

# Digitaltechnik & Rechnersysteme

## Automatenentwurf

Martin Kumm

**Hochschule Fulda**  
University of Applied Sciences



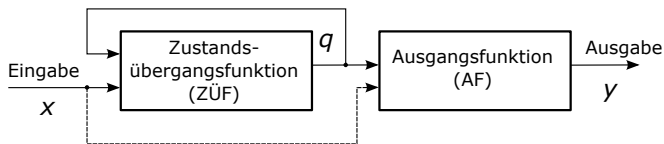
Angewandte Informatik

WiSe 2023/2024

# Was bisher geschah...



- 1 Schaltwerksanalyse
- 2 Zustandsübergangs- u. Ausgangsfunktion
- 3 Zustands(übergangs-)graph
- 4 RS-Latch
- 5 Taktpegelgesteuertes D-Latch
- 6 Taktflankengesteuertes D-Flipflop
- 7 JK-Flipflop und T-Flipflop



## Komplexere Schaltwerke

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

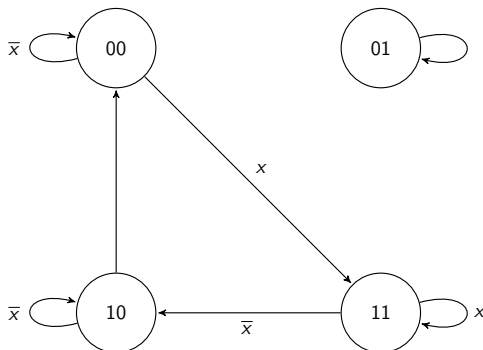
Einfache Zustandsautomaten mit wenigen Zuständen lassen sich durch Schaltwerke realisieren.

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).

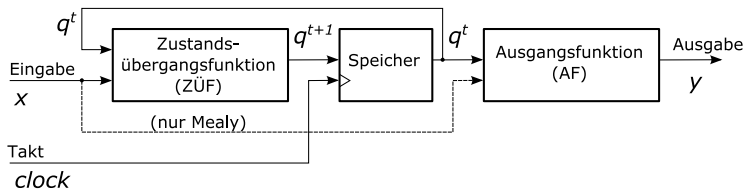
# Beispiel



Geht der Automat von 00  $\rightarrow$  11 über 01 oder 10?

$\Rightarrow$  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

---



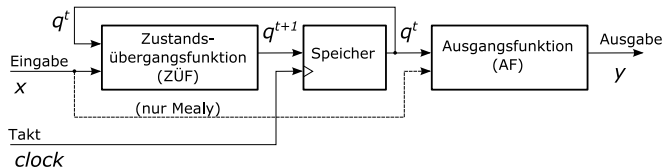
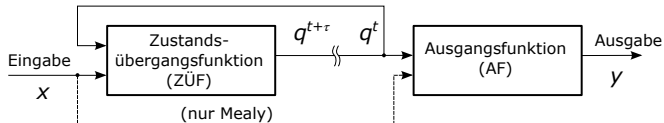
Der taktsynchrone Speicher kann prinzipiell beliebigen Typs sein

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		0		1		0		0		1		0

## Zustandsvariablen und -kodierung



Der Zustand lässt sich durch die Zustandsvariablen  $q_i$  beschreiben:

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

Mindestanzahl erforderlicher Zustandsvariablen bei  $k$  Zuständen ist  $n \geq \log_2 k$ ,  $n$  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

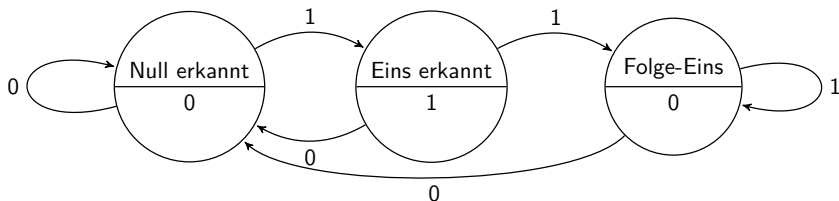
- Jedem Zustand wird eine **Binärzahl** zugeordnet
- Kompakte Kodierung, erfordert wenige Flipflops
- Beispiel für 4 Zustände: 00, 01, 10, 11

## One-Hot Codierung

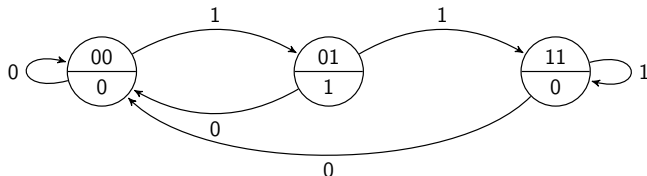
- Jedem Zustand wird ein **Zustandsbit** zugeordnet
- Zu jedem Zeitpunkt ist immer genau ein Zustandsbit »1«.
- Beispiel für 4 Zustände: 0001, 0010, 0100, 1000
- Benötigt mehr Flipflops
- Zustandsübergangs- und Ausgangsfunktionen vereinfachen sich oft

# Zustandskodierung

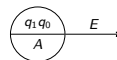
## Zustandsdiagramm:



## Zustandsdiagramm mit Zustandskodierung:



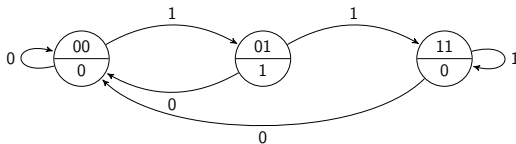
Notation:



# Eins-Detektor als Moore-Automat



## Vorlesungsaufgabe: Bestimmen Sie die Zustandsübergangs- und Ausgangstabelle

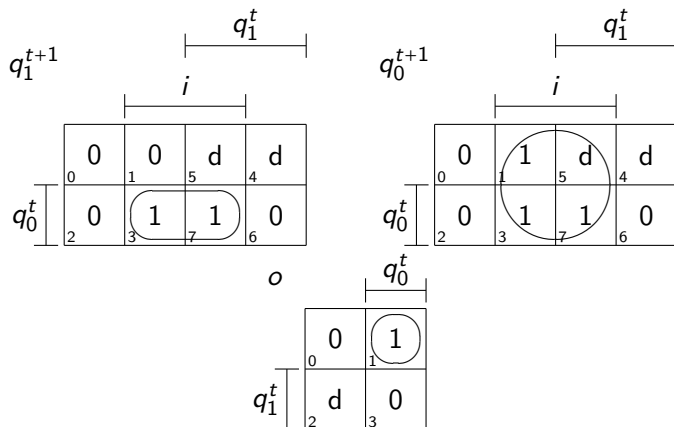


$q_1^t$	$q_0^t$	$i$	$q_1^{t+1}$	$q_0^{t+1}$
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

$q_1^t$	$q_0^t$	$o$
0	0	
0	1	
1	0	
1	1	

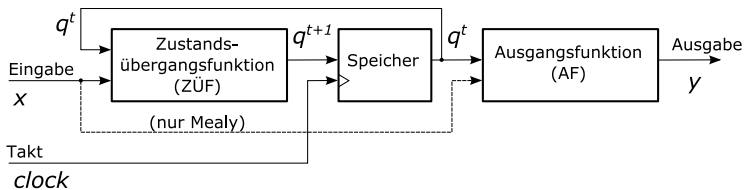
# Eins-Detektor als Moore-Automat

## Minimierung Zustandsübergangs- und Ausgangsfunktionen:



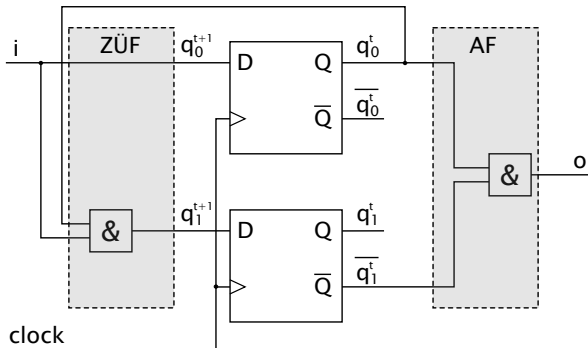
$$q_1^{t+1} = q_0^t i, \quad q_0^{t+1} = i, \quad o = \overline{q_1^t} q_0^t$$

# Wdh.: Synchroner Zustandsautomat



# Eins-Detektor als Moore-Automat

## Schaltbild:



ZÜF: Zustandsübergangsfunktion

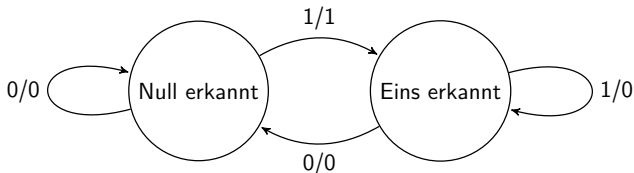
AF: Ausgangsfunktion



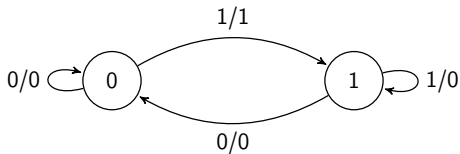
# Eins-Detektor als Mealy-Automat



## Zustandsdiagramm:



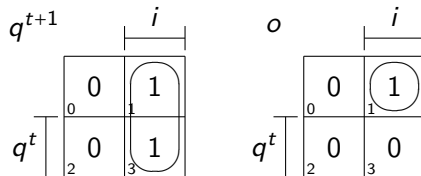
## Zustandsdiagramm mit Zustandskodierung:



# Eins-Detektor als Mealy-Automat

## Zustandsübergangs- und Ausgangstabelle + Minimierung

$q^t$	$i$	$q^{t+1}$	$o$
0	0	0	0
0	1	1	1
1	0	0	0
1	1	1	0

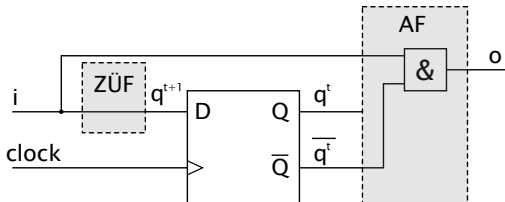


$$q^{t+1} = i, o = \overline{q^t} i$$

# Eins-Detektor als Mealy-Automat



## Schaltbild:



ZÜF: Zustandsübergangsfunktion

AF: Ausgangsfunktion

# Anmerkungen

---



Die Automaten-synthese bietet ein mächtiges Werkzeug um Systeme mit Gedächtnis zu entwerfen

Die Automaten-synthese 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

AI | Angewandte Informatik

D.h.  $Y=0,1,2,3,0,1,2,3,\dots$

Ablauf:

- 1 Festlegung der Ein- und Ausgangsvariablen ✓
  - Eingabe:  $e$ ,
  - Ausgabe:  $Y$  mit Werten zwischen 0 und 3
    - Binärcodiert, 2 Bit →  $y_1, y_0$
- 2 Festlegung ob Moore/Mealy-Automat ✓
  - Moore-Automat (aus Aufgabenstellung)
- 3 Bestimmung der Anzahl der Zustände ✓
  - 4 Zustände (→ min. 2 Zustandsbits)
- 4 Ermittlung des Zustandsgraphen

# Mod-4 Zähler mit enable

**Vorlesungsaufgabe:** Vervollständigen Sie das folgende Zustandsdiagramm zu einem mod-4 Zähler.



Notation:



# Mod-4 Zähler mit enable

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**

Prinzipiell beliebig, da hier jede Ausgabe eindeutig ist kann Zustandskodierung = Ausganskodierung gewählt werden:

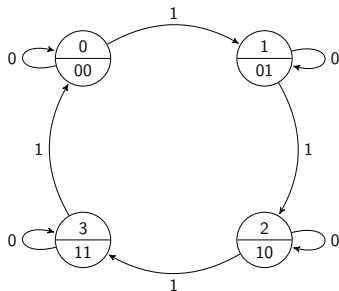
$$q_0 = y_0$$

$$q_1 = y_1$$

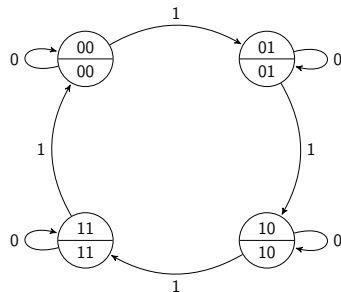


# Mod-4 Zähler mit enable

Zustandskodierung:



⇒



Notation:



Notation:

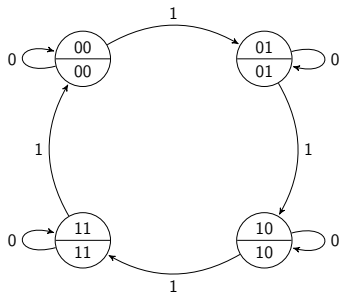


# Mod-4 Zähler mit enable

---

- 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

# Mod-4 Zähler mit enable



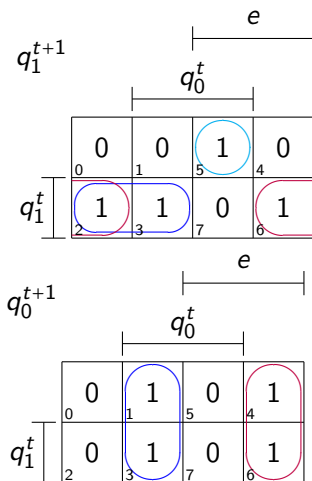
$e$	$q_1^t$	$q_0^t$	$q_1^{t+1}$	$q_0^{t+1}$
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

# Mod-4 Zähler mit enable

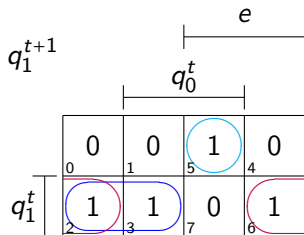
- ① Festlegung der Ein- und Ausgangsvariablen ✓
- ② Festlegung ob Moore/Mealy-Automat ✓
- ③ Bestimmung der Anzahl der Zustände ✓
- ④ Ermittlung des Zustandsgraphen ✓
- ⑤ Zustandskodierung ✓
- ⑥ Bestimmung Zustandsübergangs- und Ausgangstabelle ✓
- ⑦ Ermittlung (minimaler) Übergangs- und Ausgangsfunktionen

# Mod-4 Zähler mit enable

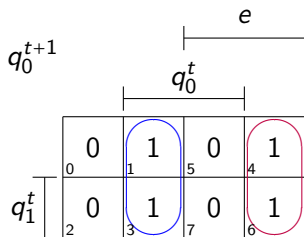
$e$	$q_1^t$	$q_0^t$	$q_1^{t+1}$	$q_0^{t+1}$
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0



# Mod-4 Zähler mit enable



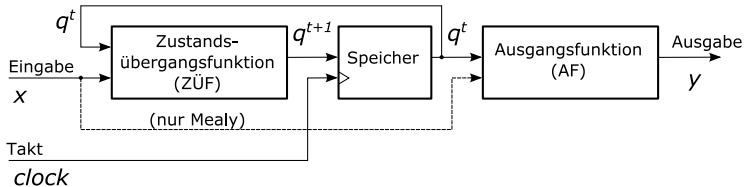
$$q_1^{t+1} = \bar{e} q_1^t + e q_0^t \bar{q}_1^t + \bar{q}_0^t q_1^t$$



$$q_0^{t+1} = \bar{e} q_0^t + e q_0^t$$

# Mod-4 Zähler mit enable

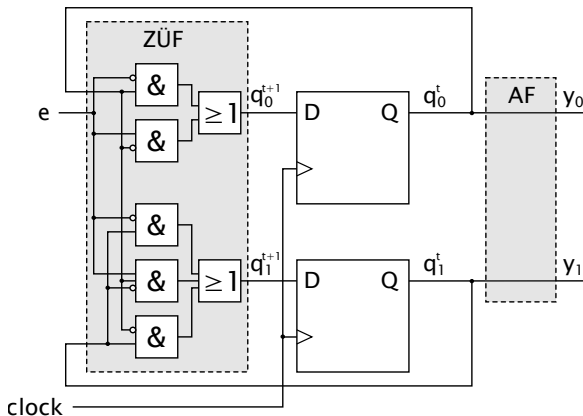
- ❶ Festlegung der Ein- und Ausgangsvariablen ✓
- ❷ Festlegung ob Moore/Mealy-Automat ✓
- ❸ Bestimmung der Anzahl der Zustände ✓
- ❹ Ermittlung des Zustandsgraphen ✓
- ❺ Zustandskodierung ✓
- ❻ Bestimmung Zustandsübergangs- und Ausgangstabelle ✓
- ❼ Ermittlung Übergangs- und Ausgangsfunktionen ✓
- ❽ Erstellung des Schaltbildes aus Gattern und FFs



# Mod-4 Zähler mit enable

$$\begin{aligned}\text{ZÜF: } q_0^{t+1} &= \bar{e} q_0^t + \bar{e} q_0^t \\ q_1^{t+1} &= \bar{e} q_1^t + e q_0^t \bar{q}_1^t + \bar{q}_0^t q_1^t\end{aligned}$$

$$\begin{aligned}\text{AF: } y_0 &= q_0^t \\ y_1 &= q_1^t\end{aligned}$$



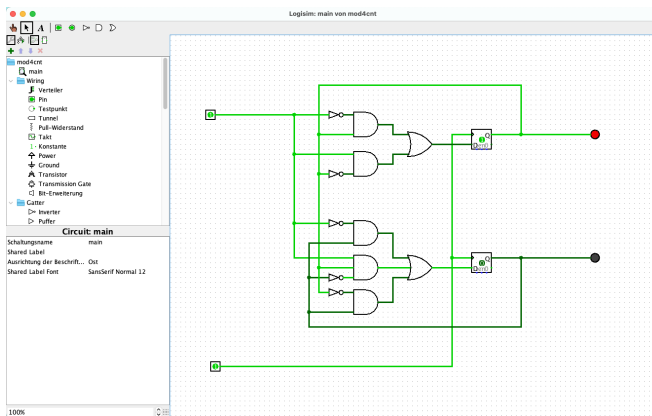


# Simulation: Mod-4 Zähler mit enable



Angewandte Informatik

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





Frohe  
Weihnachten