



Grundlagen der Rechnerarchitektur

[CS3100.010]

Wintersemester 2014/15

Heiko Falk

Institut für Eingebettete Systeme/Echtzeitsysteme
Ingenieurwissenschaften und Informatik
Universität Ulm



Kapitel 3

Sequentielle Logik

Inhalte der Vorlesung

1. Einführung
2. Kombinatorische Logik
- 3. Sequentielle Logik**
4. Technologische Grundlagen
5. Rechnerarithmetik
6. Grundlagen der Rechnerarchitektur
7. Speicher-Hardware
8. Ein-/Ausgabe

Inhalte des Kapitels (1)

3. Sequentielle Logik

- Einleitung
 - Begriff „Sequentielle Logik“
 - Schaltwerke
 - Rückgekoppelte Gatter
- Flip-Flops
 - RS-Flip-Flop
 - Analyse
 - Bedeutung
 - Zeitverhalten
 - Asynchrone und synchrone Schaltwerke
 - Getaktete Flip-Flops
 - *Master-Slave* Flip-Flops
 - Weitere Flip-Flops (D-Flip-Flop, Register, JK- & T-Flip-Flop)
- ...

Inhalte des Kapitels (2)

3. Sequentielle Logik

- ...
- Typische Schaltwerke
 - Schieberegister
 - Asynchroner Zähler
 - Synchroner Zähler
- Systematischer Schaltwerkentwurf
 - Beispiel „Hochwassererkennung“
 - Beispiel „Sequenzerkennung“
 - Vergleich von Moore- und Mealy-Automaten
 - Einfluss des Flip-Flop-Typs
 - Zustandsreduktion von Automaten

Begriff Sequentielle Logik (1)

Kombinatorische Logik

- Schaltnetze mit verzögerungsfreien Gattern
 - Sofortiges Ergebnis beim Anlegen von Werten an den Eingängen
- Keine Zyklen/Rückkopplungen im Schaltnetz

Reicht das zur Beschreibung heutiger Rechensysteme?

- Ablaufsteuerung
- Speicherelemente
- Takterzeuger
- ...

Begriff Sequentielle Logik (2)

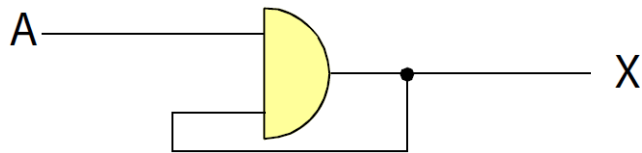
Sequentielle Logik

- Gatter haben Laufzeitverhalten
 - Annahme einer Gatterlaufzeit Δt
- Zyklen bzw. Rückkopplungen
- ☞ Schaltwerke
 - Schaltungen mit Gattern als gerichteter zyklischer Graph

Rückgekoppeltes UND-Gatter (1)

Beispiel für einfaches Schaltwerk

- Was geschieht bei Rückkopplung?



$$X(t + \Delta t) = A(t) * X(t)$$

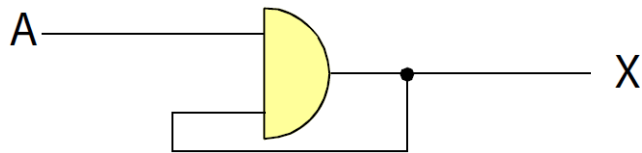
- Wertefolge für X bei Anfangskonfiguration

| A | $X(t)$ | $X(t + \Delta t)$ | $X(t + 2\Delta t)$ | $X(t + 3\Delta t)$ |
|-----|--------|-------------------|--------------------|--------------------|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

Rückgekoppeltes UND-Gatter (2)

Beispiel für einfaches Schaltwerk

- Was geschieht bei Rückkopplung?



$$X(t + \Delta t) = A(t) * X(t)$$

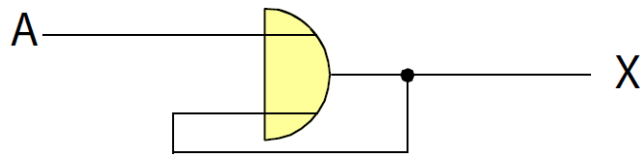
- Wertefolge für X als Funktion des vorherigen Wertes

| A | $X(t + \Delta t)$ | $X(t + 2\Delta t)$ | $X(t + 3\Delta t)$ |
|-----|-------------------|--------------------|--------------------|
| 0 | 0 | 0 | 0 |
| 1 | $X(t)$ | $X(t)$ | $X(t)$ |

Rückgekoppeltes ODER-Gatter (1)

Beispiel für einfaches Schaltwerk

- Was geschieht bei Rückkopplung?



$$X(t + \Delta t) = A(t) + X(t)$$

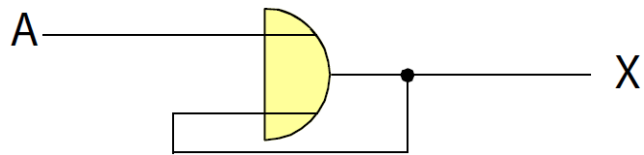
- Wertefolge für X bei Anfangskonfiguration

| A | $X(t)$ | $X(t + \Delta t)$ | $X(t + 2\Delta t)$ | $X(t + 3\Delta t)$ |
|-----|--------|-------------------|--------------------|--------------------|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 |

Rückgekoppeltes ODER-Gatter (2)

Beispiel für einfaches Schaltwerk

- Was geschieht bei Rückkopplung?



$$X(t + \Delta t) = A(t) + X(t)$$

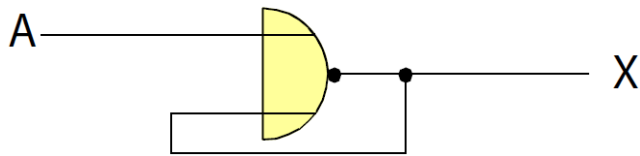
- Wertefolge für X als Funktion des vorherigen Wertes

| A | $X(t + \Delta t)$ | $X(t + 2\Delta t)$ | $X(t + 3\Delta t)$ |
|-----|-------------------|--------------------|--------------------|
| 0 | $X(t)$ | $X(t)$ | $X(t)$ |
| 1 | 1 | 1 | 1 |

Rückgekoppeltes NOR-Gatter (1)

Beispiel für einfaches Schaltwerk

- Was geschieht bei Rückkopplung?



$$X(t + \Delta t) = \overline{A(t) + X(t)}$$

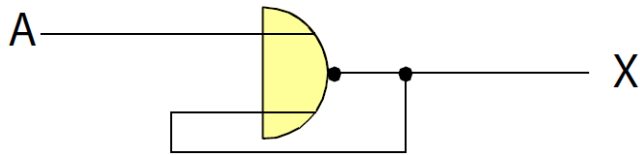
- Zur Erinnerung: Wahrheitstabelle eines NOR-Gatters

| x_1 | x_2 | $\overline{x_1 + x_2}$ |
|-------|-------|------------------------|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

Rückgekoppeltes NOR-Gatter (2)

Beispiel für einfaches Schaltwerk

- Was geschieht bei Rückkopplung?



$$X(t + \Delta t) = \overline{A(t) + X(t)}$$

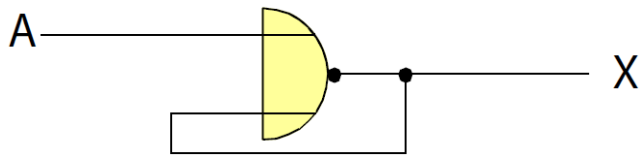
- Wertefolge für X bei Anfangskonfiguration

| A | $X(t)$ | $X(t + \Delta t)$ | $X(t + 2\Delta t)$ | $X(t + 3\Delta t)$ |
|-----|--------|-------------------|--------------------|--------------------|
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |

Rückgekoppeltes NOR-Gatter (3)

Beispiel für einfaches Schaltwerk

- Was geschieht bei Rückkopplung?



$$X(t + \Delta t) = \overline{A(t) + X(t)}$$

- Wertefolge für X als Funktion des vorherigen Wertes

| A | $X(t + \Delta t)$ | $X(t + 2\Delta t)$ | $X(t + 3\Delta t)$ |
|-----|-------------------|--------------------|--------------------|
| 0 | $\overline{X(t)}$ | $X(t)$ | $\overline{X(t)}$ |
| 1 | 0 | 0 | 0 |

☞ Die Schaltung schwingt!

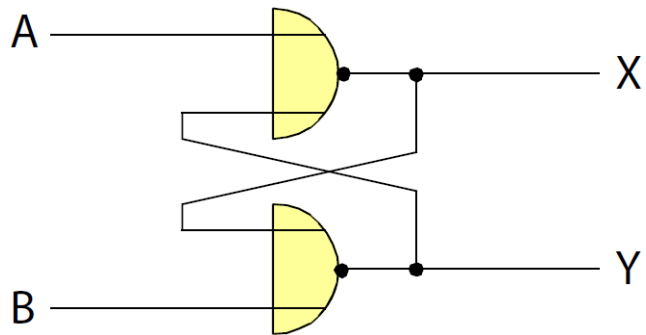
Roter Faden

3. Sequentielle Logik

- Einleitung
 - Begriff „Sequentielle Logik“
 - Schaltwerke
 - Rückgekoppelte Gatter
- Flip-Flops
- Typische Schaltwerke
- Systematischer Schaltwerkentwurf

RS-Flip-Flop

Betrachtung eines einfachen asynchronen Schaltwerks



$$X(t + \Delta t) = \overline{A(t) + Y(t)}$$

$$Y(t + \Delta t) = \overline{B(t) + X(t)}$$

- Wie verhält sich das Schaltwerk bei unterschiedlichen Eingängen?

Analyse des Schaltwerks (1)

Tabelle aller Zustandsübergänge

| A | B | $X(t)$ | $Y(t)$ | $X(t + \Delta t)$ | $Y(t + \Delta t)$ |
|-----|-----|--------|--------|-------------------|-------------------|
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 |

Analyse des Schaltwerks (2)

Tabelle aller Zustandsübergänge

| A | B | $X(t)$ | $Y(t)$ | $X(t + \Delta t)$ | $Y(t + \Delta t)$ |
|-----|-----|--------|--------|-------------------|-------------------|
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 |

Analyse des Schaltwerks (3)

Tabelle aller Zustandsübergänge

| A | B | $X(t)$ | $Y(t)$ | $X(t + \Delta t)$ | $Y(t + \Delta t)$ |
|-----|-----|--------|--------|-------------------|-------------------|
| ... | | | ... | | |
| 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 |

Analyse des Schaltwerks (4)

Tabelle aller Zustandsübergänge

| A | B | X(t) | Y(t) | X(t + Δt) | Y(t + Δt) |
|-----|---|------|------|-----------|-----------|
| ... | | | ... | | |
| 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 |

Analyse des Schaltwerks (5)

Darstellung der Werte als Funktion der Ausgangswerte

| A | B | $X(t + \Delta t)$ | $Y(t + \Delta t)$ | $X(t + 2\Delta t)$ | $Y(t + 2\Delta t)$ | $X(t + 3\Delta t)$ | $Y(t + 3\Delta t)$ |
|-----|-----|-------------------|-------------------|--------------------|--------------------|--------------------|--------------------|
| 0 | 0 | $\overline{Y(t)}$ | $\overline{X(t)}$ | $X(t)$ | $Y(t)$ | $\overline{Y(t)}$ | $\overline{X(t)}$ |
| 0 | 1 | $\overline{Y(t)}$ | 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | $\overline{X(t)}$ | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

- Für die Werte (A, B) gleich $(0, 1)$ und $(1, 0)$ ist das Schaltwerk nach $2\Delta t$ stabil
- Für den Wert (A, B) gleich $(1, 1)$ ist das Schaltwerk nach Δt stabil
- Für den Wert (A, B) gleich $(0, 0)$ scheint das Schaltwerk zu schwingen
 - Stimmt das?

Analyse des Schaltwerks (6)

Betrachtung der Ausgänge bei $A = B = 0$

| A | B | $X(t + \Delta t)$ | $Y(t + \Delta t)$ | $X(t + 2\Delta t)$ | $Y(t + 2\Delta t)$ | $X(t + 3\Delta t)$ | $Y(t + 3\Delta t)$ |
|-----|-----|-------------------|-------------------|--------------------|--------------------|--------------------|--------------------|
| 0 | 0 | $\overline{Y(t)}$ | $\overline{X(t)}$ | $X(t)$ | $Y(t)$ | $\overline{Y(t)}$ | $\overline{X(t)}$ |

– Was passiert für $X = Y = 0$?

| A | B | $X(t + \Delta t)$ | $Y(t + \Delta t)$ | $X(t + 2\Delta t)$ | $Y(t + 2\Delta t)$ | $X(t + 3\Delta t)$ | $Y(t + 3\Delta t)$ |
|-----|-----|-------------------|-------------------|--------------------|--------------------|--------------------|--------------------|
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |

– Schaltwerk schwingt (auch für $X = Y = 1$)

Analyse des Schaltwerks (7)

Betrachtung der Ausgänge bei $A = B = 0$

| A | B | $X(t + \Delta t)$ | $Y(t + \Delta t)$ | $X(t + 2\Delta t)$ | $Y(t + 2\Delta t)$ | $X(t + 3\Delta t)$ | $Y(t + 3\Delta t)$ |
|-----|-----|-------------------|-------------------|--------------------|--------------------|--------------------|--------------------|
| 0 | 0 | $\overline{Y(t)}$ | $\overline{X(t)}$ | $X(t)$ | $Y(t)$ | $\overline{Y(t)}$ | $\overline{X(t)}$ |

- Was passiert für $X = 0$ und $Y = 1$?

| A | B | $X(t + \Delta t)$ | $Y(t + \Delta t)$ | $X(t + 2\Delta t)$ | $Y(t + 2\Delta t)$ | $X(t + 3\Delta t)$ | $Y(t + 3\Delta t)$ |
|-----|-----|-------------------|-------------------|--------------------|--------------------|--------------------|--------------------|
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |

- Schaltwerk sofort stabil
- Was passiert für $X = 1$ und $Y = 0$?

| A | B | $X(t + \Delta t)$ | $Y(t + \Delta t)$ | $X(t + 2\Delta t)$ | $Y(t + 2\Delta t)$ | $X(t + 3\Delta t)$ | $Y(t + 3\Delta t)$ |
|-----|-----|-------------------|-------------------|--------------------|--------------------|--------------------|--------------------|
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |

- Schaltwerk sofort stabil

Bedeutung (1)

Verbot der Eingabekombination $A = B = 1$

- Zwei stabile Ausgabewert-Kombinationen mit $X = \bar{Y}$

Bezeichnung

- Bistabiler Speicher
- Bistabile Kippstufe
- Flip-Flop

Umbenennung der Ein- und Ausgänge

- $R = A$ – Reset, Löschen
- $S = B$ – Set, Setzen
- $Q = X$ – Ausgang
- $\bar{Q} = Y$ – negierter Ausgang

RS-Flip-Flop

- ☞ Kann einen binären Wert speichern!

Bedeutung (2)

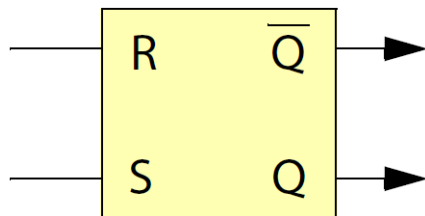
Verkürzte Wahrheitstabelle eines RS-Flip-Flops

| R | S | Q' |
|-----|-----|------|
| 0 | 0 | Q |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | – |

Q bleibt unverändert

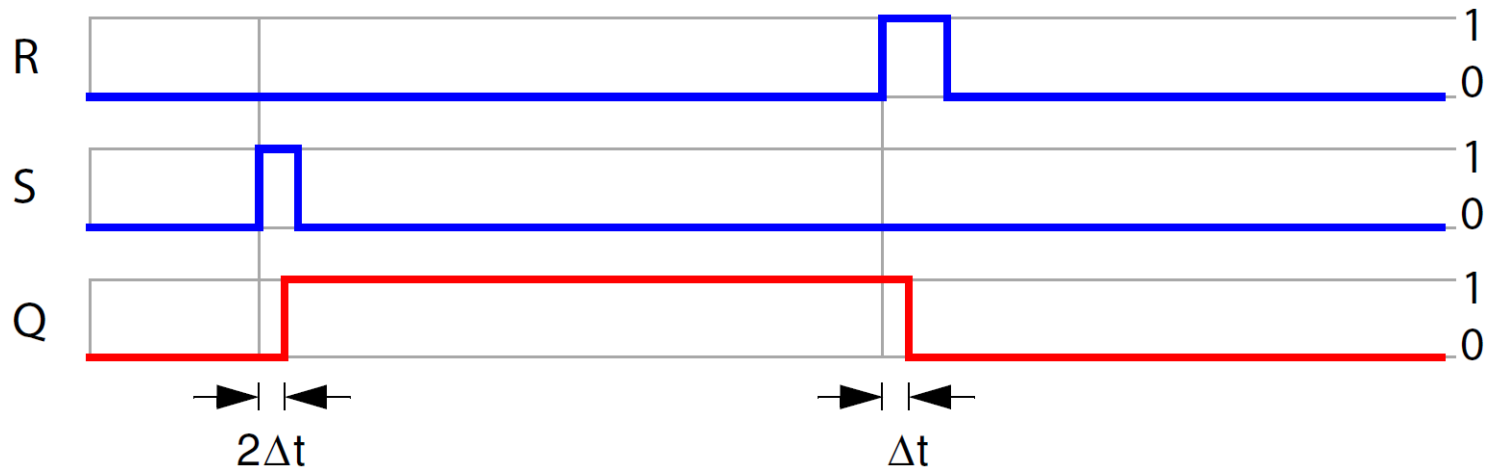
Eingabekombination nicht erlaubt

Blockschaltbild



Signalverläufe

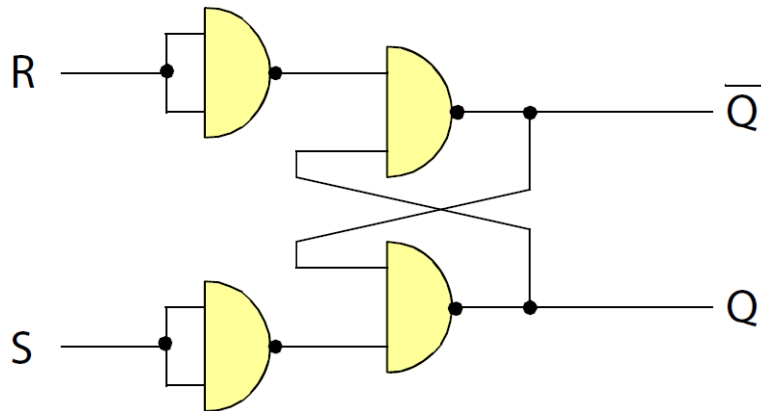
Reaktion des RS-Flip-Flops über die Zeit



- Steigende Signalfranke triggert das Umschalten des Flip-Flops
- Verzögerung und Einschwingzeit abhängig von der Gatterlaufzeit
 - Nach max. $2\Delta t$ ist Flip-Flop eingeschwungen

Realisierung mit NAND-Gattern

Einsatz von NAND- statt NOR-Gattern



- Achtung: wegen De Morgan sind Ausgänge invertiert

Asynchrone und synchrone Schaltwerke

Asynchrone Schaltwerke

- Veränderte Eingänge sorgen unmittelbar für veränderte Ergebnisse
- Gatterlaufzeit bestimmt Zeitdauer bis stabiles Ergebnis vorliegt
- Zuverlässiges Design schwierig, Entwurf sehr aufwändig
 - Zeit ist „Echtzeit“
- Sehr schnelle Schaltungen möglich

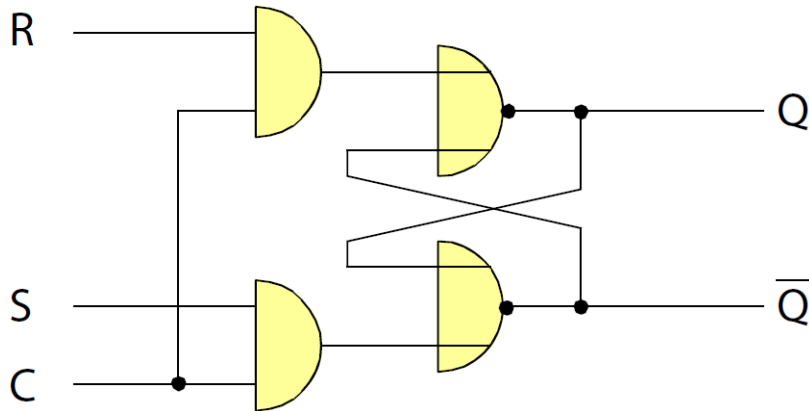
Synchrone Schaltwerke

- Zentraler Takt
- Übernahme eines Eingangs nur zu festen Zeitpunkten
 - Signal hat Zeit, stabil zu werden
- Einfacher und systematischer Entwurf
 - Zeit ist „Taktzeit“
- Langsamste Teilschaltung bestimmt maximale Taktfrequenz

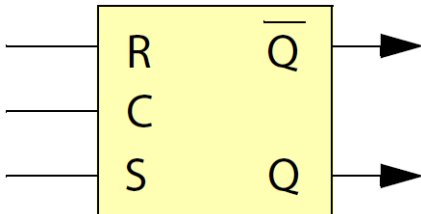
Getaktetes RS-Flip-Flop (1)

Synchrone Schaltung

- Übernahme der Eingänge nur während einer Phase des Taktsignals C (*Clock*)

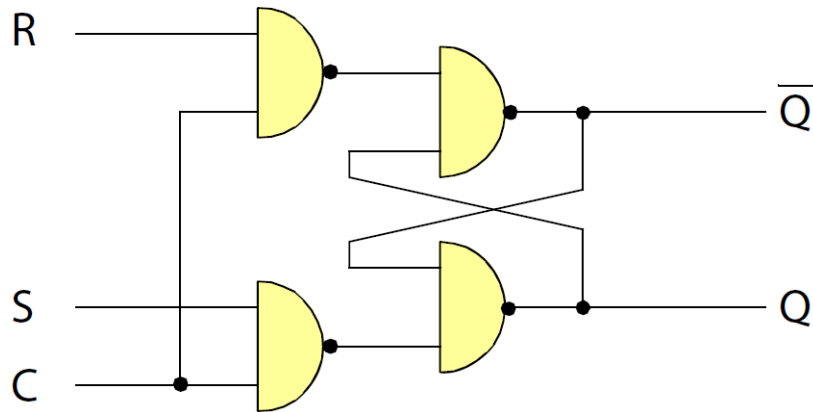


- Blockschaltbild



Getaktetes RS-Flip-Flop (2)

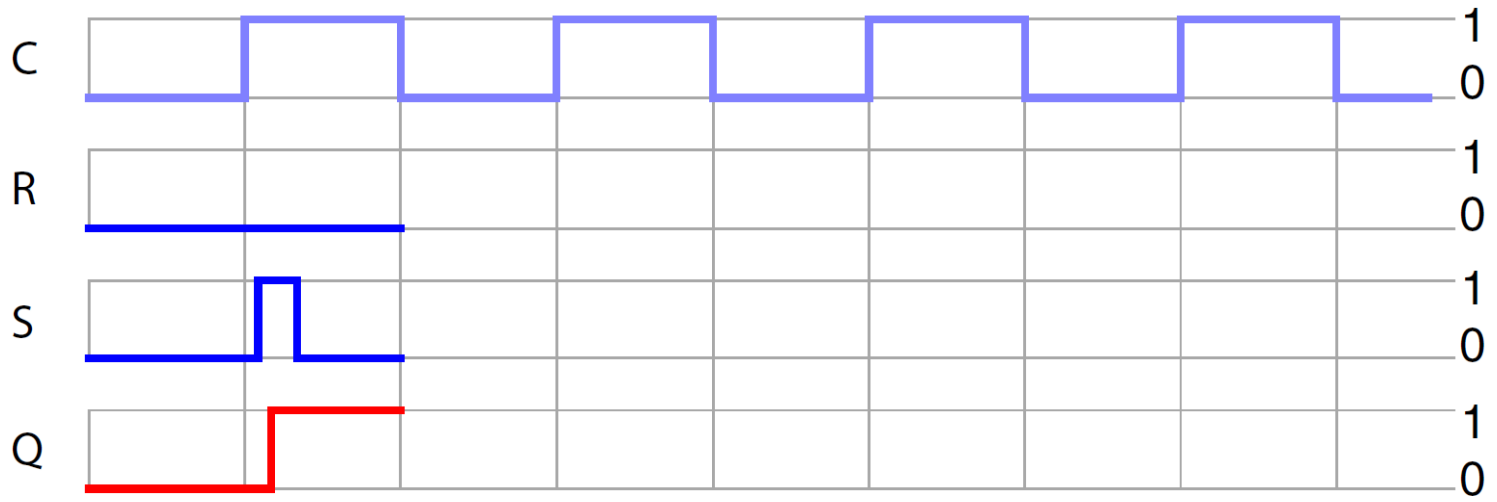
Realisierung mit NAND-Gattern



- Achtung: wegen De Morgan sind Ausgänge invertiert

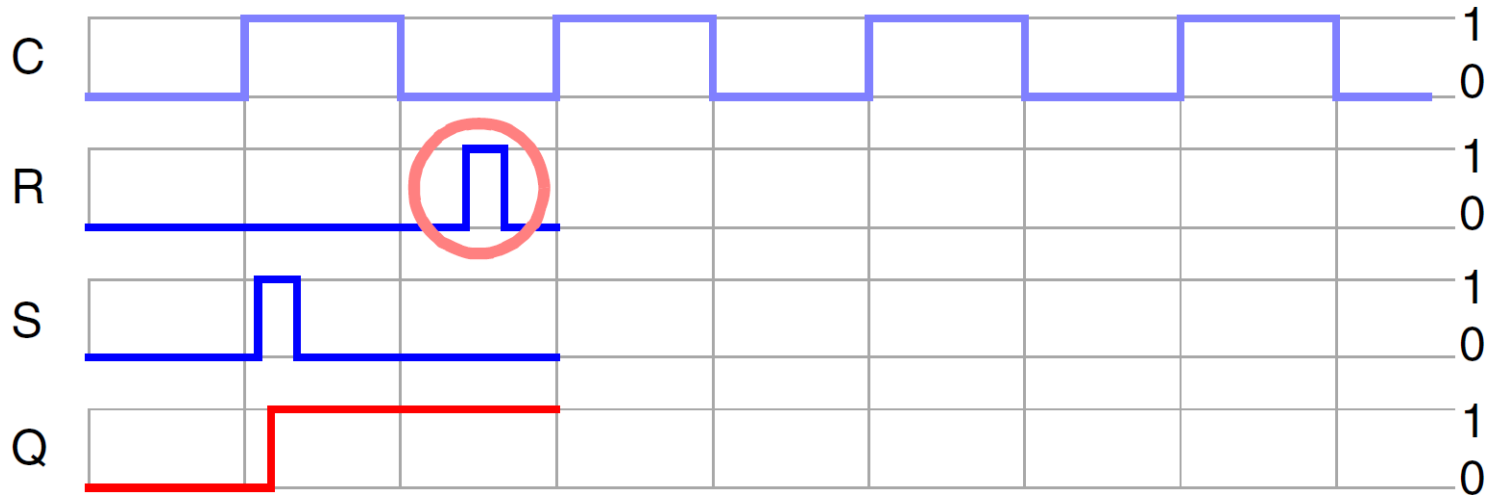
Getaktetes RS-Flip-Flop (3)

Zeitverhalten des getakteten RS-Flip-Flops



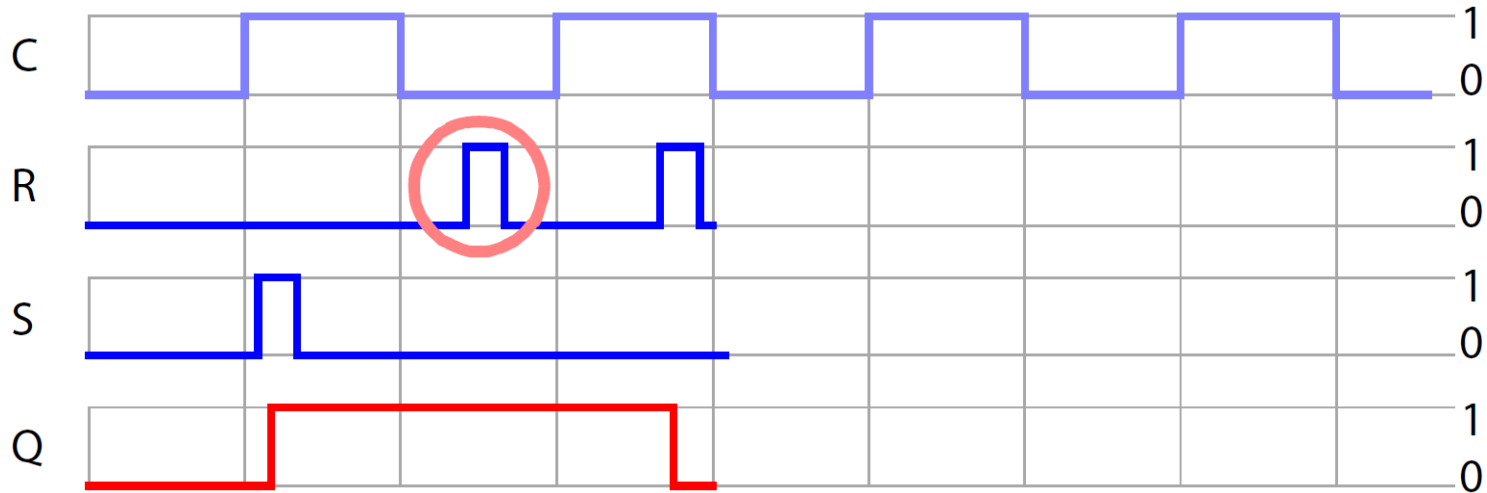
Getaktetes RS-Flip-Flop (4)

Zeitverhalten des getakteten RS-Flip-Flops



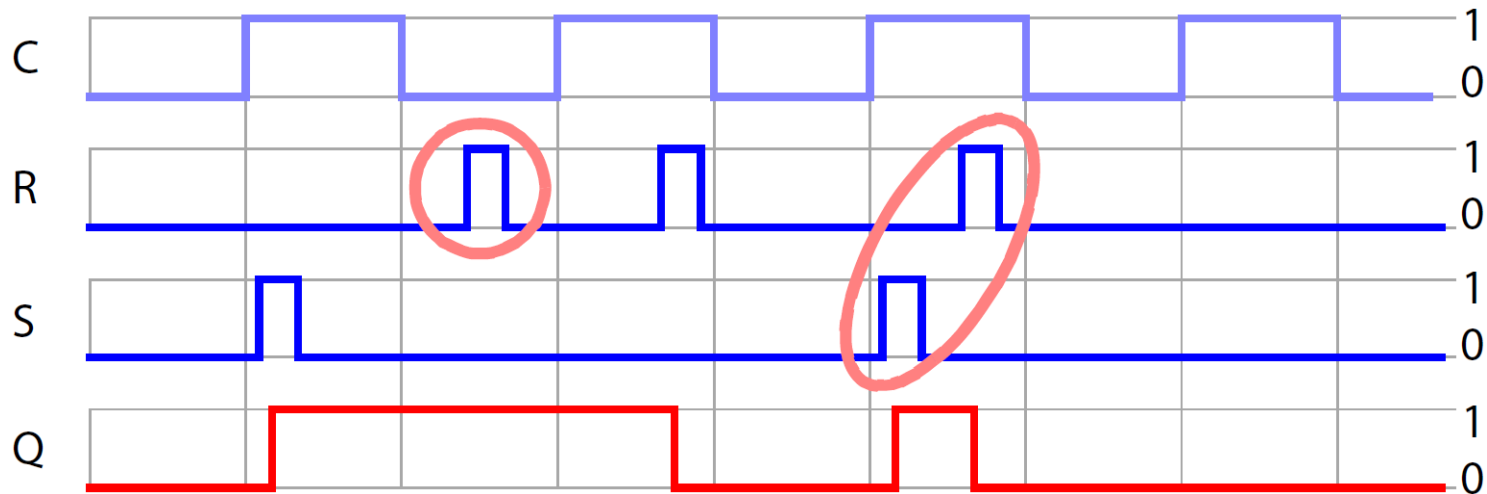
Getaktetes RS-Flip-Flop (5)

Zeitverhalten des getakteten RS-Flip-Flops



Getaktetes RS-Flip-Flop (6)

Zeitverhalten des getakteten RS-Flip-Flops

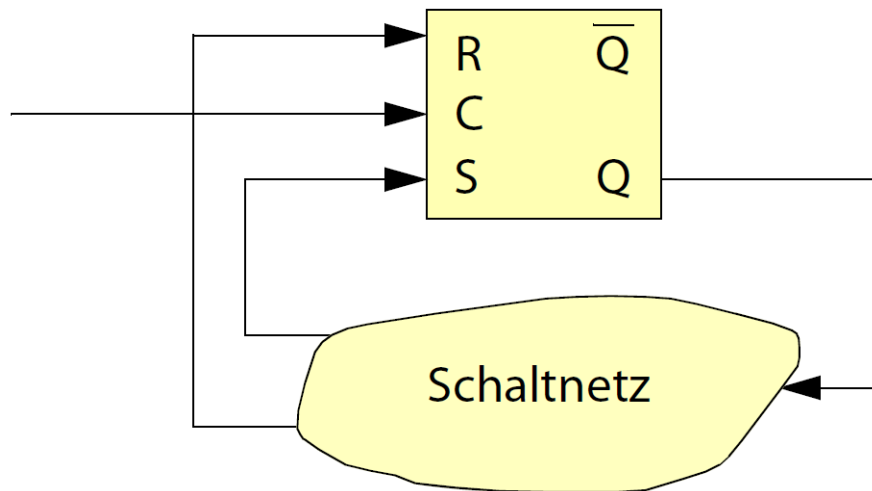


- Veränderungen finden nur während der 1-Phase des Taktsignals statt
- Mehrere Veränderungen pro Taktphase möglich

Getaktetes RS-Flip-Flop (6)

Problem

- Rückgekoppelte Schaltung vom Aus- zum Eingang des Flip-Flops

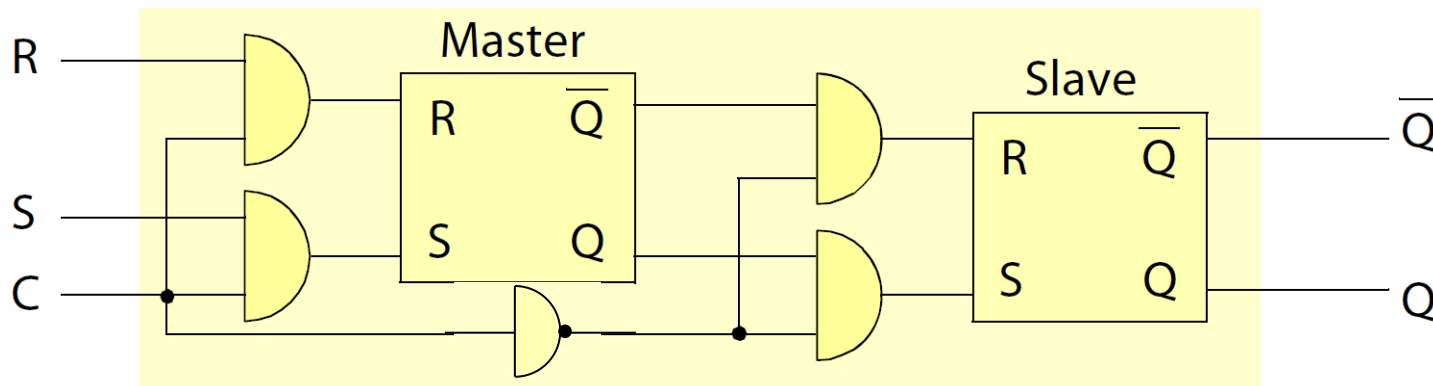


- Veränderungen am Ausgang können Veränderungen am Eingang während einer Taktphase nach sich ziehen
- Selbst bei kurzen Taktphasen sind ungewollte Rückkopplungen möglich

Master-Slave RS-Flip-Flop (1)

RS-Flip-Flop reagiert auf (absteigende) Taktflanke

- Zweistufiges Flip-Flop



Takt auf 1

- *Master* Flip-Flop nimmt Eingänge auf, *Slave* bleibt unverändert

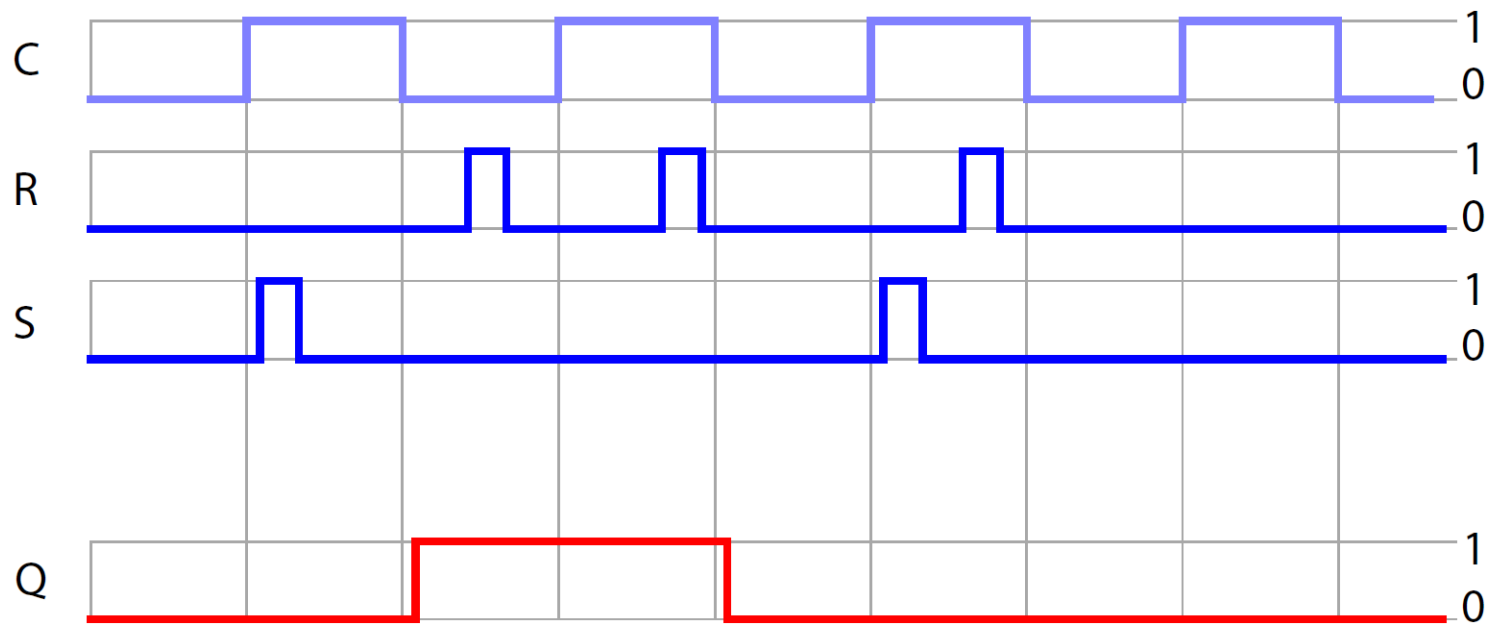
Takt auf 0

- *Master* Flip-Flop reagiert nicht mehr auf Eingänge, *Slave* übernimmt Zustand des *Masters*

Master-Slave RS-Flip-Flop (2)

Flip-Flop reagiert auf absteigende Taktflanke

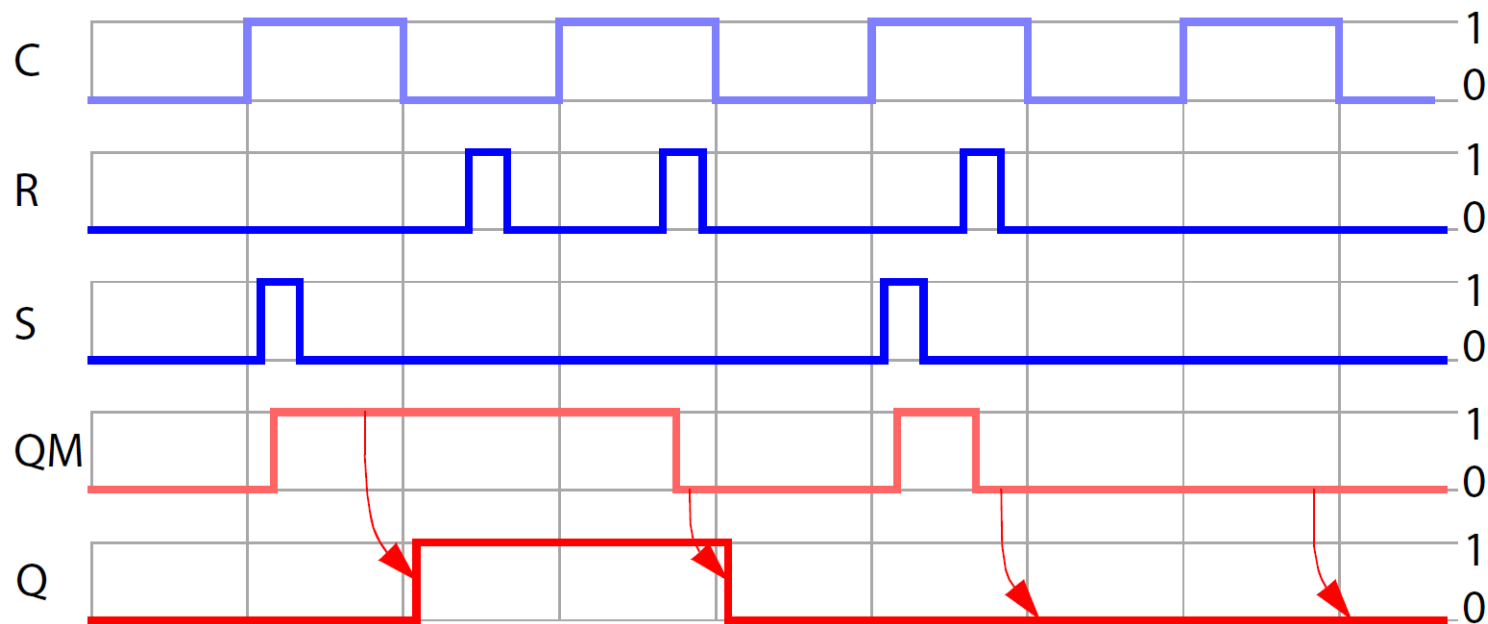
– Zeitverhalten



Master-Slave RS-Flip-Flop (3)

Flip-Flop reagiert auf absteigende Taktflanke

- Zeitverhalten

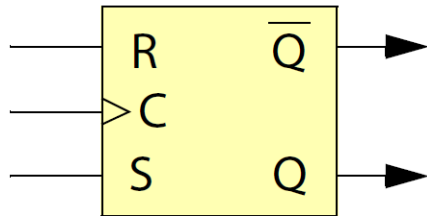


- Q_M : Q-Ausgang des *Master* Flip-Flops
- Rückkopplung über Schaltnetz nun unkritisch

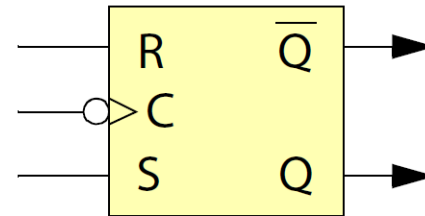
Master-Slave RS-Flip-Flop (4)

Flankengetriggerte Flip-Flops

- Nach außen sichtbares Verhalten eines *Master-Slave* RS-Flip-Flops
 - Übernahme der Eingänge nur bei Taktflanke
- Blockschaltbild flankengetriggelter RS-Flip-Flops



positive Flankentriggerung

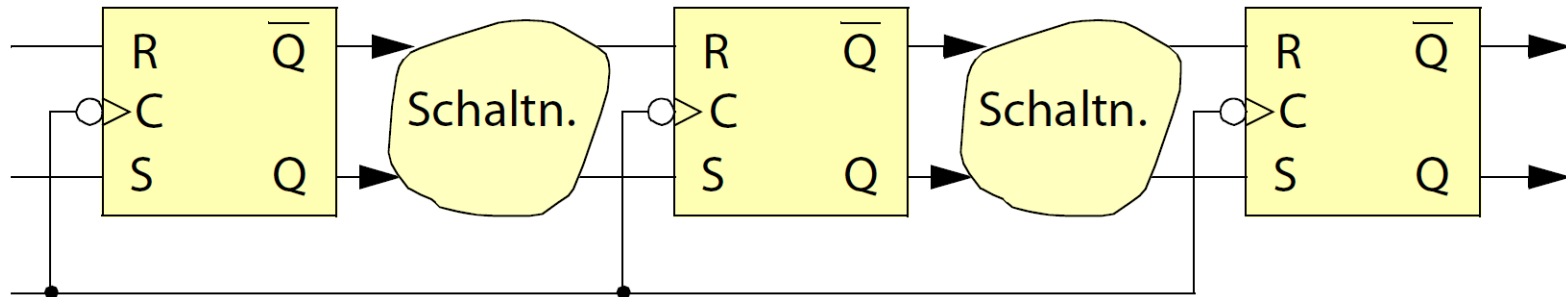


negative Flankentriggerung

- Negative Flanke = absteigende Flanke
 - Beispiel von vorherigen Folien
- Positive Flanke = aufsteigende Flanke
 - Takteingang wird negiert

Pipelining (1)

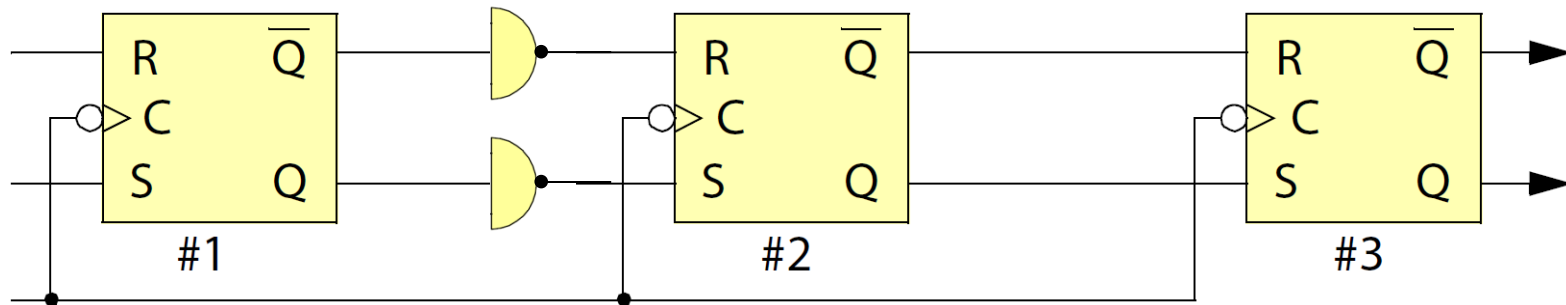
Mit *Master-Slave* Flip-Flops sind folgende Schaltwerke denkbar



- Pro Taktzyklus
 - Logische Verarbeitung in den zwischengeschalteten Schaltnetzen
 - Weitergabe der Information an nächste Stufe der *Pipeline*
 - Ausgabe am Ende der *Pipeline*
- Nach drei Takten kommt die verarbeitete Information am Ende heraus

Pipelining (2)

Einfaches Beispiel

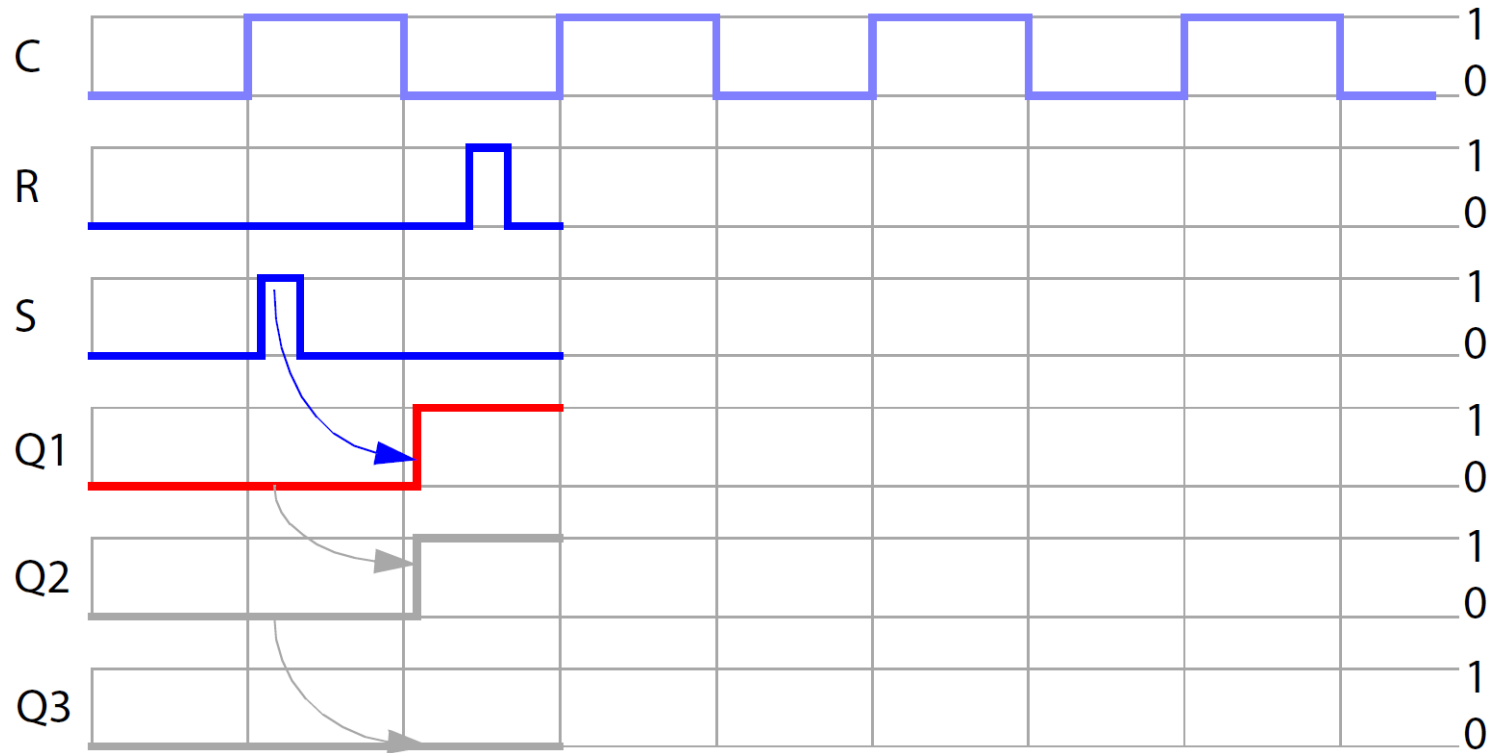


Annahme

- Alle Flip-Flops geben zu Beginn $Q = 0$ aus

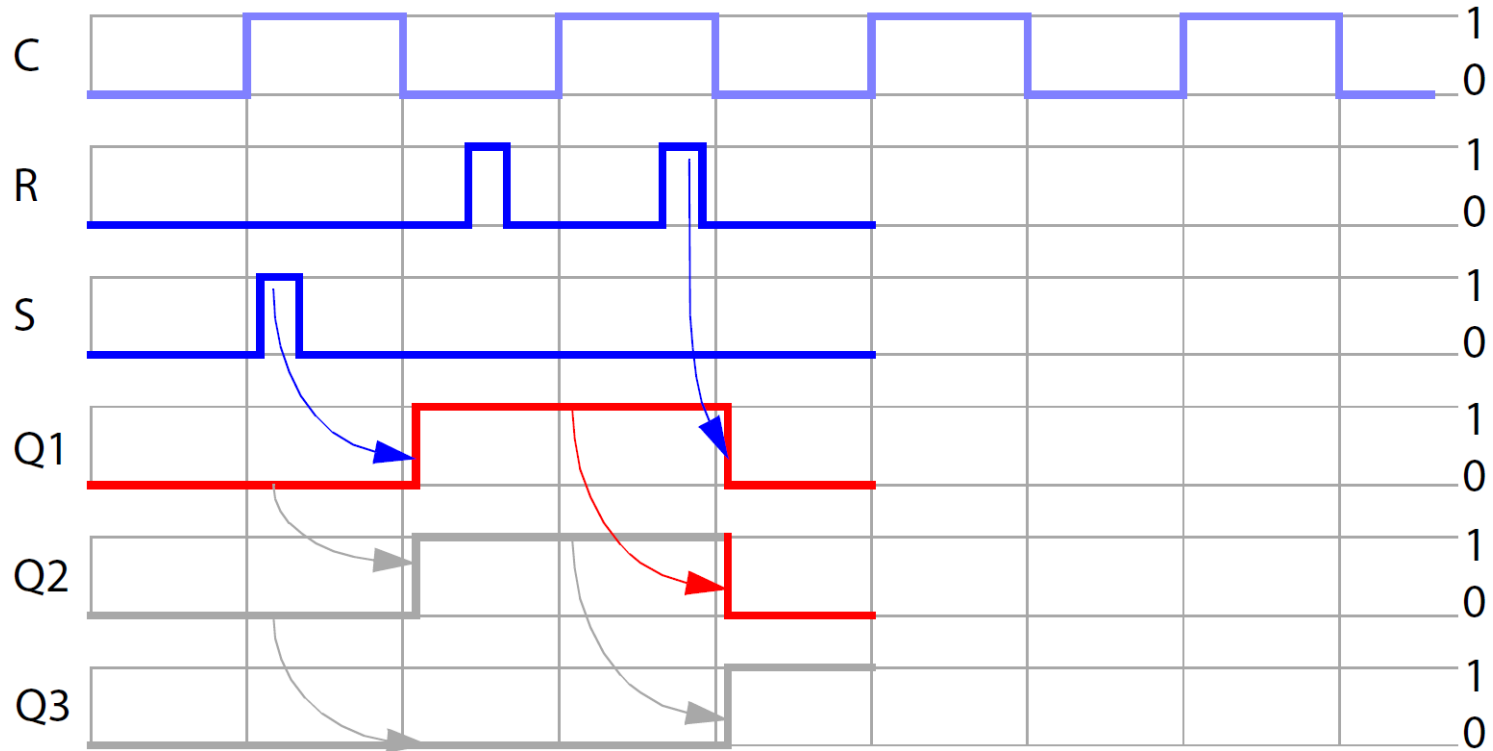
Pipelining (3)

Zeitverhalten



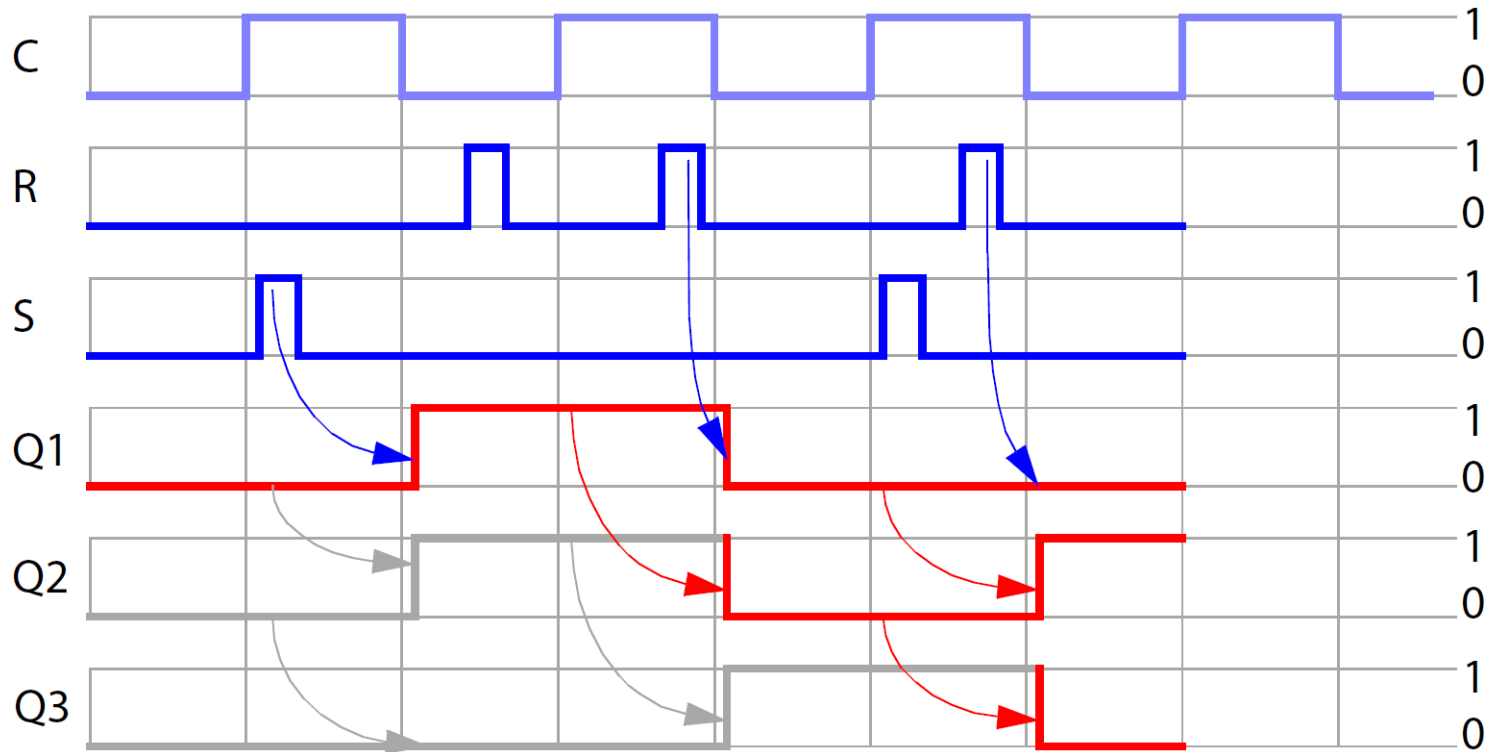
Pipelining (4)

Zeitverhalten



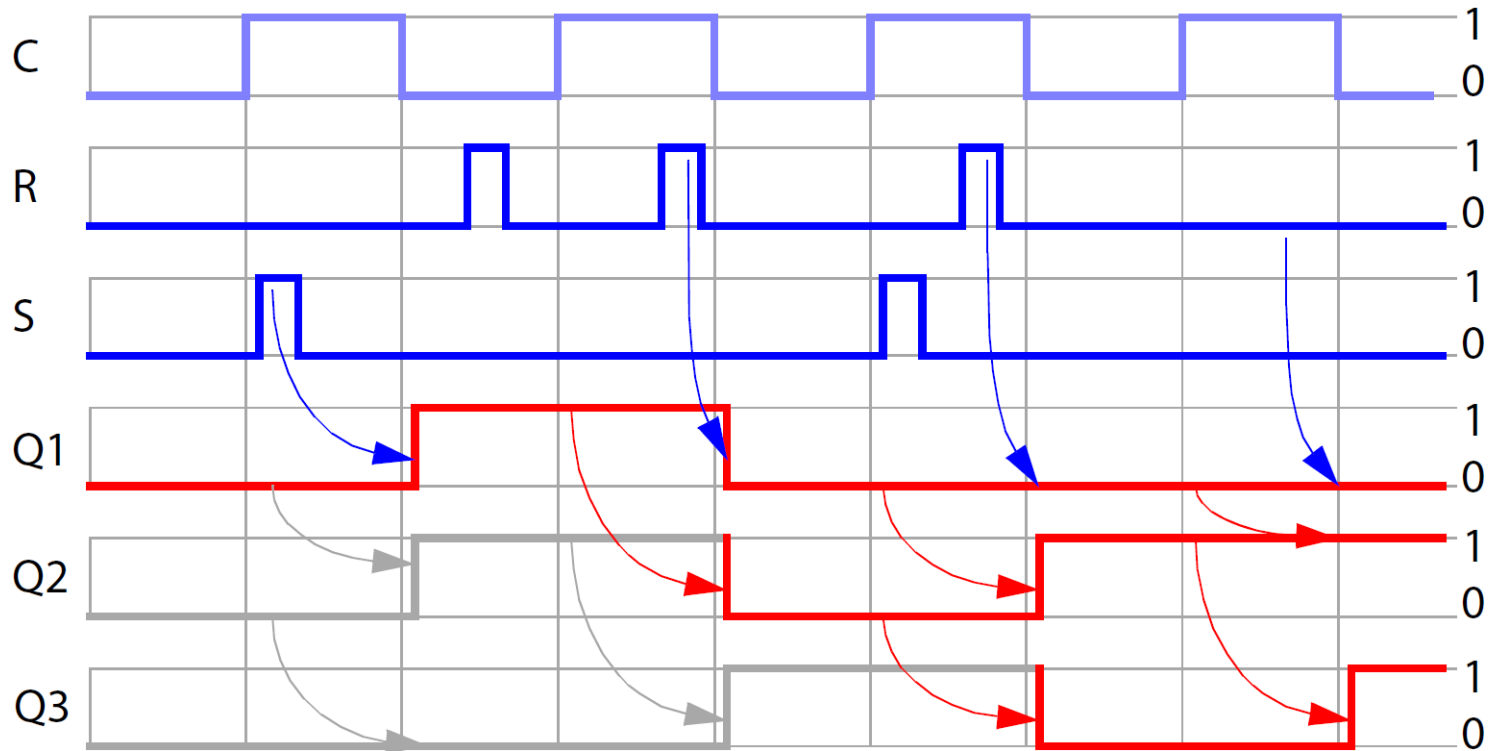
Pipelining (5)

Zeitverhalten



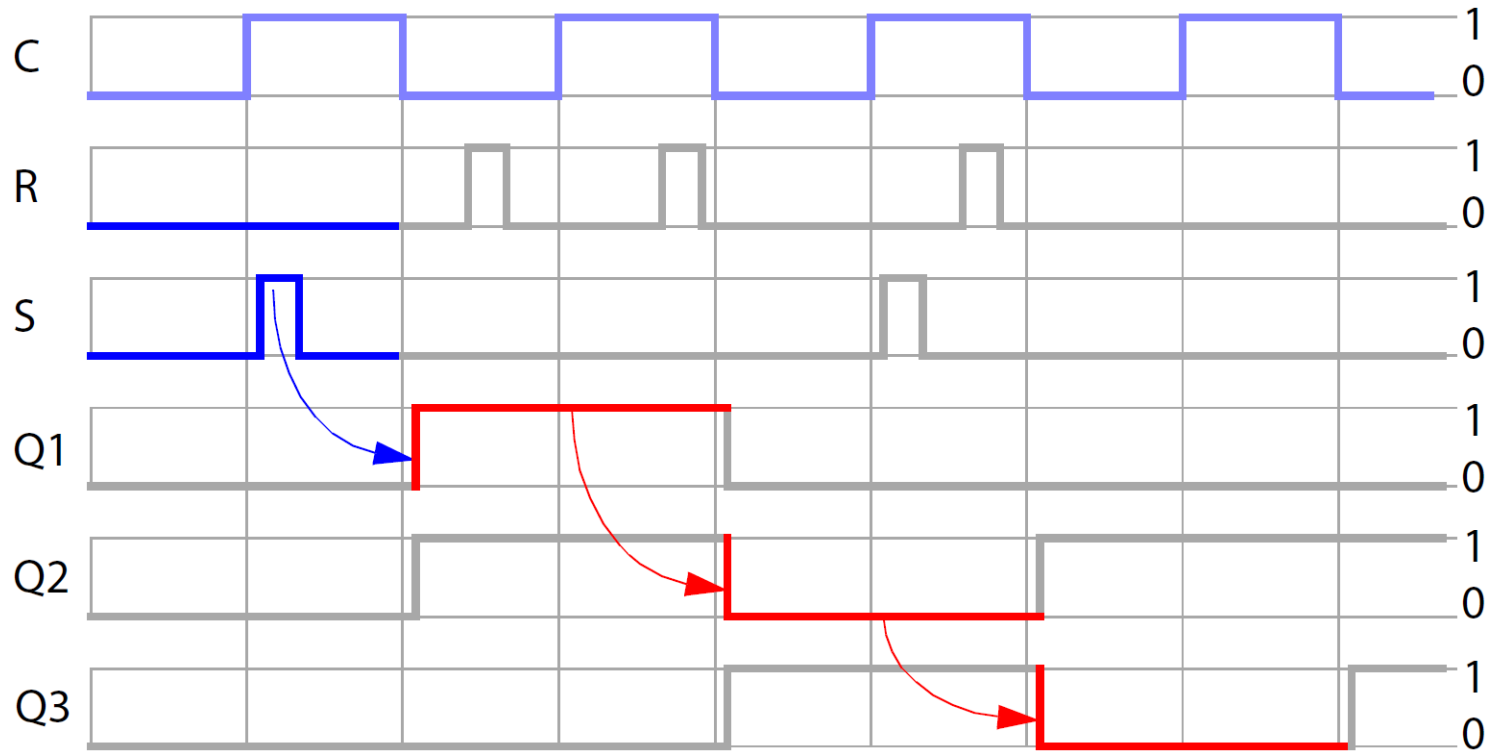
Pipelining (6)

Zeitverhalten



Pipelining (7)

Zeitverhalten



D-Flip-Flop (1)

Synchroner Baustein

- Verkürzte Wahrheitstabelle eines D-Flip-Flops

| D | C | Q' |
|-----|-----|------|
| 0 | 0 | Q |
| 0 | 1 | 0 |
| 1 | 0 | Q |
| 1 | 1 | 1 |

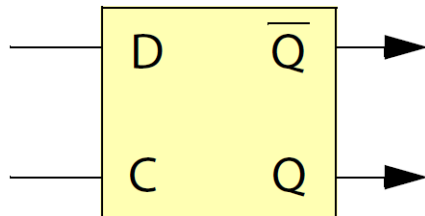
Q bleibt unverändert

0 wird übernommen

Q bleibt unverändert

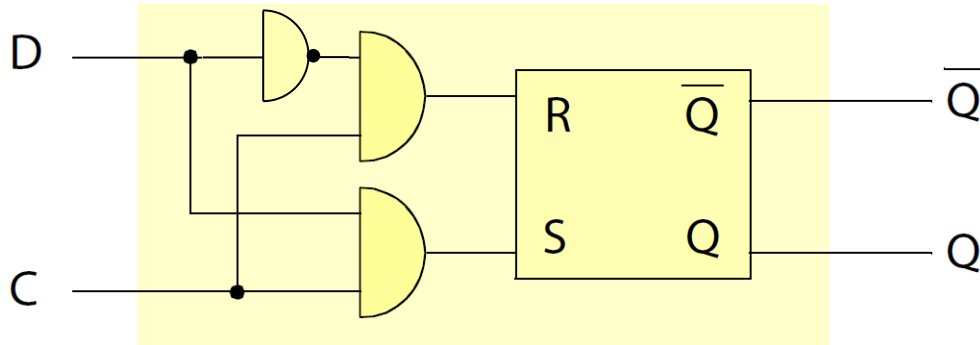
1 wird übernommen

Blockschaltbild



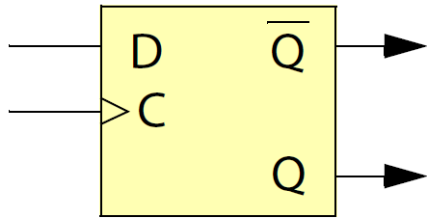
D-Flip-Flop (2)

Interner Aufbau des pegelgetriggerten D-Flip-Flops

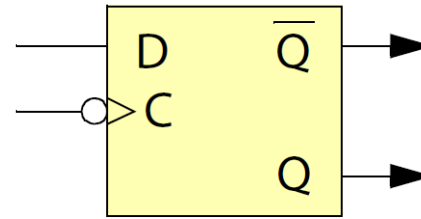


D-Flip-Flop (3)

Blockschaltbilder flankengetriggelter D-Flip-Flops

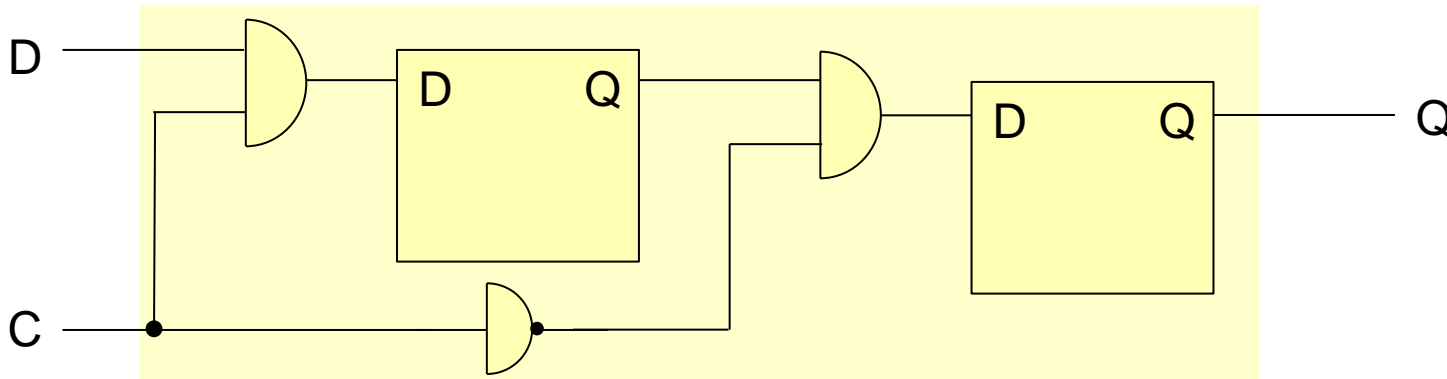


positive Flankentriggerung



negative Flankentriggerung

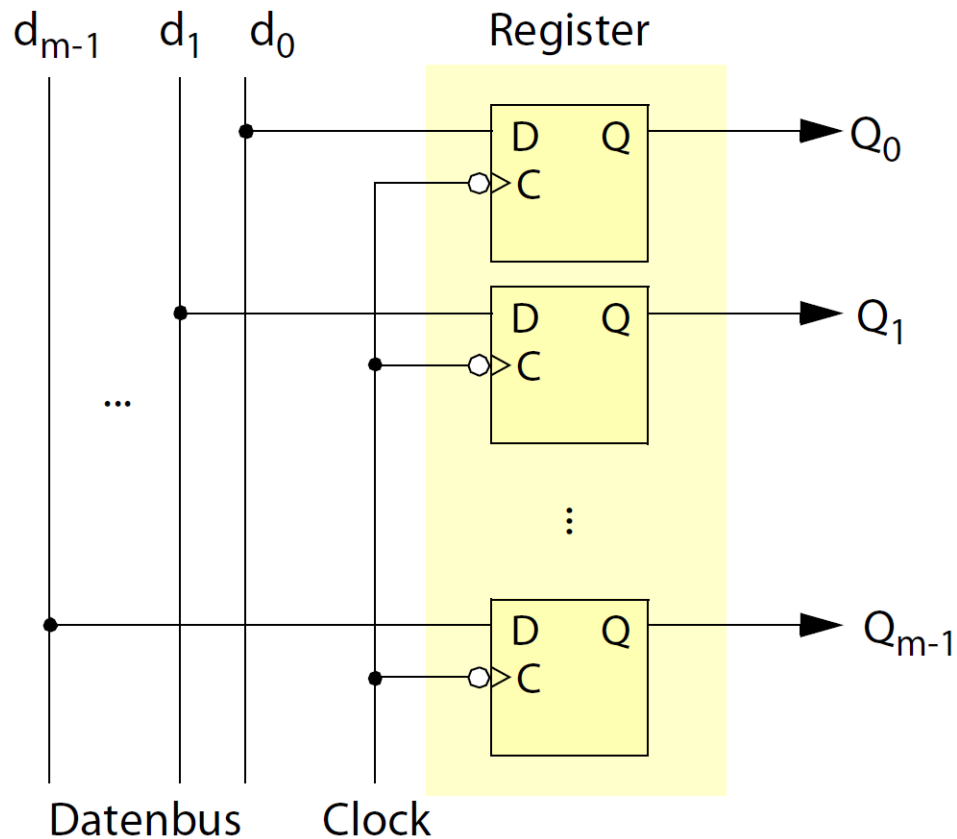
Interner Aufbau flankengetriggelter D-Flip-Flops



Register

Flip-Flops bilden ein Register

- Speicher für eine bestimmte Anzahl von Binärwerten



JK-Flip-Flop (1)

RS-Flip-Flop mit zusätzlicher Umschaltfunktion (*toggle*)

- In der Regel synchron und mit *Master-Slave*-Aufbau
- Verkürzte Wahrheitstabelle eines JK-Flip-Flops

| C | J | K | Q' |
|-----|-----|-----|----------------|
| 0 | * | * | Q |
| 1 | 0 | 0 | Q |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | \overline{Q} |

Keine Änderung ohne Takt

Rücksetzen ($K = R$)

Setzen ($J = S$)

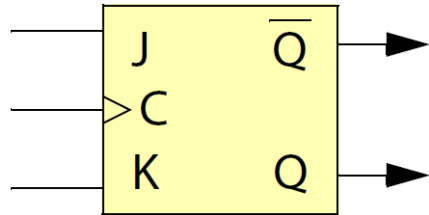
Q wird invertiert (*toggle*)

Spezialfall

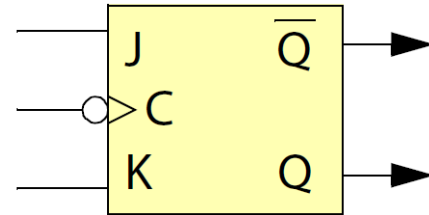
- T-Flip-Flop (*toggle*) mit einem Eingang $T = K = J$ plus Takteingang

JK-Flip-Flop (2)

Blockschaltbild flankengetriggelter JK-Flip-Flops

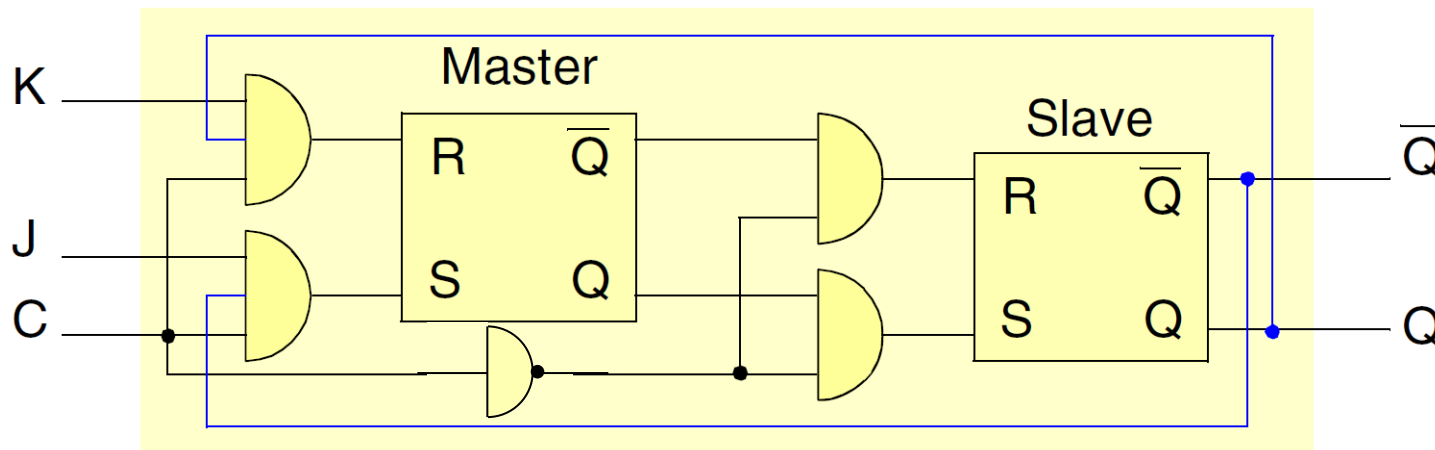


positive Flankentriggerung



negative Flankentriggerung

Realisierung eines JK-Flip-Flops mit *Master-Slave*-Aufbau



Roter Faden

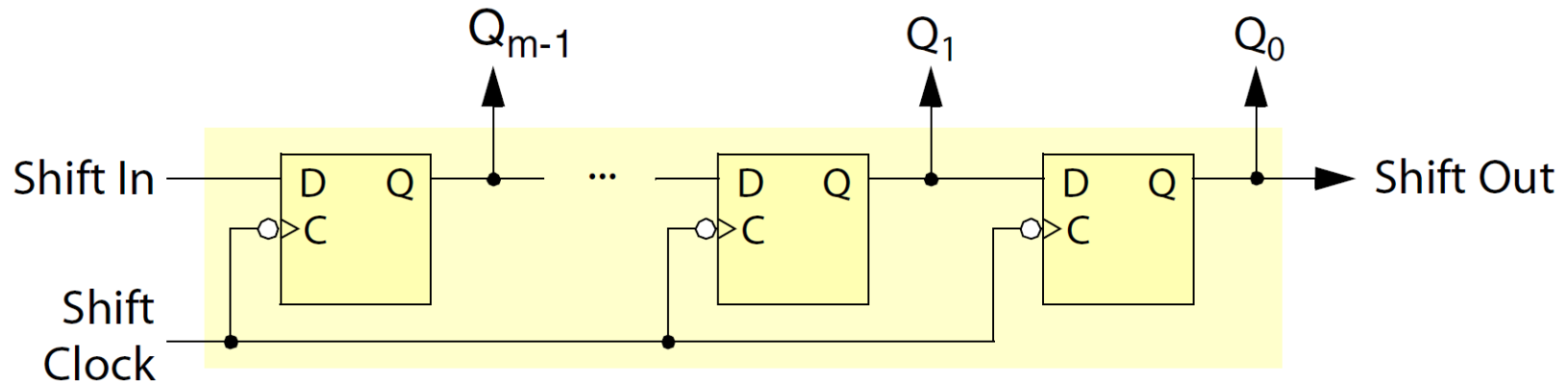
3. Sequentielle Logik

- Einleitung
- Flip-Flops
 - RS-Flip-Flop
 - Analyse
 - Bedeutung
 - Zeitverhalten
 - Asynchrone und synchrone Schaltwerke
 - Getaktete Flip-Flops
 - *Master-Slave* Flip-Flops
 - Weitere Flip-Flops (D-Flip-Flop, Register, JK- & T-Flip-Flop)
- Typische Schaltwerke
- Systematischer Schaltwerkentwurf

Typische Schaltwerke: Schieberegister (1)

Register mit Schiebeoperation

- Binärwerte werden z.B. im Register nach rechts geschoben
- Realisierung mit D-Flip-Flops

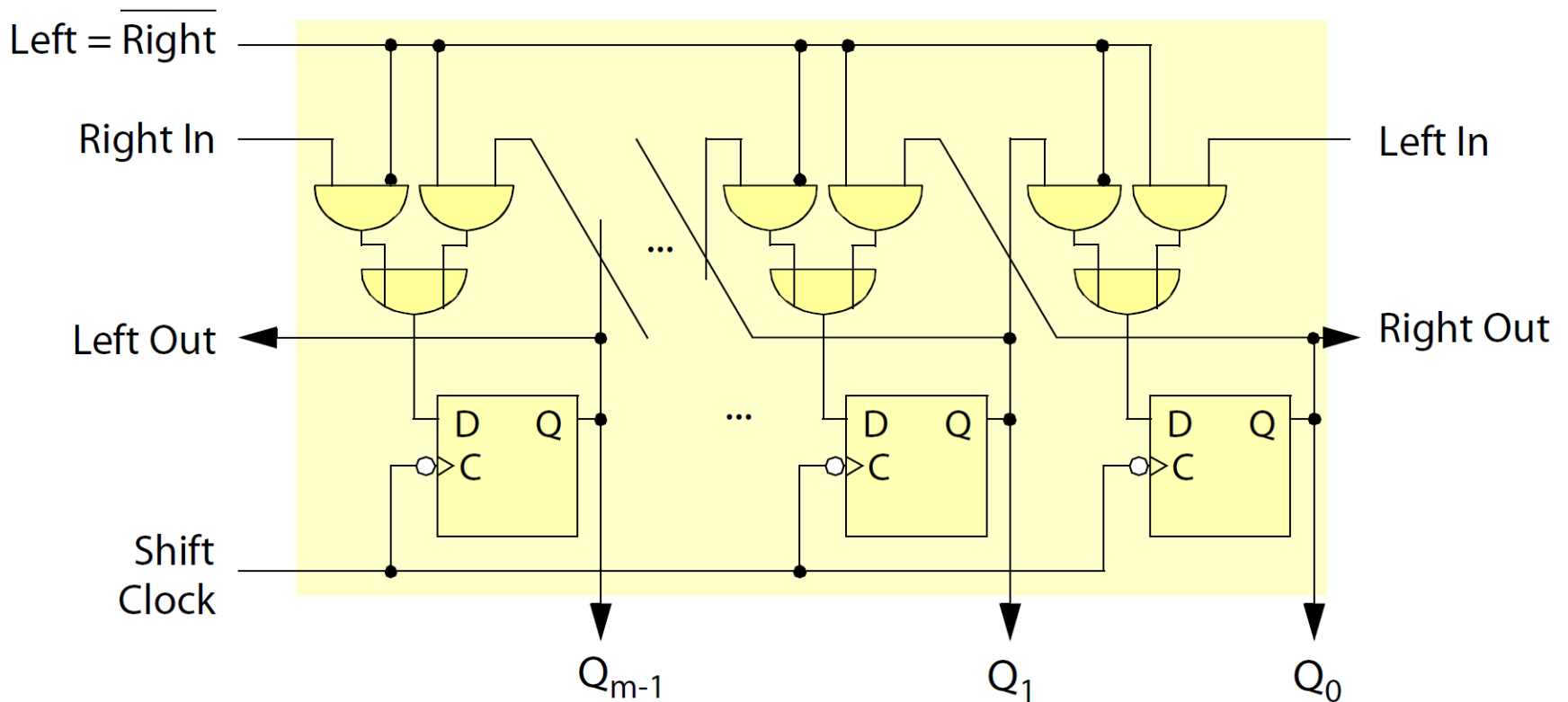


Vielfältige Anwendung

- Teil arithmetischer Operationen
- Serialisierung und Deserialisierung von Daten

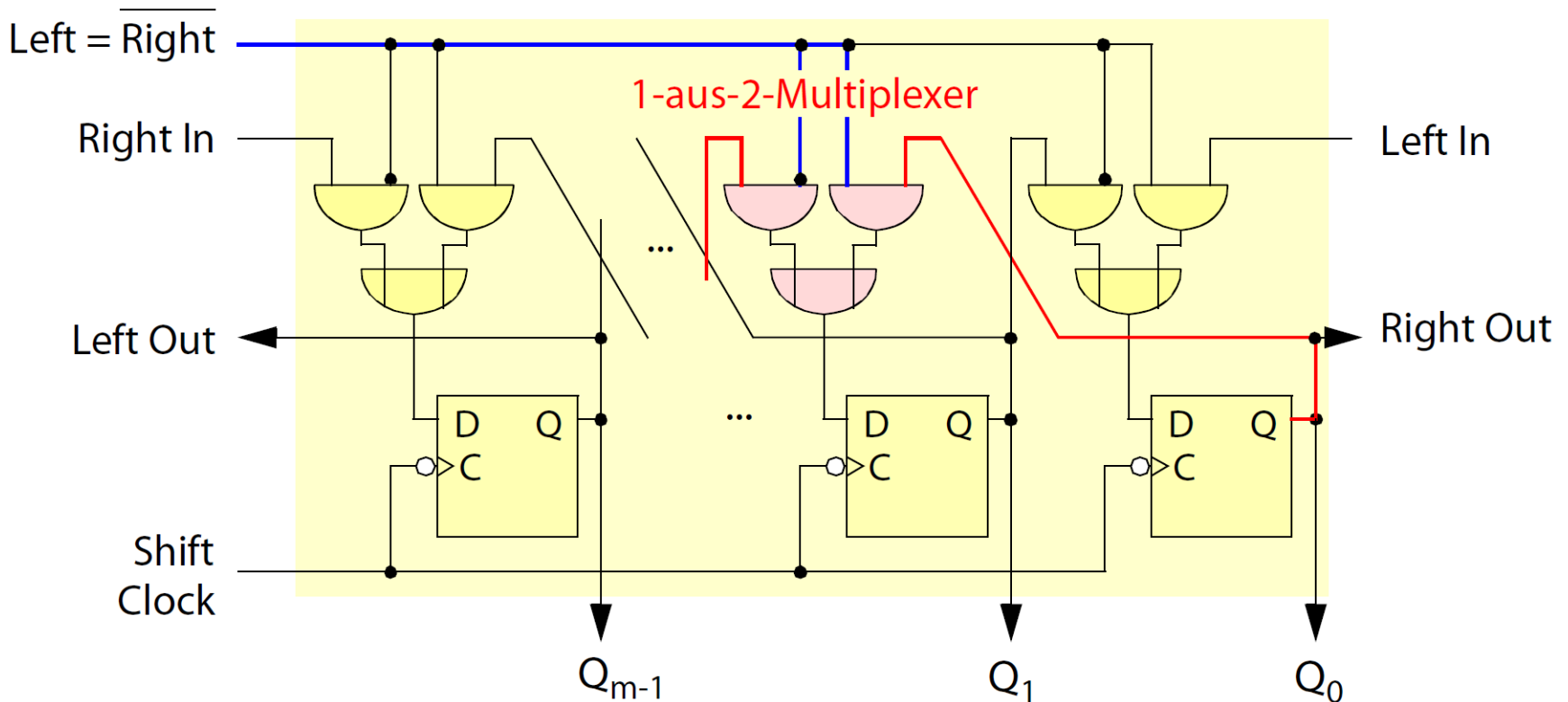
Typische Schaltwerke: Schieberegister (2)

Schieberegister für Links- und Rechtsschiebe-Operationen



Typische Schaltwerke: Schieberegister (3)

Schieberegister für Links- und Rechtsschiebe-Operationen

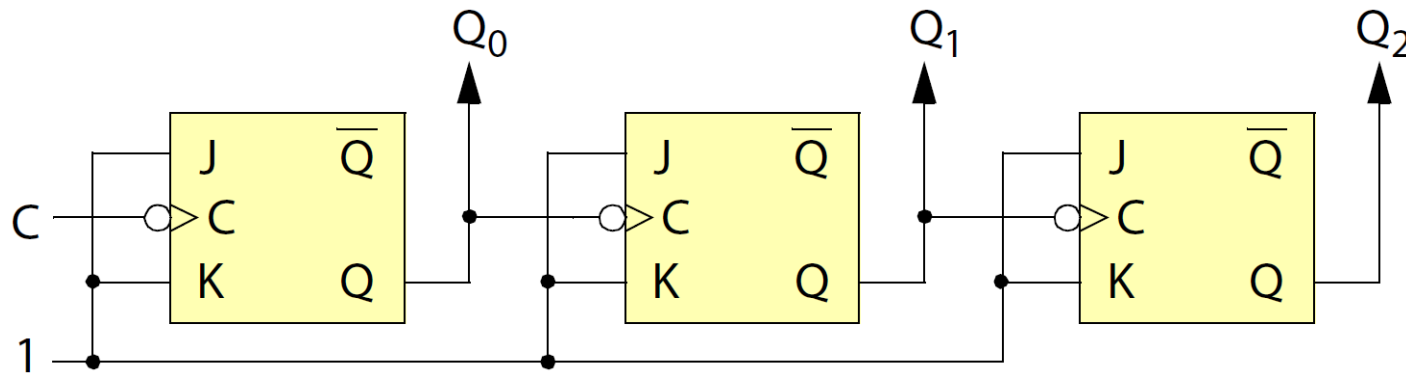


- 1-aus-2-Multiplexer: Auswahl des linken oder rechten Eingabewertes

Typische Schaltwerke: Asynchroner Zähler (1)

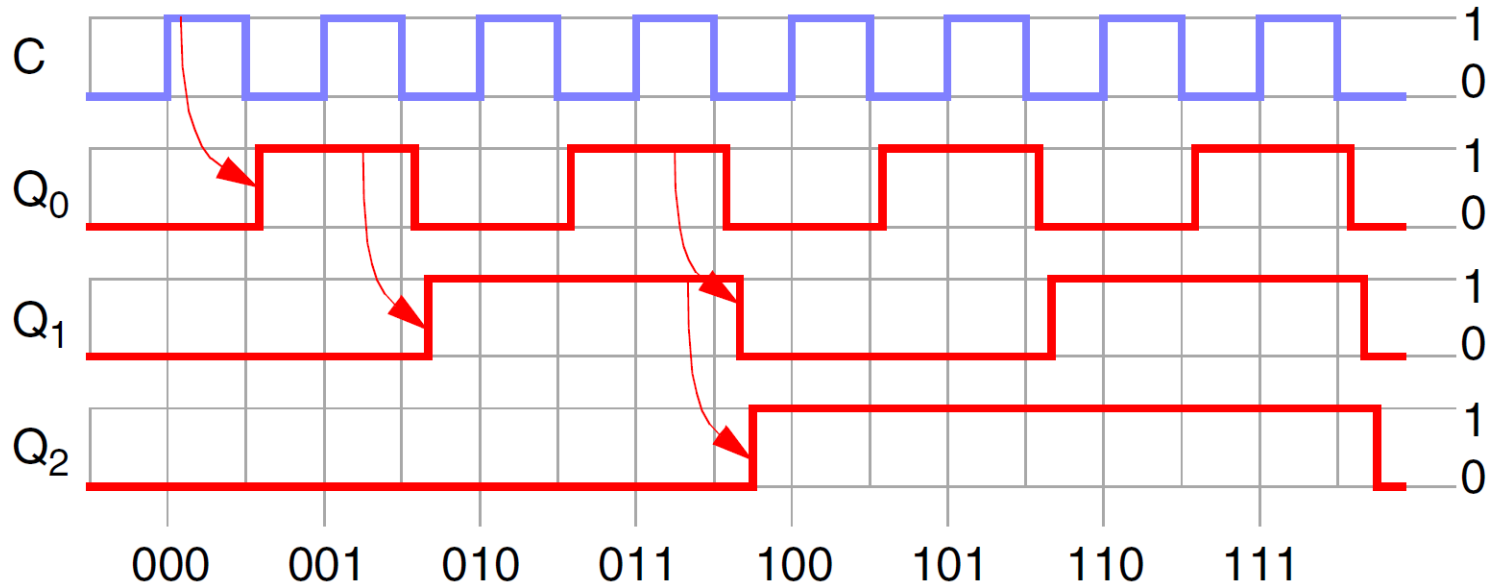
Beispiel: Dreistelliger Binärzähler zählt absteigende Taktflanken

– Aufbau mit JK-Flip-Flops



Typische Schaltwerke: Asynchroner Zähler (2)

Zeitverhalten

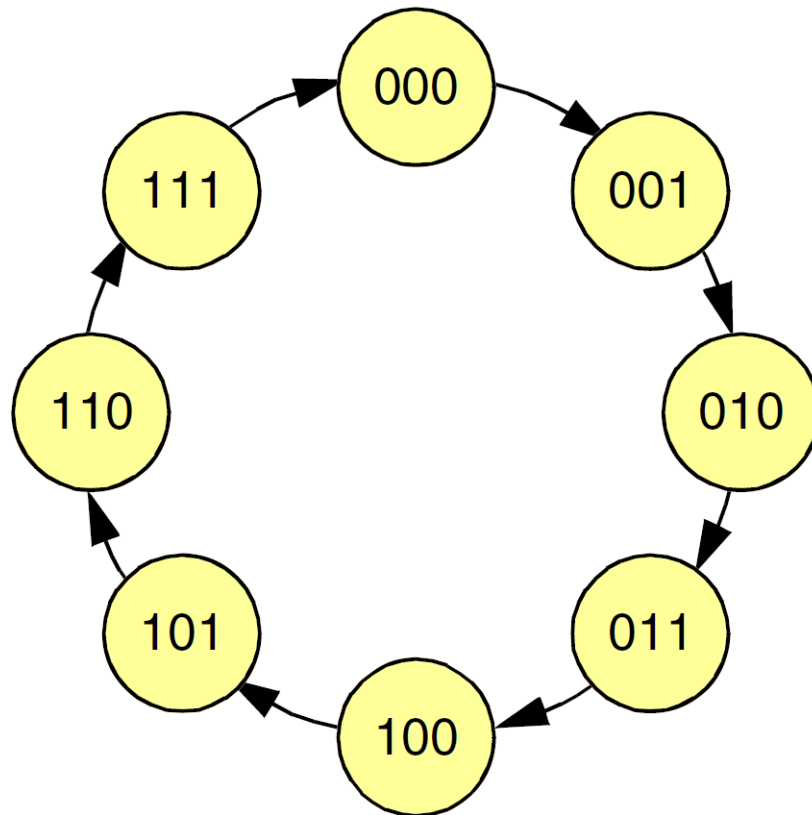


- Dominoeffekt verzögert stabilen Zustand des Zählers
- Lange Zähler sind nicht „beliebig schnell“ taktbar

Typische Schaltwerke: Synchroner Zähler (1)

Unmittelbarer Übergang aller beteiligten Flip-Flops pro Taktzyklus

Mögliche Zustände eines dreistelligen Binärzählers



- Übergänge pro Takt
- Unbedingte Übergänge

Typische Schaltwerke: Synchroner Zähler (2)

Einsatz von JK-Flip-Flops

- Aufstellen einer Zustandsübergangstabelle für das JK-Flip-Flop

| Übergang $Q \rightarrow Q'$ | J | K |
|--------------------------------|-----|-----|
| $0 \rightarrow 0$ | 0 | * |
| $0 \rightarrow 1$ | 1 | * |
| $1 \rightarrow 0$ | * | 1 |
| $1 \rightarrow 1$ | * | 0 |

Eingang K irrelevant

Eingang K irrelevant

Eingang J irrelevant

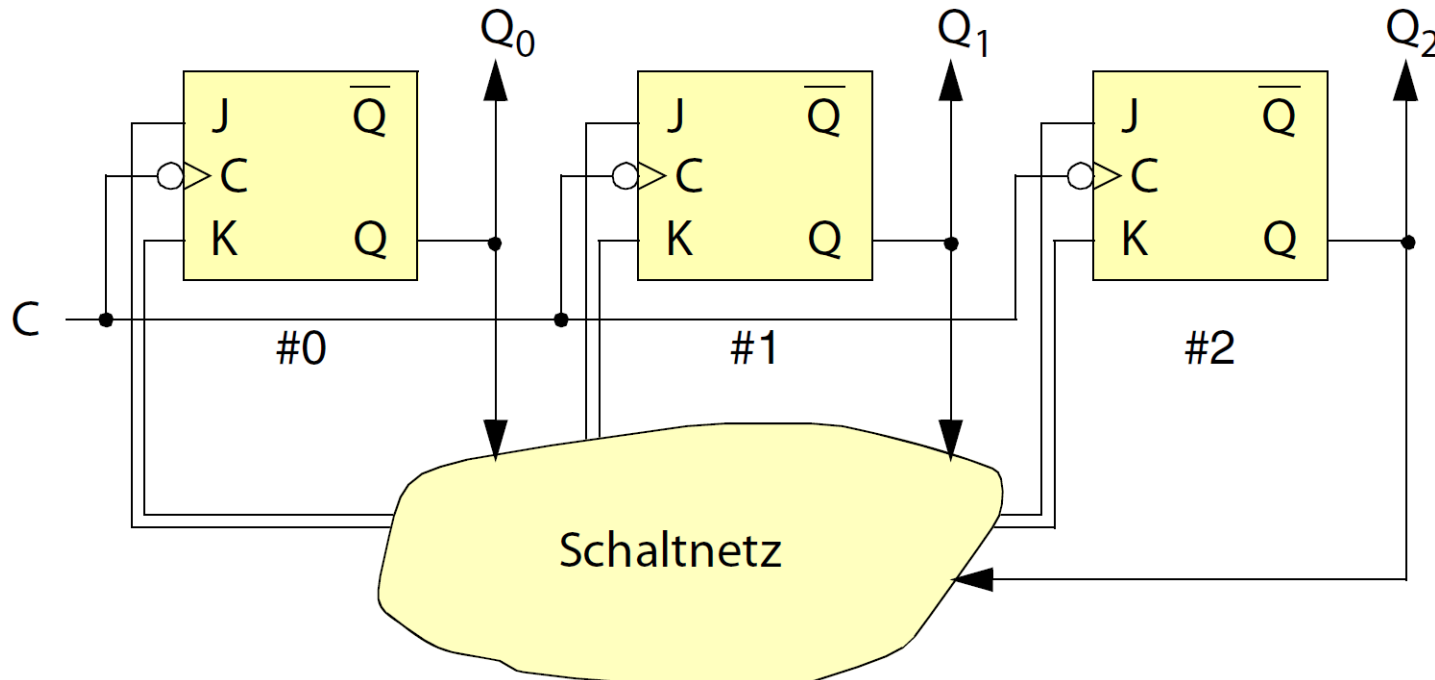
Eingang J irrelevant

- Drei JK-Flip-Flops notwendig für dreistelligen Zähler
 - Takt für alle Flip-Flops identisch
 - Wie müssen Steuereingänge J und K angesprochen werden?

Typische Schaltwerke: Synchroner Zähler (3)

Gesucht: Schaltnetze zur Ansteuerung der Flip-Flops

- Alle Flip-Flops ständig getaktet
- Ausgänge der Flip-Flops bestimmen Ansteuerung der Flip-Flops
 - Durch *Master-Slave* Flip-Flops und Takt: schrittweises Fortschalten des Schaltwerks



Typische Schaltwerke: Synchroner Zähler (4)

Zustandsübergänge des Zählers

| Zustände (gleichzeitig Ausgabe) | | | Folgezustand | | | Eingänge der Flip-Flops | | | | | |
|---------------------------------------|-------|-------|--------------|--------|--------|-------------------------|-------|-------|-------|-------|-------|
| Q_2 | Q_1 | Q_0 | Q'_2 | Q'_1 | Q'_0 | J_2 | K_2 | J_1 | K_1 | J_0 | K_0 |
| 0 | 0 | 0 | | | | | | | | | |
| 0 | 0 | 1 | | | | | | | | | |
| 0 | 1 | 0 | | | | | | | | | |
| 0 | 1 | 1 | | | | | | | | | |
| 1 | 0 | 0 | | | | | | | | | |
| 1 | 0 | 1 | | | | | | | | | |
| 1 | 1 | 0 | | | | | | | | | |
| 1 | 1 | 1 | | | | | | | | | |

Typische Schaltwerke: Synchroner Zähler (5)

Zustandsübergänge des Zählers: Folgezustände ermitteln

| Zustände (gleichzeitig Ausgabe) | | | Folgezustand | | | Eingänge der Flip-Flops | | | | | |
|---------------------------------------|-------|-------|--------------|--------|--------|-------------------------|-------|-------|-------|-------|-------|
| Q_2 | Q_1 | Q_0 | Q'_2 | Q'_1 | Q'_0 | J_2 | K_2 | J_1 | K_1 | J_0 | K_0 |
| 0 | 0 | 0 | 0 | 0 | 1 | | | | | | |
| 0 | 0 | 1 | 0 | 1 | 0 | | | | | | |
| 0 | 1 | 0 | 0 | 1 | 1 | | | | | | |
| 0 | 1 | 1 | 1 | 0 | 0 | | | | | | |
| 1 | 0 | 0 | 1 | 0 | 1 | | | | | | |
| 1 | 0 | 1 | 1 | 1 | 0 | | | | | | |
| 1 | 1 | 0 | 1 | 1 | 1 | | | | | | |
| 1 | 1 | 1 | 0 | 0 | 0 | | | | | | |

Typische Schaltwerke: Synchroner Zähler (6)

Zustandsübergänge des Zählers: Übergänge pro Flip-Flop ermitteln

| Zustände (gleichzeitig Ausgabe) | | | Folgezustand | | | Eingänge der Flip-Flops | | | | | |
|---------------------------------------|-------|-------|--------------|--------|--------|-------------------------|-------|-------|-------|-------|-------|
| Q_2 | Q_1 | Q_0 | Q'_2 | Q'_1 | Q'_0 | J_2 | K_2 | J_1 | K_1 | J_0 | K_0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | * | 0 | * | 1 | * |
| 0 | 0 | 1 | 0 | 1 | 0 | | | | | | |
| 0 | 1 | 0 | 0 | 1 | 1 | | | | | | |
| 0 | 1 | 1 | 1 | 0 | 0 | | | | | | |
| 1 | 0 | 0 | 1 | 0 | 1 | | | | | | |
| 1 | 0 | 1 | 1 | 1 | 0 | | | | | | |
| 1 | 1 | 0 | 1 | 1 | 1 | | | | | | |
| 1 | 1 | 1 | 0 | 0 | 0 | | | | | | |

Typische Schaltwerke: Synchroner Zähler (7)

Zustandsübergänge des Zählers: Übergänge pro Flip-Flop ermitteln

| Zustände (gleichzeitig Ausgabe) | | | Folgezustand | | | Eingänge der Flip-Flops | | | | | |
|---------------------------------------|-------|-------|--------------|--------|--------|-------------------------|-------|-------|-------|-------|-------|
| Q_2 | Q_1 | Q_0 | Q'_2 | Q'_1 | Q'_0 | J_2 | K_2 | J_1 | K_1 | J_0 | K_0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | * | 0 | * | 1 | * |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | * | 1 | * | * | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | * | * | 0 | 1 | * |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | * | * | 1 | * | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | * | 0 | 0 | * | 1 | * |
| 1 | 0 | 1 | 1 | 1 | 0 | * | 0 | 1 | * | * | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | * | 0 | * | 0 | 1 | * |
| 1 | 1 | 1 | 0 | 0 | 0 | * | 1 | * | 1 | * | 1 |

Typische Schaltwerke: Synchroner Zähler (8)

Schaltnetzentwurf

- Eingänge sind Ausgänge Q_i der Flip-Flops
- Ausgänge sind Ansteuerungen J_i und K_i der Flip-Flops

Einsatz von KV-Diagrammen zur Schaltungsminimierung

- Beispiel: Schaltfunktion K_2

$\overline{Q_1}$ Q_1 $\overline{Q_1}$
 ————— $\overline{Q_0}$ ————— Q_0 —————

| | | | | |
|------------------|---|---|---|---|
| $\overline{Q_2}$ | d | d | d | d |
| Q_2 | 0 | 0 | 1 | 0 |

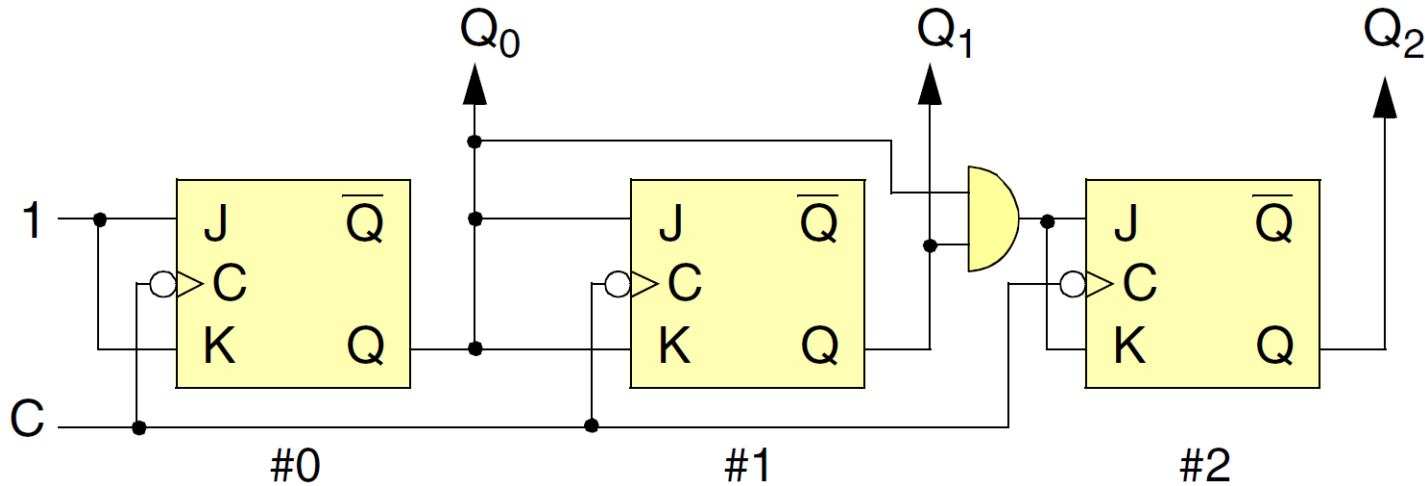
- DNF der Schaltfunktion: $K_2 = Q_0 * Q_1$

Typische Schaltwerke: Synchroner Zähler (9)

Schaltfunktionen insgesamt

- $J_2 = Q_0 * Q_1$ $K_2 = Q_0 * Q_1$
- $J_1 = Q_0$ $K_1 = Q_0$
- $J_0 = 1$ $K_0 = 1$

Realisierung der Schaltung



Systematischer Schaltwerkdentwurf (1)

Entwurf beliebiger synchroner Schaltwerke mit internem Zustand

- Wie kommt man allgemein von den Systemanforderungen zum Schaltwerk?

☞ Endliche Automaten als Systemmodell

- Endliche Menge von Zuständen
- Übergänge zwischen den Zuständen
- Abhängigkeit der Übergänge von
 - Eingabewerten und
 - vorherigen Zuständen

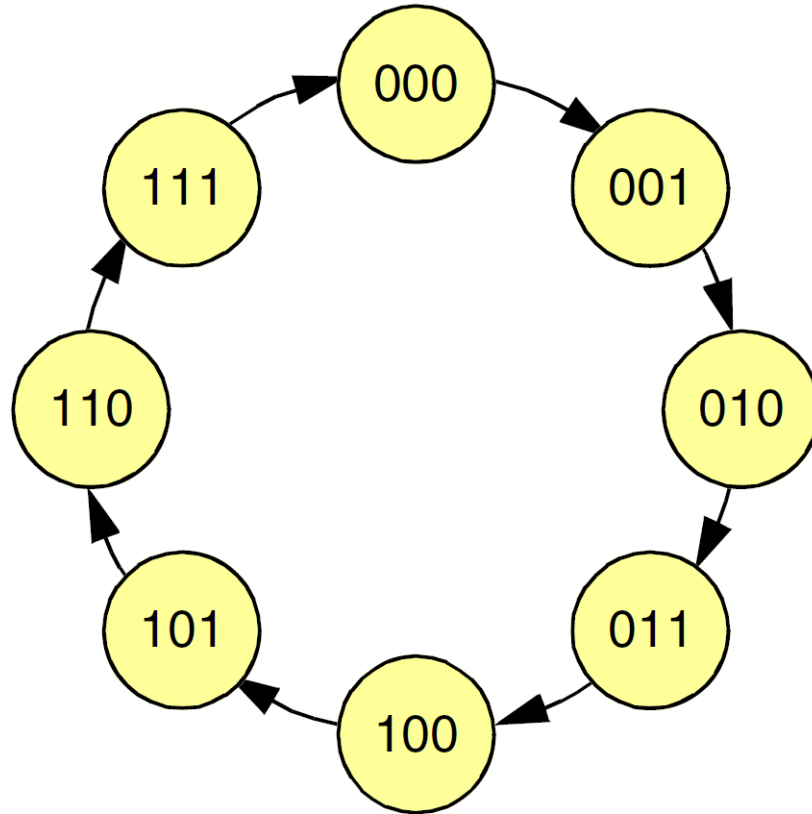
Roter Faden

3. Sequentielle Logik

- Einleitung
- Flip-Flops
- Typische Schaltwerke
 - Schieberegister
 - Asynchroner Zähler
 - Synchroner Zähler
- Systematischer Schaltwerkentwurf

Systematischer Schaltwerkdentwurf (2)

Beispiel: dreistelliger Binärzähler



- Acht Zustände
- Unbedingte Übergänge pro Taktzyklus
 - Keine Abhängigkeit von Eingabewerten
- Ausgabe
 - Direkte Ausgabe der Zustandsrepräsentation durch Flip-Flop-Ausgänge

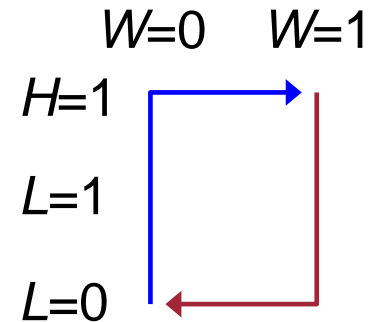
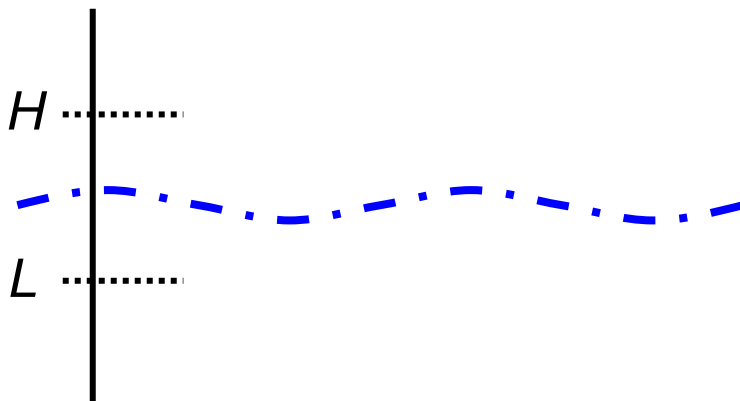


Gesucht: Allgemeines Verfahren zur Synthese synchroner Schaltwerke aus der Beschreibung endlicher Automaten

Beispiel „Hochwassererkennung“ (1)

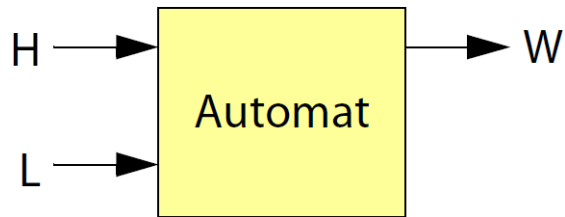
Szenario

- Wasserstandsanzeige
 - „Hochwasser“: $W = 1$
 - „Niedrigwasser“: $W = 0$
- Stabilisierung durch Hysterese
 - Doppelte Wasserstandsmessung an Punkten H und L
 - $H = 1$ bzw. $L = 1$, wenn Wasser oberhalb des jeweiligen Wasserstandssensors



Beispiel „Hochwassererkennung“ (2)

Gesucht: geeignetes Schaltwerk

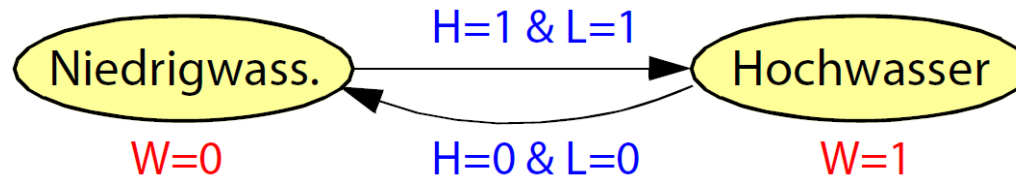


- Synchrones Schaltwerk
- Mehrere Varianten des systematischen Entwurfs denkbar

Beispiel: 1. Variante (1)

Endlicher Automat

- Ausgaben gekoppelt an Zustände

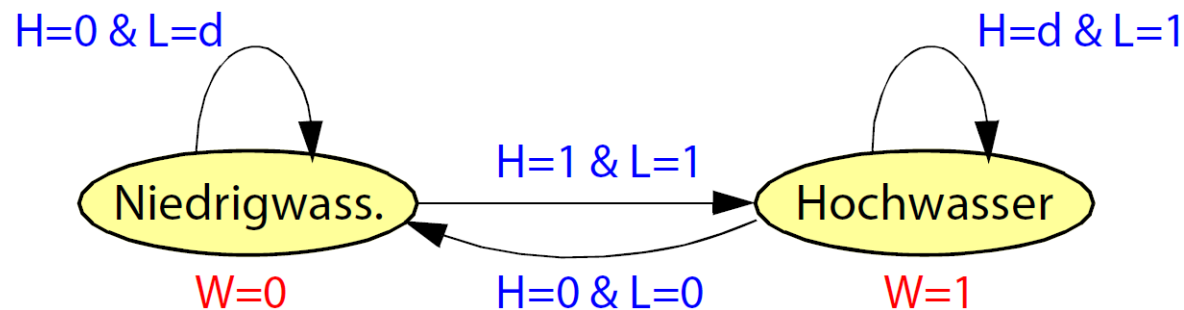


- Mindestens so viele Zustände wie mögliche Ausgaben
 - Evtl. mehr, da Schaltwerk sich irgendetwas merken muss
- Zustandsübergänge abhängig von Eingabewerten
 - Markierung der Kanten
 - Nicht angegebene Kombinationen implizieren Übergang in vorigen Zustand
 - Besser: alle Kombinationen angeben

Beispiel: 1. Variante (2)

Endlicher Automat

- Ausgaben gekoppelt an Zustände

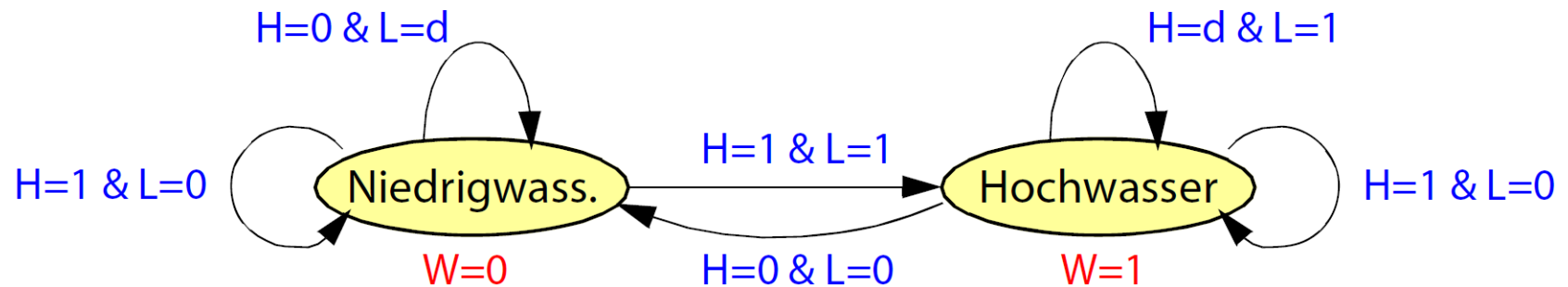


- Mindestens so viele Zustände wie mögliche Ausgaben
 - Evtl. mehr, da Schaltwerk sich irgendetwas merken muss
- Zustandsübergänge abhängig von Eingabewerten
 - Markierung der Kanten
 - Nicht angegebene Kombinationen implizieren Übergang in vorigen Zustand
 - Besser: alle Kombinationen angeben

Beispiel: 1. Variante (3)

Endlicher Automat

- Ausgaben gekoppelt an Zustände

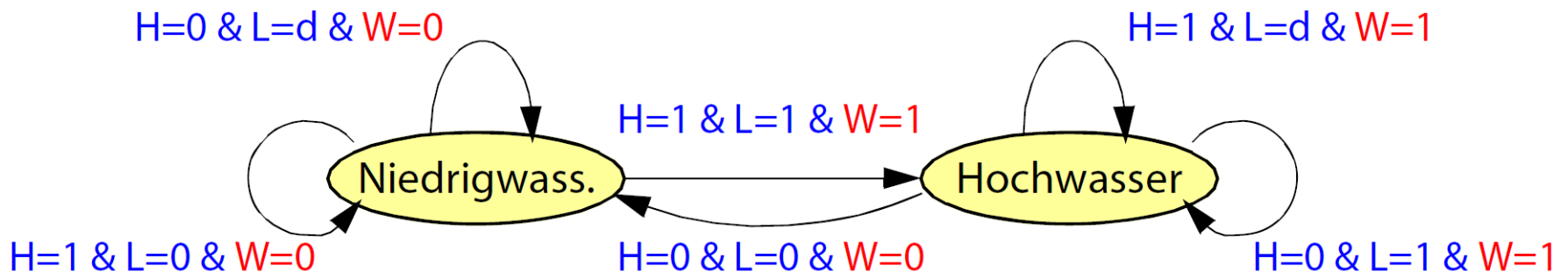


- Mindestens so viele Zustände wie mögliche Ausgaben
 - Evtl. mehr, da Schaltwerk sich irgendetwas merken muss
- Zustandsübergänge abhängig von Eingabewerten
 - Markierung der Kanten
 - Nicht angegebene Kombinationen implizieren Übergang in vorigen Zustand
 - Besser: alle Kombinationen angeben

Beispiel: 2. Variante

Endlicher Automat

- Ausgaben gekoppelt an Zustände und aktuelle Eingänge



- Zustandsübergänge abhängig von Eingabewerten
 - Markierung der Kanten mit notwendigen Eingabewerten und mit zugehörigen Ausgaben
 - Alle Kombinationen müssen angegeben werden (sonst fehlen Ausgabeinformationen)

Moore-Automat

Aufbau eines Moore-Automaten (Edward F. Moore, Bell Labs)

- Getaktetes System



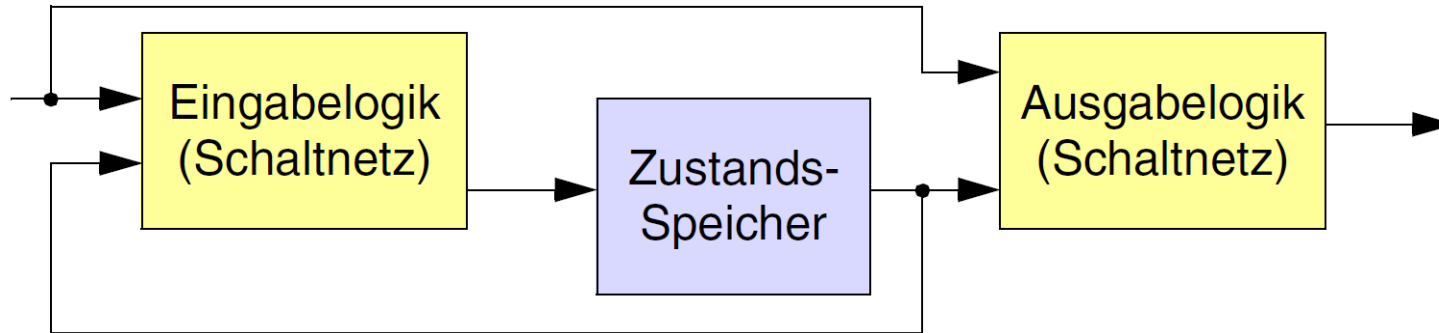
- Eingabewerte und bisheriger Zustand führen zu Zustandsveränderungen (Zustandsübergänge)
- Ausgabewerte hängen von augenblicklichem Zustand ab

☞ **Ausgabe mit Zustand assoziiert**

Mealy-Automat

Aufbau eines Mealy-Automaten (George H. Mealy, IBM)

- Getaktetes System



- Eingabewerte und bisheriger Zustand führen zu Zustandsveränderungen (Zustandsübergänge)
- Ausgabewerte hängen von Eingabewerten und augenblicklichem Zustand ab

☞ **Ausgabe mit Zustandsübergängen assoziiert**

Vorgehensweise

Einzelsschritte für Moore-Automaten

1. Zustandsdiagramm bzw. Zustandstabelle
2. Binäre Zustandskodierung, binäre Zustandstabelle
3. Auswahl eines Flip-Flop-Typs, Flip-Flop-Ansteuerung in Zustandstabelle
4. Wahrheitstabelle für Ausgabefunktionen
5. Minimierung von Ansteuerungs- und Ausgabefunktionen
6. Aufbau der Schaltung

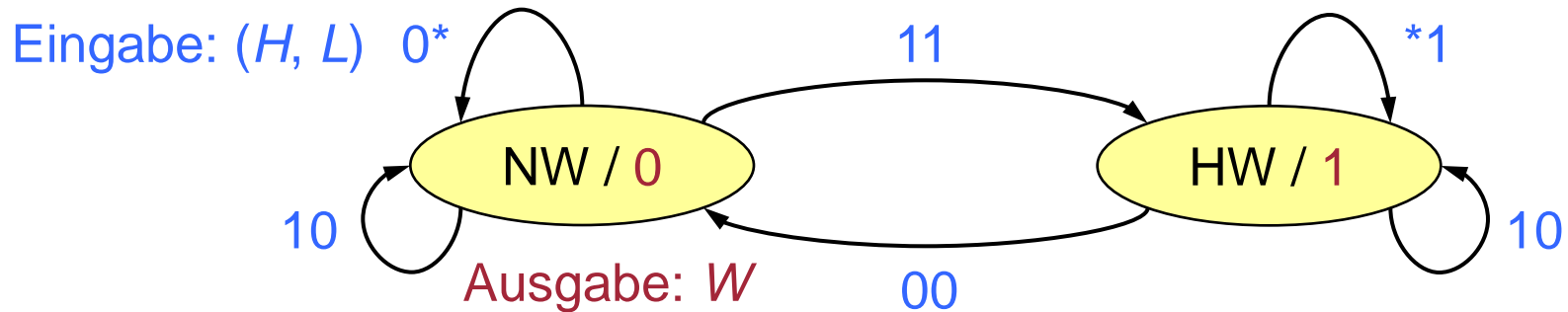
Einzelsschritte für Mealy-Automaten

1. Zustandsdiagramm bzw. Zustandstabelle *einschl. Ausgaben*
2. Binäre Zustandskodierung, binäre Zustandstabelle *einschl. Ausgaben*
3. Auswahl eines Flip-Flop-Typs, Flip-Flop-Ansteuerung in Zustandstabelle
4. Minimierung von Ansteuerungs- und Ausgabefunktionen
5. Aufbau der Schaltung

Moore-Automat für Hochwassererkennung (1)

Schritt 1a: Zustandsdiagramm

- Verkürzte Darstellung



- Ausgabewerte $Y = Y_1, Y_2, \dots, Y_m$ werden hinter die Zustandsbezeichnung geschrieben (hier: $m = 1$ und $Y_1 = W$)
- Eingabewerte werden als Tupel direkt an den Kanten notiert

Moore-Automat für Hochwassererkennung (2)

Schritt 1b: Zustandstabelle

| Zustände | Eingänge | | Folge- zustände |
|----------|----------|----------|--------------------|
| | <i>H</i> | <i>L</i> | |
| NW | 0 | * | NW |
| NW | 1 | 0 | NW |
| NW | 1 | 1 | HW |
| HW | 0 | 0 | NW |
| HW | * | 1 | HW |
| HW | 1 | 0 | HW |

- Alle Kanten bzw. Zustandsübergänge erfasst
 - Gleichwertig mit Zustandsdiagramm (außer Ausgaben)

Moore-Automat für Hochwassererkennung (3)

Schritt 2: Binäre Zustandskodierung, binäre Zustandstabelle

| Zustände | | Eingänge | | Folgezustände | |
|----------|-------|----------|-----|---------------|--------|
| | Q_0 | H | L | | Q_0' |
| NW | 1 | 0 | * | NW | 1 |
| NW | 1 | 1 | 0 | NW | 1 |
| NW | 1 | 1 | 1 | HW | 0 |
| HW | 0 | 0 | 0 | NW | 1 |
| HW | 0 | * | 1 | HW | 0 |
| HW | 0 | 1 | 0 | HW | 0 |

- Zustände müssen codiert werden
 - Hier: NW = 1, HW = 0
 - Ein Flip-Flop erforderlich zur Zustandsrepräsentation

Moore-Automat für Hochwassererkennung (4)

Schritt 2: Binäre Zustandskodierung, binäre Zustandstabelle

| Zustände | | Eingänge | | Folgezustände | |
|----------|-------|----------|-----|---------------|--------|
| | Q_0 | H | L | | Q_0' |
| NW | 1 | 0 | * | NW | 1 |
| NW | 1 | 1 | 0 | NW | 1 |
| NW | 1 | 1 | 1 | HW | 0 |
| HW | 0 | 0 | 0 | NW | 1 |
| HW | 0 | * | 1 | HW | 0 |
| HW | 0 | 1 | 0 | HW | 0 |

☞ Auf Vollständigkeit achten!

- Spätestens hier müssen alle möglichen Übergänge erfasst werden
- Auch Zustände außerhalb des Automaten müssen erfasst werden
 - Z.B. vierter Zustand bei Automat mit 3 Zuständen und 2 Flip-Flops

Moore-Automat für Hochwassererkennung (5)

Schritt 3: Auswahl JK-Flip-Flops, Ermitteln der Flip-Flop-Ansteuerung

| Zustände Q_0 | Eingänge | | Folge- zust. Q_0' | Ansteuerung | |
|-------------------|----------|-----|---------------------------|-------------|-------|
| | H | L | | J_0 | K_0 |
| 1 | 0 | * | 1 | | |
| 1 | 1 | 0 | 1 | | |
| 1 | 1 | 1 | 0 | | |
| 0 | 0 | 0 | 1 | | |
| 0 | * | 1 | 0 | | |
| 0 | 1 | 0 | 0 | | |

Moore-Automat für Hochwassererkennung (6)

Schritt 3: Auswahl JK-Flip-Flops, Ermitteln der Flip-Flop-Ansteuerung

☞ **Zustandsübergangstabelle JK-Flip-Flop** (siehe Folie 57)

| Übergang $Q \rightarrow Q'$ | J | K |
|--------------------------------|-----|-----|
| $0 \rightarrow 0$ | 0 | * |
| $0 \rightarrow 1$ | 1 | * |
| $1 \rightarrow 0$ | * | 1 |
| $1 \rightarrow 1$ | * | 0 |

Eingang K irrelevant

Eingang K irrelevant

Eingang J irrelevant

Eingang J irrelevant

Moore-Automat für Hochwassererkennung (7)

Schritt 3: Auswahl JK-Flip-Flops, Ermitteln der Flip-Flop-Ansteuerung

| Zustände Q_0 | Eingänge | | Folge- zust. Q_0' | Ansteuerung | |
|-------------------|----------|-----|---------------------------|-------------|-------|
| | H | L | | J_0 | K_0 |
| 1 | 0 | * | 1 | * | 0 |
| 1 | 1 | 0 | 1 | | |
| 1 | 1 | 1 | 0 | | |
| 0 | 0 | 0 | 1 | | |
| 0 | * | 1 | 0 | | |
| 0 | 1 | 0 | 0 | | |

Moore-Automat für Hochwassererkennung (8)

Schritt 3: Auswahl JK-Flip-Flops, Ermitteln der Flip-Flop-Ansteuerung

| Zustände Q_0 | Eingänge | | Folge- zust. Q_0' | Ansteuerung | |
|-------------------|----------|-----|---------------------------|-------------|-------|
| | H | L | | J_0 | K_0 |
| 1 | 0 | * | 1 | * | 0 |
| 1 | 1 | 0 | 1 | * | 0 |
| 1 | 1 | 1 | 0 | * | 1 |
| 0 | 0 | 0 | 1 | 1 | * |
| 0 | * | 1 | 0 | 0 | * |
| 0 | 1 | 0 | 0 | 0 | * |

Moore-Automat für Hochwassererkennung (9)

Schritt 4: Aufstellen der Ausgabefunktionen in Abhängigkeit vom Zustand

| Zustände | | Ausgabe |
|----------|-------|---------|
| | Q_0 | W |
| NW | 1 | 0 |
| HW | 0 | 1 |

Moore-Automat für Hochwassererkennung (10)

Schritt 5: Minimierung der Ansteuer- und Ausgabefunktionen

– $J_0 = \overline{L} * \overline{H}$

$$K_0 = L * H$$

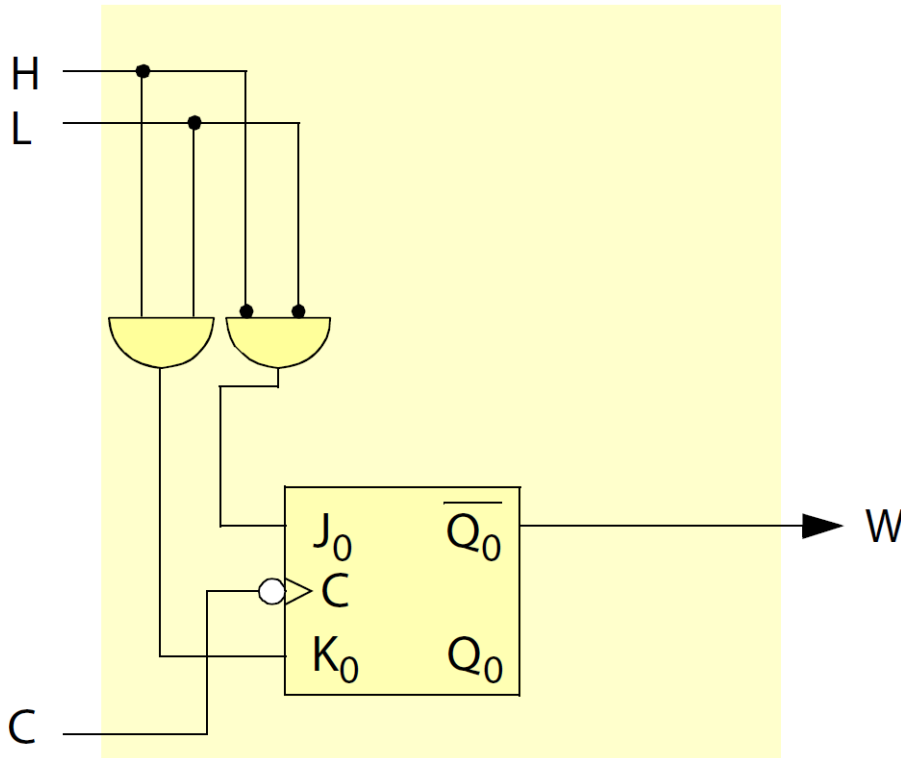
– $W = \overline{Q_0}$

☞ **Hinweis**

- Andere Zustandskodierung hätte noch einfachere Ausgabefunktion zur Folge

Moore-Automat für Hochwassererkennung (11)

Schritt 6: Aufbau der Schaltung

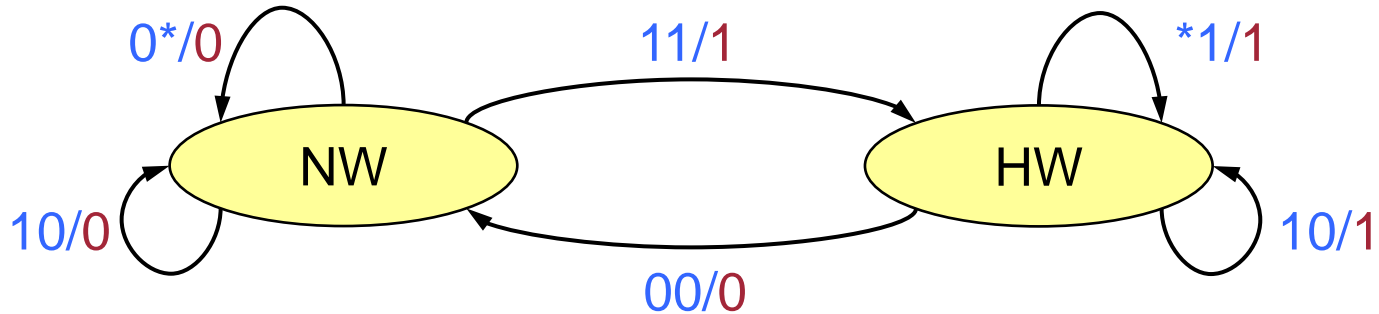


Mealy-Automat für Hochwassererkennung (1)

Schritt 1a: Zustandsdiagramm

- Verkürzte Darstellung

Eingabe/Ausgabe: $(H, L)/W$



- Eingabe- und Ausgabewerte werden als Tupel direkt an den Kanten notiert

Mealy-Automat für Hochwassererkennung (2)

Schritt 1b: Zustandstabelle

| Zustände | Eingänge | | Folge- zustände | Ausgänge W |
|----------|----------|-----|--------------------|-----------------|
| | H | L | | |
| NW | 0 | * | NW | 0 |
| NW | 1 | 0 | NW | 0 |
| NW | 1 | 1 | HW | 1 |
| HW | 0 | 0 | NW | 0 |
| HW | * | 1 | HW | 1 |
| HW | 1 | 0 | HW | 1 |

- Alle Kanten bzw. Zustandsübergänge einschließlich Ausgaben erfasst
 - Gleichwertig mit Zustandsdiagramm

Mealy-Automat für Hochwassererkennung (3)

Schritt 2: Binäre Zustandskodierung, binäre Zustandstabelle

| Zustände | | Eingänge | | Folgezustände | | Ausgänge |
|----------|-------|----------|-----|---------------|--------|----------|
| | Q_0 | H | L | | Q_0' | W |
| NW | 1 | 0 | * | NW | 1 | 0 |
| NW | 1 | 1 | 0 | NW | 1 | 0 |
| NW | 1 | 1 | 1 | HW | 0 | 1 |
| HW | 0 | 0 | 0 | NW | 1 | 0 |
| HW | 0 | * | 1 | HW | 0 | 1 |
| HW | 0 | 1 | 0 | HW | 0 | 1 |

- Zustände müssen codiert werden
 - Hier: NW = 1, HW = 0
 - Ein Flip-Flop erforderlich zur Zustandsrepräsentation

Mealy-Automat für Hochwassererkennung (4)

Schritt 2: Binäre Zustandskodierung, binäre Zustandstabelle

| Zustände | | Eingänge | | Folgezustände | | Ausgänge |
|----------|-------|----------|-----|---------------|--------|----------|
| | Q_0 | H | L | | Q_0' | W |
| NW | 1 | 0 | * | NW | 1 | 0 |
| NW | 1 | 1 | 0 | NW | 1 | 0 |
| NW | 1 | 1 | 1 | HW | 0 | 1 |
| HW | 0 | 0 | 0 | NW | 1 | 0 |
| HW | 0 | * | 1 | HW | 0 | 1 |
| HW | 0 | 1 | 0 | HW | 0 | 1 |

☞ Auf Vollständigkeit achten!

- Spätestens hier müssen alle möglichen Übergänge erfasst werden
- Wie bei Moore-Automat

Mealy-Automat für Hochwassererkennung (5)

Schritt 3: Auswahl JK-Flip-Flops, Ermitteln der Flip-Flop-Ansteuerung

| Zustände Q_0 | Eingänge | | Folge- zust. Q_0' | Ansteuerung | | Ausgänge W |
|-------------------|----------|-----|---------------------------|-------------|-------|-----------------|
| | H | L | | J_0 | K_0 | |
| 1 | 0 | * | 1 | * | 0 | 0 |
| 1 | 1 | 0 | 1 | * | 0 | 0 |
| 1 | 1 | 1 | 0 | * | 1 | 1 |
| 0 | 0 | 0 | 1 | 1 | * | 0 |
| 0 | * | 1 | 0 | 0 | * | 1 |
| 0 | 1 | 0 | 0 | 0 | * | 1 |

– Tabelle identisch zum Moore-Automaten (zusätzlich Ausgänge)

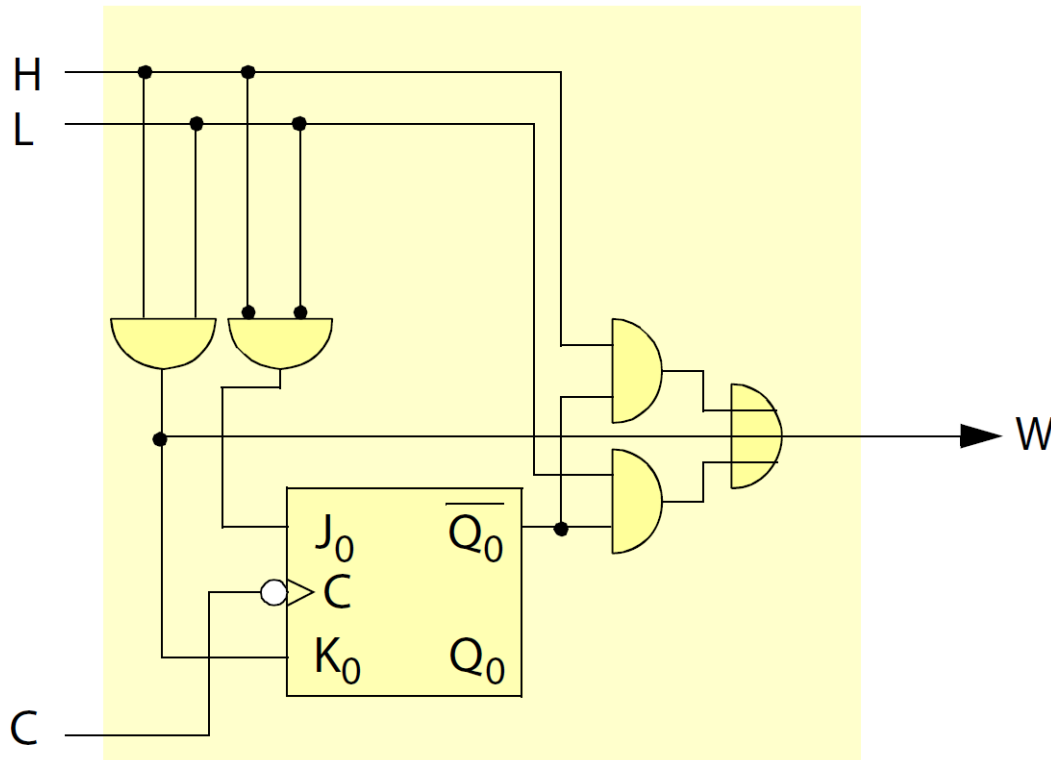
Mealy-Automat für Hochwassererkennung (6)

Schritt 4: Minimierung der Ansteuer- und Ausgabefunktionen

$$\begin{aligned} - J_0 &= \overline{L} * \overline{H} & K_0 &= L * H \\ - W &= \overline{Q_0} * H + \overline{Q_0} * L + H * L \end{aligned}$$

Mealy-Automat für Hochwassererkennung (7)

Schritt 6: Aufbau der Schaltung

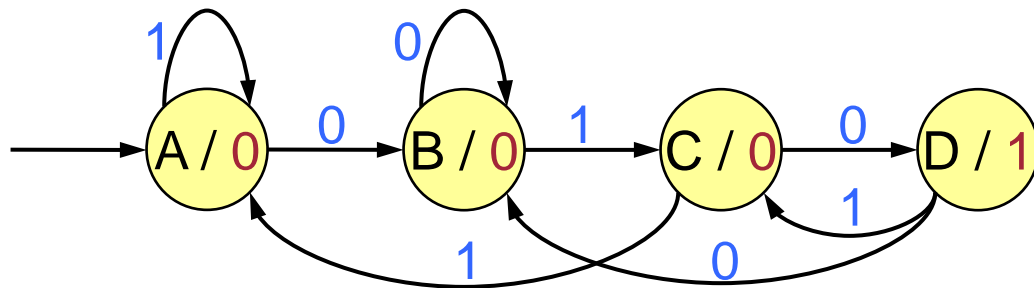


Moore-Automat zur Sequenzerkennung (1)

Sequenzerkennung

- Ein binärer Eingang E
- Ein binärer Ausgang Y der 1 ist, falls über die letzten Taktzyklen hinweg an E eine zu erkennende Sequenz von Binärwerten anlag (im folgenden: „010“)

Schritt 1a: Zustandsdiagramm



- Bedeutung der Zustände
 - A: Bisher nichts erkannt
 - B: „0“ erkannt
 - C: „01“ erkannt
 - D: „010“ erkannt

Moore-Automat zur Sequenzerkennung (2)

Schritt 1b: Zustandstabelle

| Zustände | Eingang | Folge- zustände |
|----------|---------|--------------------|
| A | 0 | B |
| A | 1 | A |
| B | 0 | B |
| B | 1 | C |
| C | 0 | D |
| C | 1 | A |
| D | 0 | B |
| D | 1 | C |

Moore-Automat zur Sequenzerkennung (3)

Schritt 2: Binäre Zustandskodierung, binäre Zustandstabelle

– Codierung

| Zustände | Eingang |
|----------|---------|
| A | 00 |
| B | 01 |
| C | 10 |
| D | 11 |

– Zustandstabelle

| Zustände | Eingang | Folgezust. |
|----------|---------|------------|
| 00 | 0 | 01 |
| 00 | 1 | 00 |
| 01 | 0 | 01 |
| 01 | 1 | 10 |
| 10 | 0 | 11 |
| 10 | 1 | 00 |
| 11 | 0 | 01 |
| 11 | 1 | 10 |

Moore-Automat zur Sequenzerkennung (4)

Schritt 3: JK-Flip-Flops und deren Ansteuerung

| Zustände Q_1, Q_0 | Eingang | Folge- zustände | J_1 | K_1 | J_0 | K_0 |
|------------------------|---------|--------------------|-------|-------|-------|-------|
| 00 | 0 | 01 | 0 | * | 1 | * |
| 00 | 1 | 00 | 0 | * | 0 | * |
| 01 | 0 | 01 | 0 | * | * | 0 |
| 01 | 1 | 10 | 1 | * | * | 1 |
| 10 | 0 | 11 | * | 0 | 1 | * |
| 10 | 1 | 00 | * | 1 | 0 | * |
| 11 | 0 | 01 | * | 1 | * | 0 |
| 11 | 1 | 10 | * | 0 | * | 1 |

Moore-Automat zur Sequenzerkennung (5)

Schritt 4: Ausgabefunktion in Abhängigkeit vom Zustand

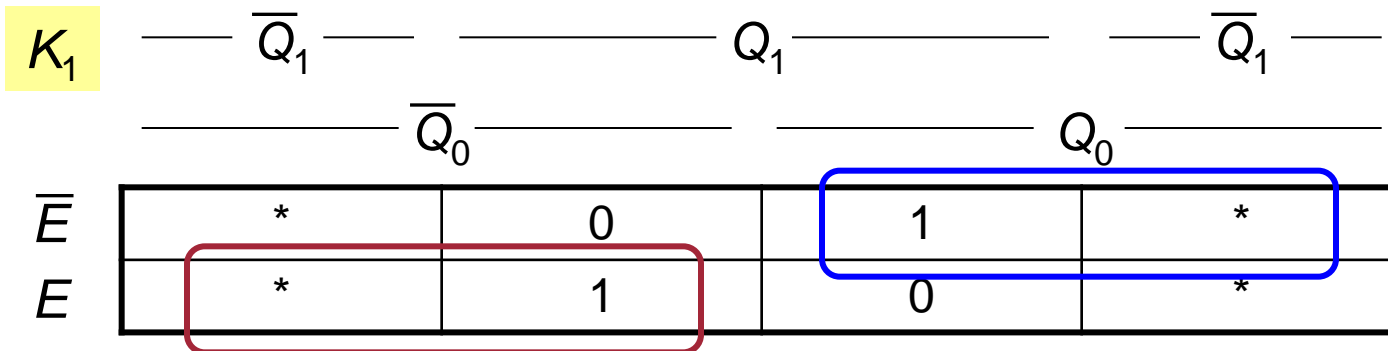
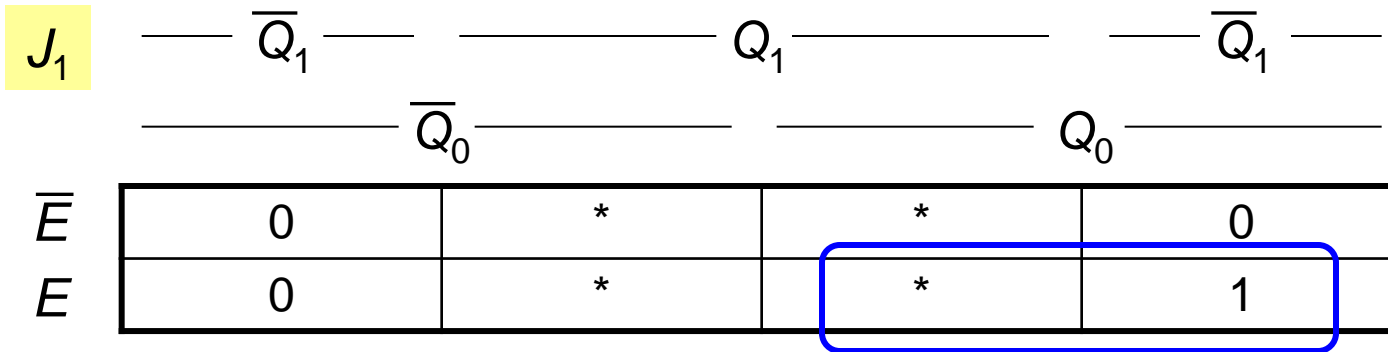
| Zustände Q_1, Q_0 | Ausgabe Y |
|------------------------|----------------|
| 00 | 0 |
| 01 | 0 |
| 10 | 0 |
| 11 | 1 |

Schritt 5: Minimierung

– $Y = Q_1 * Q_0$ trivial

Moore-Automat zur Sequenzerkennung (6)

– Flip-Flop #1

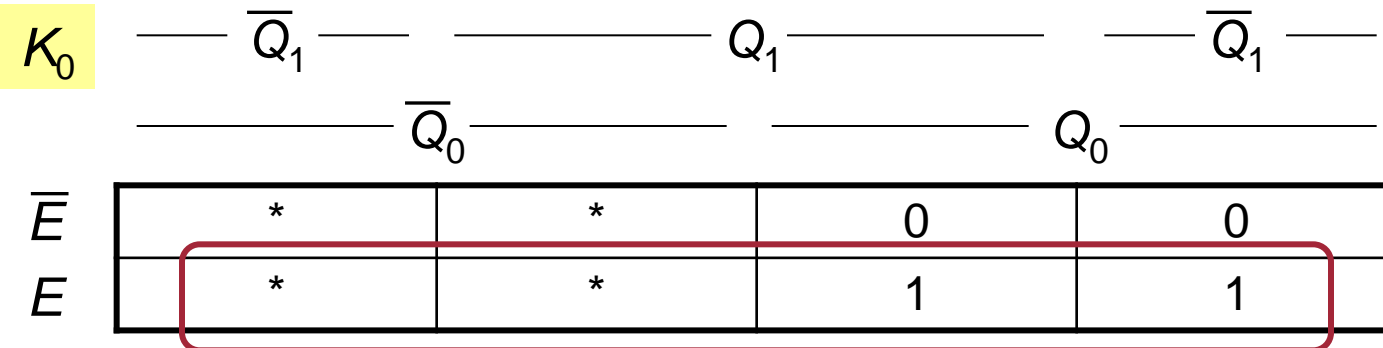
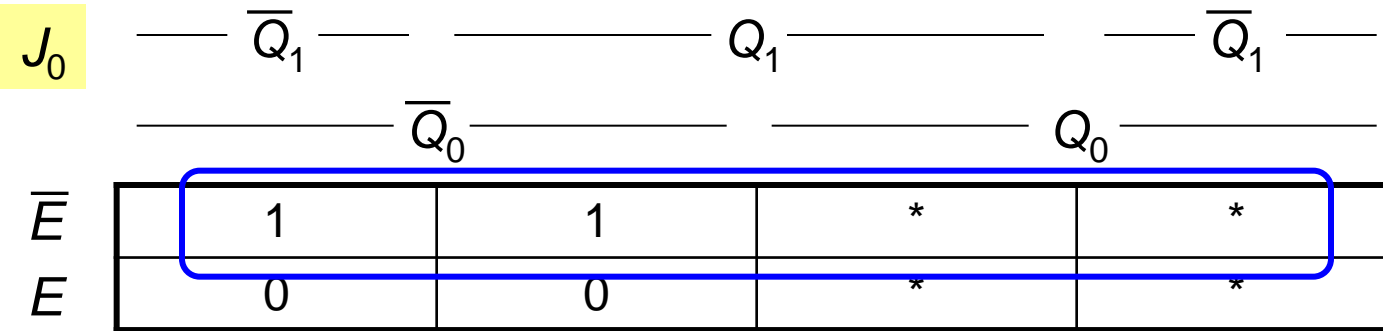


– $J_1 = Q_0 * E$

$K_1 = Q_0 * \overline{E} + \overline{Q_0} * E$

Moore-Automat zur Sequenzerkennung (7)

– Flip-Flop #0

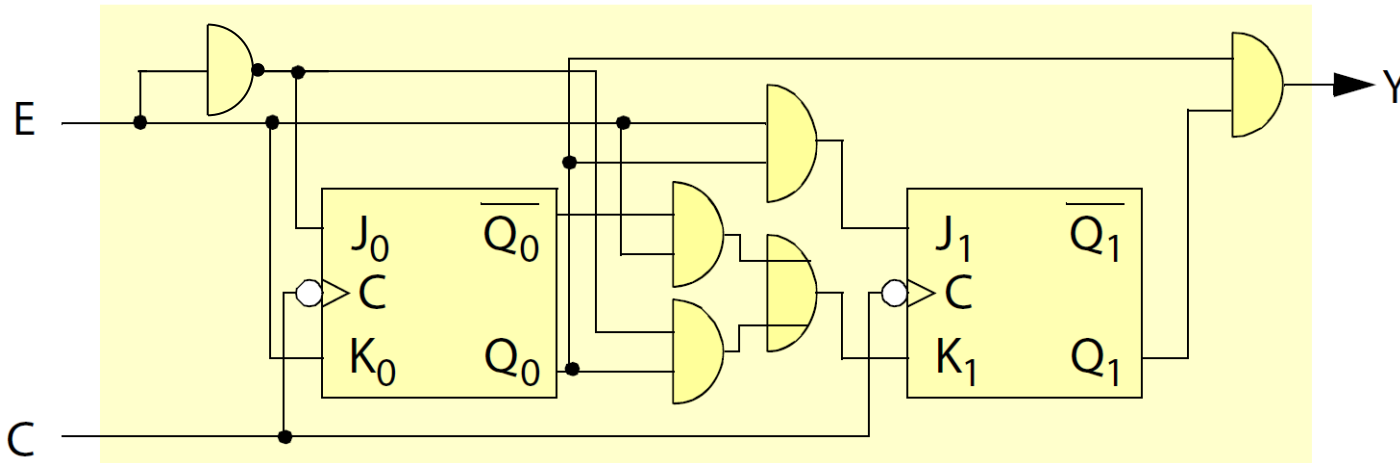


– $J_0 = \bar{E}$

$K_0 = E$

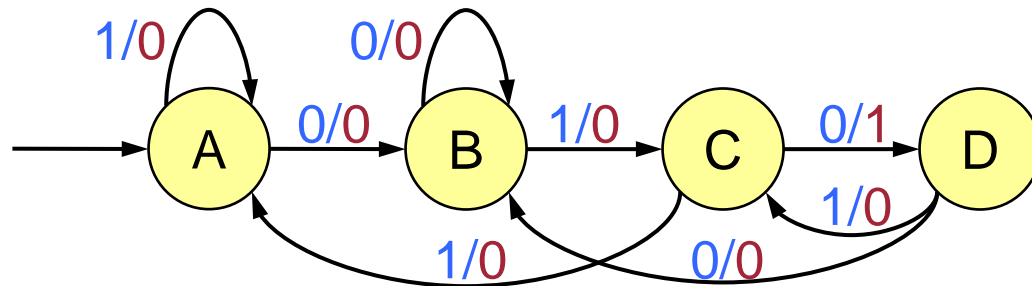
Moore-Automat zur Sequenzerkennung (8)

Schritt 6: Aufbau der Schaltung



Mealy-Automat zur Sequenzerkennung (1)

Schritt 1a: Zustandsdiagramm



- Kanten werden mit E/Y beschriftet
 - $E = E_1, E_2, \dots, E_n$ sind Eingabewerte (hier: $n = 1$)
 - $Y = Y_1, Y_2, \dots, Y_m$ sind Ausgabewerte (hier: $m = 1$)
- Bedeutung der Zustände entsprechend zum Moore-Automaten

Mealy-Automat zur Sequenzerkennung (2)

Schritt 1b: Zustandstabelle

| Zustände | Eingang | Folge- zustände | Ausgang |
|----------|---------|--------------------|---------|
| A | 0 | B | 0 |
| A | 1 | A | 0 |
| B | 0 | B | 0 |
| B | 1 | C | 0 |
| C | 0 | D | 1 |
| C | 1 | A | 0 |
| D | 0 | B | 0 |
| D | 1 | C | 0 |

- Erste drei Spalten identisch zu Moore-Automat (!)

Mealy-Automat zur Sequenzerkennung (3)

Schritt 2: Binäre Zustandskodierung, binäre Zustandstabelle

– Identisch zu Moore-Automat (bis auf zusätzliche Ausgangsspalte)

Schritt 3: JK-Flip-Flops und deren Ansteuerung

| Zustände Q_1, Q_0 | Eingang | Folge- zustände | J_1 | K_1 | J_0 | K_0 | Aus- gang |
|------------------------|---------|--------------------|-------|-------|-------|-------|--------------|
| 00 | 0 | 01 | 0 | * | 1 | * | 0 |
| 00 | 1 | 00 | 0 | * | 0 | * | 0 |
| 01 | 0 | 01 | 0 | * | * | 0 | 0 |
| 01 | 1 | 10 | 1 | * | * | 1 | 0 |
| 10 | 0 | 11 | * | 0 | 1 | * | 1 |
| 10 | 1 | 00 | * | 1 | 0 | * | 0 |
| 11 | 0 | 01 | * | 1 | * | 0 | 0 |
| 11 | 1 | 10 | * | 0 | * | 1 | 0 |

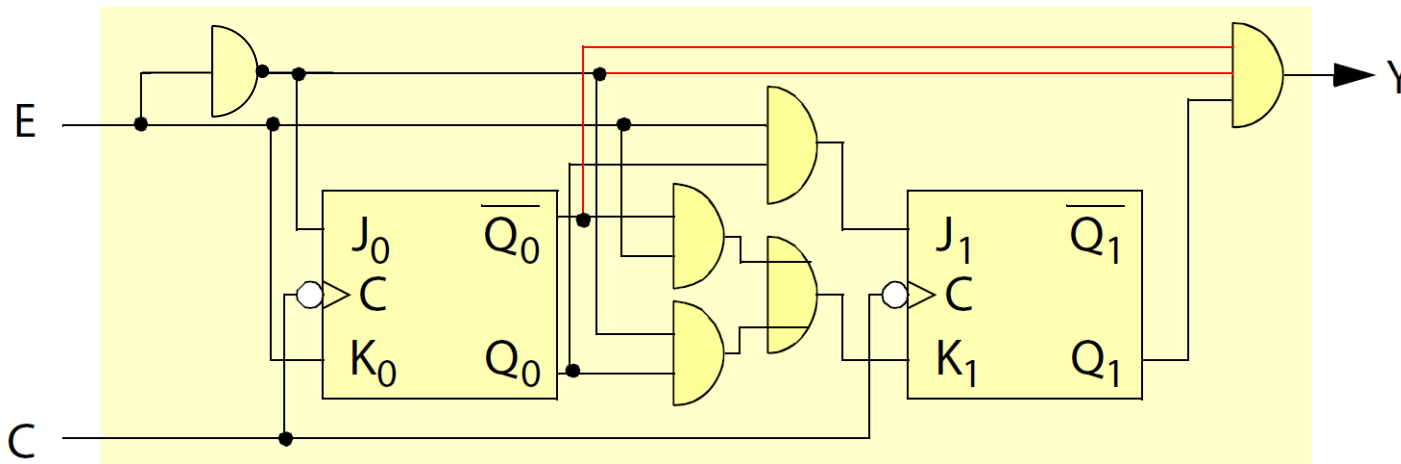
– Identisch zu Moore-Automat (bis auf zusätzliche Ausgangsspalte)

Mealy-Automat zur Sequenzerkennung (4)

Schritt 4: Minimierung

- $Y = Q_1 * \overline{Q_0} * \overline{E}$
- Flip-Flop-Ansteuerfunktionen identisch zu Moore-Automat

Schritt 5: Aufbau der Schaltung



Vergleich Moore- und Mealy-Automaten (1)

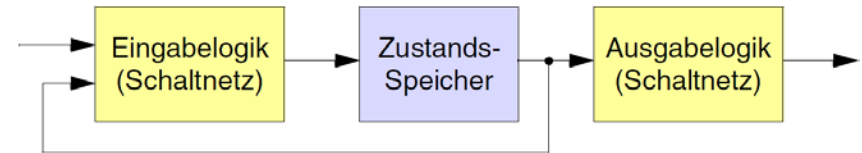
Beide geeignet zum Aufbau beliebiger synchroner Schaltwerke

Vorteile Moore-Automat

- Geringerer Schaltungsaufwand, wenn Ausgabewerte nur vom aktuellen Zustand abhängen
- Taktsynchrone Ausgabe

Nachteile Moore-Automat

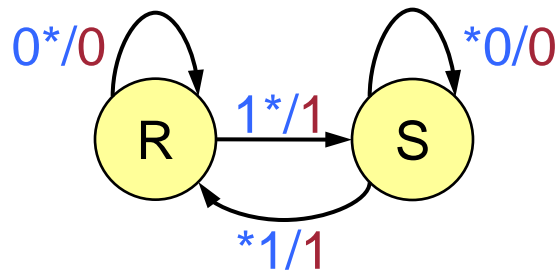
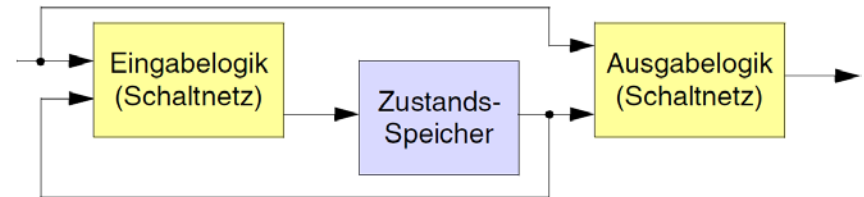
- Reaktion erst im nächsten Taktzyklus



Vergleich Moore- und Mealy-Automaten (2)

Vorteile Mealy-Automat

- Ausgang kann sofort auf Eingänge reagieren
- Geringerer Schaltungsaufwand, wenn Übergänge zu einem Zustand verschiedene Ausgabewerte erzeugen sollen
 - Beispiel: modifiziertes JK-Flip-Flop



MJK-Flip-Flop
JK/M

- Ausgabe zeigt an, dass sich Flip-Flop-Zustand geändert hat

Nachteile Mealy-Automat

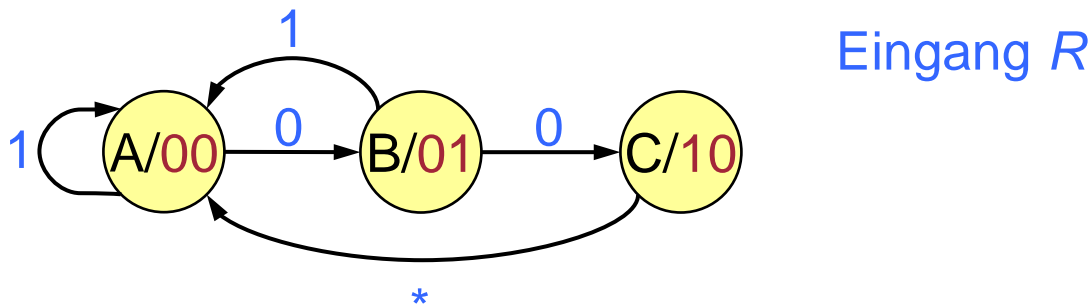
- Asynchrone Eingangesignale bewirken asynchrone Ausgangesignale

Einfluss des Flip-Flop-Typs (1)

Beispiel: Synchroner Zähler von 0 bis 2

- *Reset*-Leitung R : $R = 1 \rightarrow$ Zurück zur 0
- Realisierung als Moore-Automat

Schritt 1a: Zustandsdiagramm



Schritt 1b: Zustandstabelle

- Dem Leser überlassen

Einfluss des Flip-Flop-Typs (2)

Schritt 2: Binäre Zustandskodierung, binäre Zustandstabelle

– Codierung

| Zustände | Eingang |
|----------|---------|
| A | 00 |
| B | 01 |
| C | 10 |
| – | 11 |

– Zustandstabelle

| Zustände | Eingang | Folgezust. |
|----------|---------|------------|
| 00 | 0 | 01 |
| 00 | 1 | 00 |
| 01 | 0 | 10 |
| 01 | 1 | 00 |
| 10 | * | 00 |
| 11 | * | ** |

Einfluss des Flip-Flop-Typs (3)

Schritt 3: JK-Flip-Flops und deren Ansteuerung

| Zustände Q_1, Q_0 | Eingang R | Folge- zustände | J_1 | K_1 | J_0 | K_0 |
|------------------------|----------------|--------------------|-------|-------|-------|-------|
| 00 | 0 | 01 | 0 | * | 1 | * |
| 00 | 1 | 00 | 0 | * | 0 | * |
| 01 | 0 | 10 | 1 | * | * | 1 |
| 01 | 1 | 00 | 0 | * | * | 1 |
| 10 | * | 00 | * | 1 | 0 | * |
| 11 | * | ** | * | * | * | * |

Schritt 4: Ausgabefunktionen

- Trivial wegen geeigneter Zustände

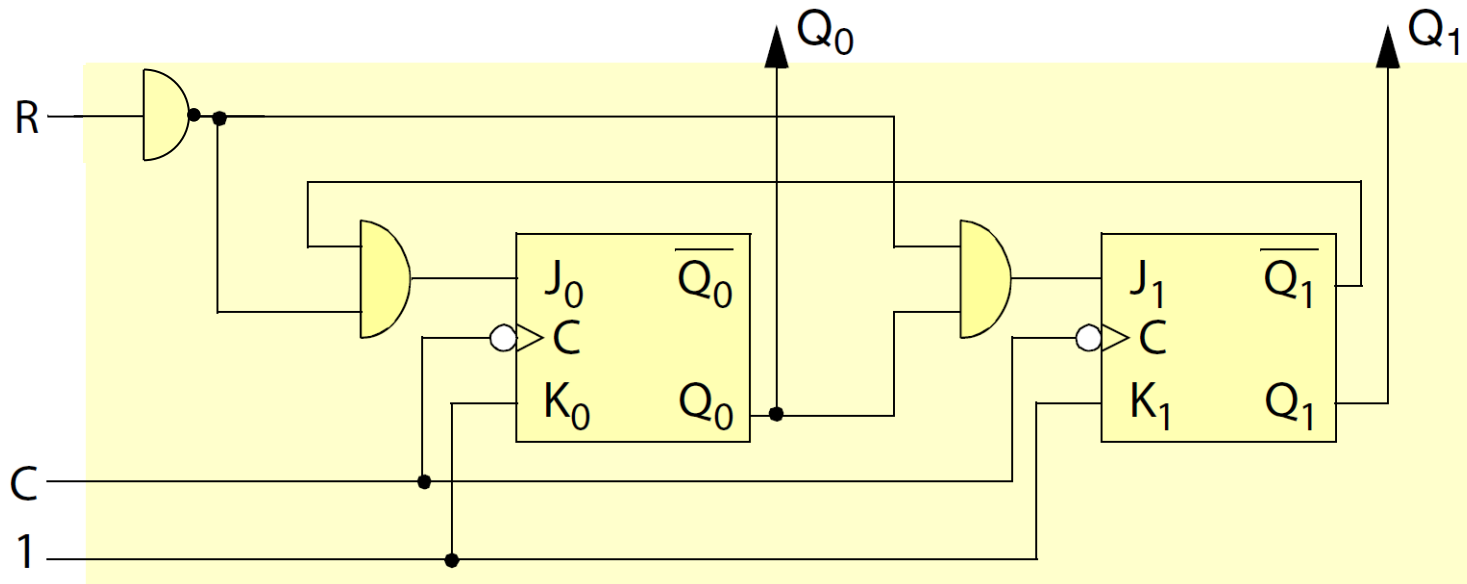
Schritt 5: Minimierung

$$- J_1 = Q_0 * \overline{R}, \quad K_1 = 1$$

$$- J_0 = \overline{Q_1} * \overline{R}, \quad K_0 = 1$$

Einfluss des Flip-Flop-Typs (4)

Schritt 6: Aufbau der Schaltung mit JK-Flip-Flops



Einfluss des Flip-Flop-Typs (5)

☞ T-Flip-Flops

Schritt 3: T-Flip-Flops und deren Ansteuerung

| Zustände Q_1, Q_0 | Eingang R | Folge- zustände | T_1 | T_0 |
|------------------------|----------------|--------------------|-------|-------|
| 00 | 0 | 01 | 0 | 1 |
| 00 | 1 | 00 | 0 | 0 |
| 01 | 0 | 10 | 1 | 1 |
| 01 | 1 | 00 | 0 | 1 |
| 10 | * | 00 | 1 | 0 |
| 11 | * | ** | * | * |

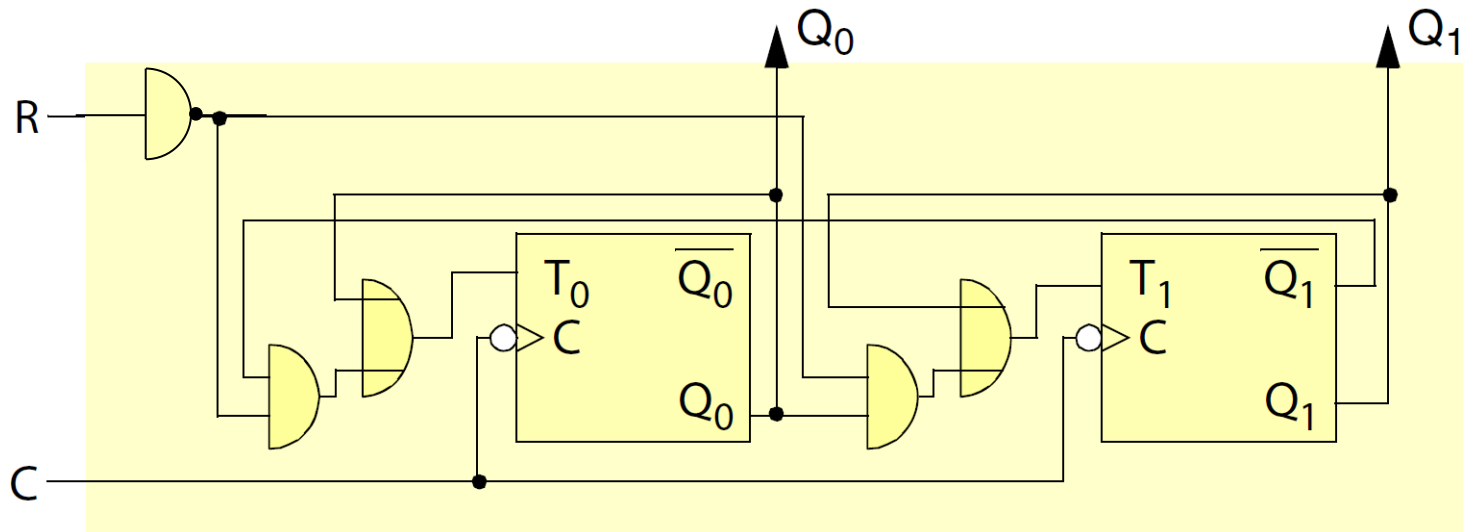
Schritt 5: Minimierung

$$- T_1 = Q_1 + Q_0 * \bar{R}$$

$$- T_0 = Q_0 + \bar{Q}_1 * \bar{R}$$

Einfluss des Flip-Flop-Typs (6)

Schritt 6: Aufbau der Schaltung mit T-Flip-Flops



Einfluss des Flip-Flop-Typs (7)

☞ D-Flip-Flops

Schritt 3: D-Flip-Flops und deren Ansteuerung

| Zustände Q_1, Q_0 | Eingang R | Folge- zustände | D_1 | D_0 |
|------------------------|----------------|--------------------|-------|-------|
| 00 | 0 | 01 | 0 | 1 |
| 00 | 1 | 00 | 0 | 0 |
| 01 | 0 | 10 | 1 | 0 |
| 01 | 1 | 00 | 0 | 0 |
| 10 | * | 00 | 0 | 0 |
| 11 | * | ** | * | * |

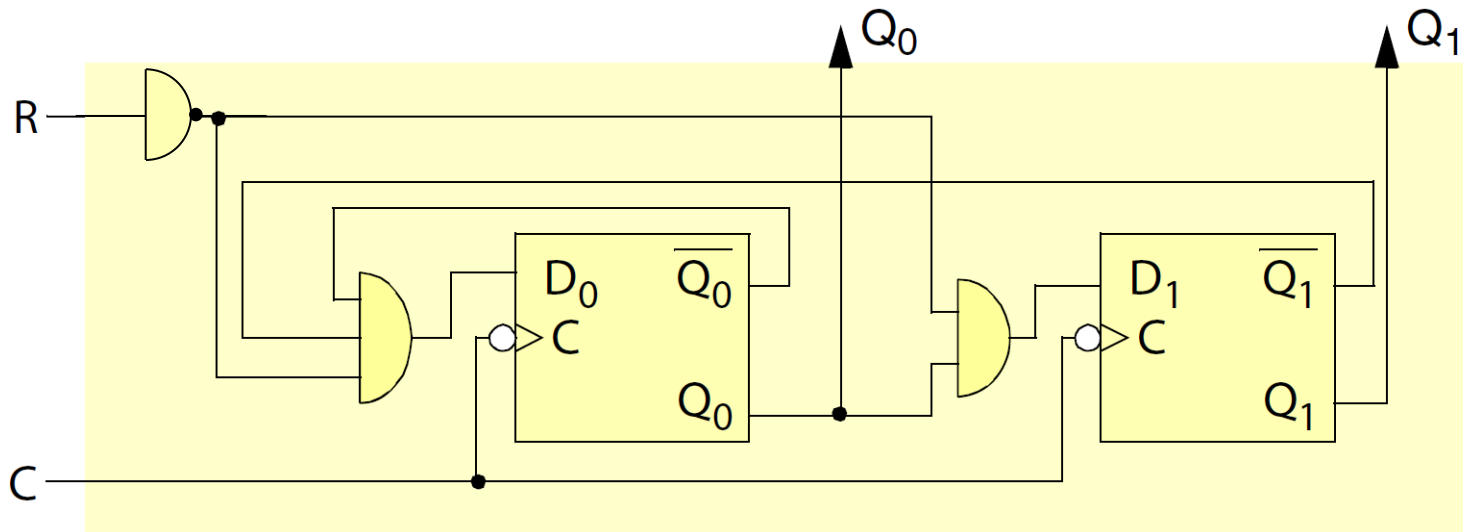
Schritt 5: Minimierung

$$- D_1 = Q_0 * \overline{R}$$

$$- D_0 = \overline{Q_0} * \overline{Q_1} * \overline{R}$$

Einfluss des Flip-Flop-Typs (8)

Schritt 6: Aufbau der Schaltung mit D-Flip-Flops



Einfluss des Flip-Flop-Typs (9)

☞ **Jedes getaktete Flip-Flop kann verwendet werden**

Unterschiede in der Schaltung

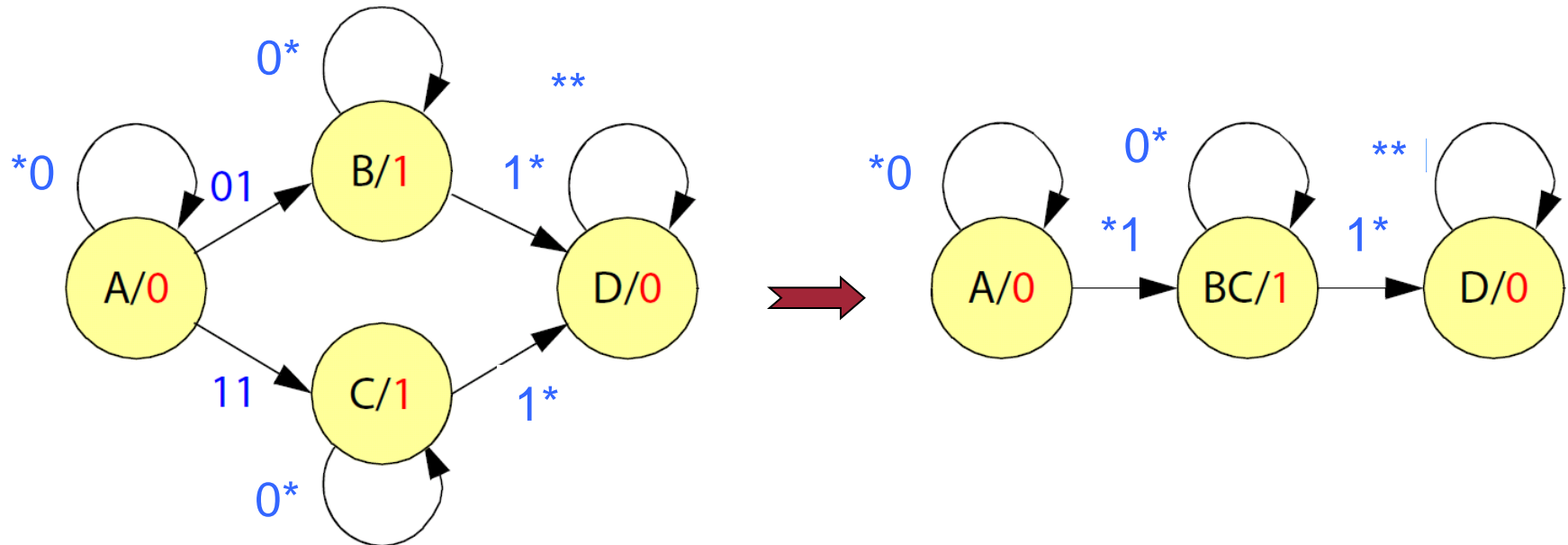
- JK-Flip-Flop tendiert zu besser zu minimierenden Ansteuerungsgleichungen
 - Viele *Don't cares* enthalten
 - Einfach anzusteuern Flip-Flops tendieren zu komplexen Ansteuerungsgleichungen
 - Z.B. D-Flip-Flops
 - Aber Beispiel zeigt: es gibt Ausnahmen
- ☞ **Wahl der Flip-Flops meist durch Verfügbarkeit von Bausteinen geprägt**

Zustandsreduktion von Automaten (1)

☞ **Weniger Flip-Flops durch Reduktion der Zustände**

Zustandsdiagramm Moore-Automat

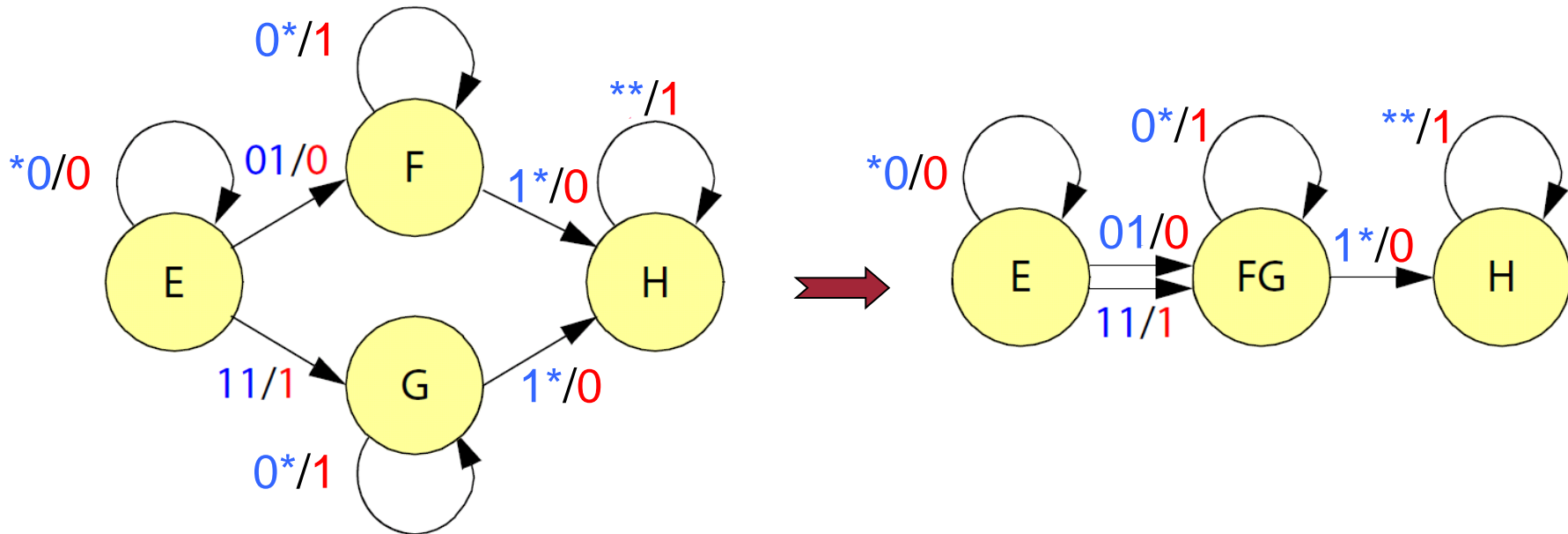
- Zusammenfassung von Zuständen mit gleicher Ausgabe und gleichen Folgezuständen



Zustandsreduktion von Automaten (2)

Zustandsdiagramm Mealy-Automat

- Zusammenfassung von Zuständen mit gleichen Folgezuständen und gleichen Ausgaben bei den Übergängen




Zusammenfassung (1)

Einleitung

- Annahme einer Gatterlaufzeit Δt
- Zyklen bzw. Rückkopplungen erlaubt, im Gegensatz zu Schaltnetzen
- Schaltwerke
- Rückkopplungen mit UND-, ODER bzw. NOR-Gattern

Zusammenfassung (2)

Flip-Flops

- RS-Flip-Flop: 2 rückgekoppelte NOR-Gatter
 - Eingänge: *Set*, *Reset*, Ausgang: gespeicherter Zustand Q
- Asynchrone Schaltwerke: verarbeiten geänderte Eingänge sofort
- Synchrone Schaltwerke: übernehmen Eingänge nur zu festen Zeiten
- *Master-Slave* Flip-Flop: zweistufiges Flip-Flop zum Vermeiden ungewollter asynchroner Rückkopplungen
- D-Flip-Flop (*Delay*): gibt Eingangswert D taktverzögert weiter
- Register: Speicher für bestimmte Anzahl von Binärwerten
- JK-Flip-Flop: erweitertes RS-Flip-Flop, das verbotene Eingabekombination $R = S = 1$ zulässt  Invertierung des Zustands
- T-Flip-Flop (*Toggle*): invertiert Zustand bei Eingang $T = 1$

Zusammenfassung (3)

Systematischer Schaltwerkdentwurf

- Weitgehend gleicher Ablauf für Moore- und Mealy-Automaten
 - Zustandsdiagramm bzw. Zustandstabelle
 - Binäre Zustandskodierung, binäre Zustandstabelle
 - Auswahl eines Flip-Flop-Typs, Flip-Flop-Ansteuerung in Zustandstabelle
 - Wahrheitstabelle für Ausgabefunktionen
 - Minimierung von Ansteuerungs- und Ausgabefunktionen
 - Aufbau der Schaltung
- Einfluss des Flip-Flop-Typs
 - JK-Flip-Flops tendiert zu kleineren Ansteuerungsfunktionen, andere Flip-Flops zu eher größeren
- Zustandsreduktion: Zusammenfassen von Automaten-Zuständen mit gleichem Ausgabeverhalten und gleichen Folgezuständen