TECHNISCHE UNIVERSITÄT DRESDEN

VLSI-EDA

Fakultät Informatik, Institut für Technische Informatik, Professur VLSI-Entwurfssysteme, Diagnostik und Architektur

Rechnerstrukturen und -organisation

Schaltwerke

Rainer G. Spallek

TU Dresden, 08.12.2020

Rainer.Spallek@tu-dresden.de http://vlsi-eda.inf.tu-dresden.de

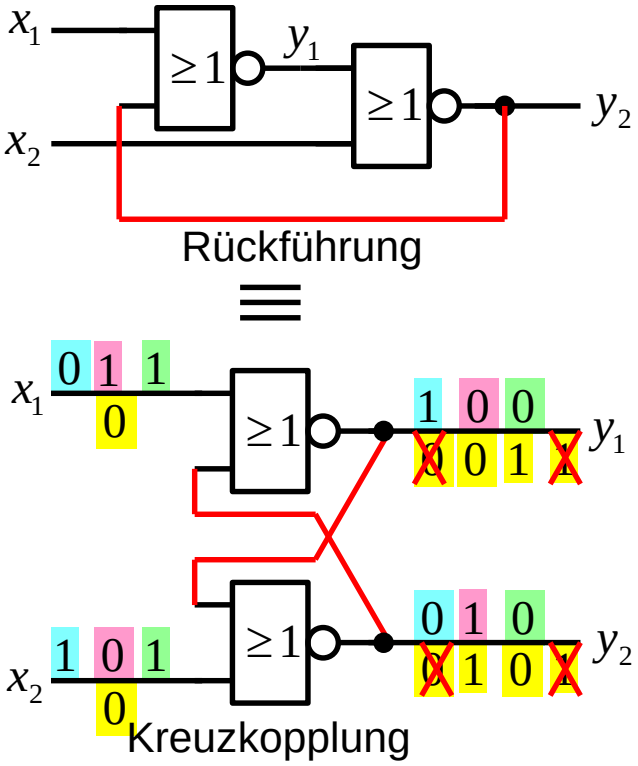
Gliederung

1 Zielstellung

- Kennenlernen einfacher rückgekoppelter Schaltnetze → Basis-Flipflop.
- Einführung der funktionellen Bedeutung der Zeit → sequentielle Schaltung.
- SR-Flipflop als elementares Speicherglied → Gegensatz zu Schaltnetzen.
- Übergang zu synchronen, taktgesteuerten und komplexen Speichergliedern.
- Einführung des taktgesteuerten D-Flipflop als universelles Speicherglied in der Computertechnik.
- Übergang von kombinatorischen zu sequentiellen Schaltungen, von Schaltnetzen zu Schaltwerken.
- Einführung zur Automatentheorie als theoretische Basis der Schaltwerke.
- Darstellungsvarianten von Schaltwerken.
- Analyse und Synthese von Schaltwerken.

2 Speicherglieder - Flipflops

Schaltnetz → Speicherglied → Schaltwerk



Wertetabelle

x_1	x_2	y_1	y_2
0	0	0	1
		1	0
0	1	1	0
1	0	0	1
1	1	0	0

Für $x_1=x_2=0$ ergibt sich:
$y_1=\overline{y_2}=0$ oder $y_1=\overline{y_2}=1$
Zuordnung:
Eingang → Ausgang
nicht mehr eindeutig
→ **kein Schaltnetz**

Die Lösungen für y_1, y_2 bei $x_1=x_2=0$ sind von der vorherigen Belegung der Ausgänge y_1, y_2 abhängig → Speicherung, Zeitverhalten
→ $y_1=\overline{y_2}$ → $y_1=y_2$ nicht stabil für $x_1=x_2=0$

Zeitverhalten, Zustand

- Außer bei der Eingangsbelegung $x_1 = \overline{x_2}$ werden immer komplementäre Ausgangsbelegungen ausgegeben → bistabile Kippschaltung, Flipflop.

  $x_1 \wedge x_2 = 0 \rightarrow y_1 = \overline{y}_2$ $\qquad$ $x_1 \wedge x_2 = 1 \rightarrow y_1 = y_2 = 0$

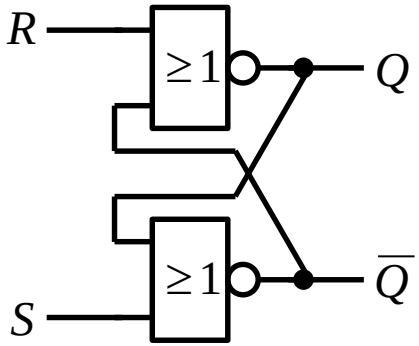
- Die bei der Eingangsbelegung $x_1 = x_2 = 0$ eingenommenen Ausgangswerte werden als Zustand des Flipflop definiert → gespeicherter Zustand.

- Die mit den Eingangsbelegungen $x_1 = 0, x_2 = 1$ bzw. $x_1 = 1, x_2 = 0$ eingenommene Ausgangswerte $y_1 = 1, y_2 = 0$ bzw. $y_1 = 0, y_2 = 1$ werden beim Übergang der Eingangswerte zu $x_1 = x_2 = 0$ im Flipflop gespeichert.

- Die für die Eingangsbelegung $x_1 = x_2 = 1$ eingenommene Ausgangswerte $y_1 = y_2 = 0$ sind beim Übergang zu $x_1 = x_2 = 0$ im Flipflop nicht speicherbar, kippen nach einer Einschwingphase zu $y_1 = 1, y_2 = 0$ oder $y_1 = 0, y_2 = 1$.

- Es handelt sich nicht um ein Schaltnetz, da keine eindeutige Abbildung der Eingangsbelegungen auf die Ausgangsbelegungen vorhanden ist.

Funktionelle Bedeutung der Zeit



SR-Flipflop

Basis - Flipflop

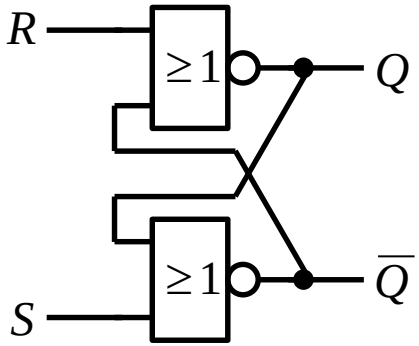
$R(t_n)$	$S(t_n)$	$Q(t_{n+1})$	$\overline{Q}(t_{n+1})$	**Zustandsfolgetabelle**
0	0	$Q(t_n)$	$Q(t_n)$	speichern
0	1	1	0	setzen
1	0	0	1	rücksetzen
1	1	–	–	nicht zulässig

→ $S(t_n) \wedge R(t_n) = 1$ nicht zulässig

Die Ausgangswerte zum aktuellen Zeitpunkt t_n werden zum Folgezeitpunkt t_{n+1} als Folgezustand (R = S = 0) des Flipflop $Q(t_{n+1})$, $\overline{Q(t_{n+1})}$ gespeichert.

$S(t_n)=0, R(t_n)=0$	Zustandsspeicherung	$Q(t_{n+1}) := Q(t_n),\ \overline{Q(t_{n+1})} := \overline{Q(t_n)}$
$S(t_n)=1, R(t_n)=0$	Zustandsänderung	$Q(t_{n+1}) := 1,\ \overline{Q(t_{n+1})} := 0$
$S(t_n)=0, R(t_n)=1$	Zustandsspeicherung	$Q(t_{n+1}) := 0,\ \overline{Q(t_{n+1})} := 1$
$S(t_n)=0, R(t_n)=0$	nicht zulässig, da Ausgangsbelegung nicht speicherbar	

SR-Flipflop, Zustandsfolgetabelle



SR-Flipflop
(Basis-Flipflop)

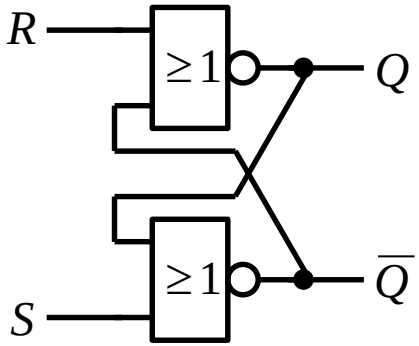
Kurzdarstellung

$$R(t_n) \rightarrow R$$
$$S(t_n) \rightarrow S$$
$$Q(t_n) \rightarrow Q$$
$$Q(t_{n+1}) \rightarrow Q^+$$

(t_n)			(t_{n+1})	
R	S	Q	Q^+	**Zustandsfolgetabelle**
0	0	0	0	speichern
0	1	0	1	setzen
1	0	0	0	rücksetzen
1	1	0	–	nicht zulässig
0	0	1	1	speichern
0	1	1	1	setzen
1	0	1	0	rücksetzen
1	1	1	–	nicht zulässig

Abbildung aktueller Zeitpunkt → Folgezeitpunkt

SR-Flipflop, Zustandsübergangstabelle



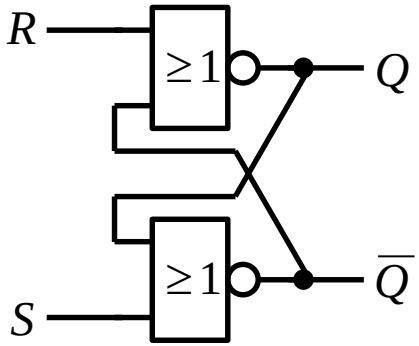
SR-Flipflop
(Basis-Flipflop)

Zustandsübergangstabelle

(t_n)	→	(t_{n+1})	(t_n)		
Q	→	Q^+	R	S	
0	→	0	X	0	speichern oder rücksetzen, nicht setzen
0	→	1	0	1	setzen
1	→	0	1	0	rücksetzen
1	→	1	0	X	speichern oder setzen, nicht rücksetzen

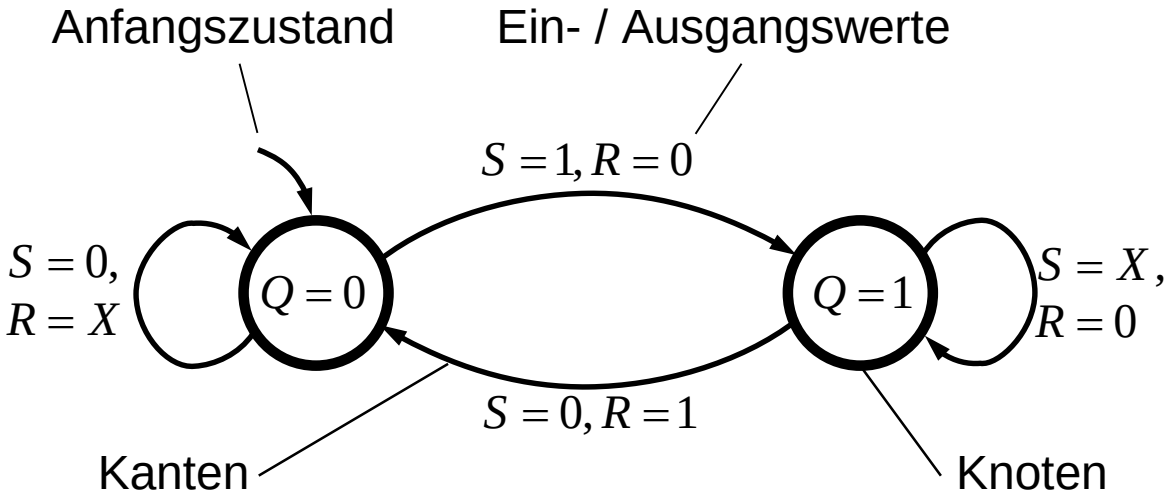
Antwort auf die Frage: Welche Belegung der Eingänge R und S ist erforderlich, um vom Zustand Q in den Zustand Q^+ zu gelangen (→ Zustandsübergang)?

SR-Flipflop, Zustandsgraph



SR-Flipflop
(Basis-Flipflop)

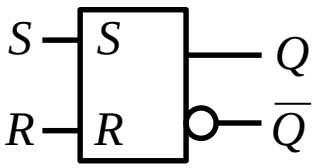
Knoten: entsprechen den Zuständen
Kanten: entsprechen den Zustandsübergängen
Anfangszustand, einseitige Kante



Zustandsfolgetabelle → Zustandsübergangstabelle → Zustandsgraph

SR-Flipflop, Darstellungsvarianten

Schaltsymbol



SR-Flipflop

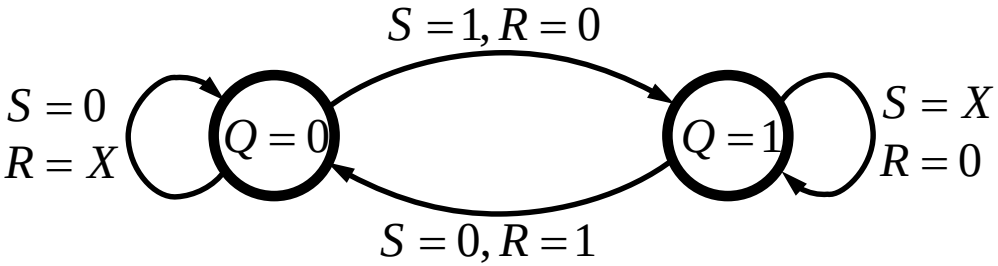
Zustandsfolgetabelle

R	S	Q^+	$\overline{Q}^+$
0	0	Q	Q
0	1	1	0
1	0	0	1
1	1	–	–

Zustandsübergangstabelle

Q	$\rightarrow$	Q^+	R	S
0	$\rightarrow$	0	X	0
0	$\rightarrow$	1	0	1
1	$\rightarrow$	0	1	0
1	$\rightarrow$	1	0	X

Zustandsgraph



Boolesche Gleichung

$$Q^+ := \left(S \wedge \overline{R}\right) \vee \left(\overline{S} \wedge \overline{R} \wedge Q\right)$$

mit $S \wedge R = 0$

3 Synchrone Speicherglieder (Flipflop)

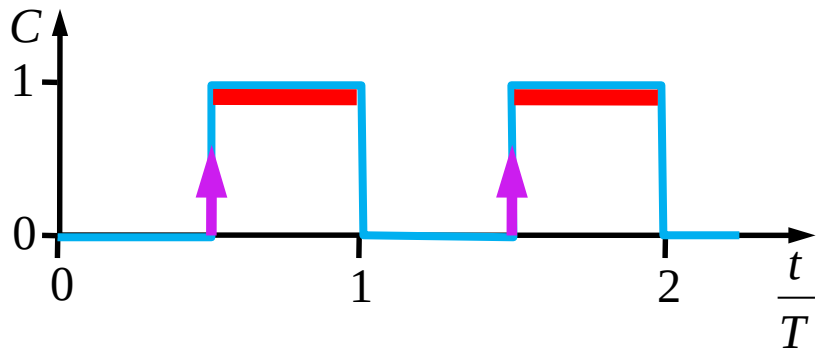
Asynchrone Speicherglieder:

Der Zustand und die Ausgangswerte ändern sich unmittelbar nach der Änderung der Eingangswerte → **Datensteuerung** (bisherige Betrachtung)

Synchrone Speicherglieder:

Der Zustand und die Ausgangswerte ändern sich synchron zu einem Taktsignal → **Taktsteuerung** (Einführung eines Taktsignales)

Taktsignal (Clock, Synchronisationssignal):

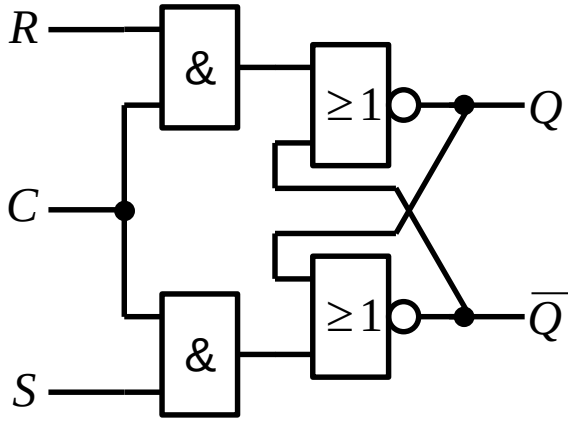


aktiver Taktzustand $C=1$
→ Taktzustandssteuerung

aktive Taktflanke $C\uparrow$ o. $C: 0 \rightarrow 1$
→ Taktflankensteuerung

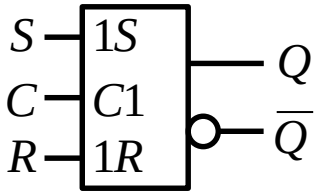
komplexere Steuerungen sind möglich

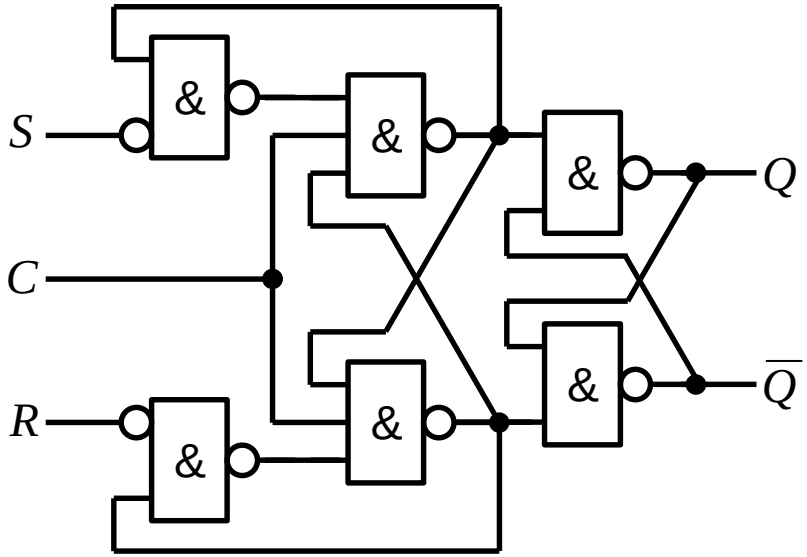
Synchrones SR-Flipflop



taktzustandsgesteuert

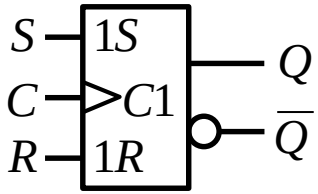
Schaltsymbol SR-FF TZS





taktflankengesteuert

Schaltsymbol SR-FF TFS



Synchrones SR-FF, Zustandsfolgetabelle

Taktzustandsgesteuertes SR-Flipflop

R	S	C	Q^+	
0	0	1	Q	speichern
0	1	1	1	setzen
1	0	1	0	rücksetzen
1	1	1	–	nicht zulässig
X	X	0	Q	speichern

Taktflankengesteuertes SR-Flipflop

R	S	C	Q^+	
0	0	$\uparrow$	Q	speichern
0	1	$\uparrow$	1	setzen
1	0	$\uparrow$	0	rücksetzen
1	1	$\uparrow$	–	nicht zulässig
X	X	sonst	Q	speichern

Auf die Angabe des Taktes in der Zustandsfolgetabelle und im Zustandsgraphen kann verzichtet werden, wenn als Nebenbedingung die Art der Taktsteuerung mit angegeben wird (TZS oder TFS).

Ist der Takt nicht aktiv (Zustand oder Flanke), so speichert das Flipflop in jedem Fall den aktuellen Zustand.

4 Komplexe Speicherglieder, D-Flipflop

Ansteuerschaltungen für das SR-Flipflop zur Vermeidung der nicht zulässigen Eingangsbelegung $S=R=1$ und für spezielle Ansteuervarianten und Funktionen

Flipflop Varianten:

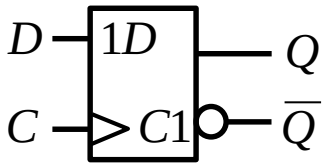
Typ	Eingänge	Funktionen
T-FF	1	invertieren, speichern
D-FF	1	setzen, rücksetzen
SR-FF	2	speichern, setzen, rücksetzen
JK-FF	2	speichern, setzen, rücksetzen, invertieren

Die einzelnen Flipflop-Typen lassen sich durch Zusatzbeschaltungen ineinander überführen. Alle Flipflop-Typen sind gleichwertig anwendbar.

In der Computertechnik dominieren D-Flipflop (Delay-Flipflop) mit Taktflankensteuerung oder mit Taktzustandssteuerung.

D-Flipflop TFS, Darstellungsvarianten

Schaltsymbol



D-Flipflop TFS

Delay-Flipflop

Zustandsfolgetabelle

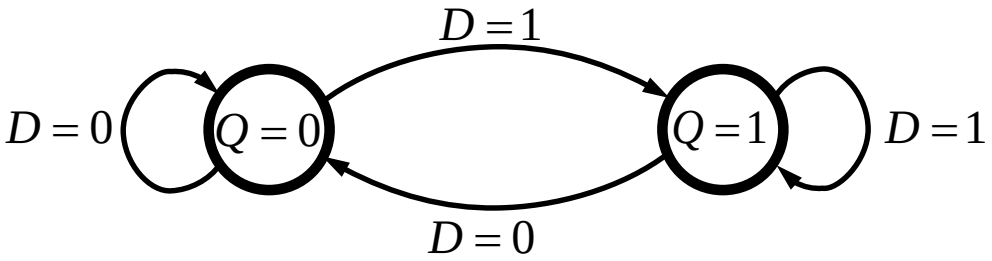
D	Q^+
0	0
1	1

Zustandsübergangstabelle

Q	$\rightarrow$	Q^+	D
0	$\rightarrow$	0	0
0	$\rightarrow$	1	1
1	$\rightarrow$	0	0
1	$\rightarrow$	1	1

Alle Zustandsübergänge erfolgen nur zur aktiven Taktflanke $C\uparrow$ (positive Taktflanke) !

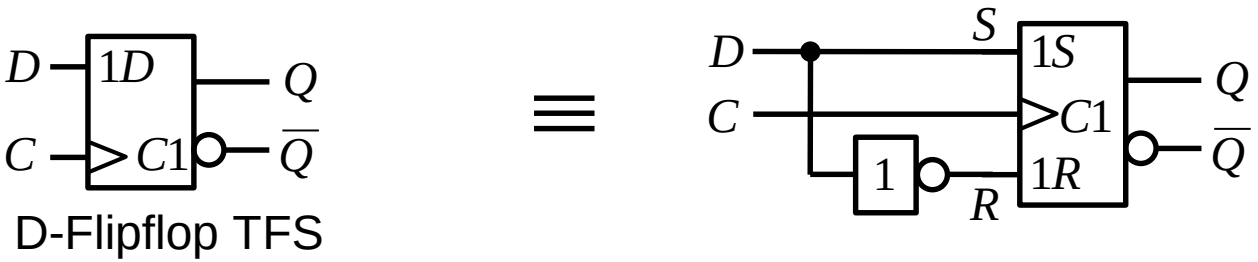
Zustandsgraph

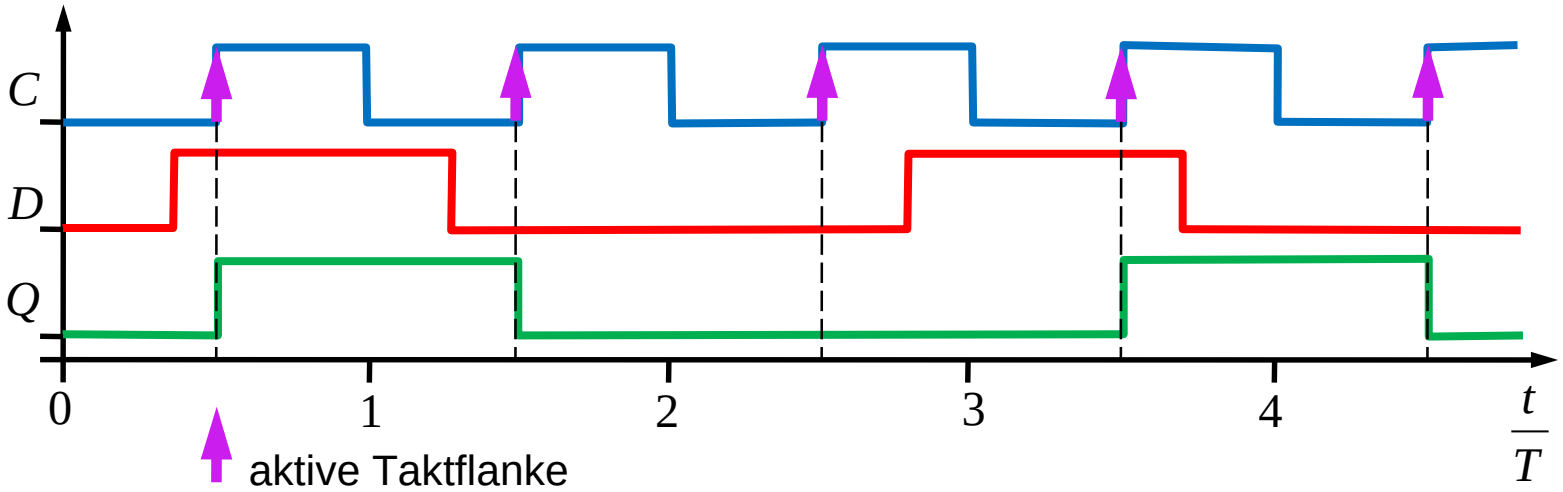


Boolesche Gleichung

$$Q^+ := D$$

D-Flipflop TFS, Datenübernahme



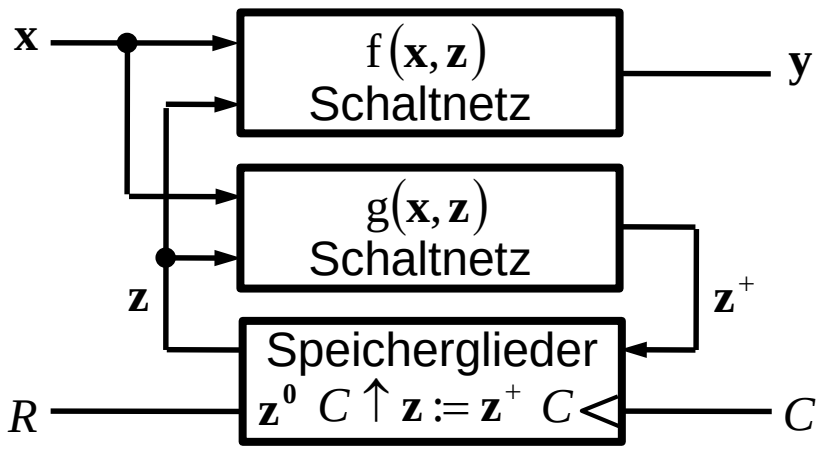


5 Schaltwerke – Sequentielle Schaltungen

Bestandteile von Schaltwerken:

- Speicherglieder (Flipflop)
- Schaltnetze
- Verbindungen, Rückführungen
- Taktsignal (→ synchrone Schaltwerke).

Allgemeine Darstellungsform (Huffman-Modell)



$\mathbf{x}$:	Eingangsvektor
$\mathbf{y}$:	Ausgangsvektor
$\mathbf{z}$:	Zustandsvektor
$\mathbf{z}^0$:	Anfangszustand
$\mathbf{z}^+$:	Folgezustandsvektor
$\mathbf{y} = f(\mathbf{x}, \mathbf{z})$	Ausgangsfunktion
$\mathbf{z}^+ = g(\mathbf{x}, \mathbf{z})$	Übergangsfunktion
C :	Taktsignal (clock)
R :	Rücksetzsignal (reset)

6 Zustandsautomaten, Automatentheorie

Automat – Modellmaschine

- Automat - abstraktes mathematisches Modell → Modellmaschine.
- Kennzeichnung durch einen inneren Zustand → Zustandsautomat.
- Automaten durchlaufen eine Abfolge von Zuständen, beginnend mit einem Anfangszustand, in Abhängigkeit von den Eingangswerten und Takt.
- Die Ausgabewerte sind nur von den aktuellen Eingabewerten und vom momentanen inneren Zustand abhängig → sequentielle Abarbeitung.

Endliche Automaten

Die Menge der möglichen Eingabezeichen (Eingabealphabet), der Ausgabezeichen (Ausgabealphabet) und die Zahl der möglichen inneren Zustände (Zustandsmenge) sind endlich.

Deterministische Automaten

Das Verhalten der Modellmaschine ist deterministisch, für eine gegebene Folge von Eingangswerten komplett vorhersagbar, determiniert.

Zustandsautomat - Schaltwerk

Automat – Schaltwerk

- Automaten bilden die Grundlage für den Entwurf und die Beschreibung von sequentiellen digitalen Systemen, Schaltwerken.
- Schaltwerke stellen eine technische Realisierung der Automaten dar.
- Schaltwerke sind deterministische endliche Automaten (DEA) (auch FSM – Finite State Machine)
- Jeder DEA kann durch ein Schaltwerk realisiert werden und umgekehrt, jedes Schaltwerk kann als DEA beschrieben werden.
- Nichtdeterministische endliche Automaten (NDEA) können durch geeignete Maßnahmen in deterministische endliche Automaten (DEA) umgeformt werden.

Deterministische Endliche Automaten (DEA)

Ein deterministischer endlicher Automat (DEA) $\mathbf{A}$ kann als 7-Tupel definiert werden:

$$\mathbf{A} = \left(\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathrm{f}, \mathrm{g}, \mathbf{z}^0, \mathbf{E}\right)$$

$$g : \mathbf{X} \times \mathbf{Z} \rightarrow \mathbf{Z}$$
$$f : \mathbf{X} \times \mathbf{Z} \rightarrow \mathbf{Y}$$

$\mathbf{X}:$	Eingabealphabet
$\mathbf{Y}:$	Ausgabealphabet
$\mathbf{Z}:$	Zustandsmenge
$\mathbf{y} = \mathrm{f}(\mathbf{x}, \mathbf{z})$	Ausgangsfunktion
$\mathbf{z}^+ = \mathrm{g}(\mathbf{x}, \mathbf{z})$	Übergangsfunktion
$\mathbf{z}^0 \in \mathbf{Z}$	Anfangszustand
$\mathbf{E} \subseteq \mathbf{Z}$	Finalzustandsmenge
$\mathbf{x} \in \mathbf{X}$	Eingangsvektor
$\mathbf{y} \in \mathbf{Y}$	Ausgangsvektor
$\mathbf{z} \in \mathbf{Z}$	Zustandsvektor
$\mathbf{z}^+ \in \mathbf{Z}$	Zustandsfolgevektor
$x \in \mathbf{U}$	Menge der Eingangsvariablen
$y \in \mathbf{V}$	Menge der Ausgangsvariablen
$z \in \mathbf{W}$	Menge der Zustandsvariablen

Beschreibung von Automaten

- Mit Hilfe der beiden Funktionen f, g und dem Anfangszustand $\mathbf{z^0}$ kann das Verhalten eines Automaten ausreichend beschrieben werden.

$\mathbf{z}^+ = \mathrm{g}(\mathbf{x},\mathbf{z})$	Übergangsfunktion, Folgezustand
$y = \mathrm{f}(\mathbf{x},\mathbf{z})$	Ausgangsfunktion, Funktionswert
$\mathbf{z}^0 = (\ )$	Anfangszustand, Startwert
$\mathbf{z} := \mathbf{z}^+$	Zustandsübergang der Speicherglieder

- Bei Schaltwerken ist zusätzlich die Beschreibung der Speicherglieder erforderlich (Typ, Steuerung).
- Die Anzahl der Zustände, die ein Automat bei Vorgabe von f, g und $\mathbf{z^0}$ überhaupt einnehmen kann, kann kleiner sein als die maximal mögliche Zustandsanzahl.

Zustandsfolge, Ausgangsfolge

$\mathbf{x}$	$\mathbf{z}$	$\mathbf{y}$
$\mathbf{x}^0$	$\mathbf{z}^0 = \mathbf{z}^0$	$\mathbf{y}^0 = \mathrm{f}\left(\mathbf{x}^0, \mathbf{z}^0\right)$
$\mathbf{x}^1$	$\mathbf{z}^1 = \mathrm{g}\left(\mathbf{x}^0, \mathbf{z}^0\right)$	$\mathbf{y}^1 = \mathrm{f}\left(\mathbf{x}^1, \mathbf{z}^1\right) = \mathrm{f}\left(\mathbf{x}^1, \mathrm{g}\left(\mathbf{x}^0, \mathbf{z}^0\right)\right)$
$\mathbf{x}^2$	$\mathbf{z}^2 = \mathrm{g}\left(\mathbf{x}^1, \mathbf{z}^1\right) = \mathrm{g}\left(\mathbf{x}^1, \mathrm{g}\left(\mathbf{x}^0, \mathbf{z}^0\right)\right)$	$\mathbf{y}^2 = \mathrm{f}\left(\mathbf{x}^2, \mathbf{z}^2\right) = \mathrm{f}\left(\mathbf{x}^2, \mathrm{g}\left(\mathbf{x}^1, \mathrm{g}\left(\mathbf{x}^0, \mathbf{z}^0\right)\right)\right)$
$\mathbf{x}^3$	$\mathbf{z}^3 = \mathrm{g}\left(\mathbf{x}^2, \mathbf{z}^2\right) = \mathrm{g}\left(\mathbf{x}^2, \mathrm{g}\left(\mathbf{x}^1, \mathrm{g}\left(\mathbf{x}^0, \mathbf{z}^0\right)\right)\right)$	$\mathbf{y}^3 = \mathrm{f}\left(\mathbf{x}^3, \mathbf{z}^3\right) = \mathrm{f}\left(\mathbf{x}^3, \mathrm{g}\left(\mathbf{x}^2, \mathrm{g}\left(\mathbf{x}^1, \mathrm{g}\left(\mathbf{x}^0, \mathbf{z}^0\right)\right)\right)\right)$
$\mathbf{x}^n$	$\mathbf{z}^n = \mathrm{g}\left(\mathbf{x}^{n-1}, \mathbf{z}^{n-1}\right)$	$\mathbf{y}^n = \mathrm{f}\left(\mathbf{x}^n, \mathbf{z}^n\right)$

Zustandsfolge

$$\mathbf{z}^n = \mathrm{g}\left(\mathbf{x}^{n-1}, \mathbf{z}^{n-1}\right) = \mathrm{g}\left(\mathbf{x}^{n-1}, \mathrm{g}\left(\mathbf{x}^{n-2}, \ldots \mathrm{g}\left(\mathbf{x}^1, \mathrm{g}\left(\mathbf{x}^0, \mathbf{z}^0\right)\right)\right)\right)$$

Ausgangsfolge

$$\mathbf{y}^n = \mathrm{f}\left(\mathbf{x}^n, \mathbf{z}^n\right) = \mathrm{f}\left(\mathbf{x}^n, \mathrm{g}\left(\mathbf{x}^{n-1}, \ldots \mathrm{g}\left(\mathbf{x}^1, \mathrm{g}\left(\mathbf{x}^0, \mathbf{z}^0\right)\right)\right)\right)$$

Zustandsbestimmung

Der Zustand $\mathbf{z}^n$ eines DEA zum Zeitpunkt t_n bestimmt sich entweder aus dem Zustand $\mathbf{z}^{n-1}$ und dem Eingangswert $\mathbf{x}^{n-1}$ zum vorherigen Zeitpunkt t_{n-1} oder aus dem Anfangszustand $\mathbf{z}^0$ und der gesamten Folge der Eingangswerte $\mathbf{x}^0$, …, $\mathbf{x}^{n-1}$ von t_0 beginnend, bis zum vorherigen Zeitpunkt t_{n-1}.

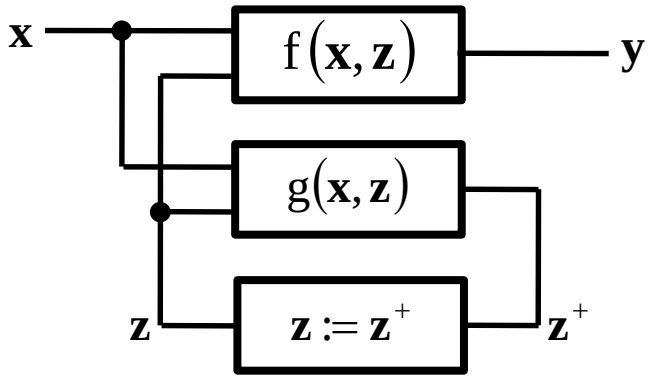
Im momentanen inneren Zustand des Automaten sind implizit alle seine vorhergehenden Zustände bis zum Anfangszustand enthalten.

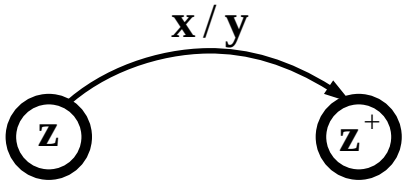
Beginnend mit einem Anfangszustand $\mathbf{z}^0$ bestimmt eine Folge von Eingangswerten $\mathbf{x}^0$, …, $\mathbf{x}^n$ eine Folge von Zuständen $\mathbf{z}^1$, …, $\mathbf{z}^n$ und entsprechend eine Folge von Ausgangswerten $\mathbf{y}^0$, …, $\mathbf{y}^n$.

➔ Der Anfangszustand $\mathbf{z}^0$ ist dabei von entscheidender Bedeutung.

➔ Hat der Zustandsvektor $\mathbf{z}$ eine Dimension von i (Zustandsvariablen), dann sind maximal 2^i verschieden Zustände möglich → Zustandskodierung.

Mealy-Automat (allgemeinste Form)



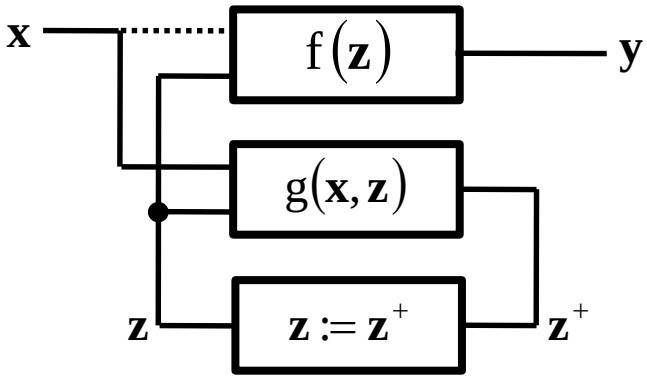


Zustandsgraph

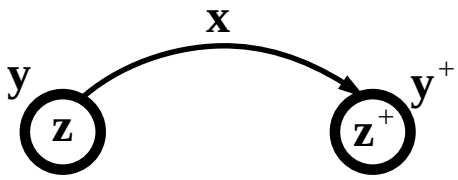
$$\mathbf{z}^{+} = \mathrm{g}(\mathbf{x}, \mathbf{z})$$
$$\mathbf{y} = \mathrm{f}(\mathbf{x}, \mathbf{z})$$
$$\mathbf{z}^{0} = (\ )$$
$$\mathbf{y}^{+} = \mathrm{f}(\mathbf{x}^{+}, \mathbf{z}^{+}) = \mathrm{f}(\mathbf{x}^{+}, \mathrm{g}(\mathbf{x}, \mathbf{z}))$$

Mealy-Automaten sind **übergangsorientiert**.
Änderungen des Einganges beeinflussen sofort den Ausgang. Sie stellen die allgemeinste Form der deterministischen endlichen Automaten dar.

Moore-Automat



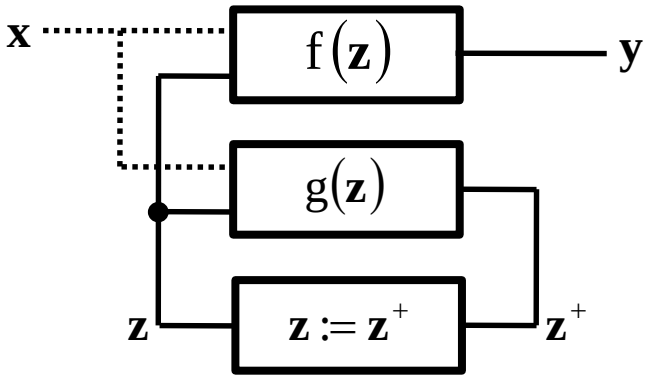
$$z^+ = g(x, z)$$
$$y = f(z)$$
$$z^0 = ( )$$
$$y^+ = f(z^+) = f(g(x, z))$$



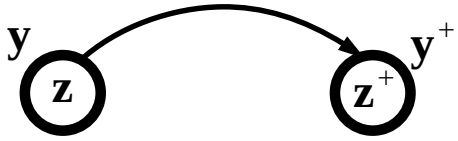
Zustandsgraph

Moore-Automaten sind **zustandsorientiert**.
Änderungen des Einganges beeinflussen den Ausgang erst zum Folgezustand. Sie stellen einen Sonderfall des Mealy-Automaten dar.

Autonomer Automat



$$z^+ = g(z)$$
$$y = f(z)$$
$$z^0 = ( )$$
$$y^+ = f(z^+) = f(g(z))$$



Zustandsgraph
(nur eine abgehende Kante pro Knoten)

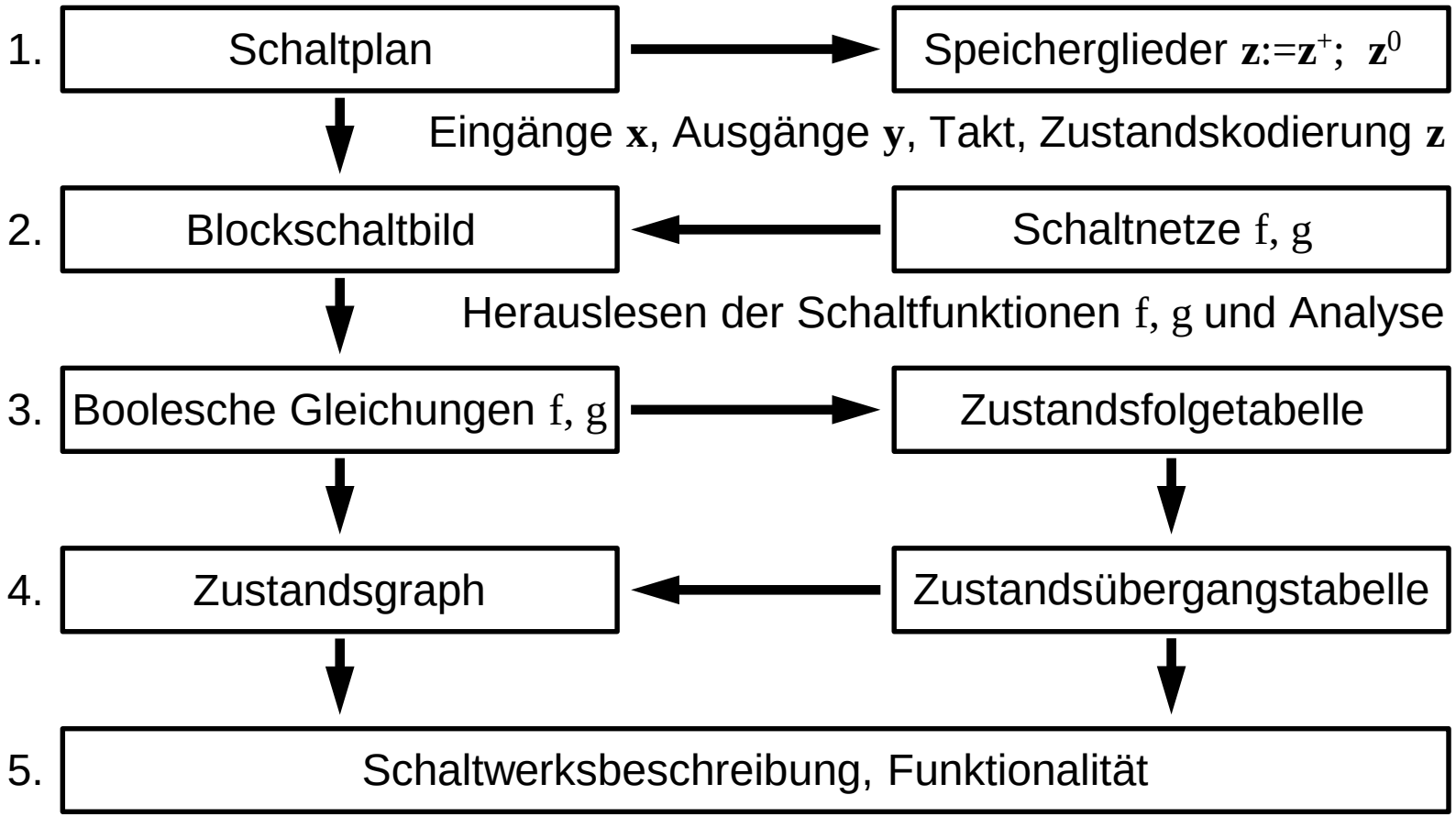
Autonome Automaten sind nicht eingangsgesteuert.
Änderungen des Einganges beeinflussen somit weder den Ausgang noch den Folgezustand. Sie stellen einen Sonderfall des Moore-Automaten dar und verfügen nicht über einen Eingang.

7 Darstellung von Schaltwerken

Es gibt verschiedene mögliche Formen der Schaltwerksdarstellung. Alle Varianten sind inhaltlich gleichwertig und können ineinander überführt werden. Bezüglich der Anschaulichkeit und Nutzbarkeit gibt es Unterschiede.

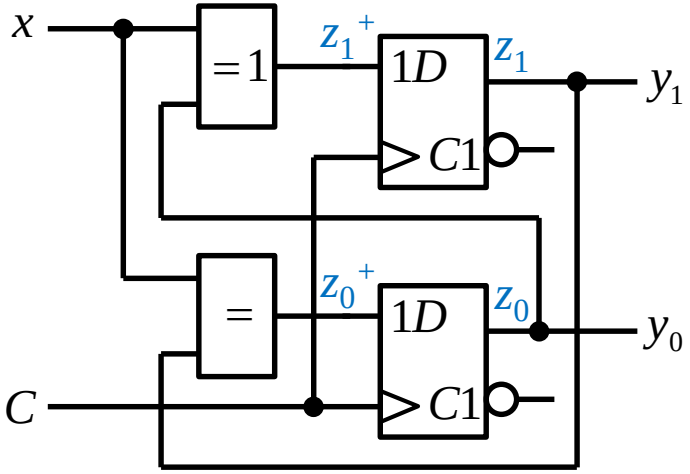
- Boolesche Gleichungen (Automatengleichungen)
- Zustandsfolgetabelle (Automatentabelle)
- Zustandsübergangstabelle
- Zustandsgraph
- Impulsfolgediagramm
- Schaltplan (Logikplan) Schaltnetze, Speicherglieder
- Schaltsymbol (DIN-Norm)
- Hardwarebeschreibungssprachen (VHDL, Verilog, SystemC, …)

8 Analyse und Synthese von Schaltwerken

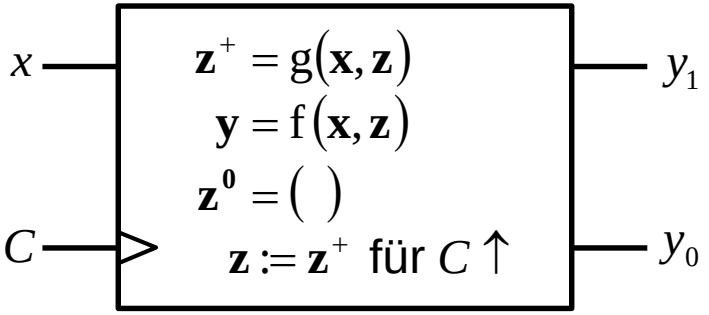


Analysebeispiel: Ausgangspunkt Schaltplan

Ausgangspunkt Schaltplan



Blockschaltbild



Zustandskodierung:

$$\mathbf{z} = (z_1, z_0)$$
$$\mathbf{z}^+ = (z_1^+, z_0^+)$$
$$\mathbf{x} = (x)$$
$$\mathbf{y} = (y_1, y_0)$$

Moore-Automat: $\mathbf{y} = \mathrm{f}(\mathbf{z})$

Boolesche Gleichungen

$$z_1^+ = x\bar{z}_0 + \bar{x}z_0 \;;\quad z_1^0 = 0$$
$$z_0^+ = xz_1 + \bar{x}\bar{z}_1 \;;\quad z_0^0 = 0$$
$$y_1 = z_1$$
$$y_0 = z_0$$

Zustandsfolgetabelle (Automatentabelle)

x	$\mathbf{x}$	z_1	z_0	$\mathbf{z}$	$\mathbf{z}^+$	z_1^+	z_0^+	y_1	y_0	$\mathbf{y}$
0	$\mathbf{x_0}$	0	0	$\mathbf{z_0}$	$\mathbf{z}_1$	0	1	0	0	$\mathbf{y_0}$
0	$\mathbf{x_0}$	0	1	$\mathbf{z_1}$	$\mathbf{z}_3$	1	1	0	1	$\mathbf{y}_1$
0	$\mathbf{x_0}$	1	0	$\mathbf{z}_2$	$\mathbf{z}_0$	0	0	1	0	$\mathbf{y}_2$
0	$\mathbf{x_0}$	1	1	$\mathbf{z}_3$	$\mathbf{z}_2$	1	0	1	1	$\mathbf{y}_3$
1	$\mathbf{x}_1$	0	0	$\mathbf{z}_0$	$\mathbf{z}_2$	1	0	0	0	$\mathbf{y_0}$
1	$\mathbf{x}_1$	0	1	$\mathbf{z}_1$	$\mathbf{z}_0$	0	0	0	1	$\mathbf{y}_1$
1	$\mathbf{x}_1$	1	0	$\mathbf{z}_2$	$\mathbf{z}_3$	1	1	1	0	$\mathbf{y}_2$
1	$\mathbf{x}_1$	1	1	$\mathbf{z}_3$	$\mathbf{z}_1$	0	1	1	1	$\mathbf{y}_3$

Anfangszustand.

$$\mathbf{x}_0 = (0)$$
$$\mathbf{x}_1 = (1)$$
$$\mathbf{y}_0 = (0,0)$$
$$\mathbf{y}_1 = (0,1)$$
$$\mathbf{y}_2 = (1,0)$$
$$\mathbf{y}_3 = (1,1)$$
$$\mathbf{z}_0 = (0,0)$$
$$\mathbf{z}_1 = (0,1)$$
$$\mathbf{z}_2 = (1,0)$$
$$\mathbf{z}_3 = (1,1)$$
$$\mathbf{z}^0 = \mathbf{z}_0$$

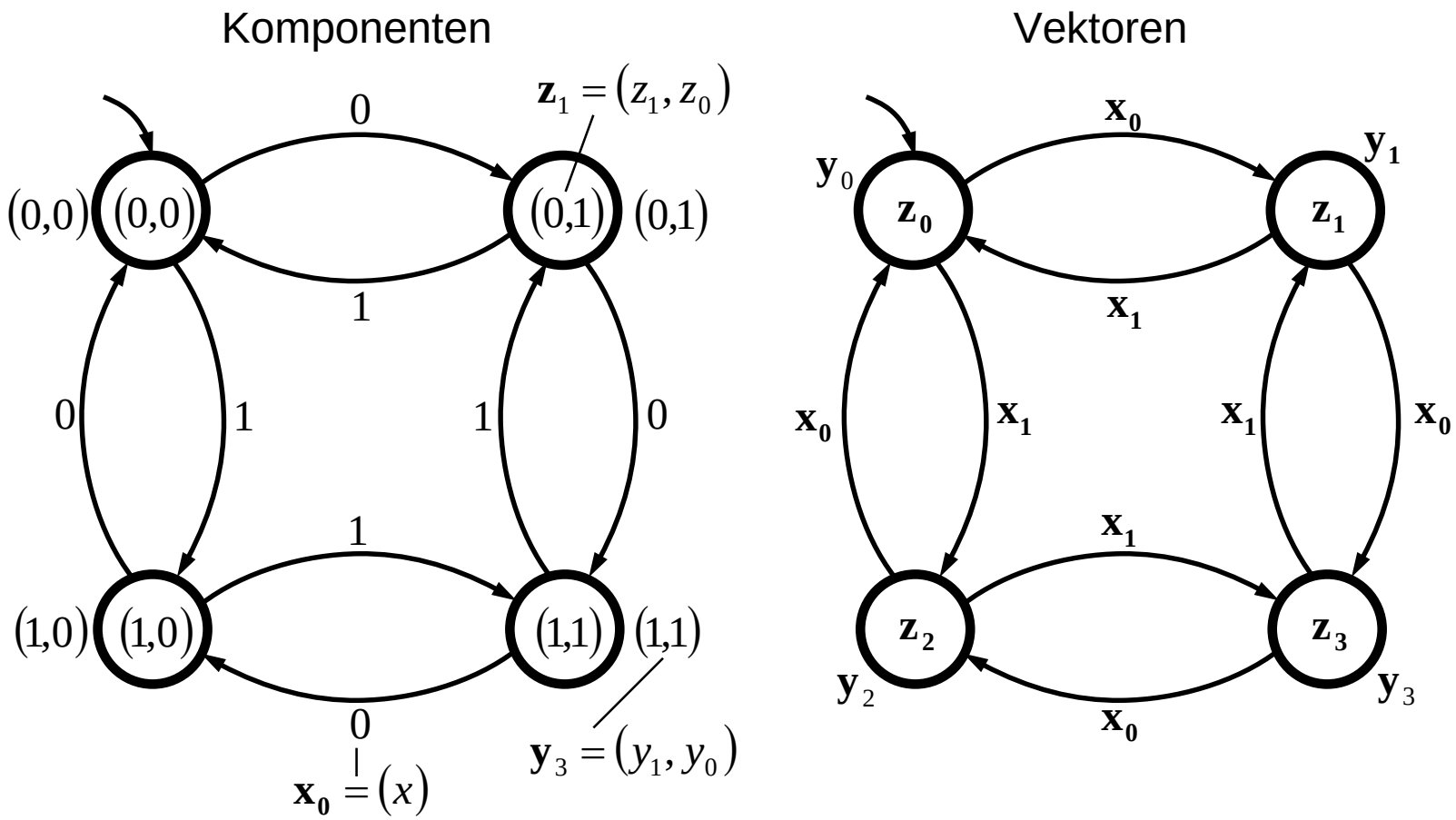
Angabe von Vektoren oder und Komponenten, je nach Übersichtlichkeit.

Zustandsübergangstabelle

			$\mathbf{x_0}$	$x=$	0	$\mathbf{x_1}$	$x=$	1			
z_1	z_0	$\mathbf{z}$	$\mathbf{z^+}$	z_1^+	z_0^+	$\mathbf{z^+}$	z_1^+	z_0^+	y_1	y_0	$\mathbf{y}$
0	0	$\mathbf{z_0}$	$\mathbf{z_1}$	0	1	$\mathbf{z_2}$	1	0	0	0	$\mathbf{y_0}$
0	1	$\mathbf{z_1}$	$\mathbf{z_3}$	1	1	$\mathbf{z_0}$	0	0	0	1	$\mathbf{y_1}$
1	0	$\mathbf{z_2}$	$\mathbf{z_0}$	0	0	$\mathbf{z_3}$	1	1	1	0	$\mathbf{y_2}$
1	1	$\mathbf{z_3}$	$\mathbf{z_2}$	1	0	$\mathbf{z_1}$	0	1	1	1	$\mathbf{y_3}$

Vektoren und oder Komponenten
Bei einem Mealy-Automaten ist der Ausgangsvektor pro Eingangsvektor anzugeben, parallel zu den Folgezuständen.
Beim Moore-Automaten reicht eine Angabe parallel zum Zustand aus.

Zustandsgraph



Schaltwerksbeschreibung

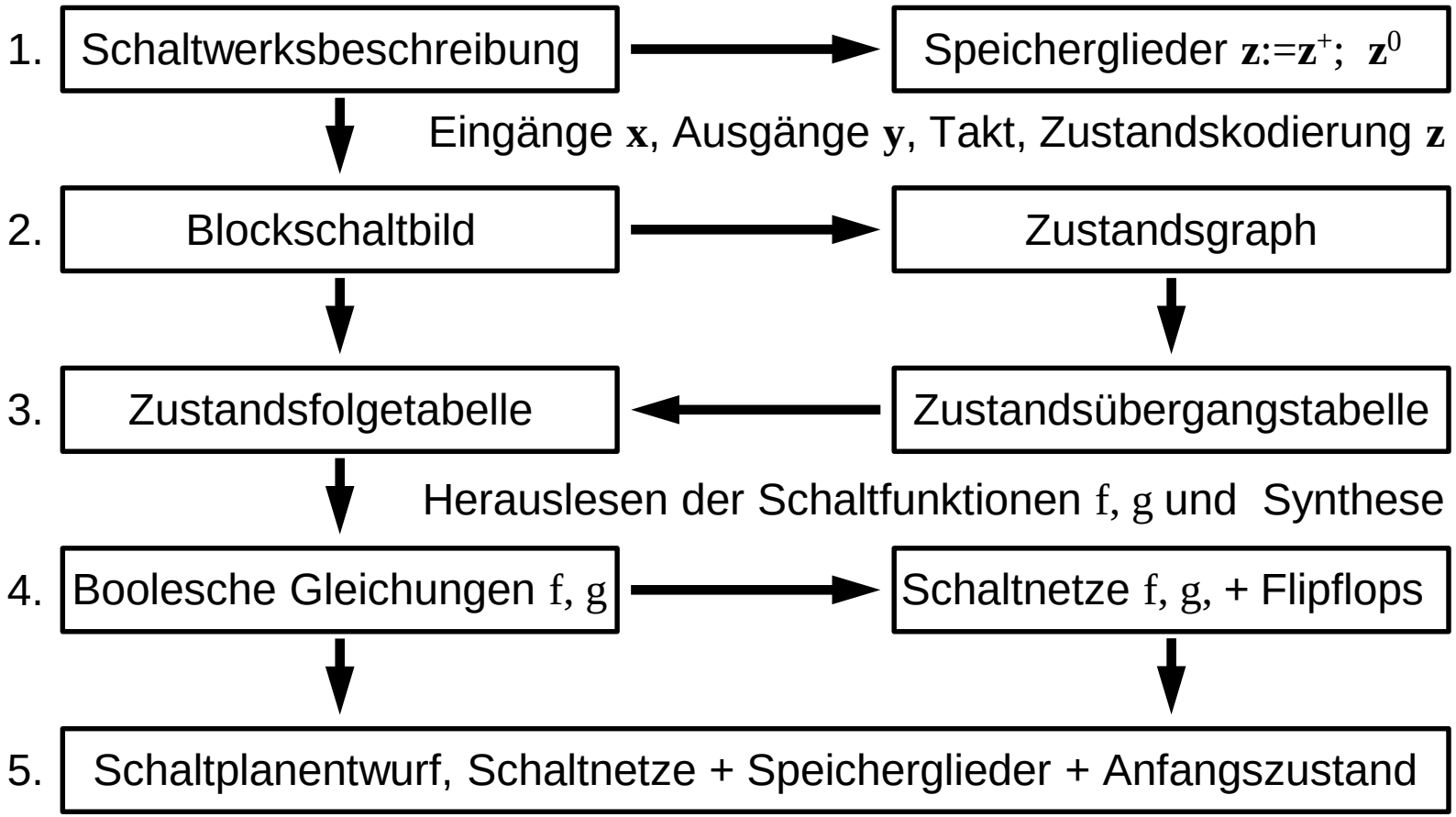
Das im Schaltplan gegebene Schaltwerk realisiert folgende Eigenschaften:

- 2 Speicherglieder → 4 Zustände → $y = f(z)$ Moore-Automat
- folgende Zustandsfolge wird in Abhängigkeit vom Eingang realisiert:

  $x = 0: \ (0,0) \to (0,1) \to (1,1) \to (1,0) \to (0,0) \to (0,1) \to \ldots$
  $x = 1: \ (0,0) \to (1,0) \to (1,1) \to (0,1) \to (0,0) \to (1,0) \to \ldots$
- es handelt sich um einen 2-bit Gray-Code zyklischen Synchronzähler mit Vor- und Rückwärtssteuerung (Zustandskodierung Gray-Code)
- Zählrichtung kann in jedem beliebigen Zustand umgekehrt werden

  $x = 0:$ vorwärts zählen
  $x = 1:$ rückwärts zählen
- ausgegeben wird direkt der Zählerzustand (Gray-Code)
- Startzustand ist $\mathbf{z}^0 = \mathbf{z}_0 = (0,0)$.

Synthese von Schaltwerken

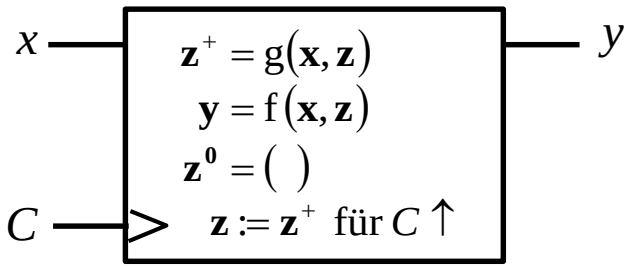


Beispiel Schaltwerkssynthese

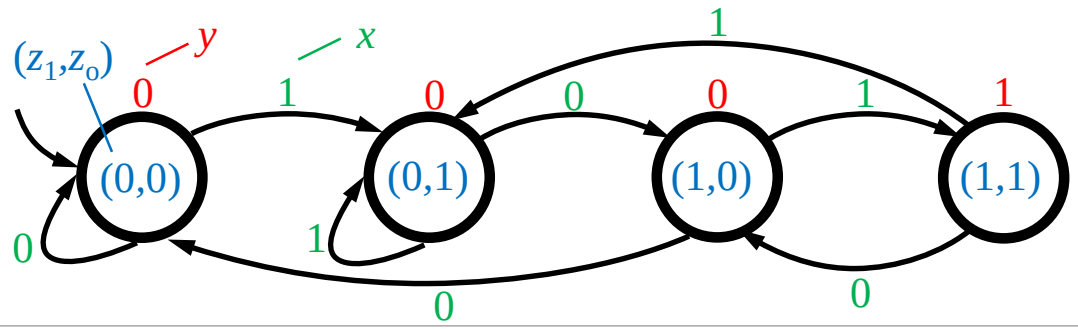
Aufgabenstellung: Es ist ein Schaltwerk zu realisieren, das die 4-bit Folge 1-0-1 einer synchronen Eingabequelle x automatisch erkennt und dann am Ausgang $y = 1$ ausgibt, sonst $y = 0$. Es sind D-FF TFS zu verwenden.

Eingang: x
Ausgang: y
Zustände: 4 → ld 4=2 → 2 D-FF
Zustandsvektor: z_1, z_0
Startzustand: $z_1 = 0, z_0 = 0$

Blockschaltbild:



Zustandsgraph:



$y = f(z_1,z_0), y \neq f(x)$
→ Moore Automat

Eingabe an der Kante
Ausgabe am Knoten
Zustand im Knoten

Zustandsfolgetabelle

C	x	z_1	z_0	z_1^+	z_0^+	y	C^+
0	0	0	0	0	0	0	0
1	0	0	1	1	0	0	2
2	0	1	0	0	0	0	0
3	0	1	1	1	0	1	2
0	1	0	0	0	1	0	1
1	1	0	1	0	1	0	1
2	1	1	0	1	1	0	3
3	1	1	1	0	1	1	1

C – Zustandscodierung
C^+ – Folgezustand

Übergangsfunktion:

$\mathbf{z}^+ = g(\mathbf{x},\mathbf{z})$, $\mathbf{z} = (z_1,z_0)$, $\mathbf{x} = (x)$

$z_0^+ = x$

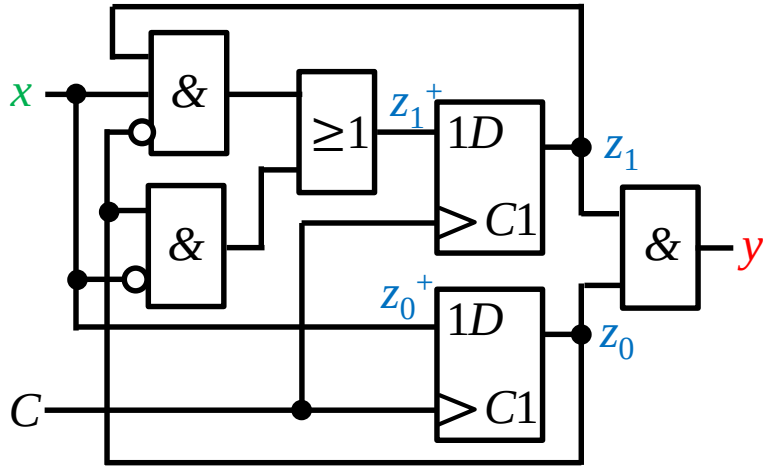
$z_1^+ = \bar{x} z_0 + x z_1 \bar{z}_0$

Ausgangsfunktion:

$\mathbf{y} = f(\mathbf{z})$, $\mathbf{y} = (y)$

$y = z_1 z_0$

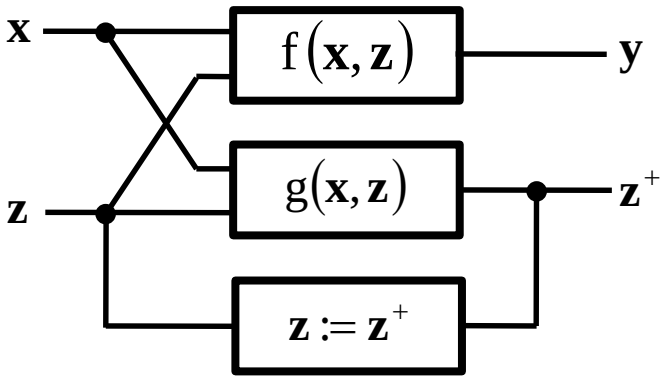
Schaltplan:

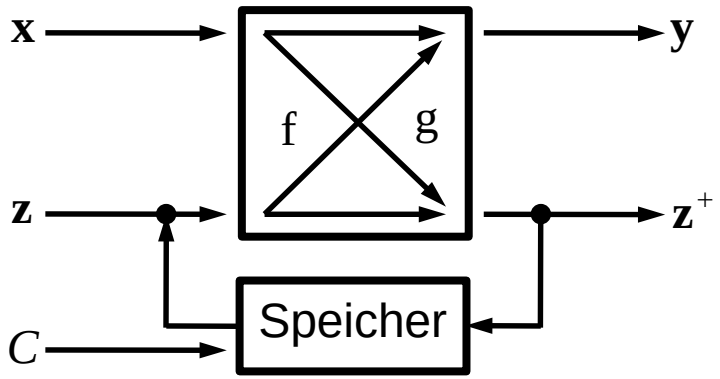


Typen von Schaltwerken

Mealy-Automat

Allgemeiner Automat $\mathbf{A} = (\mathbf{X}, \mathbf{Y}, \mathbf{Z}, f, g, \mathbf{z}^0, \mathbf{E})$





Huffman Normalform

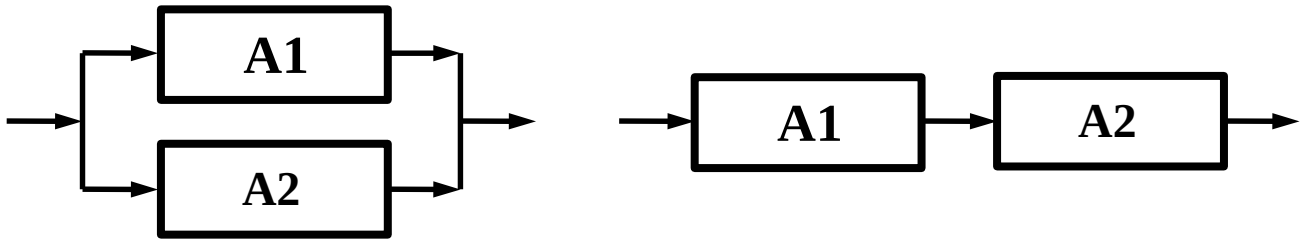
Transduktor

Ein Mealy-Automat, der nur ein Ausgabezeichen pro Ausgabe zulässt.

Komposition und Dekomposition von Schaltwerken

Ersetzung eines komplexen Schaltwerkes mit vielen Zuständen durch eine Zusammenschaltung mehrerer einfacher Schaltwerke mit weniger Zuständen bei gleichbleibender Gesamtfunktion.

Reihen- und Parallelschaltung von Schaltwerken (Automaten)



$$A = A1 \times A2$$

Kreuzproduktautomat

Die Zustandsanzahlen der Einzelautomaten multiplizieren sich!

Minimierung von Schaltwerken

- Zustände eines DEA die vom Anfangszustand aus nicht erreichbar sind, können entfernt werden.
- Zustände eines DEA heißen äquivalent, wenn sie für alle möglichen Eingangsbelegungen identisches Folgezustände haben:

$$\mathbf{p} \equiv \mathbf{q}, \text{ wenn für alle } \mathbf{x} \in \mathbf{X}$$
$$g(\mathbf{p}, \mathbf{x}) \in \mathbf{Z} \Leftrightarrow g(\mathbf{q}, \mathbf{x}) \in \mathbf{Z}$$

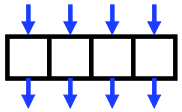
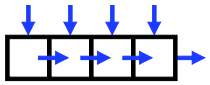
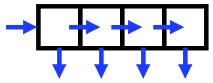
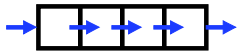
Äquivalente Zustände können zusammengefasst werden.

Eine Zustandsreduktion ist nicht immer sinnvoll:

- Eine Zustandsreduktion kann zu deutlich komplexeren Schaltnetzen f und g des Automaten führen (z.B. one-hot Code).
- Eine Zustandsreduktion führt nicht zwangsläufig zu einer Reduktion der Anzahl der Speicherglieder: n Speicherglieder → $2^{n-1}+1 \ldots 2^n$ Zustände.

Beispiel Register (Pufferspeicher)

Verkopplung (Verkettung) einzelner identischer Registerzellen (Flipflops) zu einem Register (Registerspeicher) der Länge n. Alle Registerzellen verfügen über gemeinsame Takt- und Steuersignale.

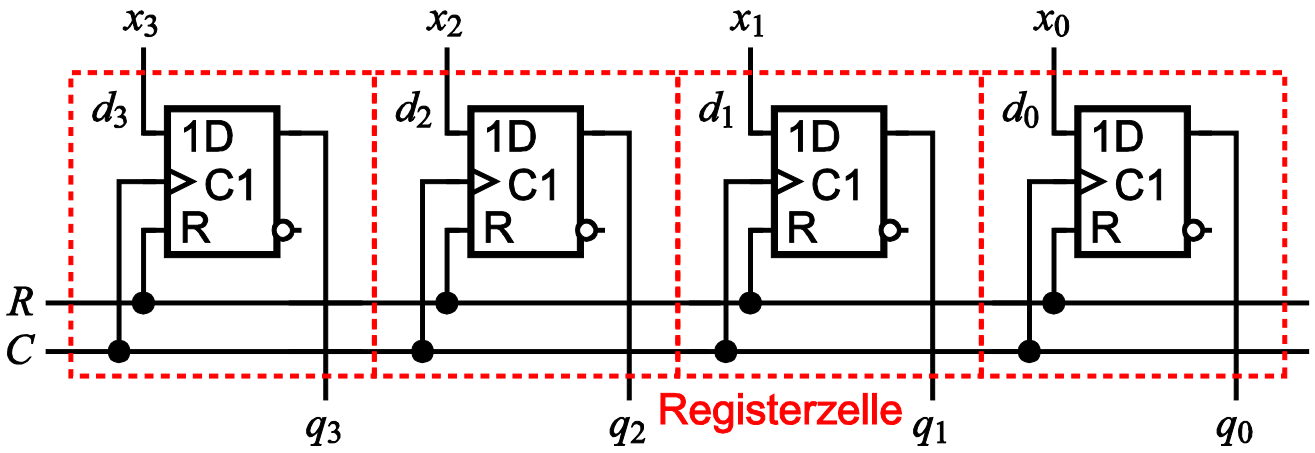
Eingang	Ausgang	Funktion	Beispiel
parallel	parallel	PIPO	
parallel	seriell	PISO	
seriell	parallel	SIPO	
seriell	seriell	SISO	

→ bitweise Datenflussrichtung im Register

PIPO – parallel-in parallel-out
PISO – parallel-in serial-out
SIPO – serial-in parallel-out
SISO – serial-in serial-out

Die Funktionen PIPO, PISO wie auch SIPO, SISO sind auch kombinierbar zu PIPOSO bzw. SIPOSO.

4-Bit D-Auffangregister (latch)



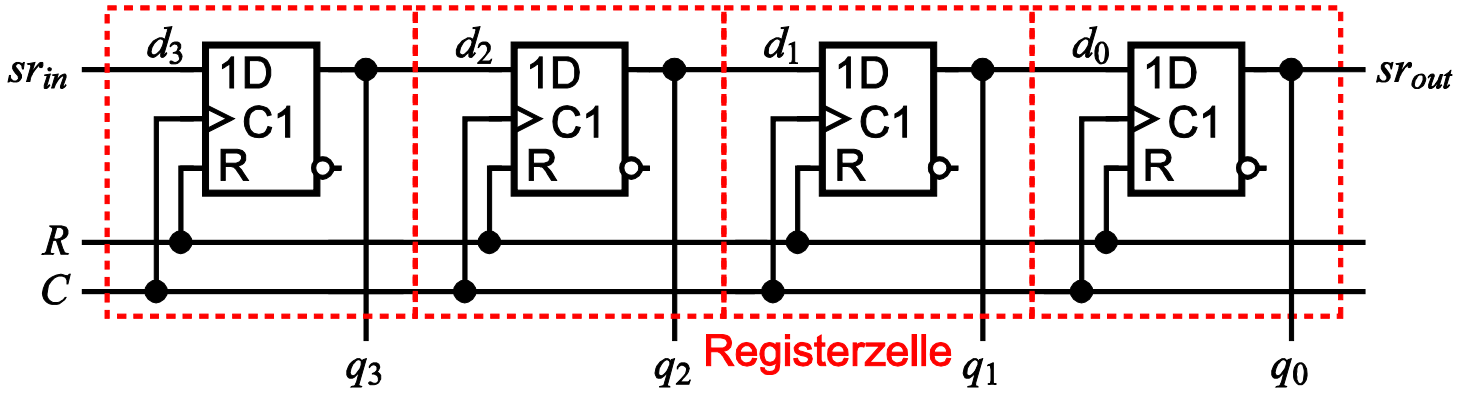
Eingang : parallel x_0, x_1, x_2, x_3

Ausgang : parallel q_0, q_1, q_2, q_3

Betriebsart (PIPO)	q_3^+	q_2^+	q_1^+	q_0^+
parallel laden	x_3	x_2	x_1	x_0

$R = 1$ - asynchrones rücksetzen

4-Bit D-Rechts-Schieberegister (right shift)

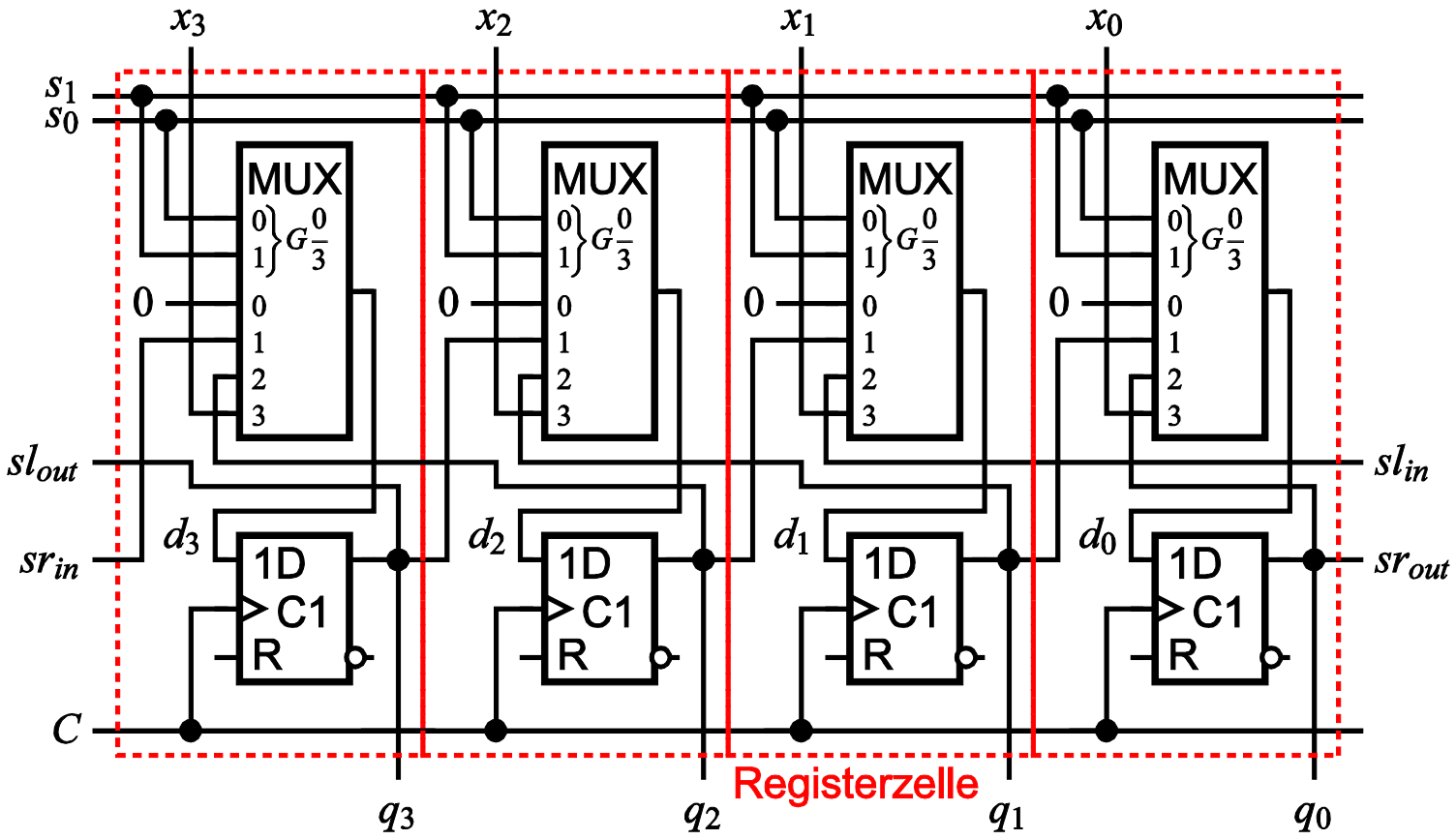


Eingang : seriell sr_{in}
Ausgang : seriell sr_{out}
parallel q_0, q_1, q_2, q_3

Betriebsart (SISO, SIPO)	q_3^+	q_2^+	q_1^+	q_0^+ sr_{out}^+
rechts schieben	sr_{in}	q_3	q_2	q_1

$R = 1$ - asynchrones rücksetzen

4-Bit Universalschieberegister (universal shift)



Steuereingänge	:		s_0, s_1
Eingänge	:	seriell	sr_{in}, sl_{in}
		parallel	x_0, x_1, x_2, x_3
Ausgänge	:	seriell	sr_{out}, sl_{out}
		parallel	q_0, q_1, q_2, q_3

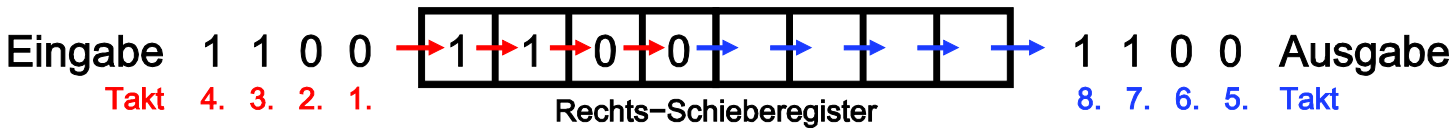
Betriebsarten:

Betriebsart (PIPO, SISO, PISO, SIPO)	s_1	s_0	q_3^+ sl_{out}^+	q_2^+	q_1^+	q_0^+ sr_{out}^+
parallel laden	1	1	x_3	x_2	x_1	x_0
rechts schieben	0	1	sr_{in}	q_3	q_2	q_1
links schieben	1	0	q_2	q_1	q_0	sl_{in}
synchrones rücksetzen	0	0	0	0	0	0

Schieberegister (SRG) als serieller Speicher

FIFO Prinzip (First-In-First-Out – Warteschlange, queue)

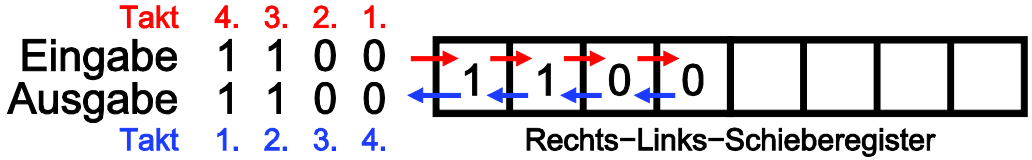
Serielle Eingabe in das Schieberegister in vorgegebener Schieberichtung.
→ Serielle Ausgabe in der gleichen Schieberichtung.



Ein- und Ausgang unterliegen dem gleichen Takt, nicht unabhängig steuerbar.
Die Ausgangsverzögerung wird durch die Länge der Schiebekette bestimmt.

LIFO Prinzip (Last-In-First-Out – Kellerspeicher, Stapelspeicher, stack)

Serielle Eingabe in das Schieberegister in vorgegebener Schieberichtung.
→ Serielle Ausgabe in umgekehrter Schieberichtung.



Anwendung von Registern (SRG)

Vorteile:

- Einfacher Aufbau, unkomplizierte Realisierung.
- Sehr kurzer kombinatorischer Pfad (einfaches Schaltnetz).
- Für hohe Taktfrequenzen geeignet.

Anwendung:

- Datenspeicherung, Registerspeicher innerhalb der CPU, Pufferspeicher.
- Zwischenspeicher für serielle, parallele Schnittstelle.
- Seriell-parallel und parallel-seriell Wandler.
- Anpassung des Datenflusses zwischen unterschiedlichen Taktdomänen.
- Aufbau rückgekoppelter Schieberegister, Zähler, Teiler, Steuerwerke.

9 Zusammenfassung

- Schaltwerke – Hauptbestandteile von Computern (v. Neumann Rechner).
- Schaltwerke enthalten Schaltnetze, Speicherglieder und Vernetzungen.
- Der in den Speichergliedern gespeicherte Zustand heißt „innerer Zustand“ des Schaltwerkes. Die Speicherglieder selbst sind auch Schaltwerke.
- Bei einem Schaltwerk hängen die Werte der Ausgangsvariablen von denen der Eingangsvariablen und vom inneren Zustand des Schaltwerkes ab.
- Einer Wertefolge der Eingangsvariablen ist damit eindeutig eine Wertefolge der Ausgangsvariablen zugeordnet → Anfangszustand signifikant.
- Charakteristisch für ein Schaltwerk ist die funktionelle Bedeutung der Zeit (diskrete Zeitpunkte werden durch ein Taktsignal realisiert).
- Die Automatentheorie ist die theoretische Basis für Schaltwerke → Moore Automaten, Mealy Automaten, autonome Automaten.