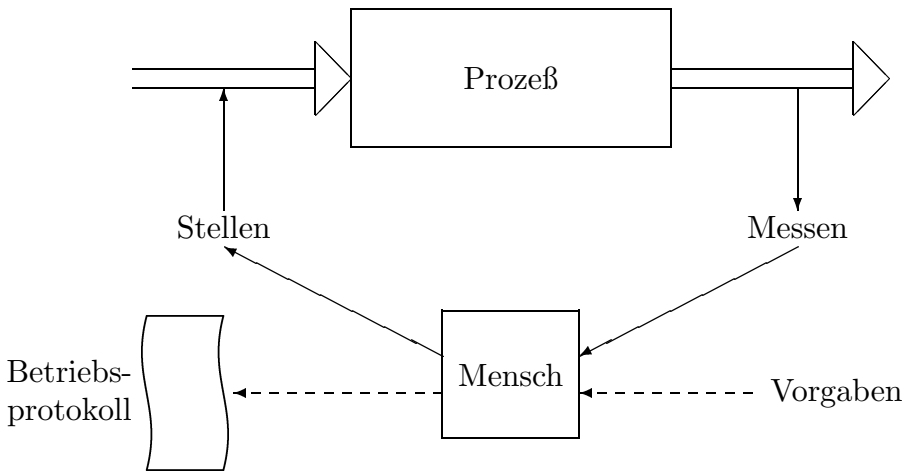
E Struktur von Prozessrechensystemen

E.1 Prozesskopplungsarten

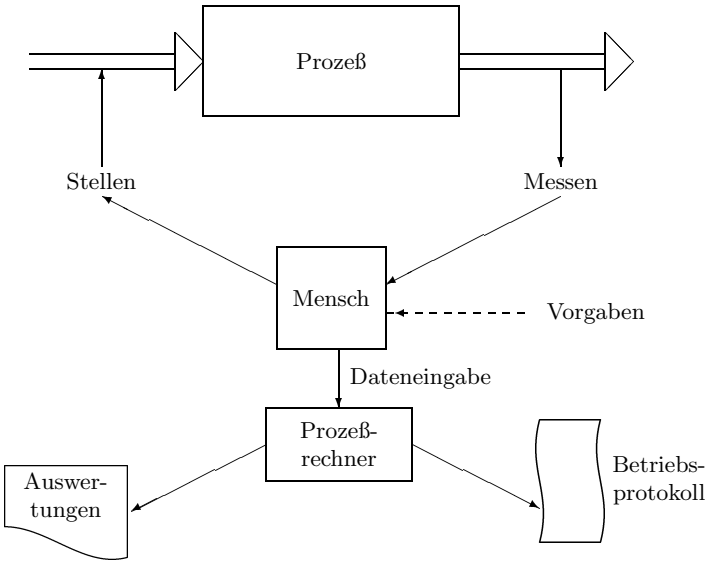
- Kein Rechnereinsatz
- Indirekte Prozesskopplung (off-line-Betrieb)
- Dialogbetrieb (in-line-Betrieb)
- Direkte Prozesskopplung (on-line-Betrieb):
  - open-loop
  - closed-loop

1 Kein Rechnereinsatz



- Die Messgeräte werden vom Bedienungspersonal selbst abgelesen.
- Unter Hinzunahme weiterer Anweisungen von außen (z.B. Vorgaben der Betriebsleitung zum Produktumfang) bedient das Bedienungspersonal entsprechend die Stellorgane.
- Über den Prozessverlauf wird, ebenfalls vom Bedienungspersonal, ein Protokoll erstellt.
- Eine eigene Abteilung ermittelt daraus weitere Anweisungen für den künftigen Prozessverlauf, die dem Bedienungspersonal als schriftliche Unterlagen wieder mitgeteilt werden.
- Ein Rechner wird hierbei nicht eingesetzt.

2 Indirekte Prozesskopplung (off-line-Betrieb)

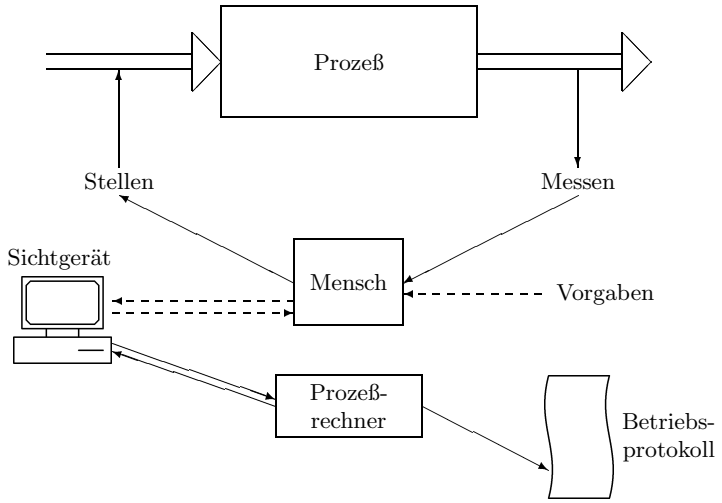


- Sowohl das Stellen als auch das Ablesen der Messungen erfolgt hier noch durch das Bedienungspersonal.
- Zusätzlich wird jedoch ein Prozessrechner eingesetzt, der die Auswertungen der abgelesenen und neu hinzugekommenen Informationen vornimmt.
- Die Daten werden vom Bedienungspersonal auf Datenträgern (z.B. Magnetbänder, -platten) abgespeichert und vom Rechner zu einem späteren Zeitpunkt sequentiell verarbeitet.
- Die daraus resultierenden Ergebnisse erhält das Bedienungspersonal wieder in Form von Ausdrucken bzw. Protokollen.
- Rechner und Prozess sind hierbei zeitlich und physikalisch entkoppelt.

- Typische Aufgaben dieser Kopplungsart sind:
  - ➤ statistische Auswertungen von Versuchsreihen
  - ➤ Berechnungen der Verluste aus gemessenen Strom- und Spannungsmessungen in einem Elektrizitätsnetz
  - ➤ Zuverlässigkeits- und Qualitätsuntersuchungen

3 Dialogbetrieb (in-line-Betrieb)

1



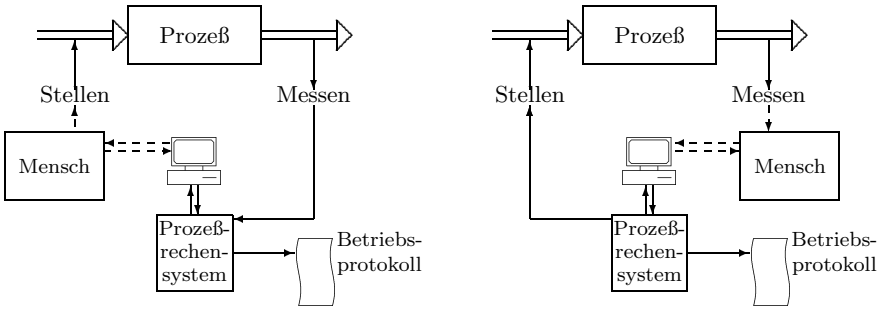
- Ein Sichtgerät für das Bedienungspersonal ermöglicht die Dateneingabe und -ausgabe von und zum Rechner.
- Rechner und der Prozess sind zeitlich gekoppelt.
- Da die Eingriffe in den Prozess hierbei immer noch auf der Seite des Menschen liegen, können Rechnerstörungen in diesem Fall nicht zu gefährlichen Prozesszuständen führen.
- Die Erfahrung des Personals kann in das Ablaufgeschehen mit einbezogen werden.
- Beispiele:
  - Platzbuchungsanlagen,
  - Auskunftssysteme,
  - Fertigungssteuerungen.

4 Direkte Prozesskoplung (on-line-Betrieb)

- Der Rechner wird in einem der beiden Bereiche (Stellen oder Messen) oder in beiden direkt durch Leitungen mit dem Prozess gekoppelt.
- Dies erfordert besondere Anforderungen an die Zuverlässigkeit der Soft- und Hardware des Rechners.
- Das Echtzeit-Betriebssystem muss im Millisekunden- manchmal auch im Mikrosekundenbereich reagieren können.

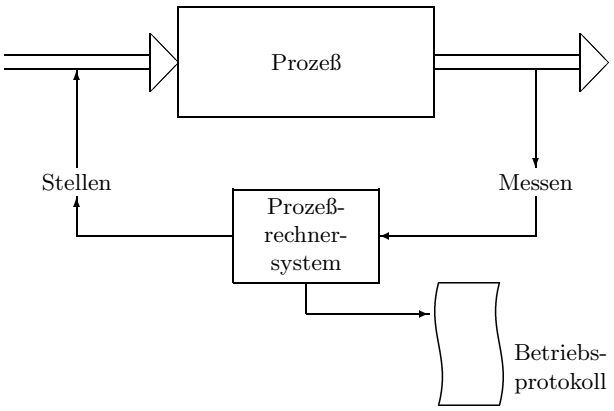
On-line-open-loop-Betrieb

1



- Nur eine physikalische Verbindung vom Prozess zum Rechner
- Der Eingriff des Personals erfolgt entweder auf der Seite der Stellgeräte oder aber auf der Seite der Messgeräte.

- On-line-closed-loop-Betrieb



- Rechner auf der Seite der Stellgeräte als auch auf der Seite der Messgeräte direkt mit dem Prozess gekoppelt.
- Überwachung des Gesamtsystems erfolgt weiterhin durch den Menschen.
- Es wird jedoch nur im Störungsfall in den Prozessablauf eingegriffen.

E.2 Sicherheit und Zuverlässigkeit

- **Sicherheit**:
  - Für die Sicherheit (security) eines Automatisierungssystems muss grundsätzlich **in allen Fällen** gesorgt sein.
  - In jedem Betriebszustand des Systems muss sichergestellt sein, dass weder der **Mensch** noch die **Umwelt** in einem unzumutbaren Maße **gefährdet** ist.
  - Gesetzlich vorgeschriebene Regeln für sog. sicherheitsrelevante technische Prozesse.
  - **Ausfälle** von Einzelsystemen oder Rechnern müssen zu einem **sicheren Prozesszustand** führen (Fail Safe).

◆ **Beispiele**:

➤ Kraftwerke

➤ Chemische Reaktoren

➤ Ölförderungsanlagen

➤ Personentransportmittel

➤ Medizinische Anlagen bzw. Geräte.

◆ Im Falle von **Störungen** müssen die Anlagen

➤ **abgeschaltet** werden oder

➤ in einen **unkritischen Zustand** gebracht werden

Beispiele:

– bei Rechnerausfall im **Schienenverkehr** werden alle Transportmittel zum **Stillstand** gebracht um Kollisionen zu vermeiden

– bei **Stanzpressen** wird die Bedienung durch Handbetrieb nur unter Verwendung **beider Hände** ermöglicht.

◆ Da ein System nicht immer fehlerfrei arbeiten kann, muss bereits bei der **Konzeption** von technischen Anlagen für eine **schnelle Fehlermeldung** des Systems und **"sicheres" Systemverhalten** gesorgt werden.

Zuverlässigkeit:

- Die Zuverlässigkeit (reliability) eines technischen Systems ist ein Maß für die **Wahrscheinlichkeit**, mit der die Aufgaben des Systems in einem **vorgegebenen Zeitraum ordungsgemäß** erfüllt werden.
- Die Zuverlässigkeitsanforderungen beziehen sich sowohl auf die **Hardwareeinrichtungen** als auch auf die installierte **Software**.
- Je **höher** die Zuverlässigkeit, desto **geringer** sind **Produktions**- bzw. **Funktionsausfälle** im Prozess.
- Für die Betriebszuverlässigkeit spielen die eingesetzten **Rechner** und der **strukturelle Aufbau des Gesamtsystems** eine wesentliche Rolle.

- Von einem Prozessrechner wird erwartet, dass er die ihm übertragenen Aufgaben **korrekt** ausführt, was zu folgenden drei **Forderungen** führt:
  - Der Rechner soll keine **falschen Befehle** an den Prozess abgeben oder **falsche Anweisungen an das Betriebspersonal** liefern.
    - ➤ **Sorgfältige Programmierung** mit Korrektheitsprüfungen
    - ➤ Wichtige Berechnungen werden auf **zwei** oder mehreren unabhängigen **Wegen** durchgeführt und nur bei übereinstimmenden Ergebnissen ausgegeben (Softwareredundanz).

- Unwiederbringliche und wichtige **Daten** sollen **zerstörungssicher** gespeichert werden.

  ➤ Alle wichtigen Daten werden auf **zwei** voneinander unabhängigen **Datenträgern** gespeichert und müssen gegen Manipulationen von außen geschützt werden (Datensicherung).

- Der Rechner soll **nicht länger** als eine --- vom Prozess abhängige --- **zulässige Zeit ausser Betrieb** sein.

  ➤ Die Einhaltung dieser Forderung, der Verringerung der Ausfallzeiten, lässt sich durch **mehrfach** vorhandene **Hardwareeinrichtungen** erfüllen

1 Verfügbarkeit

- Rechner und Rechensysteme können **nicht ständig funktionsfähig** sein.
- Sie **fallen aus** und werden ggf. wieder **repariert**
- Elektronischen und mechanischen Teile in Rechensystemen verhalten sich wie eine **Badewannenkurve**:

◆ **Mittlere Ausfallzeit $\bar{t}_A$:**

➤ Mittlere Zeitspanne vom Zeitpunkt des **Ausfalls** einer Komponente bis zur **wiederhergestellten Betriebsfähigkeit.**

➤ Wird auch als **MTTR** (Mean Time To Repair) bezeichnet.

◆ **Mittlere Betriebszeit $\bar{t}_B$:**

➤ Mittlere Zeitspanne zwischen **zwei Ausfällen.**

➤ Wird auch als **MTBF** (Mean Time Between Failures) bezeichnet.

◆ **Ausfallwahrscheinlichkeit p:**

$$p = \frac{\mathrm{MTTR}}{\mathrm{MTTR} + \mathrm{MTBF}} = \frac{\bar{t}_A}{\bar{t}_A + \bar{t}_B}$$

➤ Die Ausfallwahrscheinlichkeit p eines Rechners **vergrößert** sich mit der **zunehmenden Betriebszeit** t eines Systems.

➤ Die Ausfallwahrscheinlichkeit p lässt sich statistisch aus dem **Betriebsverhalten** einer großen Anzahl von **gleichen Rechnern** und **Bauteilen** bestimmen.

◆ **Verfügbarkeit *q*** oder Betriebswahrscheinlichkeit:

$$q = \frac{\mathrm{MTBF}}{\mathrm{MTBF} + \mathrm{MTTR}} = \frac{\bar{t}_\mathrm{B}}{\bar{t}_\mathrm{B} + \bar{t}_\mathrm{A}}$$

➤ Die Verfügbarkeit q ist die **Wahrscheinlichkeit** dafür, dass sich ein System in einem **funktionsfähigen Zustand** befindet.

➤ Die **Ausfallwahrscheinlichkeit** p beschreibt den **entgegengesetzten** Zustand.

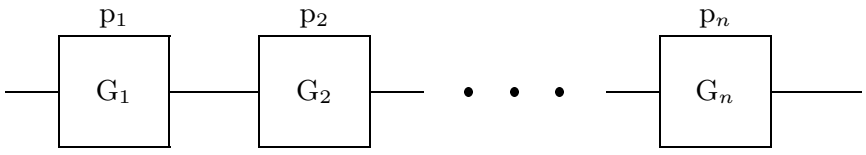
➤ Daher ergibt die **Summe** aus beiden **1** ($p + q = 1$).

2 Verfügbarkeit zusammengesetzter Systeme

Das Ausfallverhalten bzw. die **Verfügbarkeit** eines Systems hängt von der **Art der Verknüpfung** der einzelnen Komponenten ab.

■ Serienschaltung:

➤ Sind die einzelnen Geräte G_1, G_2, ... , G_n eines Rechensystems seriell miteinander verbunden, so ist das Gesamtsystem nur dann funktionsfähig, wenn sich **alle Geräte** im betriebsbereiten Zustand befinden.



➤ Die **Verfügbarkeit Q_S** des **Gesamtsystems** ergibt sich aus dem **Produkt** der Einzelverfügbarkeiten:

$$Q_\mathrm{S} = q_1 \cdot q_2 \cdot \ldots \cdot q_n = \prod_{i=1}^{n} q_i$$

- Sind statt der Einzelverfügbarkeiten die **Ausfallwahrscheinlichkeiten** gegeben, so errechnet sich Q_S wie folgt:

$$Q_S = (1 - p_1) \cdot (1 - p_2) \cdot \ldots \cdot (1 - p_n)$$

- In der Regel sind die Betriebszeiten sehr viel größer als die Ausfallzeiten, d.h. es gilt: $p_i << 1$ und damit:

$$Q_S \approx 1 - (p_1 + p_2 + \ldots + p_n)$$

- aus $P_S + Q_S = 1$ folgt für die **Ausfallwahrscheinlichkeit P_S** des **Gesamtsystems**:

$$P_S \approx \sum_{i=1}^{n} p_i$$

- Da $\mathrm{MTTR} << \mathrm{MTBF}$ gilt:

$$P_S \approx \frac{\mathrm{MTTR_{ges}}}{\mathrm{MTBF_{ges}}} \approx \frac{\mathrm{MTTR_1}}{\mathrm{MTBF_1}} + \frac{\mathrm{MTTR_2}}{\mathrm{MTBF_2}} + \ldots + \frac{\mathrm{MTTR_n}}{\mathrm{MTBF_n}}$$

- Nimmt man an, dass die mittlere Ausfallzeit MTTR_i aller Geräte i **gleich groß** ist, dann gilt, da das Gesamtsystem schon ausfällt, wenn eines seiner Geräte ausfällt: $\mathbf{MTTR_{ges} = MTTR_i}$, und für den reziproken Wert der mittleren Betriebszeit MTBF_{ges}:

$$\frac{1}{\mathrm{MTBF_{ges}}} = \frac{1}{\mathrm{MTBF_1}} + \frac{1}{\mathrm{MTBF_2}} + \ldots + \frac{1}{\mathrm{MTBF_n}}$$

➤ Mittlere **Betriebszeit** des Systems:

$$\mathrm{MTBF}_{\mathrm{ges}} = \left(\sum_{i=1}^{n} \frac{1}{\mathrm{MTBF}_i} \right)^{-1}$$

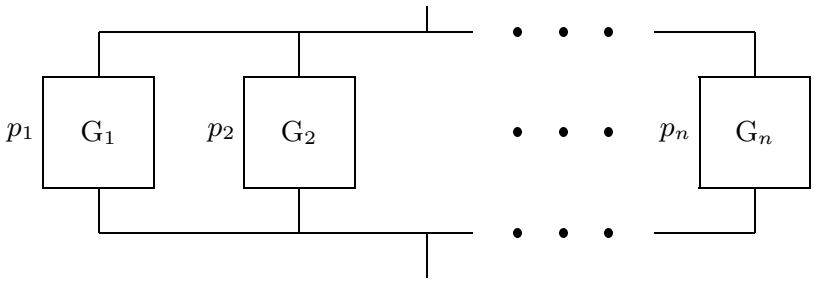
➤ Mit der Annahme, dass die mittlere **Betriebszeit** aller Geräte **gleich groß** ist:

$$\mathrm{MTBF}_{\mathrm{ges}} = \frac{\mathrm{MTBF}}{n}$$

➤ Unter den obigen Annahmen ergibt sich bei einer einer **Serienschaltung**, eine **Verringerung der Betriebszeit** des Gesamtsystems (also die Zeit zwischen zwei Ausfällen) auf das **1/*n*-fache** der Betriebszeit eines einzelnen Gerätes.

Parallelschaltung:

➤ Setzt sich ein Gesamtsystem aus seinen einzelnen Geräten so zusammen, dass jedes der ***n*** **Geräte** $G_1, G_2, \ldots, G_n$, zu den anderen **parallel** geschaltet ist, so ist das System solange betriebsfähig, solange noch mindestens **eines der *n* Geräte** funktioniert.



➤ Die **Ausfallwahrscheinlichkeit** des Gesamtsystems ergibt sich aus dem **Produkt der Einzelausfallwahrscheinlichkeiten:**

$$P_{\mathrm{S}} = p_1 \cdot p_2 \cdot \ldots \cdot p_n$$

- Haben alle Geräte die **gleiche Ausfallwahrscheinlichkeit**, d.h. $p = p_1 = p_2 = ... = p_n$, so gilt:

$$P_S = p^n$$

- Für die mittlere Gesamtbetriebszeit MTBF_{ges} gilt, unter den Annahmen:
  - $P_S = \mathrm{MTTR}_{ges} / \mathrm{MTBF}_{ges}$ und
  - $\mathrm{MTTR}_{ges} \approx \mathrm{MTTR}_i$, da i.a. immer nur ein ausgefallenes Gerät zu reparieren ist ($1 < i < n$):

$$\mathrm{MTBF}_{ges} \approx \frac{\mathrm{MTTR}_i}{p^n}$$

Gemischtschaltung:

- Ein System bestehe aus ***n* Geräten** und sei funktionsfähig, solange noch ***m* beliebige Geräte** funktionieren (m von n System).
- Dann gilt für die **Gesamtausfallwahrscheinlichkeit** P_S:

$$P_S = \sum_{r=0}^{m-1} \binom{n}{r} (1-p)^r p^{n-r}$$

- Die **Serienschaltung** ist ein Spezialfall und kann als n von n System mit $p << 1$ betrachtet werden, so dass gilt:

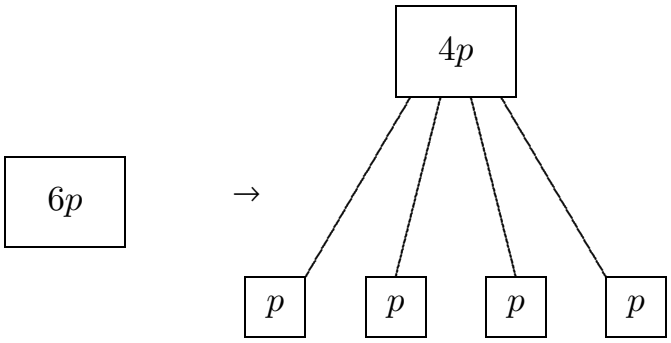
$$P_S \approx n \cdot p$$

- Die **Parallelschaltung** ist der Spezialfall eines 1 von n Systems und es ergibt sich:

$$P_S = p^n$$

3 Konfigurationen zur Erhöhung der Verfügbarkeit

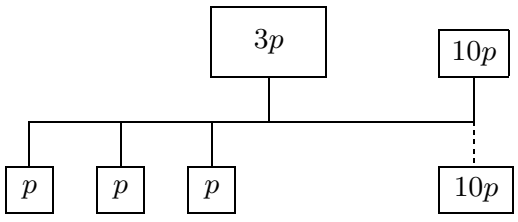
Einrechnersystem:



- Ein **Rechner** mit der Ausfallwahrscheinlichkeit **6*p*** wird ersetzt durch einen **Rechner** mit der Ausfallwahrscheinlichkeit **4*p***, an den noch **vier Mikrorechner** mit der jeweiligen Ausfallwahrscheinlichkeit ***p*** angeschlossen sind.
- Ausfallwahrscheinlichkeit für Aufgaben die **Mikrorechner** und den **übergeordneten Rechner** benötigen: $5p$

Geräteredundanz:

- Durch die **Parallelschaltung von Geräten gleichen Typs,** die besonders wichtige Aufgaben in einem Gesamtsystem übernehmen, lässt sich die Ausfallwahrscheinlichkeit einer gesamten Anlage erheblich **reduzieren**.
- Im Falle einer **Störung** des **ersten** Gerätes übernimmt das **zweite**, parallelgeschaltete Gerät dessen Aufgaben.



- Gezieltes Einsetzen **redundanter** Geräte kann die **Zuverlässigkeit** des gesamten Rechensystems erheblich **erhöhen**, erfordert allerdings zusätzlich eine **Überprüfungslogik**, die den Ausfall des entsprechenden Gerätes registriert und das Umschalten übernimmt.

➤ **Ohne Redundanz** ergibt sich für die Systemausfallwahrscheinlichkeit:

$$P_{\mathrm{S}} = 16p$$

➤ **Bei Redundanz** des Gerätes mit der hohen Ausfallwahrscheinlichkeit ergibt sich:

$$P_{\mathrm{SR}} = 6p + (10p \cdot 10p) \qquad \text{mit } p \ll 1 \quad (p \text{ ist sehr viel kleiner als } 1)$$

➤ **Zahlenbeispiel**:

$$\text{Es sei } p = \mathrm{MTTR}/\mathrm{MTBF} = 1/5000, \text{ daraus folgt } P_{\mathrm{S}} = 16/5000$$

$$\text{Mit Redundanz erhält man:}$$

$$P_{\mathrm{SR}} = 6/5000 + 0,02/5000 \approx 6/5000$$

Doppelrechnersystem:

➤ Zwei Rechner die die **gleiche** Ausfallwahrscheinlichkeit besitzen und zueinander **parallel** geschaltet sind, so dass bei **Ausfall** eines der beiden der **andere** dessen Arbeit mit übernimmt:

- Ausfallwahrscheinlichkeit eines Einzelrechners $P_{\mathrm{S}} = 3p$, mit $p = 2 * 10^{-3}$
- Mittlere Ausfallzeit: MTTR = 12 Std

➤ Mittlere Betriebszeit eines Einzelrechners:

$$\mathrm{MTBF} = \frac{\mathrm{MTTR}}{3p} = \frac{12 \text{ Std}}{6 \cdot 10^{-3}} \approx 83 \text{ Tage}$$

➤ Für das Doppelrechnersystem mit $P_{\mathrm{S}} = (3p)^2$ ergibt sich:

$$\mathrm{MTBF} = \frac{\mathrm{MTTR}}{P_{\mathrm{S}}} \approx 38 \text{ Jahre}$$

- Zwei Rechnern mit **unterschiedlicher** Ausfallwahrscheinlichkeit (Master- Slave-System):
  - $P_M = 10p$ (Master)
  - $P_S = p$ (Slave)

- Gesamtausfallwahrscheinlichkeit:

$$P_{MS} = P_M P_S = 10p^2$$

- Zahlenbeispiel mit $p = 1/5000$

$$P_M = 10/5000,\ P_S = 1/5000 \text{ und } P_{MS} = 2 \cdot 10^{-3}/5000$$

E.3 Struktureller Aufbau von Automatisierungssytemen

◆ **Zuverlässigkeitsüberlegungen** haben eine entscheidenden **Einfluss** auf die **Struktur** von Prozessautomatisierungssystemen.

1 Struktur konventioneller Automatisierungssysteme:

- In der herkömmlichen **Einzelgerätetechnik** werden Regler, Überwachungsgeräte, Melder usw. zur Erfassung und Beeinflussung einzelner **Teilvorgänge** in technischen Prozessen eingesetzt.
- Diese Geräte sind untereinander **nicht gekoppelt** und arbeiten selbständig.
- Die **Informationsverarbeitung** erfolgt **parallel** und **unabhängig** voneinander.
- In diesm Fall steigen die **Kosten linear** mit der **Anzahl** der Einzelgeräte,
- Bei dieser konventionellen Einzelgerätetechnik liegt ein sehr **zuverlässiger** Betrieb vor, da der Ausfall eines einzelnen Gerätes noch nicht zu einem Betriebsausfall führt

2 Zentraler Prozessrechner

- Hohe **Anfangskosten**
- Kosten steigen aber nicht mehr mit dem Umfang der Aufgabe
- Kann **leicht** und kostengünstig **erweitert** werden
- Viel **flexibler** als Einzelgerätetechnik
- **Zuverlässigkeit geringer** → System fällt aus wenn Rechner ausfällt
- **Erhöhung** der Verfügbarkeit durch:
  - Doppelrechnersysteme
  - Mehrrechnersysteme
  - verteilte (dezentrale) Systeme
  - redundante Einzelgeräte

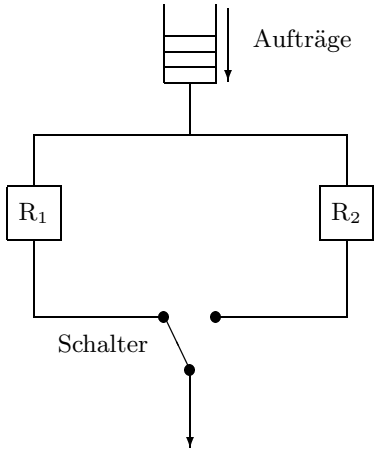
3 Redundante Einzelgeräte (Back-up-Geräte)

- Bei **Ausfall** des zentralen **Prozessrechners** wird auf **Einzelgeräte** umgeschaltet.
- Fehler müssen mithilfe von **Überwachungssystemen** erkannt werden
- Wird der Rechner auch zur **Regelung** eingesetzt, dann wird bei Rechnerausfall auf sog. **Back-up-Regler** umgeschaltet

4 Doppelrechnersysteme

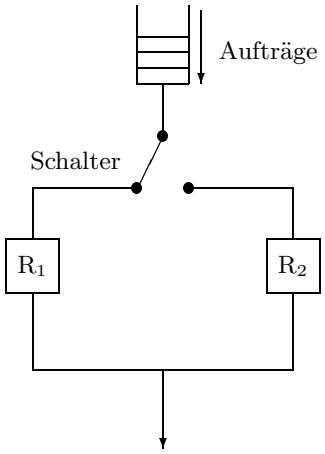
- ◆ **Zweiter Rechner überwacht** den **ersten Rechner** und übernimmt im Störungsfall den Betrieb.

■ Hot-Standby-System (Heiße Reserve)



- ➤ **Zwei identische Rechner** bearbeiten alle Aufgaben mit den gleichen Programmen **synchron**, aber nur **einer** der beiden Rechner gibt seine Ergebnisse/Daten tatsächlich weiter.
- ➤ Zu bestimmten **Zeitpunkten** --- bzw. an bestimmten Programmstellen --- werden die **Ergebnisse** aus beiden Rechnern mittels entsprechender Vergleichslogiken **überprüft**.
- ➤ Bei Abweichungen stellt ein **Diagnoseprogramm** den **gestörten** Rechner fest und sorgt dafür, dass alle weiteren Arbeiten auf dem **korrekt** arbeitenden Rechner **weiterlaufen**.
- ➤ Die eingesetzte **Vergleichslogik** muss allerdings mit **erhöhter Zuverlässigkeit** arbeiten, da sie einen erheblichen Einfluss auf die Zuverlässigkeit des gesamten Systems hat.
- ➤ Das **Problem**, das sich bei der Verwendung von zwei aktiven Rechnern ergibt, ist ihre **Synchronisation** (Taktfrequenz).
- ➤ Man setzt dazu entweder hardware- oder softwareseitig sogenannte **Wartepunkte**, die für den Gleichlauf beider Rechner sorgen.

Cold-Standby-System (Kalte Reserve)



- Die **Kopplung** der beiden Rechner ist **loser**.
- **Alle Aufgaben** laufen auf **einem Rechner**, während dem **zweiten** in festen Zeitabständen die **aktuellen Daten** übermittelt werden.

- Ein **Zeitüberwachungsprogramm** sorgt während des normalen Betriebs dafür, dass der arbeitende Rechner in **regelmäßigen Abständen** auf seine Funktionsfähigkeit **überprüft** wird.
- Das Zeitüberwachungsprogramm schaltet zwischenzeitlich --- auch bei störungsfreiem Ablauf --- auf den **zweiten Rechner** um, um so dessen Funktionsfähigkeit prophylaktisch zu überprüfen.
- Bei **Störung** des laufenden Rechners wird auf den **zweiten umgeschaltet**.

Aufgabenteilung

◆ Gleichmäßige Aufteilung

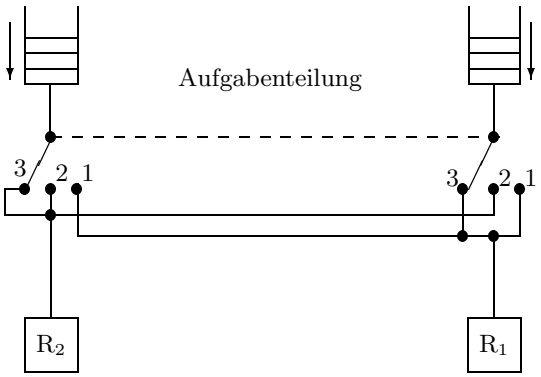


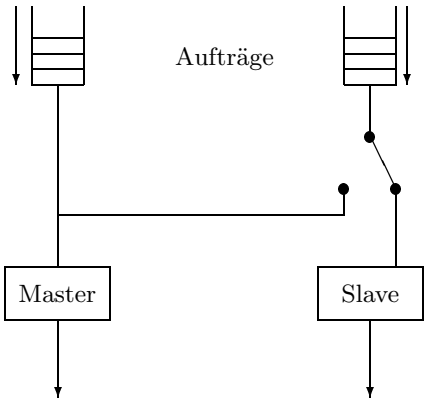
Fig. 1
In Schalterstellung 3 arbeiten beide Rechner parallel; in Schalterstellung 2 arbeitet nur Rechner 2; in Schalterstellung 1 arbeitet nur Rechner 1

➤ **Fällt einer** der beiden **aus**, so übernimmt der andere **alle Funktionen.**

➤ Es kann dann allerdings dazu kommen, dass dieser **manche Aufgaben** nur noch **untergeordnet** durchführt, da sich das Gesamtsystem in einem Notzustand befindet.

➤ Für diesen Fall müssen die einzelnen Aufgaben mit **Prioritäten** versehen werden

◆ Master-Slave-Konzept:



- Zwei Rechner mit sehr **unterschiedlichen** Ausfallwahrscheinlichkeiten arbeiten **parallel**.
- Besonders **wichtige Aufgaben** werden auf dem Rechner mit **hoher Verfügbarkeit** (niedriger Ausfallwahrscheinlichkeit) durchgeführt.
- Man nennt diesen Rechner den **Slave**.
- Für die **weniger wichtigen** und nicht zeitkritischen Aufgaben ist der **Master**, der eine geringere Verfügbarkeit besitzt, parallel geschaltet.
- Sollte der **Slave ausfallen**, so stellt der **Master** seine Aufgaben zurück und **übernimmt** die des Slaves.

5 Verdopplung einzelner Komponenten eines Rechensystems

- Häufig ist es **nicht notwendig** komplette Rechner zu **verdoppeln**.
- Stattdessen werden bestimmte **Komponenten** (die wichtigsten) innerhalb des Systems verdoppelt und **parallel** geschaltet.
- Dadurch ergibt sich eine wesentlich **höhere Verfügbarkeit**.
- Bei der Verdopplung einzelner Komponenten müssen **zwei identische Komponenten** gleichzeitig **ausfallen**, damit das Gesamtsystem ausfällt.
- Diese Maßnahme zur Erhöhung der Zuverlässigkeit wird in der Praxis häufig angewendet.

6 Dreirechnersystem

- Bei **hohen** Zuverlässigkeitsanforderungen, wie z.B. in der Luft- und Raumfahrttechnik, bei Kernreaktoren oder Massenverkehrssystemen, so setzt man häufig ein **Dreirechnersystem** ein.

- Es besteht aus **drei identischen Rechnern** von denen jeder eine **eigene Stromversorgung** und eigene Speicher besitzt.

- Bei dieser Systemkonfiguration wird **nicht nur die Zuverlässigkeit** erhöht, sondern es wird gleichzeitig für die **Sicherheit** gesorgt.

- **Fehler** durch falsche Ausgabegrößen, die zu **gefährlichen** Prozesszuständen führen könnten, werden hierbei rechtzeitig vor der Weitergabe an den Prozess **abgefangen**.

- Die von den drei Rechnern ermittelten Ausgabedaten werden von einer **Auswahllogik**, die **sehr zuverlässig** sein muss, überwacht und verglichen.

- Ein **Ausgabesignal** wird nur dann an den Prozess **weitergeleitet**, wenn mindestens **zwei der drei Ausgabewerte übereinstimmen** (2 von 3 Auswahl).

- Um die Gefahren durch Fehler in der Auswahllogik zu vermeiden, verwendet man spezielle dynamische **Fail-Safe-Schaltkreise**.

- Durch den Einsatz von Dreirechnersystemen kann allerdings **nur** das Auftreten von **Hardwarefehlern** verringert werden.

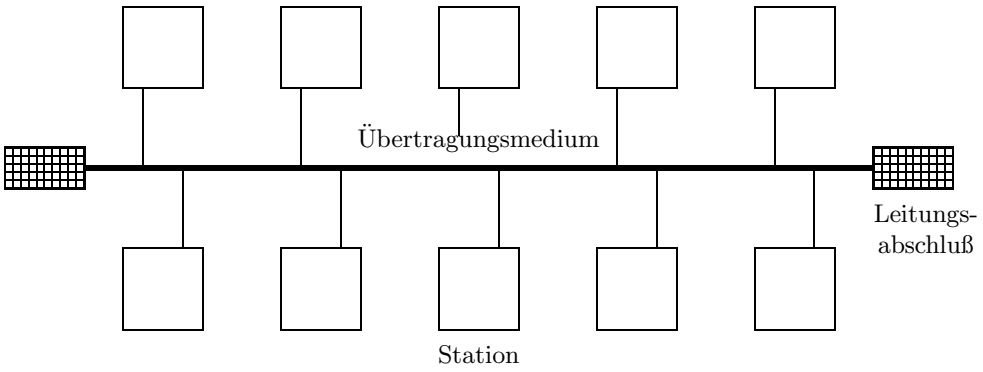
- **Softwarefehler**, wie Spezifikations-, Entwurfs- und Programmierfehler, können nur durch **strukturiertes** Programmieren und vor allem durch die Verwendung **unterschiedlicher Lösungsverfahren** und **unterschiedlicher Programmierung** vermieden werden.

- Dabei tritt allerdings das Problem auf, die unterschiedlichen Programme in ihrem Ablauf **zeitlich** zu **koordinieren**.

7 Dezentrale Prozessrechensysteme

- Es werden **mehrere Mikrorechner** so miteinander **gekoppelt**, dass sie untereinander kommunizieren können.
- Unterschiedliche Aufgaben können **gleichzeitig**, d.h. parallel abgearbeitet werden, indem sie auf die einzelnen Rechner im Rechnernetz **verteilt** werden.
- Somit ergibt sich sowohl eine **räumliche** als auch eine **aufgaben-bezogene** Verteilung der Rechner im System.
- Gegenüber den Einzelrechnern benötigt diese Konfiguration eine zusätzliche **Verwaltung** für die **Kommunikation** zwischen den einzelnen Recheneinheiten.
- Die dezentrale Struktur lässt sich überall dort einsetzen, wo sich der zu automatisierende Prozess aus **mehreren Teilprozessen** aufbaut.

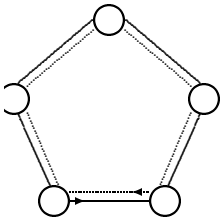
Bus-Struktur



- Ein Bussystem besteht aus bestimmten **Kontrollmechanismen** und einem Bündel funktional zusammengehriger **Übertragungsleitungen**, an denen mindestens zwei Geräte (Stationen oder auch Teilnehmer) angeschlossen sind, die miteinander kommunizieren wollen.

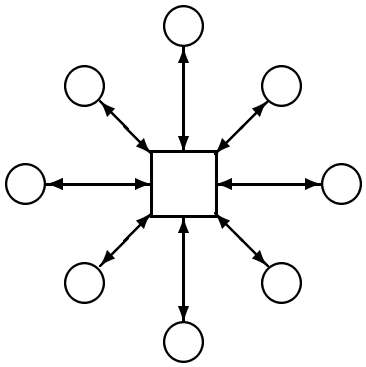
- Die **Übertragungskontrolle** übernimmt **Bus**, das hat den Vorteil, dass **neue Stationen** im laufenden Betrieb an den Bus **angeschlossen** werden können
- Entsprechend unproblematisch ist das Abschalten oder auch der Ausfall einer Station.
- Die Übertragung kann in **ungerichteter** oder auch **gerichteter** Form stattfinden (bidirektionale bzw. unidirektionale Übertragung).
- Der **Leitungsabschluss** des Busses ist ein **Widerstand**, der Reflexionen am Leitungsende unterbindet.
- Der **Vorteil** dieser Verbindungsstruktur liegt in den **geringen Verkabelungskosten** und der Möglichkeit, die **Anzahl** der Teilnehmer ohne großen Aufwand zu **erhöhen**.
- Der **Nachteil** einer reinen Bus-Struktur liegt darin, dass stets nur **eine Nachricht** über den Bus geschickt werden kann, weshalb die Busverwaltung (zentral oder dezentral) und der Fehlererkennung besondere Bedeutung haben.

Ringstruktur



- Die Einheiten sind **ringförmig** miteinander verbunden.
- Jeder Teilnehmer kann **nur** zu seinen **beiden Nachbarn** übertragen, so dass Nachrichten an weiter entfernte Teilnehmer des Rings von Teilnehmer zu Teilnehmer weitergereicht werden müssen.
- Es besteht die Möglichkeit der Kommunikation nur in einer oder in beiden Richtungen. Im zweiten Fall blockiert der Ausfall eines Übertragungs-weges nicht die gesamte Kommunikation blockieren.
- Am bekanntesten ist der sogenannte **Token-Ring**, bei dem eine Nachricht nur der Teilnehmer senden darf, der gerade im Besitz der Übertragungs-einheit, dem Token, ist.

Sternstruktur



- Ein zentraler Rechner (auch **Transitknoten** genannt) befindet sich in der Mitte des Systems.
- Jeder weitere **Rechner** (Kommunikationseinheit) ist durch eine **eigene Datenleitung** mit dem Transitknoten (Sternpunkt) verbunden.

- Je nach Art des Transitknotens können die Teilnehmer **alle gleichzeitig** an ihn **übertragen**, **oder** sie erhalten von ihm nacheinander die **Übertragungsberechtigung** zugeteilt.
- Die **Vorteile** dieser Struktur, gegenüber der vollvermaschten Netz-Struktur, liegen in der **hohen Auslastung** der Übertragungswege und den **geringen Kabelkosten**.
- **Nachteile** sind zum einen die möglichen **Umwege**, wenn zwei Einheiten miteinander kommunizieren wollen, denn sie müssen immer über den zentralen Rechner gehen, und zum anderen, dass beim Ausfall einer Datenleitung der entsprechende Rechner am Ende **keinerlei Kontaktmöglichkeit** mehr besitzt.
- Beim **Ausfall** des **zentralen** Rechners ist das gesamte **Netz kommunikationsunfähig**.

Netzstruktur

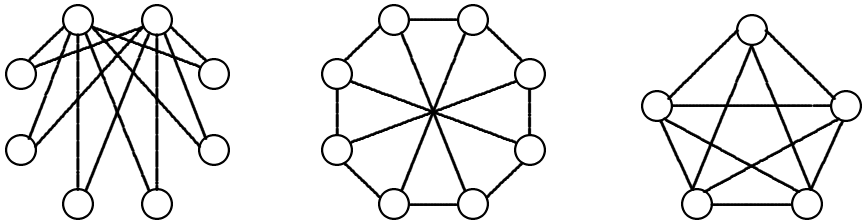
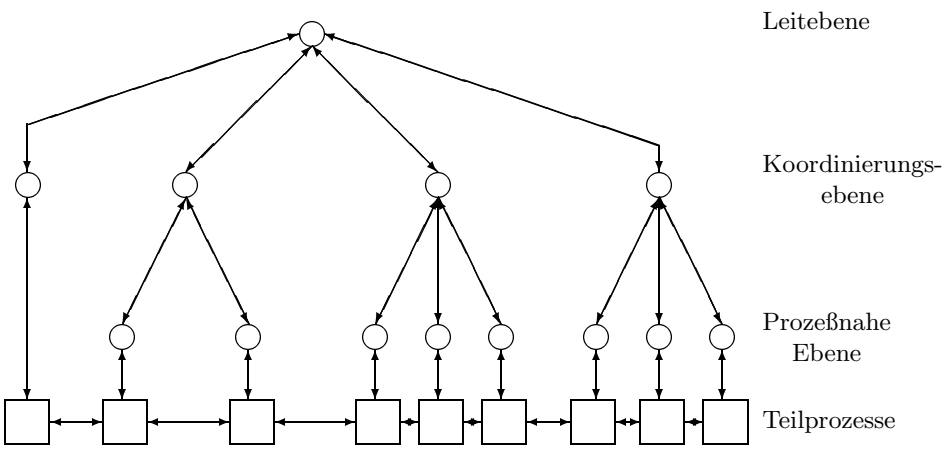


Fig. 1
Netzstrukturen: Zwei Rechner mit allen anderen vollvernetzt; Ring- und Stern -Struktur kombiniert; Volle Vernetzung

- Hoher Kostenaufwand
- Hohe Ausfallsicherheit
- Unabhängigkeit der einzelnen Übertragungswege
- Die Gestaltung des Netzes (voll-, teilvernetzt}), in Abhängigkeit von der Wichtigkeit der Teilnehmer und ihrer Verbindungswege.

Hierarchische Struktur



- Zur Automatisierung **großer technischer Prozesse** und ganzer **Werksanlagen** besonders geeignet, wegen der ebenfalls hierarchisch gegliederten Organisation und Bearbeitungsstruktur eines Betriebes.

- Im **einfachsten Fall** besteht eine hierarchische Struktur in der Prozessautomatisierung aus **zwei Ebenen**:
  - **Prozessnahe** Ebene aus Mikroprozessoren
  - **Übergeordnete** Ebene mit dem Host-Rechner, der mit der **Standardperipherie** ausgestattet ist.

- **Beispiel**
  - **Prozessnahe** Ebene: Mikroprozessoren die **Reglerfunktionen** übernommen haben.
  - **Übergeordnete** Ebene: Hostrechner ermittelt **Sollwerte**, die er an die Mikroprozessoren (Regler) weitergibt
  - **Mikroprozessoren** erfassen **Istwerte** und übertragen sie zur Auswertung und Speicherung an den Hostrechner.

Prozessleitsysteme (PLS)

- **Hierarchisches** und **dezentrales** Konzept

- Drei **Hauptebenen**:
  - Leitung und/oder Koordination
  - Bedienen und Beobachten
  - Messen, Steuern und Regeln

- Ein PLS ist ein Prozessrechensystem, das nach **oben** mit einem übergeordneten Rechensystem zur **Lastverteilung** und nach **unten** mit den einzelnen **Automatisierungsgeräten** verbunden ist.

- Die Aufgaben der **Leit- bzw. Koordinationsebene**:
  - **Sammeln** der gewünschten Anforderungen bzw. Aufträge
  - **Bestätigen** bzw. Quittieren beendeter Aufträge
  - **Speichern** von Langzeitinformationen.

➤ **Bedien- und Beobachtungsebene**

- ermittelt **Zwischenergebnisse** aus gemessenen Daten,
- **steuert** eventuelle größere **Lagerhaltungsvorgänge** oder auch das aufeinander **abgestimmte Anlaufen** von Förderbändern etc.
- **registriert** das **Ausfallen** von Einzelgeräten, Mikrorechnern und Hostrechner

➤ **Prozessnahe Ebene** enthält die **Bauelemente** die direkt am Proze\ss wirken:

- Regler,
- Fühler,
- Messgeräte,
- Grenzwertmelder etc.

➤ Prozessnahe Ebene ist für das d**irekte Weiterleiten** von **Prozessdaten** und das **Senden** von **Sollvorgaben** an den Prozess verantwortlich.

E.4 Prozessrechneraufbau

◆ Grundsätzlich **unterscheidet** sich ein Prozessrechner in seinem prinzipiellen Aufbau **nicht** von einem **konventionellen Digitalrechner.** Da er aber in Verbindung mit einem technischen Prozess eingesetzt wird, hat er **spezifische Eigenschaften**, die je nach **Einsatzgebiet** und damit je nach **Rechnertyp** unterschiedlich ausgeprägt sind.
Die **wesentlichen Eigenschaften** eines Prozessrechensystems sind:

- **Fähigkeit zum Echtzeitbetrieb** (Real-Time-System)
- **Einzelbit-Verarbeitungsmöglichkeit**
- Spezielle Einrichtungen zur **Ein- und Ausgabe** von Prozesssignalen
- **Robustheit**

1 Fähigkeit zum Echtzeitbetrieb (Real-Time-System)

- Der **Start** von Programmen ist bestimmt durch **Uhrzeit** und **Prozesszustand**.
- In vielen Fällen des Einsatzes sind dem Rechner **maximale Reaktionszeiten** (Antwortzeiten) vorgegeben.
- Der **Programmablauf** ist daher weitgehend durch den **Prozessablauf** bestimmt, dies erfordert:
  - ein komfortables und schnelles **Unterbrechungssystem**
  - einen **Echtzeitbetrieb**
  - eine komfortable **Prioritätensteuerung** zur Berücksichtigung von Zeit- und Dringlichkeitsanforderungen.

2 Einzelbit-Verarbeitungsmöglichkeit

- Bei Prozessteuerung werden z.B. **Kontaktstellungen** und **Gerätezustände** (ein/aus) abgefragt und verarbeitet (**logische Verknüpfungen**)
- Dies erfordert eine besonders effektive **Befehlsliste** für das **Bit-Handling**

3 Spezielle Einrichtungen zur Ein- und Ausgabe von Prozesssignalen

- Kennzeichen für Prozessrechner sind sehr intensive **E/A-Aktivitäten**
- Daher **schnelles E/A-System** erforderlich, das vor allem auch elektrische Signale aufnehmen, verarbeiten und wieder abgeben kann.
- Die dazu notwendige Kopplung zum Prozess wird über **geeignete Peripheriegeräte** hergestellt.
- Neben den **Messwertgebern** und **Stellgliedern** sind das vor allem die **A/D-** bzw. **D/A-Wandler,** die im allgemeinen den Prozesseinheiten zugeordnet sind.

4 Robustheit der Geräte

- Je nach Einsatzort der Prozessrechner und der dazugehörigen peripheren Geräte müssen bestimmte äußere **gerätetechnische Vorkehrungen** getroffen sein.
- Zum einen müssen sie z.B. **staubgeschützt** sein, falls sie in Zementwerken u.ä. Betrieben eingesetzt werden.
- Zum anderen müssen oft **Tastaturen** bzw. das gesamte Bedienfeld **besonders ausgelegt** sein (z.B. Spritzwasser geschützt, große Tasten oder Bedienung durch Griffel, etc.)

E.5 Busse - Verbindungswege im System

- Kommunikation erfolgt über ein **Kommunikationsmedium**.
- Kommunikation
  - zwischen Einheiten, die sich **innerhalb** eines Rechners befinden, wie dem Hauptspeicher und dem Rechnerkern,
  - für **Anschlüsse** an die **Peripherie** bzw. an andere Rechner, Baugruppen oder Steuerungseinheiten.
- Solche Verbindungs- bzw. Kommunikationswege sind heutzutage meistens als sogenannte **Busse** realisiert.
- Aufgrund der verschiedenen Einsatzgebiete und damit verbundenen unterschiedlichen Anforderungen gibt es eine Vielzahl technisch **unterschiedlich realisierter Busse.**

1 Basisfunktionen eines Bussystems

- **Buszuteilung**
- **Synchronisation**
- **Fehlerbehandlung**
- **Erfassen und Weiterleiten von Alarmen**

Buszuteilung (Busarbitrierung)

- Je nach **Anforderungen** an das Bussystem wird die **Zuteilung**, welcher der einzelnen Busteilnehmer eine Nachricht verschicken darf, unterschiedlich durchgeführt.

- Die Zuteilungsverfahren lassen sich grob nach **drei Merkmalen** gliedern:

  - dem **Ort** von dem aus die Zuteilung gesteuert wird (zentral, dezentral)

  - dem **Verfahren** für die Auswahl des nächsten Teilnehmers zur Busvergabe

  - den **zeitlichen Bedingungen** unter denen Busanforderungen von den Teilnehmern abgegeben werden dürfen

- **Ort** von dem aus die Zuteilung gesteuert wird (zentral, dezentral):

  - **zentral**, d.h., dass eine **ausgezeichnete Station** im System die einzelnen **Busanforderungen** (Busrequests) entgegennimmt.

    - Diese Station entscheidet, welchem Teilnehmer sie die Übertragung gestattet und schickt ihm ein entsprechendes Antwortsignal (bus grant).

    - Anschließend muss dieser Teilnehmer seine Übertragung sofort abwickeln.

    - **Vorteile** dieser Strategie sind
      - **schnelle Reaktionszeiten** bis zur Buszuteilung
      - die Tatsache, dass die Zuteilungslogik nur **an einer Stelle** im System vorhanden sein muss

- **dezentral**, d.h., dass sich **alle sendefähigen Teilnehmer** vor einer Übertragung davon überzeugen müssen, dass der Bus weder angefordert noch belegt ist.
  Hierfür gibt es wenigstens **vier** verschiedene **Grundprinzipien** über das Vorgehen:

  - Die **gegenseitige Abfrage** (Parallel Polling), bei der jeder Teilnehmer eine **eigene Anforderungsleitung** zu allen anderen Stationen besitzt.

  - Das **Reihungsverfahren** (Daisy-Chaining), bei dem ein **gemeinsames Busanforderungssignal** auf einer Anforderungsleitung durch alle Teilnehmer geschleift wird (Daisy-Chain), bis es wieder den Teilnehmer erreicht, der als erster den Bus angefordert hatte. Dieser sendet und reicht das Signal weiter etc.

- Sehr häufig vertreten ist die **zyklische Buszuteilung** (Token Passing), bei der ein spezielles Signal (**Token**) zyklisch von Teilnehmer zu Teilnehmer weitergereicht wird. Besteht bei Erhalt des Token ein Übertragungswunsch, so wird erst gesendet (d.h. die Nachricht an das Token angehängt) und dann das Token weitergereicht.

- Beim CSMA-Verfahren (Carrier Sense Multiple Access) **prüft** ein übertragungswilliger Teilnehmer ob auf dem Bus zur Zeit Daten transportiert werden. Wenn "Ja", dann wartet er ein Intervall ab und fragt dann wieder an, wenn "Nein" dann überträgt er seine Daten.

- **Verfahren** für die Auswahl des nächsten Teilnehmers zur Busvergabe:
  - statische **Prioritätsverfahren** (z.B. UNIBUS),
  - **faire** Zuteilungen (z.B. ETHERNET, PERKEO-Bus)
  - **sequentielle** Bearbeitung
  - **zyklisch** rotierend (z.B. PDV-Bus).
  - Mischformen
- **Zeitliche Bedingungen** unter denen Busanforderungen von den Teilnehmern abgegeben werden dürfen:
  - **fest** vorgegebene Zeitpunkte
  - **freie** Anforderungszeitpunkte

Synchronisation

- Bei allen Verfahren Buszuteilungverfahren müssen die Kommunikationspartner synchronisiert werden hinsichtlich
  - der **Signalübergabe** auf den Bus vom Sender,
  - der **Signalübernahme** vom Bus durch den Empfänger,
  - der **Bedeutung** und der **Funktion** des übermittelten Signals.
- Dabei finden sowohl **synchrone** als auch **asynchrone** Übertragungsprinzipien, wie auch deren Mischformen, Verwendung.

Fehlerbehandlung

- Bevor aufgetretene Fehler inmitten einer Übertragung behoben werden können, müssen sie vom System **erkannt** werden.

- Die **Fehlerursachen** können sehr vielfältig sein:
  - elektromagnetische Einwirkungen,
  - Kontaktprobleme,
  - Leiterbrüche,
  - Kurzschlüsse und
  - Stromversorgungsfehler.

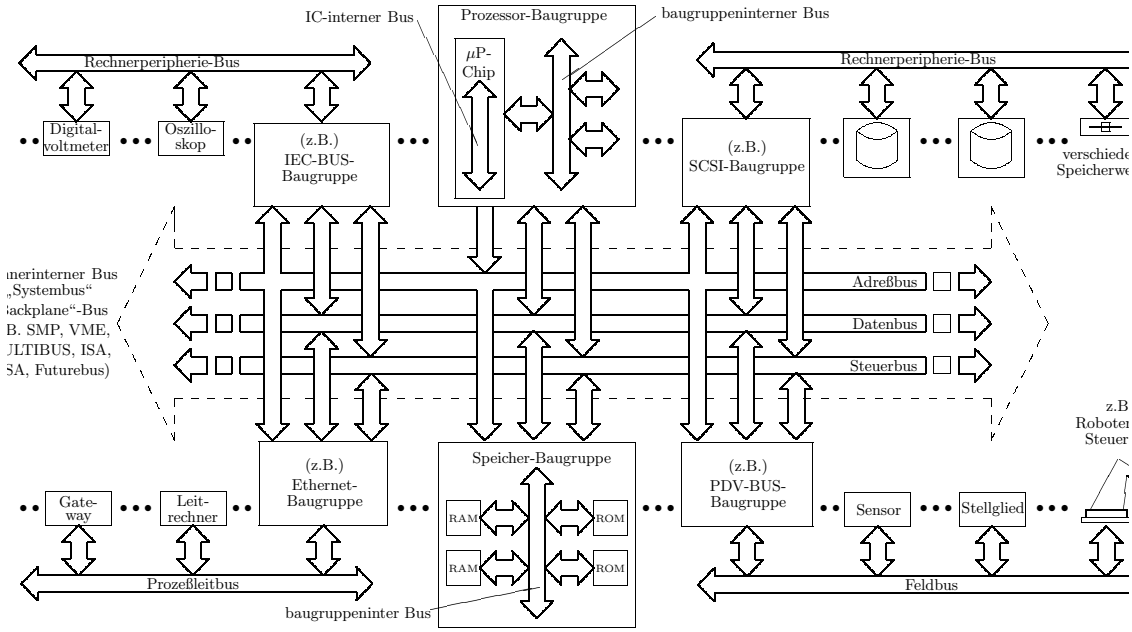
- Eine Fehlerbehandlung setzt sich aus **mehreren Teilaktionen,** die schon vor dem Auftreten eines Fehlers beginnen, zusammen:
  - Verwendung zusätzlicher **Prüfinformationen** beim Senden,
  - **Rückmeldungen** an den Sender,
  - **Zeitüberwachung** durch den Sender (Time-out),
  - **Wiederholung** im Fehlerfall,
  - Fehlerreport,
  - **Abschalten** eines als fehlerhaft erkannten Teilnehmers.

Alarmerfassung

- In der Prozessautomatisierung ist es wichtig, dass **spezielle Informationen** (z.B. ein unerlaubter Betriebszustand) schnell an andere Teilnehmer im System **übertragen** werden.
- Solche Alarme sollen **ohne Zeitverzögerung** über den Bus weitergeleitet werden.
- Sie werden als **Interrupt** bezeichnet, wenn sie zur Unterbrechung gerade laufender Programme führen.
- Für die **Art und Weise**,
  - ➤ wie eine Alarmübergabe an den Bus erfolgt,
  - ➤ welcher Teilnehmer das Alarmziel darstellt und
  - ➤ wie eine Alarmweiterleitung erfolgt,

  gibt es verschiedene Realisierungen, die hier aber nicht erläutert werden.

2 Bustypen



- Busse eines Prozessrechensystemes

Baugruppeninterner Bus

- Verbindet die **Bausteine** auf einer **Baugruppe** miteinander.
- Schafft die Kontakte zu den jeweiligen **übergeordneten** Systembussen.
- Er wird stark von den Eigenschaften der (Mikro-)Rechnerfamilie geprägt.
- Realisiert ist er meist als **geätzte** Leiterbahnen auf Platinen.
- Ein Beispiel ist der **IEC**-Bus.

Rechnerinterner Bus / Systembus / Backplane-Bus

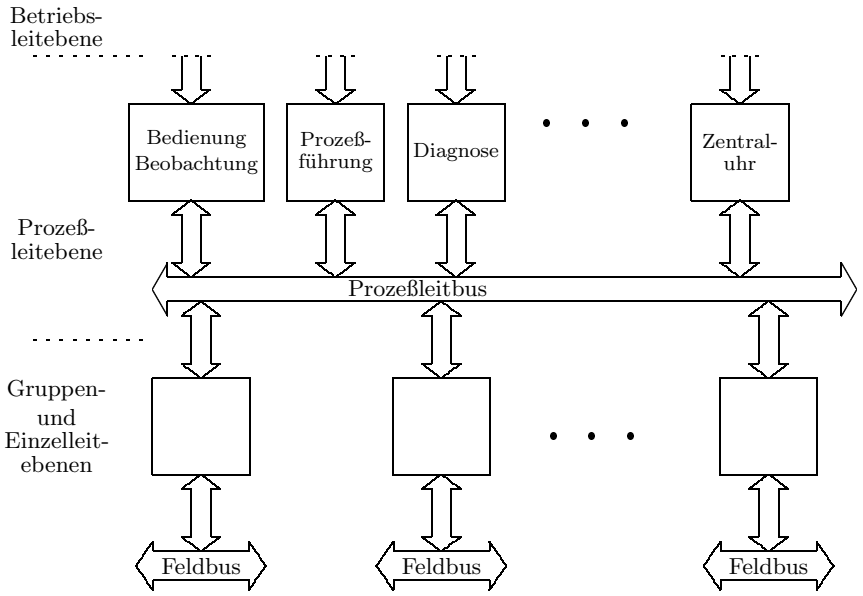
- Verbindung von **Baugruppen** in einem Einzel- oder Mehrrechnersystem
- Vorwiegend **Parallelstruktur**
- Als **geätzte** Leiterbahn, **Wire-Wrap** oder **Flachbandkabel** realisiert.
- Es wird versucht, diesen Bus **herstellerunabhängig** zu realisieren.
- Seine Ausdehnung im System beträgt ca. einen **halben** Meter.
- Bekannte rechnerinterne Busse sind der CAMAC-Dataway, der SMP-,
- VME- und der MULTIBUS.
- Übertragungsbandbreite bis zu **25** MByte/s.

Rechnerperipherie-Bus

- Verbindung von intelligenten **Peripheriegeräten** zu den einzelnen Rechnern des Rechensystems her.
- **Herstellerunabhängig**
- Parallelbus in Form eines **Flachbandkabels**
- Ausdehnung von ca. **20** Metern.
- Bekannte Vertreter sind der SCSI-Bus (Small Computer System Interface) und der IEC-Bus.

Feldbus

- Feldbus ist ein Oberbegriff für **prozessnahe** Busse die digitale Automatisierungseinheiten im unteren und mittleren Leistungsbereich miteinander verbinden.
- Vorzugsweise handelt es sich um **serielle** Busse aus **verdrillten** Zwei-Drahtleitungen.
- Ausdehnung bis zu **einem Kilometer**
- Kann **viele Teilnehmer** (ca. 200) miteinander verbinden.
- Von allen genannten Bussen haben sie die geringste Übertragungsrate von **9,6** bis max. **500** kBit/s.
- Bekannteste Vertreter sind der PDV-Bus, der BITBUS von INTEL und der PROFIBUS (z.B. SINEC-L2-Bus von SIEMENS)



◆ Feldbus, Prozessleitbus

Prozessleitbus

◆ Verbindung zwischen den Komponenten der **Prozessleitebene** und den Komponenten der **Feldebene** (z.B. auch flexible Fertigungszellen in der Fertigungsautomatisierung)

◆ Er hat meist **LAN**-Charakter (Local-Area-Network).

◆ Ausdehnung von ca. **1 Kilometer**

◆ Übertragungsraten zwischen **1** und **10 Mbps.**

◆ Die wichtigsten Vertreter sind der Token-Bus und der CSMA/CD-Bus

FDDI (Fiber Distributed Data Interface)

- Ringstruktur (Tokenring)
- Glasfasertechnologie
- Backbone-Netzwerk
- Datenrate > 100 Mbit/s
- Maximale Länge 100-200 km
- Bis zu 500-1000 Stationen, die jeweils bis zu zwei Kilometer auseinanderliegen.
- Dezentral