Digitaltechnik & Rechnersysteme

Flipflops, Automatenentwurf

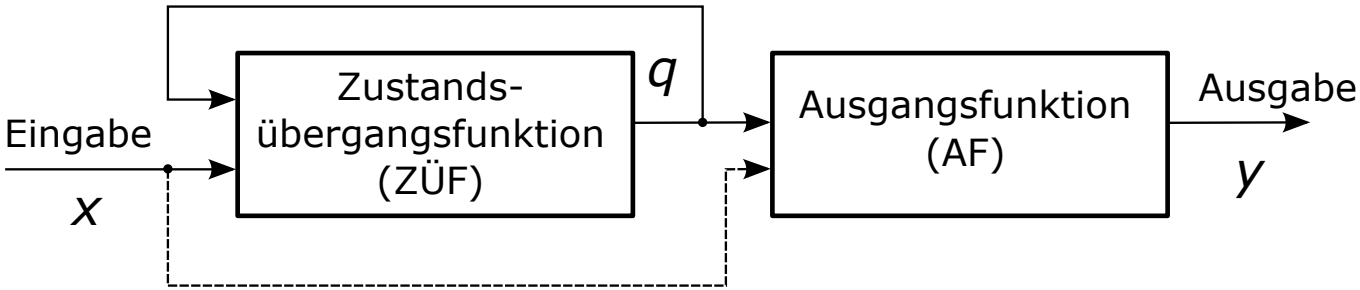
Martin Kumm

HOCHSCHULE FULDA
UNIVERSITY OF APPLIED SCIENCES

WiSe 2025/2026

Was bisher geschah...

1. Schaltwerksanalyse
2. Zustandsübergangs- u. Ausgangsfunktion
3. Zustands(übergangs-)graph
4. RS-Latch



Wiederholdung: RS-Latch

Übergangstabelle (verkürzt):

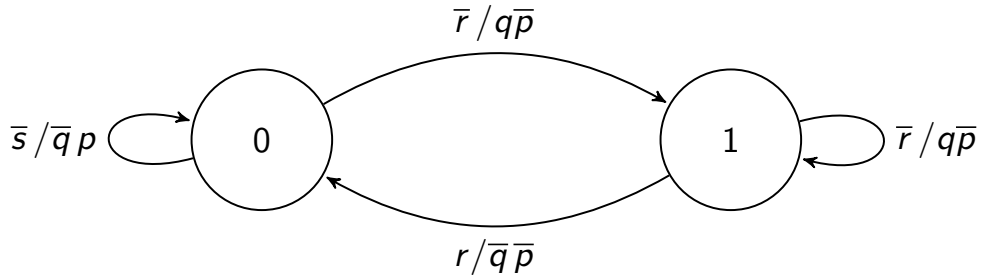
r	s	$q^{t+\tau}$	
0	0	q^t	speichern
0	1	1	setzen
1	0	0	rücksetzen
1	1	–	nicht zulässig

Wegen möglicher Instabilität ist $r = s = 1$ meist verboten!

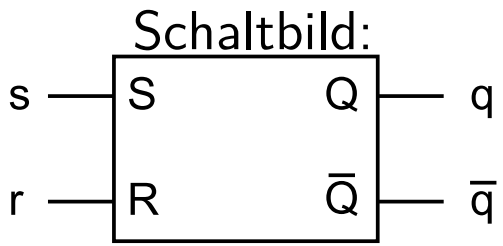
r = reset, s = set

Da $p = \overline{q}$ für stabile Zustände gilt, wird der 2. Ausgang als $\overline{q}$ bezeichnet

Das RS-Latch ist ein Element mit **zwei stabilen** Zuständen, auch **bistabiler** Speicher genannt.



Schaltbild:



Synchrones Schaltverhalten

Bisher hatten wir vorausgesetzt, dass Zustandsänderungen spontan mit der Änderung der Eingangsvariablen erfolgen.
Wir bezeichnen ein solches Schaltverhalten als **asynchron**.

D. h., beliebige Änderungen auf den Eingangsleitungen, z.B. durch Laufzeitunterschiede können sich in Form von Zustandsänderungen auswirken.

Um solche Effekte zu verhindern, wird ein **Takt**signal eingeführt, das durch regelmäßige (periodische) Wechsel zwischen 0 und 1 Zeitpunkte festlegt, zu denen Zustandswechsel erfolgen können.

Man spricht dann von **synchronem** Schaltverhalten.

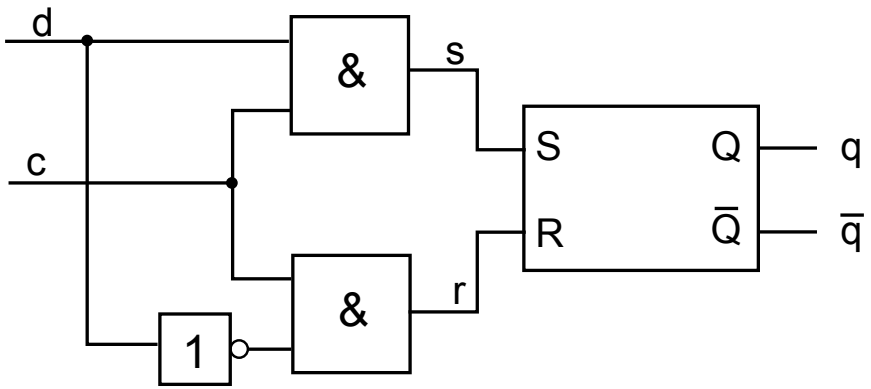
D-Latch mit Taktpegelsteuerung

Die zeitliche Übernahme wird durch einen Takt (*clock*, c) gesteuert

Set und reset werden aus einem Datensignal d und dem Takt c abgeleitet:

$$s = dc$$

$$r = \overline{d}\, c$$



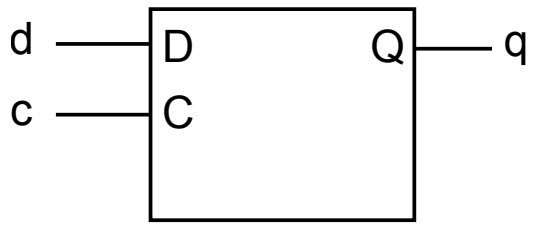
Somit wird bei $d = 1$ das Latch mit jedem Taktpuls gesetzt, für $d = 0$ mit jedem Taktpuls zurückgesetzt.

Übergangstabelle D-Latch

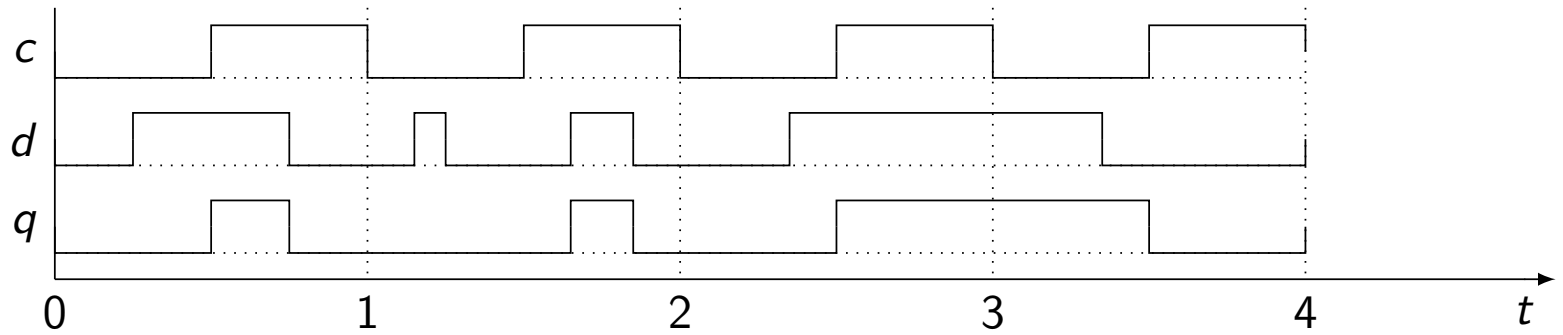
Übergangstabelle:

c	d	$q^{t+\tau}$	
0	0	q^t	speichern
0	1	q^t	speichern
1	0	0	rücksetzen
1	1	1	setzen

Schaltbild:



Verhalten des D-Latches



Die Hälfte der Periode ($c = 1$) ist Latch *transparent* (Ausgang=Eingang) die andere Hälfte ($c = 0$) wird gespeichert.

Problem:
Speicherung nur die halbe Periode, sollte möglichst lange speichern

➡ Lösung: Flankensteuerung

Taktpegel- und Taktflankensteuerung

Taktpegelsteuerung (Taktzustandssteuerung): Eingangsvariablen und Takt werden konjunktiv verknüpft
⇒ Eingänge wirken nur, während Takt 1 ist (relativ lange).
⇒ Speicherelemente werden als **Latch** bezeichnet

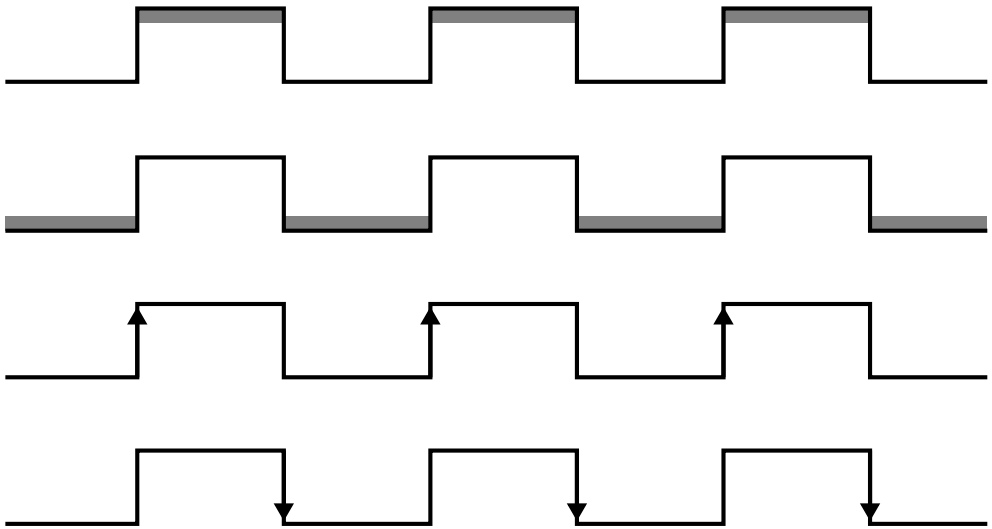
Taktflankensteuerung: Zustandsänderungen (Auswertung) nur bei den Flanken (0→1 Wechsel oder 1→ 0 Wechsel) möglich.
⇒ Speicherelemente werden als **Flipflop** bezeichnet

Pegel- und Flankensteuerung

Durch Invertieren des Taktsignals kann auch eine Reaktion auf den Low-Pegel des Taktes bzw. 1→0 Taktflanke erreicht werden.

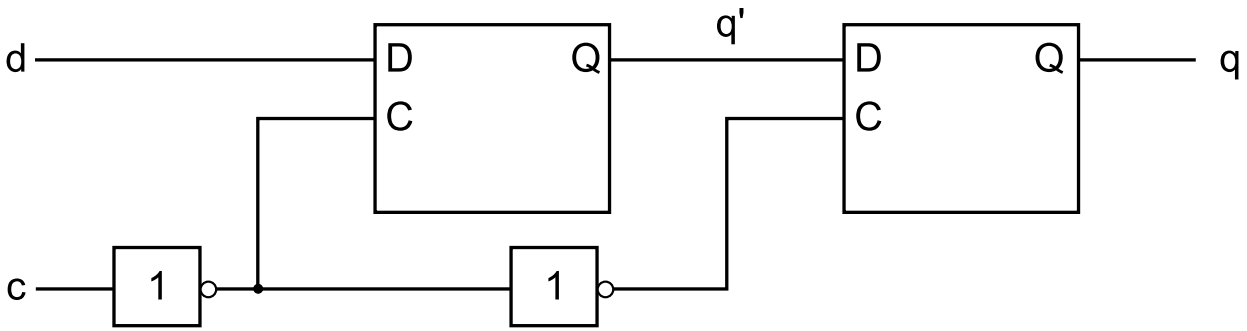
0→1 Übergänge des Takts werden als Vorderflanke, steigende Flanke oder positive Taktflanke bezeichnet.

1→0 Übergänge des Takts werden als Rückflanke, fallende Flanke oder negative Taktflanke bezeichnet.



Taktflankengesteuertes D-Flipflop

Durch Hintereinanderschalten zweier taktpegelgesteuerter D-Latches erhält man ein flankengesteuertes D-Flipflop:

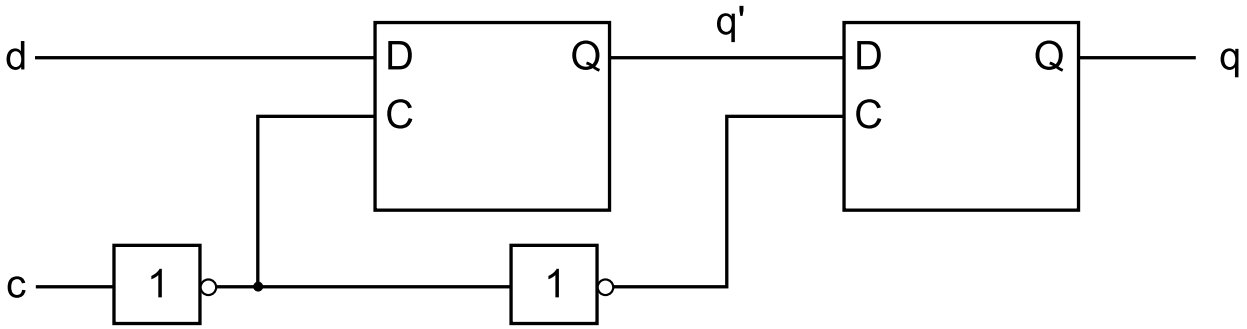


Takt= 0**:**
Das **erste Latch** ist **transparent**, das **zweite Latch speichert** .

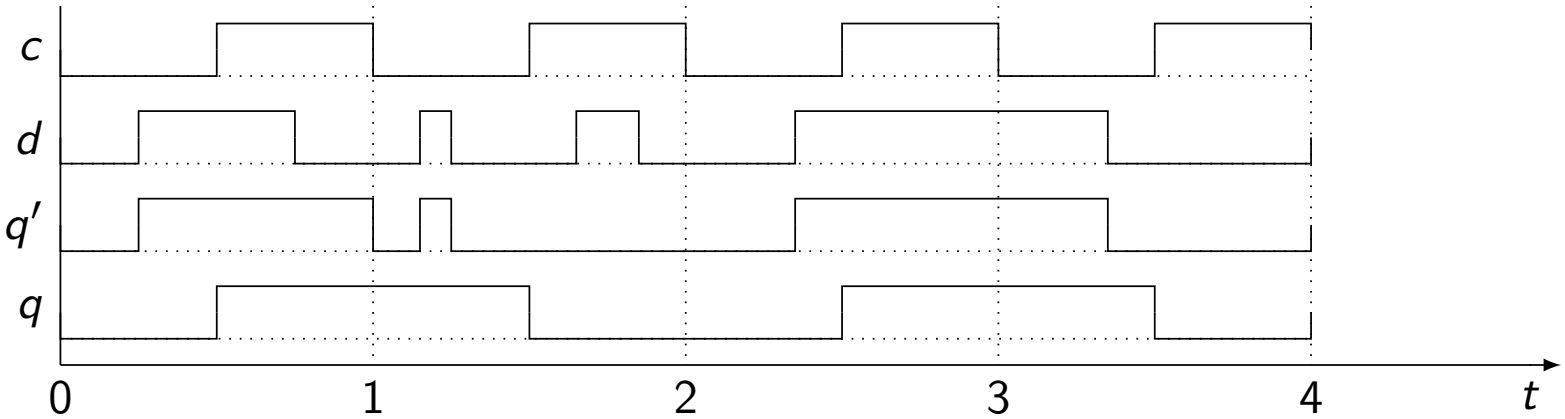
Takt= 1**:**
Das **erste Latch speichert**, das **zweite Latch** ist **transparent**.

➧ Ausgang für eine ganze Taktperiode stabil!

Taktflankengesteuertes D-Flipflop

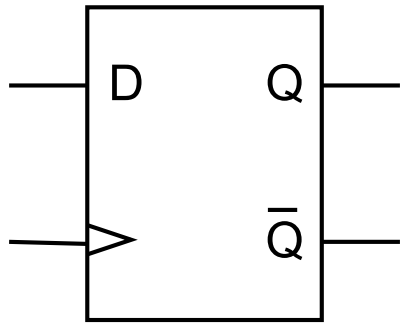


Beispiel-Timing:



Flankengesteuertes D-Flipflop

Schaltsymbol eines D-Flipflops:



Dabei zeigt das unausgefüllte Dreieck an, dass es sich um ein positiv flankengesteuertes Flipflop handelt. Eine Invertierung des Eingangs bzw. alternativ ein ausgefülltes Dreieck bezeichnen eine Steuerung mit der negativen Flanke.

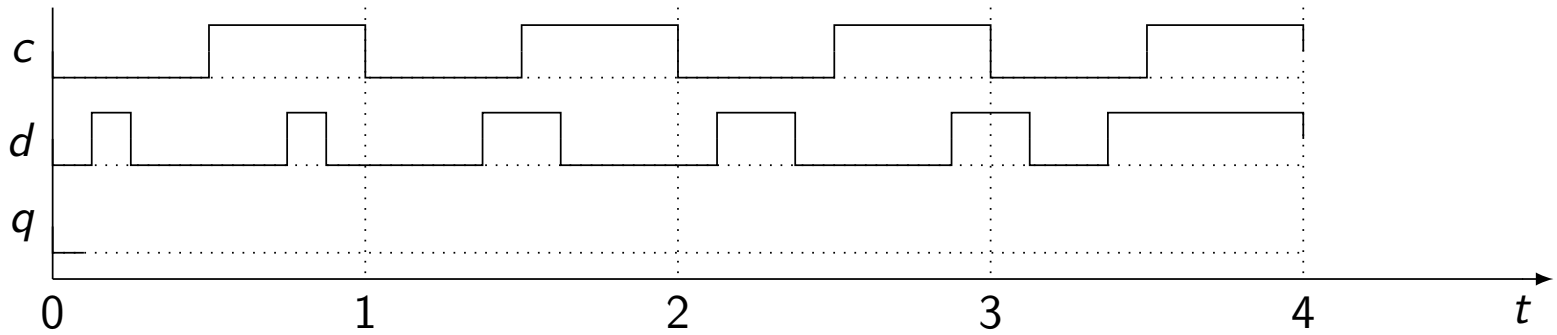
Funktion des D-Flipflops

Übergangsfunktion: $q^{t+1} = d$ für $c = 0 \rightarrow 1$

Übergangstabelle:

d	q^{t+1}
0	0
1	1

Vorlesungsaufgabe: Bestimmen Sie den zeitlichen Verlauf von Ausgang q!



JK-Flipflop

Beim RS-Flipflop ist Eingangskombination $r = s = 1$ nicht erlaubt

Durch Festlegung eines definierte Verhaltens für $r = s = 1$, nämlich der Wechsel des Zustands, erhält man aus dem RS-Flipflop ein **JK-Flipflop**.

Ein Gerücht besagt, das JK-Flipflop wurde möglicherweise nach **Jack Kilby** benannt (Physik-Nobelpreis 2000 für *„Beitrag zur Entwicklung des Integrierten Schaltkreises (IC)“*).

In Wirklichkeit beruht der Name auf einer willkürlichen „Durchnummerierung“ der Eingänge verschiedener Flipflops mit Buchstaben durch Dr. Eldred Nelson (Hughes Aircraft) um 1968.

Heute wird j als **Jump** und k als **Kill** bezeichnet.

Funktion des JK-Flipflops

Übergangstabelle:

j	k	q^{t+1}	
0	0	q^t	speichern
1	0	1	setzen (Jump)
0	1	0	rücksetzen (Kill)
1	1	$\overline{q}^t$	wechseln / toggeln

$$q^{t+1} = j\overline{q^t} + \overline{k}\,q^t \text{ für } c = 0 \to 1$$

Toggle-Flipflop

Ein JK-Flip-Flop mit $j = k = t$ führt zum Toggle-Flipflop: Für $t = 1$ ändert sich der Zustand (toggle), für $t = 0$ wird gespeichert.

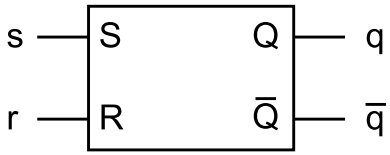
Übergangstabelle:

t	q^{t+1}	
0	q^t	speichern
1	$\overline{q^t}$	togglen

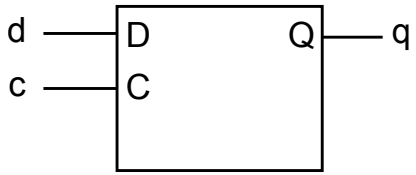
$$q^{t+1} = t\overline{q^t} + \overline{t}\, q^t \text{ für } c = 0 \to 1$$

Übersicht Flipflops

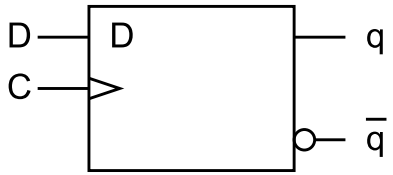
RS-Latch



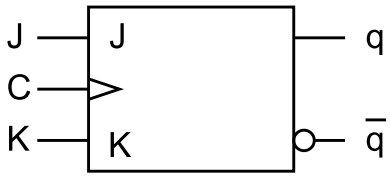
D-Latch (Taktpegelsteuerung)



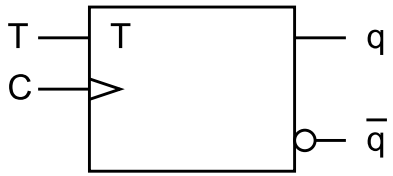
D-Flipflop (Taktflankensteuerung)



JK-Flipflop (Taktflankensteuerung)



T-Flipflop (Taktflankensteuerung)



Komplexere Schaltwerke

Eine wichtige Schaltungsklasse der Digitaltechnik bilden endliche Zustandsautomaten, im Englischen: *finite state machines* (FSM)

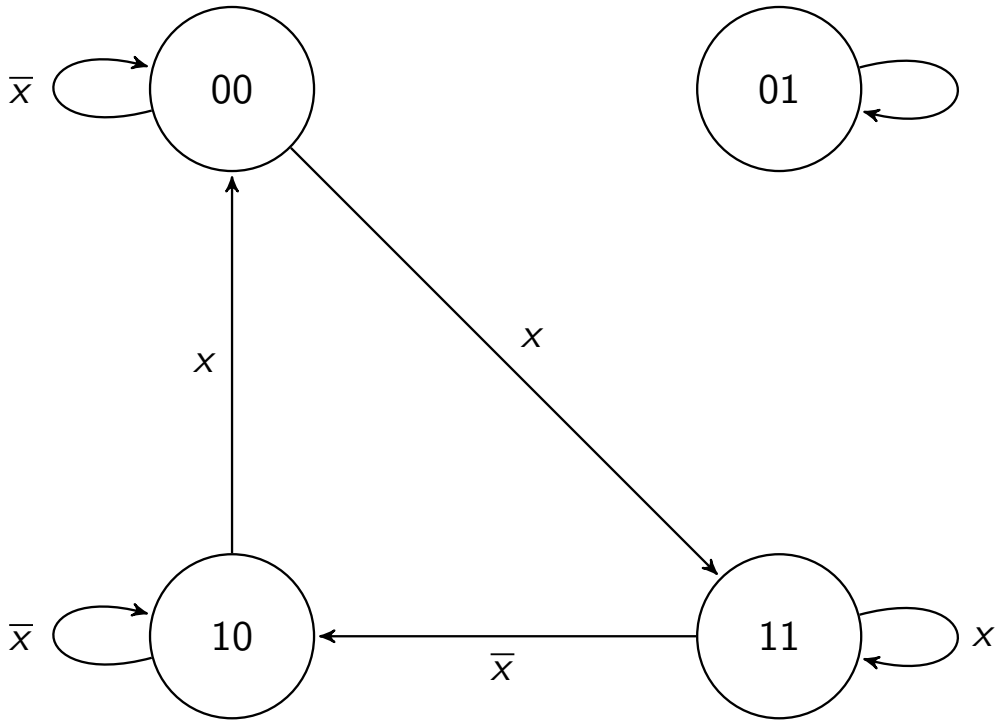
Einfache Zustandsautomaten mit wenigen Zuständen lassen sich durch Schaltwerke realisieren.

Bei mehreren Zustandsbits müssen Zustandswechsel verhindert werden bei denen sich mehr als ein Bit verändert. Ansonsten können fehlerhafte (Zwischen-)Zustände erreicht werden.

⇒ Diese lassen sich in asynchronen Schaltwerken nicht verhindern!

⇒ Für komplexere Automaten wird daher der Zustandsübergang immer mit FFs synchronisiert (⇒ synchroner Automat).

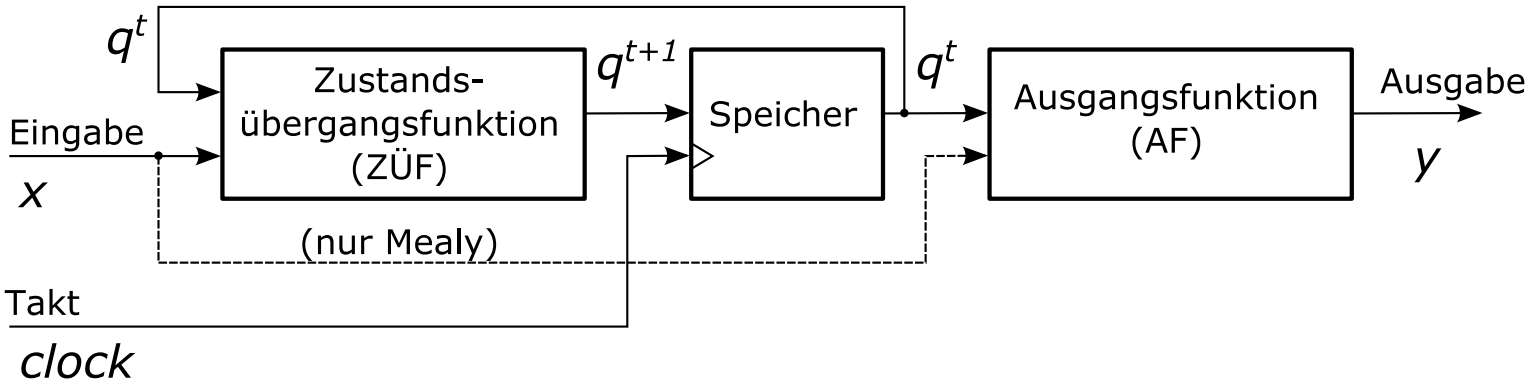
Beispiel



Geht der Automat von 00 → 11 über 01 oder 10?

⇒ Der Weg über 01 führt in eine Sackgasse!

Lösung: Synchrone Zustandsautomaten



Durch das Hinzufügen eines synchronen Speichers in die Rückkopplung wird die taktsynchrone Änderung des Zustandsvektors erzwungen.

Taktperiode muss hierbei größer als die Laufzeiten der Zustandsübergangsfunktionen sein.

Wahl des synchronen Speichers

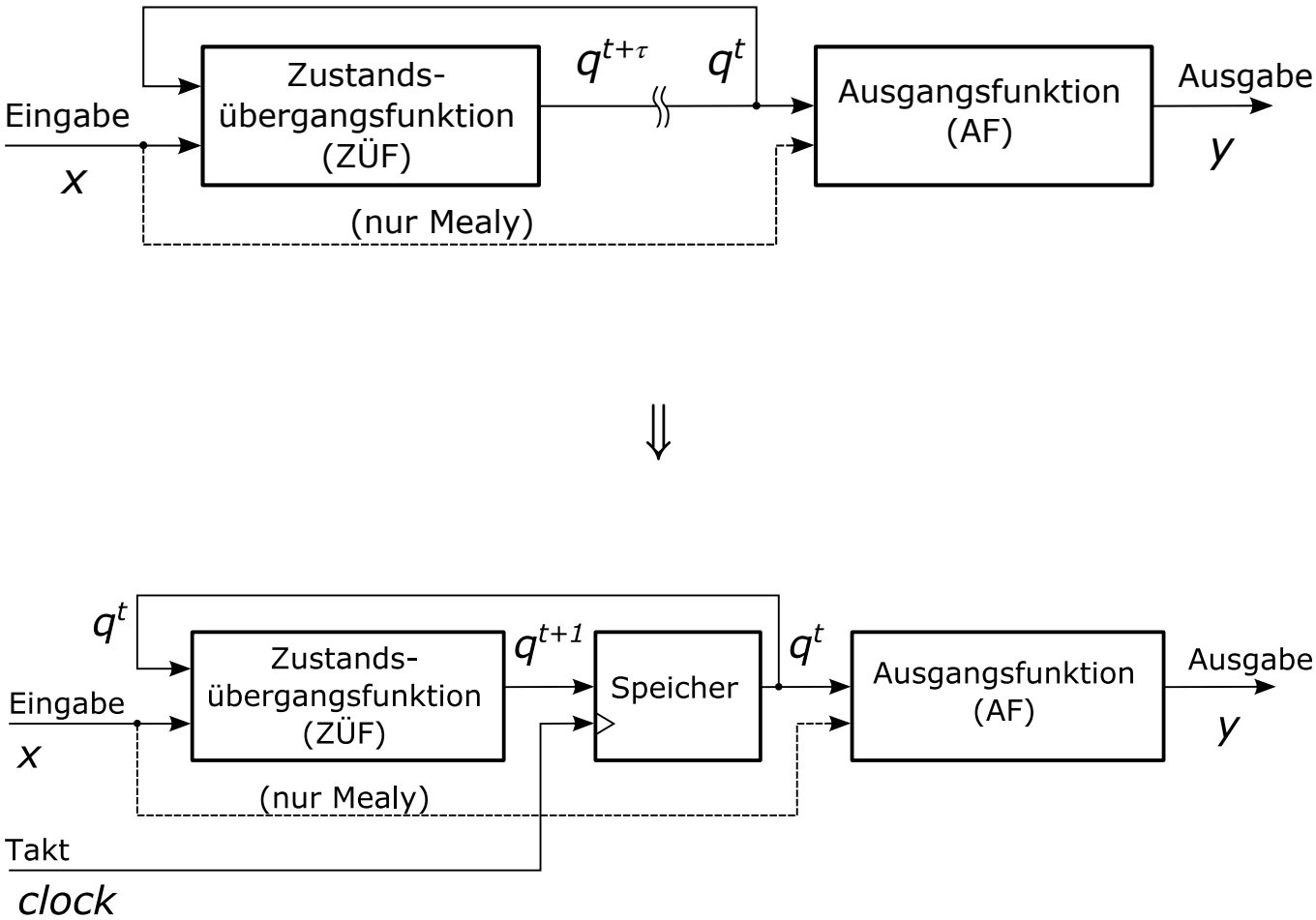
Der taktsynchrone Speicher kann prinzipiell beliebigen Typs sein

Jedoch müssen i.A. entsprechende Ansteuergleichungen ermittelt werden, bei denen bestimmt wird, wie die Eingänge gesetzt werden müssen um ein Zustandsbit zu speichern.

Im Falle von D-Flipflops als Speicher vereinfachen sich die Ansteuergleichungen drastisch (Identität).

Im Weiteren werden ausschließlich synchrone Automaten mit D-Flipflops betrachtet.

Asynchroner vs. synchroner Automat



Automatensynthese

Ausgehend von einer Problembeschreibung soll ein synchroner Automat entworfen werden, das diese Beschreibung realisiert. Ausgegangen wird meist von einer verbalen Aufgabenstellung.

Ablauf:

1. Festlegung der Ein- und Ausgangsvariablen
2. Festlegung ob Moore/Mealy-Automat
3. Bestimmung der Anzahl der Zustände
4. Ermittlung des Zustandsgraphen
5. Zustandskodierung
6. Bestimmung Zustandsübergangs- und Ausgangstabelle
7. Ermittlung (minimaler) Übergangs- und Ausgangsfunktionen
8. Erstellung des Schaltbildes aus Gattern und FFs

Beispiel: Eins-Detektor

Entwickeln Sie einen Automaten, der in einer Sequenz aus Binärziffern den Wechsel von einer Null auf eine Eins erkennt und daraufhin für einen Takt eine Eins ausgibt. Bei mehreren Einsen, die aufeinander folgen, soll der Automat lediglich für die erste Eins eine Eins ausgeben.

Beispiel:

Eingabe	0	0	1	0	0	1	1	0	1	0	1	1	1	1	0	0	0	0	1	0	0	1	1
Ausgabe	0	0	1	0	0	1	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	1	0

Zustandsvariablen und -kodierung

Der Zustand lässt sich durch die Zustandsvariablen q_i beschreiben:

$$q^t = (q_0, q_1, \ldots, q_i, \ldots, q_{n-1}), \quad q^t \in B^n$$

Mindestanzahl erforderlicher Zustandsvariabeln bei k Zuständen ist $n \geq \log_2 k, \quad n$ ganzzahlig.

Die Zuordnung der Zustände zu den Werten der Zustandsvariablen bezeichnet man als **Zustandskodierung**.

Die Zustandskodierung kann prinzipiell eine **beliebige** aber **eindeutige** Kodierung sein

Die Zustandskodierung hat Einfluss auf Komplexität und Laufzeit.

Zustandskodierung

Binärcodierung

- Jedem Zustand wird eine Binärzahl zugeordnet
- Kompakte Kodierung, erfordert wenige Flipflops
- Beispiel für 4 Zustände: 00, 01, 10, 11

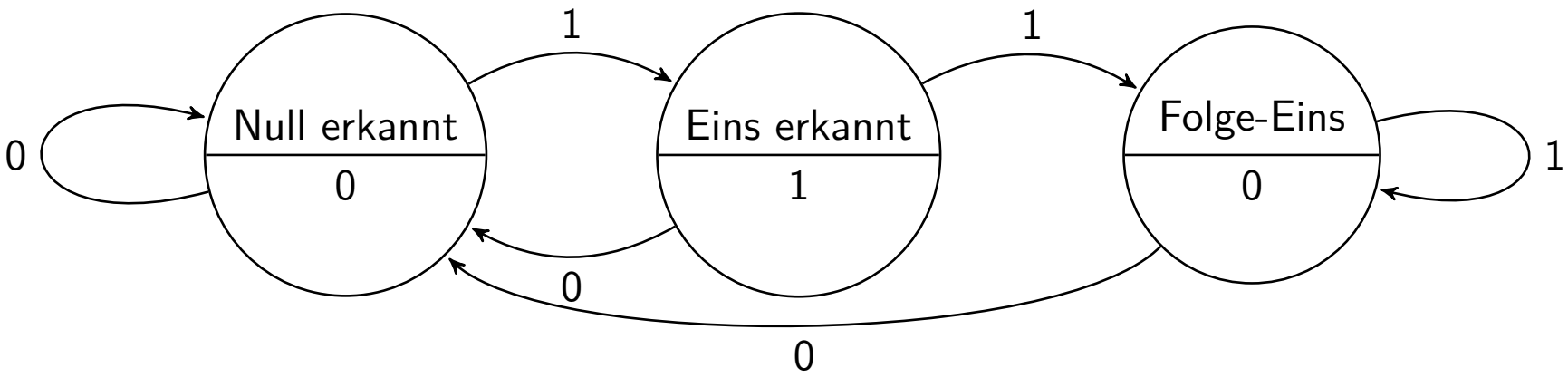
One-Hot Codierung

- Jedem Zustand wird ein Zustandsbit zugeordnet
- Zu jedem Zeitpunkt ist immer genau ein Zustandsbit »1«.
- Beispiel für 4 Zustände: 0001, 0010, 0100, 1000
- Benötigt mehr Flipflops
- Zustandsübergangs- und Ausgangsfunktionen vereinfachen sich oft

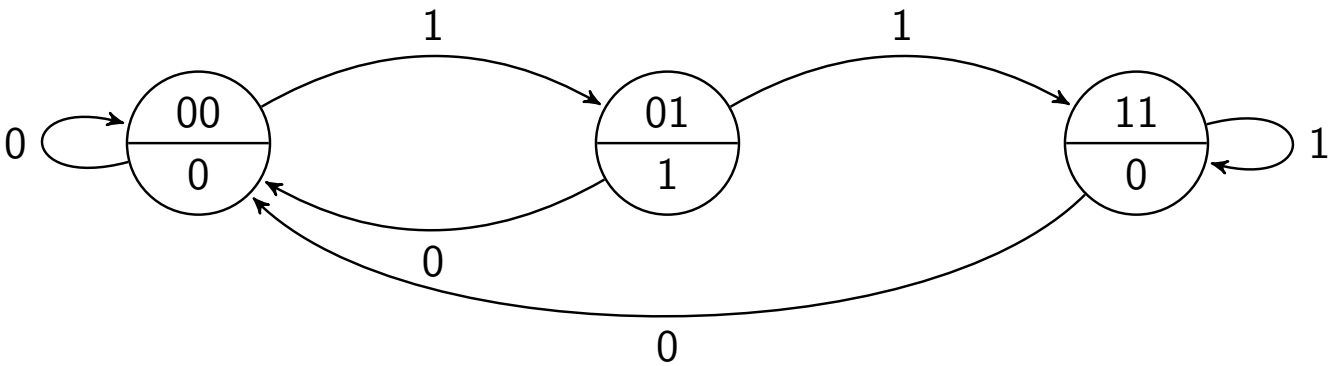
Zustandscodierung

Angewandte Informatik

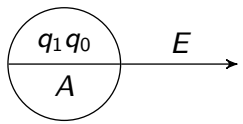
Zustandsdiagramm:



Zustandsdiagramm mit Zustandscodierung:

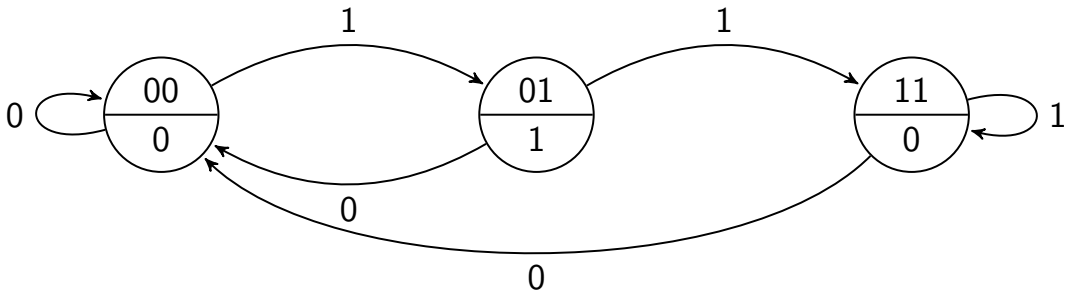


Notation:



Eins-Detektor als Moore-Automat

Vorlesungsaufgabe: Bestimmen Sie die Zustandsübergangs- und Ausgangstabelle

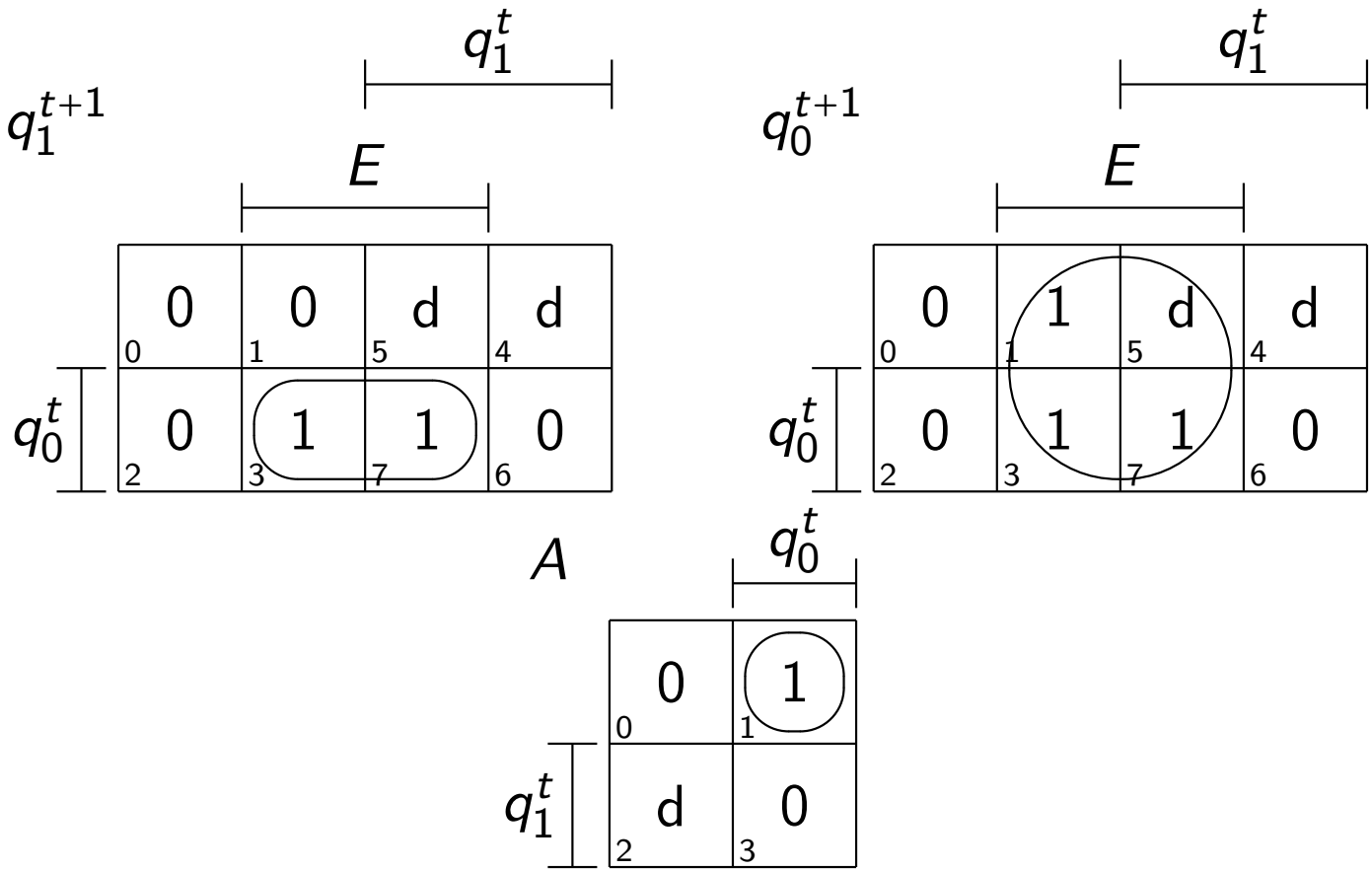


q_1^t	q_0^t	E	q_1^{t+1}	q_0^{t+1}
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

q_1^t	q_0^t	A
0	0	
0	1	
1	0	
1	1	

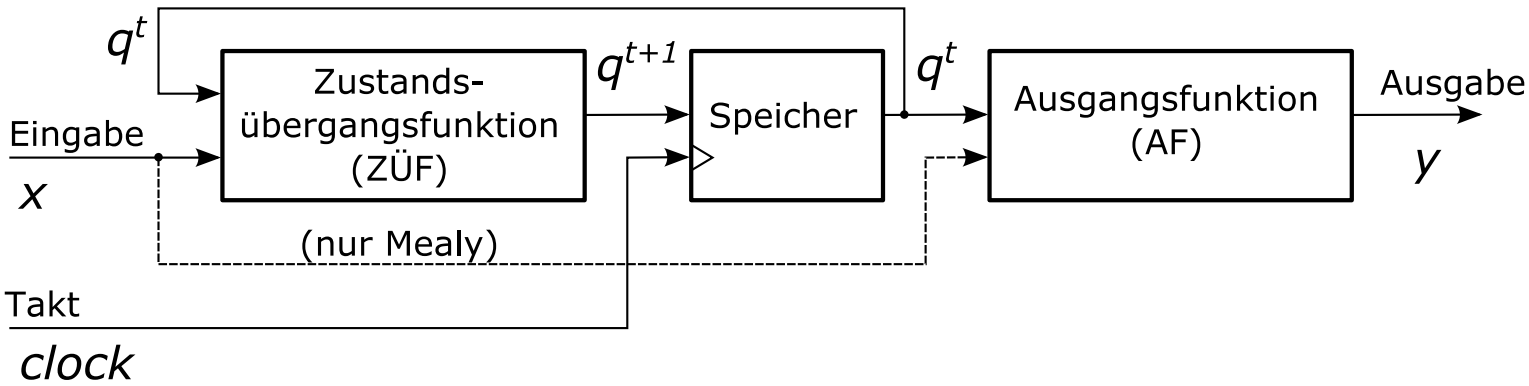
Eins-Detektor als Moore-Automat

Minimierung Zustandsübergangs- und Ausgangsfunktionen:



$$q_1^{t+1} = q_0^t E, \quad q_0^{t+1} = E, \quad A = \overline{q_1^t}\, q_0^t$$

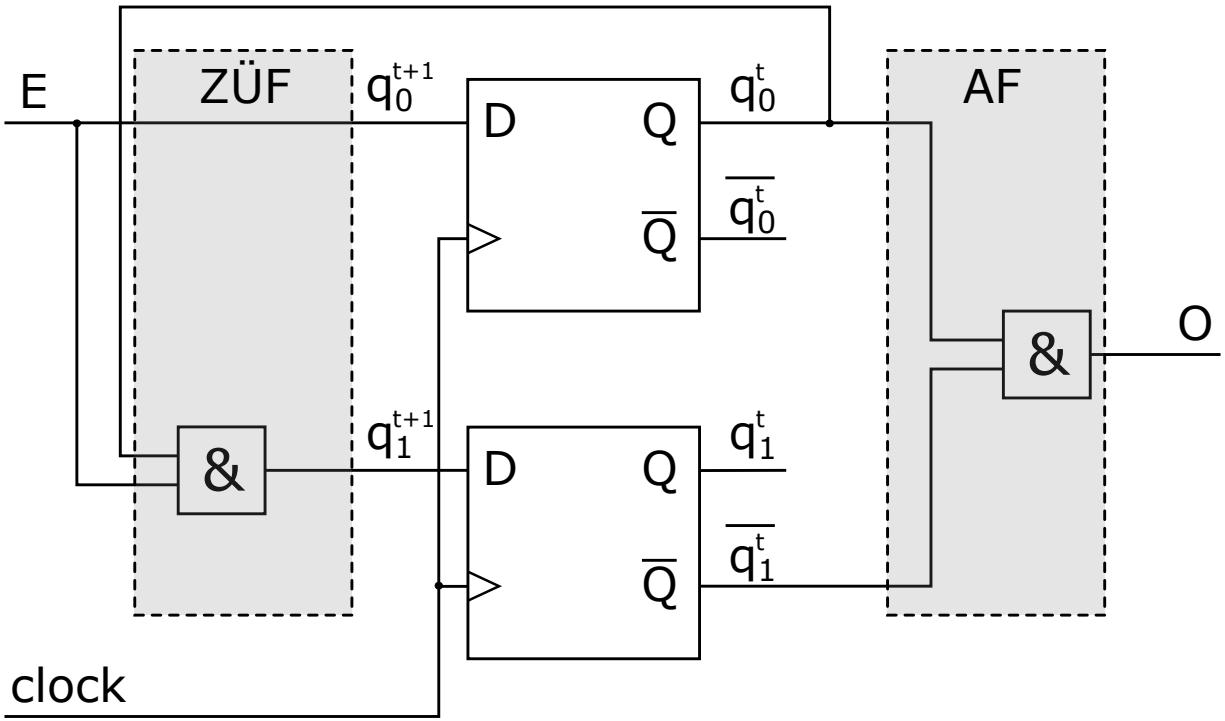
Wdh.: Synchroner Zustandsautomat



Eins-Detektor als Moore-Automat

Angewandte Informatik

Schaltbild:

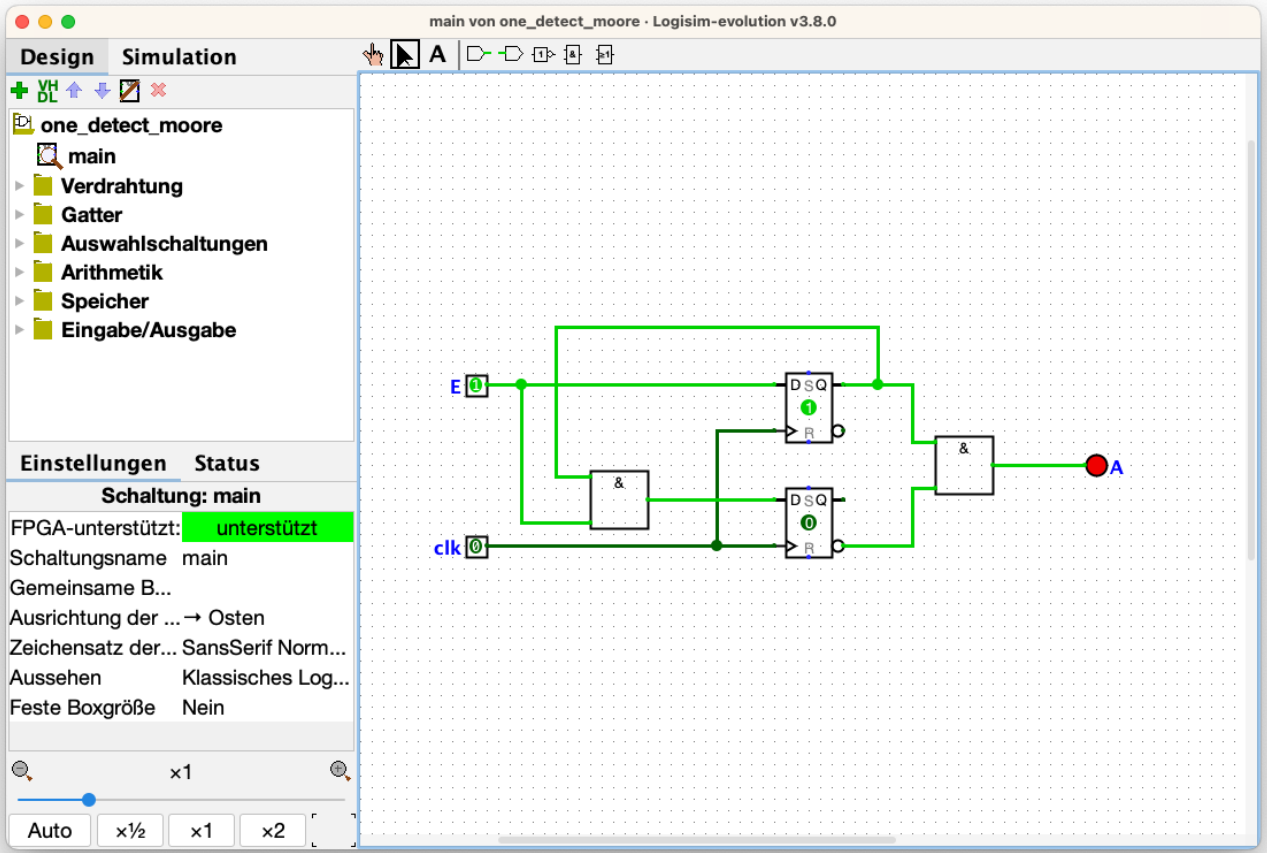


ZÜF: Zustandsübergangsfunktion

AF: Ausgangsfunktion

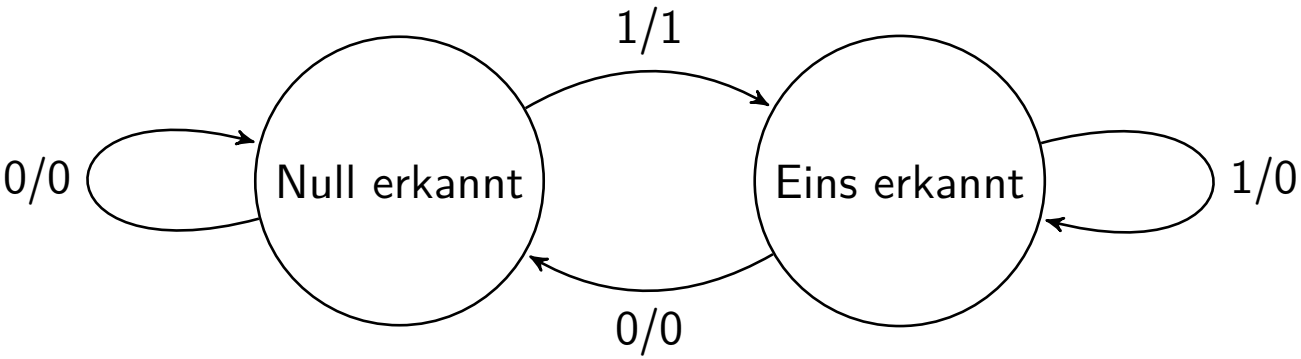
Simulation: Eins-Detektor (Moore)

Bei Automaten bietet sich die Simulation an um den Ablauf besser zu verstehen:

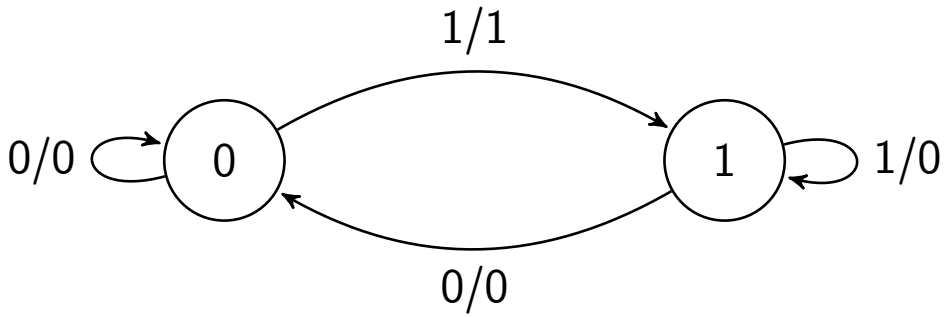


Eins-Detektor als Mealy-Automat

Zustandsdiagramm:



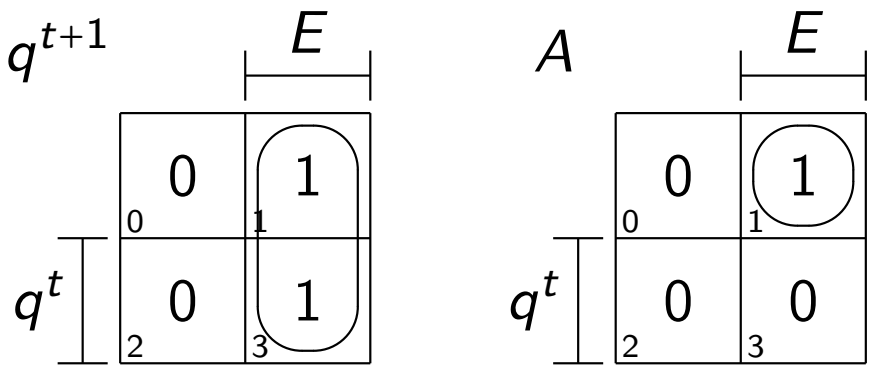
Zustandsdiagramm mit Zustandscodierung:



Eins-Detektor als Mealy-Automat

Zustandsübergangs- und Ausgangstabelle + Minimierung

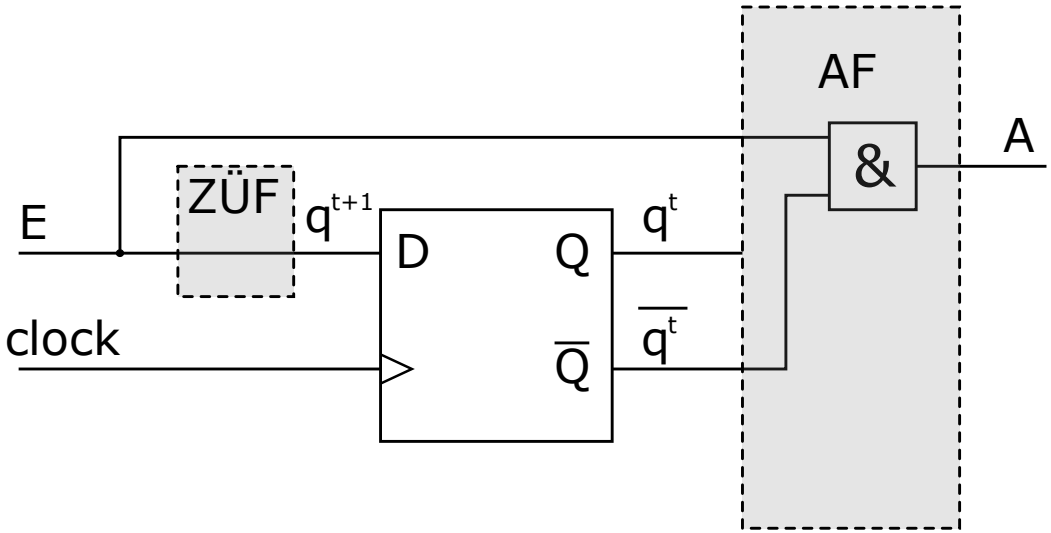
q^t	E	q^{t+1}	A
0	0	0	0
0	1	1	1
1	0	0	0
1	1	1	0



$$q^{t+1} = E,\ o = \overline{q^t}\, E$$

Eins-Detektor als Mealy-Automat

Schaltbild:

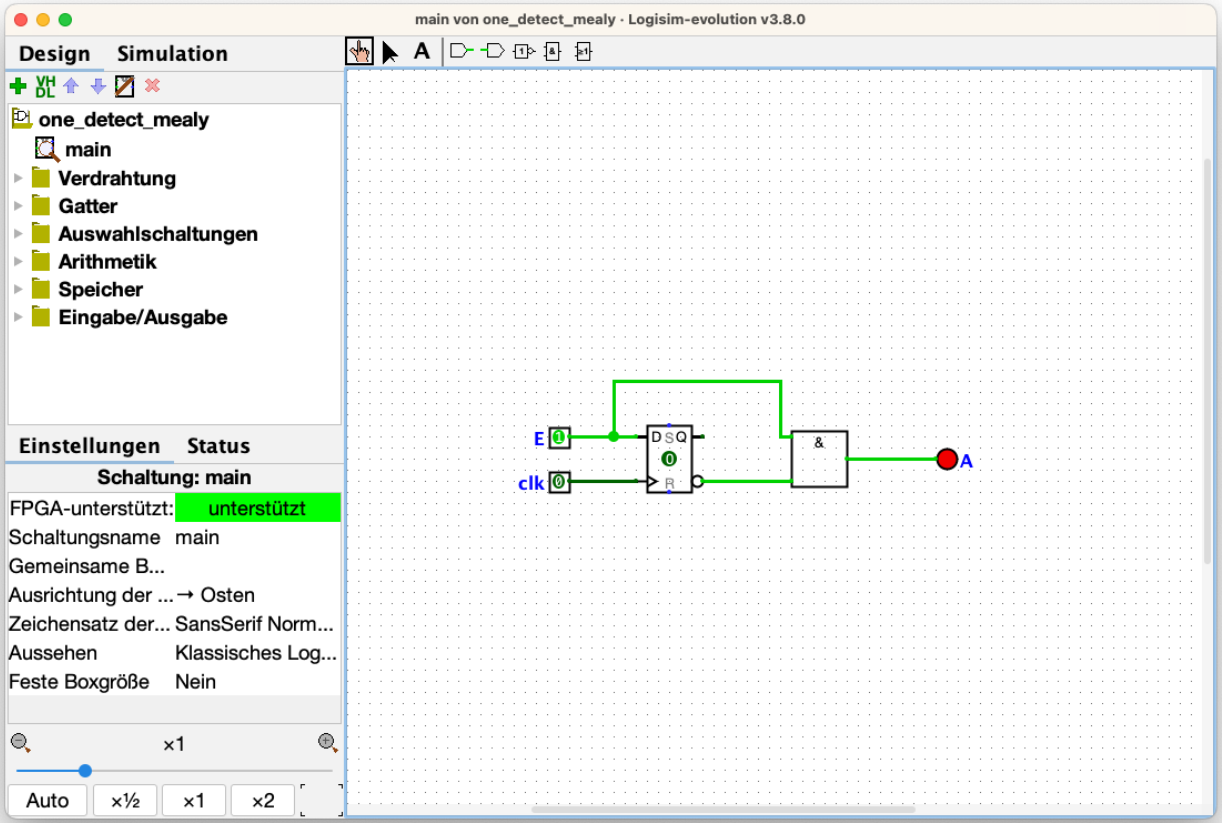


ZÜF: Zustandsübergangsfunktion

AF: Ausgangsfunktion

Simulation: Eins-Detektor (Mealy)

Bei Automaten bietet sich die Simulation an um den Ablauf besser zu verstehen:



Anmerkungen

Die Automatensynthese bietet ein mächtiges Werkzeug um Systeme mit Gedächtnis zu entwerfen

Die Automatensynthese ist ein kreativer Prozess. D.h. oft sind mehrere Iterationen notwendig um zur finalen Lösung zu kommen

Moore- und Mealy-Automaten führen je zu anderen Lösungen

Moore-Automaten benötigen oft mehr Zustände als Mealy-Automaten

Besinnliche Weihnachtsfeiertage
und die besten Wünsche für
ein gesundes und erfolgreiches
Jahr 2025!