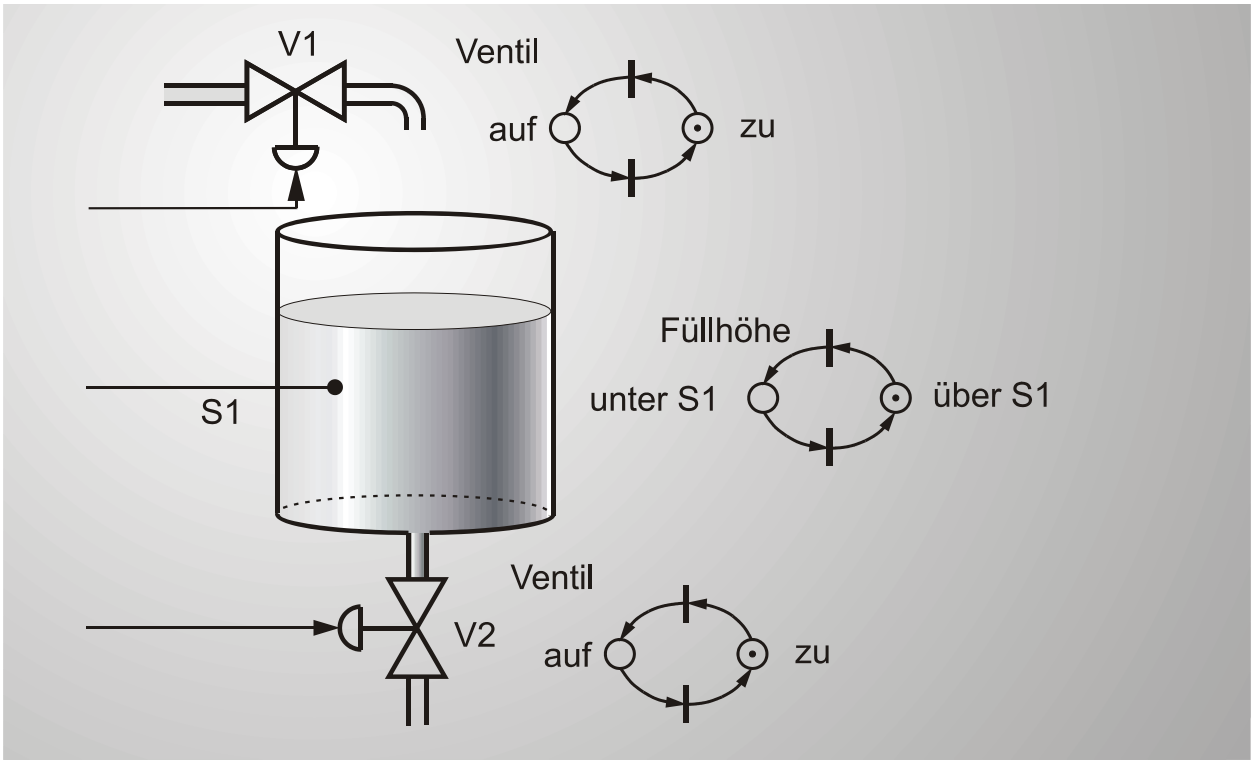
Beiblätter zur Vorlesung

Automatisierung ereignis-diskreter und hybrider Systeme

Dr.-Ing. Mathias Kluwe



Sommersemester 2016

http://www.irs.kit.edu

Karlsruher Institut für Technologie

Institut für Regelungs- und Steuerungssysteme
Institutsleiter: Prof. Dr.-Ing. Sören Hohmann

Cassandras, C. G., Lafortune, S.	Introduction to Discrete Event Systems. Springer Verlag, Netherlands, 2008.
Hopcroft, J. E., Motwani, R., Ullman, J. D.	Einführung in die Automatentheorie, formale Sprachen und Komplexitätstheorie. Adison Wesley Longman, München, 2002.
Abel, D.	Petri-Netze für Ingenieure. Springer Verlag, Berlin, 1990.
Hanisch, H.-M.	Petri-Netze in der Verfahrenstechnik. Oldenbourg Verlag, München, 1992.
Lunze J.	Ereignisdiskrete Systeme. Oldenbourg Verlag, München, 2006.
Rausch, M. P.	Modulare Modellbildung, Synthese und Codegenerierung ereignisdiskreter Steuerungssysteme. VDI Verlag, Düsseldorf, 1997.
Reinhardt, H.	Automatisierungstechnik: Theoretische und gerätetechnische Grundlagen, SPS. Springer Verlag, Berlin, 2007.
Wellenreuther, G., Zastrow, D.	Steuerungstechnik mit SPS. Vieweg Verlag, Braunschweig, 1998.
Starke, P. H.	Analyse von Petri-Netz-Modellen. Teubner Verlag, Stuttgart, 1990.
Baccelli, F., Cohen, G., Olsder, G. J., Quadrat, J.-P.	Synchronization and Linearity An Algebra for Discrete Event Systems. John Wiley & Sons, Chichester, 1992.
Moßig, K.	Algebraischer Steuerungsentwurf für eine Klasse ereignisdiskreter Systeme mittels der Max-Plus-Algebra. VDI Verlag, Düsseldorf, 1996.
Puente León, F., Kiencke, U.	Ereignisdiskrete Systeme. Oldenbourg Verlag, München, 2013.
Moody, J. O., Antsaklis, P. J.	Supervisory Control of Discrete Event Systems Using Petri Nets. Kluwer Academic Publishers, Boston, 1998.

Kontinuierliche Vorgänge

Kennzeichen:	Vorgänge, bei denen zeitabhängige kontinuierliche Prozessgrößen auftreten.
Prozessgrößen:	Physikalische Größen mit (zumindest stückweise) kontinuierlichem Wertebereich.
Beispiele:	Erzeugungsvorgänge, Umformungsvorgänge, Bewegungsabläufe.
Modelle:	Differentialgleichungen, Differenzengleichungen, Übertragungsfunktionen.

Sequentielle Vorgänge

Kennzeichen:	Vorgänge, bei denen Folgen von verschiedenen, unterscheidbaren Prozesszuständen auftreten.
Prozessgrößen:	Binäre Prozessgrößen, die das Eintreten der diskreten Prozesszustände melden oder bewirken, sowie kontinuierliche physikalische Größen, die den Prozesszuständen zugeordnet sind.
Beispiele:	Folgen von Prozesszuständen beim An- oder Abfahren einer Turbine, Folgen von Zuständen bei der Fahrt eines Aufzugs, Folgen von Zuständen bei der Fertigung mit Werkzeugmaschinen, Folgen von Prüfvorgängen bei der Geräteprüfung in einem Prüffeld.
Modelle:	Flussdiagramme, Funktionspläne nach DIN 40719, Zustandsmodelle, Petri-Netze.

Objektbezogene Vorgänge

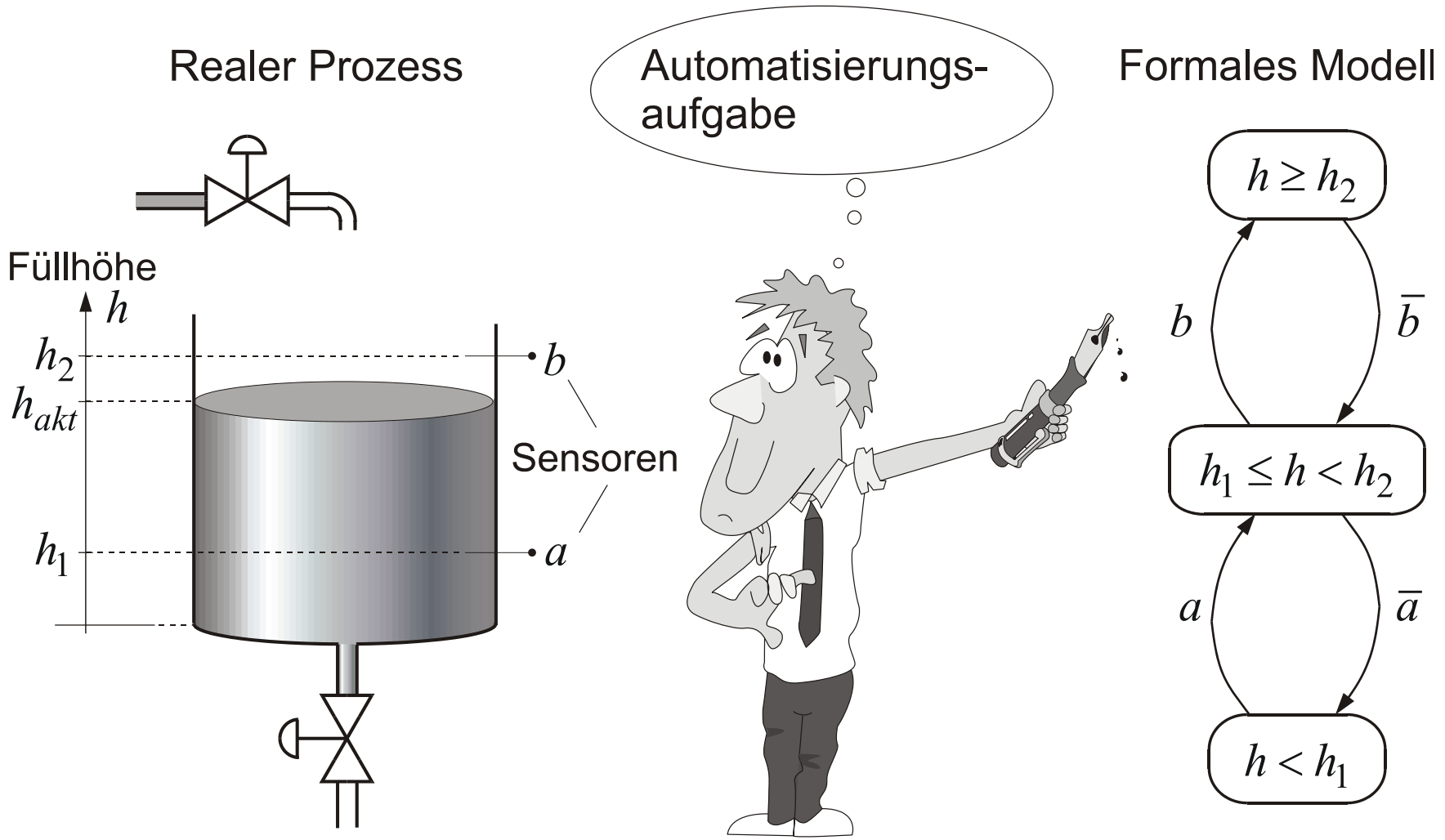
Kennzeichen:	Vorgänge, bei denen einzeln identifizierbare Objekte umgeformt, transportiert oder gespeichert werden.
Prozessgrößen:	Physikalische Größen mit kontinuierlichem Wertebereich oder nichtphysikalische Größen (wie z. B. Typ, Bauart, Verwendungszweck, Lager-Nr. usw.), die den Objekten zugeordnet sind, sowie binäre Prozesszustände, die Zustandsänderungen von Objekten auslösen oder melden.
Beispiele:	Vorgänge bei der Fertigung von Teilen, Verkehrsvorgänge, Lagervorgänge, Informationsvorgänge in Rechnern.
Modelle:	Simulationsmodelle, Warteschlangenmodelle, Graphen, Petri-Netze.

Typen von Vorgängen in technischen Produktionsprozessen

Technische Produktionsprozesse	Typen von Vorgängen
Energietechnische Prozesse	kontinuierliche Vorgänge sequentielle Vorgänge
Fertigungstechnische Prozesse	kontinuierliche Vorgänge sequentielle Vorgänge objektbezogene Vorgänge
Fördertechnische Prozesse	kontinuierliche Vorgänge sequentielle Vorgänge objektbezogene Vorgänge
Verfahrenstechnische Prozesse	kontinuierliche Vorgänge sequentielle Vorgänge

Klassifikation technischer Prozesse nach den dominierenden Vorgängen

Klassen technischer Prozesse	Dominierende Typen von Vorgängen	Beispiele
Fließprozesse	kontinuierliche Vorgänge	Energieerzeugung im Kraftwerk, Stahlerzeugung, chemischer Prozess, Heizprozess
Folgeprozesse	sequentielle Vorgänge	An- und Abfahrprozesse, Chargenprozesse, Fertigungsprozesse, Prüfprozesse
Stück(gut)-Prozesse	objektbezogene Vorgänge	Warentransportprozesse, Lagerprozesse, Verkehrsprozesse


Realer Prozess
Automatisierungs-
aufgabe
Formales Modell
Füllhöhe
h
h_2
h_{akt}
h_1
b
a
Sensoren
$h \geq h_2$
$h_1 \leq h < h_2$
$h < h_1$
b
$\bar{b}$
a
$\bar{a}$

PROZESSTYPEN Aufgaben	**WISSENSMODELLIERUNG** Beschreibungsmittel	**WISSENSVERARBEITUNG** Anwendungsgebiete
Fließprozesse Analyse, Reglerentwurf	**analytisch, algebraisch** Differential- / Differenzengleichungen	**numerisch, sequentiell** Simulation, Regelung
Folgeprozesse, Stückgutprozesse Analyse, Steuerungsentwurf	**analytisch, heuristisch** Automaten, Petri-Netze, Warteschlangen	**numerisch, symbolisch, sequentiell und parallel** Simulation, Steuerung
komplexe Prozessstrukturen Planung, Konfigurierung, Prozessüberwachung, Diagnose	**heuristisch (primär)** Regelmengen, Objektnetzwerke, Künstliche Neuronale Netze (KNN)	**symbolisch, konnektionistisch** Prozessleitsysteme
(Reale Welt)	(Mensch und Rechner)	(Rechner und reale Welt)

System (IEV 151-11-27) (system)
Gesamtheit miteinander in Verbindung stehender Objekte, die in einem bestimmten Zusammenhang als Ganzes gesehen und als von ihrer Umgebung abgegrenzt betrachtet werden.

Zustandsgröße (IEV 351-21-08) (state variable)
Elemente eines Größenvektors $\underline{x}(t)$ in einem System mit den Differentialgleichungen

$$\dot{\underline{x}}(t) = \underline{f}(\underline{x}(t), \underline{u}(t))$$
$$\underline{y}(t) = \underline{g}(\underline{x}(t), \underline{u}(t)),$$

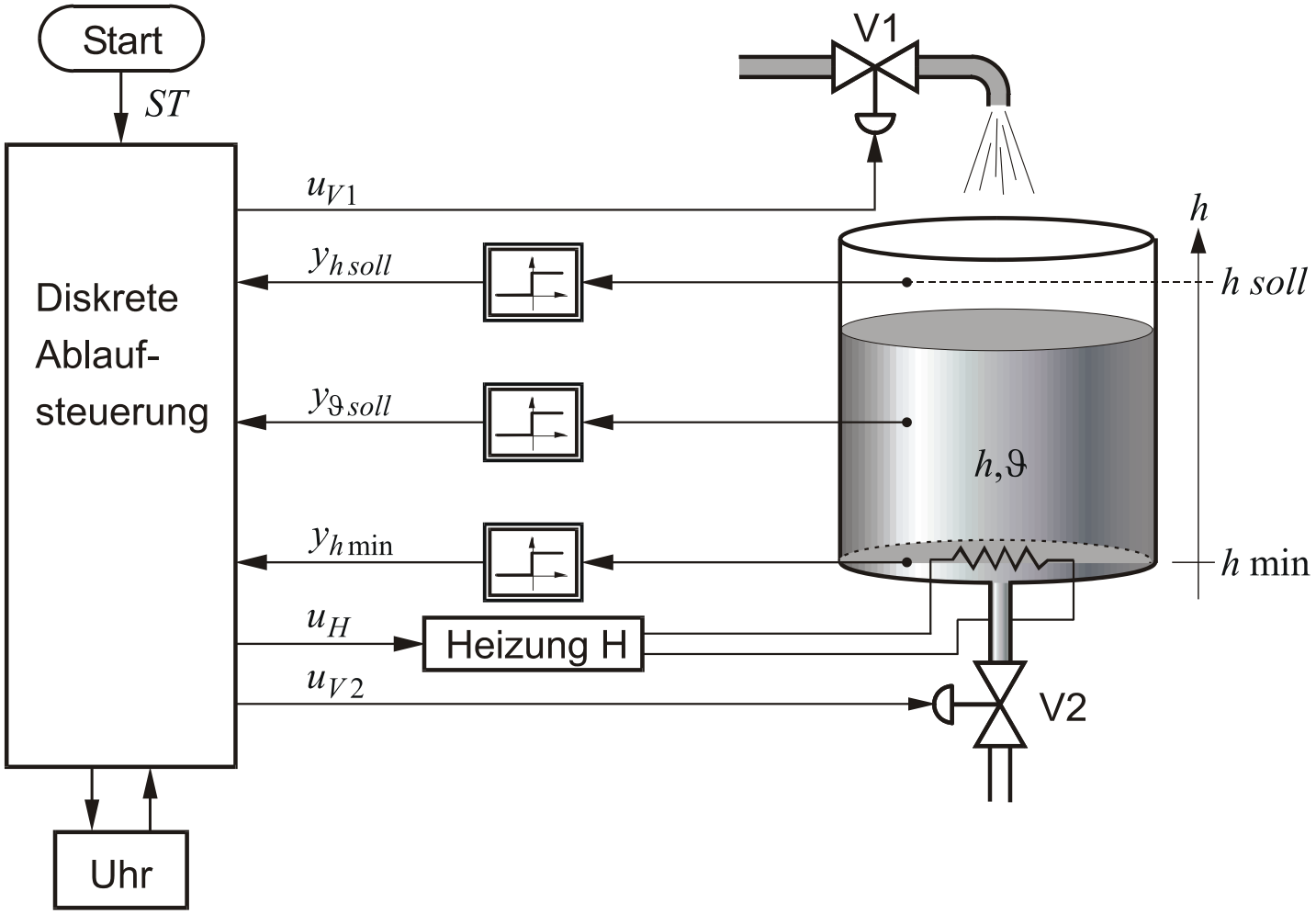
aus denen bei bekannten Anfangsbedingungen zu irgendeinem Zeitpunkt $t = t_0$ (häufig mit $t_0 = 0$) und bekanntem Zeitverlauf des Vektors der Eingangsgrößen $\underline{u}(t)$ ab dem Zeitpunkt t_0 der Zeitverlauf der Elemente des Vektors der Ausgangsgrößen $\underline{y}(t)$ berechnet werden kann.

Ereignis
Unter einem Ereignis versteht man die Änderung des Wertes einer Größe im System.

Dynamisches System
Ein dynamisches System beschreibt die Änderung des Zustandes in Abhängigkeit vom Zustand selbst und ggf. den Eingangsgrößen. Die Zustandsänderungen können dabei zeitgetrieben (zeitkontinuierlich oder zu bestimmten diskreten Zeitpunkten) oder ereignisgetrieben erfolgen.

Ereignisdiskretes System
Ein ereignisdiskretes System (discrete event system, DES) ist ein dynamisches System mit einer abzählbaren Menge diskreter Zustände, bei dem eine Zustandsänderung mit einem Ereignis einhergeht. Eine Zustandsänderung kann durch Einwirkung von außen hervorgerufen (gesteuertes Ereignis) oder allein durch das System selbst bewirkt werden (autonomes Ereignis).

Hybrides System
Ein hybrides System ist ein dynamisches System, das sich aus einem ereignisgetriebenen und einem zeitgetriebenen Systemanteil zusammensetzt, die in gegenseitiger Wechselwirkung stehen.



Prozess-Messgrößen:

$$\underline{y} = \left[y_{h\,soll}, y_{h\,\min}, y_{\vartheta\,soll} \right]^T$$

Stellgrößen:

$$\underline{u} = \left[u_{V1}, u_{V2}, u_H \right]^T$$

Bediengröße:

$$b = ST$$

Zeitverhalten / **Übergangsverhalten**	**rein kausal** Das Modell beschreibt die Abfolge der Zustände während des Prozessverlaufs.	**zeitbewertet** Das Modell beschreibt die Abfolge der Zustände während des Prozessverlaufs sowie die Verweildauer des Prozesses in den einzelnen Zuständen.	**stochastisch zeitbewertet** Das Modell beschreibt die Abfolge der Zustände während des Prozessverlaufs und beinhaltet eine statistische Verteilung der Verweildauer des Prozesses in den einzelnen Zuständen.
deterministisch Für jeden Systemzustand ist der Folgezustand bei gegebenen Eingangsgrößen eindeutig bestimmt.	• deterministische Automaten • Petri-Netze • *Formale Sprachen* • *Boolescher Differenzialkalkül* • *Polynombeschreibung*	• zeitbewertete deterministische Automaten • zeitbewertete Petri-Netze • *Temporale Logik* • *Max-Plus-Algebra*	
nichtdeterministisch Für mindestens einen Systemzustand ist der Folgezustand bei gegebenen Eingangsgrößen nicht eindeutig bestimmt.	• nichtdeterministische Automaten • Petri-Netze • *Formale Sprachen* • *Boolescher Differenzialkalkül* • *Polynombeschreibung*	• zeitbewertete nichtdeterministische Automaten • zeitbewertete Petri-Netze	
stochastisch Für mindestens einen Systemzustand ist der Folgezustand bei gegebenen Eingangsgrößen nicht eindeutig bestimmt, sondern wird mit einer bestimmten Wahrscheinlichkeit angenommen.	• stochastische Automaten		• Markov-Ketten • stochastische Petri-Netze • Warteschlangenmodelle • *Perturbationsanalyse*

Ein **endlicher Automat** A ist ein 7-Tupel

$$A = \left(X , E , f , x_0 , F, Y, g \right)$$

mit

- X : endliche Zustandsmenge,
- E : endliche Ereignismenge, auch Eingabealphabet genannt,
- f : Zustandsübergangsfunktion,
  - bei deterministischen Automaten $f : X \times E \to X$,
  - bei nichtdeterministischen Automaten $f : X \times E \to 2^X$,
- x_0 : Anfangszustand,
- F : endliche Final- oder Endzustandsmenge,
- Y : endliche Ausgabemenge, auch Ausgabealphabet genannt,
- g : Ausgabefunktion,
  - Moore-Automat (zustandsabhängig) $g : X \to Y$,
  - Mealy-Automat (zustands- und eingabeabhängig) $g : X \times E \to Y$.

Automatentafel:

Tabelle, in der die Automatenzustände über den Ereignissen aufgetragen sind. Der entsprechende Eintrag in der Tabelle gibt dann gemäß Zustandsübergangs- und Ausgabefunktion an, welcher Folgezustand und welche Ausgabe (nur bei Mealy-Automaten erforderlich) durch das betreffende Ereignis resultiert.

Automatengraph:

Gerichteter Graph, dessen Knoten (gezeichnet als Kreise) die Automatenzustände darstellen und dessen gerichtete Kanten (gezeichnet als Pfeile) die Ereignisse repräsentieren. Die Kanten sind dabei gemäß Zustandübergangs- und Ausgabefunktion mit dem Ereignis und der Ausgabe (nur bei Mealy-Automaten erforderlich) beschriftet, die den Zustandsübergang bewirken.

Der Anfangszustand und die Menge der Endzustände sind mit einem Extrapfeil bzw. Doppelkreisen besonders gekennzeichnet.

Alphabet:

Ein **Alphabet** E bezeichnet eine endliche, nichtleere Menge von Ereignissen (Symbole, Zeichen).

Wort:

Ein **Wort** (Zeichenkette) s ist eine endliche, nichtleere Sequenz von Ereignissen, deren Länge durch $|s|$ angebbar ist. Das Wort mit $|s|=0$ bezeichnet man als **leeres Wort** ε .

Unter einer **Konkatenation** $s_1 s_2$ versteht man die Aneinanderreihung zweier Wörter s_1 und s_2 , entsprechend heißt die n-fache Aneinanderreihung eines Wortes s die **Potenz** s^n .

Formale Sprache:

Bezeichnet man die Menge aller Wörter der Länge n mit $E^n\left(E^0=\{\varepsilon\}\right)$, so ergibt sich die **Menge aller möglichen Wörter** über E zu

$$E^* = \bigcup_{n=0}^{\infty} E^n \quad .$$

Eine **formale Sprache** $L(E)$ ist dann eine beliebige Teilmenge von E^*.

Die von Wörtern bekannten Operationen übertragen sich sinngemäß auf Sprachen:

- $L_1 L_2 = \{ s_1 s_2 \,|\, s_1 \in L_1 \wedge s_2 \in L_2 \}$,
- $L^n = L\,L^{n-1}$, $L^0 = \{\varepsilon\}$,
- $L^* = \bigcup_{n=0}^{\infty} L^n$.

Reguläre Sprache:

Eine **reguläre Sprache** $L(E)$ ist eine Sprache, die nur aus **regulären Ausdrücken** R besteht. Reguläre Ausdrücke sind Wörter, die rekursiv aufgebaut sind:

- $R \in E \vee R = \{\ \} \vee R = \varepsilon$ oder
- $R = (R_1 \cup R_2)$ mit R_1, R_2 regulär oder
- $R = (R_1 R_2)$ mit R_1, R_2 regulär oder
- $R = R_1^* = \{\varepsilon, R_1, R_1R_1, R_1R_1R_1, \ldots\}$ mit R_1 regulär.

Akzeptierte reguläre Sprache:

Eine von einem endlichen Automaten $A = (X, E, f, x_0, F, Y, g)$ **akzeptierte reguläre Sprache** L_A besteht aus allen Ausdrücken R, mit denen man vom Anfangszustand x_0 in die Menge F der Endzustände gelangen kann:

$$L_A = \left\{ R \in E^* : f(x_0, R) \in F \right\}.$$

Dabei gilt üblicherweise eine Erweiterung der Zustandsübergangsfunktion f:

$R = \varepsilon$: $\quad f(x, \varepsilon) = x$,

$R = R_1 R_2$: $\quad f(x, R) = f\left(f(x, R_1), R_2\right)$.

Zustandsmenge: $X = \{ x_1, x_2, x_3, x_4 \}$

Ereignismenge: $E = \{ e_1, e_2, e_3 \}$

Zustandsübergangsfunktion: $f(x_1, e_3) = f(x_4, e_2) = x_1$

$f(x_1, e_1) = f(x_3, e_3) = x_2$

$f(x_2, e_1) = x_4$, $f(x_2, e_2) = x_3$

Anfangszustand: $x_0 = x_1$

Endzustandsmenge: $F = \{ x_1 \}$

Ausgabemenge: $Y = \{ y_1, y_2 \}$

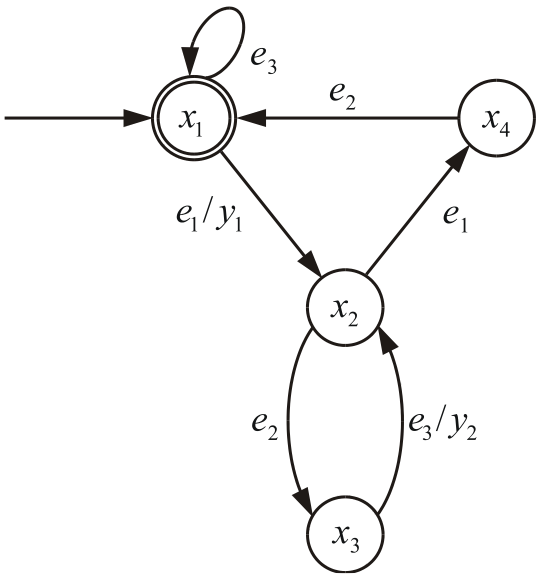
Ausgabefunktion: $g(x_1, e_1) = y_1$, $g(x_3, e_3) = y_2$ und

$g(x_1, e_3) = g(x_4, e_2) = g(x_2, e_1) = g(x_2, e_2) = \varepsilon$

Automatentafel:

	x_1	x_2	x_3	x_4
e_1	x_2 / y_1	x_4	-	-
e_2	-	x_3	-	x_1
e_3	x_1	-	x_2 / y_2	-

Automatengraph:



Akzeptierte reguläre Sprache des Automaten: $L_A = \left\{ \left(e_3^* \left(e_1 (e_2 e_3)^* e_1 e_2 \right)^* \right)^* \right\}$

Stellen: s_i, $i = 1,\ldots,n$ (im Netz als Kreise dargestellt)

- Stellenmenge $S = \{s_1,\ldots,s_n\}$
- Stellenkapazität $K(s_i)$, folgt aus $K : S \to \mathbb{N}$
  im Netz an Stelle geschrieben (keine Beschriftung: $K = 1$)

Transitionen: t_j, $j = 1,\ldots,m$ (im Netz als Balken dargestellt)

- Transitionenmenge $T = \{t_1,\ldots,t_m\}$

Kanten: (s_i, t_j) bzw. (t_j, s_i) (im Netz als Pfeile dargestellt)

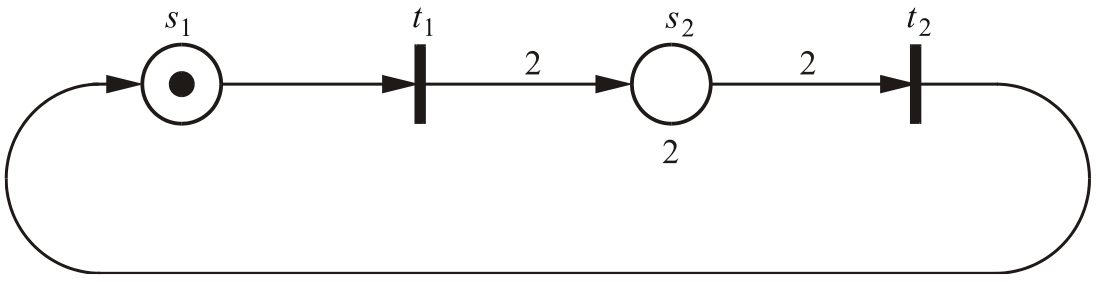
- Kantengewicht $W(s_i, t_j)$ bzw. $W(t_j, s_i)$, folgt aus $W : F \to \mathbb{N}$
  im Netz an Kante geschrieben (keine Beschriftung: $W = 1$)
- Kantenmenge $F = F_{pre} \cup F_{post} \subseteq (S \times T) \cup (T \times S)$

Marken: (im Netz als Punkte dargestellt)

- Netzmarkierung $M : S \to \mathbb{N} \cup \{0\}$, wobei $\forall i = 1,\ldots,n$ gilt: $0 \le M(s_i) \le K(s_i)$
  (Anfangsmarkierung: M_0)

Petri-Netz: $N = (S, T, F, K, W, M_0)$

Beispiel:



Vorbereich:

Der Vorbereich $\bullet s_i$ einer Stelle s_i enthält alle Transitionen, von denen eine Kante nach s_i führt:

$$\bullet s_i = \left\{ t_j \middle| (t_j, s_i) \in F_{post} \right\} \quad .$$

Der Vorbereich $\bullet t_j$ einer Transition t_j enthält alle Stellen, von denen eine Kante nach t_j führt:

$$\bullet t_j = \left\{ s_i \middle| (s_i, t_j) \in F_{pre} \right\} \quad .$$

Nachbereich:

Der Nachbereich $s_i \bullet$ einer Stelle s_i enthält alle Transitionen, zu denen von s_i ausgehend, eine Kante führt:

$$s_i \bullet = \left\{ t_j \middle| (s_i, t_j) \in F_{pre} \right\} \quad .$$

Der Nachbereich $t_j \bullet$ einer Transition t_j enthält alle Stellen, zu denen von t_j ausgehend, eine Kante führt:

$$t_j \bullet = \left\{ s_i \middle| (t_j, s_i) \in F_{post} \right\} \quad .$$

Aktivierte Transition:

Als Schaltvoraussetzung ist eine **Transition** t_j **aktiviert** (schaltfähig), wenn ihr Schalten aufgrund der Netzmarkierung zulässig ist, d.h. wenn gilt:

1) $\forall s_i \in \bullet t_j : M(s_i) \geq W(s_i, t_j)$,

2) $\forall s_i \in t_j \bullet : M(s_i) \leq K(s_i) - W(t_j, s_i)$.

Ein so genannter **Kontakt** liegt vor, wenn zwar 1) erfüllt ist, jedoch 2) nicht.

Schalten einer Transition:

Beim **Schalten** einer aktivierten Transition t_j wird jeder Stelle s_i mit $s_i \in \bullet t_j$ die Anzahl von $W(s_i, t_j)$ Marken entnommen und jeder Stelle s_k mit $s_k \in t_j \bullet$ die Anzahl von $W(t_j, s_k)$ Marken hinzugefügt.

Konflikt:

Ein **Konflikt** zwischen zwei aktiviertenTransitionen t_j und t_k tritt auf, wenn nach dem Schalten der Transition t_j die Transition t_k nicht mehr aktiviert wäre, was automatisch auch umgekehrt gilt. Im Rahmen einer Konfliktlösung muss daher bestimmt werden, welche der Transitionen schalten darf.

Schaltvorgang:

Der **Schaltvorgang** des Netzes besteht aus dem Schalten aller nach einer eventuellen Konfliktlösung aktivierten Transitionen, wobei vorab eine von zwei möglichen **Schaltregeln** festgelegt sein muss:

- **Muss-Schaltregel** (Schaltzwang):
  Jede aktivierte Transition muss beim Schaltvorgang schalten.
- **Kann-Schaltregel** (Schaltoption):
  Jede aktivierte Transition kann beim Schaltvorgang schalten, muss aber nicht.

A Stellen- bzw. Kanteneigenschaften

Gewöhnliches Petri-Netz:

Ein Petri-Netz heißt gewöhnliches Petri-Netz, wenn gilt:

$$\forall (s_i, t_j) \in F_{pre} : W(s_i, t_j) = 1 \quad \wedge \quad \forall (t_j, s_i) \in F_{post} : W(t_j, s_i) = 1 \quad .$$

Bedingungs-/Ereignis-Netz:

Ein Petri-Netz heißt B/E-Netz, wenn es sich um ein gewöhnliches Petri-Netz handelt, für das zusätzlich gilt:

$$\forall s_i \in S : K(s_i) = 1 \quad .$$

Markiertes Petri-Netz:

Ein Petri-Netz heißt markiertes Petri-Netz, wenn es sich um ein gewöhnliches und kontaktfreies Petri-Netz handelt.

B Struktureigenschaften

Reines Petri-Netz:

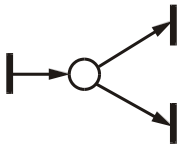
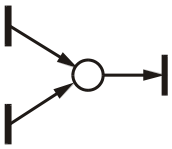
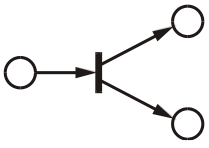
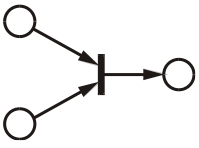
Ein Petri-Netz heißt reines Petri-Netz, wenn gilt:

$$\forall (s_i, t_j) \in F_{pre} : (t_j, s_i) \notin F_{post} \quad .$$

Zustandsmaschine:

Ein markiertes Petri-Netz heißt Zustandsmaschine, wenn gilt:

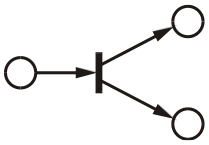
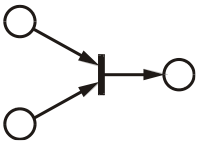
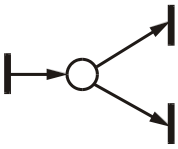
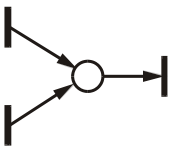
$$\forall t_j \in T : \left| \bullet t_j \right| = \left| t_j \bullet \right| = 1 \quad .$$

erlaubt		Verboten	
			
Alternative	Begegnung	Aufspaltung	Synchronisation

Synchronisationsgraph:

Ein markiertes Petri-Netz heißt Synchronisationsgraph, wenn gilt:

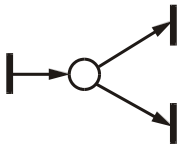
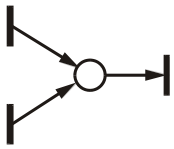
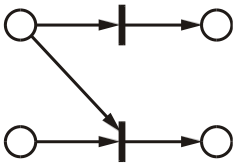
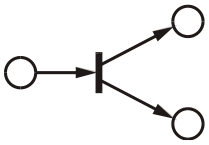
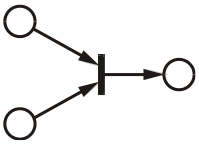
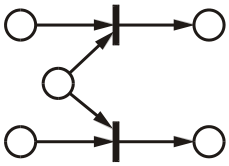
$$\forall s_i \in S: \left|\bullet s_i\right| = \left|s_i \bullet\right| = 1 \quad .$$

erlaubt		Verboten	
			
Aufspaltung	Synchronisation	Alternative	Begegnung

Free-Choice-Netz:

Ein markiertes Petri-Netz heißt Free-Choice-Netz, wenn gilt:

$$\forall s_i \in S: \left|s_i \bullet\right| = 1 \quad \vee \quad \bullet(s_i \bullet) = s_i \quad .$$

erlaubt		Verboten
		
Alternative	Begegnung	bedingte Alternative
		
Aufspaltung	Synchronisation	gemeinsame Ressource

(**Grundannahme:** Reines Petri-Netz betrachtet)

Stellenkapazitätsvektor: $\underset{(n,1)}{\underline{k}} = \left[K(s_1),\ldots,K(s_n)\right]^T$

Transitionenvektor: $\underset{(n,1)}{\underline{t}_j} := \left[t_{1j},\ldots,t_{nj}\right]^T \ (j=1,\ldots,m)$

mit $$t_{ij} := \begin{cases} -W(s_i,t_j), \text{ falls } s_i \in {}^\bullet t_j \\ W(t_j,s_i), \text{ falls } s_i \in t_j{}^\bullet \\ 0, \text{ sonst} \end{cases} \quad (i=1,\ldots,n;\ j=1,\ldots,m)$$

Netzmatrix (Inzidenzmatrix): $\underset{(n,m)}{\underline{N}} = \left(\underline{t}_1,\ldots,\underline{t}_m\right)$

Markierungsvektor:

Der Markierungsvektor $\underline{m}(k)$ gibt die Markenbelegung der Stellen nach dem k-ten Schaltschritt an:

$$\underline{m}(k) = \left[M(s_1),\ldots,M(s_n)\right]^T \quad .$$

Anfangsmarkierungsvektor:

Der Anfangsmarkierungsvektor $\underline{m}(0) = \underline{m}_0$ gibt die Markenbelegung der Stellen zum Beginn der Betrachtung an:

$$\underline{m}_0 = \left[M_0(s_1),\ldots,M_0(s_n)\right]^T \quad .$$

Schaltvoraussetzung:

Eine Transition t_j ist aktiviert (schaltfähig) unter der Markierung $\underline{m}(k)$, wenn durch deren Schalten eine zulässige Folgemarkierung erzeugt werden kann, d.h. wenn die Markenzahl auf keiner Stelle negativ wird oder die Stellenkapazität überschreitet:

$$\underline{0} \le \underline{m}(k) + \underline{t}_j \le \underline{k} \quad .$$

Schaltvektor:

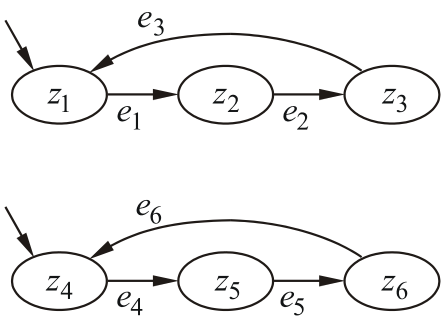
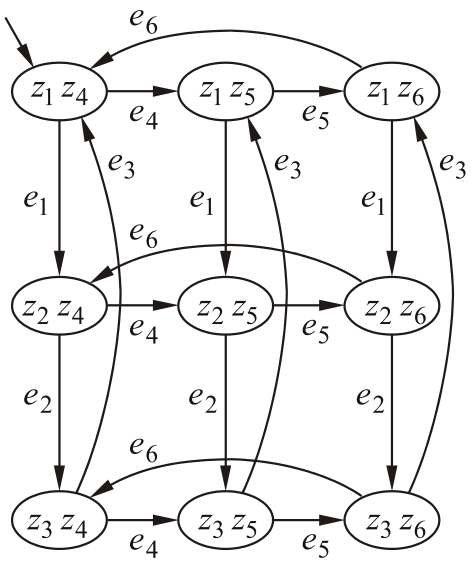
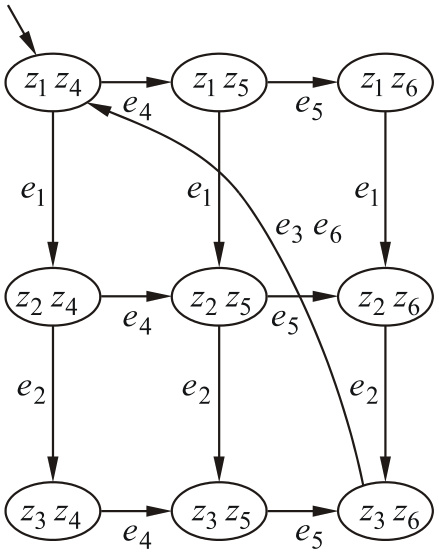
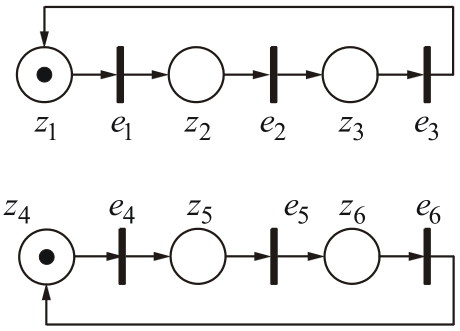
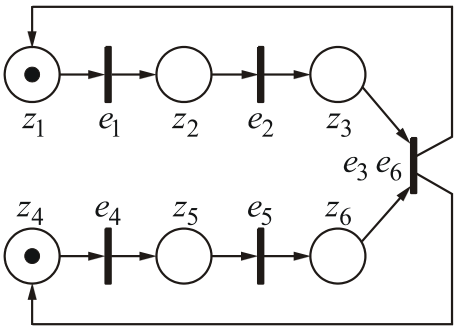
Die j-te Komponente $v_j(k)$ des Schaltvektors $\underline{v}(k)$ gibt an, ob die Transition t_j beim k-ten Schaltschritt schaltet:

$$v_j(k) = \begin{cases} 1, & t_j \text{ schaltet,} \\ 0, & t_j \text{ schaltet nicht.} \end{cases}$$

Folgemarkierung:

Die Folgemarkierung $\underline{m}(k+1)$ wird mittels der Zustandsgleichung des Petri-Netzes bestimmt:

$$\underline{m}(k+1) = \underline{m}(k) + \underline{N} \cdot \underline{v}(k+1) = \underline{m}(k) + \sum_{j \,|\, v_j(k+1)=1} \underline{t}_j \quad .$$

	asynchrone Nebenläufigkeiten	synchronisierte Nebenläufigkeiten
getrennte Zustands-graphen	 a	d
globaler Zustands-graph	 b	 e
Petri-Netz	 c	 f

Zur Abbildung einer zeitlichen Netzdynamik werden jeder Stelle s_i im Netz **lokale Uhren** zugeordnet. Diese werden jeweils bei der Belegung der Stelle durch eine Marke zum Zeitpunkt v_i gestartet und geben dann zur aktuellen Laufzeit t die Aufenthaltsdauer $t - v_i$ der entsprechenden Marke in der Stelle an.

Die Auswirkung der Markierungszeit ist dann durch zwei Kenngrößen festgelegt, die allen Prekanten (s_i, t_j) des Netzes zugeordnet sind:

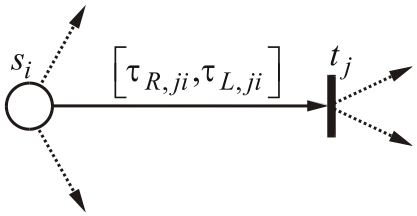
- **Retardierung** $\tau_{R,ji}$:

  gibt die Zeitspanne an, die eine Stelle $s_i \in \bullet t_j$ *mindestens* markiert sein muss, um die Transition t_j aktivieren zu können.

- **Limitierung** $\tau_{L,ji}$:

  gibt die Zeitspanne an, die eine Stelle $s_i \in \bullet t_j$ *höchstens* markiert sein darf, um die Transition t_j aktivieren zu können.

Netzdarstellung:



Schaltvoraussetzung bei Zeitbewertungen:

Mit $M_t(s_i, t_j)$ sei die Anzahl von Marken in der Stelle s_i bezeichnet, für die zum Zeitpunkt t gilt:

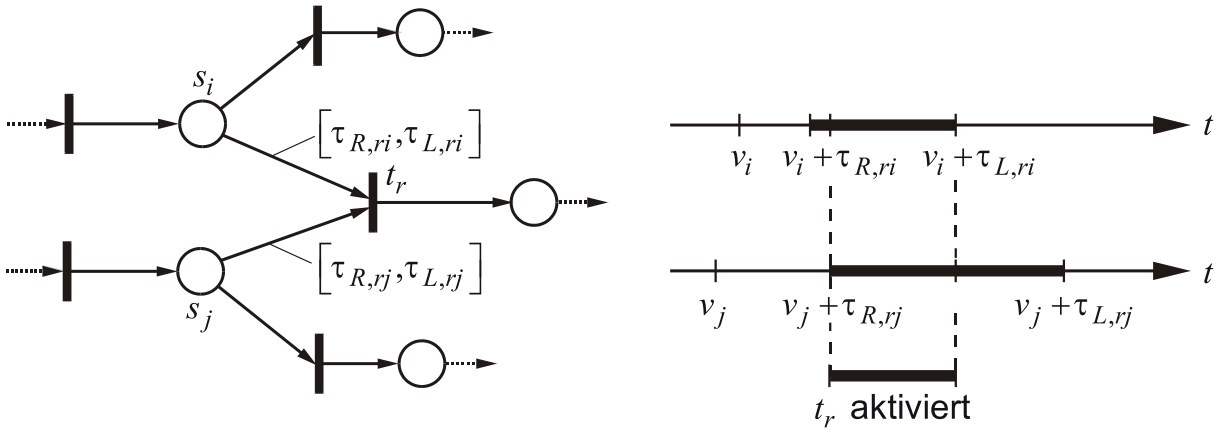
$$v_i + \tau_{R,ji} \le t \le v_i + \tau_{L,ji} \quad .$$

Eine **Transition** t_j ist dann **zum Zeitpunkt** t **aktiviert**, wenn gilt:

1) $\forall s_i \in \bullet t_j: \quad M_t(s_i, t_j) \ge W(s_i, t_j)$,

2) $\forall s_i \in t_j \bullet: \quad M(s_i) \le K(s_i) - W(t_j, s_i)$.

Beispiel:



Schaltzeitpunkt x_r der Transition t_r:

$$\max\left\{v_i + \tau_{R,ri}\,,\, v_j + \tau_{R,rj}\right\} \le x_r \le \min\left\{v_i + \tau_{L,ri}\,,\, v_j + \tau_{L,rj}\right\}$$

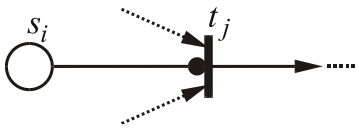
Test- bzw. Inhibitorkanten:

Eine **Testkante** (s_i, t_j) ist eine spezielle Prekante (gesamte Testkantenmenge $F_T \subseteq F_{pre} = S \times T$), die über die entsprechende *Belegung* der Stelle s_i zwar die Aktivierung der Transition t_j beeinflusst, über die aber beim Schaltvorgang kein Markenfluss stattfindet.

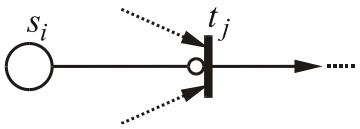
Eine **Inhibitorkante** (s_i, t_j) ist eine spezielle Prekante (gesamte Inhibitorkantenmenge $F_I \subseteq F_{pre} = S \times T$), die über die entsprechende *Nichtbelegung* der Stelle s_i zwar die Aktivierung der Transition t_j beeinflusst, über die aber beim Schaltvorgang kein Markenfluss stattfindet.

Netzdarstellungen:

Testkante



Inhibitorkante



Schaltvoraussetzung bei vorhandenen Test- bzw. Inhibitorkanten:

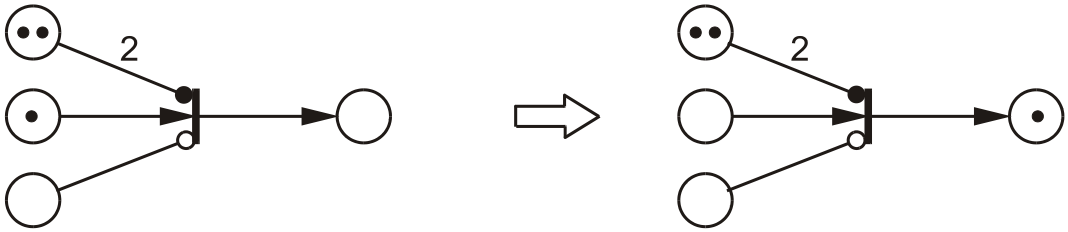
Eine **Transition** t_j ist **aktiviert** (schaltfähig), wenn:

1) $\forall s_i$ mit $(s_i, t_j) \in F_{pre} \setminus F_I$: $M(s_i) \geq W(s_i, t_j)$

2) $\forall s_i$ mit $(s_i, t_j) \in F_I$: $M(s_i) < W(s_i, t_j)$

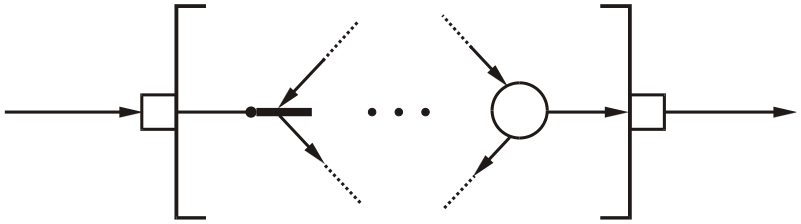
3) $\forall s_i$ mit $(t_j, s_i) \in F_{post}$: $M(s_i) \leq K(s_i) - W(t_j, s_i)$

Beispiel:

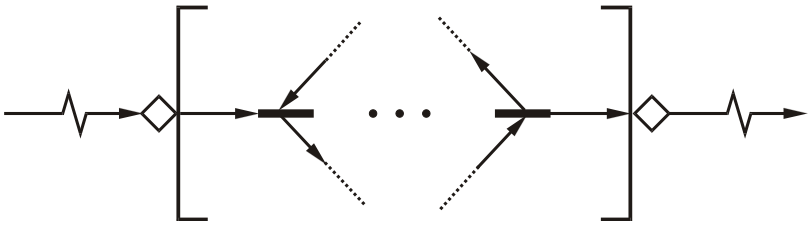


Signalsymbolik bei NCE-Systemen:

- **Conditionsignale**



- **Eventsignale**



Schaltvoraussetzung:

Eine Transition ist aktiviert (schaltfähig), wenn:

1) sie bezüglich der Markenbelegung in ihrem eigenen Teilnetz aktiviert ist,
2) und sie bezüglich aller Stellen aus anderen Teilnetzen, von denen Bedingungskanten zu der Transition führen, aktiviert ist.

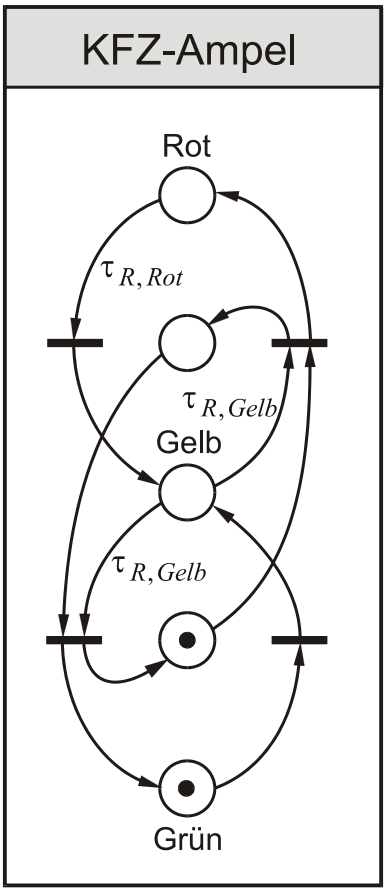
Schaltvorgang:

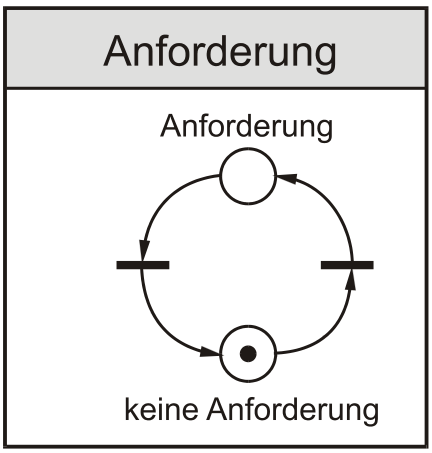
- Eine aktivierte Transition, an der keine Ereigniskante endet, kann schalten, wenn sie aktiviert ist (spontanes Schalten).
- Eine aktivierte Transition, an der eine Ereigniskante endet, schaltet genau dann, wenn sie aktiviert ist und diejenige Transition schaltet, von der die Ereigniskante ausgeht (erzwungenes Schalten).

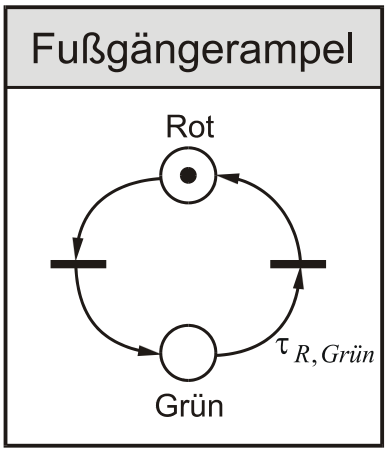
Bemerkungen:

Ein Markenfluss findet nur innerhalb der Module und nicht über die Bedingungs- oder Ereigniskanten statt.

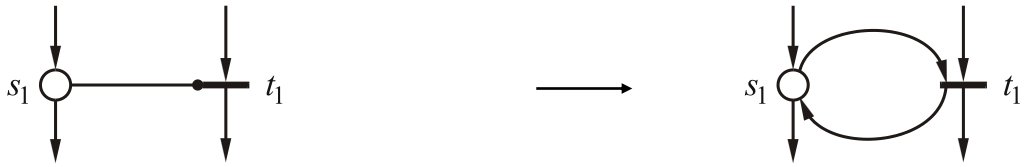
Ein Schaltvorgang wird nicht durch die Veränderung einer Bedingung erzwungen.


KFZ-Ampel
Rot
$\tau_{R,Rot}$
$\tau_{R,Gelb}$
Gelb
$\tau_{R,Gelb}$
Grün

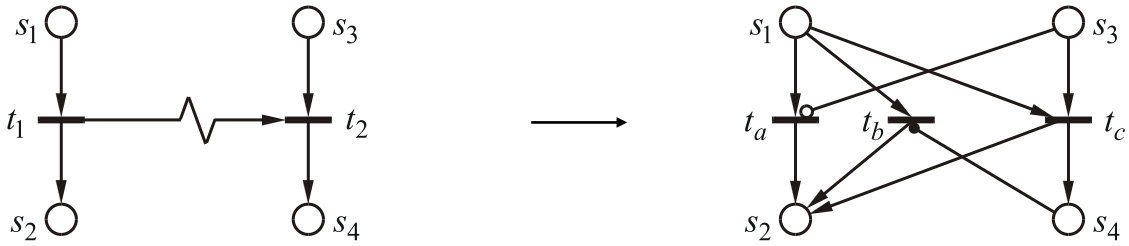

Anforderung
Anforderung
keine Anforderung


Fußgängerampel
Rot
$\tau_{R,Grün}$
Grün

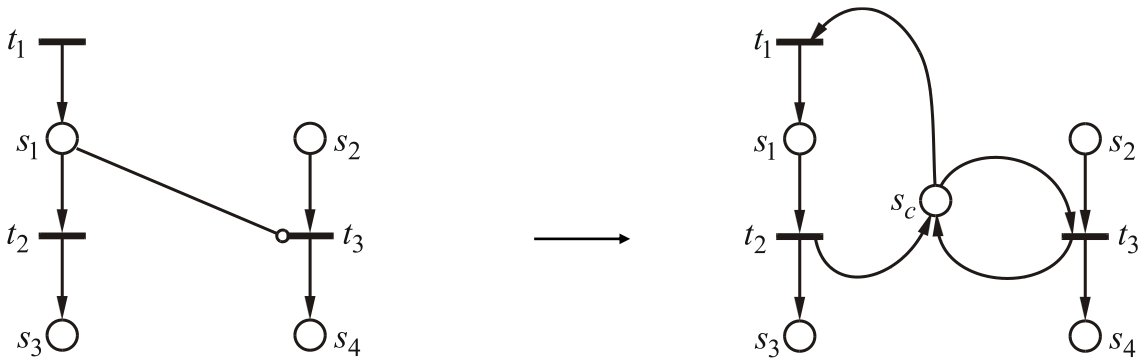
Condition-Kante (bzw. Test-Kante):

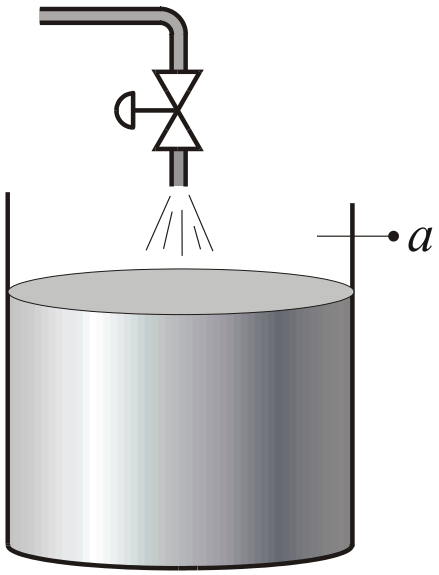


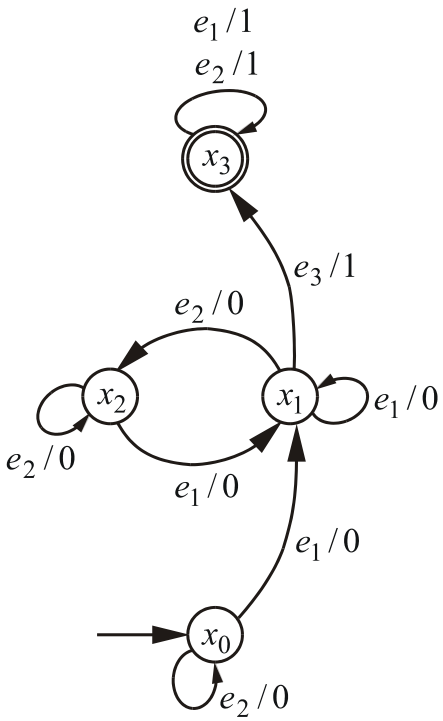
Event-Kante:



Inhibitor-Kante:







x_0 : Behälter leer

x_1 : Füllstand mittel und steigend

x_2 : Füllstand mittel und konstant

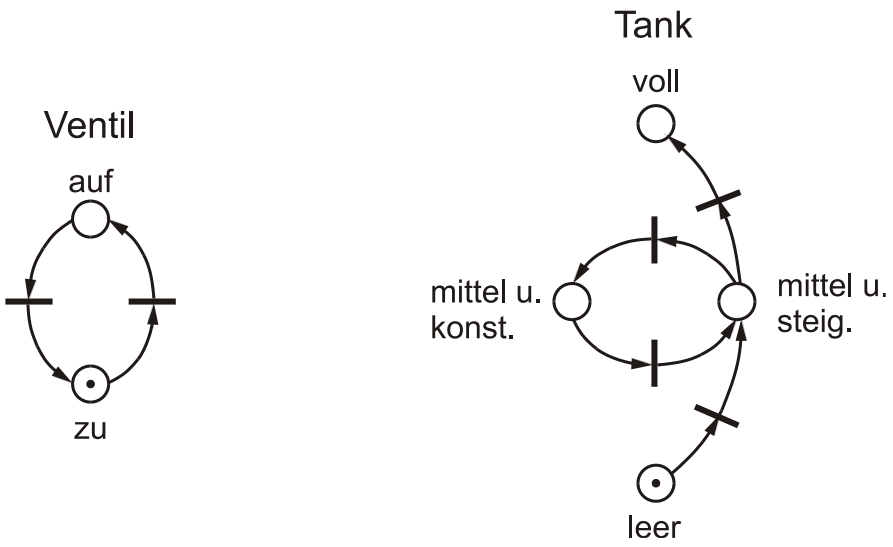
x_3 : Behälter voll

e_1 : Ventil wird geöffnet

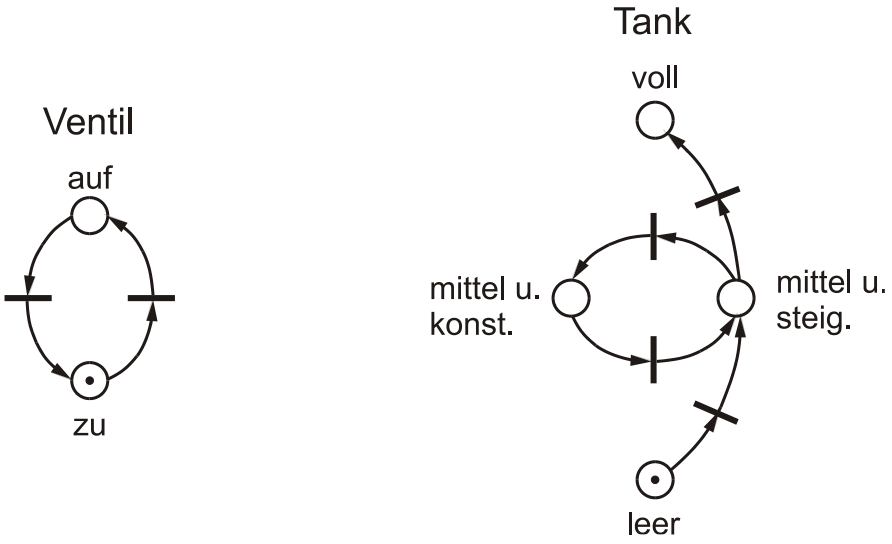
e_2 : Ventil wird geschlossen

e_3 : Sensor a spricht an

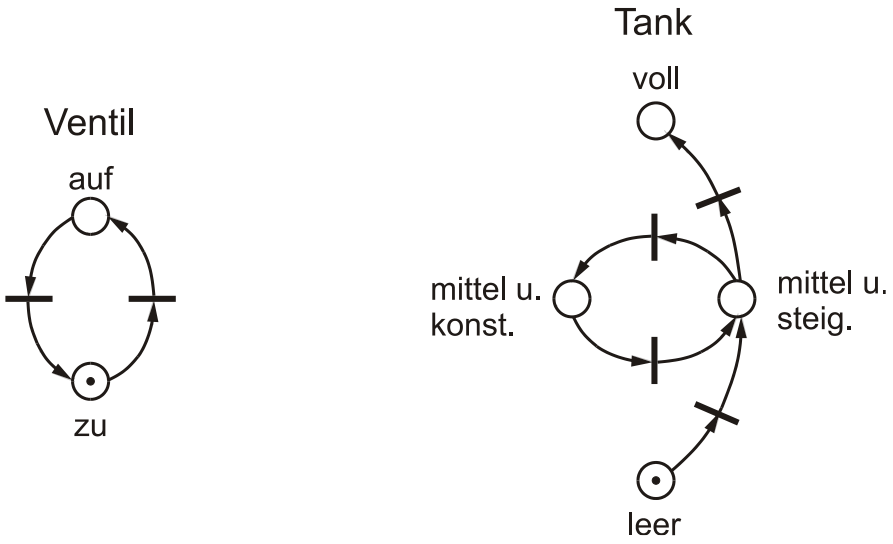
- **Separate Teilnetze**



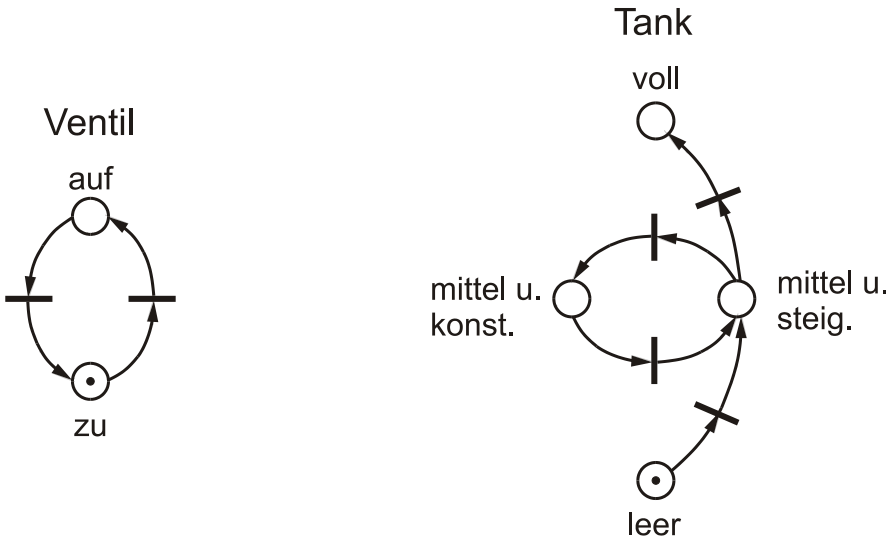
- **Verknüpfung durch Hinzufügen zusätzlicher Kanten**



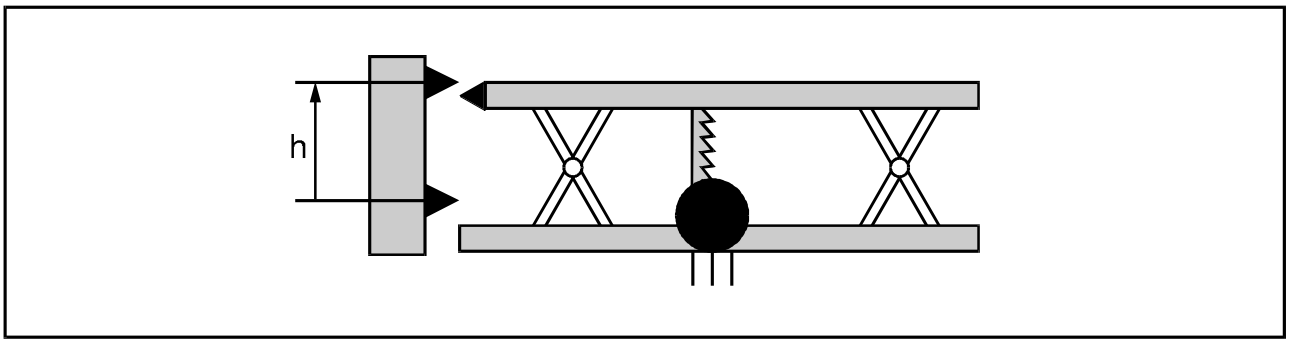
- **Verknüpfung unter Verwendung von Testkanten**



- **Modellierung als NCE-System**

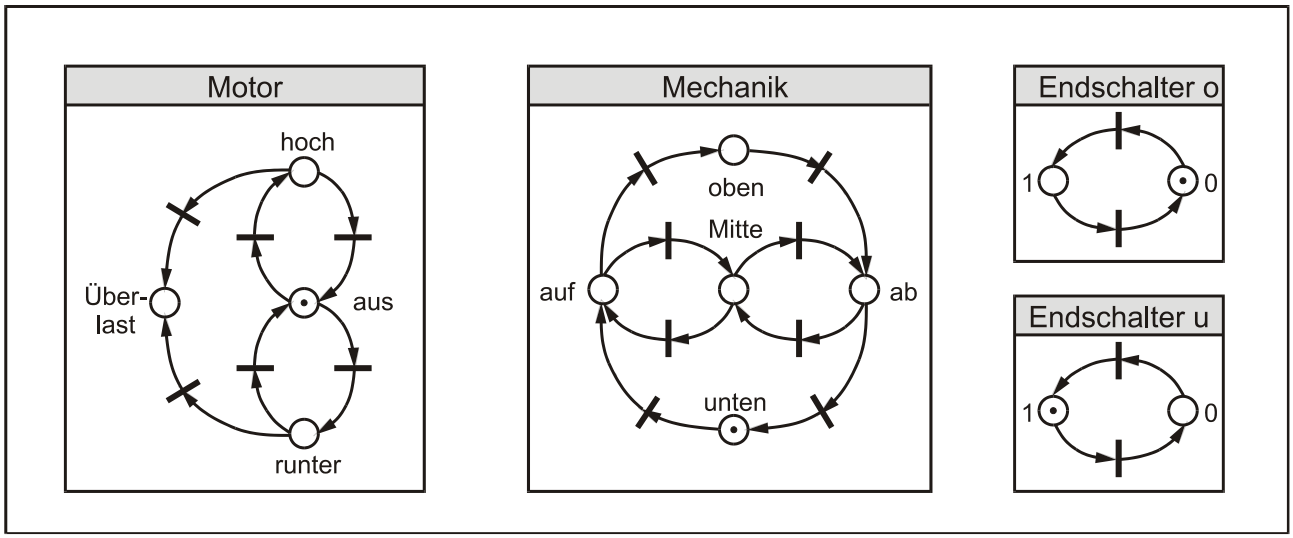


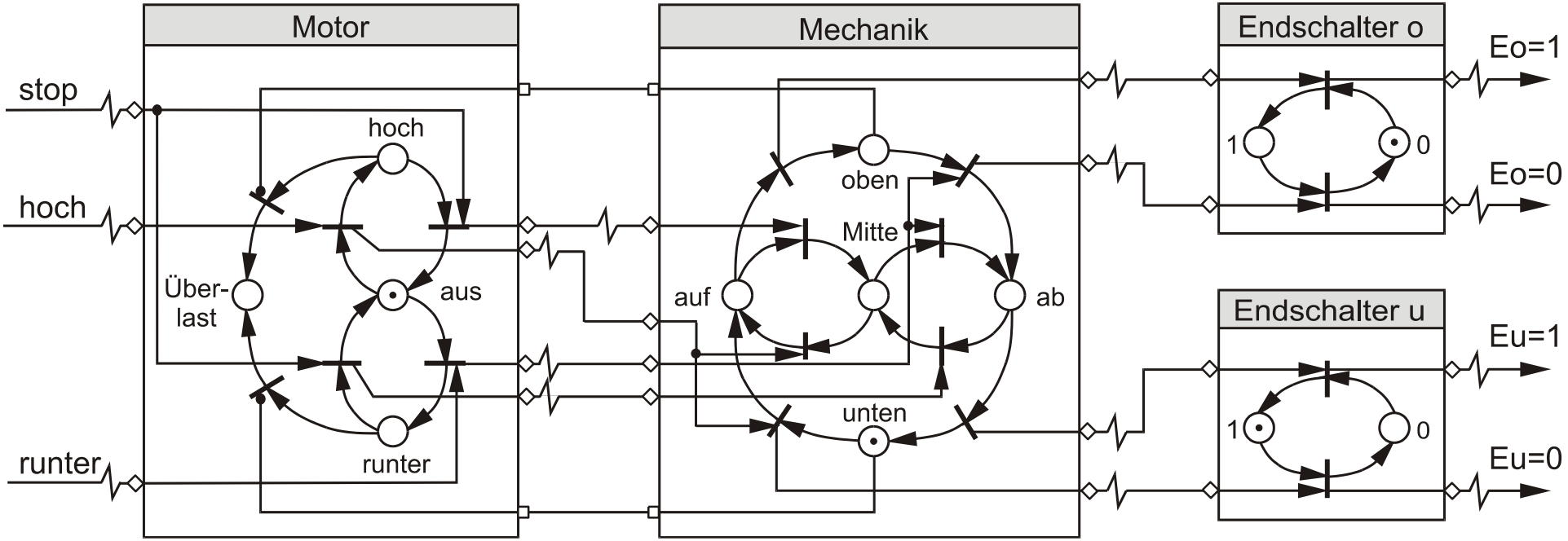
Ein Hubtisch ist ein Förderelement, das dazu dient, ein Stückgut von einer Förderstrecke auf eine andere Förderstrecke zu transportieren, die auf einer anderen Höhe liegt. Der Hubtisch wird von einem Elektromotor auf- und abbewegt. Die obere und untere Endposition werden durch Endschalter gemeldet.

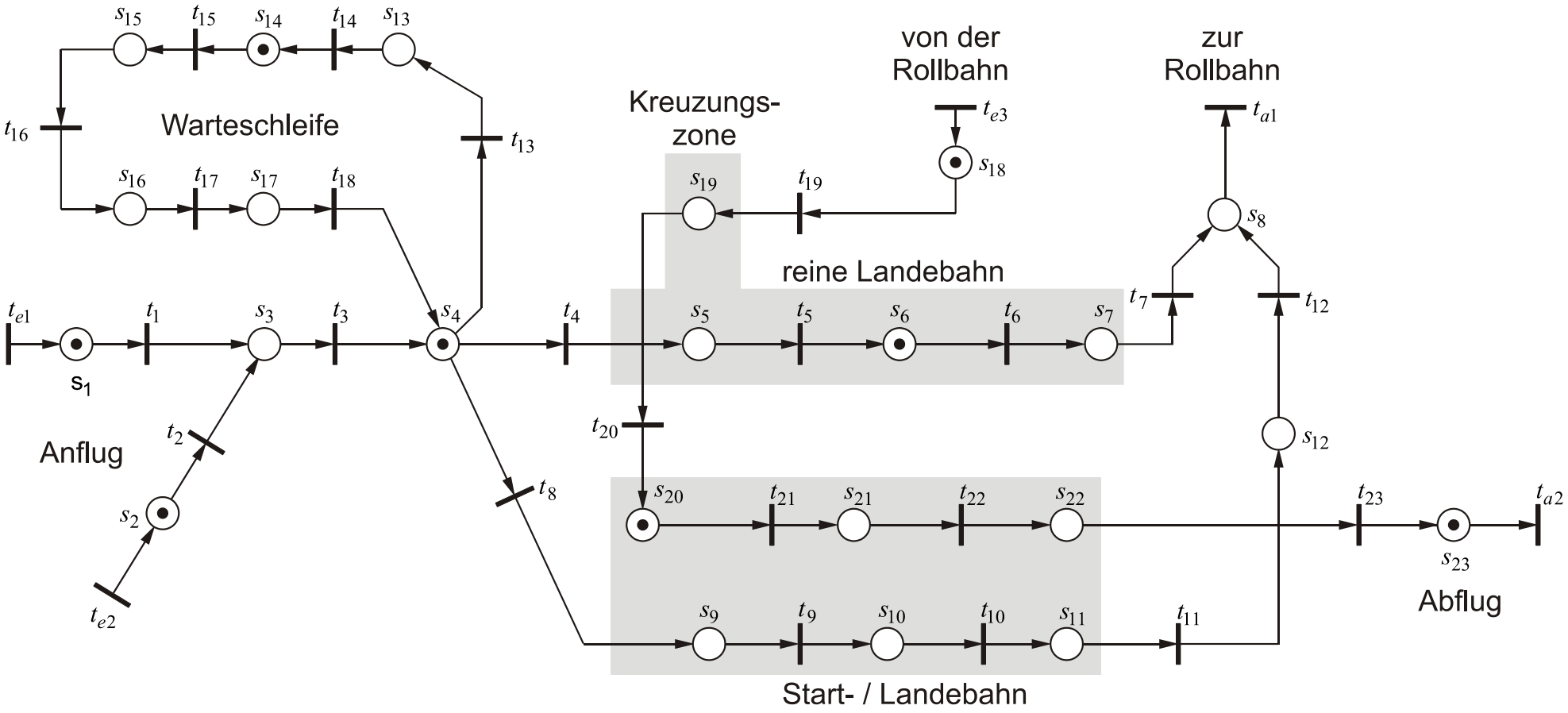


Modellierung als NCE-System

Separate NCE-Module für Prozess, Aktoren und Sensoren:




Motor
Mechanik
Endschalter o
Endschalter u
stop
hoch
runter
hoch
Über-
last
aus
runter
oben
Mitte
auf
ab
unten
1
0
Eo=1
Eo=0
Eu=1
Eu=0


s_{15} t_{15} s_{14} t_{14} s_{13}
Warteschleife
t_{16}
t_{13}
s_{16} t_{17} s_{17} t_{18}
von der Rollbahn
t_{e3}
s_{18}
Kreuzungs-zone
s_{19} t_{19}
zur Rollbahn
t_{a1}
s_8
reine Landebahn
t_{e1} t_1 s_3 t_3 s_4 t_4 s_5 t_5 s_6 t_6 s_7 t_7 t_{12}
s_1
t_{20}
s_{12}
Anflug
t_2
t_8
s_{20} t_{21} s_{21} t_{22} s_{22} t_{23} t_{a2}
s_2
s_{23}
Abflug
t_{e2}
s_9 t_9 s_{10} t_{10} s_{11} t_{11}
Start- / Landebahn

- **Graph G:**

  Ein **endlicher Graph** G lässt sich als ein Tupel

  $$G = (V, E)$$

  beschreiben, wobei gilt

  V: endliche nichtleere Knotenmenge mit $V = \{k_1, \ldots, k_n\}$,

  E: endliche Kantenmenge mit $E \subseteq \left\{ \{k_i, k_j\} \middle| k_i, k_j \in V \right\}$.

  Ein **bipartiter Graph** ist ein Graph, der zwei verschiedene Knotentypen aufweist, das heißt $V = V_1 \cup V_2$. Kanten sind dann nur zwischen verschiedenartigen Knoten vorhanden: $E \subseteq \left\{ \{k_i, k_j\} \middle| k_i \in V_1, k_j \in V_2 \right\}$.

  Ein **gerichteter Graph (Digraph)** enthält nur gerichtete Kanten, das heißt die Elemente von E bestehen aus geordneten Paaren: $E \subseteq V \times V$. Analog gilt für einen gerichteten bipartiten Graphen: $E \subseteq V_1 \times V_2 \cup V_2 \times V_1$.

  Existiert zwischen zwei Knoten jeweils nur eine Kante in einer Richtung, so spricht man von einem **einfachen Graphen**.

  Ein Knoten eines gerichteten Graphen G wird als eine **Quelle** bezeichnet, wenn keine Kante an ihm ankommt, bzw. als eine **Senke**, wenn keine Kante von ihm ausgeht.

  Ein **gewichteter Graph** ist ein Graph, dessen Kanten zusätzlich jeweils eine Zahl, das sogenannte Kantengewicht, zugeordnet ist.

- **Pfade im Graphen G:**

  Ein **Pfad** eines gerichteten Graphen G wird durch eine Folge $\rho = (k_1, \ldots, k_p)$ $(p > 1)$ von Knoten beschrieben, wobei jeweils eine gerichtete Kante vom Knoten k_i zum Knoten k_{i+1} $(i = 1, \ldots, p-1)$ existiert.

  Die **Länge eines Pfades** ist gleich der Anzahl der in ihm enthaltenen Kanten (also $p-1$) .

Das **Gewicht ω eines Pfades** ist die Summe der Kantengewichte der an dem Pfad beteiligten Kanten, das **Durchschnittsgewicht μ eines Pfades** ist dann der Quotient seines Gewichts und seiner Länge:

$$\mu = \frac{\omega}{p-1} \quad .$$

Ein **elementarer Pfad** ist ein Pfad, der keinen Knoten mehrfach enthält.

Ein **Zyklus** ist ein Pfad, der geschlossen ist und für den somit $k_p = k_1$ gilt.

Der Zyklus, der das maximale Durchschnittsgewicht aller in G enthaltenen Zyklen besitzt, heißt **kritischer Zyklus.** Die Knoten und Kanten aller kritischen Zyklen von G bilden dann den **kritischen Graphen** G_{krit}.

- **Grapheneigenschaften:**

  Ein gerichteter Graph G heißt **zusammenhängend**, wenn von jedem Knoten ohne Berücksichtigung der Kantenrichtungen ein Pfad zu jedem anderen Knoten führt.

  Ein gerichteter Graph G heißt **stark zusammenhängend**, wenn von jedem Knoten ein gerichteter Pfad zu jedem anderen Knoten führt.

  Gilt der starke Zusammenhang nur für eine Teilmenge der Knoten, so bildet diese Teilmenge für sich eine **stark zusammenhängende Komponente** des Graphen G.

  Verliert eine stark zusammenhängende Komponente von G bei Hinzunahme eines beliebigen weiteren Knotens von G diese Eigenschaft, so ist sie eine maximale stark zusammenhängende Komponente und wird auch als **starke Komponente** von G bezeichnet.

- **Kondensation G^K eines Graphen G:**

  Die **Kondensation** G^K eines gerichteten Graphen G stellt einen aus G gewonnenen reduzierten Graphen dar, dessen Knoten den starken Komponenten von G entsprechen und dessen Kanten den Zusammenhang zwischen den starken Komponenten untereinander wiedergeben. So besitzt G^K genau dann eine gerichtete Kante zwischen seinem Knoten k_i^K und k_j^K, wenn in G mindestens ein Pfad existiert, der einen Knoten aus seiner starken Komponente k_i^K mit einem aus der starken Komponente k_j^K verbindet.

a) Algebraische Bestimmung der Kondensation G^K:

Algebraisch lässt sich die Kondensation eines Graphen G mit Hilfe der sogenannten **Adjazenzmatrix** $\underline{A}$ (vgl. auch die Vorlesung „Modellbildung und Identifikation“) bestimmen. Diese gibt allgemein die Verbindung zweier Knoten über Kanten wieder:

$$\underset{(n,n)}{\underline{A}} := \left[a_{ij} \right] \text{ mit } a_{ij} = \begin{cases} m, \text{ falls } m \text{ Kanten vom Knoten } i \text{ zum Knoten } j \text{ führen,} \\ 0, \text{ sonst.} \end{cases}$$

Bei einfachen Graphen mit nur einer Kante zwischen zwei Knoten wird anstelle der Kantenanzahl häufig auch das Gewicht der Kante eingetragen.

Betrachtet man speziell einfache gerichtete Graphen, so gibt $\underline{A}$ offenbar alle Pfade der Länge 1 zwischen den vorhandenen n Knoten an. Folglich lassen sich mit Hilfe eines **logischen Matrizenprodukts**

$$\underline{A} \odot \underline{A} := \underline{A}^2 = \left[a_{ij} \right] \text{ mit } a_{ij} = \left(a_{i1} \wedge a_{1j} \right) \vee \left(a_{i2} \wedge a_{2j} \right) \vee \cdots \vee \left(a_{in} \wedge a_{nj} \right)$$

alle Pfade der Länge 2, durch entsprechende $(k-1)$-fache Anwendung allgemein alle **Pfade der Länge** k durch die k-te Potenz von $\underline{A}$ berechnen:

$$\underline{A}^k = \underline{A} \odot \underline{A}^{k-1}.$$

Da für die Kondensation von G der Zusammenhang der Knoten über beliebig lange Pfade interessiert, werden alle Pfade durch die sogenannte **Transitive Hülle** $\underline{A}^+$ zusammengefasst. Diese ist als Vereinigung (als elementweises ODER der Matrixelemente zu berechnen) aller Potenzen $\underline{A}^k$ von $\underline{A}$ definiert, wobei sich vereinfachend ausnutzen lässt, dass die elementaren Pfade in einem gerichteten Graphen mit n Knoten nur die maximale Länge n besitzen können:

$$\underline{A}^+ := \bigcup_{k=1}^{\infty} \underline{A}^k \overset{!}{=} \bigcup_{k=1}^{n} \underline{A}^k.$$

Zur Ermittlung der starken Komponenten erweitert man dann die Transitive Hülle sinnvollerweise noch um die Einheitsmatrix $\underline{I}$ zur sogenannten **Reflexiv Transitiven Hülle** $\underline{A}^*$:

$$\underline{A}^* := \underline{A}^+ \cup \underline{I}.$$

Eine einfache schematische Untersuchung der Elemente von $\underline{A}^* = \left[a_{ij}^* \right]$ führt dann zu der gesuchten Kondensation G^K. Hierbei gelten zur Bestimmung der starken Komponenten und der sie verbindenden Kanten folgende Regeln zur Analyse der Elemente a_{ij}^* $(i = 1 \ldots n, j = 1 \ldots n)$:

- Gilt für zwei Nebendiagonalelemente von $\underline{A}^*$ die Bedingung $a_{ij}^* = a_{ji}^* = 1$, so gehören die Knoten i und j zu der gleichen starken Komponente.
- Ist lediglich das Hauptdiagonalelement $a_{ii}^* = 1$, gilt jedoch für die übrigen Nebendiagonalelemente $a_{ij}^* = 0$, so bildet der Knoten i eine eigene starke Komponente.
- Ist $a_{ij}^* = 1$ und $a_{ji}^* = 0$, so liegt in der Kondensation eine gerichtete Kante vor von der starken Komponente mit dem Knoten i zu der starken Komponente mit dem Knoten j.

b) Algorithmus zur grafischen Bestimmung der Kondensation G^K:

1. Wahl eines Startknotens
   Man wähle einen beliebigen Knoten $i \in V$ und kennzeichne ihn sowohl mit $\oplus$ als auch mit $\ominus$.
2. Vorwärtsmarkierung
   Man kennzeichne jeden Knoten j, der von i aus erreicht werden kann, d.h. für den ein Pfad $\rho = (i, \ldots, j)$ mit $j \in V$ existiert, mit $\oplus$.
3. Rückwärtsmarkierung
   Man kennzeichne jeden Knoten j, von dem aus i erreicht werden kann, d.h. für den ein Pfad $\rho = (j, \ldots, i)$ mit $j \in V$ existiert, mit $\ominus$.
4. Alle diejenigen Knoten, die mit $\oplus$ und $\ominus$ gekennzeichnet sind, bilden eine starke Komponente von G.
5. Für die Knoten des Graphen, die nicht in bereits ermittelten starken Komponenten enthalten sind, sind die Schritte 1. - 4. beginnend mit einem beliebigen dieser Knoten zu wiederholen.
6. Zwei starke Komponenten sind dann durch eine Kante verbunden, wenn es mindestens eine Kante zwischen ihren jeweils enthaltenen Knoten gibt.

In den folgenden Definitionen wird eine Markierung M häufig algebraisch durch ihren Markierungsvektor $\underline{m}$ beschrieben (vgl. Beiblatt AEH 2-11 bzw. 2-12). Gleiches gilt für eine Transition t_j, die algebraisch durch ihren Transitionenvektor $\underline{t}_j$ repräsentiert wird.

Schaltsequenz:

Eine Transitionenfolge $\sigma = t_1', t_2', \ldots, t_x'$ mit $t_i' \in T$ heißt **Schaltsequenz**.

Eine **Schaltsequenz** σ heißt **anwendbar** unter der Markierung M, wenn gilt:

$$\forall i \in \{1, 2, \ldots, x\}: \quad \underline{0} \le \underline{m} + \sum_{l=1}^{i} \underline{t}_l' \le \underline{k} \quad .$$

Erreichbarkeit:

Eine **Markierung** M eines Petri-Netzes N heißt **erreichbar** unter der Anfangsmarkierung M_0, falls eine anwendbare Schaltsequenz σ existiert, die M_0 in M überführt.

Erreichbarkeitsmenge:

Die **Erreichbarkeitsmenge** $R_N(M_0)$ ist die Menge aller im Petri-Netz N von M_0 aus erreichbaren Markierungen, einschließlich M_0 selbst.

Erreichbarkeitsgraph:

Der Erreichbarkeitsgraph ist ein Tupel

$$E_N = (A, B) \quad ,$$

wobei gilt

A: Knotenmenge gemäß $A = R_N(M_0)$,

B: Kantenmenge gemäß $B = \left\{ \left(M, t_j, M' \right) \middle| M \in A \wedge \underline{m}' = \underline{m} + \underline{t}_j \right\}$.

Reversibilität:

Ein Petri-Netz N heißt **reversibel**, wenn gilt:

$$\forall M_1, M_2 \in R_N(M_0): \quad M_1 \in R_N(M_2) \quad .$$

Lebendigkeit:

Eine **Transition** $t_j \in T$ eines Petri-Netzes N heißt **lebendig** unter M_0 , wenn bei jeder Folgemarkierung von M_0 mindestens eine Markierung M' erreichbar ist, unter der t_j aktiviert ist, d.h. wenn gilt:

$$\forall M \in R_N(M_0): \quad \exists M' \in R_N(M): \quad t_j \text{ ist aktiviert unter } M' \ .$$

Ein **Petri-Netz** N heißt **lebendig**, wenn alle Transitionen t_j von N unter M_0 lebendig sind.

Tote Transition/Markierung:

Eine **Transition** $t_j \in T$ eines Petri-Netzes N heißt **tot** unter M_0, wenn t_j unter M_0 und allen Folgemarkierungen von M_0 nicht aktiviert ist, d.h. wenn gilt:

$$\nexists M \in R_N(M_0): \quad t_j \text{ ist aktiviert unter } M \ .$$

Eine **Markierung** M heißt **tot**, wenn unter M keine Transition aktiviert ist.

Verklemmung:

In einem Petri-Netz N liegt eine **totale Verklemmung** vor, wenn eine tote Markierung M erreicht wird.

Wird in einem Petri-Netz N eine Markierung M erreicht, die zwar nicht tot ist, unter der aber eine oder mehrere Transitionen tot sind, liegt eine **partielle Verklemmung** vor.

Beschränktheit:

Eine **Stelle** $s_i \in S$ eines Petri-Netzes N heißt k-**beschränkt** bei M_0, falls gilt:

$$\exists k \in \mathbb{N}, \quad \forall M \in R_N(M_0): \quad M(s_i) \le k \ .$$

Ein **Netz** N heißt **beschränkt** bei M_0, wenn jede Stelle $s_i \in S$ k-beschränkt ist.

Ein **Netz** N heißt **sicher**, wenn jede Stelle $s_i \in S$ 1-beschränkt ist.

Gegeben sei der Erreichbarkeitsgraph $E_N = (A, B)$ eines Petri-Netzes N mit (vgl. AEH 4-4):

$$A = R_N(M_0) \quad ,$$

$$B = \left\{ \left(M, t_j, M' \right) \middle| M \in A \wedge \underline{m}' = \underline{m} + \underline{t}_j \right\} \quad .$$

Folgende Eigenschaften von N lassen sich dann an E_N ablesen:

- **Erreichbarkeit:**

  E_N besitzt einen Knoten M, d.h. $M \in A$.
  $\Leftrightarrow$ Die Markierung M ist in N erreichbar, d.h. $M \in R_N(M_0)$.

- **Reversibilität:**

  Der Erreichbarkeitsgraph E_N eines Netzes N ist stark zusammenhängend.
  $\Leftrightarrow$ N ist reversibel.

- **Lebendigkeit:**

  - Keine Kante von E_N ist mit t_j beschriftet.
    $\Leftrightarrow$ t_j ist eine tote Transition.
  - E_N besitzt einen Knoten M ohne auslaufende Kante.
    $\Leftrightarrow$ In N ist eine totale Verklemmung möglich.
  - Der Erreichbarkeitsgraph E_N ist stark zusammenhängend und jede Transition tritt als Beschriftung an mindestens einer Kante von E_N auf.
    $\Rightarrow$ N ist lebendig.

  Eine **starke Komponente** des Erreichbarkeitsgraphen E_N heißt **lebendig**, wenn sie zu jeder Transition $t_j \in T$ mindestens eine entsprechende Kante enthält. Dann gilt:

  - Jede Senke der Kondensation E_N^K des Erreichbarkeitsgraphen E_N ist eine lebendige Komponente von E_N.
    $\Leftrightarrow$ N ist lebendig.

Gegeben sei ein Petri-Netz N mit zugehöriger Netzmatrix $\underline{N}$.

S-Invariante:

Ein Vektor $\underline{i}_S \in \mathbb{Z}^{|S|}$ heißt **S-Invariante** von N, falls gilt: $\underline{N}^T \cdot \underline{i}_S = \underline{0}$.

In N bleibt dann bei jedem Schaltvorgang die mit $\underline{i}_S$ gewichtete Anzahl von Marken konstant: $\underline{m}^T \cdot \underline{i}_S = \underline{m}_0^T \cdot \underline{i}_S$.
Gilt daher für die S-Invariante $\underline{i}_S \in \{0,1\}^{|S|}$, so beschreibt sie eine Menge von Stellen, bei der bei Schaltvorgängen die Gesamtzahl von Marken unverändert bleibt, die Marken also nur zwischen diesen Stellen verschoben werden.

T-Invariante:

Ein Vektor $\underline{i}_T \in \mathbb{Z}^{|T|}$ heißt **T-Invariante** von N, falls gilt: $\underline{N} \cdot \underline{i}_T = \underline{0}$.

Eine T-Invariante gibt an, wie oft jede Transition des Petri-Netzes schalten muss, um eine beliebige Markierung zu reproduzieren.

An den S- und T-Invarianten von N lassen sich dann folgende Eigenschaften ablesen:

- **Beschränktheit:**

  Besitzt N eine positive S-Invariante, so ist N beschränkt:

  $$\exists\, \underline{i}_S \in (\mathbb{N} \setminus \{0\})^{|S|} : \quad \underline{N}^T \cdot \underline{i}_S = \underline{0} \;\Rightarrow\; N \text{ ist beschränkt.}$$

- **Reversibilität:**

  Jedes reversible Petri-Netz besitzt eine nichtnegative T-Invariante:

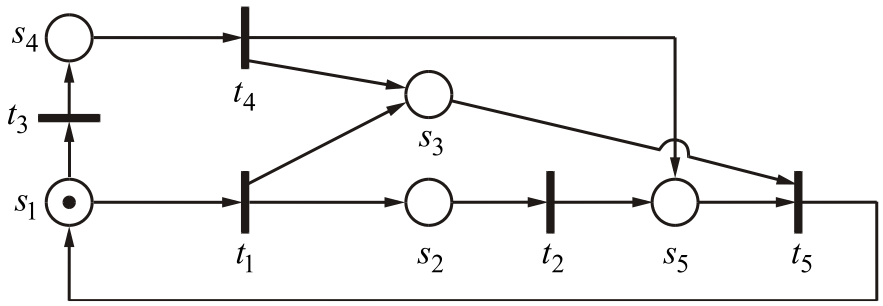
  $$N \text{ ist reversibel.} \Rightarrow \exists\, \underline{i}_T \in \mathbb{N}^{|T|} : \quad \underline{N} \cdot \underline{i}_T = \underline{0} \ .$$

- **Lebendigkeit:**

  Jedes lebendige und beschränkte Petri-Netz besitzt eine positive T-Invariante:

  $$N \text{ ist lebendig und beschränkt.} \Rightarrow \exists\, \underline{i}_T \in (\mathbb{N} \setminus \{0\})^{|T|} : \quad \underline{N} \cdot \underline{i}_T = \underline{0} \ .$$

Beispiel zur Berechnung der T-Invarianten.



$$\underline{N} = \begin{bmatrix} -1 & 0 & -1 & 0 & 1 \\ 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & 1 & -1 \end{bmatrix} \qquad \underline{N} \cdot \underline{i}_T = \underline{0} \quad \text{liefert:}$$

$$\begin{array}{rrrrrcll} -i_{T1} & & -i_{T3} & & +i_{T5} & = & 0 & (1) \\ i_{T1} & -i_{T2} & & & & = & 0 & (2) \\ i_{T1} & & & +i_{T4} & -i_{T5} & = & 0 & (3) \\ & & i_{T3} & -i_{T4} & & = & 0 & (4) \\ & i_{T2} & & +i_{T4} & -i_{T5} & = & 0 & (5) \end{array}$$

I .) gewählt: $i_{T1} = 0$

$\overset{(2)}{\rightarrow} \; i_{T2} = 0$

$\overset{(1)}{\rightarrow} \; i_{T3} = i_{T5}$ beliebig, z.B. $i_{T3} = 1$

$\overset{(4)}{\rightarrow} \; i_{T4} = i_{T3} = 1$

$$\underline{i}_T^{\,I} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

II .) gewählt: $i_{T4} = 0$

$\overset{(3)}{\rightarrow} \; i_{T1} = i_{T5}$ beliebig, z.B. $i_{T1} = 1$

$\overset{(2)}{\rightarrow} \; i_{T2} = i_{T1} = 1$

$\overset{(4)}{\rightarrow} \; i_{T3} = i_{T4} = 0$

$$\underline{i}_T^{\,II} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Allgemein: Linearkombination linear unabhängiger T-Invarianten liefert ebenfalls T-Invariante

$\underline{i}_T = c_1 \cdot \underline{i}_T^I + c_2 \cdot \underline{i}_T^{II}$ mit $c_1, c_2 \in \mathbb{Z}$

gewählt $c_1, c_2 = 1$

$$\underline{i}_T = \underline{i}_T^I + \underline{i}_T^{II} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 2 \end{bmatrix}$$

Anmerkung: T-Invarianten bei bestimmten Markierungen unter Umständen nicht realisierbar $\Rightarrow$ nur notwendige Bedingungen für die PN-Analyse herleitbar.

Gegeben sei ein markiertes Petri-Netz N.

Deadlock:

Eine nichtleere Stellenmenge S_D von N heißt **Deadlock**, falls jede Transition, die Marken in S_D hineinschaltet, auch Marken aus S_D entnimmt, d.h.

$$\forall t_j \in T : t_j \in {}^\bullet S_D \Rightarrow t_j \in S_D {}^\bullet \quad .$$

Ein Deadlock ist also eine Stellenmenge, die nie wieder markiert werden kann, wenn sie einmal unmarkiert ist.

Trap:

Eine nichtleere Stellenmenge S_T von N heißt **Trap**, falls jede Transition, die beim Schalten Marken aus S_T entnimmt, dabei auch Marken in S_T hinzufügt, d.h.

$$\forall t_j \in T : t_j \in S_T {}^\bullet \Rightarrow t_j \in {}^\bullet S_T \quad .$$

Ein Trap ist somit eine Stellenmenge, die nie mehr alle Marken verlieren kann, wenn sie einmal mindestens eine Marke enthalten hat.

Mit Deadlocks und Traps lässt sich für das markierte Petri-Netz N folgende Eigenschaft angeben:

- Falls jeder Deadlock von N einen unter M_0 markierten Trap enthält, so gibt es in der Erreichbarkeitsmenge $R_N(M_0)$ keine tote Markierung.

Speziell für ein Free-Choice-Netz N gilt:

- **Lebendigkeit eines Free-Choice-Netzes**

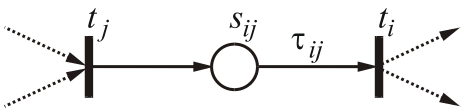
  Jeder Deadlock S_D eines Free-Choice-Netzes N enthält als Teilmenge einen unter M_0 markierten Trap $S_T\,(S_T \subseteq S_D)$.
  $\Leftrightarrow$ N ist lebendig.

Voraussetzungen:

- lebendige und sichere zeitbewertete Synchronisationsgraphen
- Doppelindizierung der Stellen so, dass ihre Lage zwischen Transitionen deutlich wird:

$$s_{ij} \in {}^\bullet t_i \wedge s_{ij} \in t_j{}^\bullet$$

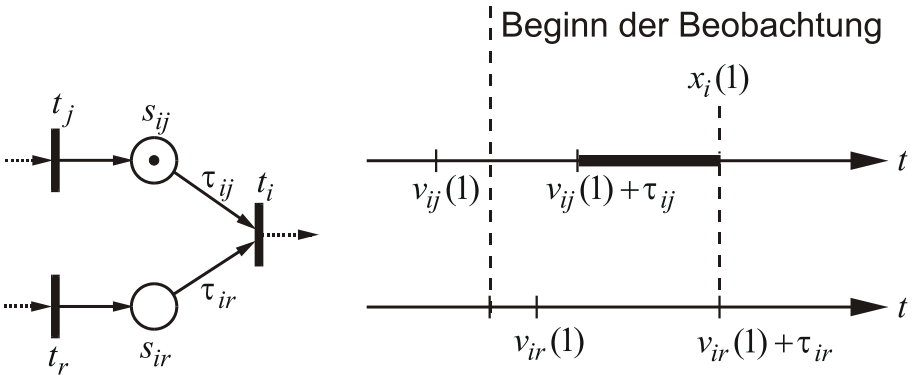
- Zeitbewertung der Prekanten entsprechend, dabei Vereinbarung:
  - $\tau_{R,ij} =: \tau_{ij}$
  - $\tau_{L,ij} = \infty$ (keine Limitierung)



Schaltvorgänge der Transitionen:

- $x_i(k)$: Zeitpunkt, an dem die Transition t_i zum k-ten Mal schaltet.
- $v_{ij}(k)$: Zeitpunkt, an dem die Stelle s_{ij} zum k-ten Mal belegt wird.

Beispiel:



$$x_i(1) = \max\ (\underbrace{v_{ir}(1) + \tau_{ir}}_{=x_r(1)}\ ,\ \underbrace{v_{ij}(1) + \tau_{ij}}_{=x_j(0)})$$

allgemein gilt (Muss-Schaltregel):

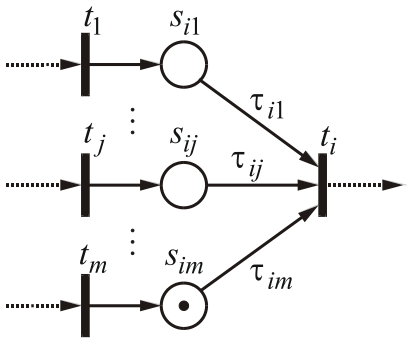
$$x_i(k+1) = \max\left\{ \max_{M_0(s_{ij})=0} \left\{ x_j(k+1) + \tau_{ij} \right\}\ ,\ \max_{M_0(s_{ij})=1} \left\{ x_j(k) + \tau_{ij} \right\} \right\}$$

Max-Plus-Algebra

Die Operationen sowie die zugehörigen Eigenschaften der Max-Plus-Algebra sind im ersten Teil der folgenden Tabelle erläutert. Im zweiten Teil finden sich Grundlagen der Matrizenrechnung in der Max-Plus-Algebra, die wie in der konventionellen Algebra aufgebaut ist.

Operation / **Eigenschaft**	**Addition** $a \oplus b = \max(a,b)$	**Multiplikation** $a \otimes b = a + b$
Abgeschlossenheit	$a \oplus b \in \mathbb{R}_{\max}$	$a \otimes b \in \mathbb{R}_{\max}$
Kommutativität	$a \oplus b = b \oplus a$	$a \otimes b = b \otimes a$
Assoziativität	$(a \oplus b) \oplus c = a \oplus (b \oplus c)$	$(a \otimes b) \otimes c = a \otimes (b \otimes c)$
Distributivität	$(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$	
Neutrales Element	$a \oplus \varepsilon = a$	$a \otimes 0 = a$
Absorbierendes Element		$a \otimes \varepsilon = \varepsilon$
Inverses Element		$a \otimes (-a) = 0$
Idempotenz	$a \oplus a = a$	
Matrizenrechnung		
	$\underline{A} \oplus \underline{B}$, $(\underline{A} \oplus \underline{B})_{ij} = a_{ij} \oplus b_{ij}$	$\underline{A} \otimes \underline{B}$, $(\underline{A} \otimes \underline{B})_{ij} = \bigoplus_k (a_{ik} \otimes b_{kj})$
Neutrales Element	$\underline{A} \oplus \underline{N} = \underline{A}$, $(\underline{N})_{ij} = \varepsilon$	$\underline{A} \otimes \underline{I} = \underline{A}$, $(\underline{I})_{ij} = \begin{cases} 0, i = j \\ \varepsilon, \text{sonst} \end{cases}$
Absorbierendes Element		$\underline{A} \otimes \underline{N} = \underline{N}$
Idempotenz	$\underline{A} \oplus \underline{A} = \underline{A}$	
Matrixpotenz		$\underline{A}^k = \underbrace{\underline{A} \otimes \ldots \otimes \underline{A}}_{k}$, $\underline{A}^0 = \underline{I}$

(Bezeichnung: $\varepsilon := -\infty$)



$$x_i(k+1) = \left[\bigoplus_{M_0(s_{ij})=0} \left(x_j(k+1) \otimes \tau_{ij} \right) \right] \oplus \left[\bigoplus_{M_0(s_{ij})=1} \left(x_j(k) \otimes \tau_{ij} \right) \right]$$

Die Gleichungen für die Schaltzeitpunkte der einzelnen Transitionen lassen sich übersichtlich in Matrizenform zusammenfassen:

- **Implizite Darstellung:**

$$\underline{x}(k+1) = \underline{A}_0 \underline{x}(k+1) \oplus \underline{A}_1 \underline{x}(k)$$

mit:
$$\left(\underline{A}_0\right)_{ij} = \begin{cases} \tau_{ij} \ , & M_0\left(s_{ij}\right) = 0 \\ \varepsilon \ , & \text{sonst} \end{cases}$$

$$\left(\underline{A}_1\right)_{ij} = \begin{cases} \tau_{ij} \ , & M_0\left(s_{ij}\right) = 1 \\ \varepsilon \ , & \text{sonst} \end{cases}$$

- **Rekursive Darstellung:**

$$\underline{x}(k+1) = \underline{A}\,\underline{x}(k)$$

mit:
$$\underline{A} = \underbrace{\left(\underline{I} \oplus \underline{A}_0 \oplus \ldots \underline{A}_0^{n-1} \right)}_{=: \underline{A}_0^*} \underline{A}_1 = \underline{A}_0^* \underline{A}_1$$

Eigenwert, Eigenvektor:

Existieren zu einer Matrix $\underline{A} \in \left(\mathbb{R}_{\max} \right)^{n \times n}$ ein $\lambda \in \mathbb{R}_{\max}$ und ein $\underline{v} \in \left(\mathbb{R}_{\max} \right)^{n}$, so dass

$$\underline{A} \otimes \underline{v} = \lambda \otimes \underline{v}$$

gilt, dann heißen λ **Eigenwert** und $\underline{v}$ **Eigenvektor** von $\underline{A}$. Im Unterschied zur konventionellen Algebra können zu einem Eigenwert mehrere linear unabhängige Eigenvektoren existieren.

Eigenwerte und Eigenvektoren lassen sich wie folgt berechnen:

- **Eigenwerte**

  Für den Eigenwert der Matrix $\underline{A}$ mit der Dimension $n \times n$ gilt:

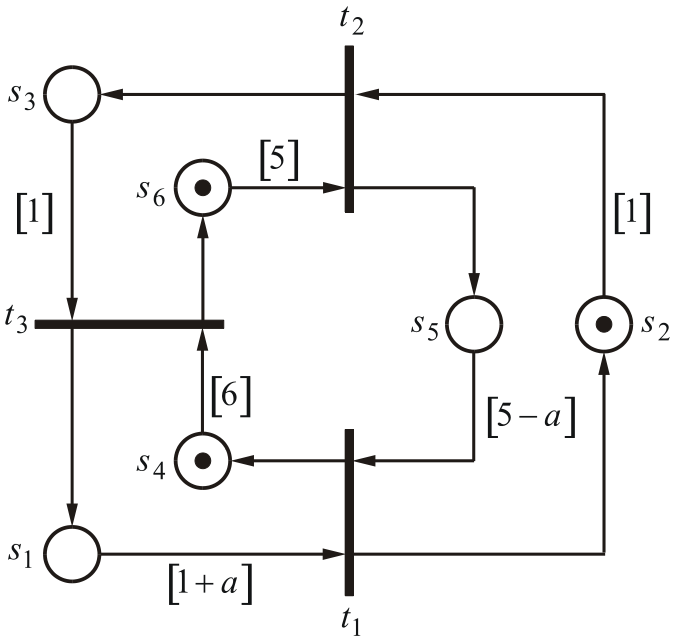
$$\lambda = \bigoplus_{j=1}^{n} \left(\bigoplus_{i=1}^{n} \left(\underline{A}^{j} \right)_{ii} \right)^{1/j} \quad .$$

- **Eigenvektoren**

  Mit der normalisierten Matrix $\underline{A}_n = (-\lambda) \otimes \underline{A}$ zu $\underline{A}$ und $\underline{A}_n^{+} = \underline{A}_n \oplus \underline{A}_n^{2} \oplus \ldots \oplus \underline{A}_n^{n}$ gilt:

$$\left(\underline{A}_n^{+} \right)_{ii} = 0 \quad \Rightarrow \quad \underline{v} = \left(\underline{A}_n^{+} \right)_{\bullet i} \quad ,$$

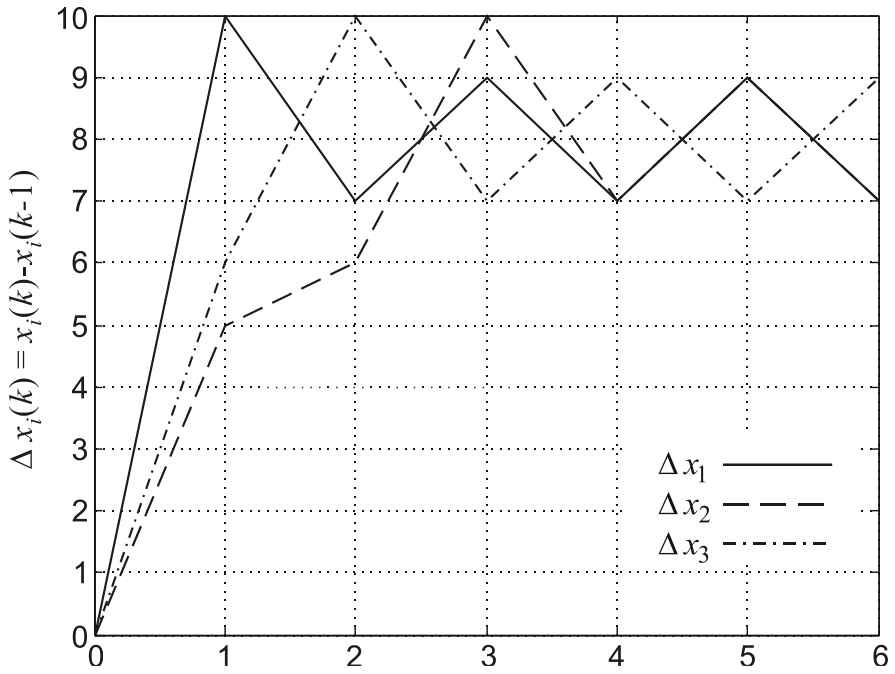
  d.h. die i-te Spalte von $\underline{A}_n^{+}$ ist ein Eigenvektor von $\underline{A}$ bzw. $\underline{A}_n$, falls das i-te Hauptdiagonalelement von $\underline{A}_n^{+}$ verschwindet.



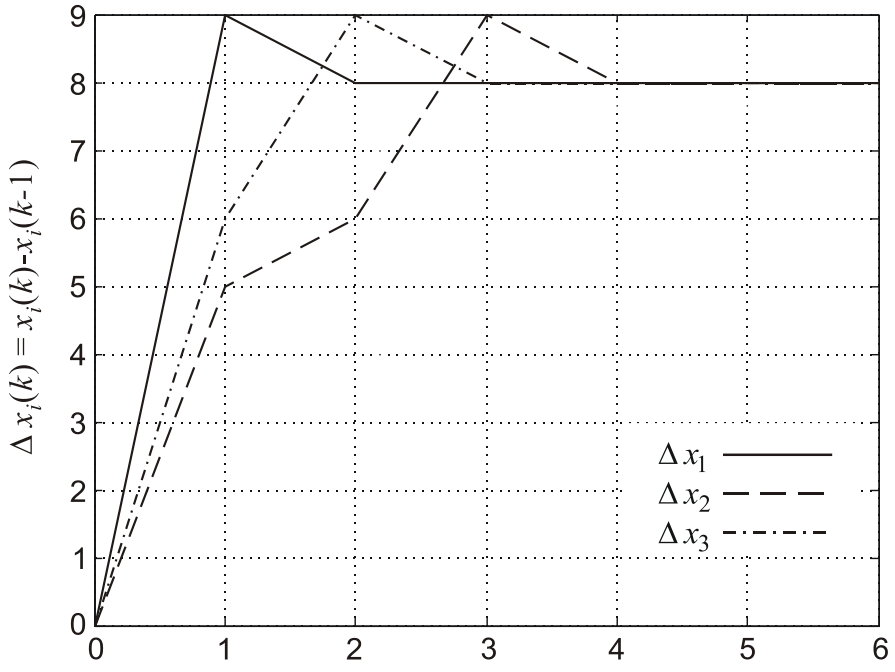
$$\underline{x}(k+1) = \underbrace{\begin{bmatrix} \varepsilon & 5-a & 1+a \\ \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 1 & \varepsilon \end{bmatrix}}_{\underline{A}_0} \underline{x}(k+1) \oplus \underbrace{\begin{bmatrix} \varepsilon & \varepsilon & \varepsilon \\ 1 & \varepsilon & 5 \\ 6 & \varepsilon & \varepsilon \end{bmatrix}}_{\underline{A}_1} \underline{x}(k)$$

$$\underline{x}(k+1) = \underline{A}_0^* \underline{A}_1 \underline{x}(k) = \begin{bmatrix} (6-a) \oplus (7+a) & \varepsilon & (10-a) \oplus (7+a) \\ 1 & \varepsilon & 5 \\ 6 & \varepsilon & 6 \end{bmatrix} \underline{x}(k)$$

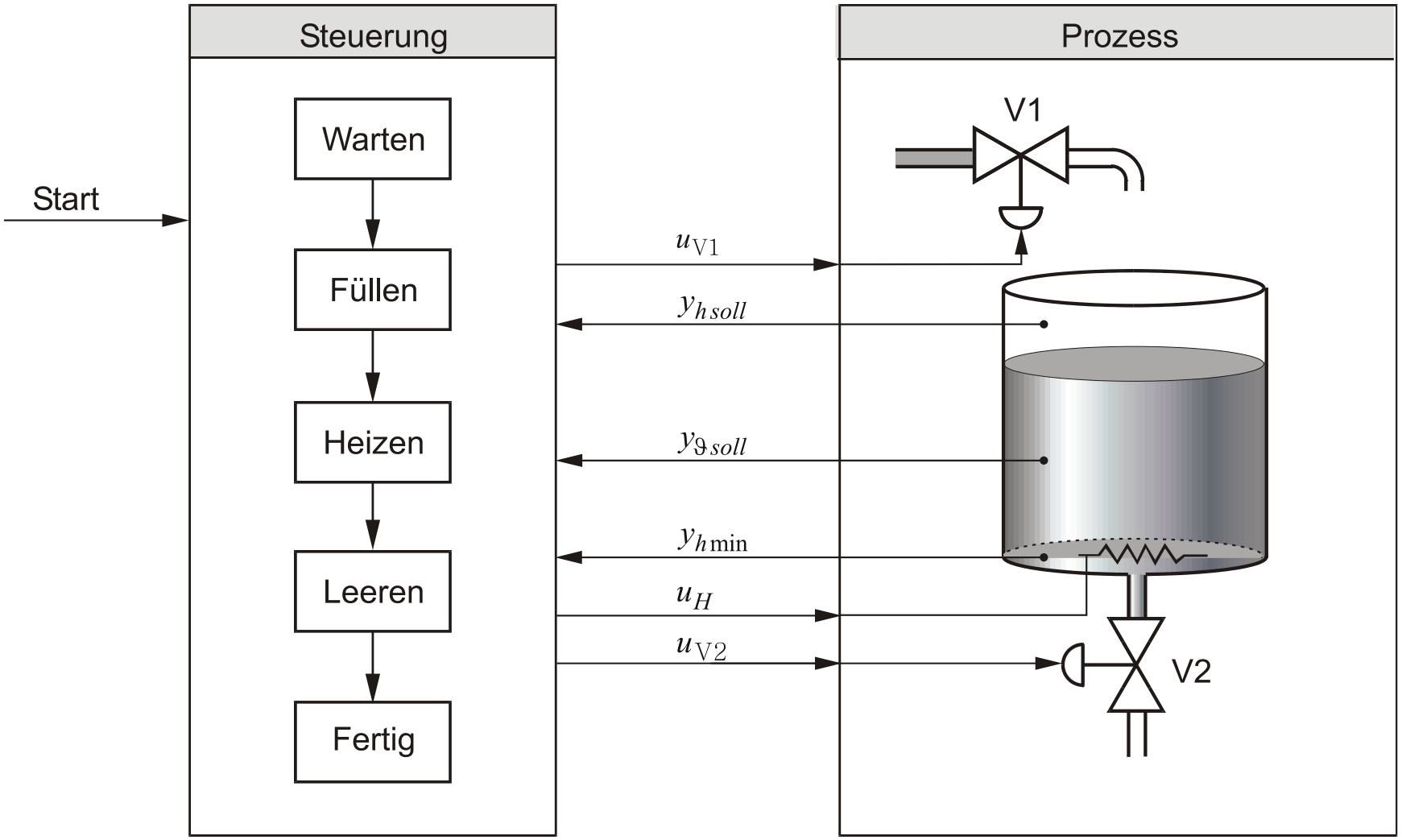
- $a = 0$, $\lambda = 8$, $\gamma = 2$



- $a = 1$, $\lambda = 8$, $\gamma = 1$



	Projektschritt		Dokumentation
1	**Anforderungsdefinition**	Beschreibung der: • technischen Randbedingungen • Sensorik / Aktorik • Betriebsarten • Funktionen	Pflichtenheft (VDI 3694): textuelle Kurzbeschreibung, Lagepläne, Bauteileliste Funktionspläne, Zustandsgraphen, Programmablaufpläne
2	**Projektierung**	*Hardware:* Funktionalität (Steuern, Melden, Regeln, Diagnose, Visualisieren ...), Speicherkapazität, Echtzeitfähigkeit, Ausbaufähigkeit, Kommunikation *Software:* Betriebssystem Anwenderprogramm: Funktionsklärung, Strukturierung Softwareentwurf	Hardwareunterlagen Programmdokumentation: Funktionsplan, Zustandsgraph, Petri-Netz IEC-Sprachen (AWL, KOP, FUP, ST, SFC)
3	**Test und Inbetriebnahme**	Hardware-Prüfung, Programmanalyse, Simulation, Debugging	Revidierte Dokumentation
4	**Wartung und Pflege**	Diagnose, Instandhaltung, Erweiterungen	Protokolle, Fehlermeldungen


Steuerung
Prozess
Warten
Start
Füllen
Heizen
Leeren
Fertig
V1
V2
u_{V1}
$y_{h\,soll}$
$y_{\vartheta\,soll}$
$y_{h\,min}$
u_H
u_{V2}

Regelung (IEC 60050-351-26-01) (feedback control):

Vorgang, bei dem fortlaufend eine variable Größe, die Regelgröße, erfasst, mit einer anderen variablen Größe, der Führungsgröße, verglichen und im Sinne einer Angleichung an die Führungsgröße beeinflusst wird.
Kennzeichen ist der Wirkungsablauf in einem geschlossenen Kreis.

Steuerung (IEC 60050-351-26-02) (open loop control):

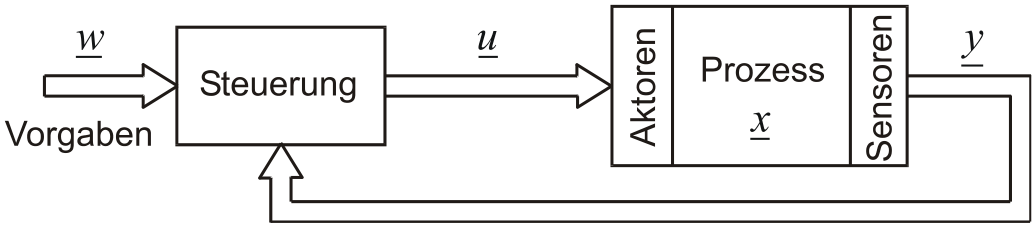
Vorgang in einem System, bei dem eine oder mehrere variable Größen als Eingangsgrößen andere Größen als Ausgangsgrößen auf Grund der dem System eigenen Gesetzmäßigkeiten beeinflussen.
Kennzeichen für das System ist der offene Wirkungsablauf.

Ereignisdiskrete Steuerung (discrete event control):

Ist die zielorientierte Umsetzung von Ereignisinformation in (steuernde) Aktion. Dabei werden von der Steuerung ausschließlich diskrete (häufig binäre) Signale produziert und ausgewertet; deshalb spricht man von einer ereignisdiskreten Steuerung (auch digitale oder binäre Steuerung).

- Ein Ereignis tritt immer dann auf, wenn ein diskretes Signal sich ändert, z.B. wenn ein binäres Signal von 0 auf 1 wechselt oder umgekehrt.
- Eine ereignisdiskrete Steuerung weist demnach einen geschlossenen Wirkungsablauf auf, bei dem allerdings der Soll-/Istwertvergleich fehlt.

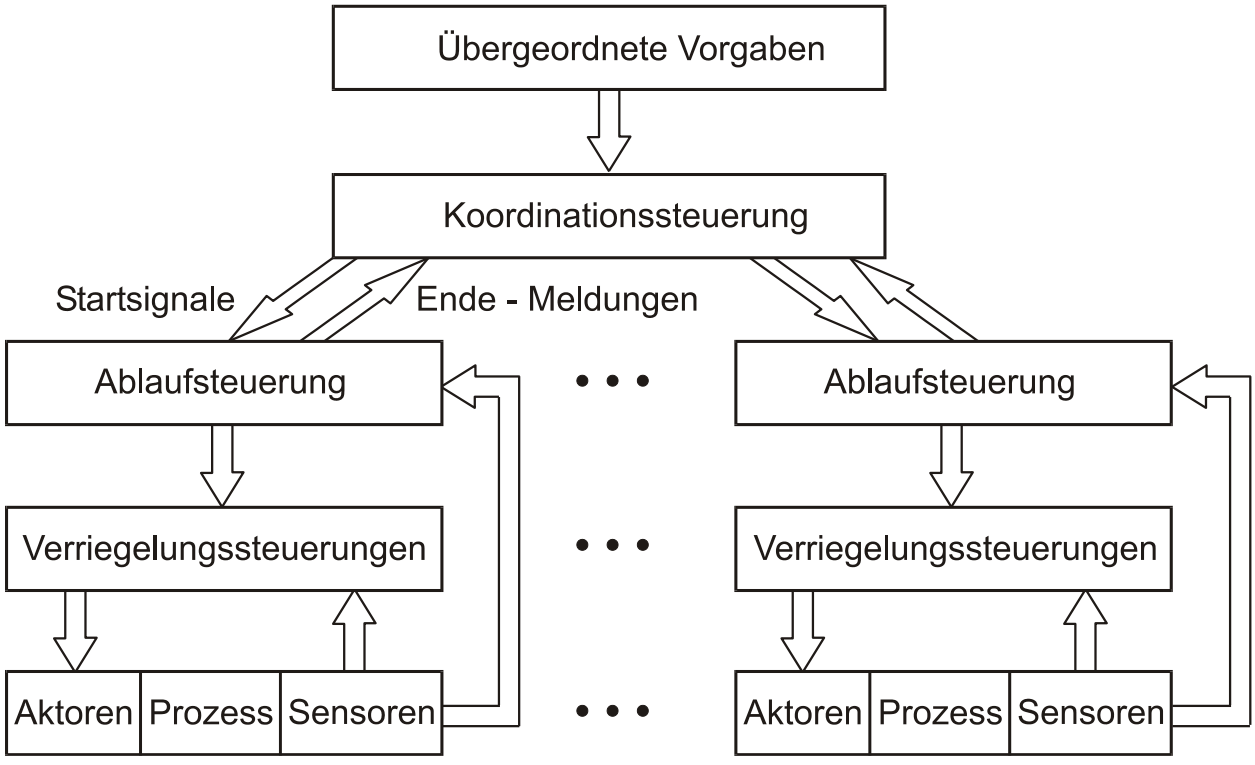
Steuerungsstruktur:



Steuerungsziele:

- **Verriegelung von Prozesszuständen:**
  Der Prozess darf bestimmte als verboten gekennzeichnete Zustände nicht erreichen.
- **Vorgabe von Abläufen:**
  Die Aufeinanderfolge von Prozesszuständen wird vorgegeben; z.B. ausgehend von Zustand 1 soll zuerst Zustand 2 und anschließend Zustand 3 erreicht werden.
- **Optimaler Prozessablauf:**
  Teilabläufe, die sich zumindest teilweise gegenseitig ausschließen (z.B. wegen Nutzung gemeinsamer Ressourcen), müssen so gestartet bzw. blockiert werden, dass ein optimaler Prozessablauf erzielt wird (Scheduling/Routing).

Steuerungshierarchie:



Unter der **Steuerungsspezifikation** versteht man die exakte und vollständige Beschreibung einer Automatisierungsaufgabe. Sie entsteht aus der Auswertung der im Pflichtenheft festgelegten Anforderungen und deren Strukturierung in Teilaufgaben.

Verschiedene Darstellungsformen sind gebräuchlich und werden in Kombination miteinander verwandt. Dabei ist darauf zu achten, dass für einzelne Teilaufgaben während des gesamten Entwurfsprozesses von der Spezifikation bis zur Implementierung die gleiche Darstellungsform benutzt wird.

Eine mögliche Darstellungsform ist die **textuelle Spezifikation**, bei der Anlagenteile und Anforderungen verbal beschrieben werden. Dies kann z.T. in tabellarischer Form geschehen und durch technische Skizzen unterstützt werden.

In der internationalen Norm **IEC 1131-3 (DIN EN 61131)** sind textuelle und graphische Beschreibungsformen zusammengefasst, in denen direkt Steuerungscode programmiert werden kann. Sie sind beliebig miteinander kombinierbar:

- **Anweisungsliste (AWL):** assemblerähnliche Sprache
- **Kontaktplan (KOP):** Darstellung nach Art eines Stromlaufplans
- **Funktionsplan (FUP):** Verknüpfung von Logikbausteinen
- **Ablaufsprache (SFC):** Graphische Programmiersprache
- **Strukturierter Text (ST):** PASCAL-ähnliches Sprachkonzept

Neben der Ablaufsprache gewinnen graphische Beschreibungsformen wie **Zustandsgraphen** und **Petri-Netze** zunehmend an Bedeutung. Letztere erlauben eine hierarchisch gegliederte Funktionsbeschreibung der Steuerungsaufgabe und dienen als interdisziplinäres Verständigungsmittel (VDI-Richtlinie 3683).

Anweisungsliste (AWL) / Instruction List (IL):

Assemblerähnliche Programmiersprache

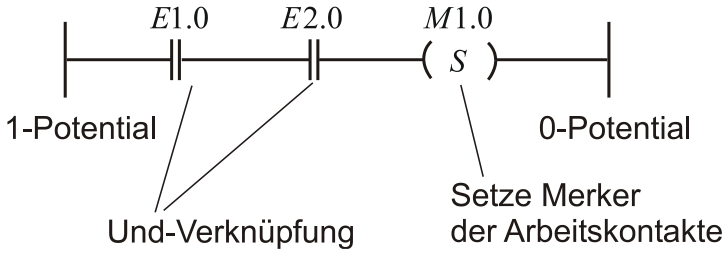
Beispiel: Zweikanalsicherheitssteuerung

$$\left.\begin{array}{ll} U & E1.0 \\ U & E2.0 \end{array}\right\} \text{Und-Verknüpfung von } \underset{(W1)}{E1.0} \text{ und } \underset{(W2)}{E2.0}$$

$S \quad M1.0$ Setze Merker $M1.0$

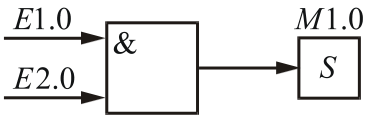
Kontaktplan (KOP) / Ladder Diagram (LD):

Beispiel:

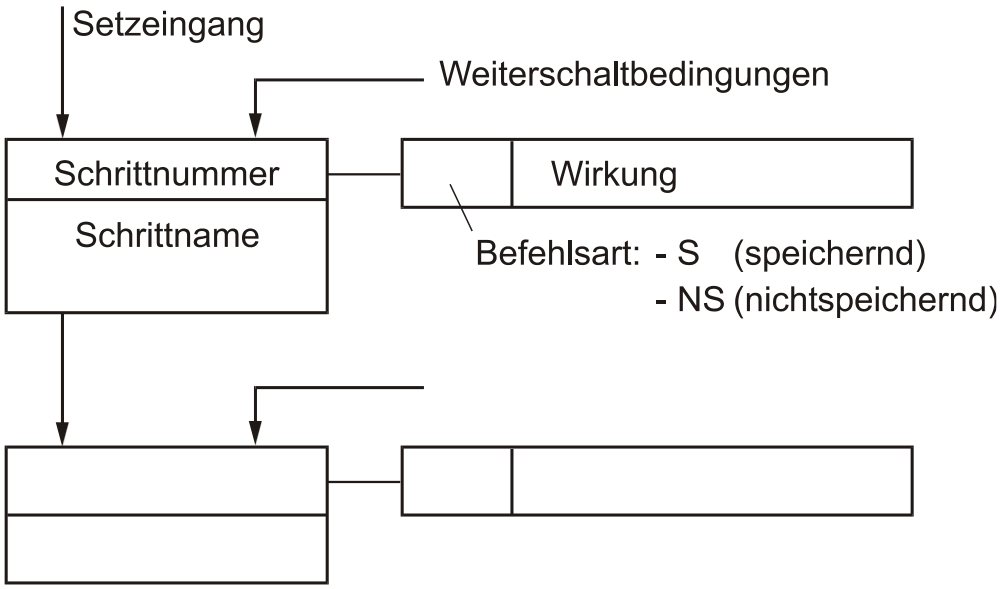


Funktionsplan (FUP) / Function Block Diagram (FBD):

Beispiel:



Ablaufsprache / Sequential Function Chart (SFC):



Strukturierter Text / Structured Control Language (SCL):

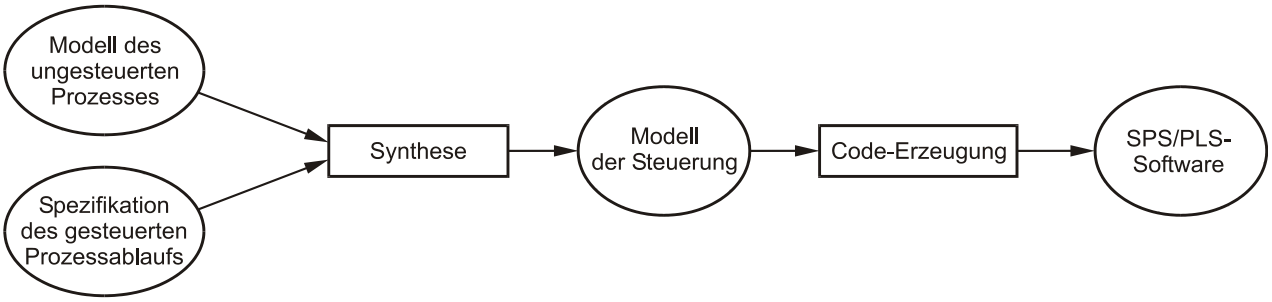
Beispiel:

```
IF   E1.0   AND   E2.0   THEN
                  M1.0 := 1 ;
ELSE
                  M1.0 := 0 ;
END_IF ;
```

Entwurf von Ablaufsteuerungen:

Ablaufsteuerungen haben die Aufgabe, gewünschte Prozessabläufe zu realisieren.

Ausgangspunkt des intuitiven Entwurfs ist ein Modell des ungesteuerten Streckenverhaltens mit allen möglichen Mess- und Stellsignalen, das durch Modularisierung und Hierarchisierung in überschaubare Teilmodelle gegliedert wird. Es schließt sich die (rechnerunterstützte) Spezifikation aller Einzelheiten des gewünschten Prozessablaufs und die damit einhergehende Festlegung der binären Mess- und Stellsignale mit Hilfe einer bestimmten Modellierungsmethodik an, aus der direkt ein formales Modell der Steuerung entsteht, das im Anschluss häufig automatisch in Steuerungscode umgewandelt wird.



Entwurf von Verriegelungssteuerungen (Verbotene-Zustände-Problem):

Verriegelungssteuerungen haben die Aufgabe, Verbotsspezifikationen zu realisieren.

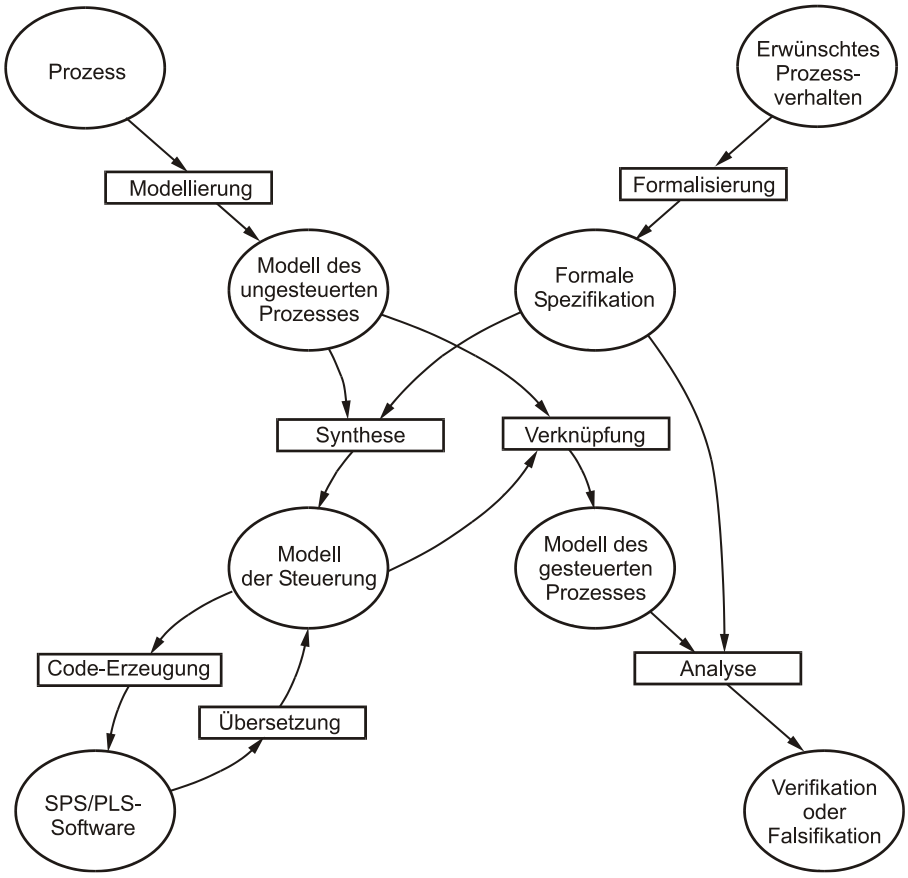
Problem: Im Prozess können häufig nur bestimmte Ereignisse durch die Steuereinrichtung beeinflusst werden.

Lösung: Eine Verriegelungssteuerung muss daher das Erreichen eines solchen Zustands verhindern, von dem aus Ereignisfolgen möglich sind, deren Eintreten nicht steuernd unterbunden werden kann und

- die zu einem verbotenen Zustand führen oder
- die eine verbotene Folge von Zuständen bewirken.

Der Entwurf der Verriegelungssteuerung erfolgt entweder intuitiv wie bei der Ablaufsteuerung oder algorithmisch zum Beispiel auf Basis von kontrollierten Petri-Netzen (Holloway, Krogh, 1991) oder NCE-Systemen.

Vorgehen bei der Verifikation von Steuerungen:



Spezifikation:

1. Der Zustand Z (die Markierung M) darf während des Anlagenbetriebs niemals auftreten.

2. Von jedem Zustand aus muss ein anderer Zustand aus erreichbar sein.

Deutung:

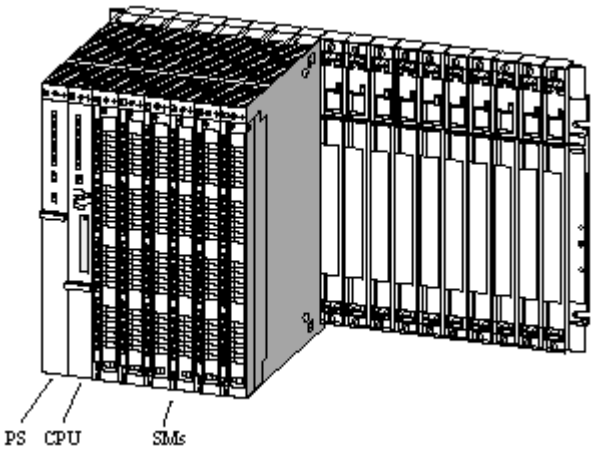
1. Ist $M \in R_N(M_0)$?
   Falls nein, ist Spezifikation erfüllt.

2. Kann eine totale Verklemmung auftreten?
   Falls nein, ist Spezifikation erfüllt.

Eine Speicherprogrammierbare Steuerung SPS (engl. Programmable Logic Controller PLC) ist nach IEC-1131 „ein digitales, elektronisches System, das über einen Programmspeicher zur internen Speicherung der Anweisungen des Anwenderprogramms verfügt, welches die Festlegung der anwendungsspezifischen, vom System auszuführenden Operationen enthält."

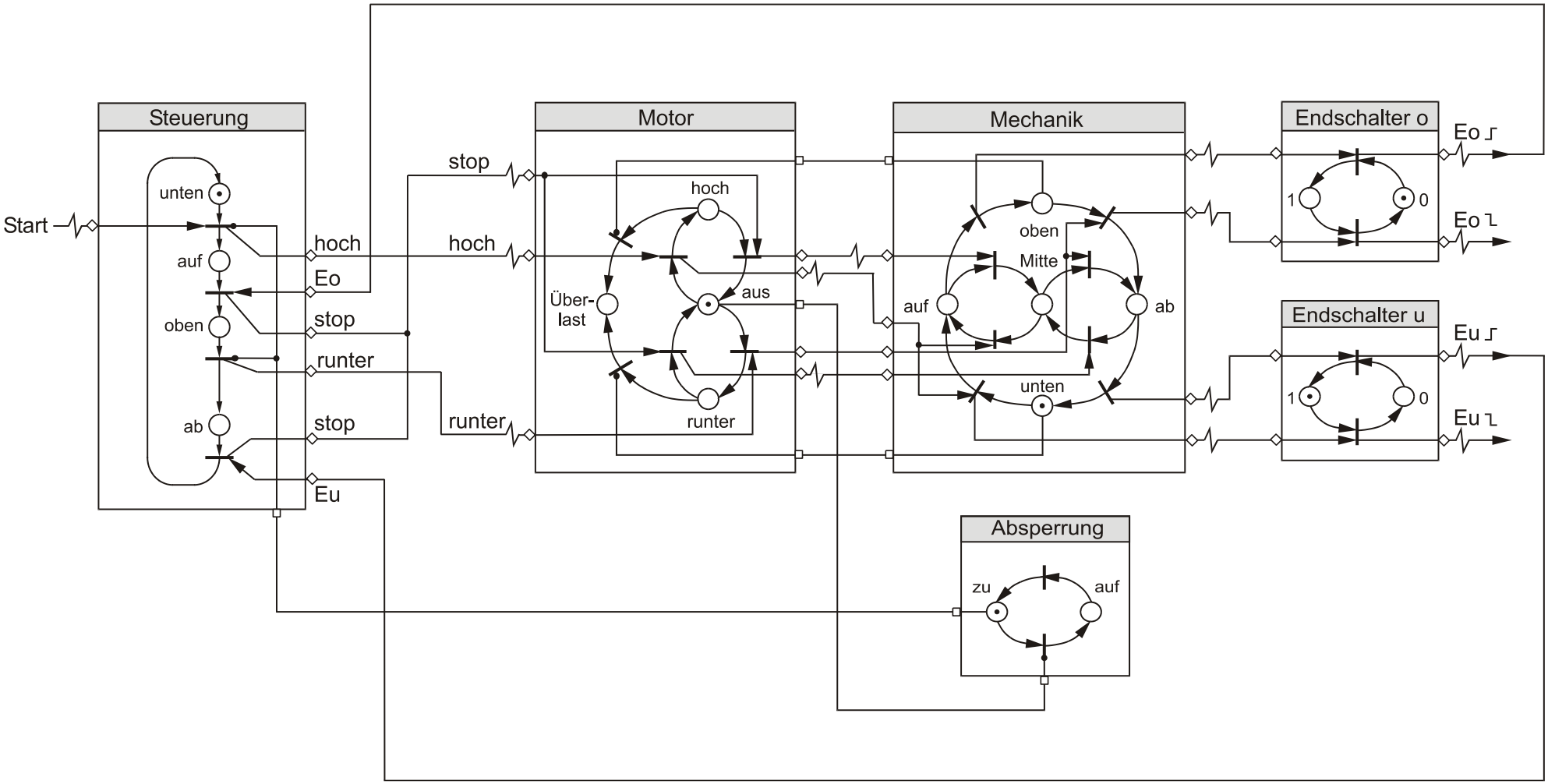
Durch besondere Schutzmaßnahmen sind SPSen der rauen industriellen Umgebung angepasst. Sie übernehmen heute neben der ursprünglichen Funktionalität *Steuern* u.a. die Funktionen *Melden, Regeln, Positionieren, Rechnen, Diagnose, Datenverwaltung, Visualisieren* und *Kommunikation*.

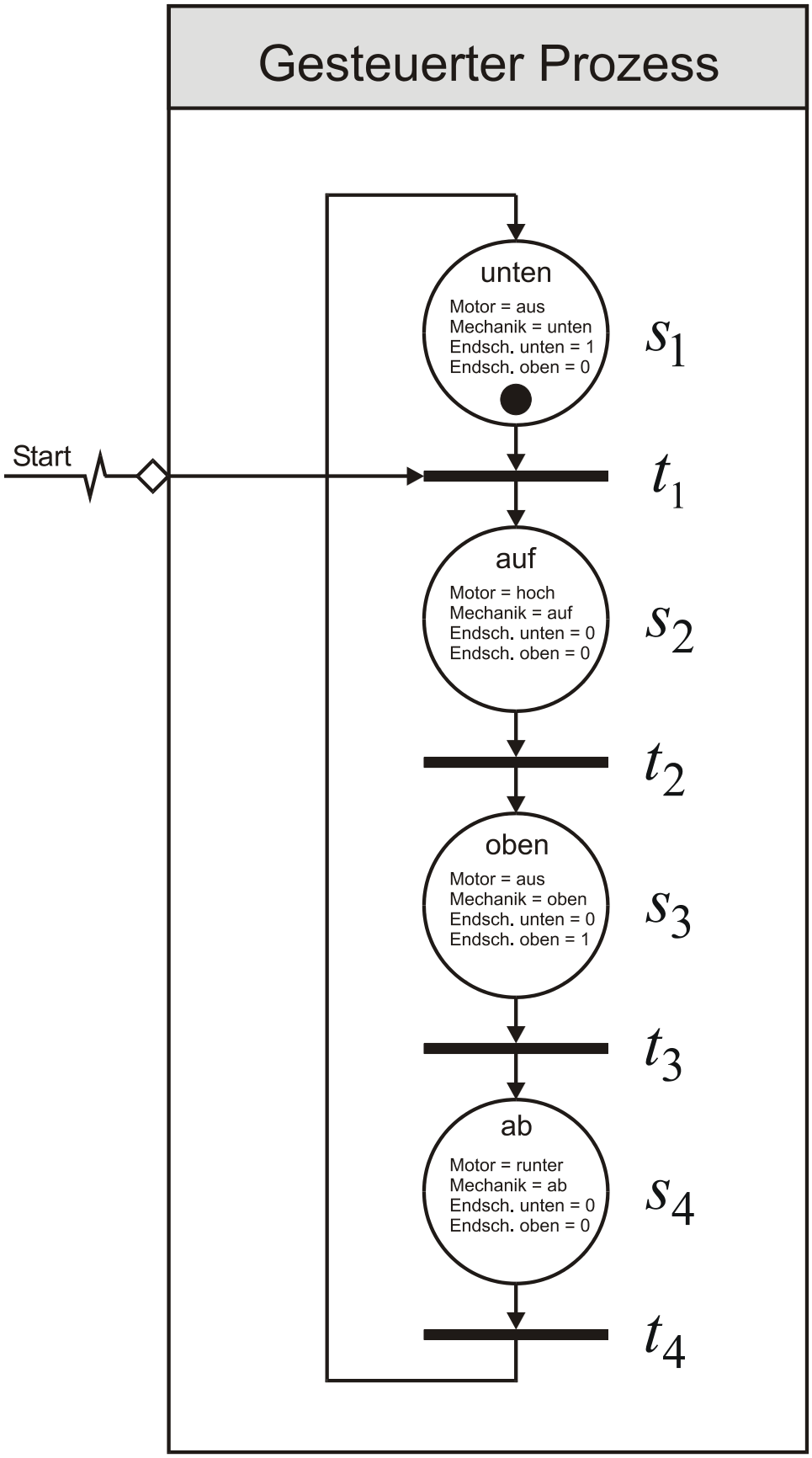
Es werden zwei Bauformen unterschieden: *Kompaktsteuerungen* enthalten in einem Gehäuse alle Funktionsblöcke und sind für kleinere Automatisierungsaufgaben geeignet. Die Mehrzahl der SPSen sind *modulare Steuerungen*, bei denen die Funktionseinheiten als getrennte Baugruppen (Zentraleinheit und Anwenderspeicher, Netzteil, Signalbaugruppen etc.) auf einen Baugruppenträger mit Systembus aufgesteckt werden.

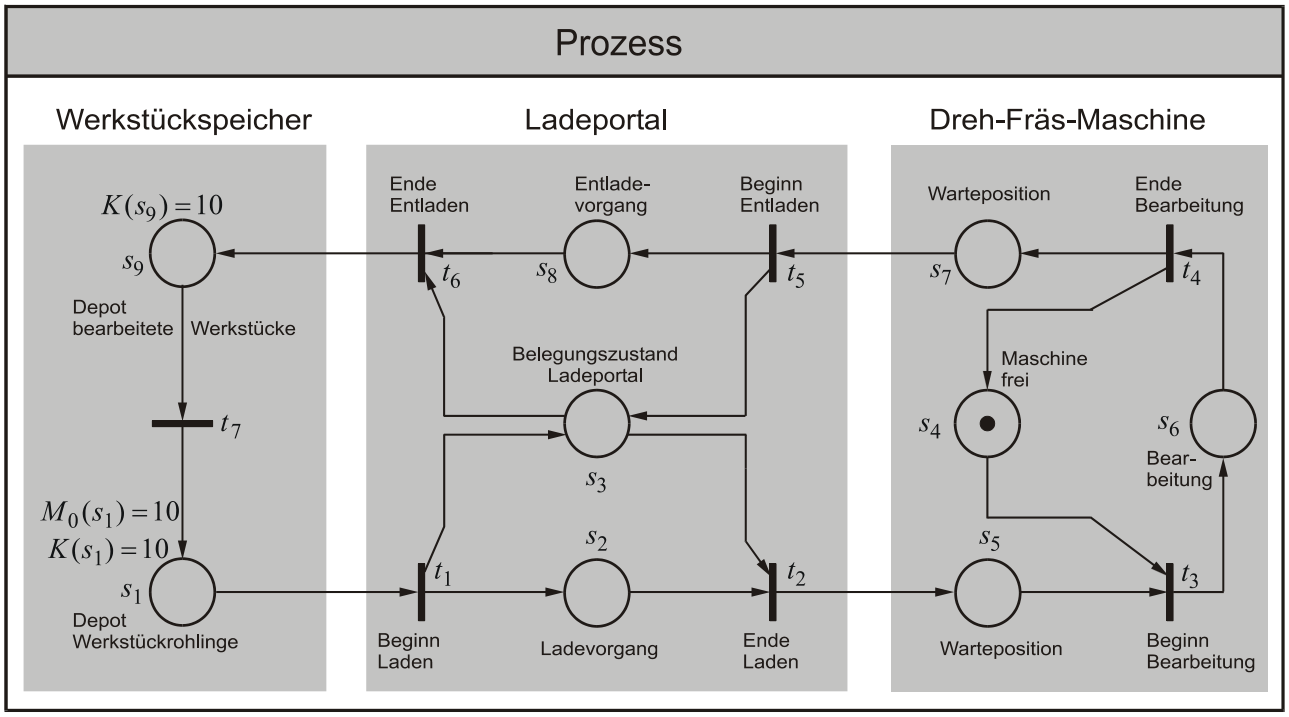


Modulare SPS Siemens S7-400 mit Stromversorgung (PS), Zentraleinheit (CPU) und Signalbaugruppen (SMs) auf einem Baugruppenträger

Die Abarbeitung eines SPS-Programmes erfolgt zyklisch, d.h. nach einem Programmdurchlauf wird das Programm erneut von oben nach unten abgearbeitet. Vor Beginn jedes Zyklus erstellt das SPS-Betriebssystem ein Prozessabbild der Eingänge in einem Puffer. Während der Bearbeitung des Programms wird nur auf diesen Puffer zugegriffen. Erst am Zyklusende wird das während des Programmlaufs erzeugte Prozessabbild der Ausgänge auf die physikalischen Ausgänge übertragen.


Steuerung
Start
unten
auf
oben
ab
hoch
Eo
stop
runter
stop
Eu
stop
hoch
runter
Motor
hoch
Über-
last
aus
runter
Mechanik
oben
Mitte
auf
ab
unten
Endschalter o
1
0
Eo ⌟
Eo ⌝
Endschalter u
1
0
Eu ⌟
Eu ⌝
Absperrung
zu
auf


Gesteuerter Prozess
unten
Motor = aus
Mechanik = unten
Endsch. unten = 1
Endsch. oben = 0
s_1
Start
t_1
auf
Motor = hoch
Mechanik = auf
Endsch. unten = 0
Endsch. oben = 0
s_2
t_2
oben
Motor = aus
Mechanik = oben
Endsch. unten = 0
Endsch. oben = 1
s_3
t_3
ab
Motor = runter
Mechanik = ab
Endsch. unten = 0
Endsch. oben = 0
s_4
t_4



Netzmatrix:

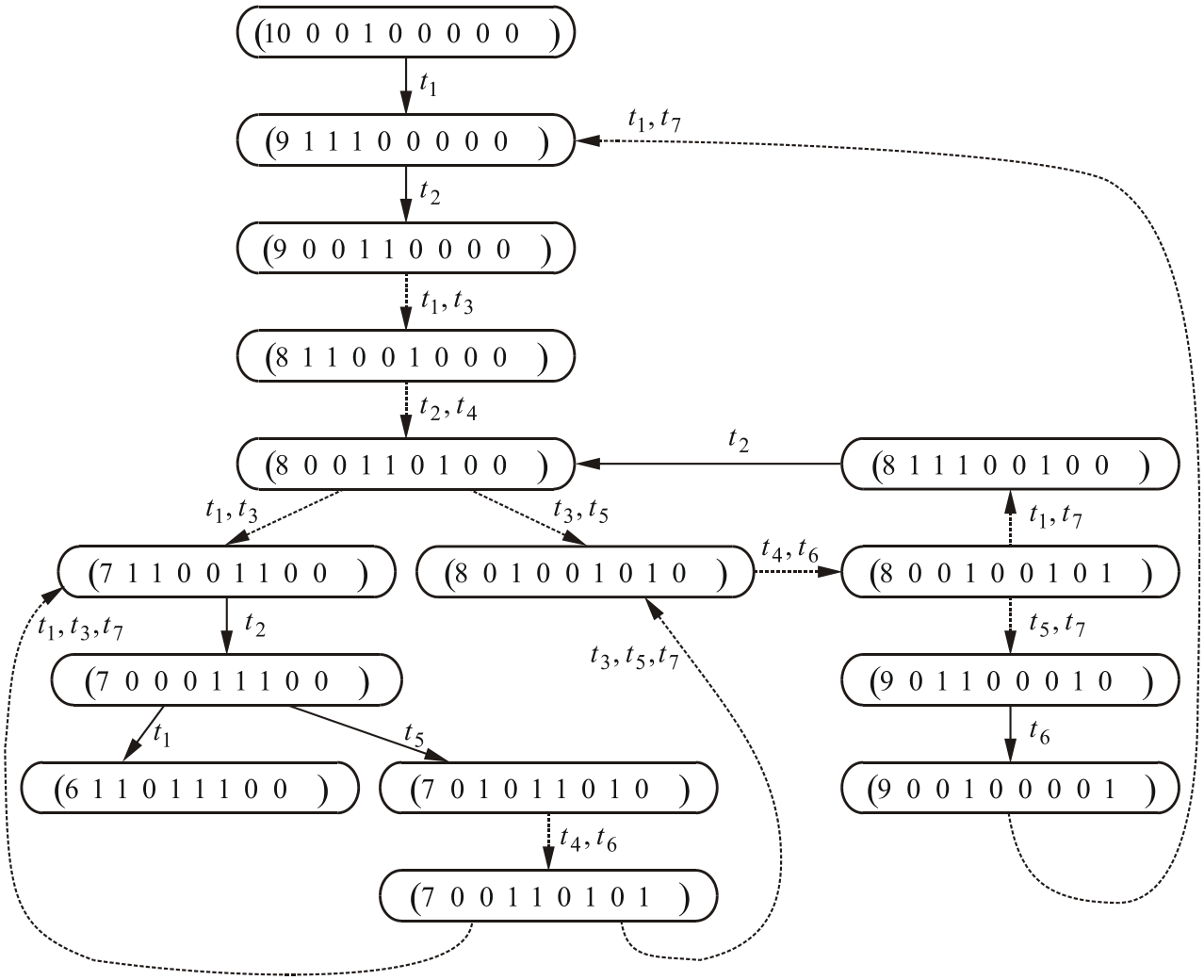
$$\underline{N} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix},$$

$$\underline{m}^T(0) = \underline{m}_0^T = \begin{bmatrix} 10 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Für das Petri-Netz der Fertigungsanlage ergibt sich der nachfolgende Erreichbarkeitsgraph. Dieser ist für den Steuerungsentwurf aus Übersichtsgründen komprimiert dargestellt, so dass einige Markierungen mit zugehörigen Übergängen nicht sichtbar sind.

Diese Übergänge und Markierungen verbergen sich hinter den gestrichelten Kanten, die mit den jeweils parallel schaltbaren Transitionen beschriftet sind.

Ungesteuerter Prozess:



Aufgabe:

Vermeidung der toten Markierung (siehe oben)

$$\underline{m}_{tot}^{T} = \begin{bmatrix} 6 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} \quad .$$

a) **Entwurf mittels S-Invarianten:**

Spezifikation: $m_2+m_3+m_5+m_6+m_7 \le 4$,

$$\underline{L}\,\underline{m}(k) \le 4 \quad \text{mit} \quad \underline{L} = \underline{l}^T = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} .$$

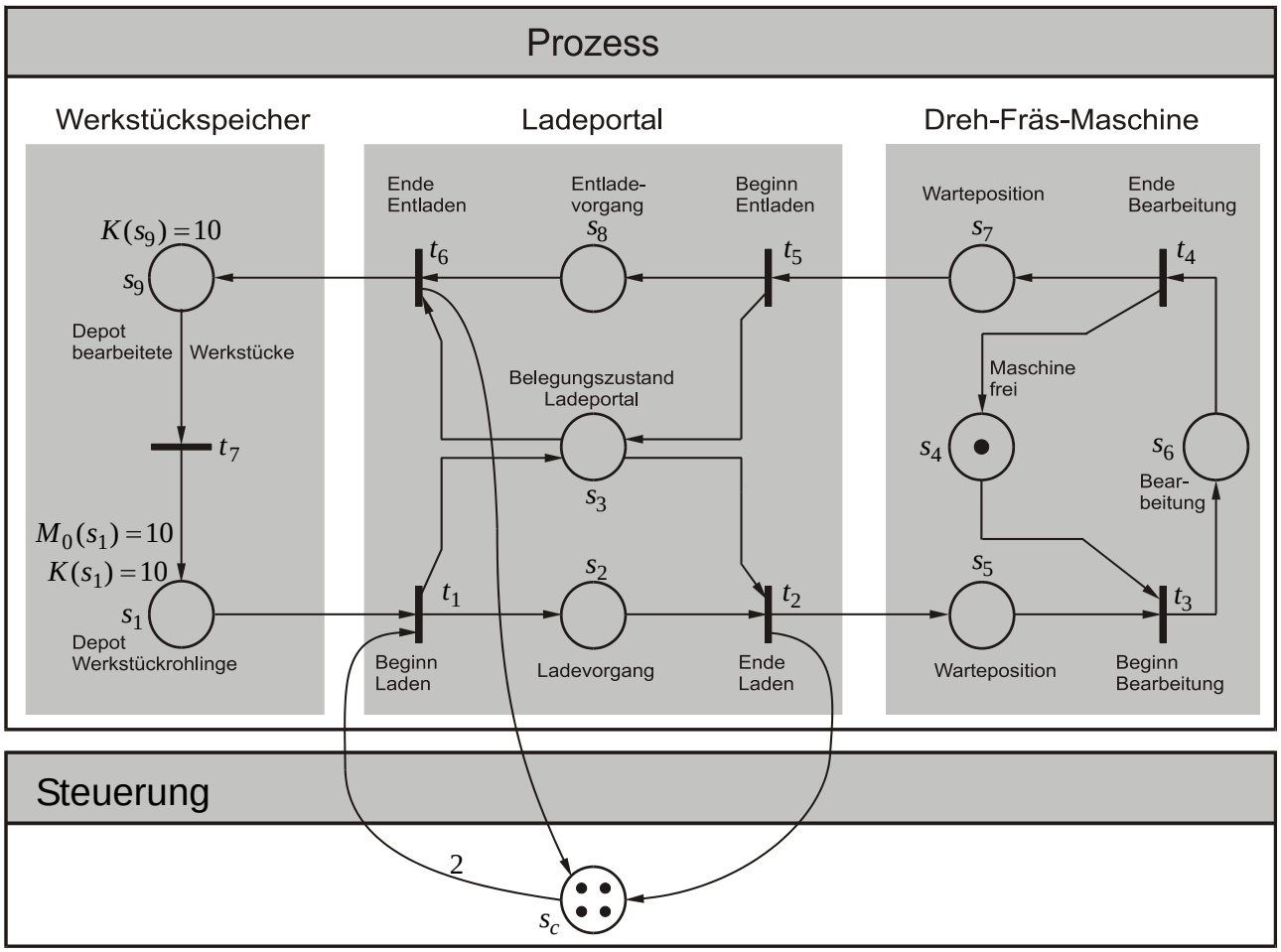
Entwurf: Einfügen einer Kontroll-Stelle s_c : $\underline{l}^T\,\underline{m}(k) + m_c(k) = 4 \quad (m_c(k) \ge 0)$,

$$\begin{bmatrix} \underline{l}^T & 1 \end{bmatrix} \begin{bmatrix} \underline{m}(k) \\ m_c(k) \end{bmatrix} = 4 \quad .$$

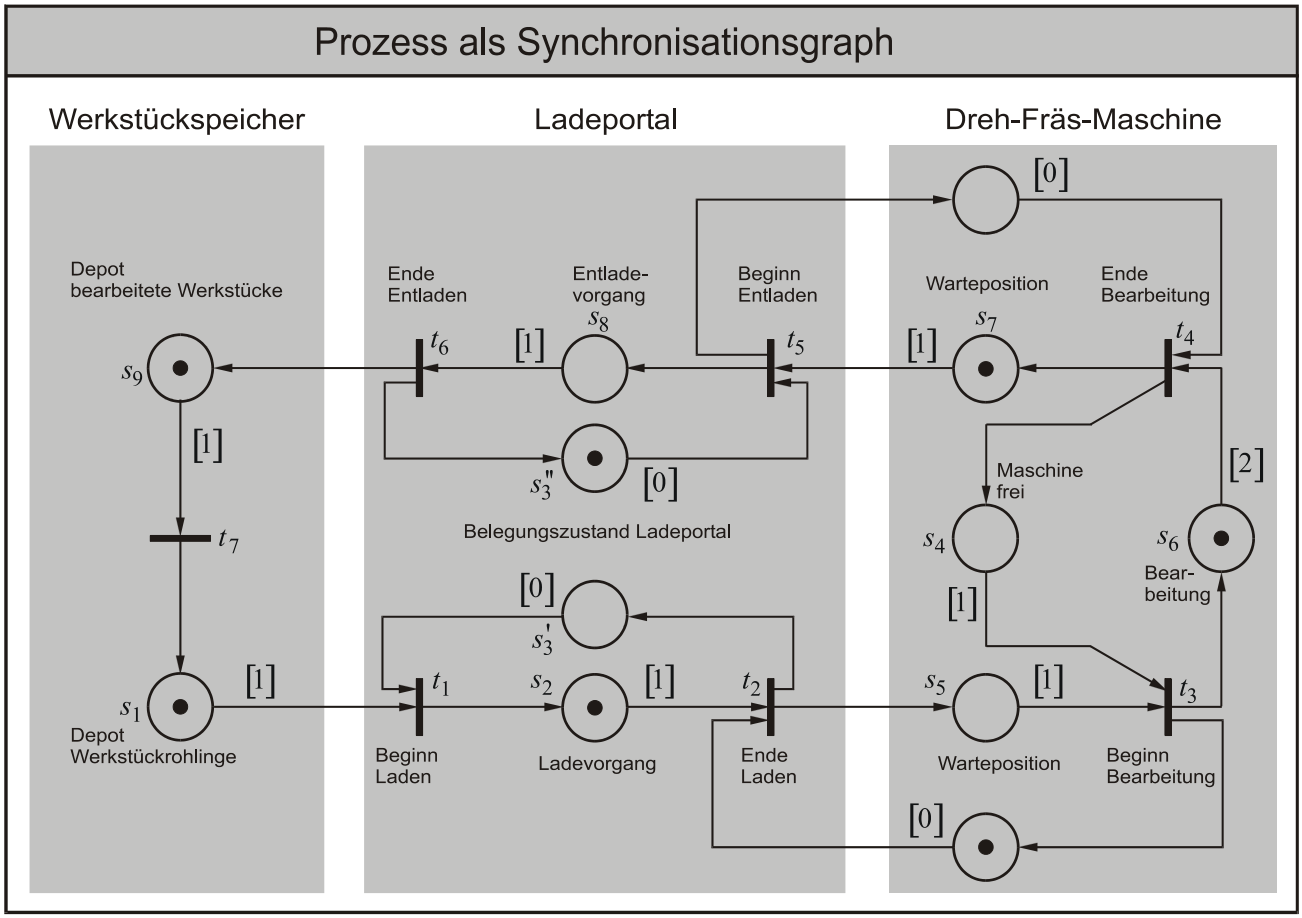
$\begin{bmatrix} \underline{l}^T & 1 \end{bmatrix}^T$ ist also eine S-Invariante des gesteuerten Prozesses mit $\begin{bmatrix} \underline{N} \\ \underline{N}_c \end{bmatrix} = \begin{bmatrix} \underline{N} \\ \underline{n}_c^T \end{bmatrix}$:

$$\begin{bmatrix} \underline{N} \\ \underline{n}_c^T \end{bmatrix}^T \begin{bmatrix} \underline{l} \\ 1 \end{bmatrix} = \underline{0} \quad \Rightarrow \begin{bmatrix} \underline{l}^T & 1 \end{bmatrix} \begin{bmatrix} \underline{N} \\ \underline{n}_c^T \end{bmatrix} = \underline{0}^T \quad \Rightarrow \quad \underline{n}_c^T = -\underline{l}^T\,\underline{N} = \begin{bmatrix} -2 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} ,$$

$$m_c(0) = 4 - \underline{l}^T\,\underline{m}(0) = 4 \quad .$$



b) **Entwurf mittels der Max-Plus-Algebra:**



Implizite Darstellung des ungesteuerten Prozesses:

$$\underline{x}(k+1) = \underline{A}_0 \underline{x}(k+1) \oplus \underline{A}_1 \underline{x}(k)$$

mit $\underline{A}_0 = \begin{bmatrix} \varepsilon & 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 1 & \varepsilon & 1 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & 0 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & 1 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{bmatrix}$, $\underline{A}_1 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 1 \\ 1 & \varepsilon & 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 2 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 1 & \varepsilon & 0 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 1 & \varepsilon \end{bmatrix}$.

Explizite Darstellung des ungesteuerten Prozesses:

$$\underline{x}(k+1) = \underline{A}\underline{x}(k)$$

$$\text{mit} \quad \underline{A} = \begin{bmatrix} 1 & \varepsilon & 0 & \varepsilon & \varepsilon & \varepsilon & 1 \\ 1 & \varepsilon & 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 2 & \varepsilon & 3 & 2 & \varepsilon & 1 & \varepsilon \\ \varepsilon & \varepsilon & 2 & 1 & \varepsilon & 0 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 1 & \varepsilon & 0 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 2 & \varepsilon & 1 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & 1 & \varepsilon \end{bmatrix},$$

$$\lambda = 3, \quad \underline{v}^T = \begin{bmatrix} -3 & -3 & 0 & -1 & -3 & -2 & -4 \end{bmatrix}.$$

Verbotsspezifikation:

Kein gleichzeitiges Schalten der Transitionen t_1 (Beginn Laden) und t_5 (Beginn Entladen)

$\Rightarrow$ verbotener Zustand: $v_1 = v_5$.

Steuerung:

Erweiterung des Synchronisationsgraphen um zwei anfangs nicht markierte Stellen s_{12} und s_{52} mit $\tau_{12} = 2$ und $\tau_{52} = 1$

$\Rightarrow$ neue Einträge in $\underline{A}_0$: $\underline{A}_0(1,2) = 2$,

$$\underline{A}_0(5,2) = 1 \ .$$

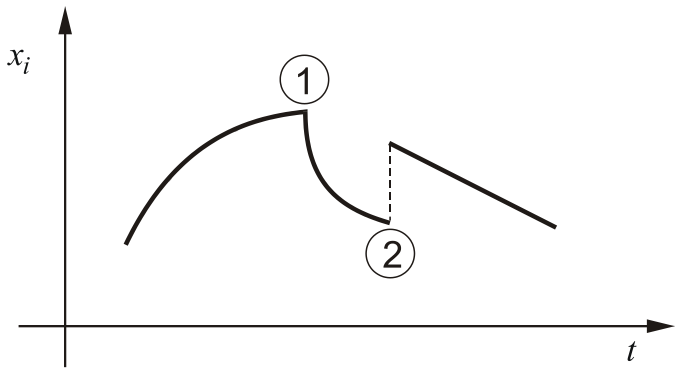
Eigenwert und Eigenvektoren des gesteuerten Systems:

$$\lambda_R = 3, \quad \underline{v}_{R,1}^T = \begin{bmatrix} 0 & -2 & 0 & -1 & -1 & 0 & -2 \end{bmatrix},$$

$$\underline{v}_{R,2}^T = \begin{bmatrix} -1 & -3 & 0 & -1 & -2 & -1 & -2 \end{bmatrix}$$

$\Rightarrow$ verbotener Zustand wird vermieden.

Zeitverlauf einer Zustandsgröße eines hybriden Systems:



Hybride Phänomene:

① **Umschalten:**

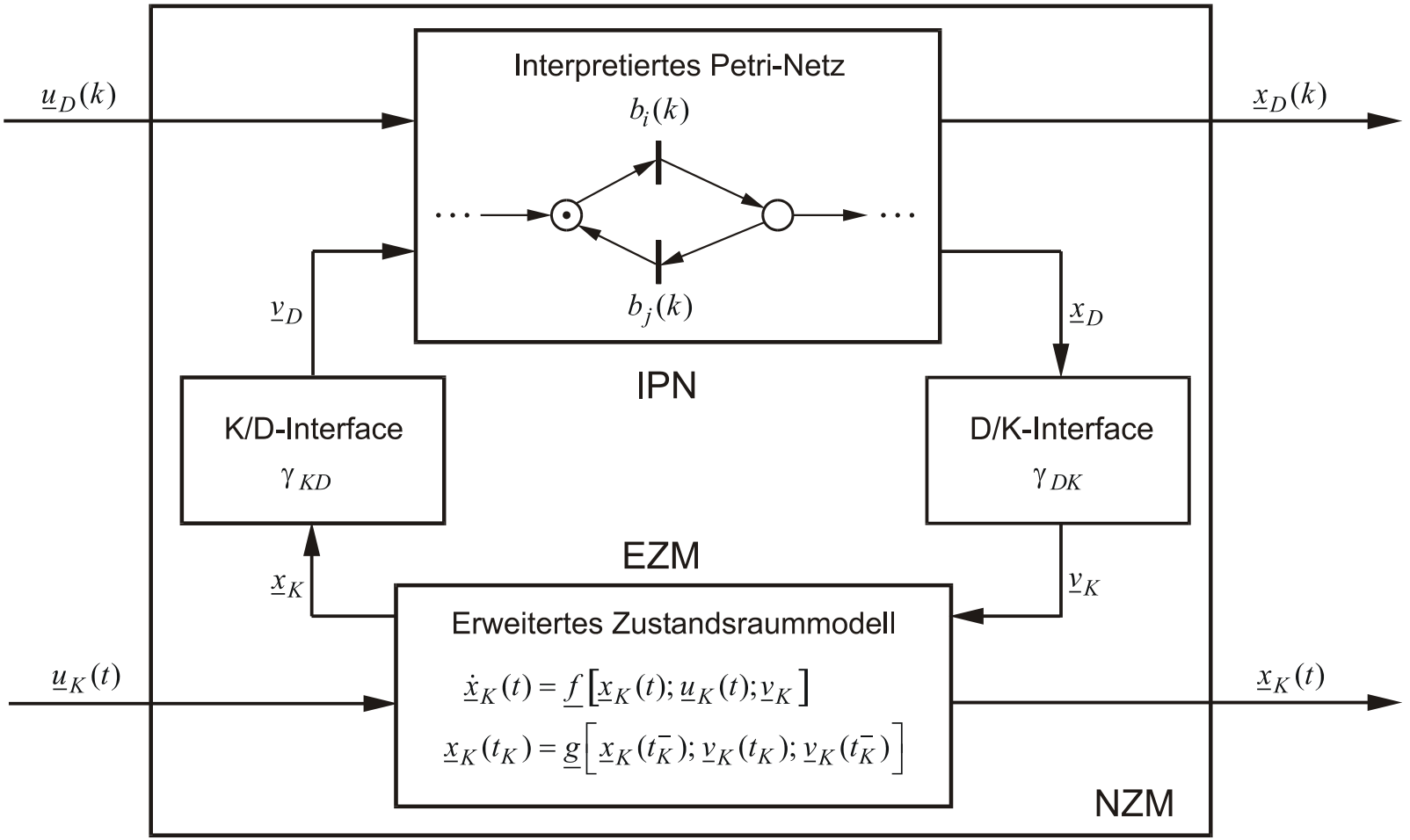
Kennzeichen: Zustandsableitung $\dot{x}_i$ ändert sich

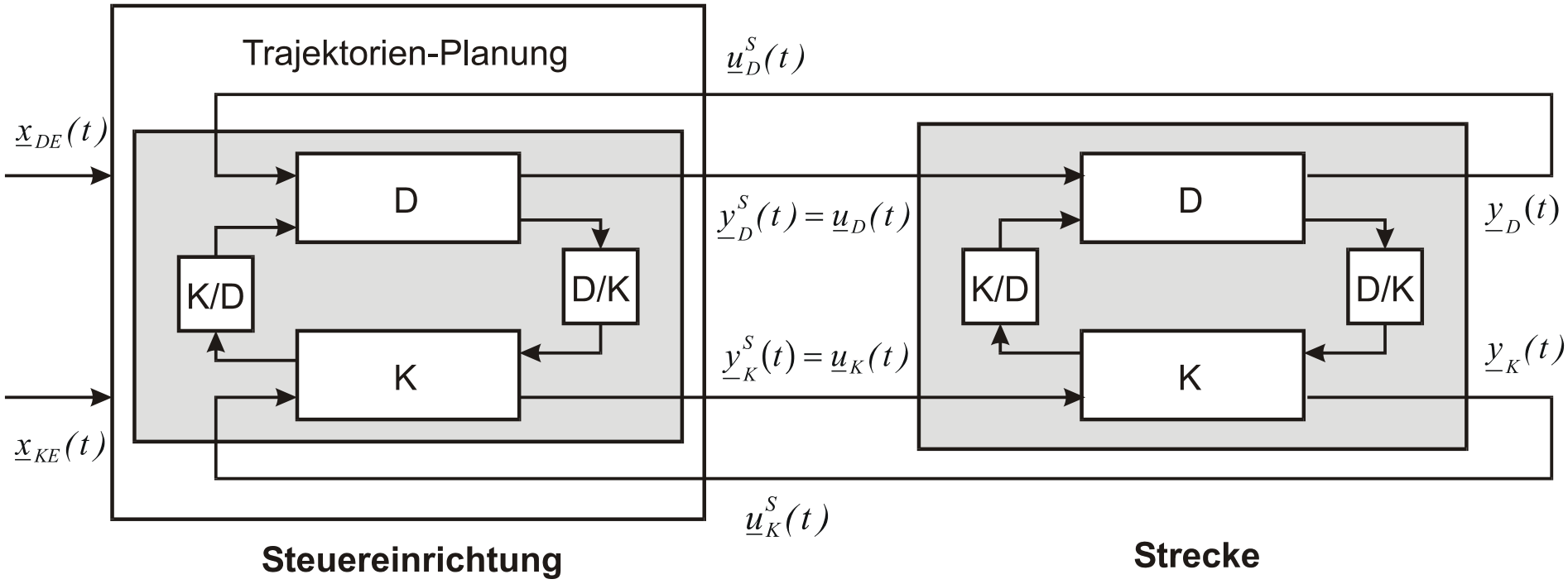
Zustandsänderung:
- **selbständig**
  abhängig von internen Systemgrenzen
- **gesteuert**
  abhängig von Eingangsgrößen

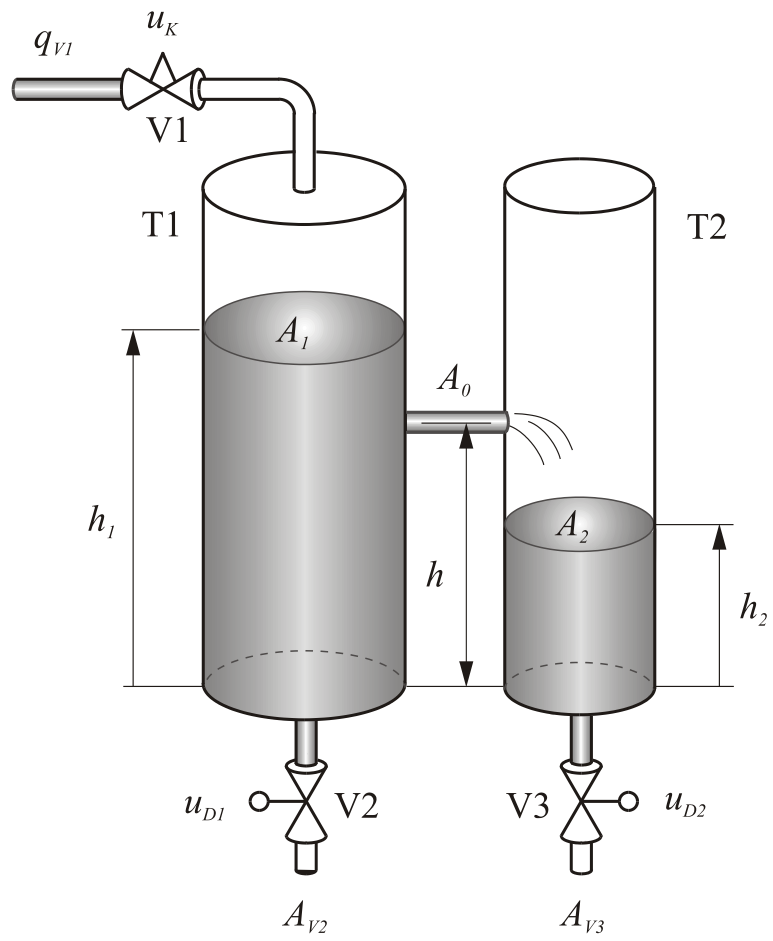
② **Sprung:**

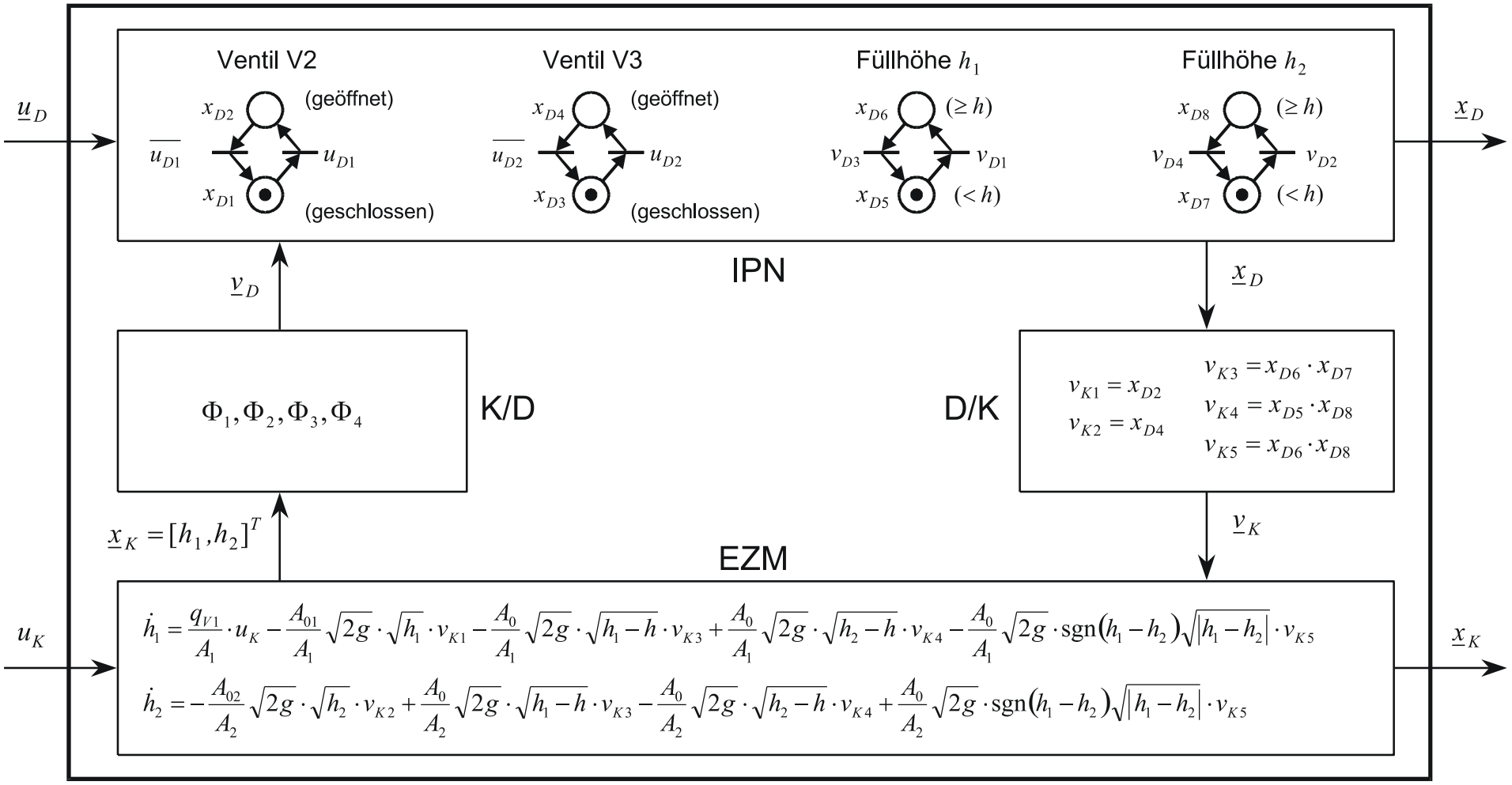
Kennzeichen: Zustandsableitung $\dot{x}_i$ bleibt gleich

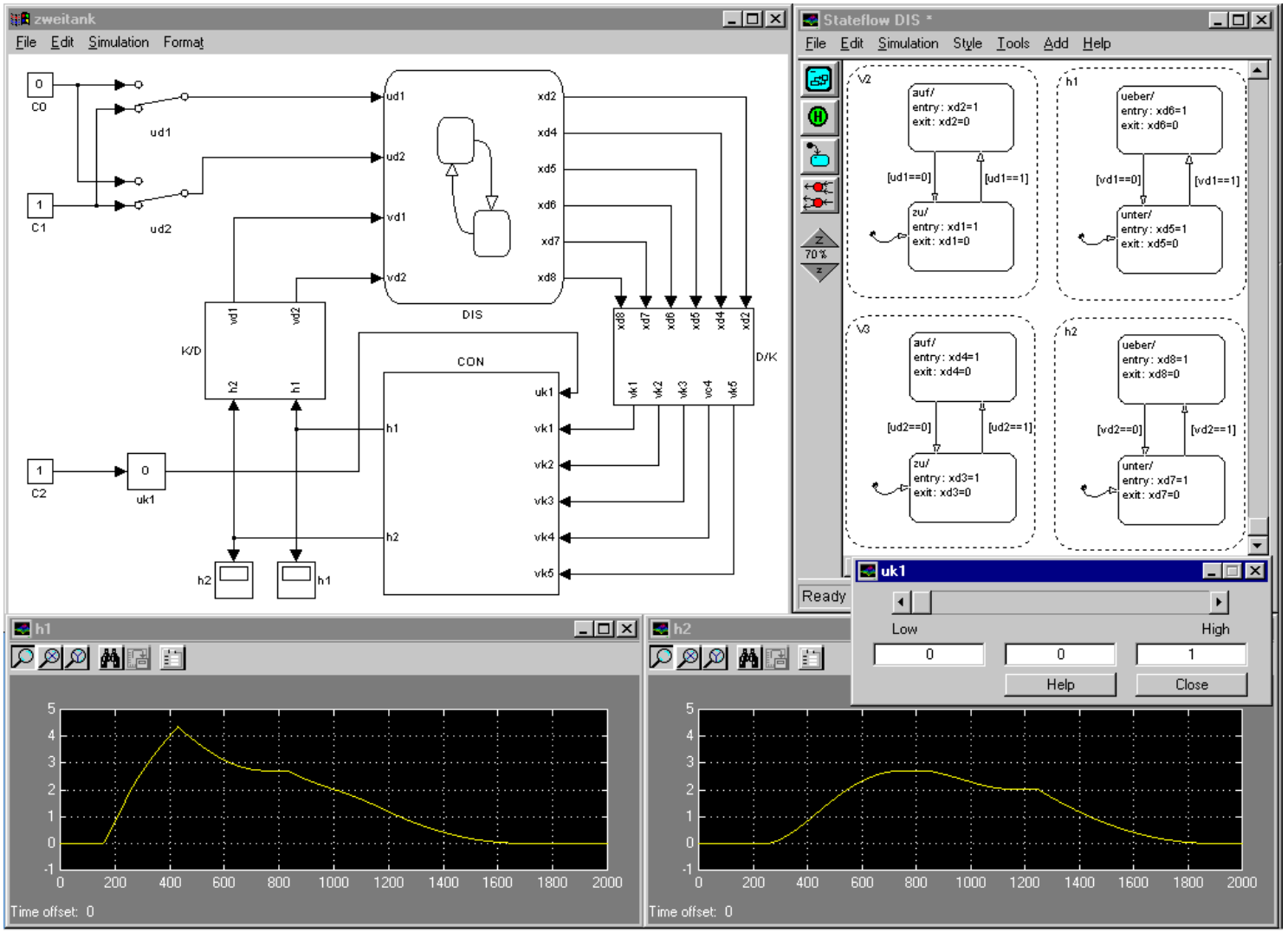
Zustandsänderung:
- **selbständig**
  abhängig von internen Systemgrenzen
- **gesteuert**
  abhängig von Eingangsgrößen

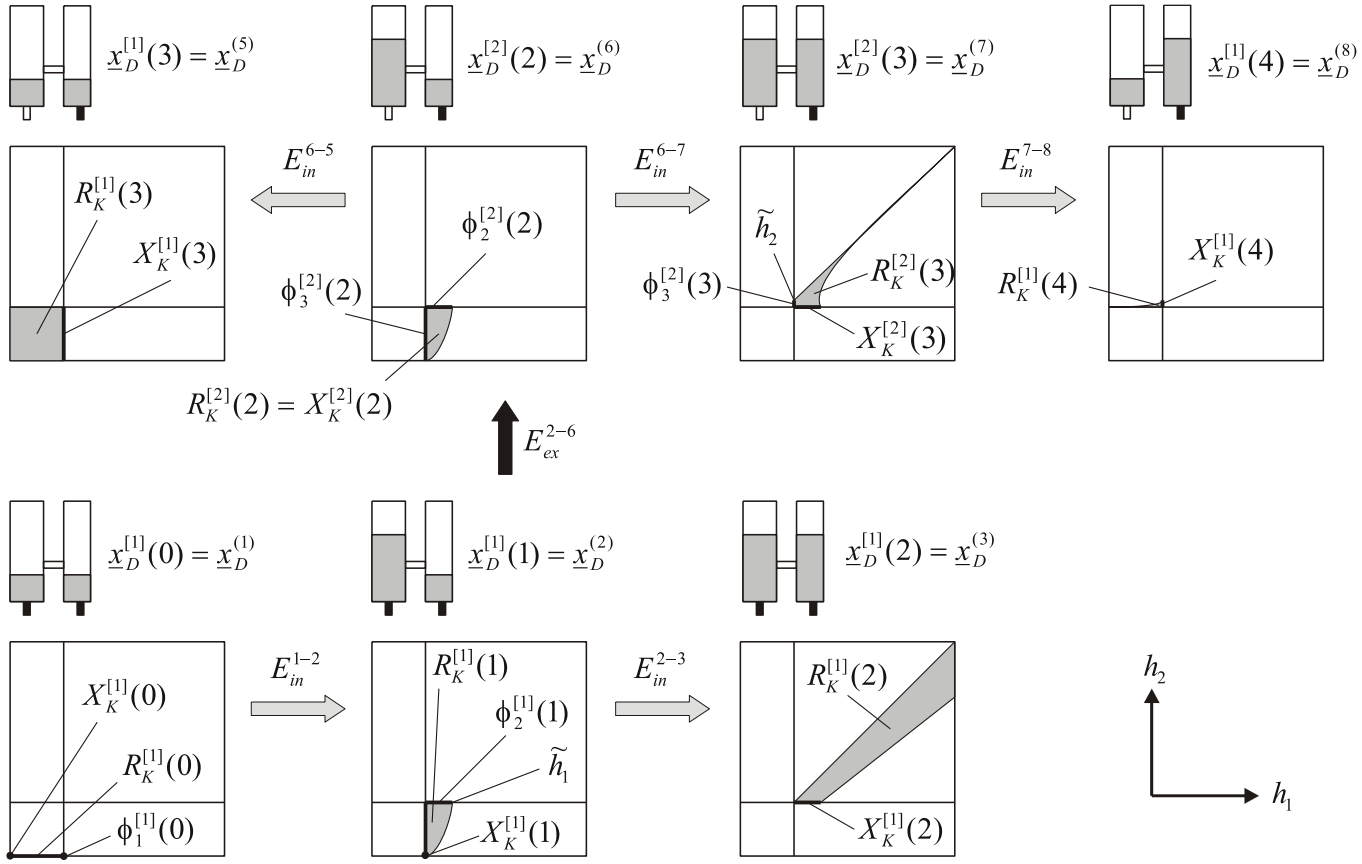

Interpretiertes Petri-Netz
$b_i(k)$
$b_j(k)$
$\underline{u}_D(k)$
$\underline{x}_D(k)$
$\underline{v}_D$
$\underline{x}_D$
IPN
K/D-Interface
γ_{KD}
D/K-Interface
γ_{DK}
EZM
$\underline{x}_K$
$\underline{v}_K$
Erweitertes Zustandsraummodell
$\dot{\underline{x}}_K(t) = \underline{f}\left[\underline{x}_K(t); \underline{u}_K(t); \underline{v}_K\right]$
$\underline{x}_K(t_K) = \underline{g}\left[\underline{x}_K(t_K^-); \underline{v}_K(t_K); \underline{v}_K(t_K^-)\right]$
$\underline{u}_K(t)$
$\underline{x}_K(t)$
NZM


Trajektorien-Planung
$\underline{u}_D^S(t)$
$\underline{x}_{DE}(t)$
D
K/D
D/K
K
$\underline{y}_D^S(t) = \underline{u}_D(t)$
$\underline{y}_K^S(t) = \underline{u}_K(t)$
$\underline{x}_{KE}(t)$
$\underline{u}_K^S(t)$
Steuereinrichtung
D
K/D
D/K
K
$\underline{y}_D(t)$
$\underline{y}_K(t)$
Strecke


q_{V1}
u_K
V1
T1
T2
A_1
A_0
A_2
h_1
h
h_2
u_{D1}
V2
V3
u_{D2}
A_{V2}
A_{V3}


Ventil V2
(geöffnet)
(geschlossen)
Ventil V3
(geöffnet)
(geschlossen)
Füllhöhe h_1
Füllhöhe h_2
$(\geq h)$
$(< h)$
IPN
K/D
D/K
EZM
$\Phi_1, \Phi_2, \Phi_3, \Phi_4$
$v_{K1} = x_{D2}$
$v_{K2} = x_{D4}$
$v_{K3} = x_{D6} \cdot x_{D7}$
$v_{K4} = x_{D5} \cdot x_{D8}$
$v_{K5} = x_{D6} \cdot x_{D8}$
$\underline{x}_K = [h_1, h_2]^T$
$\dot{h}_1 = \frac{q_{V1}}{A_1} \cdot u_K - \frac{A_{01}}{A_1}\sqrt{2g} \cdot \sqrt{h_1} \cdot v_{K1} - \frac{A_0}{A_1}\sqrt{2g} \cdot \sqrt{h_1 - h} \cdot v_{K3} + \frac{A_0}{A_1}\sqrt{2g} \cdot \sqrt{h_2 - h} \cdot v_{K4} - \frac{A_0}{A_1}\sqrt{2g} \cdot \mathrm{sgn}(h_1 - h_2)\sqrt{|h_1 - h_2|} \cdot v_{K5}$
$\dot{h}_2 = -\frac{A_{02}}{A_2}\sqrt{2g} \cdot \sqrt{h_2} \cdot v_{K2} + \frac{A_0}{A_2}\sqrt{2g} \cdot \sqrt{h_1 - h} \cdot v_{K3} - \frac{A_0}{A_2}\sqrt{2g} \cdot \sqrt{h_2 - h} \cdot v_{K4} + \frac{A_0}{A_2}\sqrt{2g} \cdot \mathrm{sgn}(h_1 - h_2)\sqrt{|h_1 - h_2|} \cdot v_{K5}$


zweitank
File Edit Simulation Format
C0
C1
C2
ud1
ud2
vd1
vd2
DIS
K/D
D/K
CON
uk1
h1
h2
Stateflow DIS *
File Edit Simulation Style Tools Add Help
auf/ entry: xd2=1 exit: xd2=0
zu/ entry: xd1=1 exit: xd1=0
ueber/ entry: xd6=1 exit: xd6=0
unter/ entry: xd5=1 exit: xd5=0
auf/ entry: xd4=1 exit: xd4=0
zu/ entry: xd3=1 exit: xd3=0
ueber/ entry: xd8=1 exit: xd8=0
unter/ entry: xd7=1 exit: xd7=0
Ready
uk1
Low
High
Help
Close
Time offset: 0


$\underline{x}_D^{[1]}(3) = \underline{x}_D^{(5)}$
$\underline{x}_D^{[2]}(2) = \underline{x}_D^{(6)}$
$\underline{x}_D^{[2]}(3) = \underline{x}_D^{(7)}$
$\underline{x}_D^{[1]}(4) = \underline{x}_D^{(8)}$
E_{in}^{6-5}
E_{in}^{6-7}
E_{in}^{7-8}
$R_K^{[1]}(3)$
$X_K^{[1]}(3)$
$\phi_2^{[2]}(2)$
$\phi_3^{[2]}(2)$
$\tilde{h}_2$
$\phi_3^{[2]}(3)$
$R_K^{[2]}(3)$
$X_K^{[2]}(3)$
$R_K^{[1]}(4)$
$X_K^{[1]}(4)$
$R_K^{[2]}(2) = X_K^{[2]}(2)$
E_{ex}^{2-6}
$\underline{x}_D^{[1]}(0) = \underline{x}_D^{(1)}$
$\underline{x}_D^{[1]}(1) = \underline{x}_D^{(2)}$
$\underline{x}_D^{[1]}(2) = \underline{x}_D^{(3)}$
E_{in}^{1-2}
E_{in}^{2-3}
$X_K^{[1]}(0)$
$R_K^{[1]}(0)$
$\phi_1^{[1]}(0)$
$R_K^{[1]}(1)$
$\phi_2^{[1]}(1)$
$\tilde{h}_1$
$X_K^{[1]}(1)$
$R_K^{[1]}(2)$
$X_K^{[1]}(2)$
h_2
h_1