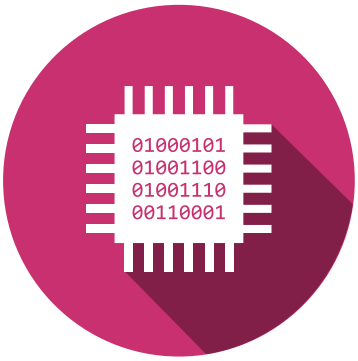
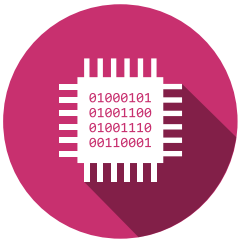


Zustandmaschinen (FSM)

Vorlesung Digitales Design

Hes·so VALAIS WALLIS
Haute Ecole d'Ingénierie
Hochschule für Ingenieurwissenschaften

Orientierung: Informatik und Kommunikationssysteme (ISC)
Kurs: Digitales Design (DiD)
Verfasser: Christophe Bianchi, François Corthay, Pierre Pompili, Silvan Zahno
Datum: 25. August 2022
Version: v2.1



Inhaltsverzeichnis

1 Einführung **2**

2 Logische Synchronsysteme **3**
2.1 Definition: logisches Synchronsystem . . . 3
2.2 Taktsignal . . . 3

3 Moore-Maschinen **4**
3.1 Aufbau . . . 4
3.1.1 Definition: Eingangszustand . . . 4
3.1.2 Definition: interner Zustand . . . 4
3.1.3 Definition: Ausgangszustand . . . 4
3.2 Wahrheitstabelle der kombinatorischen Blöcke . . . 4
3.3 Zustandsgraph . . . 5
3.4 Definition: stabiler Gesamtzustand . . . 6

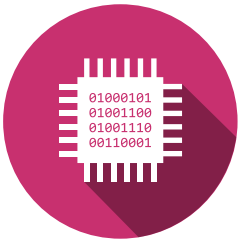
4 Mealy-Maschinen **7**
4.1 Aufbau . . . 7
4.2 Zeitverhalten . . . 7
4.3 Zustandsgraph . . . 7

5 Erstellen des Zustandsgraphen **9**
5.1 Entwurf der Zustandsmaschinen . . . 9
5.2 Entwicklung von einem beliebigen Zustand aus . . . 9
5.3 Entwicklung von einem Szenario aus . . . 11
5.4 Entwicklung von einer Zustandsliste aus . . . 12

6 Codierung der Zustände **14**
6.1 Minimale Codierung . . . 14
6.2 Codierung 1 unter m . . . 14

Literatur **16**

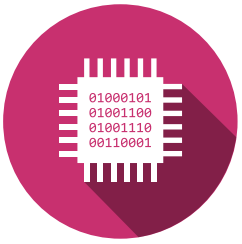
Akronyme **16**



1 Einführung

Schaltkreise zur Kontrolle von Systemen werden im allgemeinen mit Hilfe von **Zustandsmaschinen** realisiert.

Es gibt zwei Arten von Zustandsmaschinen: Moore-Maschinen, bei denen die Ausgänge nur von den inneren Zuständen abhängen, und Mealy-Maschinen, bei denen die Ausgänge von den inneren Zuständen und von den Eingängen abhängen.



2 Logische Synchronsysteme

2.1 Definition: logisches Synchronsystem

Hauptmerkmal eines logischen Synchronsystems ist die Tatsache, dass alle Ausgänge der Flipflops nur bei einer Taktflanke des Hauptsystemtakts ändern können. Dies setzt voraus, dass

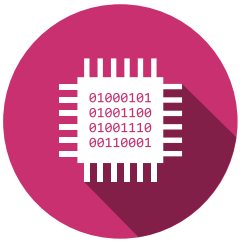
- alle Flipflops das selbe Taktsignal haben,
- kein Flipflop asynchrone Eingänge hat.

Aufgrund der Initialisierung und der Prüfbarkeit werden normalerweise alle Flipflops mit einer Nullstellungssteuerung verknüpft. Das System wird auch dann als synchron angesehen, wenn die Nullstellungssteuerung von ausserhalb des Schaltkreises kommt und für alle Flipflops identisch ist.

2.2 Taktsignal

Das Taktsignal entspringt einem Oszillator, der eine regelmässige Frequenz hat und nicht durch die logischen Gatter blockiert werden soll, um den Schaltkreis eine gewisse Zeit stillzulegen.

Im allgemeinen hat das Taktsignal eine viel höhere Frequenz als die Eingangssignale des logischen Synchronsystems.



3 Moore-Maschinen

3.1 Aufbau

Eine Moore-Zustandsmaschine ist ein logisches Synchronsystem mit

- Flipflops, die einen inneren Zustand speichern,
- einem logisch-kombinatorischen Block, der den zukünftigen inneren Zustand in Abhängigkeit des gegenwärtigen inneren Zustands und der Eingänge bestimmt,
- einem logisch-kombinatorischen Block, der den Wert der Ausgänge in Abhängigkeit des gegenwärtigen inneren Zustands bestimmt.

In Abbildung 1 ist ein Beispiel einer Moore-Zustandsmaschine dargestellt.

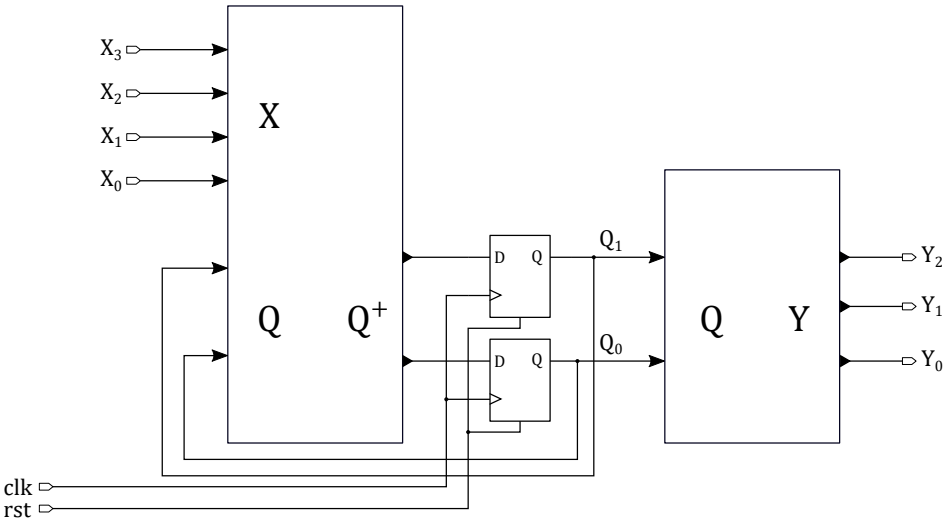


Abbildung 1: Architektur einer Moore-Zustandsmaschine

3.1.1 Definition: Eingangszustand

Ein input state (Eingangszustand) ist eine Kombination von Werten der Eingangssignale.

Bei einem System mit n Eingängen gibt es 2^n mögliche Eingangszustände.

3.1.2 Definition: interner Zustand

Ein internal state (interner Zustand) ist eine Kombination von Werten der Flipflopausgänge.

Bei einem System mit m Flipflops gibt es 2^m mögliche innere Zustände.

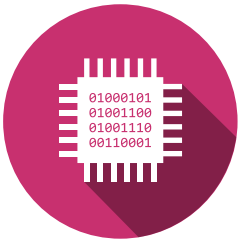
3.1.3 Definition: Ausgangszustand

Ein output state (Ausgangszustand) ist eine Kombination von Werten der Ausgangssignale.

Bei einem System mit l Eingängen gibt es 2^l mögliche Ausgangszustände.

3.2 Wahrheitstabelle der kombinatorischen Blöcke

Das Funktionieren einer Zustandsmaschine kann mit Hilfe einer Wahrheitstabelle untersucht werden, die für jeden Eingangszustand und jeden vorhandenen inneren Zustand den zukünftigen



inneren Zustand sowie den Ausgangszustand angibt.

Beispiel: kaskadierbarer Zähler

Abbildung 2 zeigt das Schema eines kaskadierbaren Zählers.

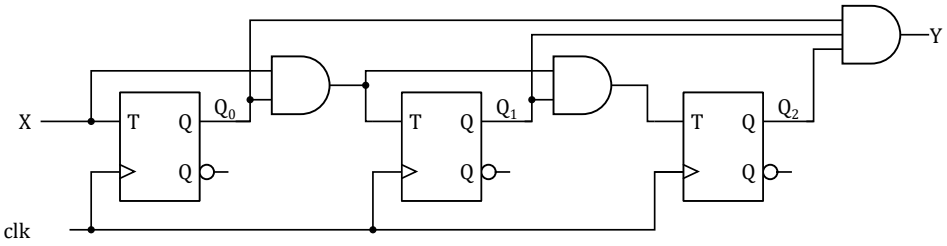


Abbildung 2: Kaskadierbarer Zähler

Tabelle 1 stellt die entsprechende Wahrheitstabelle dar.

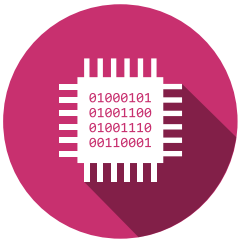
X	$Q_{2\ldots0}$	$T_{2\ldots0}$	$Q^+{}_{2\ldots0}$	Y
0	000	000	000	0
0	001	000	001	0
0	010	000	010	0
0	011	000	011	0
0	100	000	100	0
0	101	000	101	0
0	110	000	110	0
0	111	000	111	1
1	000	001	001	0
1	001	011	010	0
1	010	001	011	0
1	011	111	100	0
1	100	001	101	0
1	101	011	110	0
1	110	001	111	0
1	111	111	000	1

Tabelle 1: Wahrheitstabelle der kombinatorischen Blöcke des kaskadierbaren Zählers

3.3 Zustandsgraph

Durch die Wahrheitstabelle der kombinatorischen Blöcke kann der Zustandsgraph gezeichnet werden, der eine verständlichere Darstellung des Funktionierens der Zustandsmaschine liefert.

Eine Zustandsmaschine mit m Flipflops hat 2^m Zustände. Jeder Zustand ist durch einen Kreis dargestellt. Bei einer Maschine mit Eingängen gilt es, 2^n Fälle zu spezifizieren: von jedem Zustand aus gehen 2^n Pfeile weg. Bei einer Moore-Maschine hängen die Ausgänge lediglich von den Zuständen ab. Somit ist es am einfachsten, den Wert darzustellen, den die Ausgänge innerhalb des dem Zustand entsprechenden Kreises innehaben.



Beispiel: kaskadierbarer Zähler

Der in Abbildung 3 dargestellte Zustandsgraph des kaskadierbaren Zählers entsteht aus der Wahrheitstabelle der kombinatorischen Blöcke aus Tabelle 1.

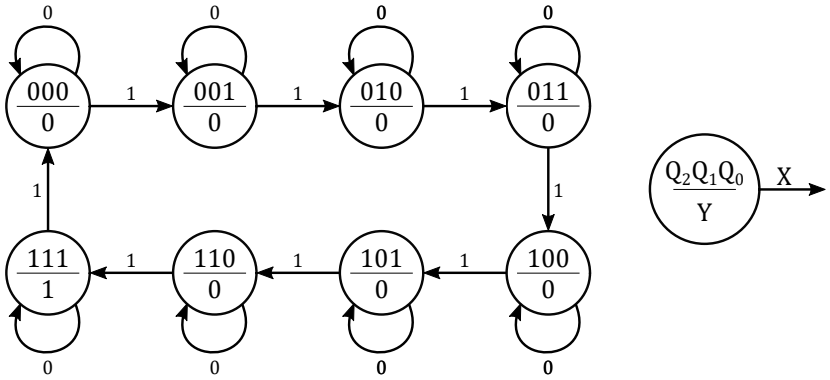


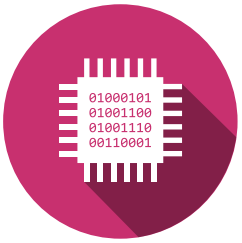
Abbildung 3: Zustandsgraph des kaskadierbaren Zählers

Es kann festgestellt werden, dass der Zähler bei $X =$ '0' stehen bleibt und bei $X =$ '1' regelmässig läuft. Erlangt der Zähler den Höchstwert, tätigt er einen Übertrag $Y =$ '1' für die Kaskadierung.

Für die Darstellung des Zustandsgraphen ist es wichtig, die Reihenfolge festzulegen, in der die Signale angegeben werden.

3.4 Definition: stabiler Gesamtzustand

Bei einem stabilen Gesamtzustand bleibt die Maschine in diesem Zustand, jedenfalls für wenigstens einen der Eingangszustände. Im Zustandsgraphen führt ein Pfeil vom stabilen Gesamtzustand weg und wieder zu ihm zurück.



4 Mealy-Maschinen

4.1 Aufbau

Im Unterschied zu den Moore-Maschinen haben Mealy-Maschinen Ausgänge, die auch von den Eingängen abhängen können. In Abbildung 4 ist das Beispiel einer Mealy-Zustandsmaschine dargestellt.

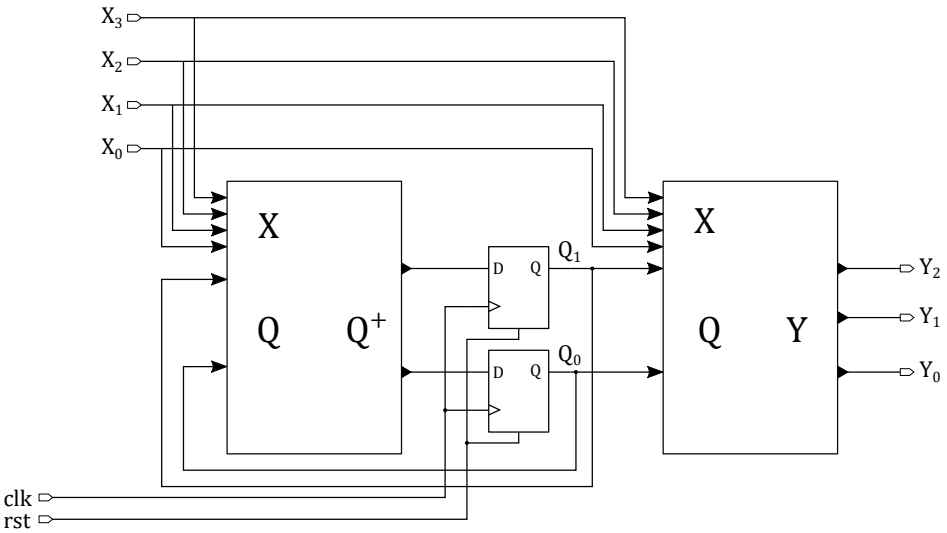


Abbildung 4: Architektur einer Mealy-Zustandsmaschine

4.2 Zeitverhalten

In Abbildung 5 ist das Zeitverhalten der Ausgangssignale einer Mealy-Maschine dargestellt.

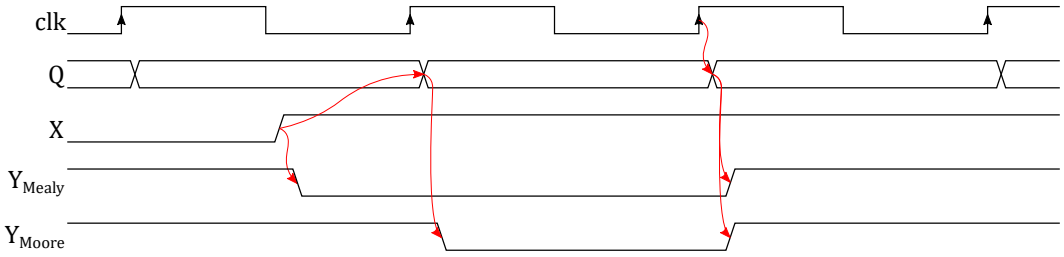


Abbildung 5: Zeitliches verhalten einer Mealy-Maschine

Bei einer Mealy-Maschine können die Ausgänge direkt auf die Variation eines Eingangs reagieren; bei einer Moore-Maschine hingegen ändern sie erst nach einer aktiven Taktflanke.

4.3 Zustandsgraph

Der Zustandsgraph einer Mealy-Maschine muss anzeigen, inwiefern die Ausgänge auch von den Eingängen abhängen. Die Ausgangswerte werden neben den Eingangswerten dargestellt.

Beispiel: kaskadierbarer Zähler mit Inhibition des Übertrags

In Abbildung 6 ist ein kaskadierbarer Zähler dargestellt, bei dem das Übertragungssignal auf $Y =$ '0' gebracht wird, wenn der Eingang auf $X =$ '0' ist.

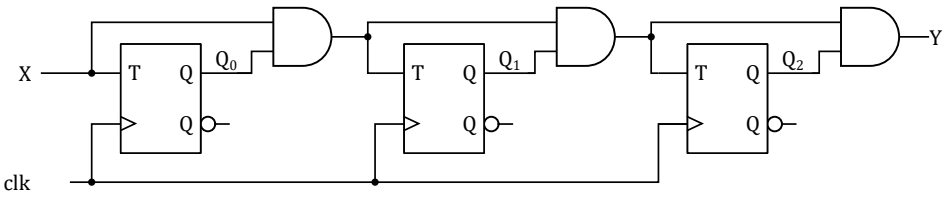


Abbildung 6: Kaskadierbarer Zähler mit Ausschalten des Übertrags

Der entsprechende Zustandsgraph ist in Abbildung 7 dargestellt.

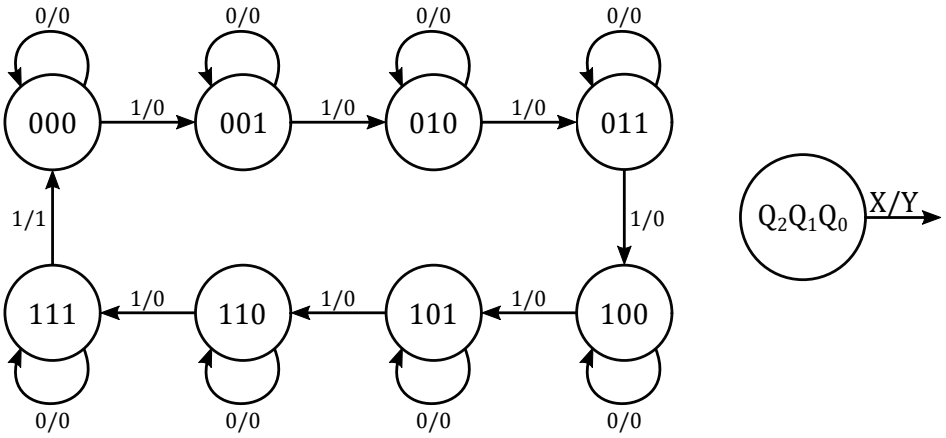
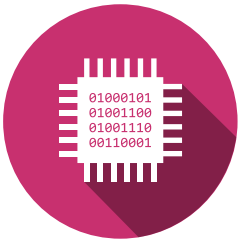


Abbildung 7: Zustandsgraph des Zählers mit Ausschalten des Übertrags

Der neben einem Pfeil aufgeführte Wert der Ausgänge gilt für jenen Zustand, von dem der Pfeil wegführt und nicht für den Zustand, zu dem er hinführt.



5 Erstellen des Zustandsgraphen

5.1 Entwurf der Zustandsmaschinen

Die Zustandsmaschinen können wie folgt entworfen werden:

- Ausdrücken des Pflichtenhefts durch einen Zustandsgraphen,
- Ausfüllen der Zustandstabelle,
- Reduktion der Anzahl Zustände,
- Codierung der Zustände,
- Realisation des logischen Schaltkreises.

Am heikelsten ist es, ausgehend vom Pflichtenheft des Systems einen Zustandsgraphen zu entwickeln. Zu diesem Zweck sind nachfolgend 3 Methoden aufgeführt.

5.2 Entwicklung von einem beliebigen Zustand aus

Eine Möglichkeit, einen Zustandsgraphen zu erstellen, ist das Vorgehen von Zustand zu Zustand. Dazu muss

- ein bestimmter Zustand der Maschine definiert werden.
- Für diesen Zustand müssen alle möglichen Eingangsbedingungen (2^n Fälle für n Eingänge) analysiert sowie die entsprechenden Pfeile gezeichnet werden. Je nach Fall entstehen neue Zustände.
- Schrittweise alle noch nicht behandelten Zustände vornehmen, alle möglichen Eingangsbedingungen (2^n Fälle für n Eingänge) analysieren und die entsprechenden Pfeile zeichnen. Jedesmal, wenn ein neuer Zustand entsteht, muss überprüft werden, ob er nicht bereits im sich entwickelnden Graphen existiert. Diesen Schritt wiederholen, bis der Graph vollständig ist.

Es kann Situationen geben, in denen sich gewisse Eingangsbedingungen nicht einstellen können. Dies ist bezeichnenderweise der Fall, wenn durch den Aufbau mehrere Eingangssignale nicht simultan ändern können. In diesem Fall führen weniger als 2^n Pfeile von jedem Zustand weg.

Beispiel: Entprellerschaltung

Die zu entwerfende Entprellerschaltung ist durch folgendes Pflichtenheft gegeben: der Schaltkreis überträgt das Eingangssignal erst, wenn dieses während den letzten drei Taktschlägen nicht variiert hat; andernfalls behält der Schaltkreis den selben Ausgangswert wie zuvor inne.

Es handelt sich hierbei ganz eindeutig um eine Moore-Maschine: eine kurzzeitige Variation des Eingangs darf der Ausgang keinesfalls beeinflussen.

Definieren wir einen bestimmten Zustand der Maschine: das Eingangssignal lag genügend lange Zeit bei '0', und der Ausgang ist gleich '0'.

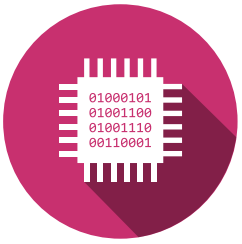


Abbildung 8: Entwicklung des Graphen: Anfangszustand

Die Analyse aller möglichen Eingangsbedingungen ergibt zwei mögliche Fälle: Entweder ist das Eingangssignal bei '0', und man bleibt im selben Zustand, oder aber das Signal liegt bei '1', und der Zustand wird gewechselt, um sich dem Zustand anzunähern, in dem das Ausgangssignal auf '1' kippen kann.

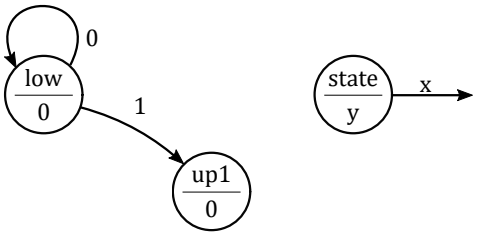


Abbildung 9: Entwicklung des Graphen: Analyse aller möglichen Eingangsbedingungen

Ein neuer Zustand ist entstanden. Nun müssen also wieder alle möglichen Eingangsbedingungen dieses neuen Zustands analysiert werden.

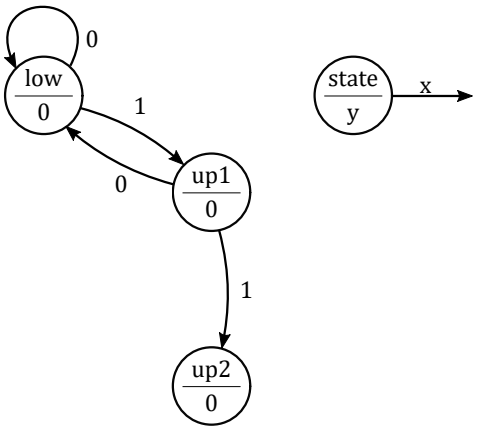


Abbildung 10: Entwicklung des Graphen, Fortsetzung

Durch schrittweise Entwicklung der verschiedenen Zustände mittels Analyse aller möglichen Eingangsbedingungen erhält man den in Abbildung 11 dargestellten Graphen.

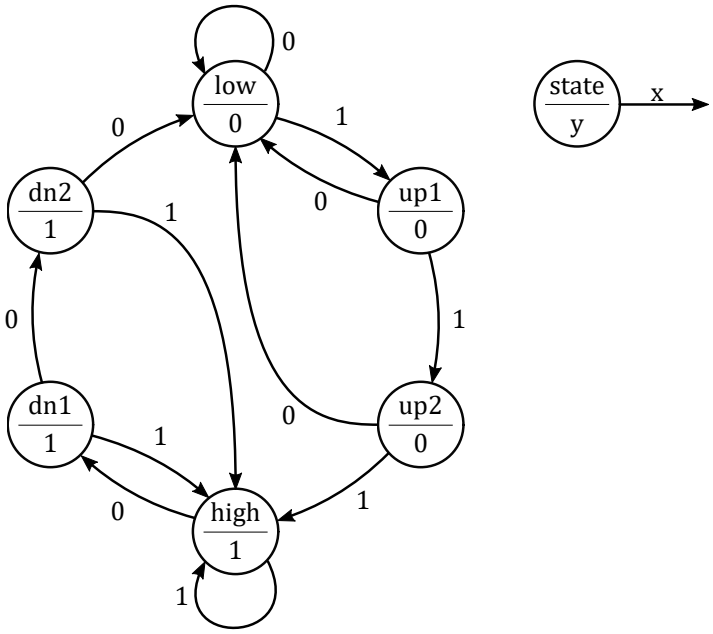


Abbildung 11: Zustandsgraph der Entprellerschaltung

5.3 Entwicklung von einem Szenario aus

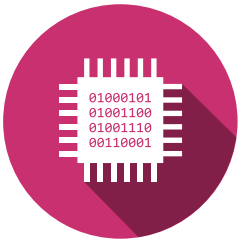
Die zweite Möglichkeit, einen Zustandsgraphen zu entwickeln, ist die Definition eines Szenarios. Dazu muss

- ein Szenario definiert werden, das einen Grossteil des Pflichtenhefts abdeckt.
- Das Szenario muss in Form eines Graphen entwickelt werden, wobei die Zustände den Etappen des Szenarios und die Pfeile den Handlungen entsprechen müssen.
- Für jeden so definierten Zustand müssen alle möglichen Eingangsbedingungen (2^n Fälle für n Eingänge) analysiert und die noch nicht vorhandenen Pfeile gezeichnet werden. Manchmal entstehen neue Zustände, die in diesem Fall vollständig analysiert werden müssen.

Beispiel: Entprellerschaltung

Im folgenden wird ein einfaches Szenario für die Entwicklung eines Graphen der Entprellerschaltung beschrieben: Das Eingangssignal bleibt genügend lange auf '0', damit der Ausgang auf '0' gehen kann, das Signal geht zu '1', bleibt genügend lange dort, damit der Ausgang es ihm gleichtun kann, und fällt dann auf '0' zurück. Dies wird durch den Graphen in Abbildung 12 dargestellt.

Der Graph ist noch nicht vollständig. Für jeden Zustand müssen alle möglichen Eingangsbedingungen analysiert werden, damit der Graph entsprechend vervollständigt werden kann. So erhält man den Graphen in Abbildung 11.



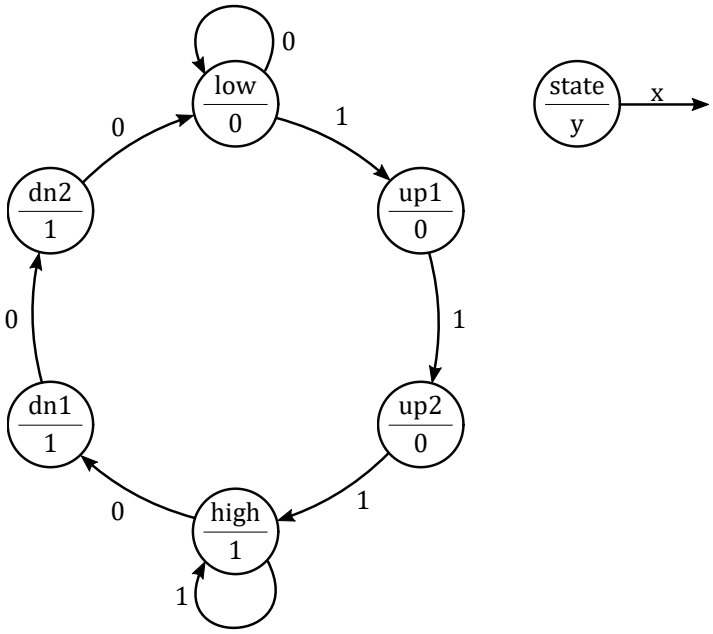


Abbildung 12: Szenario für die Entprellerschaltung

5.4 Entwicklung von einer Zustandsliste aus

Um einen Zustandsgraphen auf die dritte Art und Weise zu entwikkeln, müssen seine Zustände im voraus bekannt sein. Dazu müssen

- die einzuspeichernden Ereignisse oder Zustände bestimmt werden.
- Für jedes einzuspeichernde Ereignis wird ein Zustand gezeichnet.
- Für jeden so definierten Zustand werden alle möglichen Eingangsbedingungen (2^n Fälle für n Eingänge) analysiert und die entsprechenden Pfeile gezeichnet.

Beispiel: Entprellerschaltung

Das Einzuspeichernde wird bestimmt: Um zu wissen, dass das Signal nicht variiert hat, müssen die beiden letzten Werte des Eingangssignals sowie sein gegenwärtiger Wert bekannt sein; wenn das Signal variiert hat, muss man sich an den vorherigen Wert des Ausgangs erinnern. Es müssen also 3 Binärwerte im Gedächnis behalten werden, wozu man 8 Zustände benötigt, was in Abbildung 13 dargestellt ist.

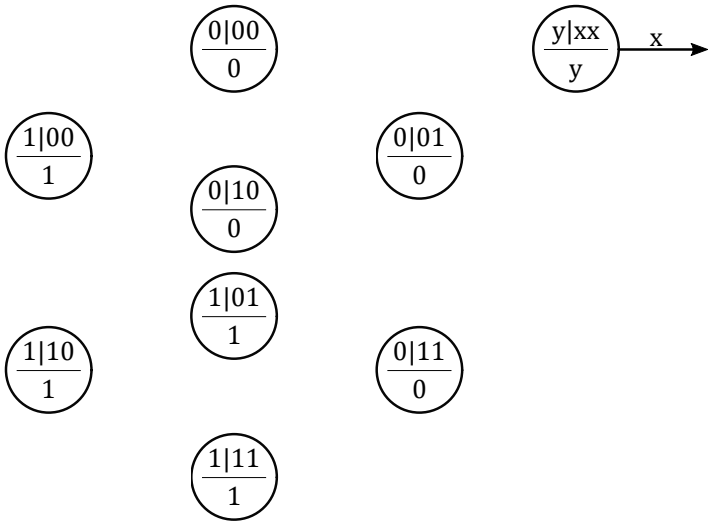


Abbildung 13: Einzuspeichernde Zustände

Für jeden so definierten Zustand müssen alle möglichen Eingangsbedingungen analysiert sowie die entsprechenden Pfeile gezeichnet werden, was den Graphen in Abbildung 14 ergibt.

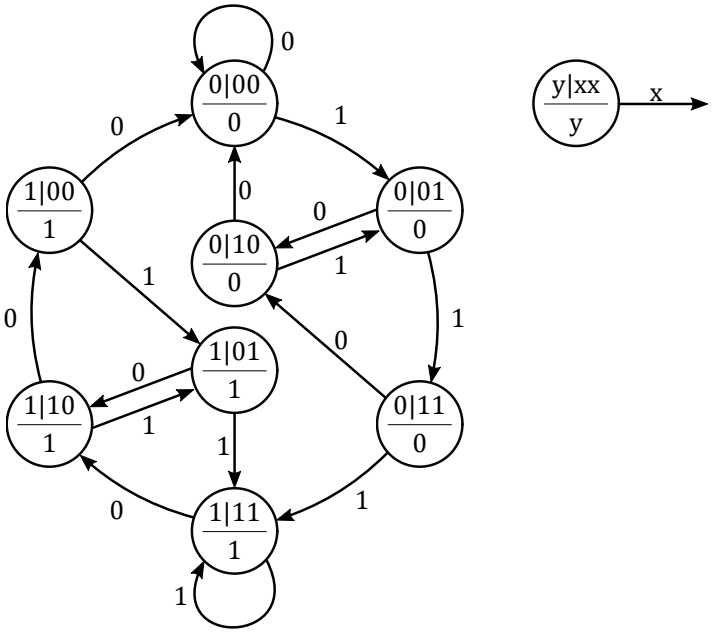
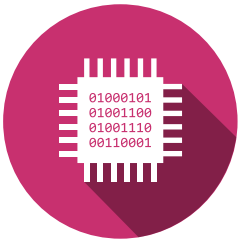


Abbildung 14: Zustandsgraph anhand der Einzuspeichernde Zustände

Die Graphen, die von verschiedenen Methoden ausgehend entwickelt wurden, haben nicht unbedingt die selbe Anzahl Zustände, auch dann nicht, wenn sie identisch funktionieren.



6 Codierung der Zustände

Nach der Reduktion des Zustandsgraphen muss den verschiedenen Zuständen noch ein Binärcode zugewiesen werden, um die Wahrheitstabelle der logisch-kombinatorischen Blöcke auszufüllen und das Schema der Zustandsmaschine zu bestimmen.

6.1 Minimale Codierung

Die Mindestanzahl Bits, m, die notwendig ist, um die M Zustände einer Maschine zu codieren, ist gegeben durch

$$2^{m-1} < M < 2^m \tag{1}$$

Diese Ungleichung gibt die Mindestanzahl der Flipflops an, die für eine Realisierung der Zustandsmaschine notwendig sind.

Die Zuweisung der Codes kann willkürlich erfolgen, indem der erste Binärcode dem ersten beliebigen Zustand zugewiesen wird, der zweite dem nächsten Zustand usw.

Beispiel: Entprellerschaltung

Die Tabelle 2 gibt eine mögliche Codierung der Zustände für die Entprellerschaltung.

$state$	Q
low	000
$up1$	001
$up2$	010
$high$	111
$dn1$	110
$dn2$	011

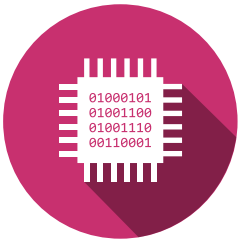
Tabelle 2: Binärcodierung der Zustände

Mittels dieser Codierung erhält man für die logisch-kombinatorischen Blöcke der Zustandsmaschine folgende Gleichungen:

$$\begin{cases} D_2 & = xQ_1 + Q_2Q_0 \\ D_1 & = xQ_1 + x\overline{Q_0} + Q_2 \\ D_0 & = xQ_1 + x\overline{Q_0} + Q_2\overline{Q_0} \\ Y & = Q_2 + Q_1Q_0 \end{cases} \tag{2}$$

6.2 Codierung 1 unter m

Eine andere Möglichkeit der Codierung besteht in der Verwendung von M Bits, und somit von M Flipflops, um die M Zustände einer Maschine zu codieren. Die Codierung wird so vorgenommen, dass für jeden Zustand nur ein Bit bei '1' liegt.



Diese Codierung benötigt viele Flipflops, aber die damit verbundenen kombinatorischen Funktionen sind im allgemeinen sehr einfach.
Vor allem aber können so die Zustände zufallssicher decodiert werden. Dies ist besonders wichtig bei Zustandsmaschinen für das sequentielle Ordnen von Operationen.
In einem Code 1 unter m wird die Mehrheit der möglichen inneren Zustände nicht benutzt. Die Wahrscheinlichkeit, abgesehen von der Hauptschleife des Graphen unerwünschte Schleifen zu erhalten, ist somit gross. Aus diesem Grunde ist es wichtig, eine Logik vorzusehen, um die Maschine in einen der Zustände der Hauptschleife zu bringen.

Beispiel: Entprellerschaltung

In Tabelle 3 ist ein Beispiel einer Codierung 1 unter m für die Zustände des Graphen der Abbildung 11 dargestellt.

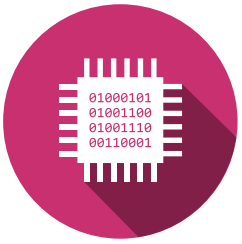
$state$	$Q_5 \dots Q_0$
low	000001
$up1$	000010
$up2$	000100
$high$	001000
$dn1$	010000
$dn2$	100000

Tabelle 3: Codierung 1 unter m

Die Gleichungen können direkt dem Zustandsgraphen entnommen werden. Tatsächlich ist jeder Zustand mit einem Zustandsbit verbunden. Die Bedingung «der zukünftige Zustand ist $up1$, wenn der gegenwärtige Zustand low und der Eingang gleich '1' sind» wird als $D_1 = Q_0 x$ausgedrückt. Die Gleichungen für die logisch-kombinatorischen Blöcke der Zustandsmaschine sind also gegeben durch

$$\begin{cases} D_0 & = \bar{x}Q_0 + \bar{x}Q_1 + \bar{x}Q_2 + \bar{x}Q_5 \\ D_1 & = xQ_0 \\ D_2 & = xQ_1 \\ D_3 & = xQ_2 + xQ_3 + xQ_4 + xQ_5 \\ D_4 & = \bar{x}Q_3 \\ D_5 & = \bar{x}Q_4 \\ Y & = Q_3 + Q_4 + Q_5 \end{cases} \tag{3}$$

Für die Initialisierung muss vorgesehen werden, alle Flipflops auf '0' zu stellen, ausser einem, der auf '1' zu stellen ist.



Literatur

[1] Suhail Almani. *Electronic Logic Systems*. second edition. New-Jersey: Prentice-Hall, 1989.

[2] Michael D. Ciletti und M. Morris Mano. *Digital Design*. second edition. New-Jersey: Prentice-Hall, 2007.

[3] David J. Comer. *Digital Logic and State Machine Design*. Saunders College Publishing, 1995.

[4] Randy H. Katz und Gaetano Borriello. *Contemporary Logic Design*. California: The Benjamin/Cummings Publishing Company Inc, 2005.

[5] Martin V. Künzli und Marcel Meli. *Vom Gatter Zu VHDL: Eine Einführung in Die Digitaltechnik*. vdf Hochschulverlag AG, 2007. ISBN: 3 7281 2472 9.

[6] David Lewin und Douglas Protheroe. *Design of Logic Systems*. second edition. Hong Kong: Springer, 2013.

[7] Daniel Mange. *Analyse et synthèse des systèmes logiques*. Editions Géorgi. Bd. Traité d'électricité, volume V. St Saphorin: PPUR presses polytechniques, 1995. 362 S. ISBN: 978-2-88074-045-0. Google Books: 5NSdD4GRl3cC.

[8] John F. Wakerly. *Digital Design: Principles And Practices*. 3rd edition. Prentice-Hall, 2008. ISBN: 0-13-082599-9.

Akronyme

Ausgangszustand output state. 4

Eingangszustand input state. 4

interner Zustand internal state. 4

machine d'état Finite State Machine, FSM. 1

Zustandsmaschine Finite State Machine, FSM. 4, 7, 9

état d'entrée input state. 1

état interne internal state. 1