarbeiter und viele weitere Informationen in Frage. Daher ist die Festlegung des Datentyps wichtig. Für Stückzahlen benötigt man z.B. einen anderen Datentyp als für Namen, Zeit oder Datum. In den Codebausteinen kann während des laufenden Betriebs auf diese gespeicherten Daten zurückgegriffen werden.
- Datenbausteine enthalten kein Programmteil, sie enthalten nur den Deklarationsteil.
- In einem Datenbaustein wird kein Code abgelegt, sondern nur Daten. Als Daten kämen z.B. Stückzahlen, Uhrzeiten, Namen der Fertigungsmitarbeiter und viele weitere Informationen in Frage. Daher ist die Festlegung des Datentyps wichtig. Für Stückzahlen benötigt man z.B. einen anderen Datentyp als für Namen, Zeit oder Datum. In den Codebausteinen kann während des laufenden Betriebs auf diese gespeicherten Daten zurückgegriffen werden.

# Datenbausteine – DBs (3)

Adresse	Name	Typ	Anfangswert	Kommentar
0.0		STRUCT		
+0.0	nv_zunder_abb_zeit	DINT	L#0	nv-zunder abblasen: soll-zeit
+4.0	wks_ev_zeit	DINT	L#0	werkzeugspanvorrichtung: soll-zeit
+8.0	r_008	ARRAY[0..99]		
+1.0		BYTE		
+100.0	stoessel	STRUCT		stössselverstellung
+0.0	ein_auf	BOOL	FALSE	l=dauer ein auf
+0.1	ein_ab	BOOL	FALSE	l=dauer ein ab
+0.2	aus	BOOL	FALSE	l=aus
+0.3	tip_auf	BOOL	FALSE	l=tippen auf
+0.4	tip_ab	BOOL	FALSE	l=tippen ab
+2.0		END_STRUCT		
+102.0	zaehler	STRUCT		bedienung zähler
+0.0	gesamt_reset_t1	BOOL	FALSE	l=gesamtzähler auf null setzen (teil 1)
+0.1	gesamt_reset_t2	BOOL	FALSE	l=gesamtzähler auf null setzen (teil 2)
+2.0		END_STRUCT		
+104.0	sicherheit	STRUCT		
+0.0	schutzzaun_ok	BOOL	FALSE	l=schutzzaun ist ok
+2.0		END_STRUCT		
+106.0	r_0106	ARRAY[106..199]		
+1.0		BYTE		
+200.0		END_STRUCT		

# Datenbausteine – DBs (4)

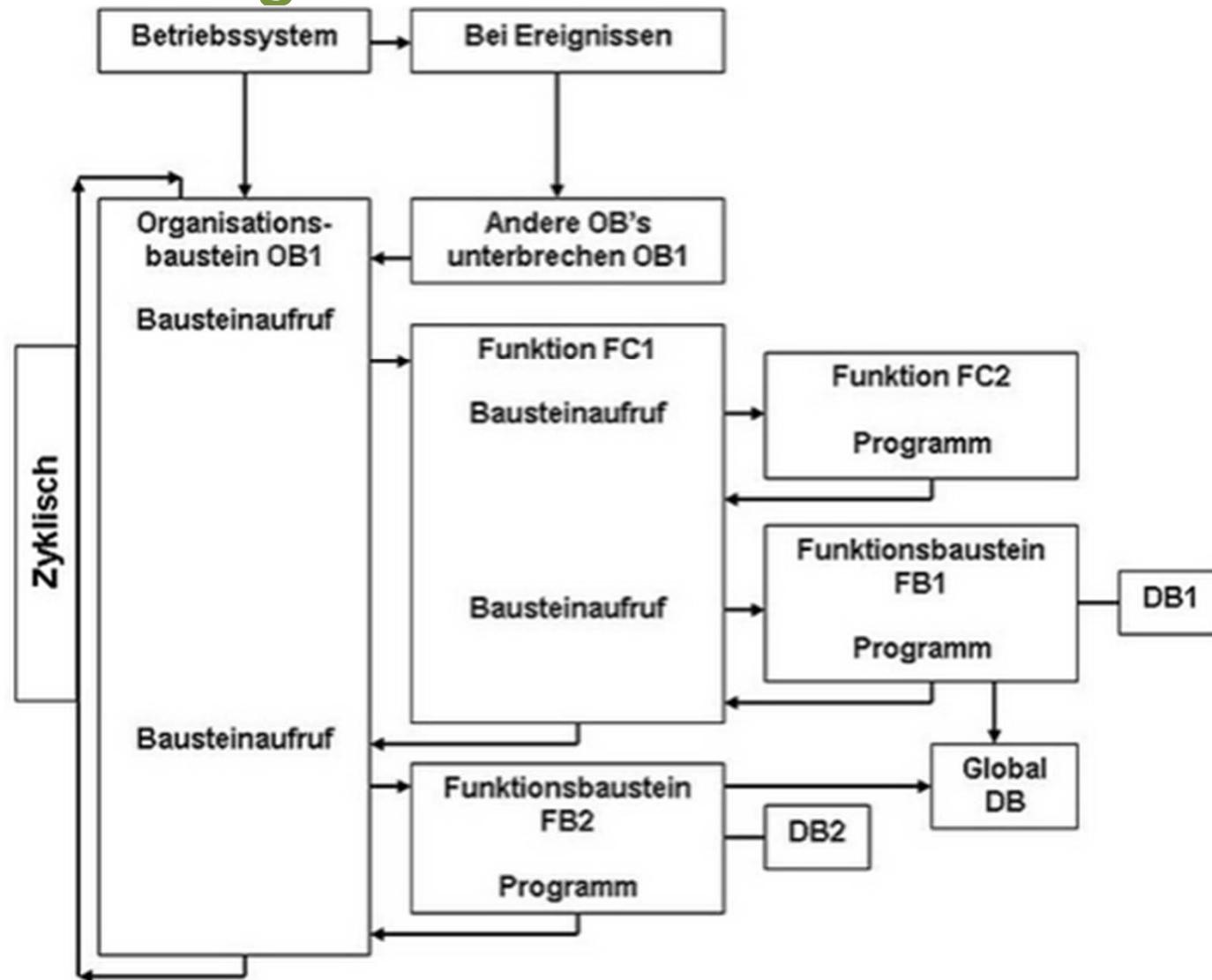
- Im einem Fenster werden die Einstellungen für den Datenbaustein eingegeben. Insbesondere ist hier der **Name des Datenbausteins** wichtig. Über den Namen kann später der Datenbaustein ausgewählt und auf die Werte zugegriffen werden. Es ist auch wichtig auszuwählen, ob der Datenbaustein ein Globaldatenbaustein ist oder ein Instanzdatenbaustein.
  - Bei Typ ist die Option Instanzdatenbausteine erst verfügbar, wenn zuerst ein Funktionsbaustein existiert. Denn ein Instanzdatenbaustein ist der Datenbaustein für einen Funktionsbaustein. Ist ein Funktionsbaustein vorhanden und wurde als Typ Instanz-DB ausgewählt, kann, sofern mehrere Funktionsbausteine vorhanden sind, der Funktionsbaustein ausgewählt werden, dem der Datenbaustein zugeordnet werden soll.



## Datenbausteine – DBs (5)

- Datenbausteine können auch mit symbolischen Namen und einem Symbolkommentar versehen werden, z.B. Prüfdaten, Akustikwerte etc. So kann bei symbolischer Programmierung mit den symbolischen Namen gearbeitet werden. Denn einfacher ist es symbolische Namen wie z.B. den Datenbaustein "Pruefdaten" zu merken als beispielsweise DB26.
- Die Erstsprache ist bei Datenbausteinen standardmäßig immer DB. Projektpfad und Speicherort werden automatisch vergeben und können nicht verändert werden. Zum Schluss kann der Datenbaustein mit einem längeren Kommentar versehen werden.

# Übersicht Programmaufbau



Schematische Darstellung einer Programmstruktur mit Codebausteinen und Datenbausteinen

# Mathematische Grundlage

- Mathematik ist die Grundlage für die Programmierung und Funktion eines Computers und somit auch einer SPS. Die Menschen denken und arbeiten in der Regel im Dezimalsystem. Im Dezimalsystem kann man Rechenoperationen wie Addition, Subtraktion, Multiplikation, Division, Potenzen usw. durchführen und den Menschen ist der Umgang mit dem Dezimalsystem bekannt.
- Ein Computer, wozu auch eine SPS zählt, arbeitet jedoch nicht mit dem Dezimalsystem. Die Ursache hierfür liegt im technischen Grundaufbau eines Computers, der eben nur erkennen kann, ob Strom fließt oder nicht fließt. Für den Computer ist daher nur erkennbar, ob das Signal 1 (Strom fließt) oder 0 (Strom fließt nicht) ist.
- Daher benötigt eine SPS ein Zahlensystem, mit dem gearbeitet werden kann und das ist grundsätzlich das Dualsystem. Das Dualsystem kennt nur zwei Ziffern (0, 1) und eignet sich daher sehr gut für die interne Verarbeitung von Zahlen in einer SPS. Das Dualsystem hat jedoch einen großen Nachteil. Für selbst kleinste Zahlen benötigt man riesige Ziffernfolgen. Daher haben sich zwei weitere Zahlensysteme etabliert, mit denen bei der SPS-Programmierung gearbeitet wird. Das sind die Zahlensysteme Hexadezimalsystem und BCD Code.

# Zahlensysteme

- Ein SPS-Programmierer sollte sich daher mit diesen 3 Zahlensystemen vertraut machen. Häufig werden Eingaben in BCD-Format, als Hexadezimalzahl oder als Dualzahl erwartet oder man muss die entsprechenden Ziffernfolgen auswerten können.
- Intern wird jedoch immer mit dem Dualsystem gearbeitet, da eine SPS nur die logischen Zustände 0 und 1 (An und Aus) kennt und die Arbeitsweise einer Schaltung durch diese beiden logischen Zustände beschrieben werden kann.

# Merkmale von Zahlensystemen

- Zahlensysteme werden durch drei Merkmale beschrieben:
  - **Basis:** Gibt im Zahlensystem die Anzahl der Ziffern an.
    - Im Dezimalsystem hat man z.B. 10 Ziffern (0 - 9), also ist die Basis 10.  $1_{10}$
    - Im Dualsystem hat man 2 Ziffern (0, 1), also ist die Basis 2.  $1_2$
    - Im Hexadezimalsystem hat man 16 Ziffern (0 - 9 und A - F).  $1_{16}$
  - **Ziffern:** Damit wird angegeben, welche Ziffern im jeweiligen Zahlensystem Gültigkeit haben.
    - Die Ziffer X kommt z.B. im Dezimalsystem nicht vor. Bei den römischen Zahlen wäre es dagegen gültig.
  - **Stellenwerte:** Der Stellenwert einer Ziffer gibt, jeweils höherwertig von rechts nach links, die Wertigkeit im Zahlensystem an.
    - Die Ziffernfolge 10 hat im Dezimalsystem die Wertigkeit von 10, da die Basis 10 ist und die Ziffer 1 sich an der zweiten Stelle befindet und daher mit  $10^1$  multipliziert wird.
    - Im Dualsystem hätte die Ziffernfolge nur die Wertigkeit von 2 (Multiplikation mit  $2^1$ ). Denn die Basis der Dualzahl ist 2.
    - Im Hexadezimalsystem hätte die Ziffernfolge 10 die Wertigkeit von 16 (Multiplikation mit  $16^1$ ).
    - Für jede Stelle nach links muss man den Exponent der Potenz um 1 erhöhen, um die Wertigkeit zu ermitteln. Beispielsweise hätte die Ziffernfolge 100 im Hexadezimalsystem die Wertigkeit von 256 ( $1 \times 16 \times 16$  oder  $1 \times 16^2$ ). Die Ziffernfolge 1000 hätte die Wertigkeit von 4096 ( $1 \times 16 \times 16 \times 16$  oder  $1 \times 16^3$ ).

# Zu Übung: Umrechnen von Zahlensystemen

- Jede Stelle einer anderen Zahl hat eine Wertigkeit, diese wird in der folgenden Form ausgedrückt.

- Allgemein gilt:

$$- N = d_0 * R^0 + d_1 * R^1 + \dots + d_n * R^n$$

- N: Zahl im Zahlensystem
- R: Basis
- $R^n$ : Wertigkeit der n-ten Stelle
- $d_i$ : Ziffer der Stelle n

- Beispiele:

$$- 1689_{10} = 9 * 10^0 + 8 * 10^1 + 6 * 10^2 + 1 * 10^3 = 1689$$

$$- 1011_2 = 1 * 2^0 + 1 * 2^1 + 0 * 2^2 + 1 * 2^3 = 11$$

# Übung: Wandlung Dual- in das Dezimalsystem

- Gegeben Dualzahl: 110101
- Gesucht Dezimalzahl:

1	1	0	1	0	1	Dual					
$1 * 2^5$	+	$1 * 2^4$	+	$0 * 2^3$	+	$1 * 2^2$	+	$0 * 2^1$	+	$1 * 2^0$	Dual
32	+	16	+	0	+	4	+	0	+	1	Dez
<hr/>											
53											Dez

# Übung: Wandlung Hexa- in das Dezimalsystem

- Gegeben Hexadezimalzahl:  $FA7E_{16}$
- Gesucht Dezimalzahl:

$F$		$A$		$7$		$E$		Hexa
$F * 16^3$	+	$A * 16^2$	+	$7 * 16^1$	+	$E * 16^0$		Hexa
$15 * 16^3$	+	$10 * 16^2$	+	$7 * 16^1$	+	$14 * 16^0$		Dez
61440	+	2560	+	112	+	14		Dez
<hr/>								
64126								Dez

# Übung: Wandlung Hexa- aus dem Dezimalsystem

- Gegeben Dezimalzahl: 61604
- Gesucht Hexadezimalzahl:

				Ganzzahl quotient	Rest	Umrechnungs ergebnis
61604	:	16	=	3850 * 16 +	4	4
2850	:	16	=	240 * 16 +	10 (A)	A4
240	:	16	=	15 * 16 +	0	0A4
15	:	16	=	0 * 16 +	15 (F)	F0A4

---

$$F0A4_{16} \triangleq 61604_{10}$$

# Übung: Wandlung Dual- aus dem Dezimalsystem

- Gegeben Dezimalzahl: 53
- Gesucht Dualzahl:

				Ganzzahl quotient	Rest	Umrechnungs ergebnis	
53	:	2	=	26 * 2 + 1	26	1	1
26	:	2	=	13 * 2 + 0	13	0	01
13	:	2	=	6 * 2 + 1	6	1	101
6	:	2	=	3 * 2 + 0	3	0	0101
3	:	2	=	1 * 2 + 1	1	1	10101
1	:	2	=	0 * 2 + 1	0	1	110101

---

$110101_2 \triangleq 53_{10}$

# Übung: Wandlung Dual- ins Hexadezimalsystem

- Gegeben Dualzahl:  $101010110110101$ 
  - Tipp: Die Dualzahl kann von rechts in Viererblöcke unterteilt werden!
- Gesucht Dualzahl:

101	0101	1011	0101	Dual
$1 * 2^2 + 1 * 2^0$	$1 * 2^2 + 1 * 2^0$	$1 * 2^3 + 1 * 2^1 + 1 * 2^0$	$1 * 2^2 + 1 * 2^0$	Dez
4+1	4+1	8+2+1	4+1	Dez
5	5	B (11)	5	Hexa

---

$55B5_{16} \triangleq 101010110110101_2$

# Beispiele mit Ziffernfolgen bei Zahlensystemen

Dezimalsystem	Dualsystem	Hexadezimalsystem	BCD-Code
0	0	0	0000
1	1	1	0001
2	10	2	0010
3	11	3	0011
4	100	4	0100
5	101	5	0101
6	110	6	0110
7	111	7	0111
8	1000	8	1000
9	1001	9	1001
10	1010	A	0001 0000
11	1011	B	0001 0001
12	1100	C	0001 0010
13	1101	D	0001 0011
14	1110	E	0001 0100
15	1111	F	0001 0101
16	10000	10	0001 0110
17	10001	11	0001 0111
47	101111	2F	0100 0111
82	1010010	52	1000 0010
99	1100011	63	1001 1001
100	1100100	64	0001 0000 0000

# Bit und Byte (1)

- Die Einheit des Dualsystems wird als binäres Signal bezeichnet oder anders ausgedrückt **BIT** genannt. Ein Bit ist im Grunde eine Ziffer im dualen Zahlensystem, der nur die beiden Werte 0 und 1 haben kann. **Bit** ist die Abkürzung für **Binary Digit** und stellt die kleinste Einheit in der Computertechnologie dar.
- Nun hat es keinen Sinn, lediglich die Ziffern 0 und 1 darzustellen. Vielmehr sollen technologisch gesehen höhere Zahlenwerte verarbeitet werden. Der Trick liegt darin, dass man die Bitlängen erhöht.
  - Mit 2 Bits kann man z.B. die Zahlen 00, 01, 10 und 11 darstellen, womit man bereits 4 Zustände hätte.

## Bit und Byte (2)

- Für die Erweiterung der Bitlängen hat man Namen vergeben.
- Diese sind:
  - **Halbbyte:** Wenn 4 Bits zusammengefasst werden, nennt man das Halbbyte oder auch Nybble.
  - **Byte:** 8 Bits werden zu einem Byte zusammengefasst. Das wären 2 Halbbytes. Mit 1 Byte kann man 256 verschiedene Zustände abbilden. Damit könnte man z.B. die Zahlenwerte 0 bis 255, -127 bis +127 oder auch Buchstaben aus dem ASCII-Zeichenvorrat darstellen.
  - **Word:** Kommt aus dem Englischen und damit ist auch das Wort gemeint. Ein Wort besteht aus mindestens 2 Buchstaben. Daher besteht ein Wort aus 2 Bytes, was insgesamt 16 Bits wären.
  - **DWord:** Es gibt noch das Doppelwort, das aus 2 Wörtern besteht. Das sind 32 Bits oder 4 Bytes.

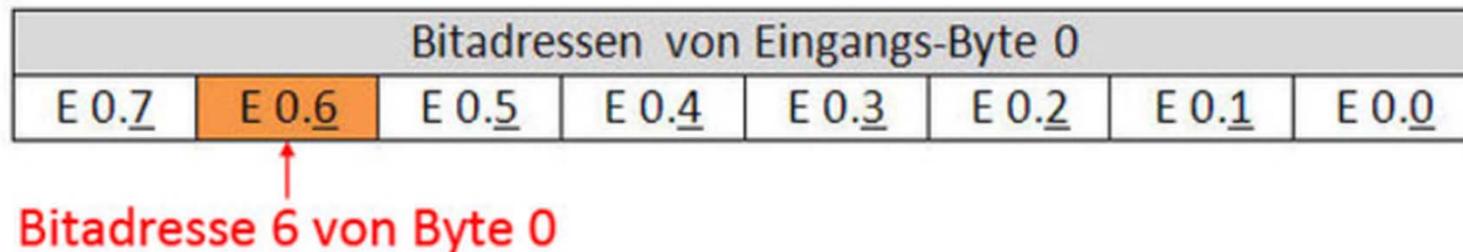
## Bit und Byte (3)

- Auch bei Speicherprogrammierbaren Steuerungen bzw. bei der SPS-Programmierung werden die Begriffe Bit, Byte, Wort und Doppelwort verwendet.
- Wenn man also meint, die Zahl ist ein Doppelwort lang, dann meint man eine Bitlänge von 32 Bits.
  - Dabei gibt es Bitadressen, Byteadressen, Wort- und Doppelwortadressen.
  - Die Begriffe werden auch häufig in Verbindung mit der Adressierung benutzt. Möchte man z.B. einen Taster abfragen, schließt man den Taster an die SPS an. Dabei bekommt der Taster eine eindeutige Adresse. Über diese Adresse kann der Taster dann eindeutig identifiziert und z.B. im SPS-Programm abgefragt werden.

# Bitadressen

- Ein Byte besteht aus 8 Bits. Für jedes Bit wird dabei eine Bitadresse zugeordnet. So wird jedes Bit einer eindeutigen Adresse zugeordnet, der darüber auch direkt angesprochen werden kann. Die Adressierung eines Bytes fängt dabei mit der ersten Bitstelle auf der rechten Seite an und erhöht sich mit jeder Bitstelle nach links um 1. In Zahlen ausgedrückt bedeutet das, die Bitstelle ganz rechts erhält die Bitadresse 0 und die Bitstelle ganz links die Bitadresse 7.

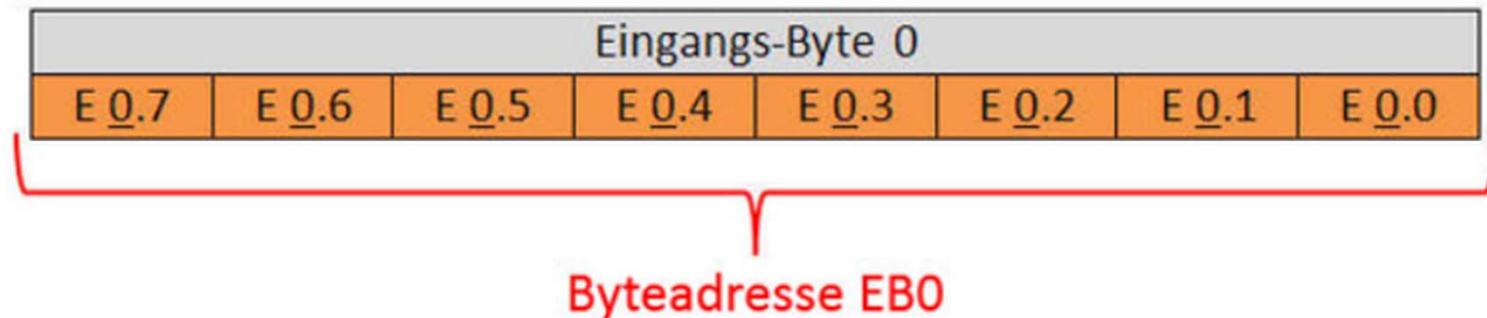
Darstellung der Bitadressen 0 bis 7:



# Byteadressen

- Jedes einzelne Byte wiederum erhält ebenfalls eine Nummer, die Byteadresse. Außerdem wird noch der Operand bezeichnet. EB1 steht für Eingangsbyte 1 oder AB2 für Ausgangsbyte 2. Ein einzelnes Bit wird eindeutig durch die Kombination von Bit- und Byteadresse adressiert, z.B. E0.1 für Bitadresse 1 von Eingangsbyte 0. Hierbei wird die Bitadresse durch einen Punkt von der Byteadresse getrennt. Links vom Punkt steht die Byteadresse und rechts vom Punkt die Bitadresse. Bei dem Ausgang A2.3 handelt es sich also um das 4. Bit (die Bitnummern fangen bei 0 an) im zweiten Byte einer Ausgangsbaugruppe.

Darstellung einer Byteadresse:



# Wortadressen

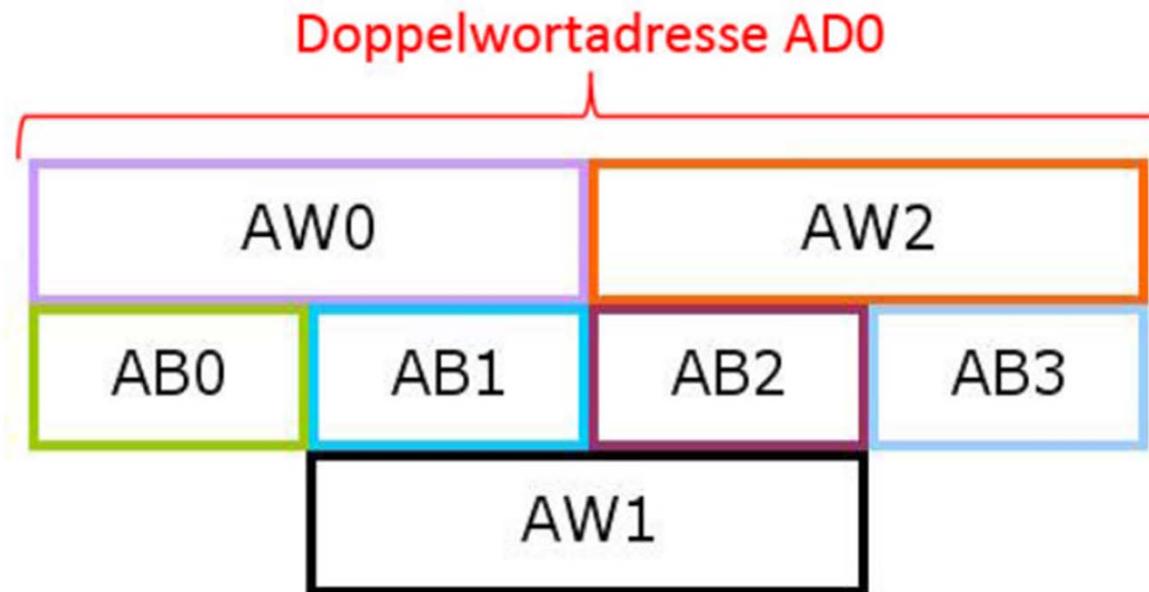
- Ein Wort besteht aus 2 Bytes. Wenn 2 Bytes zusammengefasst adressiert werden sollen, benutzt man Wortadressen, die nummeriert werden. Die Wortadresse bei der Verwendung von Eingangswörtern EW, Ausgangswörtern AW und Merkerwörtern MW ist immer die kleinere Byteadresse der beiden im Wort befindlichen Bytes.

Darstellung von Wortadressen:



# Doppelwortadressen

- Ein Doppelwort besteht aus 2 Wörtern, die sich jeweils wiederum aus 2 Bytes zusammensetzen (32 Bits). Auch hier gilt, dass man bei der Verwendung von Doppelwörtern bei Eingängen (ED), Ausgängen (AD) und Merkern (MD) immer die kleinere Byteadresse nummeriert. Wenn man also AD0 nummeriert, dann erstreckt sich das Doppelwort über die Ausgangsbytes AB0, AB1, AB2 und AB3.



# Adressierungsregeln

- Hinweis: Wenn man ein Byte, Wort oder Doppelwort bereits adressiert, dann kann man innerhalb des adressierten Bereichs ein Bit, Byte, Wort oder Doppelwort nicht noch einmal adressieren, da hier eine Überschneidung stattfinden würde.
- Würde man z.B. das Ausgangsdoppelwort AD0 adressieren, dann belegt das Doppelwort die Ausgangsbytes 0, 1, 2 und 3. Möchte man ein weiteres Ausgangsdoppelwort adressieren, dann kann man nicht einfach AD1 nehmen. Das nächste Ausgangsdoppelwort, das für die Adressierung frei ist, wäre in diesem Fall AD4.
- Dies gilt für alle Adressierungen bei Eingängen, Ausgängen und Merkern ebenso wie bei der Adressvergabe bei Ganzzahlen (INT, DINT) und Gleitkommazahlen (REAL).

# Datentypen bei der Programmierung (1)

- Bei der Erstellung von SPS-Programmen hat man es mit verschiedenen Datentypen zu tun. Mit den verschiedenen Datentypen werden unterschiedliche Speicherinhalte voneinander abgegrenzt, für die unterschiedliche Speicherbereiche reserviert werden.
  - Beispiel für das Signal 1: Datentyp Bool
  - Benötigter Speicherplatz: 1 Bit
  - Menge der darstellbaren Ziffern: 2 (0 und 1)
- Die Angabe eines Datentyps wird insbesondere bei der Variablendeklaration von Funktionen oder Funktionsbausteinen benötigt. Bei der Deklaration einer Variable "weiß" das Programm z.B. nicht, ob es sich um einen Booleschen Wert (1 Bit groß) handelt, ob es sich um eine Byteadresse (8 Bits) handelt usw. Durch die Angabe des Datentypen legt man im Grunde fest, in welcher Form die Daten bei einer Abfrage erwartet werden oder wie die Daten bei einer Ausgabe abgelegt werden sollen.

# Datentypen bei der Programmierung (2)

- Innerhalb eines SPS-Programms werden die Daten auch häufig mit einem bestimmten Datentypen erwartet. Wenn die SPS z.B. für die Durchführung eines Befehls für den Wert eine Bitlänge von 16 Bits erwartet, kann man nicht eine Bitlänge von 8 Bits übergeben. Durch den Datentypen werden außerdem viele Wertebereiche begrenzt.  
Beispiel:
  - Datentyp für den Zeitwert in S7: S5TIME
  - Datenlänge: 16 Bits, wovon die letzten 2 Bits ignoriert werden
  - Darstellbare Wertebereiche: 10 Millisekunden bis 9990 Sekunden

# Datentypen nach IEC 61131-3

- In der **Programmiersnorm 61131-3** wurden für Steuerungsaufgaben verschiedene Datentypen festgelegt. Dabei wurden auch sogenannte **Schlüsselwörter** für Datentypen festgelegt, die nicht anderweitig verwendet werden dürfen.
- Die Datentypen werden in 2 Kategorien eingeteilt. Diese sind:
  - **Elementare Datentypen:** Diese haben eine maximale Länge von 32 Bits. Damit werden, wie der Name schon andeutet, elementare Daten beschrieben, z.B. eine Ganzzahl mit einer Länge von 16 Bits.
  - **Zusammengesetzte Datentypen:** Diese Datentypen können aus mehreren elementaren Datentypen bestehen. Wenn man z.B. einen Datentypen mit einer Ganzzahl und einer Zeichenkette benutzt, so hat man es hier mit einem zusammengesetzten Datentypen zu tun. Daher können diese auch größer als 32 Bits sein. Diese sind auch vordefiniert. Das bedeutet, der Programmierer hat nicht die Auswahl festzulegen, um welchen Datentypen es sich dabei handelt, wenn er in der Variablendeklaration zusammengesetzte Datentypen benutzt.

# Elementare Datentypen in Step7

<b>Name</b>	<b>Datentyp</b>	<b>Größe</b>	<b>Wertebereich</b>
<b>BOOL</b>	Boolescher Wert	1Bit	False (logisch 0), True (logisch 1)
<u><b>BYTE</b></u>	Dual- und Hexadezimalzahlen mit 8 Bits	8 Bits	Dualzahlen 00000000 - 11111111 Hexadezimal B#16#0 - B#16#FF
<u><b>WORD</b></u>	Dual- und Hexadezimalzahlen mit 16 Bits	16 Bits	Dualzahlen von 0 bis 1111 1111 1111 1111 Hexadezimal W#16#0 - W#16#FFFF BCD-Zahlen von 0 bis 999 Dezimal ohne Vorz. B#(0,0) bis B#(255,255)
<u><b>DWORD</b></u>	Dual- und Hexadezimalzahlen mit 32 Bits	32 Bits	Dualzahlen von 0 bis 1111 1111 1111 1111 1111 1111 1111 1111 Hexadezimal von DW#16#0000 0000 bis DW#16#FFFF FFFF Dezimal ohne Vorz. von B#(0,0,0,0) bis B#(255,255,255,255)



# Zahlenformate - Datentypen

- In einer SPS werden viele Zahlen verarbeitet. Dabei gibt es unterschiedliche Zahlenformate, für die es unterschiedliche Datentypen gibt.
- Grundsätzlich wird unterschieden zwischen:
  - **Ganzzahlen**
  - **Gleitpunktzahlen**
- Eine häufige Fehlerquelle ist, dass man versucht, Zahlen mit unterschiedlichen Datentypen zu addieren. Hierbei kommt es dann häufig zu Problemen bzw. es ist erst gar nicht möglich. Denn Zahlen mit unterschiedlichen Datentypen kann man nicht addieren. In solchen Fällen muss man zuerst die Zahl in einen identischen Datentypen umwandeln.

# Ganzzahlen

- Mit Ganzzahlen sind ganze Zahlen, ohne Nachkommastelle gemeint, also z.B. die Dezimalzahl  $875_{10}$ . Der darstellbare Zahlenraum hängt von der Anzahl der zur Verfügung stehenden Bits ab. Mit nur einem Bit kann man z.B. nur die Ziffern 0 und 1 darstellen.
- Bei Speicherprogrammierbaren Steuerungen werden Ganzzahlen mit unterschiedlichen Datentypen verwendet. Diese sind:
  - **Integer-Zahlen, Kurzform INT:** Ganzzahlen vom Typ Integer haben eine Bitlänge von 16 (Bit 0 bis Bit 15). Die belegte Bitlänge wird auch als ein Wort oder 2 Bytes bezeichnet. Als Vorzeichenbit MSB, das für die Unterscheidung zwischen negativen und positiven Zahlen benötigt wird, wird dabei Bit 15 benutzt.
  - **Doppelinteger-Zahlen, Kurzform DINT:** Ganzzahlen vom Typ Doppelinteger haben eine Bitlänge von 32 (Bit 0 bis Bit 31). Die Bitlänge wird auch als Doppelwort oder 4 Bytes genannt. Als Vorzeichenbit wird bei DINT-Zahlen Bit 31 benutzt.

# Ganzzahlen - MSB

- Zur Unterscheidung von positiven und negativen Zahlen wird ein Bit des zur Verfügung stehenden Speicherplatzes benötigt.
- Dies ist das **Vorzeichenbit MSB - Most Significant Bit**

- **INTEGER:**

Bei Ganzzahlen vom **Typ INT** wird **Bit 15 als Vorzeichenbit** benutzt. Dementsprechend bleiben 15 Bits (Bit 0 bis Bit 14) für die Darstellung der Zahlenwerte übrig. Da sowohl positive als auch negative Zahlen dargestellt werden können, erstreckt sich der darstellbare **Zahlenbereich von  $-32768_{10}$  ( $-2^{15}$ ) bis  $+32767_{10}$  ( $2^{15}$ ).**

- **DOPPELINTEGER:**

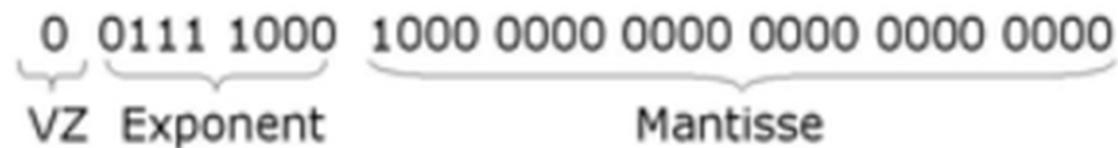
Bei Ganzzahlen vom **Typ DINT** wird **Bit 31 als Vorzeichenbit** benutzt. Für die Darstellung der Zahlenwerte bleiben somit 31 Bits (Bit 0 bis Bit 30) übrig. Damit kann man einen **Zahlenbereich von  $-2147483648_{10}$  ( $-2^{31}$ ) bis  $+2147483647_{10}$  ( $2^{31}$ )** darstellen.

# Gleitpunktzahlen (1)

- Mit Gleitpunktzahlen sind Zahlen gemeint, die zwei Nachkommastellen haben, also z.B. die Dezimalzahl  $8,75_{10}$ .
- Bei Speicherprogrammierbaren Steuerungen werden Gleitpunktzahlen mit folgendem Datentypen verwendet:
  - **REAL:** Gleitpunktzahlen vom Datentyp REAL haben eine Bitlänge von 32 (Bit 0 bis Bit 31). Auch hierbei wird das Bit mit dem höchsten Stellenwert als Vorzeichenbit MSB (Bit 31) benutzt. Mit dem Datentyp REAL können einerseits sehr große Zahlen darstellen. Andererseits kann man damit auch sehr kleine Zahlen sehr genau darstellen, da man auch Nachkommastellen abbilden kann.

## Gleitpunktzahlen (2)

- Für die Darstellung von Gleitpunktzahlen vom Datentyp REAL werden die zur Verfügung stehenden 32 Bits in 3 Bereiche aufgeteilt. Diese werden bei der Auswertung einzeln berücksichtigt.
  - Bit 31 wird als Vorzeichenbit MSB benutzt.
  - Bit 30 ( $2^7$ ) bis Bit 23 ( $2^0$ ), (8 Bits) wird als Exponent benutzt. Der Exponent gibt die Potenz der Basis an, die man mit der Zahl multiplizieren muss. Bei Dualzahlen ist die Basis die 2. Bei Speicherprogrammierbaren Steuerungen wird der Exponent als 8 Bit Dualzahl ohne Vorzeichen dargestellt und ergibt sich aus der Subtraktion der Zahl  $127_{10}$ . Damit wird die Kommaposition abgebildet.
  - Bit 22 ( $2^{-1}$ ) bis Bit 0 ( $2^{-23}$ ) wird als Mantisse benutzt. Hier werden die Zahlenwerte rechts vom Komma abgelegt. Somit entscheidet die Mantisse darüber, wie genau die Gleitpunktzahlen dargestellt werden können. Der Mantisse wird eine 1 vorangestellt, die nicht darstellbar ist und auch nicht gespeichert wird. Mit dem Zahlenwert der Mantisse ergibt das einen Zahlenwert zwischen  $1_{10}$  und  $2_{10}$ .

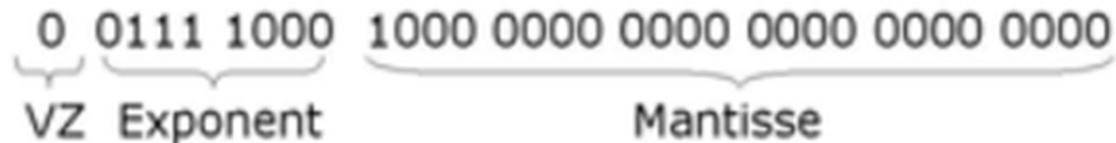


# Übung: Umrechnen einer Gleitkommazahl

- Die Dezimalzahl 18,4 soll in die binäre Gleitkommadarstellung umgerechnet werden. Der Ablauf besteht in der Regel aus 6 Schritten.
- Prinzipiell besteht der Ablauf daraus, die Zahl umzurechnen, zu normalisieren, den neuen Exponenten zu ermitteln, das Vorzeichen zu bilden und anschließend die Werte Vorzeichen, Charakteristik (Exponent) und Mantisse zusammen zu setzen.
  1. Vorkommazahl umrechnen
  2. Nachkommazahl umrechnen
  3. Normieren bzw. Normalisieren (Mantisse ermitteln)
  4. Exponent umrechnen (Charakteristik ermitteln)
  5. Vorzeichen bestimmen
  6. Gleitkommazahl bilden (mit einfacher Genauigkeit)

# Übung/Beispiel Gleitkommazahlen

- Folgendes Bitmuster ist gegeben:



- Vorzeichenbit MSB: 0 Die Zahl ist positiv.
  - Exponent: 0111 1000 Bedeutet =  $120_{10}$ .
  - Mantisse: Bit 22 ist "1", entspricht  $2^{-1}$  also 0,5. Die übrigen Ziffern sind 0 und werden daher nicht betrachtet.
- Formel für die Umrechnung:
    - $1 + \text{Mantisse} \cdot (2^{(\text{Exp}-127)}) = 1,5 \cdot (2^{(120-127)}) = 1,5 \cdot (2^{-7}) = 1,5 \cdot 0,0078125 = 0,01171875$ .
    - Zum Schluss noch das Vorzeichen setzen: + 0,01171875

# Lösung: Wandlung Gleitkommazahlen

- Gegeben Dezimalzahl: 18,4

## 1. Vorkommazahl umrechnen:

- Im ersten Schritt wird für die Vorkommazahl 18 die Dualzahl ermittelt. Hier wird das Teiler- bzw. Divisions-Verfahren angewendet.

					Rest	
18	:	2	=	9	→	0 (letztes Bit/Least Significant Bit)
9	:	2	=	4,5	→	1
4	:	2	=	2	→	0
2	:	2	=	1	→	0
1	:	2	=	0,5	→	1 (erstes Bit/Most Significant Bit)

---

$10010_2$

# Lösung: Wandlung Dual- aus dem Dezimalsystem

## 2. Nachkommazahl umrechnen:

Im zweiten Schritt wird für die Nachkommazahl 0,4 die Dualzahl ermittelt. Dabei wird die Multiplikationsmethode angewendet.

Als Zwischenergebnis entsteht eine duale gebrochene Zahl.

						Rest
0,4	*	2	=	0,8	→	0
0,8	*	2	=	1,6	→	1
0,6	*	2	=	1,2	→	1
0,2	*	2	=	0,4	→	0
0,4	*	2	=	0,8	→	0
0,8	*	2	=	1,6	→	1
0,6				1,2	→	1
0,2				0,4	→	0
...						...
...						...
0,6				1,2	→	1

---

$0,01100110011..._2$

# Lösung: Wandlung Dual- aus dem Dezimalsystem

- Ergebnis aus Schritt 1 und 2:
  - $18,4 = 10010,011001100110011\dots$

## 3. Normieren bzw. Normalisieren (Mantisse ermitteln)

Bei der Normalisierung verschiebt man das Komma so, das man eine normalisierte Zahl erhält. Zum Beispiel  $1,0101$  (2) oder  $0,123$  (10). Dabei greift man auf die Exponentialdarstellung zurück, damit die Zahl ihren Wert behält.

$$\begin{aligned} &10010,01100110011\dots * 2^0 \\ &1,001001100110011\dots * 2^4 \text{ (Normalisierung)} \end{aligned}$$

Bei der Normalisierung geht es darum, dass man nur die Nachkommastellen speichern möchte. Bei der binären Darstellung von Zahlen steht vorne immer eine Eins (1). Diese Vorkomma-Eins kann man beim Speichern bzw. bei der Darstellung weglassen (hidden bit), weil hier immer eine Eins steht. Dafür hat man hinten eine Stelle mehr für die Genauigkeit.

# Lösung: Wandlung Dual- aus dem Dezimalsystem

## 4. Exponenten umrechnen (Charakteristik ermitteln)

- Der Exponent wird auch als Charakteristik bezeichnet, weil sein Wertebereich verschoben ist (Bias). Das macht man deshalb, damit der Exponent immer positiv oder mindestens Null ist. Dadurch spart man sich ein Bit für Darstellung des Vorzeichens des Exponenten. Die Bias-Darstellung erleichtert auch den Größenvergleich. Nachteil, es gibt eine positive Null (+0) und eine negative Null (-0).
- Der Bias oder Exzess hängt von der gewählten Genauigkeit (Anzahl der Bits) ab.
  - Einfach Genauigkeit (32 Bit) bedeutet einen Bias von 127.
  - Doppelte Genauigkeit (64 Bit) bedeutet einen Bias von 1023.
- Da wir mit einer einfachen Genauigkeit (32 Bit) arbeiten entspricht das einem Bias von 127.

# Lösung: Wandlung Dual- aus dem Dezimalsystem

## 4. Exponenten umrechnen (Charakteristik ermitteln)

- Charakteristik (neuer Exponent) = Exponent + Bias = 4 + 127 = 131

				Rest	
131	:	2	=	65,5	→ 1 (letztes Bit/Least Significant Bit)
65	:	2	=	32,5	→ 1
32	:	2	=	16	→ 0
16	:	2	=	8	→ 0
8	:	2	=	4	→ 0
4	:	2	=	2	→ 0
2	:	2	=	1	→ 0
1	:	2	=	0,5	→ 1 (erstes Bit/Most Significant Bit)

---

$1000011_2$

# Lösung: Wandlung Dual- aus dem Dezimalsystem

## 5. Vorzeichen bestimmen

- Positiv = 0
- Negativ = 1

## 6. Gleitkommazahl bilden

Aus den vorherigen Schritten:

2.  $10010,01100110011... * 2^0$
3.  $1,001001100110011... * 2^4$  (Normalisierung)
4.  $10000011$

Ergibt sich damit, die normalisierte Dualzahl

$1 V$	$8 E$	$23 M$
$0$	$10000011$	$00100110011001100110011$

# Definitionen

Name	Datentyp	Größe	Wertebereich
CHAR	ASCII-Zeichen	8 Bit	Buchstaben, z.B. A, B, C, D usw.
<u>INT</u>	Ganze Zahlen mit Vorzeichen <u>(Festpunktzahlen)</u>	16 Bit	-32768 bis +32768
<u>DINT</u>	Ganze Zahlen mit Vorzeichen <u>(Festpunktzahlen)</u>	32 Bit	-2147483648 bis +2147483648
<u>REAL</u>	Reelle Zahlen <u>(IEEE Gleitpunktzahlen)</u>	32 Bit	Dezimalzahl mit Punkt (156,235) Exponentialdarstellung (1.568 E+04)
<u>S5TIME</u>	Zeitdauer im S5T#-Format Step7 Zeit in Schritten von 10ms	16 Bit	S5T#0H_0M_0S_10MS bis S5T#2H_46M_30S_0MS und S5T#0H_0M_0S_0MS
TIME	Zeitdauer im IEC-Format IEC-Zeit in Schritten von 1ms, Ganzzahl mit Vorzeichen	32 Bit	-T#24D_20H_31M_23S_648MS bis T#24D_20H_31M_23S_647MS
TIME_OF_DAY	Uhrzeit (Tageszeit) in Schritten von 1ms	32 Bit	TOD#0:0:0.0 bis TOD#23:59:59.999 z.B. TIME_OF_DAY#23:59:59,9
DATE	IEC-Datum in Schritten von 1 Tag	16 Bit	D#1990-1-1 bis D#2168-12-31 z.B. DATE#2006-05-10



# Zusammengesetzte Datentypen

- In Step7 benutzt man neben den elementaren Datentypen auch noch die zusammengesetzten Datentypen. Damit werden Datengruppen definiert, die größer als 32 Bit sind. In Step7 kann man folgende zusammengesetzte Datentypen definieren:
  - DATE\_AND\_TIME, DT
  - STRING
  - ARRAY (Feld)
  - STRUCT (Struktur)
  - UDT (User Defined Data Typ, Anwenderdefinierte Datentypen)
  - FB (Funktionsbaustein)
  - SFB (Systemfunktionsbaustein)

# Zusammengesetzte Datentypen Übersicht (1)

Schlüsselwort	Datentyp	Beschreibung
DATE_AND_TIME	DATE_AND_TIME definiert einen Bereich mit 64 Bits (8 Bytes) und wird in einem binärcodierten Dezimalformat gespeichert.	DT#1990-1-1-0:0:0.0 bis DT#2089-12-31-23:59:59.999 (erlaubter Bereich für die Eingabe von Datum und Zeit)
ARRAY	Mit diesem Datentyp definiert man ein Feld bzw. eine Gruppierung von Komponenten gleichen Datentyps (entweder elementar oder zusammengesetzt). Man kann maximal 6 Dimensionen in einem Feld definieren.	<p>Messwerte: ARRAY[1...3] OF INT; Jede Feldkomponente wird über den Feldindex angesprochen:</p> <p>L Messwerte [Index] T Messwerte [Index]</p> <p>Der Index kann ein beliebiger ganzzahliger Wert von -32768 bis +32767 sein und kann zur Laufzeit des Programms nicht verändert werden.</p> <p>Über SFC20 können ARRAY-Variablen kopiert werden.</p>

## Zusammengesetzte Datentypen Übersicht (2)

Schlüsselwort	Datentyp	Beschreibung
STRUCT	Mit dem Datentyp STRUCT kann man eine Gruppierung von beliebig kombinierten Datentypen bzw. Komponenten definieren, z. B. ein Feld aus Strukturen oder eine Struktur aus Strukturen und Feldern.	<p>Messwerte: STRUCT            Temperatur: INT;            Geschwindigkeit: REAL;            END_STRUCT;</p> <p>In AWL sind die Komponenten einzeln ansprechbar, z.B:</p> <p>T Messwerte. Geschwindigkeit</p> <p>Über SFC20 können STRUCT-Variablen kopiert werden.</p>
FB, SFB	Mit FB oder SFB wird die Struktur des zugeordneten Instanzdatenbausteins bestimmt.	Die Bestimmung der Struktur ermöglicht die Übergabe von Instanzdaten für mehrere FB-Aufrufe innerhalb eines Instanzdatenbausteins.

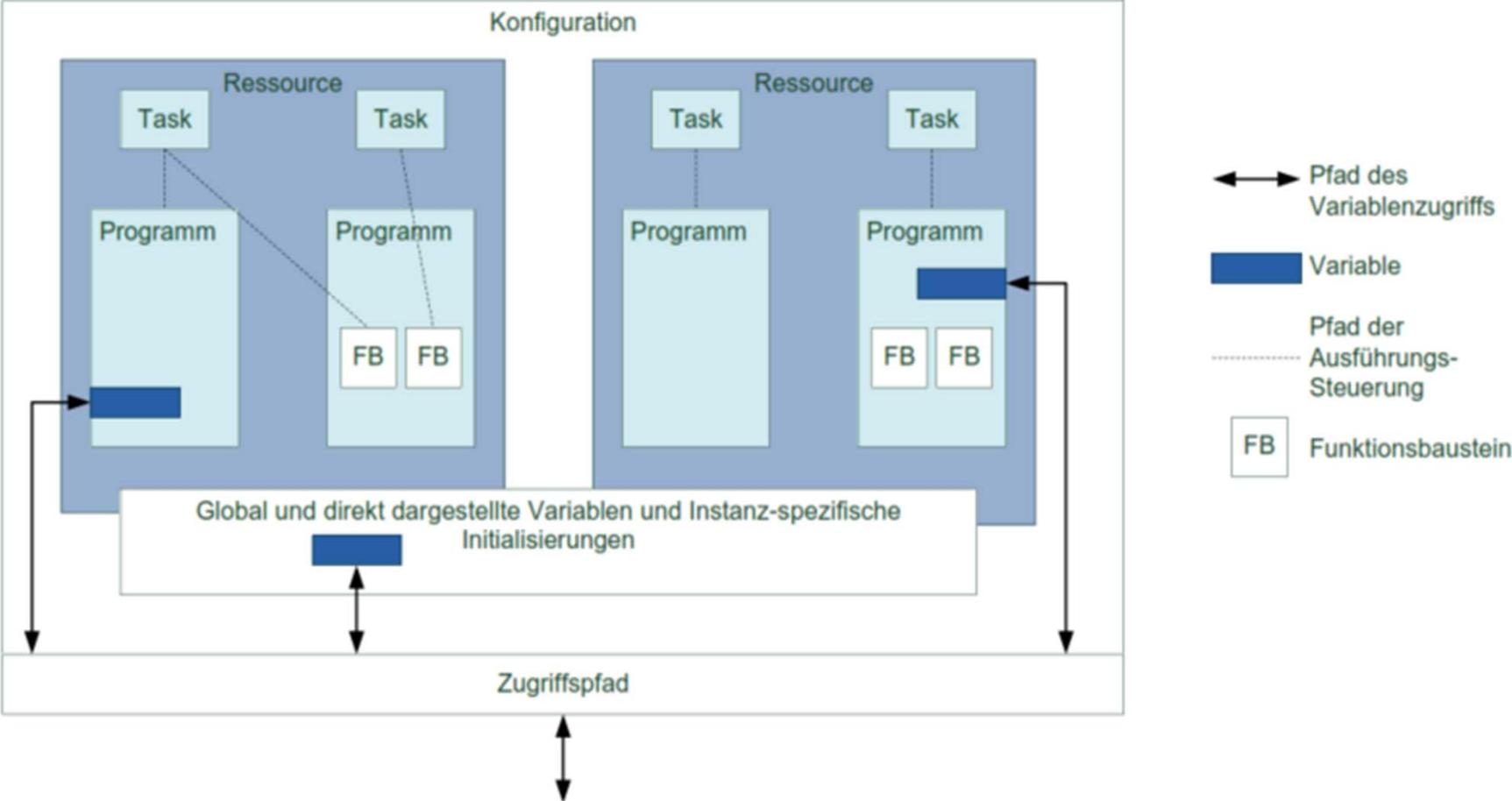
# Zusammengesetzte Datentypen Übersicht (3)

Schlüsselwort	Datentyp	Beschreibung
FB, SFB	Mit FB oder SFB wird die Struktur des zugeordneten Instanzdatenbausteins bestimmt.	Die Bestimmung der Struktur ermöglicht die Übergabe von Instanzdaten für mehrere FB-Aufrufe innerhalb eines Instanzdatenbausteins.
UDT	<p>User Defined Data Typ:</p> <p>Durch die Verwendung von UDTs ist es möglich große Datenmengen zu strukturieren. Dies vereinfacht das Eingeben von Datentypen bei der Erzeugung von Datenbausteinen und bei der Deklaration von Variablen. In Step7 können elementare und zusammengesetzte Datentypen verknüpft werden. Dadurch entstehen UDTs, die einen eigenen Namen haben und dadurch mehrfach verwendbar sind.</p>	Ein UDT ist in allen Bausteinen global gültig. Ist vom Aufbau her wie ein STRUCT.

# SPS-Softwaremodell

- Konfiguration:
  - SPS-System, enthält unter Umständen mehrere Ressourcen (entspricht SPS)
- Ressource:
  - Signalverarbeitung, incl. Sensor/Aktorschnittstellen (entspricht CPU+IO-Bus)
- Programm:
  - Wird zur Steuerung von Task ausgeführt
  - Kann mehrere Funktionsbausteine oder andere Elemente enthalten

# SPS-Softwaremodell

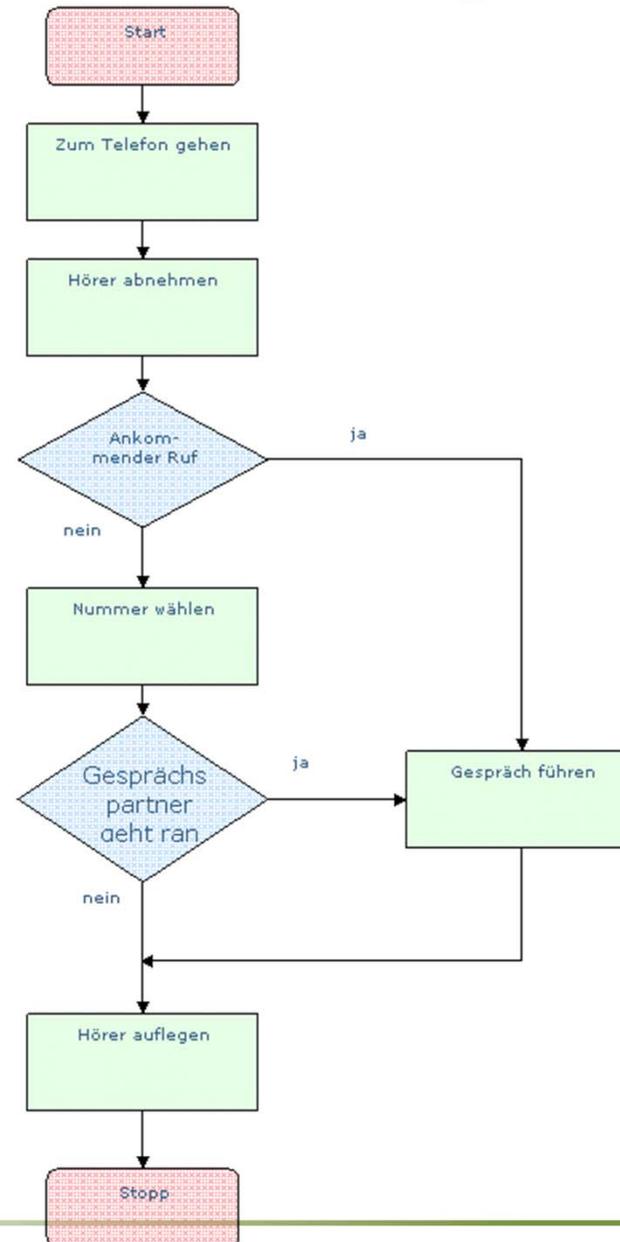


# Programmorganisationseinheit (POE)

- Programm (PRG): Hauptprogramm
  - Zuordnung der Peripherie
  - Globale und lokale Variablen
  - Kann FB und FC aufrufen
- Funktionsbaustein (FB): Baustein
  - Eingangs- und Ausgangsvariablen
  - Statische Variablen (Gedächtnis)
  - Kann FB und FC aufrufen
- Funktion (FC): Unterprogramm
  - Eingangs- und Ausgangsvariablen
  - Programm, welches bei jedem Aufruf abgearbeitet wird
  - Kann weitere FC Aufrufen

# Technische Problemlösungen mit Flussdiagrammen

- Flussdiagramme dienen der Visualisierung von allen möglichen Arten von Prozessen.



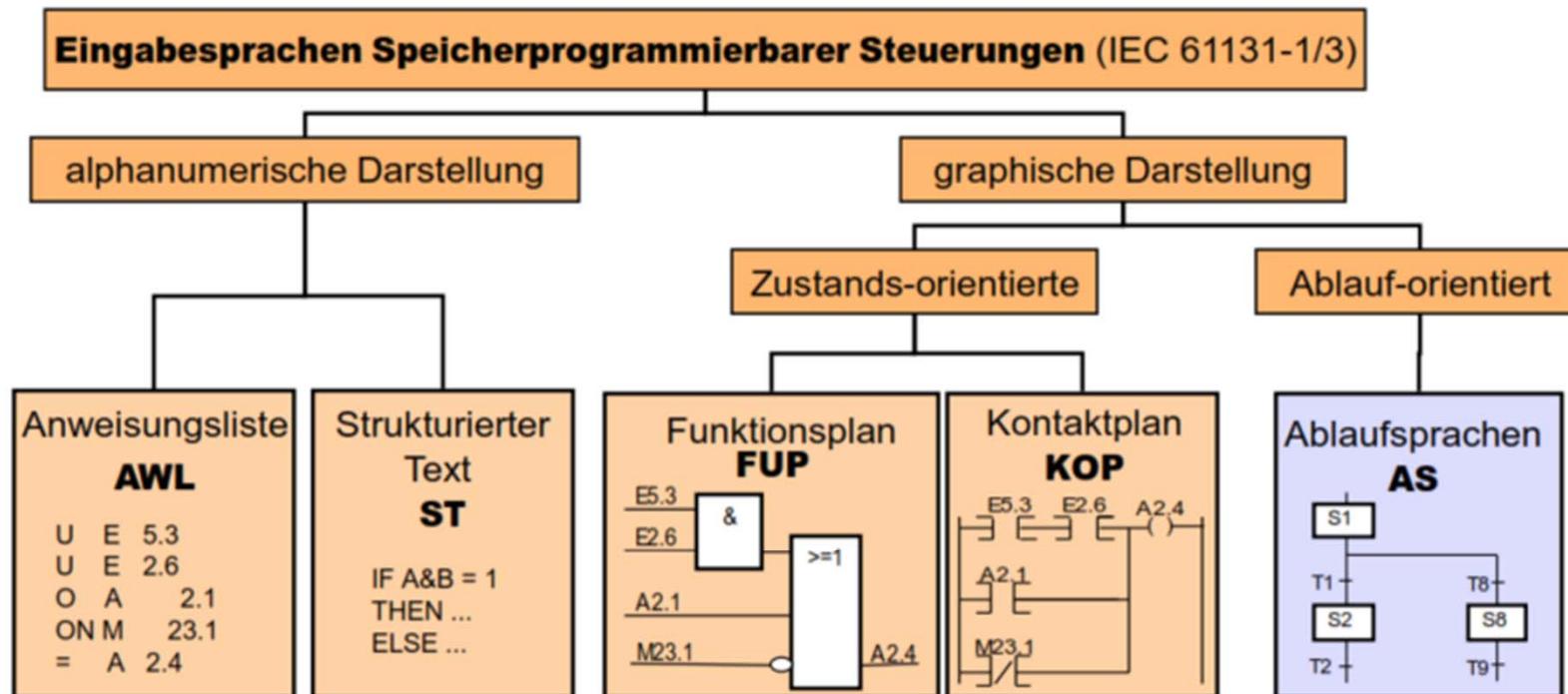
# 61131-3 Programmiersprachen

- Die Problemstellungen können aus der Textform oder dem Flussdiagramm in andere Darstellungsarten überführt werden.
- Für die Programmierung von Speicherprogrammierbaren Steuerungen existiert eine große Anzahl von Sprachen, da z.T. jeder Hersteller für seine Steuerung eine eigene Sprache entwirft.
- IEC 61131-3 spezifiziert 5 Programmiersprachen

Textbasiert	Grafisch
Anweisungsliste (AWL)	Kontaktplan (KOP)
Strukturierter Text (ST)	Funktionsbausteinsprache (FBS)
Ablaufsprache (AS)	

# Eingabesprachen Speicherprogrammierter Steuerungen (IEC 61121-1/3)

– IEC 61131 definiert grafische und textuelle Grundsprachen

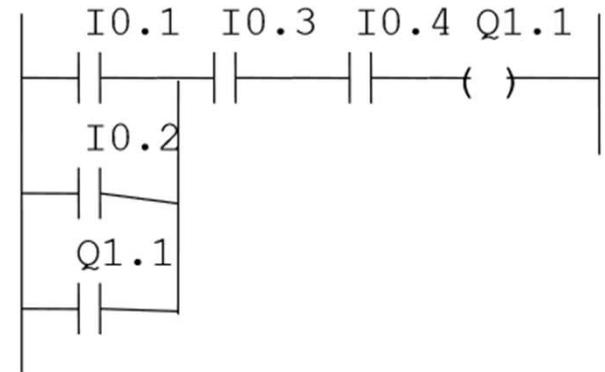
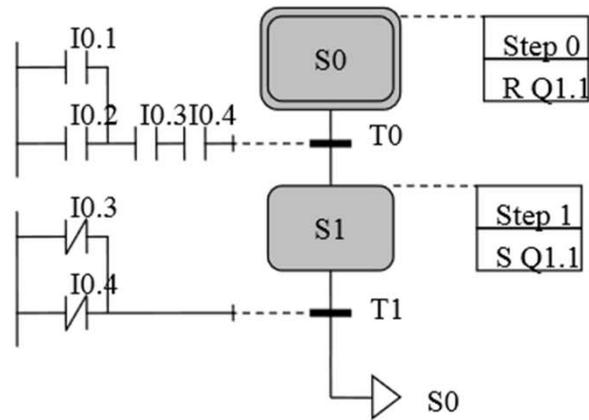


– IEC 61131 definiert keine Befehle!

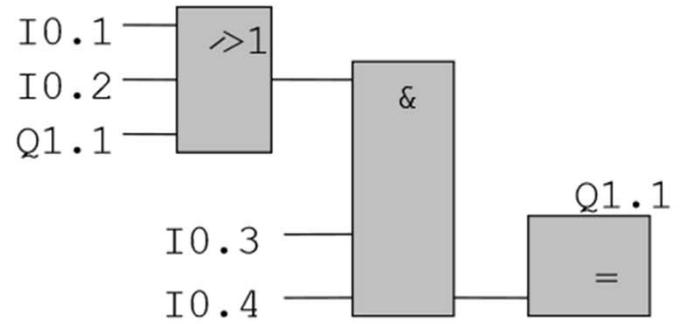
⇒ sowohl deutsche als auch englische Bezeichner für Operatoren

# Verschiedene Programmiersprachen

```
LD      %IX0.1
OR      %IX0.2
OR      %QX1.1
AND     %IX0.3
AND     %IX0.4
ST      %QX1.1
```



```
Q1.1 := (I0.1 OR I0.2 OR Q1.1)
        AND I0.3
        AND I0.4;
```



# Grundlage aller 5 Programmiersprachen (1)

- Basis sind die logischen Verknüpfungen
- Erweiterung um Möglichkeiten der Zeitverarbeitung
- Darstellungsart je nach Aufgabenstellung unterschiedlich geeignet
- Bestimmte Operationen sind nur in AWL programmierbar
  - (Bit-Schiebeoperationen)

# Grundlage aller 5 Programmiersprachen (2)

- Zeichensatz
  - Textelemente Spalten
  - Zusätzlich Kleinbuchstaben (aber nicht case sensitiv), # oder £, \$ oder ¤, | oder !
- Bezeichner:
  - Folge von Buchstaben, Ziffern und Unterstrich
  - Muss mit Buchstaben oder Unterstrich beginnen
  - Mindestens sechs Zeichen werden zur eindeutigen Unterscheidung genutzt
- Leerzeichen, Kommentare (\* \*), Numerische Literale, Zeichenfolgeliteral, Zeitdauer

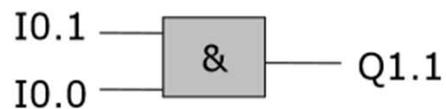
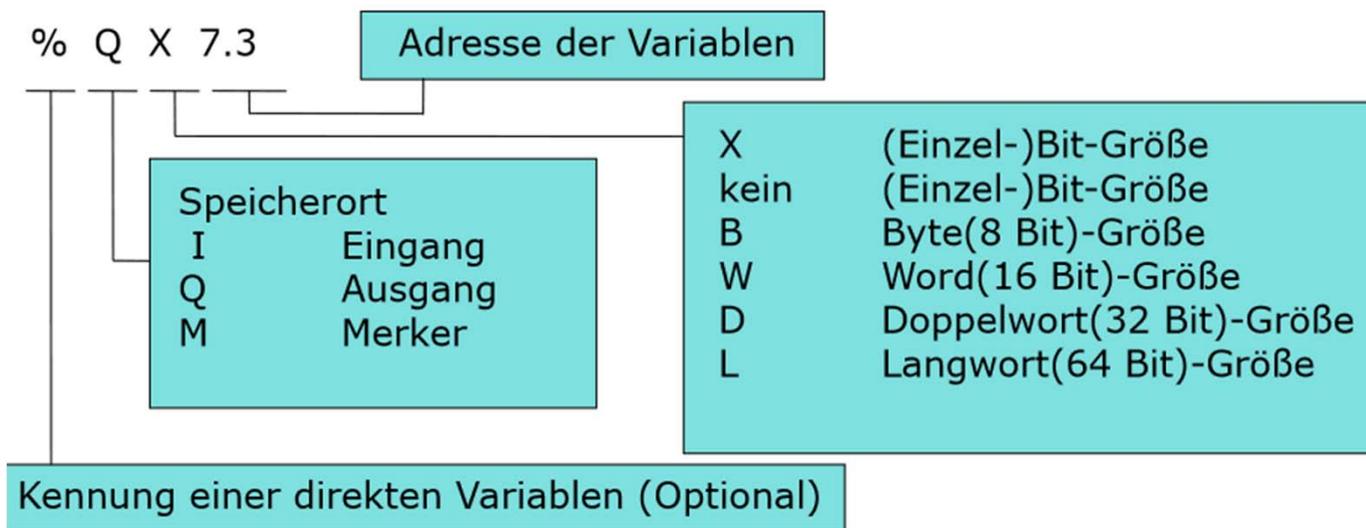
# Befehlssyntax

- Einheitliche Syntax der Befehlszeile

– Einheitlicher Aufbau einer Befehlszeile

`Marke (opt.): Operator Operand Kommentar (opt.)`

– Programmierung durch Verknüpfung von Signalen

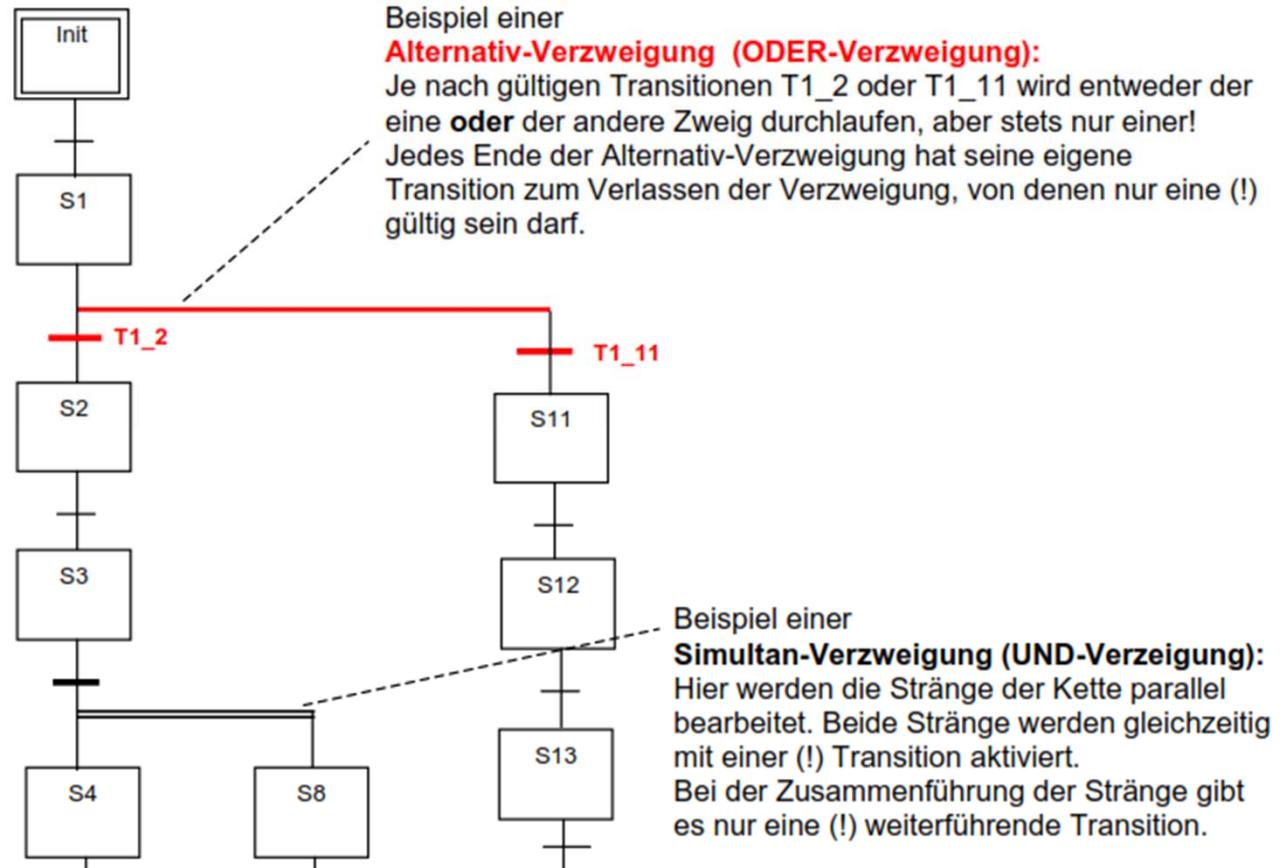


# Ablaufsprache AS

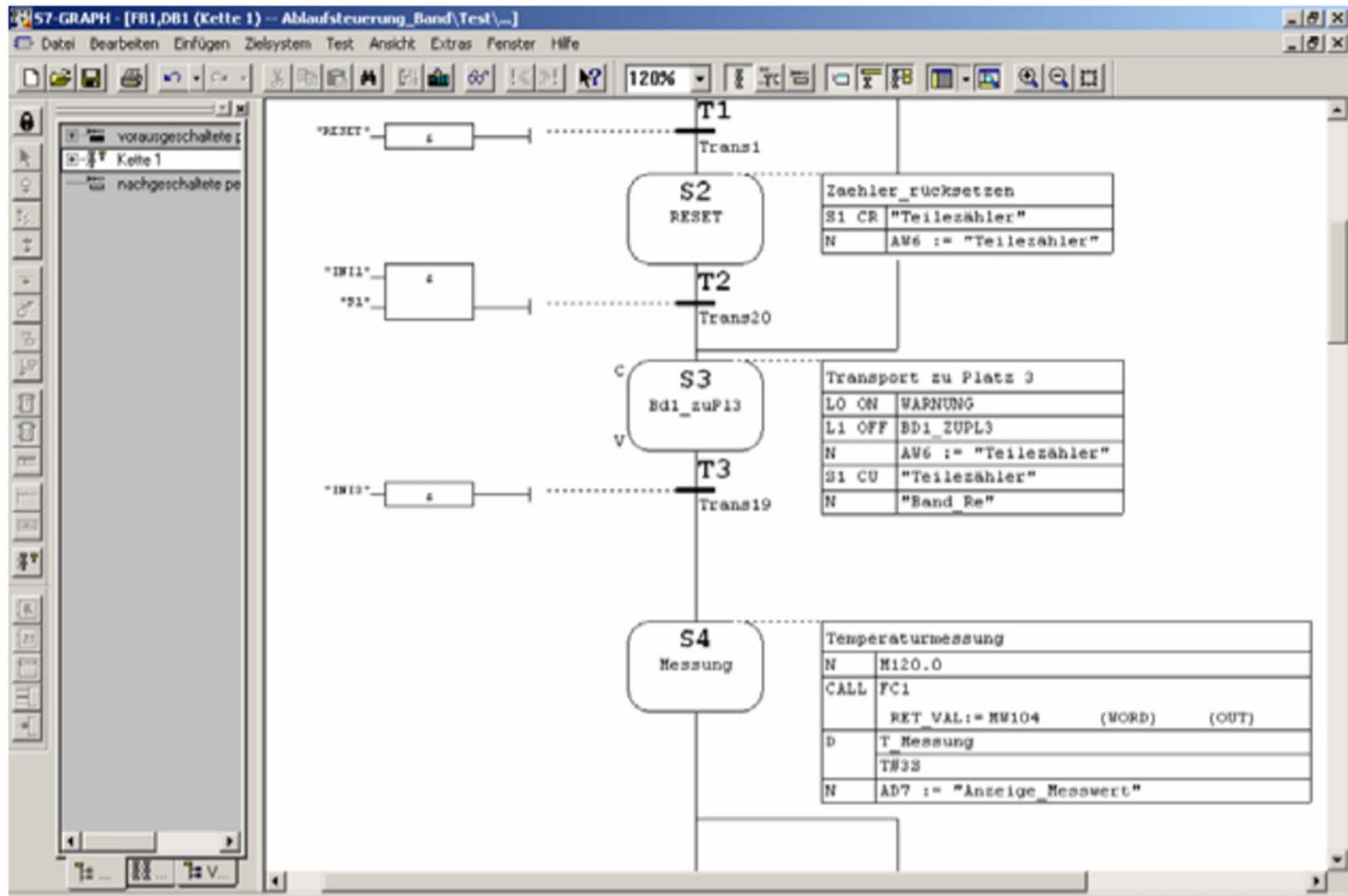
- Automatisierungsaufgaben lassen sich häufig als Sequenz einzelner Schritte darstellen
  - Ausführung der Schritte hängt vom Erreichen einer Bedingung ab
  - Verschiedene Schritte können parallel ausgeführt werden
- Die Ablaufsprache ist stark an das Flussdiagramm angelehnt
- Netze:
  - Übergang von einem Schritt zu einem oder mehreren nachfolgenden Schritten erfolgt durch eine Übergangsbedingung (Transition)
  - Aktionen und Transitionen werden in einer der vorhergehenden Sprachen spezifiziert

# Darstellung Ablaufsprache AS

- Auch in der Ablaufsprache werden die Elementargatter verwendet und dargestellt.



# Beispiel Ablaufsprache AS



# Anweisungsliste – AWL/IL (1)

- Universell einsetzbare Maschinensprache, ähnlich Assembler
  - weit verbreitet im SPS-Bereich
  - Zum Teil unterschiedliche Operatoren und Sprachumfänge
- Alle Funktionen einer SPS sind uneingeschränkt programmierbar.
- Jede andere Programmiersprache lässt sich in AWL umwandeln, aber nicht zwingend umgekehrt.

# Beispiel AWL (1)

```
FB1 : Title:
Comment:

Network 1: Title:
Comment:

//
//call of sfcl7 for first message
//
CALL "ALARM_SQ"           SFC17           -- Generate Block-Related Messages with Acknowledgment
  SIG      :=#SIG1
  ID       :=W#16#EEEE
  EV_ID    :=#EV_ID1
  SD       :=#SD1
  RET_VAL:=#RET_iVal
L   #RET_iVal
T   #RET_VAL1

//
//call of sfcl7 for second message
//
CALL "ALARM_SQ"           SFC17           -- Generate Block-Related Messages with Acknowledgment
  SIG      :=#SIG2
  ID       :=W#16#EEEE
  EV_ID    :=#EV_ID2
  SD       :=#SD2
  RET_VAL:=#RET_iVal
L   #RET_iVal
T   #RET_VAL2
```

## Beispiel AWL (2)

```
LD      %IX0.1 (* Lade Eingang 0.1 in Akku *)
OR      %IX0.2 (* Akku=Akku oder Eingang 0.2 *)
OR      %QX1.1 (* Akku=Akku oder Ausgang 1.1 *)
AND     %IX0.3 (* Akku=Akku und Eingang 0.3 *)
AND     %IX0.4 (* Akku=Akku und Eingang 0.4 *)
ST      %QX1.1 (* Ausgang 1.1=Akku *)
```

# Kontaktplan - KOP

- Funktionen werden durch Schaltsymbole aus der Elektrotechnik dargestellt:

- Schließer



**Schließer**

- Öffner



**Öffner**

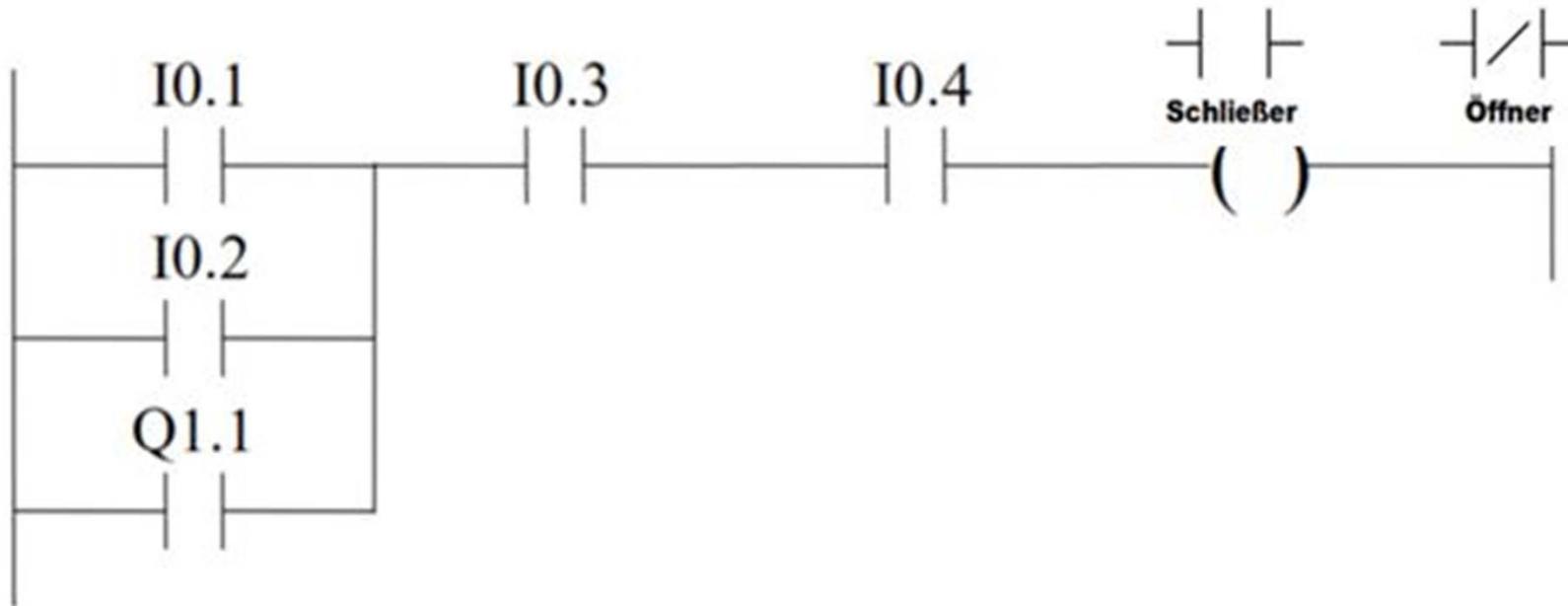
- Usw. werden ähnliche Schaltbilder zu Netzwerken zusammengefügt

- Beschränkt sich im Wesentlichen auf Boolesche Signale

- Reihenschaltung: UND

- Parallelschaltung: ODER

# Beispiel KOP



Eingang:

Ausgang:

Negation:

Schaltkontakt

Relaisspule

-- | | --

-- ( ) --

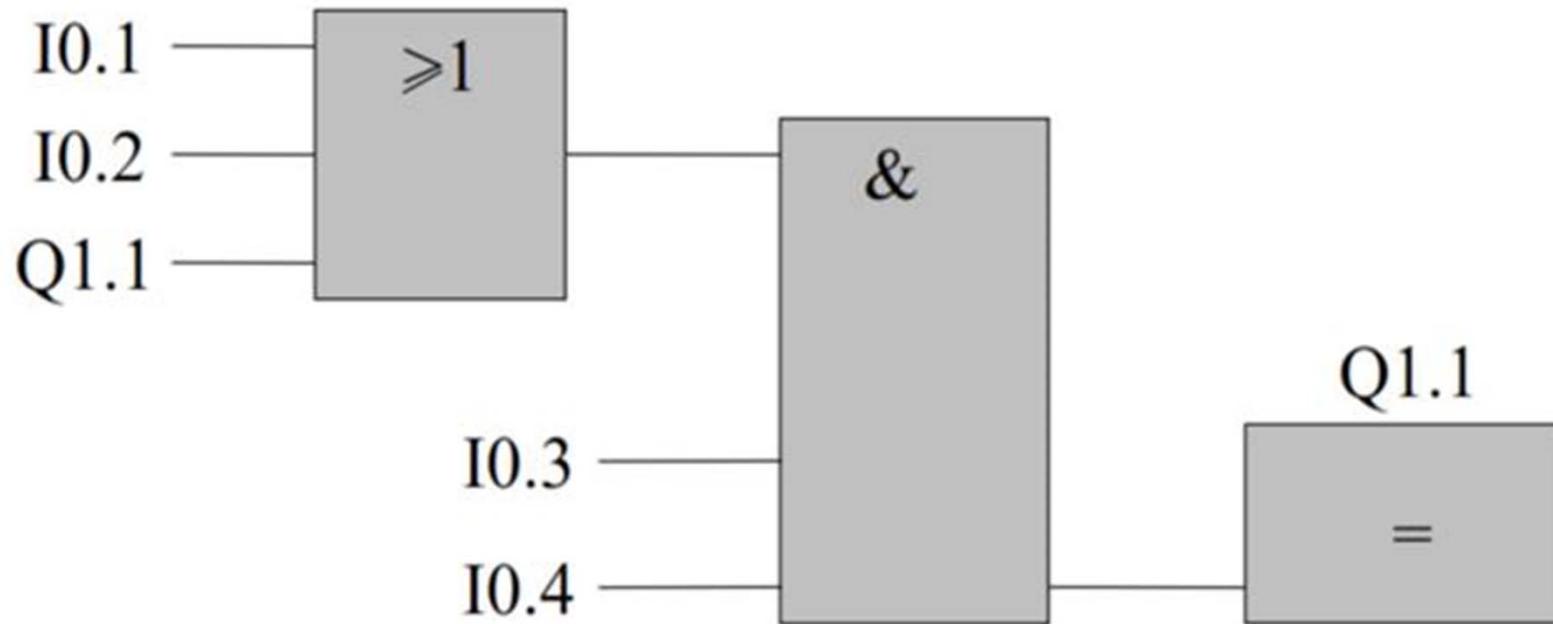
-- | / | --

-- ( / ) --

# Funktionsbausteinsprache FBS/FUP

- Symbolik der digitalen Schaltungen angelehnt an Logikglieder, die miteinander verknüpft werden.
  - UND-, ODER-Gatter
  - INVERTIERTER/NEGIERTER Eingang
  - Merker, Zähler, Zeitgeber, usw.
- Gut strukturierte und übersichtliche Programmierung
- Grafisch anschauliche Programmierung des Informationsflusses

# Beispiel Funktionsbausteinsprache FBS/FUP



# Strukturierter Text - ST

- Höhere Programmiersprache
- Vorteile:
  - Sehr kompakte Formulierung
  - Abstrakte maschinenferne Befehle
  - Übersichtlicher Aufbau durch Anweisungsblöcke
  - Umfangreiche, komplexe Aufgaben realisierbar
- Nachteile:
  - Qualität des Maschinencodes ist abhängig vom Compiler (Übersetzer)
  - Effizienzverlust bei der Laufzeit durch höhere Abstraktion (ST-code i.d.R. langsamer als AWL-Code)
- Zusätzliche Sprachmittel für Zeitverarbeitung und Prozessdatenzugriff

# Ausdrücke (1)

- Auswertereihenfolge
  - Punkt vor Strich  $A+B*C$
  - Auswertung von links nach rechts  $A+B+C$
  - Auswertung linker Operand zuerst  $\text{Sin}(A)*\text{Sin}(B)$
- Boolesche Ausdrücke
  - $(A>B) \& (C>D)$
- Operatoren
  - Funktionsaufruf:  $\text{Sin}(A)$
  - -(negation), NOT, +
  - $<, >, \leq, \geq, =$
  - $\&, \text{AND}, \text{OR}, \text{XOR}$

# Beispiele ST

## (Viele Wege führen nach Rom)

- **Beispiel 1:**

```
Q1.1 := (I0.1 OR I0.2 OR Q1.1) AND I0.3 AND I0.4;
```

- **Beispiel 2:**

```
M = Q1.1; Q1.1 := 0;  
IF (I0.1 OR I0.2 OR M) THEN  
  IF (I0.3 AND I0.4) THEN  
    Q1.1 := 1;  
  END_IF;  
END_IF;
```

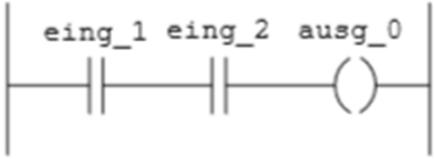
- **Negativbeispiel 3:**

```
Q1.1 := ((I0.1 + I0.2 + Q1.1) > 0) * I0.3 * I0.4
```

# Siemens S7 vs. IEC 61131-3

- Die Programmiersprachen verglichen mit der Norm
  - KOP entspricht dem „Kontaktplan“
  - FUP entspricht der „Funktionsbausteinsprache“
- AWL entspricht der Sprache „Anweisungsliste“, aber hinsichtlich der Operationen bestehen wesentliche Unterschiede
- S7-Graph entspricht der Sprache „Ablaufsprache“

# Zusammenfassung Siemens S7 und EN 61131-3

ISO DIN EN 61131-3		Ältere Sprachen nach DIN 19239 bzw. VDI 2880
AWL – Anweisungsliste IL – Instruction List	LD eingang_1 AND eingang_2 ST ausgang_0	AWL – Anweisungsliste
ST – Strukturierter Text ST – Structured Text	C := A AND NOT B	
KOP – Kontaktplan LD – Ladder Diagram		KOP – Kontaktplan
FBS – Funktionsbaustein-Sprache FBD – Function Block Diagram		FUP – Funktionsplan
AS – Ablaufsprache SFC – Sequential Function Chart		Funktionsplan nach DIN 40719, Teil 6

# Logische Funktionen

- Mit der SPS lassen sich Schaltnetze (logische Funktionen) programmieren.
- In Digitalschaltungen werden logische Steuersignale zusammengeführt und nach festgelegten Gesetzen der Booleschen Algebra ausgewertet.
- George Boole entwickelte um 1850 eine Algebra, die mit zwei Konstanten 0 und 1 und den Operatoren UND, ODER sowie NICHT auskommt. Jede Variable kann nur zwischen den Werten WAHR/TRUE und FALSCH/FALSE wechseln, vereinfacht zu 0 und 1.
- Das Ergebnis ist ebenfalls als Variable zu betrachten und kann daher auch nur die Werte 0 oder 1 annehmen. Die Schreibweise der Operatorsymbole ist nicht immer einheitlich.

# Schaltalgebra

- Bei Speicherprogrammierbaren Steuerungen wird mit den binären Verknüpfungen UND, ODER und NICHT eine Aussagenlogik realisiert. Mit einer Aussagenlogik wird im Grunde eine Schaltfunktion abgeleitet.
- Es gibt die Möglichkeit, solch eine Aussagenlogik mit Hilfe einer Schaltwerttabelle darzustellen und abzulesen. Alternativ könnte man eine Schaltfunktion auch mit Hilfe einer besonderen Art mathematisch darstellen. Die Darstellung in mathematischer Form nennt man **Schaltalgebra**. Diese hat ihre Grundlagen in der **Booleschen Algebra**, weshalb auch hier die Gesetze der gewöhnlichen Algebra angewendet werden.
- Der Vorteil, eine Schaltfunktion in einen schaltalgebraischen Ausdruck zu bringen ist, dass man diese mathematisch berechnen und mit Hilfe der algebraischen Gesetze umwandeln kann. Ein SPS-Programmierer sollte daher die Schaltalgebra sowie die Gesetze für die Umwandlung eines schaltalgebraischen Ausdrucks kennen.
- Ein schaltalgebraischer Ausdruck kann man wie eine mathematische Formel betrachten, z.B.  $S1 \cdot S2 = Q$ . Es enthält Variablen, bei der die Buchstaben in Großbuchstaben dargestellt werden. Die Variablen können dabei, beziehend auf das Dualsystem in der Digitaltechnik, nur die beiden Zustände 0 (Aus) oder 1 (An) haben.

# Darstellung und Verknüpfung

- Um eine Aussagenlogik mit den binären Verknüpfungen UND, ODER und NICHT in einem schaltalgebraischen Ausdruck darzustellen, haben sich verschiedene Darstellungsarten etabliert. Hierbei wird grundsätzlich zwischen der Darstellungsart in Boolescher Algebra und in mathematischer Form unterschieden.

Zeichen für schaltalgebraischen Ausdruck			
Darstellungsart	UND (Konjunktion)	ODER (Disjunktion)	NICHT (Negation)
Boolesche Algebra	$\wedge$	$\vee$	$\neg$ oder $\bar{\phantom{x}}$
Mathematisch	$\cdot$	$+$	$-$
<b>Beispiele</b>			
Boolesche Algebra	$S1 \wedge S2$	$S1 \vee S2$	$\neg S1$ oder $\bar{S1}$
Mathematisch	$S1 \cdot S2$	$S1 + S2$	$\bar{S1}$

# Gesetze und Eigenschaften

Boolesche Algebra		„gewöhnliche“ Algebra	Gesetze
$A \wedge B = B \wedge A$	X	$A \bullet B = B \bullet A$	1. Kommutativgesetz
$A \vee B = B \vee A$	X	$A + B = B + A$	2. Kommutativgesetz
$A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$	X	$A \bullet (B + C) = A \bullet B + A \bullet C$	1. Distributivgesetz
$A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$		$A + (B \bullet C) \neq (A + B) \bullet (A + C)$	2. Distributivgesetz
$A \wedge (B \wedge C) = (A \wedge B) \wedge C = A \wedge B \wedge C$	X	$A \bullet (B \bullet C) = (A \bullet B) \bullet C = A \bullet B \bullet C$	1. Assoziativgesetz
$A \vee (B \vee C) = (A \vee B) \vee C = A \vee B \vee C$	X	$A + (B + C) = (A + B) + C = A + B + C$	2. Assoziativgesetz
$\overline{A \wedge B} = \overline{A} \vee \overline{B}$			1. De Morgan Theorem
$\overline{A \vee B} = \overline{A} \wedge \overline{B}$			2. De Morgan Theorem
$A \vee (A \wedge B) = A$			1. Absorbtionsgesetz
$A \wedge (A \vee B) = A$			2. Absorbtionsgesetz
$A \wedge (\overline{A} \vee B) = (A \wedge B)$			3. Absorbtionsgesetz
$A \vee (\overline{A} \wedge B) = (A \vee B)$			4. Absorbtionsgesetz



# Logische Signale und Funktionen

- Die Funktion einer Schaltung wird durch die **Wahrheitstabelle/Schaltwerttabelle** vollständig beschrieben.
- Die Wahrheitstabelle beschreibt vollständig eine kombinatorische logische Schaltung.
- Unter kombinatorischer Schaltung versteht man eine Schaltung, bei der die Ausgangsgrößen ausschließlich von den Eingangsgröße abhängen.

Eingang Ausgang

E1	A1
0	0
1	1

2 Zustände möglich

Eine einfache Schaltwerttabelle mit einem Eingang und einem Ausgang

← Eingänge →      → Ausgang ←

E2	E1	A1
0	0	0
0	1	1
1	0	1
1	1	0

4 Zustände möglich

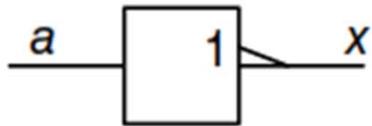
Schaltwerttabelle mit 2 Eingängen und einem Ausgang, vergleichbar mit einer Schaltung für einen Flur mit zwei Schaltern und einer Lampe.

# Elementare Gatter

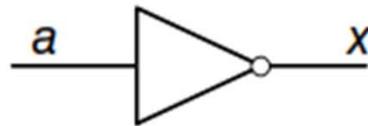
- Schaltungen können durch die Verwendung und Verknüpfung von binären Grundverknüpfungen beschrieben werden.
  - Es werden sowohl die gültigen IEC-Symbole aufgelistet als auch die veralteten amerikanischen Schaltzeichen.
- Die folgenden Gatter werden nur mit der minimalen Anzahl von Eingangsgrößen gezeichnet. Die Erweiterung auf mehrere Eingangsgrößen ist jedoch einfach.

# Elementare Gatter - Inverter

Symbol



amerik. Symbol



Wahrheitstabelle

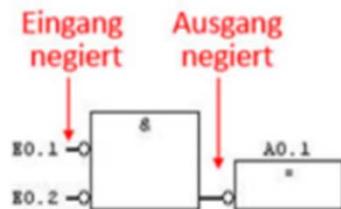
a	x
0	1
1	0

*Boolsche Funktion*

DIN-Schreibweise:  $x = \bar{y}$  (x = nicht a)

amerik. Schreibweise:  $x = a'$  (x = not a)

Funktionsplan - FUP



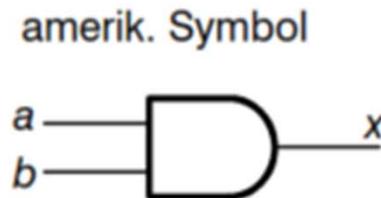
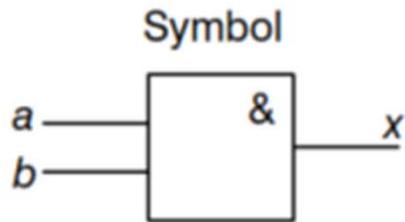
Kontaktplan- KOP



Anweisungsliste - AWL

UN E 0.1  
= A 0.3

# Elementare Gatter - Konjunktion AND



Wahrheitstabelle

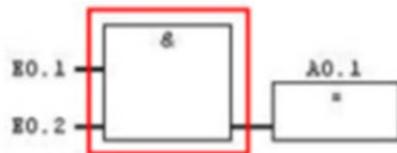
a	b	x
0	0	0
0	1	0
1	0	0
1	1	1

*Boolsche Funktion*

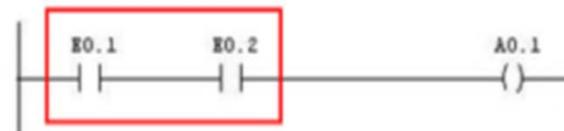
DIN-Schreibweise:  $x = a \wedge b$  (x = a und b)

amerik. Schreibweise:  $x = a \cdot b$  (x = a and b)

Funktionsplan - FUP



Kontaktplan- KOP

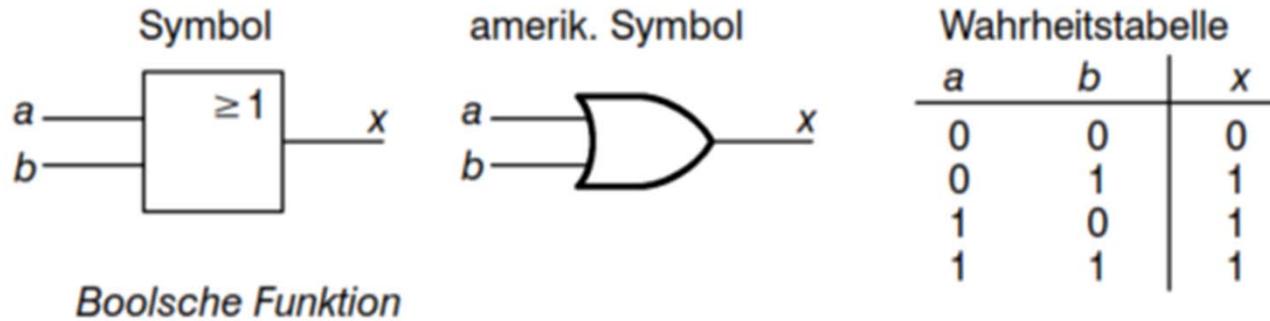


Anweisungsliste - AWL

```

U E 0.1
U E 0.2
= A 0.3
    
```

# Elementare Gatter - Disjunktion OR

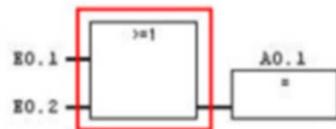


Boolsche Funktion

DIN-Schreibweise:  $x = a \vee b$  (x = a oder b)

amerik. Schreibweise:  $x = a + b$  (x = a or b)

Funktionsplan - FUP



Kontaktplan- KOP

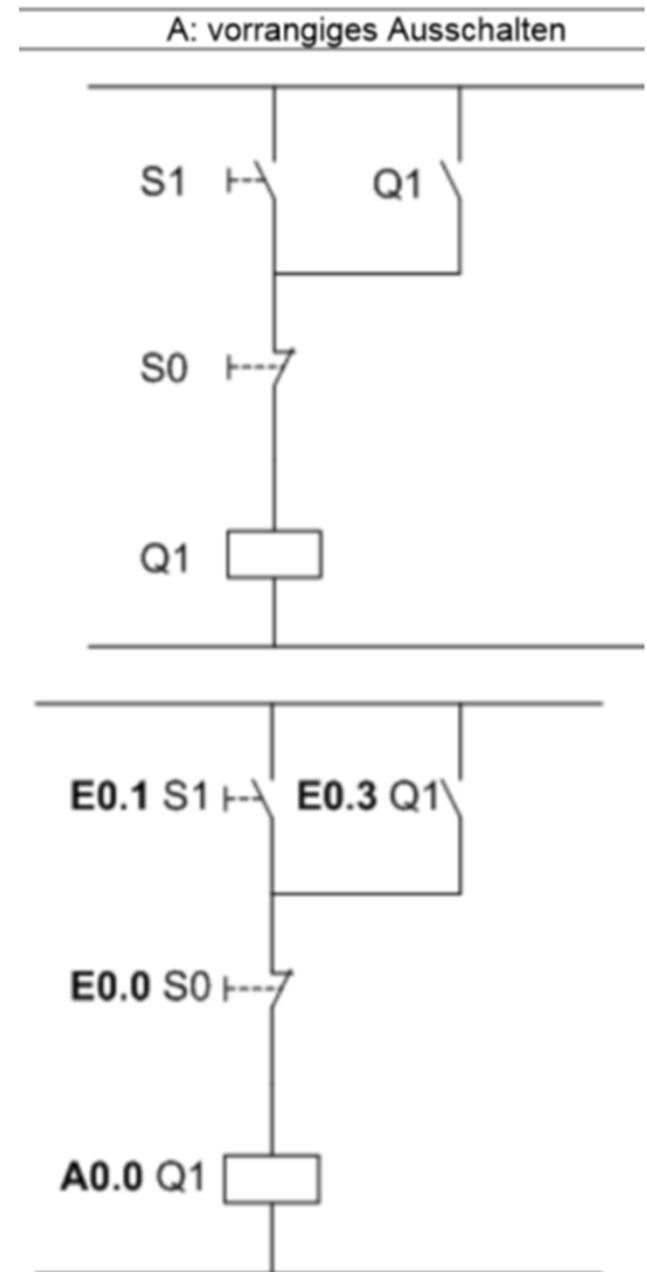


Anweisungsliste - AWL

O E 0.1  
 O E 0.2  
 = A 0.3

# Übung: Motorsteuerung

- Aufgabe: Vorrangiges Ausschalten
  - Bei Betätigung des Tasters S1 zieht das Schütz Q1 an und schließt den Schließerkontakt Q1. Das Schütz Q1 bleibt auch betätigt, nachdem der Taster S1 nicht mehr betätigt wird. Erst wenn der Taster S0 gedrückt wird, fällt das Schütz Q1 ab.
  - Beim gleichzeitigen Betätigen der beiden Taster zieht das Schütz Q1 nicht an, da der Stromkreis nicht geschlossen ist.



# Lösung: Motorsteuerung

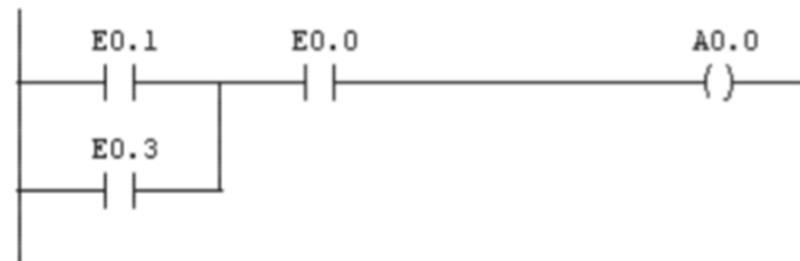
## AWL

```
U(  
0   E   0.1           //Ein-Taster S1  
0   E   0.3           //Schützkontakt Q1  
)  
U   E   0.0           //Aus-Taster S0  
=   A   0.0           //Schütz Q1
```

## FUP



## KOP



# Übung: Klammersetzung

- Wie in der Mathematik wird mit der Klammer „()“ in der AWL-Programmierung festgelegt, in welcher Reihenfolge das Programm abgearbeitet wird.
- Aufgabe: Erstelle zu den beiden vorgegebenen AWL-Programmabschnitten, jeweils den entsprechenden **Funktionsplan**

- AWL zu Programm 1:

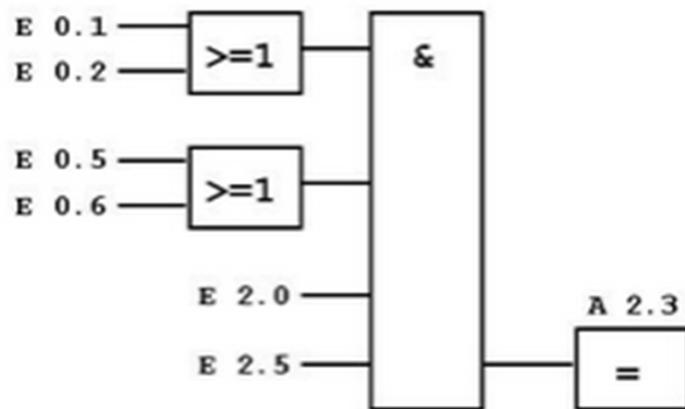
```
U(  
O E 0.1  
O E 0.2  
)  
U(  
O E 0.5  
O E 0.6  
)  
U E 2.0  
U E 2.5  
= A 2.3
```

- Programm 2, ohne Klammern:

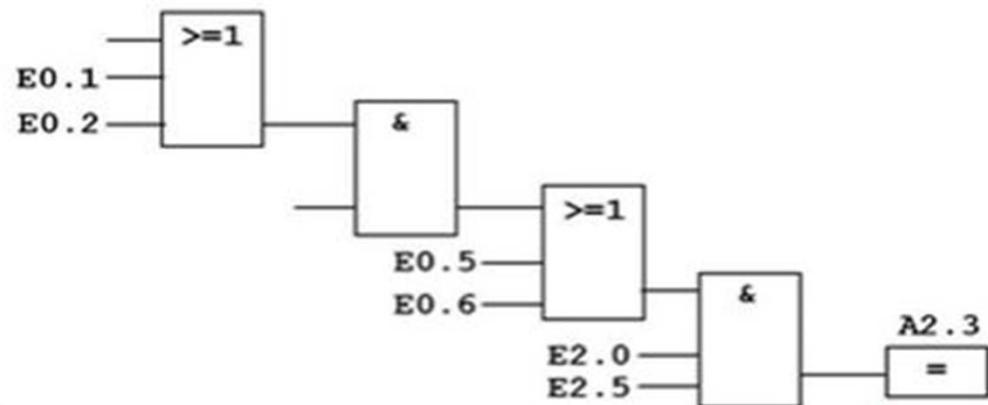
```
U  
O E 0.1  
O E 0.2  
U  
O E 0.5  
O E 0.6  
U E 2.0  
U E 2.5  
= A 2.3
```

# Lösung: Klammersetzung

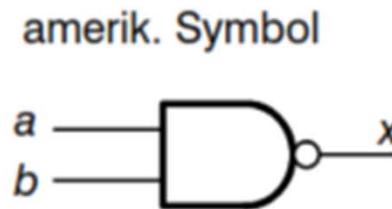
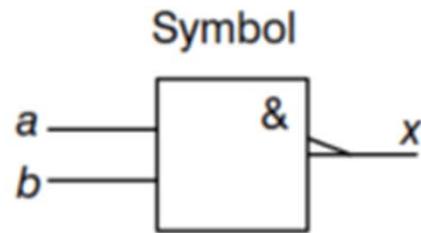
- Wenn man die Klammer weglässt, entsteht eine ganz andere Reihenfolge der Abarbeitung des Programms.
- Darstellung des Programms 1 in FUP:



- Programm 2, ohne Klammern:



# NAND



Wahrheitstabelle

a	b	x
0	0	1
0	1	1
1	0	1
1	1	0

*Boolsche Funktion*

DIN-Schreibweise:  $x = \overline{a \wedge b}$

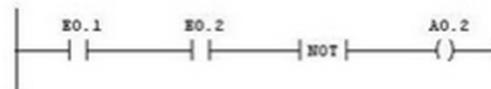
amerik. Schreibweise:  $x = (a \cdot b)'$  ( $x = a \text{ nand } b$ )

Mit NAND-Gattern lassen sich beliebige logische Funktionen realisieren.

Funktionsplan - FUP



Kontaktplan- KOP

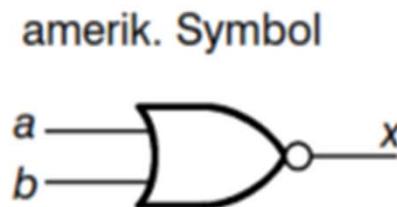
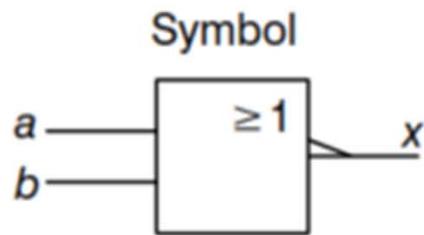


Anweisungsliste - AWL

```

U E 0.1 //Erstabfrage, Ergebnis wird ins VKE geschrieben
U E 0.2 //Es wird abgefragt, ob EO.2 UND das VKE eine 1 sind
NOT //Negiere das VKE
= A 0.3 //Zuweisung des VKE an die Adresse A0.3
    
```

# NOR



Wahrheitstabelle

<i>a</i>	<i>b</i>	<i>x</i>
0	0	1
0	1	0
1	0	0
1	1	0

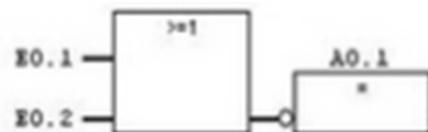
*Boolsche Funktion*

DIN-Schreibweise:  $x = \overline{a \vee b}$

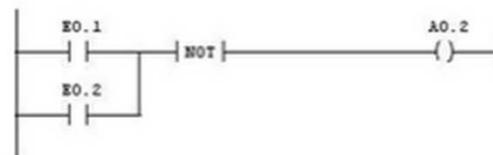
amerik. Schreibweise:  $x = (a + b)'$  ( $x = a \text{ nor } b$ )

Mit NOR-Gattern lassen sich beliebige logische Funktionen realisieren.

Funktionsplan - FUP



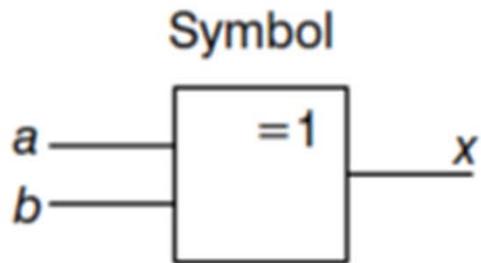
Kontaktplan- KOP



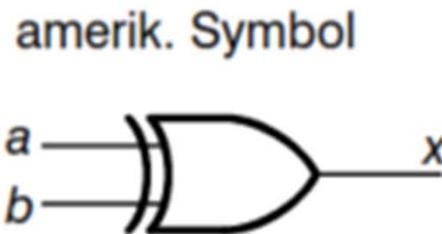
```

O E 0.1 //Erstabfrage, Ergebnis wird ins VKE geschrieben
O E 0.2 //Es wird abgefragt, ob E0.2 ODER das VKE eine 1 sind
NOT //Negiere das VKE
= A 0.3 //Zuweisung des VKE an die Adresse A0.3
    
```

# Antivalenz (exklusiv ODER, XOR)



Boolsche Funktion



Wahrheitstabelle

a	b	x
0	0	0
0	1	1
1	0	1
1	1	0

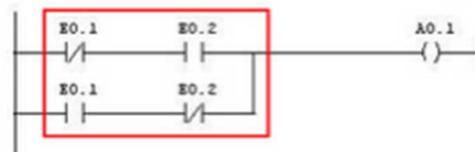
DIN-Schreibweise:  $x = a \oplus b = (\bar{a} \wedge b) \vee (a \wedge \bar{b})$

amerik. Schreibweise:  $x = (a \oplus b)$  ( $x = a \text{ xor } b$ )

Funktionsplan - FUP



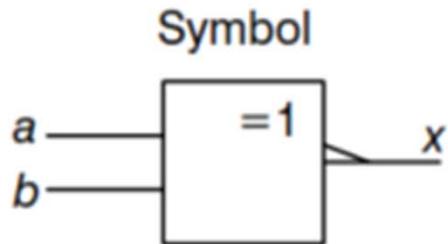
Kontaktplan- KOP



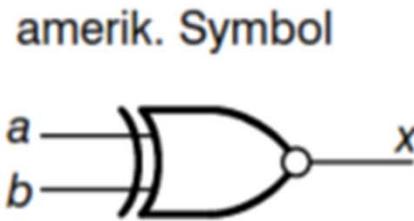
Anweisungsliste - AWL

```
X E 0.1 //Erstabfrage, Ergebnis wird ins VKE geschrieben
X E 0.2 //Es wird abgefragt, ob E0.2 oder das VKE exklusiv eine 1 ist
= A 0.3 //Zuweisung des VKE an den Ausgang A0.3
```

# Äquivalenz (XNOR)



Boolsche Funktion



Wahrheitstabelle

a	b	x
0	0	1
0	1	0
1	0	0
1	1	1

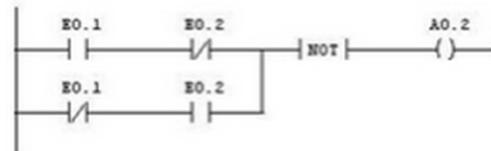
DIN-Schreibweise:  $x = \overline{a \oplus b} = (\overline{a} \wedge \overline{b}) \vee (a \wedge b)$

amerik. Schreibweise:  $x = (a \oplus b)'$  ( $x = a \text{ xnor } b$ )

Funktionsplan - FUP



Kontaktplan- KOP



Anweisungsliste - AWL

```
XN E 0.1 //Erstabfrage, Ergebnis wird ins VKE geschrieben
XN E 0.2 //Es wird abgefragt, ob E0.2 exklusiv 0 ODER das VKE 1 ist
= A 0.3 //Zuweisung des VKE an die Adresse A0.3
```

# Darstellung der Operanden

- Für die Darstellung der Eingänge, Ausgänge etc. werden Operanden benötigt, damit die SPS weiß, was sie mit wem machen soll. Hierfür gibt es verschiedene Arten von Operanden.
- Wenn beispielsweise der Signalzustand eines Schalters abgefragt werden soll, dann wird dafür ein Operand vom Typ Eingang benutzt.
- Soll z.B. ein Motor eingeschaltet werden, wird dafür ein Operand vom Typ Ausgang benutzt.
- Jeder Operand hat einen Adressierungsbereich (Bit, Byte, Wort, Doppelwort). Welcher Adressierungsbereich benutzt wird, hängt vom Datentyp ab. Die Bezeichnung der Operanden ist entweder vom Hersteller vorgegeben oder kann international nach der Norm EN 61131 vorgenommen werden.
- Nachfolgend eine Übersicht über die Operanden, aufgeteilt nach IEC (Norm 61131) oder die Bezeichnung in Step7.

# Übersicht Darstellung Operanden (1)

IEC	Step7	Beschreibung, Beispiel	Datentyp	Adressbereich
I	E	<a href="#">Eingangsbit</a> , E0.0	BOOL	0.0 bis 65535.7
IB	EB	<a href="#">Eingangsbyte</a> , EB1	BYTE, CHAR	0 bis 65535
IW	EW	<a href="#">Eingangswort</a> , EW10	WORD, INT, S5TIME, DATE	0 bis 65534
ID	ED	<a href="#">Eingangsdoublewort</a> , ED20	DWORD, DINT, REAL, TOD, TIME	0 bis 65532
Q	A	Ausgangsbit, A0.0	BOOL	0.0 bis 65535.7
QB	AB	Ausgangsbyte, AB1	BYTE, CHAR	0 bis 65535
QW	AW	Ausgangswort, AW10	WORD, INT, S5TIME, DATE	0 bis 65534
QD	AD	Ausgangsdoublewort, AD20	DWORD, DINT, REAL, TOD, TIME	0 bis 65532
M	M	Merkerbit, M0.0	BOOL	0.0 bis 65535.7
MB	MB	Merkerbyte, MB1	BYTE, CHAR	0 bis 65535
MW	MW	Merkerwort, MW10	WORD, INT, S5TIME, DATE	0 bis 65534
MD	MD	Merkerdoublewort, MD20	DWORD, DINT, REAL, TOD, TIME	0 bis 65532

# Übersicht Darstellung Operanden (2)

IEC	Step7	Beschreibung, Beispiel	Datentyp	Adressbereich
PIB	PEB	Peripherieeingangsbyte, PEB10	BYTE, CHAR	0 bis 65535
PQB	PAB	Peripherieausgangsbyte, PAB10	BYTE, CHAR	0 bis 65535
PIW	PEW	Peripherieeingangswort, PEW10	WORD, INT, S5TIME, DATE	0 bis 65534
PQW	PAW	Peripherieausgangswort, PAW10	WORD, INT, S5TIME, DATE	0 bis 65534
PID	PED	Peripherieeingangs- Doppelwort, PED10	DWORD, DINT, REAL, TOD, TIME	0 bis 65532
PQD	PAD	Peripherieausgangs- Doppelwort, PAD10	DWORD, DINT, REAL, TOD, TIME	0 bis 65532
T	T	<a href="#">Timer</a> , T1	TIMER	0 bis 65535
C	Z	<a href="#">Zähler</a> , Z1	COUNTER	0 bis 65535
FB	FB	<a href="#">Funktionsbaustein</a> , FB10	FB	0 bis 65535
OB	OB	<a href="#">Organisationsbaustein</a> , OB1	OB	1 bis 65535
DB	DB	<a href="#">Datenbaustein</a> , DB10, DBX0.0, DBB1, DBW1	DB, FB, SFB, UDT	1 bis 65535
FC	FC	<a href="#">Funktion</a> , FC1	FC	0 bis 65535
SFB	SFB	<a href="#">Systemfunktionsbaustein</a> , SFB30	SFB	0 bis 65535
SFC	SFC	<a href="#">Systemfunktion</a> , SFC30	SFC	0 bis 65535
VAT	VAT	Variablentabelle		0 bis 65535
UDT	UDT	Userdefinierter Datentyp	UDT	0 bis 65535

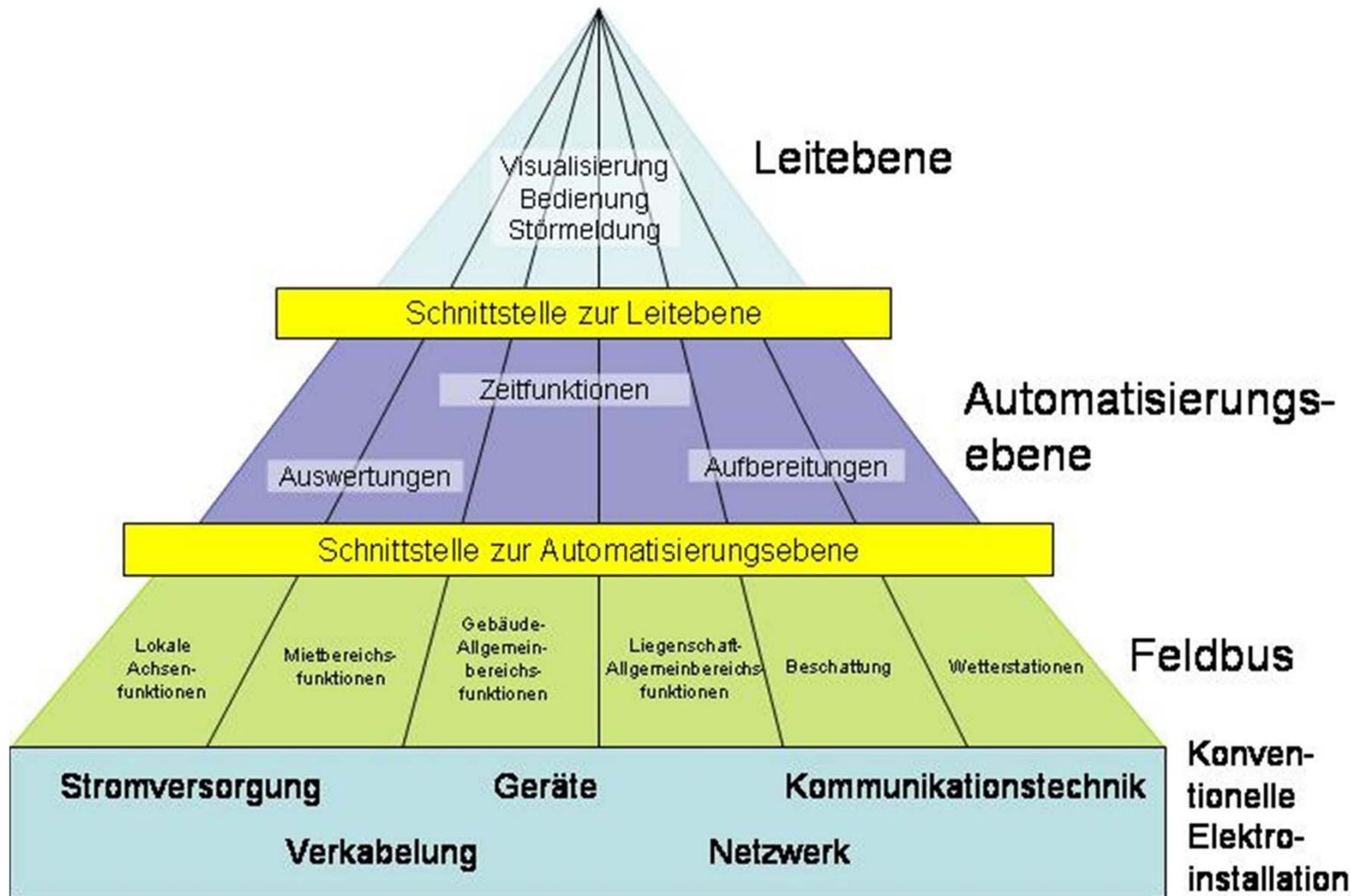
Kommunikation/Schnittstellen

# BUSSYSTEME

# Einleitung

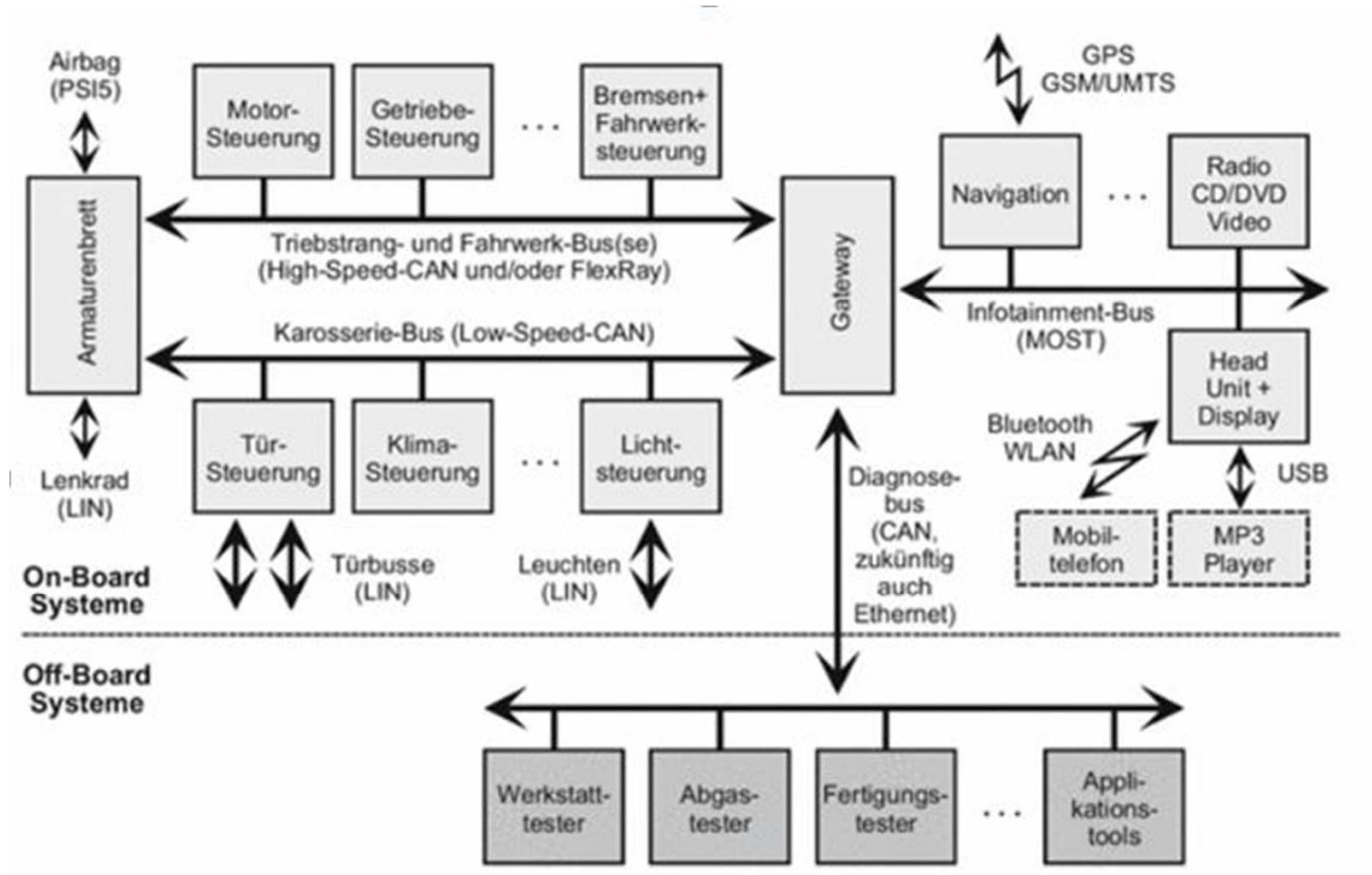
- Fertigungsprozesse werden immer stärker automatisiert. Dazu müssen über Sensoren viele unterschiedliche Signale erfasst und in einem Automatisierungssystem (z.B. SPS oder Industrie-PC) verarbeitet werden.
- Außerdem erhalten viele Aktoren ihre Signale vom Automatisierungssystem.
- Damit nicht alle Sensoren und Aktoren einzeln an das System angeschlossen werden müssen, werden häufig Bussysteme verwendet. Gründe für deren Verwendung sind im Wesentlichen, Aufwand und Kosten für die Verlegung und den Anschluss einzusparen.

# Automatisierungspyramide



Quelle: FH Dortmund

# Bussysteme eines modernen Fahrzeugs



Quelle: Jahr ?

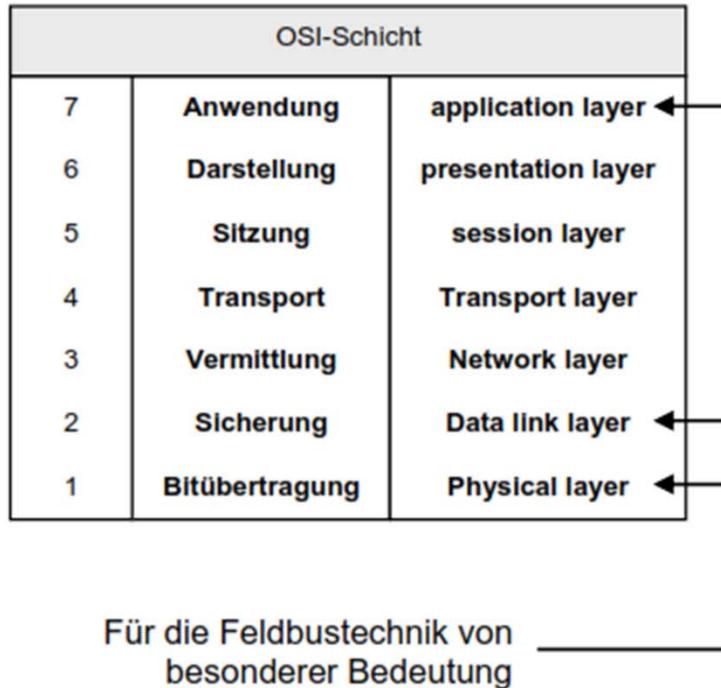
JPS3



# OSI

- Wo immer Geräte unterschiedlicher Hersteller kommunizieren sollen, kommt es zu Problemen durch nicht abgestimmte Protokolle und Zugriffsverfahren. Um solche Probleme zu lösen und zu Standards der Kommunikation zu gelangen, wurde bereits in den siebziger Jahren das Open System Interconnection Reference Model (OSI-Modell) standardisiert.
- Das Modell untergliedert formal die Kommunikation in sieben aufeinanderfolgende Schichten mit klar definierten Aufgaben. Die internationale Normung führte zu dem Namen ISO / OSI Referenzmodell, umgangssprachlich 7-Schichten-Modell.
- Wo immer Bussysteme entwickelt oder über Feldbussysteme publiziert wird, erfolgt dies auf der Basis des OSI-Referenzmodells, genannt auch **7-Schichten-Modell**. Für die Feldbustechnik von besonderer Bedeutung sind zumeist die Schichten 1, 2 und 7.

# OSI



Die Kommunikation durchläuft vom Anwender bis zum Übertragungskanal maximal 7 Schichten.

Jede Schicht übernimmt eine abgegrenzte Teilfunktion.

Zwischen benachbarten Schichten sind die erforderlichen Schnittstellen festgelegt.

Jede Schicht übernimmt von der im Datenfluß vorhergehenden Schicht Daten und übergibt sie an die nachfolgende Schicht.

Jede Schicht bietet der übergeordneten Schicht Dienste an.

Die übergeordnete Schicht fordert Dienste an.

Die Dienste werden nach Protokollen ausgeführt.

# Einleitung

- In der Vergangenheit konkurrierten eine große Anzahl von Feldbussystemen.
- Heute hat sich eine Reihe von Feldbussen etabliert:

PROFIBUS-DP	Siemens Simatic	Maschinenbau, Fertigungstechnik
PROFIBUS-PA	Siemens Simatic	Chemische Industrie, Verfahrenstechnik
CAN	Bosch	Fahrzeugtechnik, Antriebstechnik, Maschinensteuerung (Textil, Aufzüge)
CANopen	Modifikation von CAN 2002 durch CAN-in-Automation Nutzer- und Herstellervereinigung (CiA)	
Device Net	Allen Bradley / Rockwell	USA, wie CAN
MODBUS	AEG Modicon	Maschinensteuerungen
INTERBUS	Phoenix Contact	Antriebstechnik, Maschinensteuerung, Automobilbau
LON und KNX		Gebäudeautomatisierung

# Kommunikation über Bussysteme

- Kommunikation zwischen Geräten verlangt immer
  1. eine bestimmte Physik der Übertragungsmedien wie Kupfer-Doppelader, LWL-Leiter, Funk, Stecker etc.,
  2. geeignete Vereinbarungen (Protokolle) für den (seriellen) Versand der Datenpakete. Protokolle bestimmen den Aufbau der Telegramme, die Regeln der Sicherung der Übertragung, die Regeln der Adressierung der Teilnehmer u.a., sowie
  3. ein bestimmtes Zugriffsverfahren der Bussysteme auf das JPS1 gemeinsame Übertragungsmedium.



# Anforderungen an Feldbusse

- Zuverlässige und montagefreundliche Busanschlüsse
- Bidirektionaler Informationsfluss
- An- bzw. Abkoppeln von Geräten ohne Auswirkung
- Ausfall von Geräten ohne Funktionsbeeinträchtigung für den restlichen Bus, bzw. Möglichkeit zur Fehlerlokalisierung
- Galvanisch getrennt
- Versorgungsspannung über Bus
- Offenheit
- Deterministisches Zeitverhalten
- Echtzeitfähigkeit
- Nicht jeder Feldbus erfüllt alle Anforderungen

# Hauptanforderungen an Feldbusse

- Echtzeitfähigkeit
  - Kann ein System unter allen Betriebsbedingungen richtig und rechtzeitig auf alle auftretenden Ereignisse reagieren, so gilt es als echtzeitfähig.
- Rechtzeitigkeit
  - Dies kann bedeuten, dass für die Zeit zwischen zwei Ereignissen ein bestimmte Maximalzeit, die sogenannte Deadline, nicht überschritten werden darf. Dies bezeichnet man als die Forderung nach der rechtzeitigen Ausführung einer Aktion.
- Determiniertheit
  - Ist ein System, dessen Zustand zu jedem Zeitpunkt eindeutig festgelegt ist. In einem deterministischen Kommunikationssystem sind damit alle Zustände der Datenübertragung zeitlich exakt berechenbar.

# M-Bus (Meter-Bus)

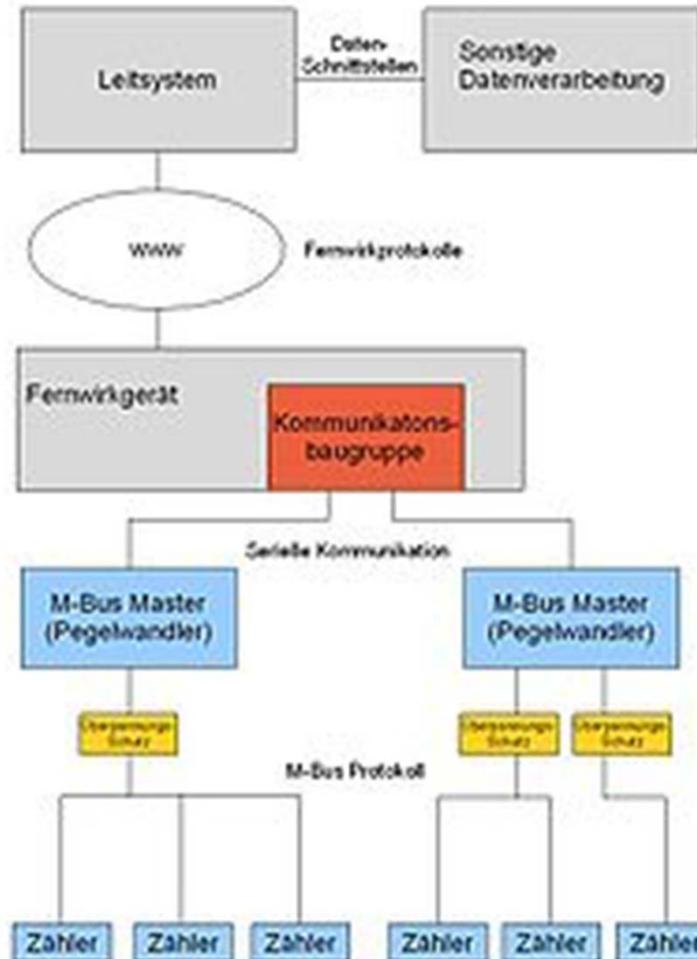
- Der **M-Bus**, kurz für Meter-Bus, ist eine technische Norm, durch deren Anwendung zum Beispiel in Stromzählern der Verbrauch von Strom als Messdaten übertragen werden kann. Auch der Verbrauch von Gas, Wärme oder Wasser kann gemessen und durch Zähler mit M-Bus übertragen werden.
- Als sogenanntes Bussystem ist es eine Technik elektrischer Datenübertragung **von gemessenen Werten genutzter Energie**. Es ist ein "Kommunikationssystem zur Zählerdatenübertragung für verschieden**JPS2**ensoren und Aktoren".
- Sensoren sind technische Fühler zur Messwertaufnahme, Aktoren sind Elemente, die ferngesteuert aktiv in den Energiefluss eingreifen können, z.B. die Strom- oder Gaszufuhr abschalten oder begrenzen können.
- Die europäische Leitlinie dieser Technik geht auf die Entwicklung des M-Bus im Jahre 1992 durch Horst Ziegler von der Uni Paderborn in Zusammenarbeit mit den Unternehmen Techem und Texas Instruments zurück. Als Teil intelligenter Energiezähler spielen sie unter dem Schlagwort Smart Metering eine Rolle. Das Besondere ist dabei das Fernauslesen, bei dem die erhobenen Daten durch angeschlossene weitere Geräte unter anderem über das Internet oder das mobile Funknetz übertragen und damit das Ablesen durch Menschen ersetzen können.



# Verwendung M-Bus

Es gibt M-Bus-Geräte für folgende Messaufgaben:

- Gaszähler
- Stromzähler
- Wärmezähler
- Wasserzähler
- Impulsumsetzer
- Analog Digital Umsetzer
- Feuchte-/ Temperatursensoren



Quelle: wikipedia.org

# Funktionsweise M-Bus

- In einem M-Bus-System wird ein Master benötigt, welcher gleichzeitig den Bus mit Strom versorgt. Dieser ist dafür zuständig, dass die Daten von den M-Bus-Slaves gesammelt und gegebenenfalls gespeichert und/oder weiterverarbeitet werden. Es darf nur einen Master in dem Bus-System geben, da es sonst zu Spannungs- und Paketkollisionen kommt.
- Der Adressraum für primäre Busadressen erstreckt sich über die Adressen 1 bis 250. Es können also 250 Slaves an einem Netz angeschlossen sein. Wenn alle Busteilnehmer auch die 8-stellige sekundäre Adressierung unterstützen, können folglich sehr viel mehr Geräte angeschlossen werden. Das Auslesen über die sekundäre Adresse braucht ungefähr doppelt so lange wie über die primäre Adresse.

# Vorteile M-BUS

- Der M-Bus ist auf der Geräteseite preiswert und einfach zu realisieren.
- Endgeräte können über den Bus mit Strom versorgt werden.
- Die digitale Kommunikation über den Bus ermöglicht, präzise Messwerte an nachfolgende Auswerteeinrichtungen zu übertragen.
- Viele Slaves können an einem Bus (Segment) betrieben werden.
- Mit Hilfe von Repeatern können große Netze aufgebaut werden.
- Durch den Einsatz von M-Bus-Modem-Mastern ist es möglich, den Verbrauch weit entfernter Anlagen zu erfassen.
- Es fallen keine Lizenzkosten an.
- Es sind keine speziellen Kabel notwendig.

# Nachteile M-BUS

- Die Anschlusselemente (Stecker) sind nicht genormt.
- Die Datenübertragung ist vergleichsweise langsam (2400 Baud) und für Prozesssteuerung ungeeignet.
- Die Standardisierung auf Protokollebene ist lückenhaft. Vor dem Einsatz neuer Slaves ist die Kompatibilität zur Auswerteeinheit sicherzustellen.
- Es gibt keine Zertifizierungsstelle, welche die ordnungsgemäße Funktion von Master und Slaves sicherstellt.
- Die Auswerteelektronik (Master) wird mit steigender Gerätezahl am Bus immer aufwändiger und teurer.

# Modbus

- Modbus ist ein Anwendungsprotokoll für den Austausch von Nachrichten zwischen intelligenten Modbus-Controllern, unabhängig von der Netzstruktur. Das Modbus-Protokoll ist auf der Anwendungsschicht des OSI-Referenzmodells angesiedelt und unterstützt den Master-Slave-Betrieb zwischen intelligenten Geräten.  
Beim Modbus-Protokoll handelt sich um ein von Gould-Modicon 1979 entwickeltes Automatisierungs-Protokoll, das als industrieller De-facto-Standard bezeichnet werden kann.
- Das Modbus-Protokoll definiert den Nachrichtentyp, den die Masterstation benutzt und der unabhängig von den Netzwerken ist. Über diesen Nachrichtentyp kommunizieren die Modbus-Controller untereinander. Das Protokoll beschreibt wie ein Modbus-Controller über eine Anfrage Zugang zu einem anderen Controller aufnimmt, wie dieser die Anfrage beantwortet und wie Fehler erkannt und dokumentiert werden. Das Modbus-Protokoll arbeitet auf Anfrage-Antwort-Basis und bietet verschiedene Dienste, die durch Funktions-Codes spezifiziert werden. Während der Kommunikation bestimmt das Modbus-Protokoll wie jeder Controller die Geräte-Adresse erfährt und Nachrichten erkennt, die für ihn bestimmt sind. Außerdem bestimmt es die auszulösenden Aktionen und welche Informationen der Modbus-Controller aus dem Nachrichtenfluss entnehmen kann. Wenn eine Antwort erforderlich ist, dann wird diese im Controller aufgebaut und mit dem Modbus-Protokoll zu der entsprechenden Station gesendet.

# PROFIBUS (process field bus) (1)

- Der Profibus ist ein standardisierter Feldbus für die Automation und Fertigungstechnik. Er ist herstellerunabhängig, hat ein offenes Konzept, ist international standardisiert in EN 50170, EN 50254, IEC 61158 und DIN 19245 und hat eine hierarchische Struktur mit den Ebenen:
  - Sensoren/Aktoren
  - Feldebene
  - Prozessebene.
- Auf der Feldebene werden die digitalen Signale der Sensoren und Aktoren über das AS-Interface übertragen. Über dieses Sensor-Aktor-Netzwerk in Bustechnologie werden die Versorgungsspannung von 24 V für die Endkomponenten und die Datensignale über ein gemeinsames Übertragungsmedium übertragen.
- Auf der Feldebene befinden sich die verteilten peripheren Einheiten, wie die I/O-Module, Mess-Transduktoren, die Steuereinheiten für Ventile und Motoren sowie die Bedienerkonsolen der Administratoren. Die Prozessdaten werden zyklisch übertragen, während Alarmsignale, Parameter und Diagnosedaten auch antizyklisch übertragen werden können.

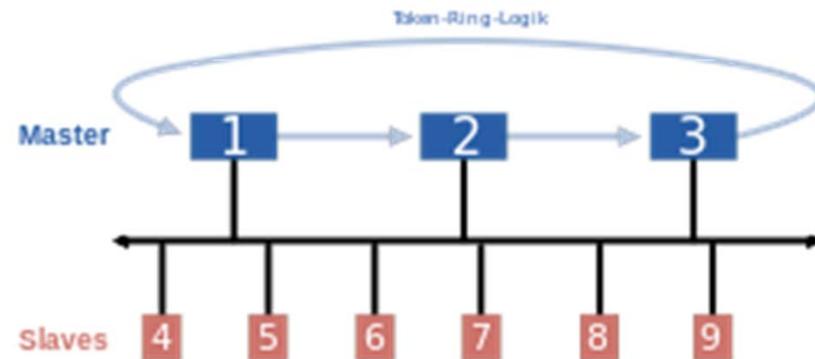
## PROFIBUS (process field bus) (2)

- Auf der Zellenebene befinden sich die programmierbaren Controller, die untereinander kommunizieren. Auf dieser Funktionsebene müssen große Datenpakete mit leistungsfähigen Kommunikationsfunktionen übertragen werden. Darüber hinaus sollten die Profibus-Systeme in die Kommunikations-Infrastruktur des Unternehmens, in Internet und Intranet, integriert werden und dabei die gängigen Netzwerk- und Transportprotokolle benutzen. Daher setzt Profibus auf Ethernet als Übertragungsverfahren in der Automatisierung.
- Das Profibus-Konzept arbeitet im Master-Slave-Betrieb mit dem Zugangsverfahren Token Passing, bei dem die Slaves nur auf Anforderung der Master auf den Profibus zugreifen dürfen. Es gibt mehrere Masters, die die Zugangsberechtigung untereinander austauschen. Wie im Token Ring hat jeder Master eine Vorgängerstation und eine Folgestation und erhält das Token nach einer fest vorgegebenen Laufzeit. Die Teilnehmerzahl ist unbegrenzt.
- Den Profibus, der vernetzt zum Profinet führt, gibt es in mehreren Varianten:
  - Profibus-FMS
  - Profibus-DP
  - Profibus-PA.
- Die Profibus-Aktivitäten werden von der Profibus-Nutzerorganisation (PNO), der größten Feldbus-Organisation, begleitet.

# Profibus-Verfahren: Master-Slave

JPS5

- **Master-Slave-Verfahren**
  - Ein Master, mehrere Slaves
  - Slaves werden i.d.R. zyklisch angesprochen
- **Vorteile:**
  - Einfache Slaves (kostengünstiger)
  - Buszugriff für Teilnehmer festgelegt, damit möglich max. Zeit für Datenübertragung anzugeben,
  - Echtzeit
- **Nachteile:**
  - Master defekt -> Bus steht
  - Slave zu Slave-Übertragung umständlich
  - Zyklisches Abfragen kann auch alte Daten liefern
  - Wachsende Teilnehmerzahl führt zu wachsender Zykluszeit



Quelle: wikipedia.org

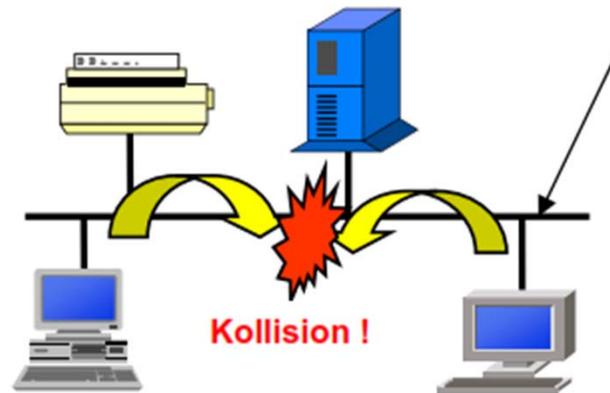


# Profibus-Verfahren: Token passing

- **Token passing Verfahren**
  - Buszugriffsberechtigung reihum an alle Teilnehmer vergeben, die dann als Master wirken
- **Vorteile:**
  - Teilnehmer können direkt miteinander kommunizieren
  - Ein Master kann den anderen überwachen
  - Max. Token Umlaufzeit ist garantiert, grundsätzlich echtzeitfähig
- **Nachteile:**
  - Busverwaltung relativ komplex (teures Businterface)
  - Wachsende Teilnehmerzahl führt zu langen Tokenumlaufzeiten
  - Hinzufügen oder Entfernen einer Station führt zur Rekonfiguration des gesamten Bussystems.

# Buszugriff

- Ein grundlegendes Problem der Kommunikation über Feldbusse ist die Art des Zugriffs. Es muss geklärt werden, was geschehen soll, wenn mehrere Busteilnehmer gleichzeitig auf ein Medium zugreifen.
- Kollisionen von Datentelegrammen müssen verhindert oder in ihren Auswirkungen beherrscht werden.



Quelle: [http://www.ipsta.de/download/freies/Feldbus\\_OSIModell.pdf](http://www.ipsta.de/download/freies/Feldbus_OSIModell.pdf)

# Buszugriff

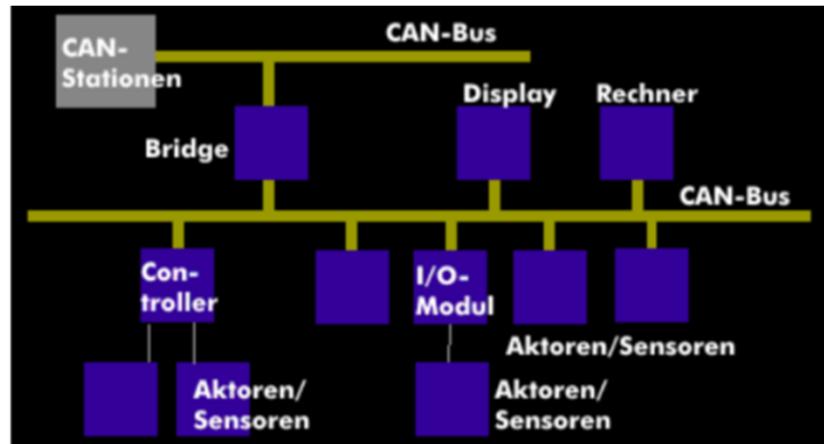
- Dazu haben sich verschiedene Verfahren herausgebildet:
  - Kontrollierte Verfahren:
    - Buszuteilung über festes Zeitraster
      - Jedem Teilnehmer wird unabhängig von seinem Bedarf nacheinander ein Zeitfenster für die Kommunikation zur Verfügung gestellt.
  - Zufällige Verfahren:
    - Bedarfsabhängige Buszuteilung
      - Das Medium wird vor dem Zugriff abgetastet, sodass eine Kollision durch zeitgleichen Zugriff zweier Teilnehmer erkannt wird. Der Zugriff (Sendeversuch) wird dann abgebrochen und nach einer zufällige gewählten Verzögerung erfolgt ein erneuter Versuch.

# CAN (controller area network)

- Ein Controller Area Network (CAN) ist ein von der Internationalen Standardisierungs-Organisation (ISO) standardisierter echtzeitfähiger Feldbus für die serielle Datenübertragung in der Automotive-Technik, der Automation und Fertigungstechnik.
- Um die Kabelbäume in Fahrzeugen (bis zu 2 km) zu reduzieren und dadurch Gewicht zu sparen, wurde der CAN-Bus 1983 von Bosch für die Vernetzung von Steuergeräten in Automobilen entwickelt und 1987 zusammen mit Intel vorgestellt.
- CAN ist als ISO 11898 international standardisiert und definiert die Layer 1 (physikalische Schicht) und 2 (Datensicherungsschicht) im ISO/OSI-Referenzmodell. Die beiden gängigsten Realisierungen der physikalischen Schichten nach ISO 11898-2 (Highspeed-CAN) und ISO 11898-3 (Lowspeed-CAN) unterscheiden sich außer in der Datenrate in weiteren Eigenschaften und sind deshalb nicht zueinander kompatibel.

# Aufbau und Topologie des CAN-Busses.

- Das CAN-Bussystem ist ein seriell arbeitender 2-Draht-Bus mit offener Architektur, der als Übertragungsmedium TP-Kabel mit einer Impedanz von 120 Ohm benutzt. Bei den auf den beiden Leitungen liegenden Signalen handelt es sich um differenzielle Signale, die mit entgegengesetzter Polarität übertragen werden. Andere Übertragungsmedium können vom Benutzer definiert werden. Der CAN-Bus hat eine Linientopologie.

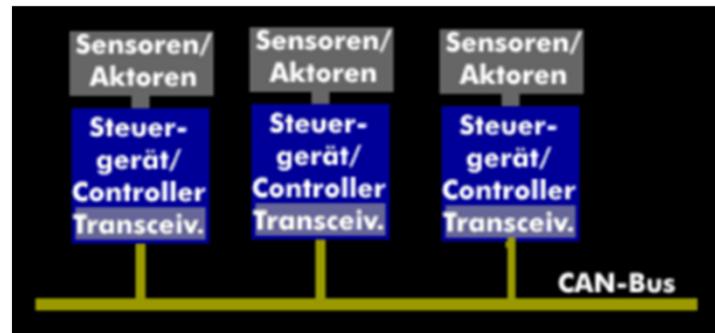


Quelle: <http://www.itwissen.info/definition/lexikon/controller-area-network-CAN-CAN-Bus.html>

- Alle Stationen sind direkt über einen Transceiver an den CAN-Bus angeschlossen. Als Stecker kommt dabei der 9-polige Sub-D-Stecker zum Einsatz. Weitere 4- und 5-polige Stecker sind ebenfalls definiert.

# Aufbau und Topologie des CAN-Busses.

- Die Stationen selbst bestehen aus einem Steuergerät und einem Controller, der den Zugang auf den Bus steuert. Da alle Stationen gleichzeitig am CAN-Bus angeschlossen sind, können sie alle für sie relevanten Daten empfangen. Da das gleichzeitige Senden von mehreren Stationen zu Kollisionen führen würde, benutzt der CAN-Bus ein prioritätengesteuertes Zugangsverfahren, die Arbitration. Die Daten werden dabei in den CAN-Datenrahmen verpackt, der zusätzliche Datenfelder für das Zugangsverfahren und die Fehlersicherung enthält.



Quelle: <http://www.itwissen.info/definition/lexikon/controller-area-network-CAN-CAN-Bus.html>

# Übertragungsprotokolle – Buchauszug (1)

Quelle: Digitale Schnittstellen und Bussysteme: Einführung für das technische Studium JPS6

## 5.2 Übertragungsprotokoll

Im Übertragungsprotokoll wird als Erstes die Übertragungsart festgelegt. Wenn die Information zwischen zwei Teilnehmern abwechselnd ausgetauscht wird, spricht man von Halbduplex. Wird dagegen von beiden Teilnehmern gleichzeitig gesendet und empfangen, so nennt man dies Duplex- oder Vollduplexbetrieb. Bei nur einer Richtung der Datenübertragung hat man Simplexbetrieb.

Darüber hinaus werden im Übertragungsprotokoll die Übertragungsparameter und Protokollverfahren festgelegt.



# Übertragungsprotokolle – Buchauszug (2)

Quelle: Digitale Schnittstellen und Bussysteme: Einführung für das technische Studium

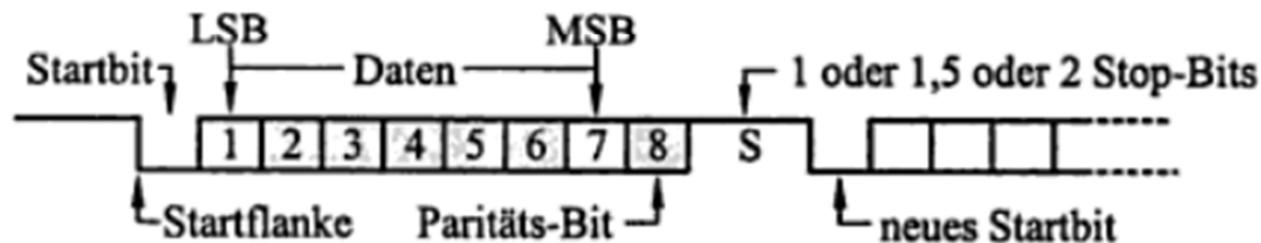
## 5.2.1 Übertragungsparameter

Die *Übertragungsrate* wird in *Baud* angegeben. Sie entspricht bei dualer Codierung  
 $\text{Schritt/s} = \text{Bit/s} = \text{bps (bit per second)} = 1 \text{ Baud}$ .

Im Folgenden werden einige Beispielswerte für Baudraten aufgezählt:

150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 76800, 115200.

Die *Informationstransferrate* ist niedriger als die Übertragungsrate. Sie ergibt sich aus dem Verhältnis von Nutzinformation zu übertragener Information, die sich aus Nutzinformation und Steuerinformation zusammensetzt. Dies soll für das *Datenformat* des asynchronen Start-Stop-Verfahrens dargestellt werden (s. Bild 5-2).



**Bild 5-2: Start-Stop-Format**

# Übertragungsprotokolle – Buchauszug (3)

Bei der asynchronen Datenübertragung werden die Daten in einzelne Zeichen zerlegt. Dem Zeichen, das in dem dargestellten Beispiel aus 7 Bit – LSB (Least Significant Bit) ist das niedrigstwertige und MSB (Most Significant Bit) das höchstwertige Bit – besteht, wird ein Startbit vorangestellt und ein Paritätsbit angefügt. Abschließend folgt noch ein 1 oder 1,5 oder 2 Schritte langes Stop-Bit. In Sendepausen kann dieser Zeitraum auch beliebig lang sein, da der Empfänger durch die Startflanke des Startbits auf die nächste Bitfolge synchronisiert wird.

Legen wir 7 Nutzbits und bei je einem Start-, Paritäts- und Stop-Bit drei Bit Steuerinformation zugrunde, so ergibt sich eine Informationstransferrate von  $7/10$  der Baudrate.

Die Baudrate muss für Sender und Empfänger einheitlich festgelegt werden, ebenso die Codierung der Zeichen, z.B. 7-Bit-ASCII (s. Kap. 5.2.3) oder codeunabhängig. Auch die Verwendung des Paritätsbits muss gleich eingestellt werden.

Das Paritätsbit hat die Aufgabe, die Sicherheit der Datenübertragung zu erhöhen. Man unterscheidet zwischen gerader (even) und ungerader (odd) Parität (parity).

Bei gerader Parität prüft der Sender, ob die Anzahl der Einsen im Datenwort gerade ist. Das Paritätsbit wird dann so gesetzt, dass die Anzahl der Einsen - einschließlich des Paritätsbits - gerade wird. Der Empfänger prüft dann, ob das von ihm berechnete Paritätsbit mit dem übertragenen übereinstimmt.

Für ungerade Parität wird das Paritätsbit entsprechend so gesetzt, dass die Anzahl der Einsen ungerade ist, wieder einschließlich Paritätsbit.

Einzeln auftretende Fehler, sogenannte Einzelfehler, können mit dieser Methode erkannt werden.

# Übertragungsprotokolle – Buchauszug (4)

## 5.2.2 Protokollverfahren

Im Protokollverfahren wird die Empfangs/Sendebereitschaft der Geräte mitgeteilt. Diese wird über die normale Datenleitung oder über zusätzliche Steuerleitungen angezeigt.

### 5.2 Übertragungsprotokoll

---

41

Auch eine Mischung aus beiden ist üblich. Das in der Praxis häufig angewandte *XON/XOFF-Protokoll* wird rein softwaremäßig realisiert. Wenn der Empfänger, z.B. ein Drucker, zunächst keine weiteren Daten entgegennehmen kann, weil der Datenpuffer voll ist, sendet er über die zweite Datenleitung das Steuerzeichen XOFF (13h, entsprechend DC 3 im ASCII-Code). Der Sender, z.B. der Computer, erkennt das Zeichen und unterbricht die Übertragung bis der Empfänger mit dem Steuerzeichen XON (11h, entsprechend DC 1 bei ASCII) seine Empfangsbereitschaft signalisiert. Die zugehörige Schaltung ist in Bild 5-9 (Zweidrahtverbindung) (Kapitel 5.3.6) dargestellt.

Mit zusätzlichen Steuerleitungen, z.B. RTS/CTS in Bild 5-11 (Vierdrahtverbindung) oder DSR/DTR in Bild 5-12 (Mehrdrahthandshake) kann die Empfangsbereitschaft hardwaremäßig, d.h. durch HIGH- oder LOW-Pegel, angezeigt werden. In der Praxis werden die unterschiedlichsten Kombinationen eingesetzt.

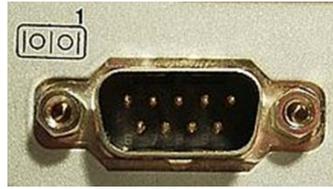
Kommunikation/Schnittstellen

# PROPRIETÄRE SCHNITTSTELLEN

RS-485

RS-232

# RS-232/EIA-232



- Die aus dem Jahre 1962 stammende RS232-Schnittstelle ist zwar mittlerweile ein Veteran unter den seriellen Schnittstellen in der Computertechnik, aber unter Bastlern und bei industriellen Anwenden ist sie nach wie vor sehr beliebt. Die Ursache dafür sind ihre geringen Ansprüche an Hardware und Software.
- RS232 benutzt für die Datenübertragung ein einfaches asynchrones serielles Verfahren. Seriell bedeutet, dass die einzelnen Bits des zu übertragenden Bytes nacheinander über eine einzige Datenleitung geschoben werden. Asynchron heißt, dass es keine Taktleitung gibt, die dem Datenempfänger genau sagt, wann das nächste Bit auf der Datenleitung liegt. So ein Verfahren kann nur funktionieren, wenn Sender und Empfänger mit genau dem gleichen internen Takt arbeiten, und wenn der Empfänger gesagt bekommt, wann das erste Bit genau anfängt (Synchronisation).
- Alle RS232-Leitungen (mit Ausnahme der Masseleitung) arbeiten mit den Spannungspegeln +12V (für eine logische '0') und -12V (für eine logische '1'). (Erlaubt sind jeweils 5V..15V.) Der Datenempfänger erwartet eine Spannung von über +3V für eine 0 und von unter -3V für eine 1.
- Um Daten minimalistisch von einer Datenquelle zu einem Datenempfänger zu übertragen, werden eigentlich nur 2 Drähte benötigt - eine Masseleitung und eine Datenleitung. Für eine bidirektionale Verbindung reichen also 3 Leitungen.
- Um zu vermeiden, dass der Sender Daten sendet, obwohl der Empfänger noch nicht bereit ist, werden 2 bis 4 zusätzliche Handshakeleitungen benötigt.

# RS-232

## Aktuelle Verwendung

- Es werden weltweit kaum noch Geräte produziert, die eine RS-232-Schnittstelle aufweisen. Ausgenommen sind Service- und Konfigurationsanschlüsse bei diversen Geräten (z. B. Router, Switches, Speichersysteme) und POS-Terminals. Moderne Bussysteme und Datenübertragungsprotokolle bieten zuverlässigere und schnellere Verbindungsmöglichkeiten. Nur noch wenige PCs werden mit einem COM-Port ausgeliefert, Notebookhersteller bieten diese Ausstattungsoption nahezu gar nicht mehr an.
- Zur Mikrocontroller-Programmierung oder um ältere, noch in Gebrauch befindliche Geräte, wie z. B. Telefonanlagen und Steuergeräte, mit modernen Computern zu betreiben und programmieren zu können, sind auf dem Markt Schnittstellen-Wandler von USB nach seriell erhältlich.

# RS-485/EIA-485 (1)

- Die RS485-Schnittstelle ist für serielle Hochgeschwindigkeits-Datenübertragungen über große Entfernungen entwickelt worden. Aufgrund der symmetrischen Signalübertragung ist EIA-485 durch eine hohe Toleranz gegenüber elektromagnetischen Störungen gekennzeichnet.
- Im Bereich der industriellen Automatisierung ist die RS485-Schnittstelle immer noch sehr stark verbreitet und wird erst langsam durch Ethernet-basierende Schnittstellen verdrängt.
- Die RS485 ist als bidirektionales Bussystem mit bis zu 32 Teilnehmern konzipiert. Mit modernen Transceiver-ICs ist durch Reduzierung der durch den Busknoten verursachten Belastung der Anschluss von bis zu 128 Teilnehmern an ein Bussystem möglich.
- Da mehrere Sender auf einer gemeinsamen Leitung arbeiten, muss durch ein Protokoll sichergestellt werden, dass zu jedem Zeitpunkt maximal ein Datensender aktiv ist.
- Endgeräte ohne RS485-Anschluss lassen sich, je nach vorhandenen Schnittstellen, auf verschiedenen Wegen problemlos mit einer RS485-Schnittstelle nach- bzw. ausrüsten.

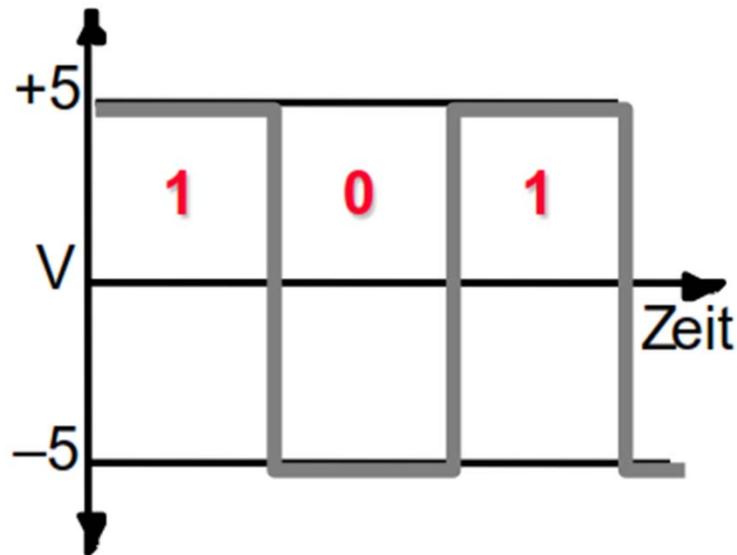
## RS-485/EIA-485 (2)

- Da die EIA-485-Schnittstelle ein Bussystem (im Gegensatz zur Punkt-zu-Punkt-Verbindung bei EIA-232) darstellt, sollten die Leitungsenden (zumindest bei größeren Leitungslängen bzw. größeren Übertragungsraten) abgeschlossen werden. In der Regel wird ein passiver Abschluss durch Verbinden der Signalleitungen über jeweils einen 120- $\Omega$ -Widerstand an den beiden Busenden verwendet.
- EIA-485 spezifiziert nur die elektrischen Eigenschaften des Interfaces, es definiert kein Protokoll und auch keine Steckerbelegung. Deshalb existiert keine einheitliche Pinbelegung eines EIA-485-Steckers, so dass bei Verwendung verschiedener EIA-485-Geräte immer die Dokumentation des Gerätes beachtet werden muss. Beim Profibus, der auf der EIA-485-Norm basiert, werden beispielsweise die Pins 3 und 8 von 9-poligen D-Sub-Steckern und -Dosen für die Datenleitung benutzt.

# Digitale und analoge Signalverläufe

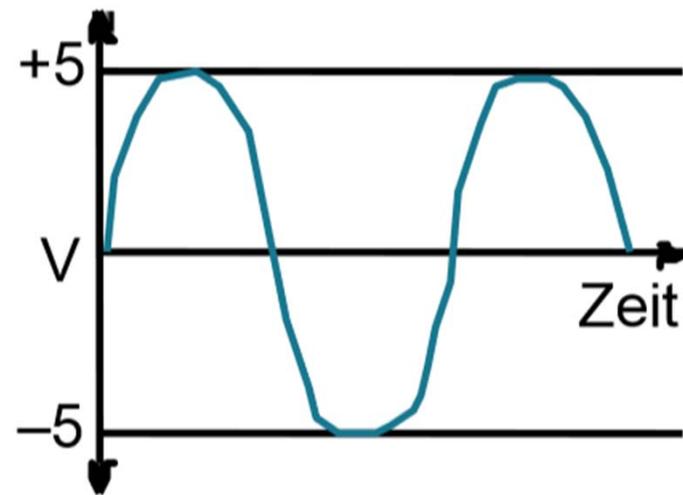
- **Digital:**

- Geht von diskreten Werten aus
- Von physikalischen Signalverläufen wird abstrahiert



- **Analog:**

- Werte sind kontinuierlich
- Darstellung real existierender Signalverläufe



# Merkmale Analog- und Digitaltechnik

Analogtechnik	Digitaltechnik
Die Maßzahl entspricht dem aktuellen Wert. Analogsignale sind wertkontinuierlich.	digitus - lateinisch der Finger. Es sind nur festgelegte Zählwerte zugelassen. Digitalwerte sind wertdiskret.
Genauigkeit wird von den Messgeräten und der Ablesegenauigkeit bestimmt.	Genauigkeit wird von der Zahl der Quantisierungsstufen bestimmt. Eine höhere Stufenzahl erfordert viel Aufwand. Es entstehen hohe Schrittzahlen (Bits).
Datenverarbeitung und -übertragung sind möglich. Der Bandbreitenbedarf wird vom verwendeten Modulationsverfahren bestimmt.	Datenverarbeitung und -übertragung sind möglich. Der Bandbreitenbedarf ist infolge der Abtastung und Übertragung digitaler Impulse oftmals größer.
Die Störanfälligkeit ist bei kleinen Signalamplituden relativ groß.	Die Störanfälligkeit zwei gut unterscheidbarer Amplitudenwerten ist klein. Störungen lassen sich einfacher erkennen und verhindern.
Analogschaltungen sollten alle Signaleigenschaften unverzerrt verarbeiten können.	In Binärschaltungen muss die Elektronik nur zwei Schaltzustände berücksichtigen.

Quelle: [http://elektroniktutor.oszkim.de/digitaltechnik/ad\\_vergl.html](http://elektroniktutor.oszkim.de/digitaltechnik/ad_vergl.html)

# Verwendung Analog- Digitaltechnik

- Computer verwenden intern nur digitale Signale
- Aber: Schnittstellen, Stromversorgungen und vor allem die Kommunikationseinheiten verwenden jedoch in der Regel analoge Komponenten zur Verarbeitung der vorliegenden Signale.

# Signal

- Darstellung einer Information durch physikalische, insbesondere elektrische Größen.
  - (Spannung, Strom ,Feldstärke).
- Die Information ist durch einen Parameter der Größe beschrieben.
  - (Amplitude, Frequenz, Impulsdauer u.a.)

# AD-Wandler (1)

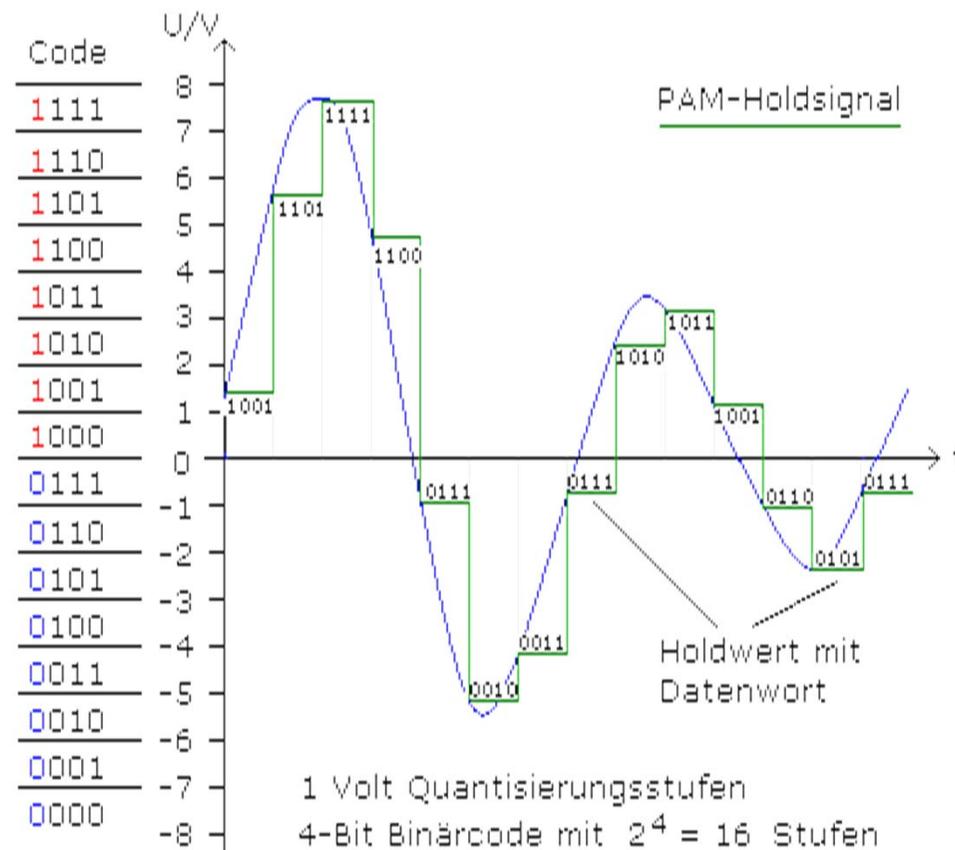
- Zweck: Umwandlung einer analogen Eingangsgröße in ein digitales Ausgangssignal
- Schritte der Digitalisierung:
  1. Zeitquantisierung (Abtastung) des Analogsignals
  2. Diskretisierung (Amplitudenquantisierung) des abgetasteten Signals
  3. Kodierung (Verschlüsselung) des Signals

## AD-Wandler (2)

- Die Abkürzung AD-Wandler steht für einen Analog-Digital-Wandler (teilweise auch ADU, Analog-Digital-Umsetzer; im Englischen ADC Analog-Digital-Converter). Dieser wandelt eine analoge Größe, meist eine Spannung, in einen digitalen Wert um.
- Die Auflösung, mit der die analoge Größe gemessen wird, bewegt sich zwischen 1 und 24 Bit. Den durch die Wandlung entstehenden Fehler zwischen dem tatsächlichen Wert und dem ausgegebenen (gewandelten) Wert nennt man Quantisierungsfehler. Er entsteht durch die unvermeidbare Rundung. Ferner entsteht durch die Nichtlinearität des Bauteils ebenfalls ein Fehler.
- Der Wandlungsvorgang benötigt immer eine gewisse Zeit, während der die Eingangsgröße konstant bleiben muss. Das gilt auch für den Flash-ADC, da nicht alle Komparatoren gleich schnell sind. Hierfür werden sogenannte 'Track and Hold' bzw. 'Sample and Hold' Schaltungen verwendet, welche das Eingangssignal "einfrieren", während die AD-Wandlung läuft.

# AD-Wandler (3)

- Ist die Amplitudenachse nach einem festgelegten Maßstab unterteilt, also quantisiert, dann befinden sich nach dem zeitlichen Abtasten eines Analogsignals die Amplitudenwerte in einer bestimmten Quantisierungsstufe. Folgen die Quantisierungsstufen einem bestimmten Code, wird das Ausgangssignal zum Digitalsignal.
- Der Quantisierungs- und Codiervorgang findet gemeinsam im Analog-Digital-Wandler statt.



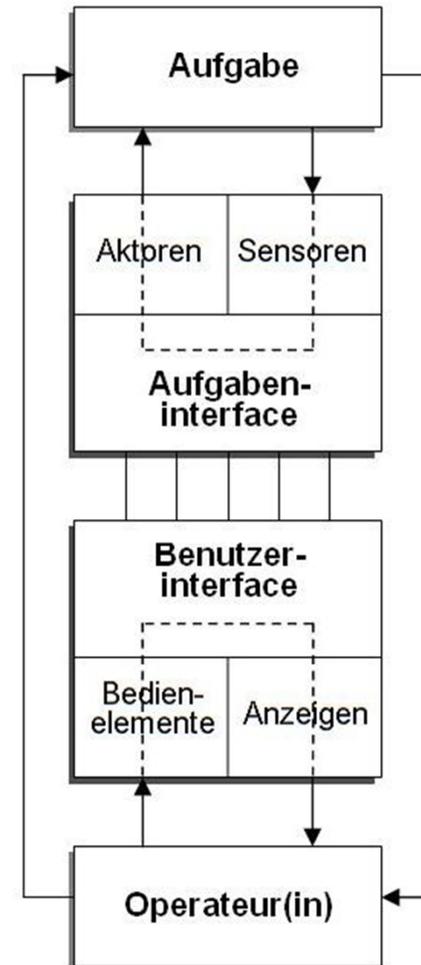
Quelle: [http://elektroniktutor.oszkim.de/digitaltechnik/digi\\_pict/analog5.gif](http://elektroniktutor.oszkim.de/digitaltechnik/digi_pict/analog5.gif)

# Human Machine Interface (1)

- Die Mensch-Maschine-Schnittstelle (MMS), das Human Machine Interface (HMI) oder Man Machine Interface (MMI), ist die Benutzerschnittstelle.
- Man findet sie überall dort, wo ein Menü auf einem Display dargestellt wird und über dieses Menü ein Dialog zwischen Mensch und Maschine erfolgt, der eine Interaktion bewirkt.
- Das Interface erlaubt dem Benutzer neben dem Bedienen unter Umständen auch das Beobachten der Anlagenzustände und das Eingreifen in den Prozess.
- Die Bereitstellung der Informationen erfolgt entweder über Bedienpulte mit Signallampen, Anzeigefeldern und Tastern oder per Software über ein Visualisierungssystem, das zum Beispiel auf einem Terminal läuft.
- Der Fahrerbereich eines Autos weist zahlreiche Benutzerschnittstellen auf, von den Befehlsgebern (Pedale, Lenkrad, Schalt- und Blinkerhebel usw.) bis zu den optischen Rückmeldungen der „Maschine“ Auto (Anzeigen für Geschwindigkeit, Reichweite, Radioprogramm, Navigationssystem usw.).

# Human Machine Interface (2)

- Erfolgen die bei der Aufgabenbewältigung erforderlichen Tätigkeiten schrittweise in mehreren Teilhandlungen, so ist davon auszugehen, dass das in jedem Schritt erreichte Ergebnis einer einzelnen Aktion mit dem gedanklich vorweggenommenen jeweils verglichen und im Folgeschritt gegebenenfalls korrigiert wird.
- Nach Sheridan (1987) hat man dabei verschiedene Formen von Wechselwirkungen zwischen den handelnden Personen (Operateure oder Operateurinnen) und Aufgaben zu unterscheiden: direkte Steuerung, manuelle Steuerung, Überwachungssteuerung und vollautomatische Steuerung.



Quelle: [http://de.wikipedia.org/wiki/Mensch-Maschine-System#mediaviewer/File:Sheridan\\_1987\\_hoch.jpg](http://de.wikipedia.org/wiki/Mensch-Maschine-System#mediaviewer/File:Sheridan_1987_hoch.jpg)

# Human Machine Interface (3)

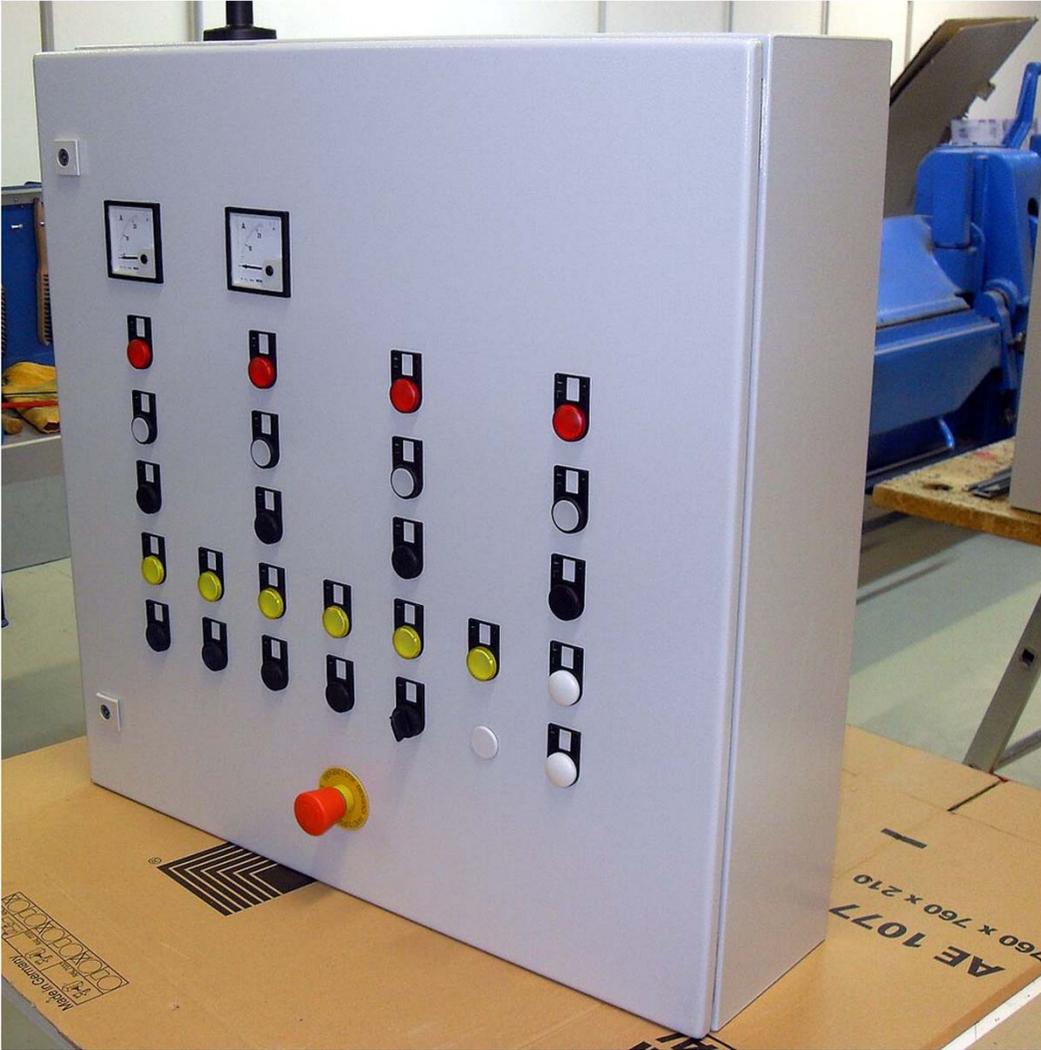
- Das Human-Machine-Interface bestimmt die Art und Weise wie Mensch und Maschine unmittelbar miteinander kommunizieren, wie der Mensch seine Anweisungen an die Maschine übermittelt und in welcher Form diese die Anweisungen ausführt und die Ergebnisse ausgibt.
- Systematisch betrachtet gehört die Benutzerschnittstelle zu den Mensch-Maschine-Systemen (MMS):  
*Mensch ↔ Mensch-Maschine-Schnittstelle ↔ Maschine.*
- Verschiedene Wissenschaften widmen sich dem Thema, etwa die Informatik, die Kognitionsforschung und die Psychologie.
- Damit eine Benutzerschnittstelle für den Menschen nutzbar und sinnvoll ist, muss sie auf seine Bedürfnisse und Fähigkeiten angepasst sein.

# Historische Mensch-Maschine-Schnittstelle einer Dampflokomotive



Quelle: wikipedia.org

# HMI in einer Zuckerfabrik



Quelle: wikipedia.org

# Moderne HMI eines ICEs



Quelle: wikipedia.org

# Character User Interface (CUI)

- In den Anfangsjahren der Computertechnik waren die Benutzereingaben textorientiert. Der Bedienende musste spezielle Kommandos eingeben, die vom Betriebssystem interpretiert wurden. Die Schnittstelle nannte sich Character User Interface (CUI).

<b>Benutzerschnittstelle</b>	<b>Akro.</b>	<b>Merkmale</b>
<b>Character User Interface</b>	<b>CUI</b>	<b>Textorientierte Benutzereingaben. Der Anwender gibt Kommandos in eine Textzeile ein. Beispiel: Command Line Interface (CLI).</b>
<b>Graphical User Interface</b>	<b>GUI</b>	<b>Die Benutzeroberfläche ist in bestimmte Bereiche unterteilt in denen Programme und Funktionen durch grafische Symbole abgelegt sind.</b>
<b>Voice User Interface</b>	<b>VUI</b>	<b>Die Benutzerschnittstelle arbeitet mit Spracherkennung und arbeitet mit Sprachsteuerung. Sie kann auch eine Sprachausgabe haben.</b>
<b>Natural User Interface</b>	<b>NUI</b>	<b>Weiterentwicklung der GUI und VUI kombiniert mit Gestenerkennung. Zweifingerbewegungen für die Ausführung von Befehlen auf einem Touchscreen.</b>

Quelle: <http://www.itwissen.info/definition/lexikon/human-machine-interface-HMI-Mensch-Maschine-Schnittstelle.html>

# Weitere User Interfaces (1)

- Da in den Anfangsjahren die Eingaben sehr aufwendig und nicht intuitiv waren, wurde die visualisierte Schnittstelle entwickelt, bei der die Eingaben manuell über Maus und Tastatur, auch über Touchscreens ausgeführt werden. Das sind grafische Benutzerschnittstellen, Graphical User Interface (GUI), die durch Sprach- oder Gesteneingaben, wie beim Natural User Interface (NUI) ergänzt werden können. Die Benutzeroberfläche ist dabei abhängig von der HMI-Kommunikation, für die sie optimiert ist.

## Weitere User Interfaces (2)

- Der Facettenreichtum der HMI-Schnittstelle zeigt sich an den Beispielen Computer, Handy, Fernseher, Braille-Terminal oder numerische Steuerung, deren Benutzeroberflächen vollkommen unterschiedlich sind.
- Neben den visuellen HMI-Schnittstellen gibt es noch akustische, die zunehmend an Bedeutung gewinnen. Bekannt sind die einkanalige akustische Ein- und Ausgabe von Sprache über das Headset. Daneben gibt es aber auch mehrkanalige Verfahren, bei denen der Nutzer nicht mit dem Equipment in Berührung kommt und sich darüber hinaus frei bewegen kann. Diese Verfahren erfordern mehrkanalige Auf- und Wiedergabetechniken sowie eine intelligente Signalverarbeitung.
- Um Barrierefreiheit für Behinderte zu gewährleisten, gibt es Benutzerschnittstellen mit Sprach-, Gesten- und auch mit Gedankensteuerung: Brain Computer Interfaces (BCI).

# Kennzeichnung (1)

- Weil von vielen Maschinen eine erhebliche Gefahr ausgehen kann, soll eine Mensch-Maschine-Schnittstelle so übersichtlich und eindeutig sein, dass sie selbst in Panik sicher bedient werden kann.
- An den Notausschalter werden deshalb besondere Anforderungen gestellt. Internationale Standards nach IEC/EN 60073 (VDE 0199), IEC/EN 60204-1 (VDE 0113 Teil 1) orientieren sich für Anzeigeleuchten an den vertrauten Farben von Lichtzeichenanlagen im Straßenverkehr und leiten davon die Farben der Bedienteile von Drucktastern ab.



## Kennzeichnung (2)

- Die Farben bei Anzeigesäulen von Maschinen sollen in der dargestellten Reihenfolge verwendet werden.

Farbe	Bedeutung	Bemerkung
Rot	gefährlicher Zustand	Warnung vor möglicher Gefahr oder Zuständen, die ein sofortiges Eingreifen erfordern
Gelb	anormaler Zustand	bevorstehender kritischer Zustand
Blau	Handeln zwingend	Handeln durch Bediener erforderlich
Grün	normaler Zustand	Anzeige sicherer Betriebsverhältnisse oder Freigabe eines Betriebsablaufs
Weiß	Neutral	Bestätigung oder jede andere Bedeutung, wenn nicht klar ist, welche der Farben ROT, GELB, BLAU oder GRÜN geeignet wäre

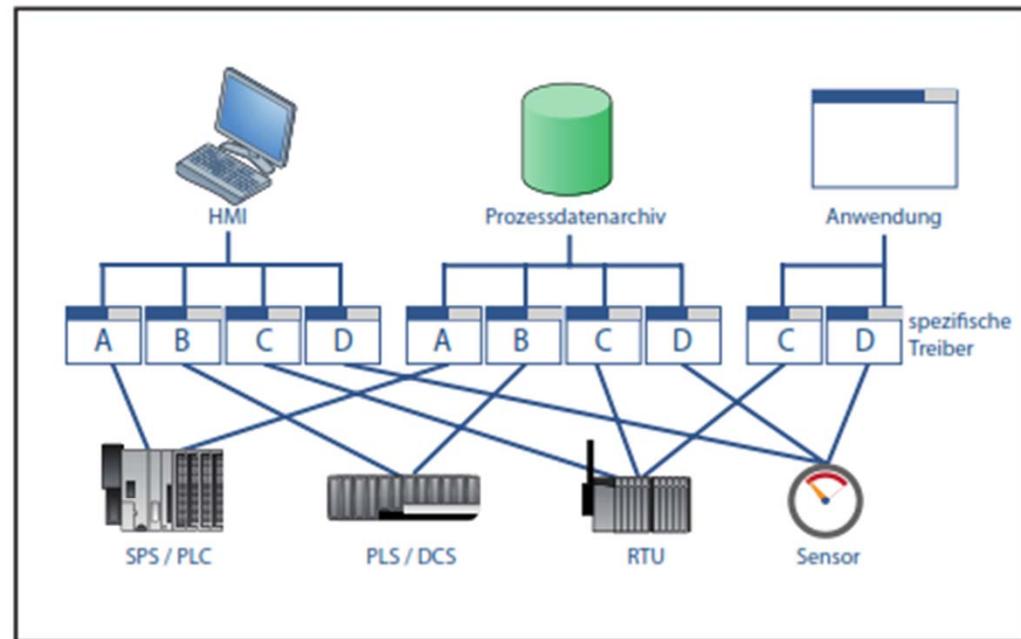
Quelle: wikipedia.org

Kommunikation/Schnittstellen

# OPC

# Problem der Konnektivität

Heutzutage wird Automatisierung in allen wichtigen Industriezweigen genutzt. Dabei werden häufig unterschiedliche spezifische Geräte, Steuerungssysteme und Anwendungen verwendet. Dadurch werden die Unternehmen mit der Frage konfrontiert, wie sich Daten sowohl zwischen diesen Komponenten als auch mit dem Rest des Betriebes austauschen lassen.



*Abbildung 1: Das Problem spezifischer Treiber: Jede Anwendung benötigt einen geräte- oder anwendungsspezifischen Treiber, damit sie mit den jeweiligen Geräten kommunizieren kann. Treiber können nicht von Anwendung zu Anwendung wiederverwendet werden, weil jede Anwendung ihre eigenen Datenformate verwendet.*

Quelle:

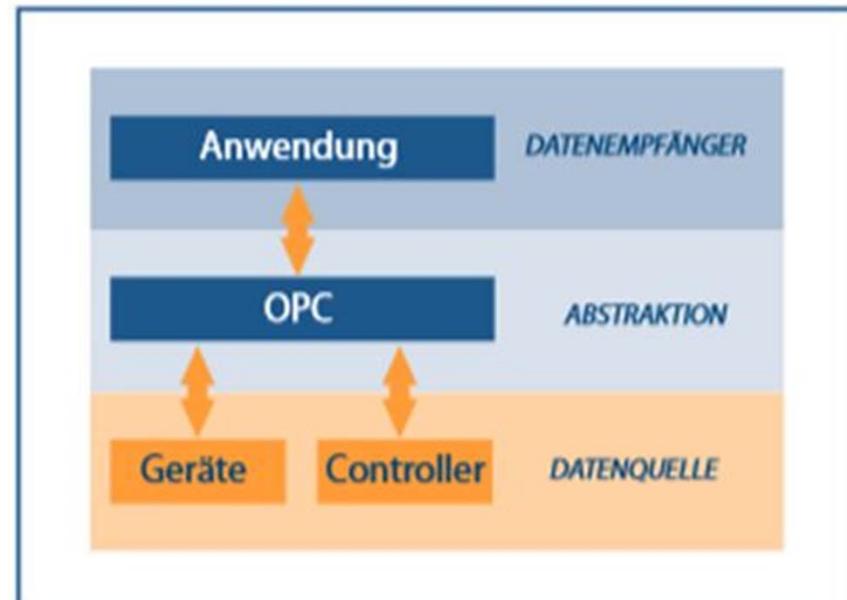
[http://www.wirautomatisierer.de/c/document\\_library/get\\_file?p\\_l\\_id=3269163&folderId=29426690&name=DLFE-25425912.pdf](http://www.wirautomatisierer.de/c/document_library/get_file?p_l_id=3269163&folderId=29426690&name=DLFE-25425912.pdf)

# Was ist OPC?

- Weltweit am stärksten verbreitete standardbasierte Datenkonnektivitätsmethode.
- Dient zur Kommunikation zwischen Geräten, Controllern und Anwendungen, unabhängig von den verschiedenen Treibern.
- Die Implementierungseigenschaften von Datenquellen und Datenempfänger werden abstrahiert. So lassen sich Daten zwischen ihnen austauschen, ohne dass die Quelle etwas über das Verbindungsprotokoll und die interne Datenorganisation des Empfängers wissen muss.

# Konzeptuelle Funktion der OPC-Kommunikation

- Man kann sich OPC als eine Art abstrakte Schicht vorstellen, die zwischen der Datenquelle und dem Datenempfänger liegt, sodass diese Daten austauschen können, ohne irgendetwas voneinander zu wissen.



Quelle:

[http://www.wirautomatisierer.de/c/document\\_library/get\\_file?p\\_l\\_id=3269163&folderId=29426690&name=DLFE-25425912.pdf](http://www.wirautomatisierer.de/c/document_library/get_file?p_l_id=3269163&folderId=29426690&name=DLFE-25425912.pdf)

# Funktion der OPC-Kommunikation

- Die Abstraktion wird erreicht, indem zwei OPC-Komponenten verwendet werden, der OPC-Client und der OPC-Server.
- Die Protokolle und/oder Schnittstellen der Quelle bzw. Empfänger werden weiterhin benötigt und kommunizieren nun mit einer der beiden OPC-Komponenten.
- Die OPC-Komponenten tauschen die Informationen miteinander aus.
- Daten können von der Anwendung zum Gerät übertragen werden, ohne dass diese direkt miteinander kommunizieren müssen.

# Vorteile der OPC-Konnektivität

1. Es können so viele Verbindungen wie nötig hergestellt werden. Es gibt in OPC keine Begrenzung für die Anzahl der hergestellten Verbindungen.
2. OPC ist heute weit verbreitet, für fast jedes Gerät ist ein OPC-Anschluss verfügbar.
3. Die OPC-fähigen Datenquellen können ausgetauscht, ausgewechselt oder aufgerüstet werden, ohne dass die Treiber der Datenempfänger aktualisiert werden müssen. Nur der OPC-Server für die entsprechende Datenquelle muss aktualisiert werden.
4. Die Geräte, Controller und Anwendungen können frei und somit optimal für das Projekt gewählt werden, ohne mögliche Kommunikationsprobleme zu berücksichtigen.



# OPC-Spezifikationen

- OPC-Verbindungen müssen aber nicht alle OPC-Spezifikationen unterstützen.
- Alle drei OPC-Spezifikationen nutzen die gleich OPC-Architektur. Dennoch müssen der Client und der Server dieselbe Spezifikation unterstützen, um die Übergabe der Daten und die Zusammenarbeit von Datenquelle und –Empfänger von einem zum anderen richtig zu koordinieren.
- Man sollte sich deshalb vor der Nutzung erkundigen, welche OPC-Spezifikationen eine OPC-Verbindung unterstützt.

# OPC UA

- Im Bereich OPC dreht sich alles um Interoperabilität und Standardisierung. Durch konventionelle OPC-Lösungen konnte das Problem der Interoperabilität zwischen verschiedenen Geräten auf Steuerungsebene gelöst werden, doch war der gleiche Standardisierungsgrad auch auf Unternehmensebene erforderlich.
- Klassische OPC-Standards basieren auf Microsoft DCOM, das auf dieser Ebene Angriffsmöglichkeiten bietet. Der drängende Ruf nach Einfachheit, maximaler Interoperabilität und Sicherheit führte zu einer einheitlichen Datenaustauschmethode für Teile der bestehenden OPC-Spezifikationen für DA, HDA, A&E und Security.
- Die OPC Unified Architecture stellt eine Erweiterung des erfolgreichen OPC-Datenaustauschprotokolls dar. Sie ermöglicht eine zuverlässige und sichere Datenerfassung und -modellierung sowie Kommunikation zwischen der Produktionsstätte und dem Unternehmen.

# Funktionen und Vorteile von OPC UA

- Plattformneutralität: Läuft auf jedem Betriebssystem
- Zukunftsfähig und in bestehende Systemen integrierbar
- Einfache Konfiguration und Wartung
- Dienstgestützte Technologie
- Höhere Transparenz
- Umfassendere Konnektivität
- Höhere Performance

# Was ist ein OPC-Server?

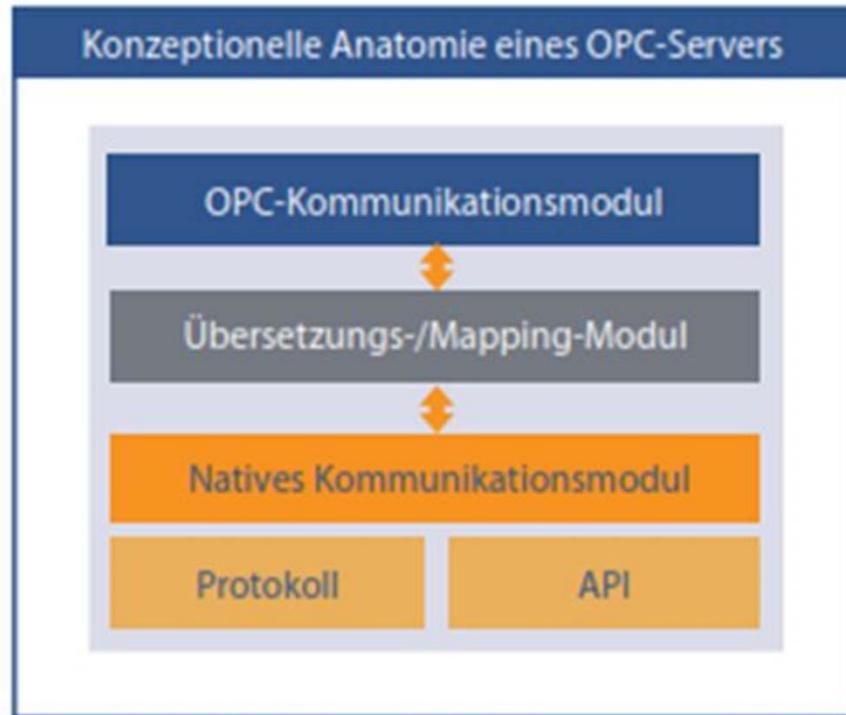
- OPC-Server = Software Anwendung, ein standardisierter Treiber, der eine oder mehrere Spezifikationen umsetzen kann.
- Server bezieht sich nicht auf den verwendeten Computertyp, sondern spiegelt die Beziehung zum Gegenstück, dem OPC-Client, wieder.
- Den Server bildet einen Übersetzer zwischen dem OPC-Bereich und der Kommunikationsschnittstelle der Datenquelle.
- OPC ist bidirektional, es kann sowohl von einer Datenquelle gelesen als auch in sie geschrieben werden.

# Womit kann ein OPC-Server kommunizieren?

- OPC-Server können praktisch mit jeder Datenquelle kommunizieren.
  - Deren Ausgabe kann über elektronische Mittel gelesen oder geschrieben werden-
  - z.B. SPS, elektr. Waagen, Datenbanken, Webseiten, ...
- Notwendig ist nur ein Server, der das native Protokoll/Schnittstelle des Gerätes unterstützt.

# Funktionen des OPC-Servers (1)

- Kommunikationsmodul
  - Verantwortlich für korrekte Kommunikation mit Client
  - Wichtig: Einhaltung der OPC-Spezifikationen, zur korrekten Datenübermittlung
- Übersetzungs-/Mapping-Modul
  - Interpretation der eingehenden Client Anfragen und Umsetzung in entsprechende native Anfragen
  - Bei effizienter Umsetzung: minimale Last der Datenquelle, bei maximalem Datendurchsatz

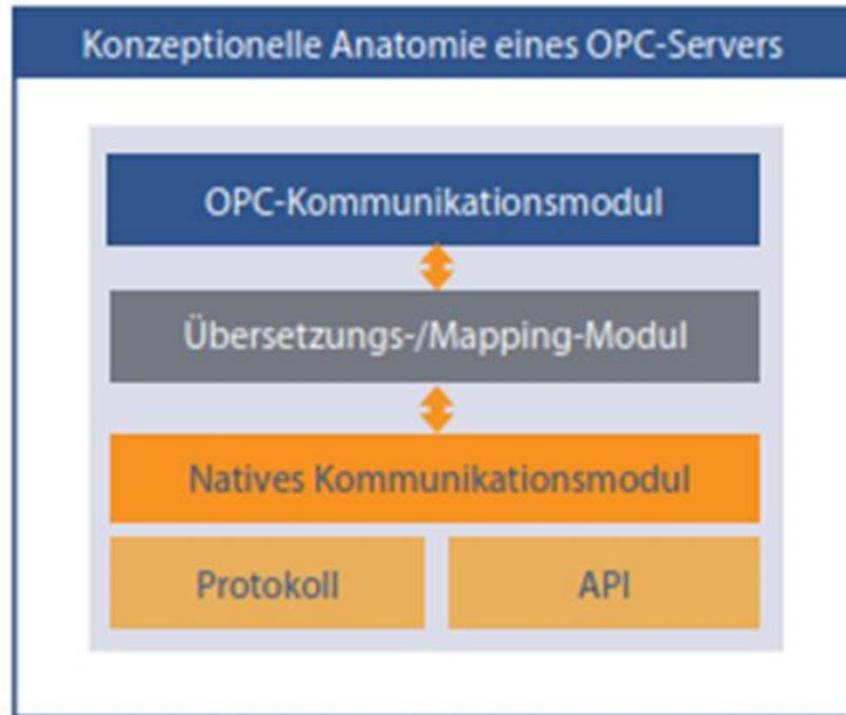


Quelle:

[http://www.wirautomatisierer.de/c/document\\_library/get\\_file?p\\_l\\_id=3269163&folderId=29426690&name=DLFE-25425912.pdf](http://www.wirautomatisierer.de/c/document_library/get_file?p_l_id=3269163&folderId=29426690&name=DLFE-25425912.pdf)

# Funktion des OPC-Servers (2)

- Natives Kommunikationsmodul
  - Auswahl der effizientesten Kommunikationsmethode mit der Datenquelle, über:
    - Natives Protokoll
    - Deren spezifischen Treiber mittels einer Anwendungsprogrammierschnittstelle (API)



Quelle:  
[http://www.wirautomatisierer.de/c/document\\_library/get\\_file?p\\_l\\_id=3269163&folderId=29426690&name=DLFE-25425912.pdf](http://www.wirautomatisierer.de/c/document_library/get_file?p_l_id=3269163&folderId=29426690&name=DLFE-25425912.pdf)

# Was ist ein OPC-Client?

- OPC-Client= Software Anwendung zur Kommunikation mit OPC-Konnektoren.
- Art des Datenaustausches ist definiert über bestimmte Spezifikationen, welche durch die OPC-Foundation definiert sind.

# Funktion des OPC-Client

- Repräsentieren einen Datenempfänger und werden von einer Anwendung verwendet, um mit jedem konformen Server im Netzwerk kommunizieren zu können.
- Initiieren und steuern die Kommunikation mit dem Server.
- Anfragen einer gegebenen Anwendung werden in eine OPC-äquivalente Anfrage umgesetzt und an den entsprechenden OPC-Server zur Weiterverarbeitung gesendet.
- Daten vom OPC-Server, werden in das native Format der Anwendung übersetzt.

# Kommunizieren mit mehreren Geräten

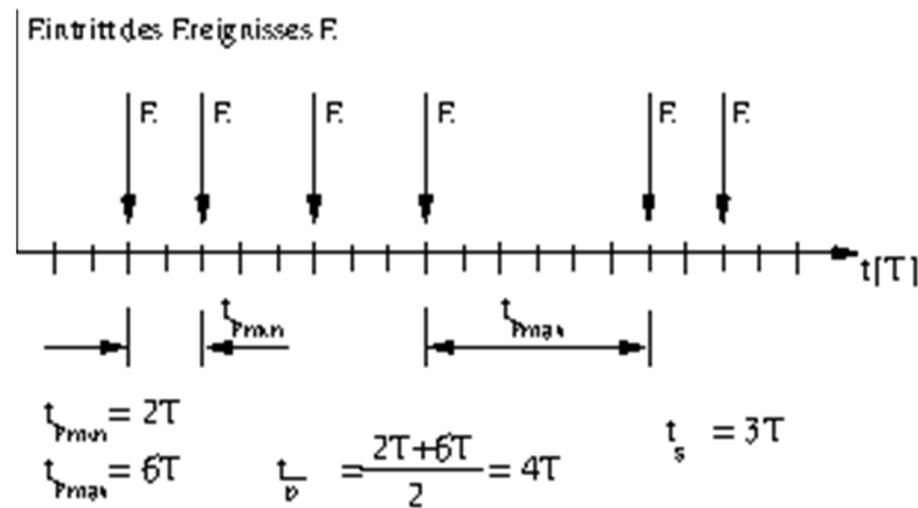
- Auslegungsmäßig kommuniziert der Client mit dem Server, nicht mit Endgeräten.
- Gleichzeitige Kommunikation mit mehreren OPC-Servern ist möglich, Client kann Daten von mehreren Datenquellen über die jeweiligen Server lesen und in die selbigen schreiben.
- Es besteht keine Beschränkung, zu wie vielen Servern ein Client eine Verbindung herstellen kann.

# Schritthaltende Verarbeitung und Echtzeitbetrieb

- Unter schritthaltender Verarbeitung bzw. Echtzeitbetrieb versteht man die fristgerechte Bearbeitung von Anforderungen aus einem technischen Prozess. Was aus Sicht des technischen Prozesses „Anforderungen“ sind, sind aus Sicht der Steuerung Ereignisse. Die Steuerung muss also Ereignisse fristgerecht bearbeiten.
- Können die Ereignisse durch die Steuerung nicht innerhalb der durch den technischen Prozess gestellten Zeitschranken bearbeitet werden, sind die Folgen möglicherweise katastrophal. Daher ist es notwendig, die zeitlichen Anforderungen und die zeitlichen Kenndaten bezüglich der Steuerung zu kennen.

# Echtzeitbedingungen

- Der technische Prozess stellt Anforderungen unterschiedlicher Art an die Echtzeitsteuerung. Alle Anforderungen müssen aber von der Steuerung bearbeitet werden können, auch wenn sämtliche Ereignisse zu einem Zeitpunkt auftreten (erste Echtzeitforderung).
- Das zeitliche Auftreten der Ereignisse wird in einer Anforderungsfunktion dargestellt. Dazu werden die Anforderungen (Sichtweise des Steuerungs- bzw. Betriebssystems) bzw. Ereignisse (Sichtweise des technischen Prozess) in Form von Pfeilen über der Zeit aufgetragen.



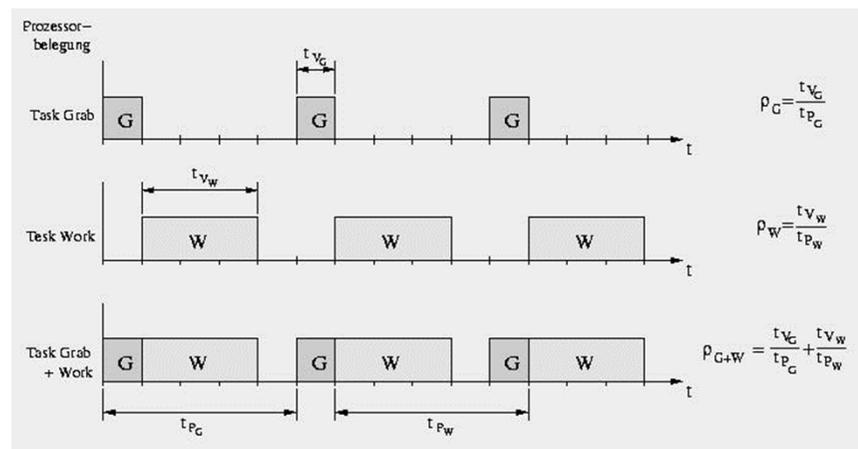
Quelle: <https://ezs.kr.hs-niederrhein.de/lectures/ezs/html/x306.html>

# Echtzeitbedingungen

- Untersucht man eine Echtzeit Objektverfolgung, ergeben sich zwei Anforderungen, die erfüllt werden sollten.
  - Es soll gewährleistet werden, dass eine gewisse Anzahl von Bildern pro Sekunde verarbeitet wird. Hierbei ist wichtig, dass zwischen zwei Bildern nur eine gewisse Zeit zur Verfügung steht, in der ein Bild bearbeitet werden kann. Danach muss mit der Bearbeitung des nächsten Bildes begonnen werden, und die Bearbeitung des vorhergehenden Bildes abgeschlossen sein.
  - Zum anderen muss das System so viel Rechenleistung zur Verfügung stellen können, dass es alle ihm gestellten Aufgaben bearbeiten kann.
- In diesen Punkten lassen sich die beiden Echtzeitbedingungen erkennen.

# Auslastung

- Abhängig von der Auftrittshäufigkeit eines Ereignisses und der damit anfallenden Verarbeitungszeit ist der Rechnerkern (Prozessor) ausgelastet.
  - Ein Realzeitsystem muss in der Lage sein, die auftretenden Anforderungen in der Summe bearbeiten zu können (oft auch als Forderung nach *Gleichzeitigkeit* bezeichnet). Mathematisch bedeutet dieses, dass die Gesamtauslastung kleiner als 100% sein muss.
  - Die Gesamtauslastung ergibt sich aus der Summe der Auslastungen der einzelnen Jobs.
- Die Auslastung ist eine notwendige, aber keine hinreichende Bedingung!



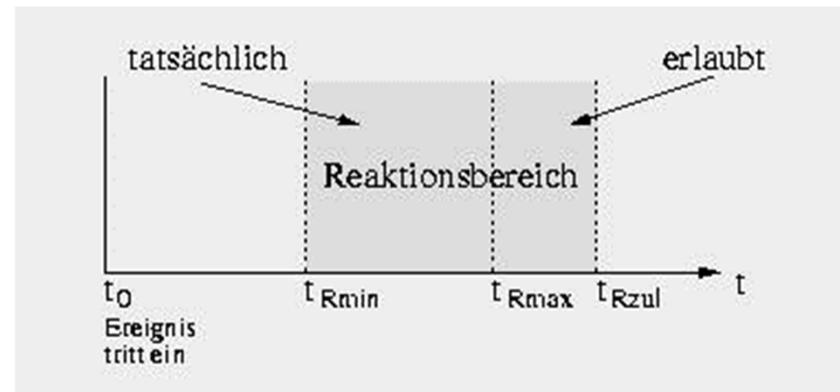
Quelle: <https://ezs.kr.hs-niederrhein.de/lectures/ezs/html/x306.html>

# Pünktlichkeit (1)

- Aus dem technischen Prozess kommen immer wieder Anforderungen an die Echtzeitsteuerung. Aufgabe der Steuerung ist es, *alle* Anforderungen *zeitgerecht* zu bearbeiten. *Alle* bedeutet in diesem Falle, dass der Rechner leistungsfähig sein muss, um mit der vorgegebenen Last fertig zu werden. *Zeitgerecht* bedeutet, dass der Rechner die Anforderungen *rechtzeitig* bzw. *pünktlich* bearbeitet.
- Das wesentliche Kriterium für ein Realzeitsystem ist *Pünktlichkeit* in Abgrenzung zu *Schnelligkeit*.
- Allerdings lassen sich Anforderungen an Pünktlichkeit mit schnellen Systemen leichter erfüllen als mit langsamen. Prinzipiell muss ein Realzeitsystem eine Aufgabe in einem vorgegebenen Zeitfenster erfüllt haben, wie es das tut, ist dabei unwesentlich.

## Pünktlichkeit (2)

- Unter Pünktlichkeit wird hier also verstanden, dass die Aufgabe
  - nicht vor einem weiteren spezifizierten Zeitpunkt  $t_{Dmin}$  erledigt ist und
  - bis zu einem spezifizierten Zeitpunkt  $t_{Dmax}$  erledigt wird (Rechtzeitigkeit).
- Die Forderung, dass eine Ausgabe nicht vor einem bestimmten Zeitpunkt durchgeführt wird, fehlt oft und ist in den übrigen Fällen vielfach trivial zu erfüllen. Die Forderung nach Rechtzeitigkeit stellt dagegen hohe Anforderungen an das System und damit letztlich auch an den Ingenieur bzw. Informatiker, der das System konzipiert und realisiert.



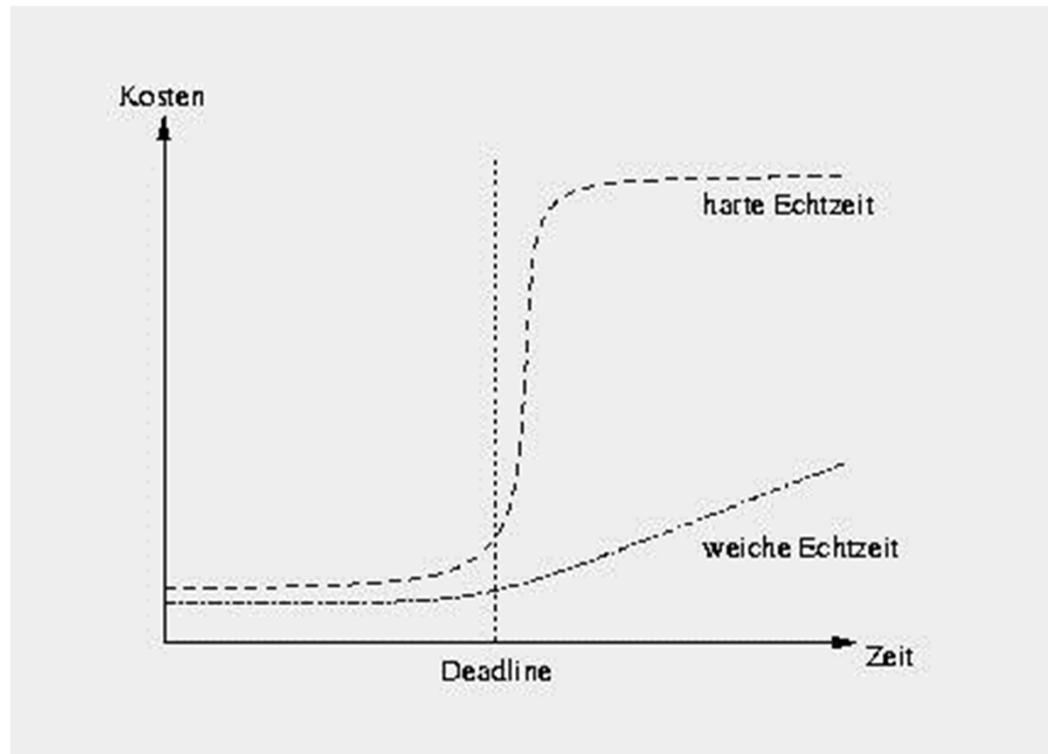
Quelle: <https://ezs.kr.hs-niederrhein.de/lectures/ezs/html/x306.html>

# Harte und weiche Echtzeit (1)

Die Forderung nach Pünktlichkeit ist nicht bei jedem System gleich stark. Während das nicht rechtzeitige Absprengen der Zusatz tanks bei einem Raumschiff zum Verlust desselbigen führen kann, entstehen durch Verletzung der Pünktlichkeitsbedingung bei einem Multimediasystem allenfalls Komfortverluste. Vielfach findet man den Begriff **harte Echtzeit**, wenn die Verletzung der Realzeitanforderungen katastrophale Folgen hat und daher nicht toleriert werden kann. Man findet den Begriff der **weichen Echtzeit**, wenn die Verletzung der Realzeitanforderungen verkraftet werden kann. Allerdings kann man nicht ausschließlich zwischen „harter“ und „weicher“ Echtzeit unterscheiden, **sondern jegliche Abstufung** zwischen den beiden Extremen (weich und hart, soft und hard) ist möglich. Die Graduation hängt von der Aufgabenstellung ab.

## Harte und weiche Echtzeit (2)

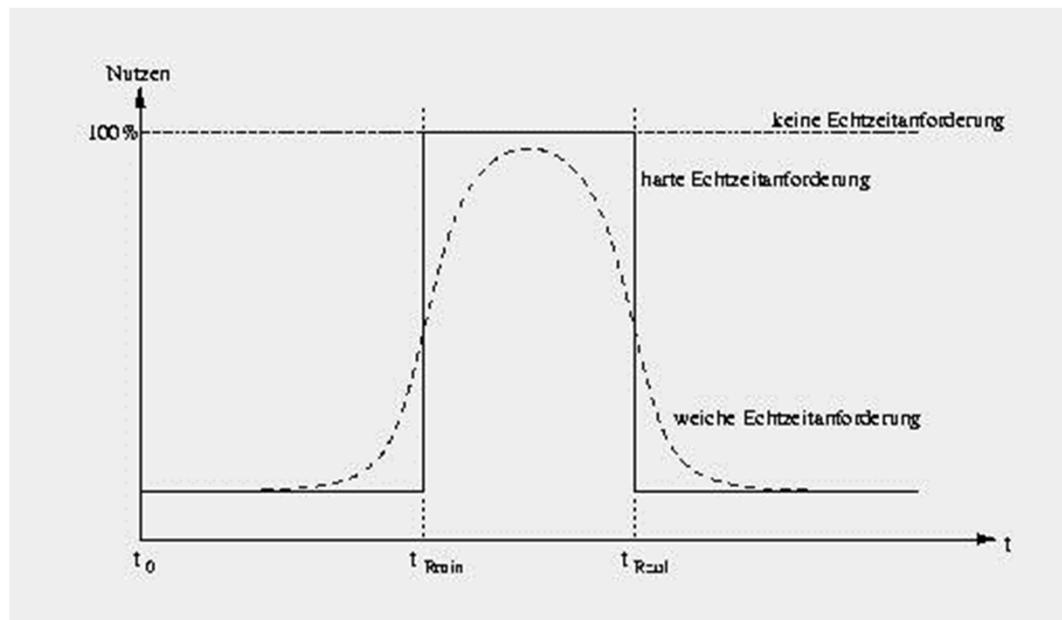
Die Unterschiede bei der Forderung nach Pünktlichkeit werden oft anhand einer Kostenfunktion verdeutlicht. Bei den sogenannten *weichen Echtzeitsystemen* bedeutet das Verletzen der Realzeitbedingung einen leichten Anstieg der Kosten. Bei den sogenannten *harten Echtzeitsystemen* steigen jedoch die Kosten durch die Verletzung der Realzeitbedingung massiv an.



Quelle: <http://maurer-tech.com/Debian/v4/c81.html>

## Harte und weiche Echtzeit (3)

Eine andere Art der Darstellung ist die sogenannte **Benefit function** (Nutzenfunktion). Hier wird der *Nutzen* der Reaktion des Realzeitrechners auf das Ereignis über die Zeit aufgetragen. Bei Systemen ohne Echtzeitanforderungen ist der Nutzen unabhängig von dem Zeitpunkt der erfolgten Reaktion, bei einem absolut harten Echtzeitsystem ist nur dann ein Nutzen vorhanden, wenn die Reaktion innerhalb des aufgestellten Zeitfensters erfolgt. Dazwischen ist jegliche Abstufung an Realzeitanforderungen möglich. Hier kann man (je nach Graduation) von weichen Realzeitanforderungen sprechen.



Quelle: <http://maurer-tech.com/Debian/v4l/c81.html>

Regelung und Steuerung

# OLE FOR PROCESS CONTROL

# Problem der Konnektivität

- Heutzutage wird Automatisierung in allen wichtigen Industriezweigen genutzt. Dabei werden häufig unterschiedliche spezifische Geräte, Steuerungssysteme und Anwendungen verwendet.
- Dadurch werden die Unternehmen mit der Frage konfrontiert, wie sich Daten sowohl zwischen diesen Komponenten als auch mit dem Rest des Betriebes austauschen lassen.

# Was ist OPC?

- Weltweit am stärksten verbreitete standardbasierte Datenkonnektivitätsmethode
- Dient zur Kommunikation zwischen Geräten, Controllern und Anwendungen, unabhängig von den verschiedenen Treibern.
- Die Implementierungseigenschaften von Datenquellen und Datenempfänger werden abstrahiert. So lassen sich Daten zwischen ihnen austauschen, ohne das die Quelle etwas über das Verbindungsprotokoll und die interne Datenorganisation des Empfängers wissen muss.

# Konzeptuelle Funktion der OPC-Kommunikation

- Man kann sich OPC als eine Art abstrakte Schicht vorstellen, die zwischen der Datenquelle und dem Datenempfänger liegt, sodass diese Daten austauschen können, ohne irgendetwas voneinander zu wissen.

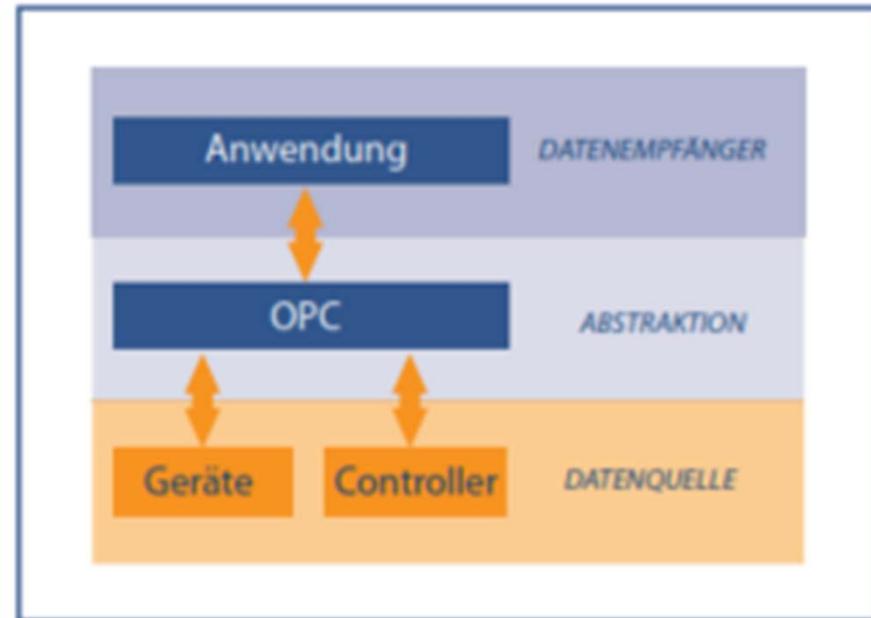


Abbildung 2: OPC dient als abstrakte Schicht zwischen Datenquellen und Datenempfänger – und ermöglicht so die Kommunikation, ohne dass die eine Seite etwas über das native Protokoll der anderen wissen muss.

# Konzeptuelle Funktion der OPC-Kommunikation

- Die Abstraktion wird erreicht, indem zwei OPC-Komponenten verwendet werden, der OPC-Client und der OPC-Server.
- Die Protokolle und/oder Schnittstellen der Quelle bzw. Empfänger werden weiterhin benötigt und kommunizieren nun mit einer der beide OPC-Komponenten.
- Die OPC-Komponenten tauschen die Informationen miteinander aus.
- Daten können von der Anwendung zum Gerät übertragen werden, ohne dass dieses direkt miteinander kommunizieren müssen.

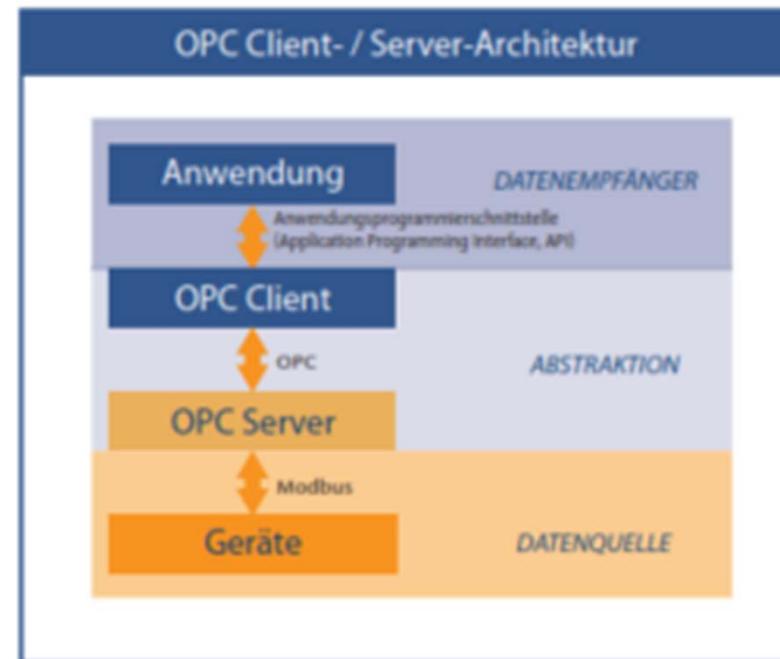


Abbildung 3: OPC-Client/Server-Architektur – bei näherer Betrachtung der OPC-Abstraktionsschicht werden zwei Komponenten deutlich: ein OPC-Client und ein OPC-Server. OPC-Spezifikationen legen den Datenaustausch zwischen diesen beiden Komponenten fest.

# Vorteile der OPC-Konnektivität

- Es können so viele Verbindungen wie nötig hergestellt werden. Es gibt in OPC keine Begrenzung für die Anzahl der hergestellten Verbindungen.
- OPC ist heute weit verbreitet, für fast jedes Gerät ist ein OPC-Anschluss verfügbar.
- Die OPC-fähige Datenquellen können ausgetauscht, ausgewechselt oder aufgerüstet werden, ohne dass die Treiber der Datenempfänger aktualisiert werden müssen. Nur der OPC-Server für die entsprechende Datenquelle muss aktualisiert werden.
- Die Geräte, Controller und Anwendungen können frei, und somit optimal für das Projekt, gewählt werden, ohne sich über mögliche Kommunikationsprobleme miteinander Gedanken zu machen.

# Unterstützte Daten

- Die geläufigsten Automatisierungsdaten, die übertragen werden, verteilen sich auf drei große Datenkategorien.
- Jede Kategorie unterstützt verschiedene Datentypen. Dadurch ergeben sich einige Herausforderungen, bei der Arbeit mit diesen Datenkategorien
- Diesen sind ebenso drei OPC-Spezifikationen zugeordnet, die unabhängig spezifiziert werden.
  - Echtzeit-Daten                      OPC Data Access (OPC DA)
  - Verlaufsdaten                        OPC Historical Data Access (OPC HDA)
  - Alarm- und Ereignisdaten        OPC Alarms & Events (OPC A&E)

# OPC-Spezifikationen

- OPC-Verbindungen müssen aber nicht alle der OPC-Spezifikationen unterstützen.
- Am häufigsten liegt die Verwendung von Echtzeitdaten vor.
- Alle drei OPC-Spezifikationen nutzen die gleich OPC-Architektur. Dennoch müssen der Client und der Server dieselbe Spezifikation unterstützen, um die Übergabe der Daten und die Zusammenarbeit von Datenquelle und -empfänger von einem zum anderen richtig zu koordinieren.
- Man sollte sich deshalb vor der Nutzung erkundigen, welche OPC-Spezifikationen eine OPC-Verbindung unterstützt.

Regelung und Steuerung

# OPC-SERVER

# Was ist ein OPC-Server?

- OPC-Server = Software Anwendung, ein standardisierter Treiber, der eine oder mehrere Spezifikationen umsetzen kann.
- Server bezieht sich nicht auf den verwendeten Computertyp, sondern spiegelt die Beziehung zum Gegenstück, dem OPC-Client, wieder.
- Den Server bildet einen Übersetzer zwischen dem OPC-Bereich und der Kommunikationsschnittstelle der Datenquelle.
- OPC ist bidirektional, es kann sowohl von einer Datenquelle gelesen als auch in sie geschrieben werden.

# Womit kann ein OPC-Server kommunizieren?

- OPC-Server können praktisch mit jeder Datenquelle kommunizieren
  - Deren Ausgabe über elektronische Mittel gelesen oder geschrieben werden kann
  - z.B. SPS, el. Waagen, Datenbanken, Webseiten, ...
- Notwendig ist nur ein Server, der das native Protokoll/Schnittstelle des Gerätes unterstützt

# Funktionen des OPC-Servers

- Kommunikationsmodul
  - Verantwortlich für korrekte Kommunikation mit Client
  - Wichtig: Einhaltung der OPC-Spezifikationen, zur korrekten Datenübermittlung
- Übersetzungs-/Mapping-Modul
  - Interpretation der eingehenden Client Anfragen und Umsetzung in entsprechende native Anfragen
  - Bei effizienter Umsetzung: minimale Last der Datenquelle, bei maximalem Datendurchsatz

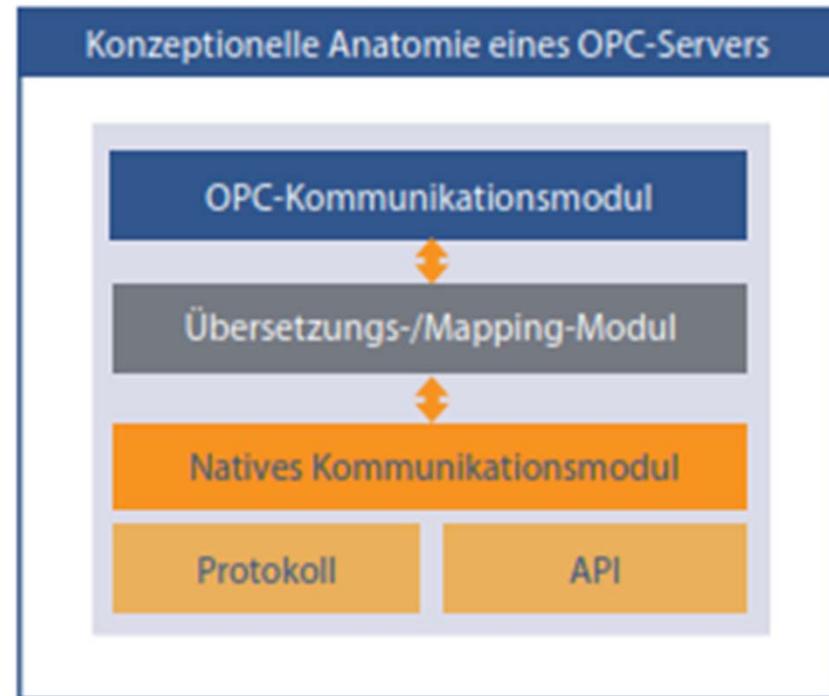


Abbildung 4: Konzeptionelle Anatomie eines OPC-Servers – konzeptionell kann ein OPC-Server in drei Module gegliedert werden: das OPC-Kommunikationsmodul, das Übersetzungs-/Mapping-Modul und das native Kommunikationsmodul.

# Funktionen des OPC-Servers

- Natives Kommunikationsmodul
  - Auswahl der effizientesten Kommunikationsmethode mit der Datenquelle, über:
    - Natives Protokoll
    - Deren spezifischen Treiber mittels einer Anwendungsprogrammierschnittstelle (API)

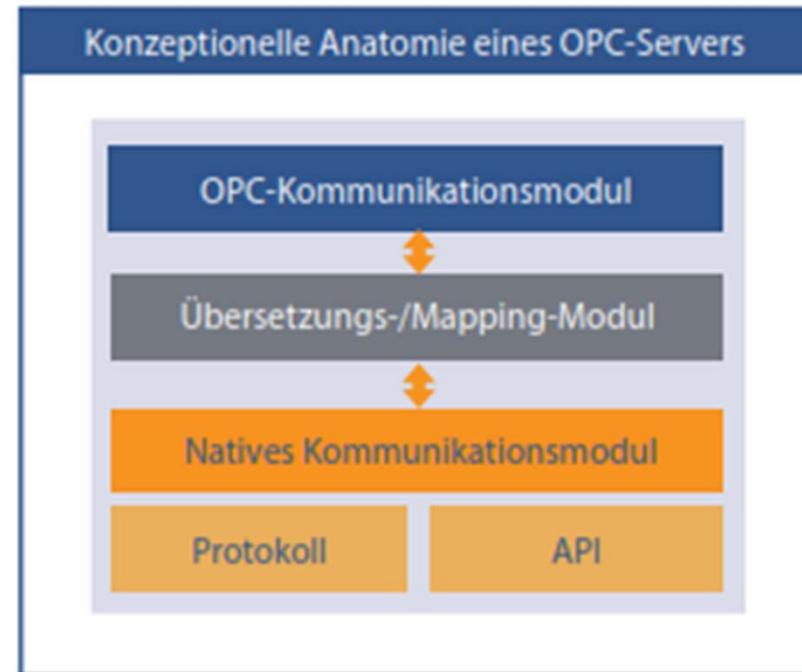


Abbildung 4: Konzeptionelle Anatomie eines OPC-Servers – konzeptionell kann ein OPC-Server in drei Module gegliedert werden: das OPC-Kommunikationsmodul, das Übersetzungs-/Mapping-Modul und das native Kommunikationsmodul.

# Verwendung von mehreren Servern

- OPC-Server kommunizieren nicht direkt miteinander
- spezielle Data Manager: Entwickelt um die Server-zu-Server-Kommunikation zu erleichtern

Regelung und Steuerung

# OPC-CLIENT

# Was ist ein OPC-Client?

- OPC-Client= Software Anwendung, zur Kommunikation mit OPC-Konnektoren
- Art des Datenaustausches ist definiert über bestimmte Spezifikationen, welche durch die OPC-Foundation definiert wurden

# Funktionen des OPC-Clients

- Repräsentieren einen Datenempfänger, werden von einer Anwendung verwendet um mit jedem konformen Server im Netzwerk kommunizieren zu können
- Initiieren und steuern die Kommunikation mit dem Server
- Anfragen einer gegebenen Anwendung werden in eine OPC-äquivalente Anfrage umgesetzt und an den entsprechenden OPC-Server zur Weiterverarbeitung gesendet
- Daten vom OPC-Server, werden in das native Format der Anwendung übersetzt

# Kommunikation mit mehreren Geräten

- Auslegungsgemäß kommuniziert der Client mit dem Server, nicht mit Endgeräten
- Gleichzeitige Kommunikation mit mehreren OPC-Servern ist möglich, Client kann Daten von mehreren Datenquellen über die jeweiligen Server lesen und in sie schreiben
- Es besteht keine Beschränkung zu wie vielen Servern ein Client eine Verbindung herstellen kann

# Funktionen des OPC-Clients

- Kommunikationsmodul
  - Verantwortlich für korrekte Kommunikation mit Server
  - Wichtig: korrektes Verhalten beim herstellen und trennen einer Verbindung, um Instabilität des Servers zu vermeiden
- Übersetzungs-/Mapping-Modul
  - Bidirektionale Übersetzung von Informationen

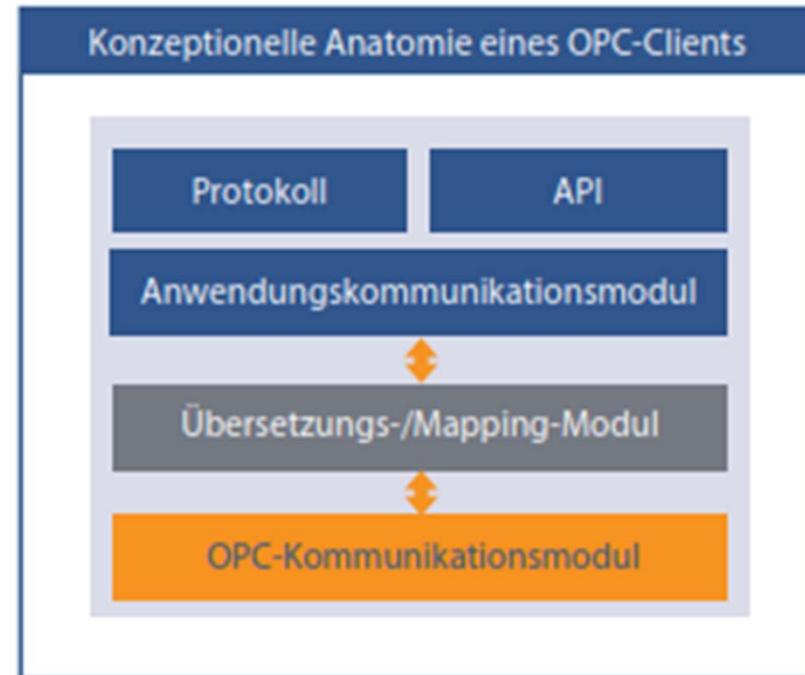


Abbildung 3: Konzeptionelle Anatomie eines OPC-Clients – da der OPC-Client ein Spiegelbild des OPC-Servers ist, kann man ihn sich ebenfalls als aus drei Modulen bestehend vorstellen: aus der nativen Anwendungskommunikation, dem Übersetzungs-/Mapping-Modul und dem Kommunikationsmodul.

# Funktion des OPC-Clients

- Anwendungs-kommunikationsmodul
  - Funktioniert innerhalb einer speziellen Anwendung
  - Nutzt wenige Aufrufe der API, zur Datenübergabe
  - Oder Anwendung kommuniziert über ein Protokoll, wenn ein solchen unterstützt wird

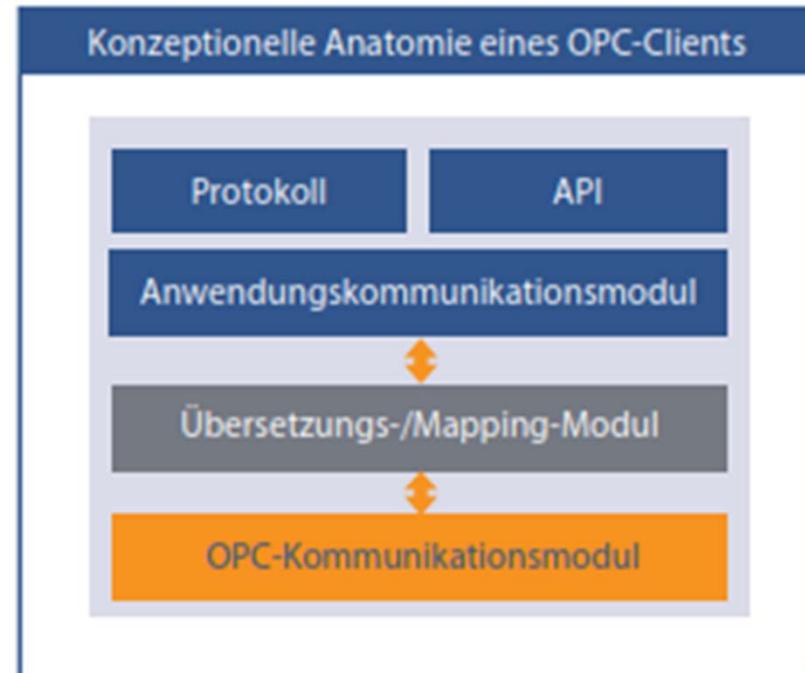


Abbildung 3: Konzeptionelle Anatomie eines OPC-Clients – da der OPC-Client ein Spiegelbild des OPC-Servers ist, kann man ihn sich ebenfalls als aus drei Modulen bestehend vorstellen: aus der nativen Anwendungskommunikation, dem Übersetzungs-/Mapping-Modul und dem Kommunikationsmodul.