Digitaltechnik & Rechnersysteme

Automatenentwurf

Martin Kumm

HOCHSCHULE FULDA
UNIVERSITY OF APPLIED SCIENCES

WiSe 2025/2026

Was bisher geschah...

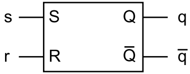
Angewandte Informatik

- Schaltwerk eines RS-Latch
- Taktpegelgesteuertes D-Latch
  - S und S aus D und Takt ableiten
  - Vorteil: Ein Datenbit, gemeinsamer Takt für viele Bits
  - Nachteil: Speicherung nur halbe Taktperiode
- Taktflankengesteuertes D-Flipflop
  - Zwei D-Latches die sich abwechseln
  - Vorteil: Speicherung über volle Taktperiode
- JK-Flipflop und T-Flipflop

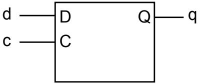
Übersicht Latches / Flipflops

AI Angewandte Informatik

RS-Latch



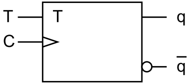
D-Latch (Taktpegelsteuerung)



D-Flipflop (Taktflankensteuerung)

JK-Flipflop (Taktflankensteuerung)

T-Flipflop (Taktflankensteuerung)



Komplexere Schaltwerke

Angewandte Informatik

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

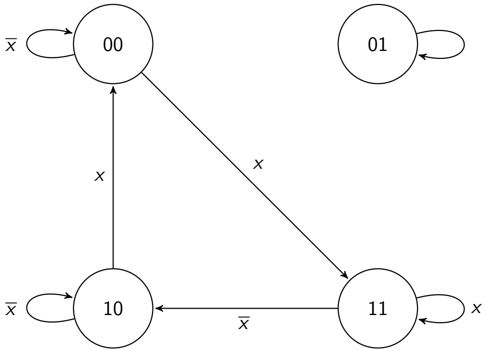
Einfache Zustandsautomaten mit wenigen Zuständen lassen sich durch Schaltwerke realisieren.

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

$\Rightarrow$ Diese lassen sich in asynchronen Schaltwerken nicht verhindern!

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

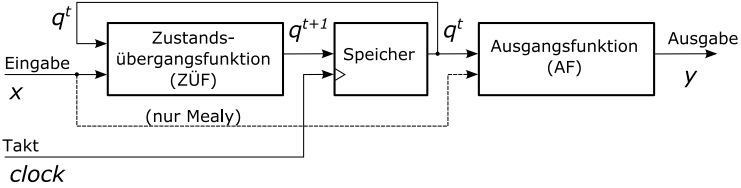
Beispiel



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.

Wahl des synchronen Speichers

Angewandte Informatik

Der taktsynchrone Speicher kann prinzipiell beliebigen Typs sein

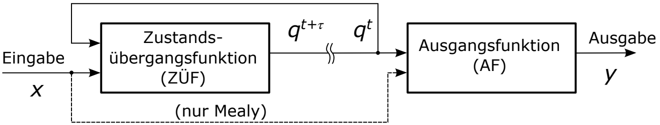
Jedoch müssen i.A. entsprechende Ansteuergleichungen ermittelt werden, bei denen bestimmt wird, wie die Eingänge gesetzt werden müssen um ein Zustandsbit zu speichern.

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

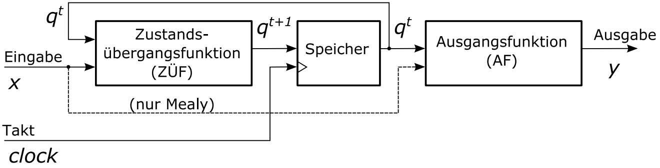
Im Weiteren werden ausschließlich synchrone Automaten mit D-Flipflops betrachtet.

Asynchroner vs. synchroner Automat

Angewandte Informatik



$\Downarrow$



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

Angewandte Informatik

Binärcodierung

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

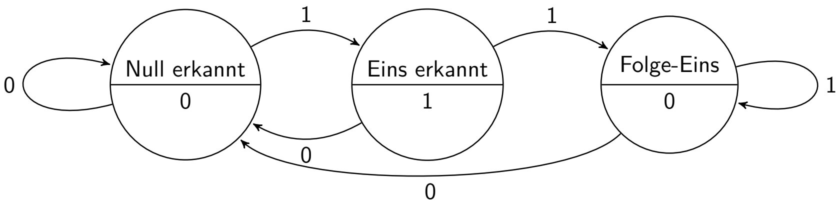
One-Hot Codierung

- 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

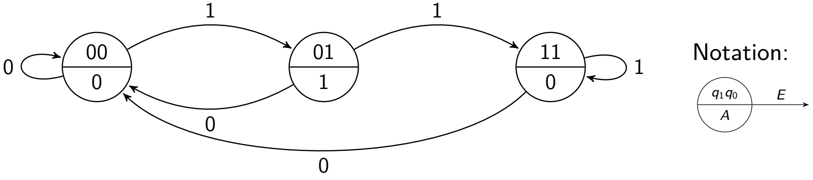
Zustandscodierung

Angewandte Informatik

Zustandsdiagramm:



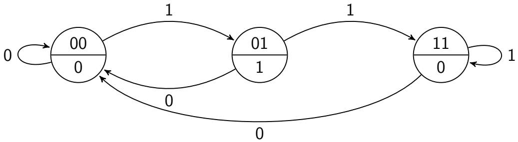
Zustandsdiagramm mit Zustandscodierung:



Eins-Detektor als Moore-Automat

Angewandte Informatik

Vorlesungsaufgabe: Bestimmen Sie die Zustandsübergangs- und Ausgangstabelle



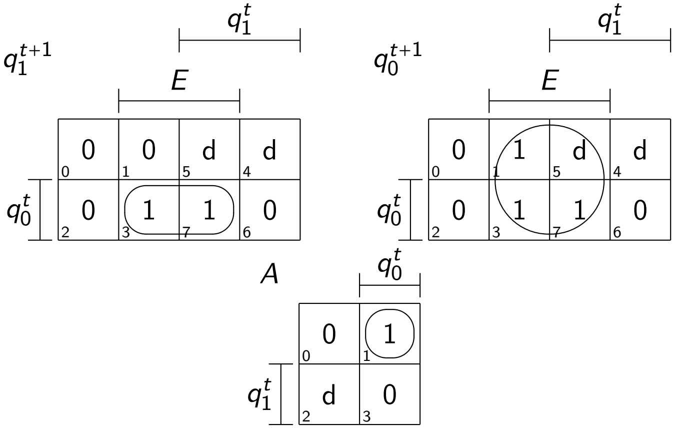
q_1^t	q_0^t	E	q_1^{t+1}	q_0^{t+1}
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

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

Eins-Detektor als Moore-Automat

Angewandte Informatik

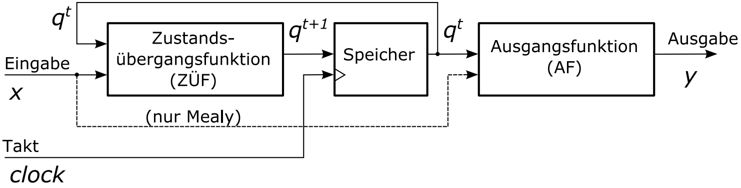
Minimierung Zustandsübergangs- und Ausgangsfunktionen:



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

Wdh.: Synchroner Zustandsautomat

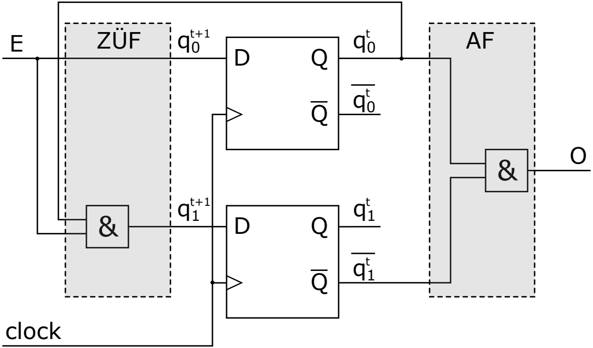
Angewandte Informatik



Eins-Detektor als Moore-Automat

Angewandte Informatik

Schaltbild:



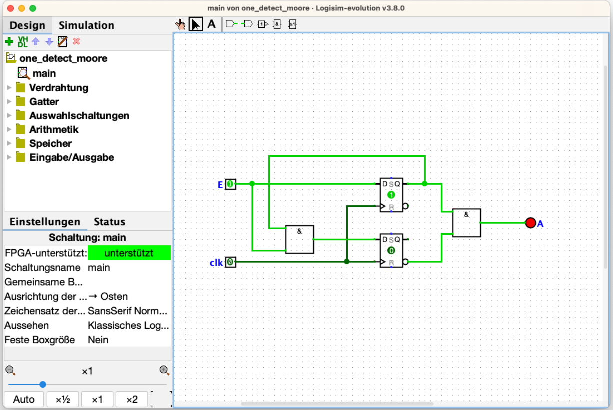
ZÜF: Zustandsübergangsfunktion

AF: Ausgangsfunktion

Simulation: Eins-Detektor (Moore)

Angewandte Informatik

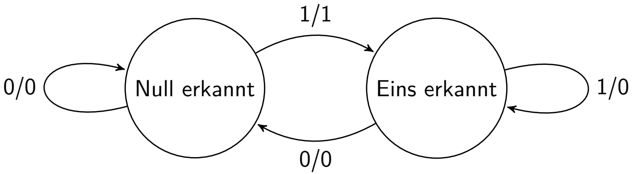
Bei Automaten bietet sich die Simulation an, um den Ablauf besser zu verstehen:



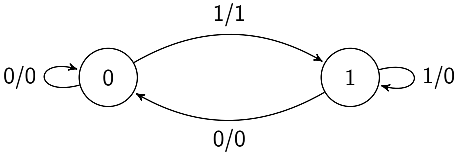
Eins-Detektor als Mealy-Automat

Angewandte Informatik

Zustandsdiagramm:



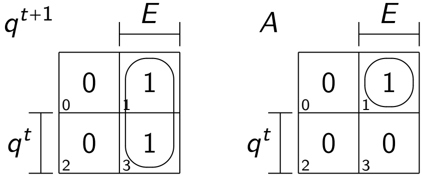
Zustandsdiagramm mit Zustandscodierung:



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

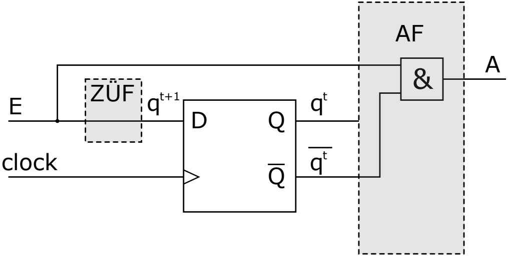


$$q^{t+1} = E, \; o = \overline{q^t}\, E$$

Eins-Detektor als Mealy-Automat

Angewandte Informatik

Schaltbild:



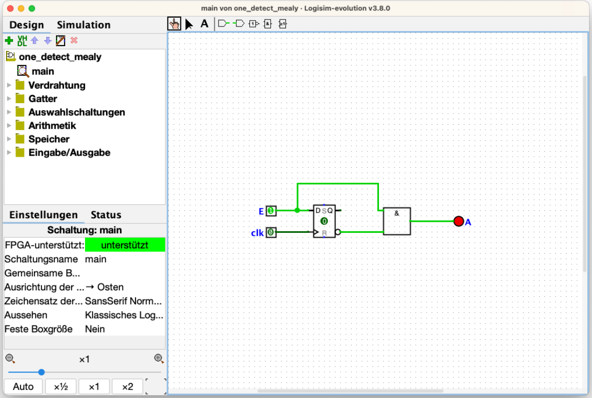
ZÜF: Zustandsübergangsfunktion

AF: Ausgangsfunktion

Simulation: Eins-Detektor (Mealy)

Angewandte Informatik

Auch hier bietet sich die Simulation an, um den Ablauf besser zu verstehen:



Anmerkungen

Angewandte Informatik

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

Mod-4 Zähler mit enable

Im Folgenden entwickeln wir zusammen einen sog. Modulo-4 (Mod-4) Zähler mit enable.

Dieser soll immer wenn $e = 1$ ist die Ausgabe Y um eins erhöhen.

Nach $Y = 3$ fängt dieser wieder von vorne an (daher der Name Modulo Zähler, $Y = (3 + 1) \bmod 4 = 0$)

D.h. Y=0,1,2,3,0,1,2,3,. . .

Mod-4 Zähler mit enable

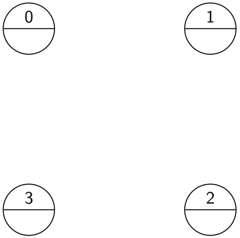
Angewandte Informatik

Ablauf:

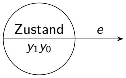
1. Festlegung der Ein- und Ausgangsvariablen ✓
   → Eingabe: e,
   → Ausgabe: Y mit Werten zwischen 0 und 3
   → Binärcodiert, 2 Bit → y_1, y_0
2. Festlegung ob Moore/Mealy-Automat ✓
   → Moore-Automat (aus Aufgabenstellung)
3. Bestimmung der Anzahl der Zustände ✓
   → 4 Zustände (→ min. 2 Zustandsbits)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

Angewandte Informatik

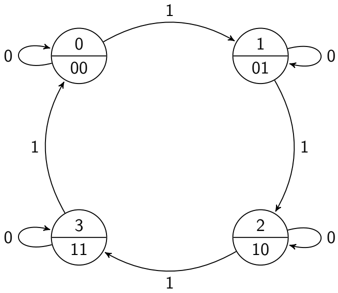
Ablauf:

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 = Ausgangskodierung gewählt werden:
   $q_0 = y_0$
   $q_1 = y_1$

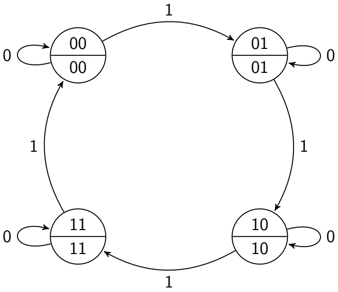
Mod-4 Zähler mit enable

Angewandte Informatik

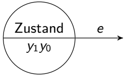
Zustandskodierung:



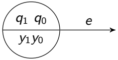
⇒



Notation:



Notation:



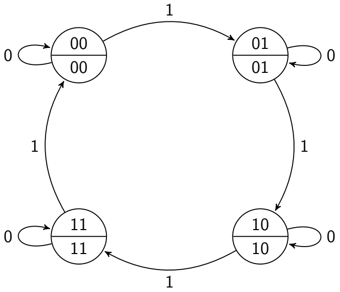
Mod-4 Zähler mit enable

Angewandte Informatik

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

Angewandte Informatik



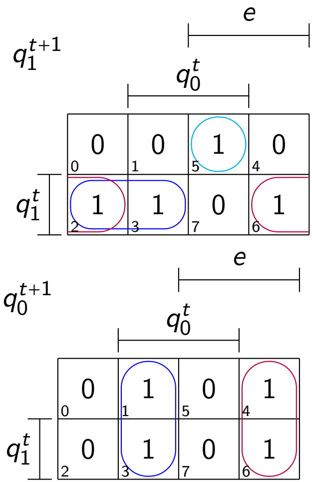
e	q_1^t	q_0^t	q_1^{t+1}	q_0^{t+1}
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

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

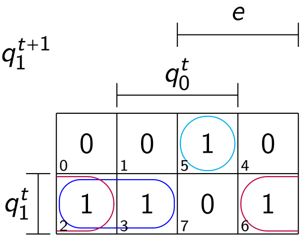
Mod-4 Zähler mit enable

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

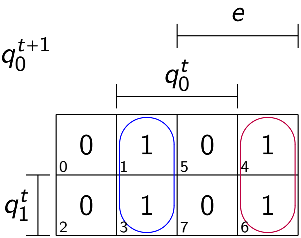


Mod-4 Zähler mit enable

Angewandte Informatik



$$q_1^{t+1} = \overline{e}\, q_1^t + e q_0^t \overline{q_1^t} + \overline{q_0^t}\, q_1^t$$

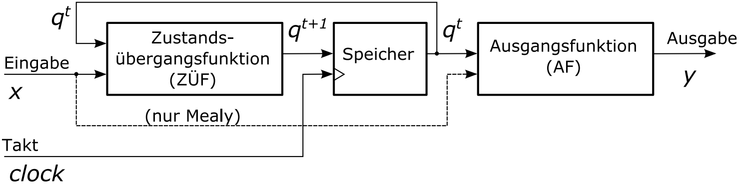


$$q_0^{t+1} = \overline{e}\, q_0^t + e\overline{q_0^t}$$

Mod-4 Zähler mit enable

Angewandte Informatik

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 Übergangs- und Ausgangsfunktionen ✓
8. Erstellung des Schaltbildes aus Gattern und FFs



Mod-4 Zähler mit enable

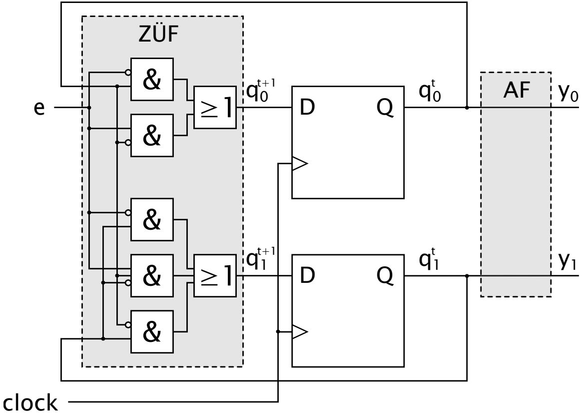
Angewandte Informatik

ZÜF: $q_0^{t+1} = \overline{e}\, q_0^t + e\overline{q_0^t}$

$q_1^{t+1} = \overline{e}\, q_1^t + e q_0^t \overline{q_1^t} + \overline{q_0^t}\, q_1^t$

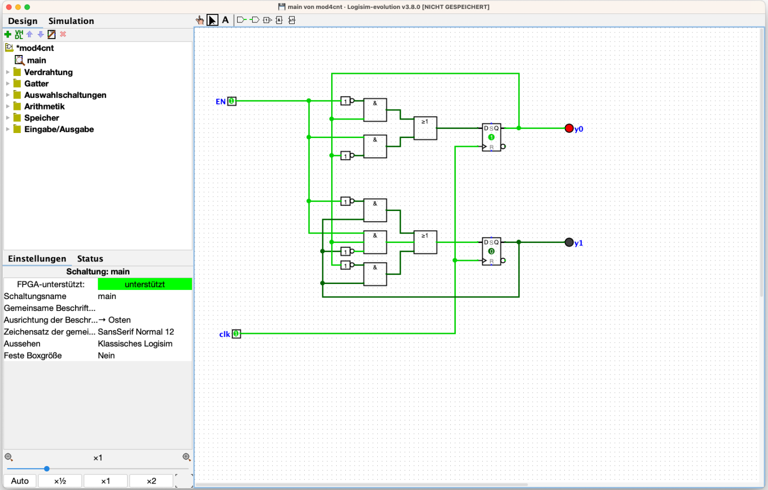
AF: $y_0 = q_0^t$

$y_1 = q_1^t$



Simulation: Mod-4 Zähler mit enable

Angewandte Informatik



Frohe
Weihnachten