

Rechnerstrukturen WS 2011/12

- ▶ Register
 - ▶ Schieberegister
 - ▶ Datenbus

- ▶ Taktung von Digitalrechnern
 - ▶ Takt- und Phasenableitung

- ▶ Zusammenfassung

Hinweis: *Folien teilweise a. d. Basis von Materialien von Thomas Jansen*

29. August 2011

Speicher

Spezielle Speicher: Register

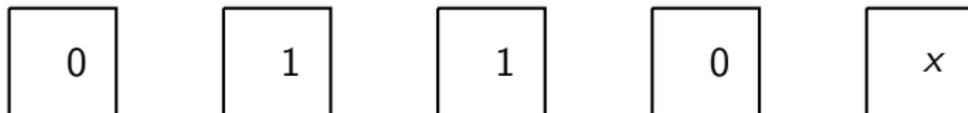
Fakt in Computern meist keine direkte Speicheranipulation

stattdessen spezielle, direkt der CPU zugehörige Speicherzellen
Register

klar Register sind auch nur Flip-Flops

aber spezielle Funktionalität \rightsquigarrow gesonderte Betrachtung

Wir betrachten Schieberegister



Funktion Speicherinhalt nach links oder rechts verschieben

Was machen wir an den Rändern?

möglich zyklisch verschieben

möglich streichen und beliebig auffüllen

Entwurf Schieberegister

Entwurf nicht-zyklisches Schieberegister mit drei Bits

Warum gerade drei Bits?

- ▶ klein genug, um auf die Folie zu passen
- ▶ groß genug, um alles Wichtige zu enthalten
 - ▶ Bit am linken Rand
 - ▶ Bit am rechten Rand
 - ▶ mittleres Bit

Eingänge

- ▶ d (direction) Schieberichtung ($d = 0 \hat{=}$ links, $d = 1 \hat{=}$ rechts)
- ▶ x aufzufüllender Wert

Vereinfachung: betrachten nicht Ein-/Ausgänge zum Schreiben/Lesen des Registers als Ganzem

Mealy-Automat zum nicht-zyklischen Schieberegister

$$\Sigma = \{00, 01, 10, 11\} \quad \text{Interpretation} \quad dx \in \Sigma$$

$$\Delta = \emptyset \quad \text{also } \forall q \in Q, w \in \Sigma: \lambda(q, w) = \varepsilon$$

$$Q = \{000, 001, 010, 011, 100, 101, 110, 111\} \quad \text{Interpretation}$$

Registerinhalt

$$q_0 = 000 \quad (\text{völlig willkürlich})$$

$$\delta: Q \times \Sigma \rightarrow Q$$

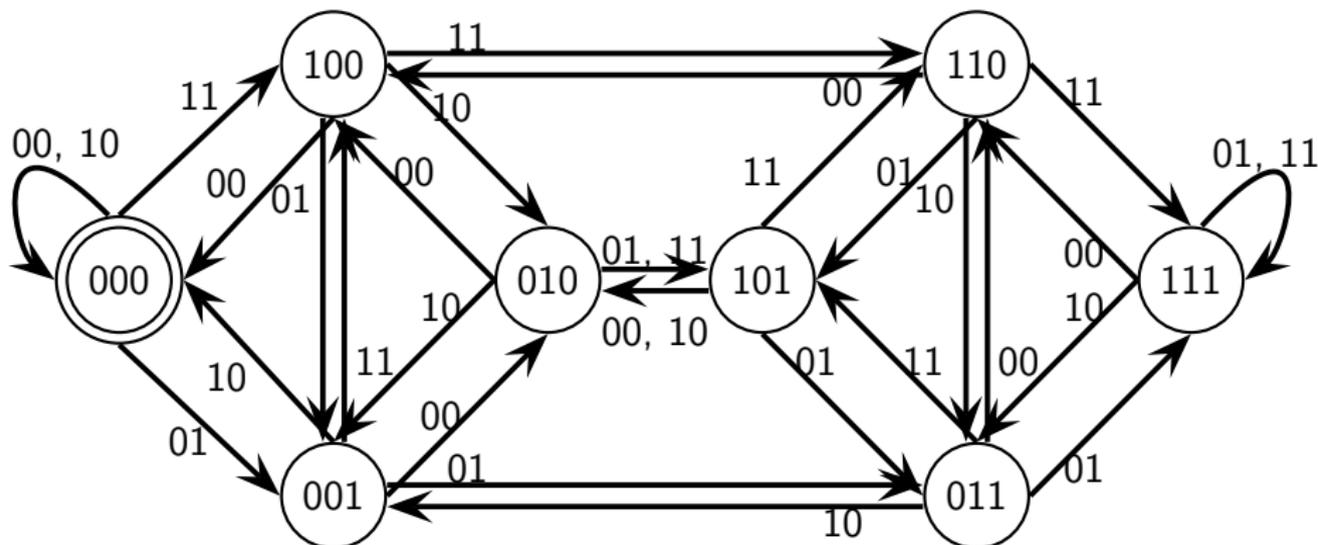
$q \in Q$	$w \in \Sigma$	$\delta(q, w) \in Q$
000	00	000
000	01	001
000	10	000
000	11	100
\vdots	\vdots	\vdots

insgesamt 32 Zeilen

Grafische Darstellung Mealy-Automat

$\Sigma = \{00, 01, 10, 11\}$, $w = d \times$ ($d = 0 \hat{=} \text{links}$, $d = 1 \hat{=} \text{rechts}$)

$\Delta = \emptyset$, $Q = \{0, 1\}^3$, $q_0 = 000$, $\lambda(q, w) = \varepsilon$



Schaltwerk-Synthese abgekürzt

Codierung

kanonisch $w = d \times$ durch d, x
 $q = q_l q_m q_r \in \{0, 1\}^3$ durch q_l, q_m, q_r

Müssen wir jetzt Tabellen aufstellen?

Einsicht Strukturverständnis kann helfen

zum linken Bit

1. Fall $d = 0 \hat{=}$ links

klar $q_{l(\text{neu})} = q_m$

2. Fall $d = 1 \hat{=}$ rechts

klar $q_{l(\text{neu})} = x$

also $q_{l(\text{neu})} = \bar{d} q_m \vee d x$

Boolesche Funktionen für mittleres und rechtes Bit

zum rechten Bit

1. Fall $d = 0 \hat{=} \text{links}$

klar $q_{r(\text{neu})} = x$

2. Fall $d = 1 \hat{=} \text{rechts}$

klar $q_{r(\text{neu})} = q_m$

also $q_{r(\text{neu})} = \bar{d} x \vee d q_m$

zum mittleren Bit

1. Fall $d = 0 \hat{=} \text{links}$

klar $q_{m(\text{neu})} = q_r$

2. Fall $d = 1 \hat{=} \text{rechts}$

klar $q_{m(\text{neu})} = q_l$

also $q_{m(\text{neu})} = \bar{d} q_r \vee d q_l$

Schaltwerk nicht-zyklisches Schieberegister

Moment! Hätten wir nicht Flip-Flops wählen und
und Ansteuerfunktionen berechnen müssen?

Einsicht nicht, wenn man D-Flip-Flops verwendet

also nur noch

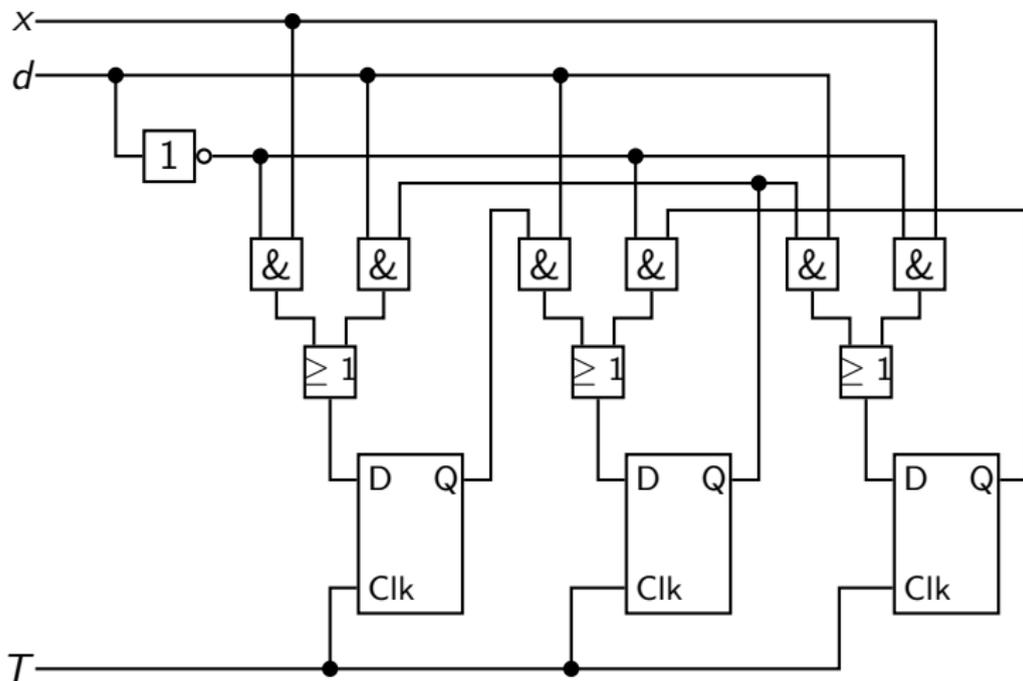
$$q_{l(\text{neu})} = \overline{d} q_m \vee d x$$

$$q_{m(\text{neu})} = \overline{d} q_r \vee d q_l$$

$$q_{r(\text{neu})} = \overline{d} x \vee d q_m$$

direkt in ein Schaltwerk übertragen

Schaltwerk nicht-zyklisches Schieberegister



Zyklisches Schieberegister

jetzt zyklisches Schieberegister
wieder für drei Bits
wieder ohne Daten-Eingabe

zuerst Mealy-Automat

wie vorhin $Q = \{0, 1\}^3$, $\Delta = \emptyset$, $\forall q \in Q, w \in \Sigma: \lambda(q, w) = \varepsilon$

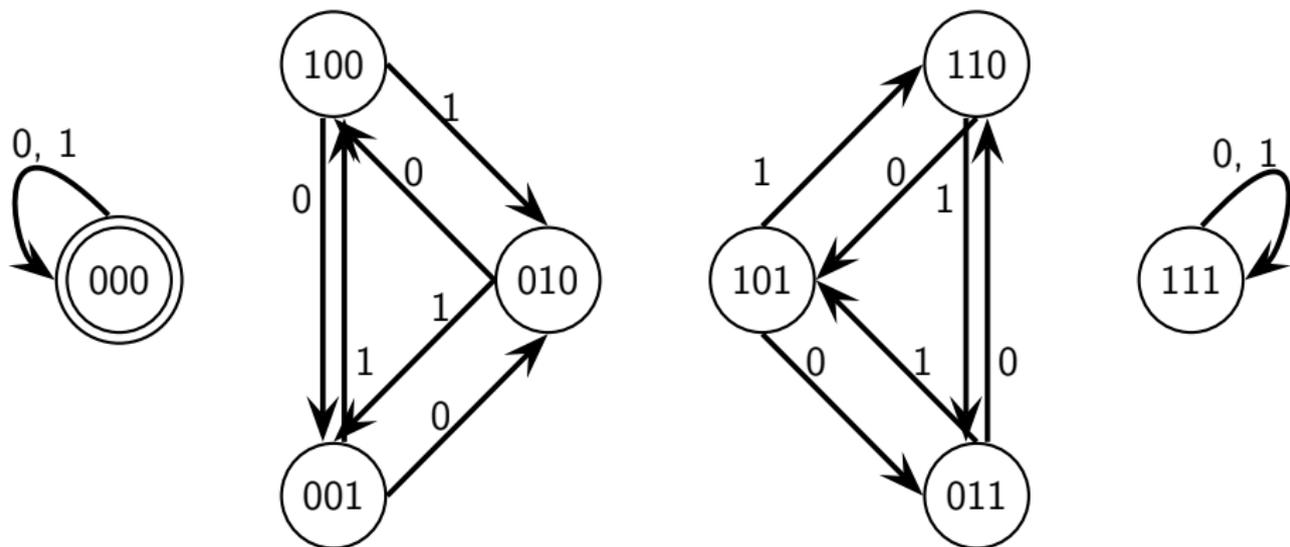
anders $\Sigma = \{0, 1\}$ (weil x fehlt)

wie vorhin Interpretation $0 \hat{=}$ links, $1 \hat{=}$ rechts

Mealy-Automat zyklisches Schieberegister

$\Sigma = \{0, 1\}$, $w = d$ ($d = 0 \hat{=}$ links, $d = 1 \hat{=}$ rechts)

$\Delta = \emptyset$, $Q = \{0, 1\}^3$, $q_0 = 000$, $\lambda(q, w) = \varepsilon$



Boolesche Funktionen zum Mealy-Automaten

wie vorhin Funktionen für q_l , q_m , q_r direkt ableiten

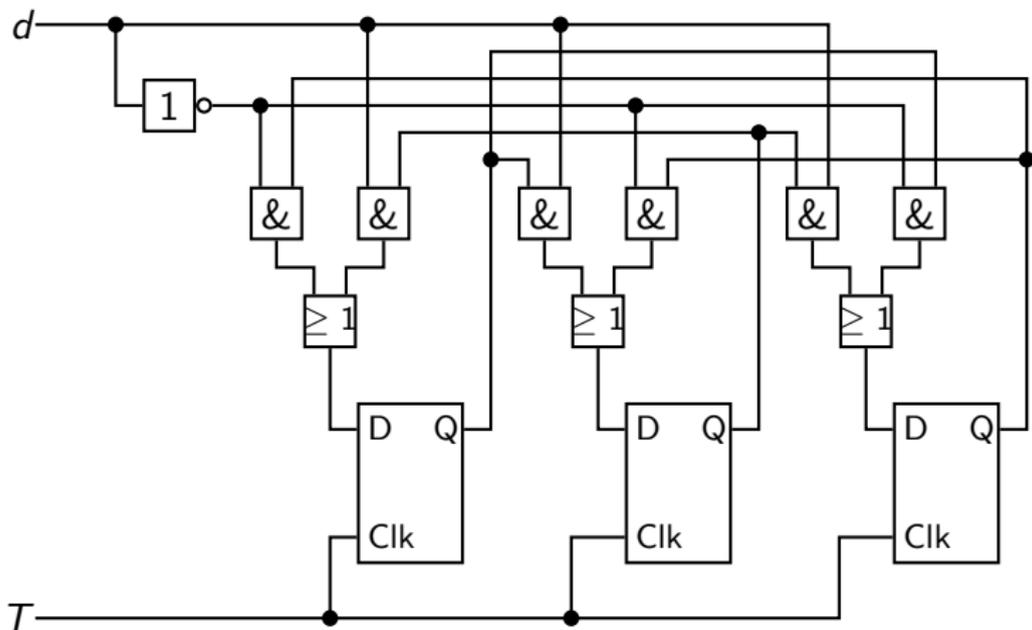
▶ $q_l = \bar{d} q_m \vee d q_r$

▶ $q_m = \bar{d} q_r \vee d q_l$

▶ $q_r = \bar{d} q_l \vee d q_m$

jetzt Schaltwerk direkt angebar
bei Verwendung von D-Flip-Flops

Schaltwerk zyklisches Schieberegister



Schieberegister allgemein

Können wir die Funktionen für die Bits auch allgemein angeben?

Notation Bits q_{n-1}, \dots, q_1, q_0
 q_{n-1} am linken Rand, q_0 am rechten Rand

zyklisches Schieberegister

$$q_i = \bar{d} q_{(i-1) \bmod n} \vee d q_{(i+1) \bmod n} \quad \text{auch am Rand}$$

Auch am Rand?

nicht-zyklisches Schieberegister

$$q_i = \bar{d} q_{i-1} \vee d q_{i+1} \quad \text{auch am Rand mit } q_{-1} := x, q_n := x$$

Welche Operationen können Schieberegister realisieren?

Datentransfer

klar Daten müssen gelegentlich zwischen Registern transferiert werden

Beispiel Zwischenergebnis merken

Wie kann man Daten von einem Register in ein anderes transferieren?

klar auf Leitungen zwischen den Registern

Wie verbindet man Register geschickt?

Kriterien:

- ▶ Einfachheit (d. technischen Realisierung)
- ▶ Sparsamkeit (i.S.v. Schaltungsaufwand)
- ▶ Möglichkeit zum effizienten Datenaustausch

Register verbinden

eine Möglichkeit paarweise verbinden

Vorteile

- ▶ direkter Datenaustausch, also **schnell**
- ▶ bis $n/2$ Registerpaare gleichzeitig aktiv, also **gute Parallelisierung**

Nachteil

- ▶ $\binom{n}{2}$ Verbindungen

Beispiel $\binom{16}{2} = \frac{16 \cdot 15}{2} = 120$

Anmerkung meist nicht realisiert

Register sparsam verbinden

andere Möglichkeit alle Register mit einer
gemeinsamen Leitung verbinden

Vorteile

- ▶ nur eine Leitung, also **einfach**
- ▶ Datenaustausch direkt, also **schnell**

Nachteile

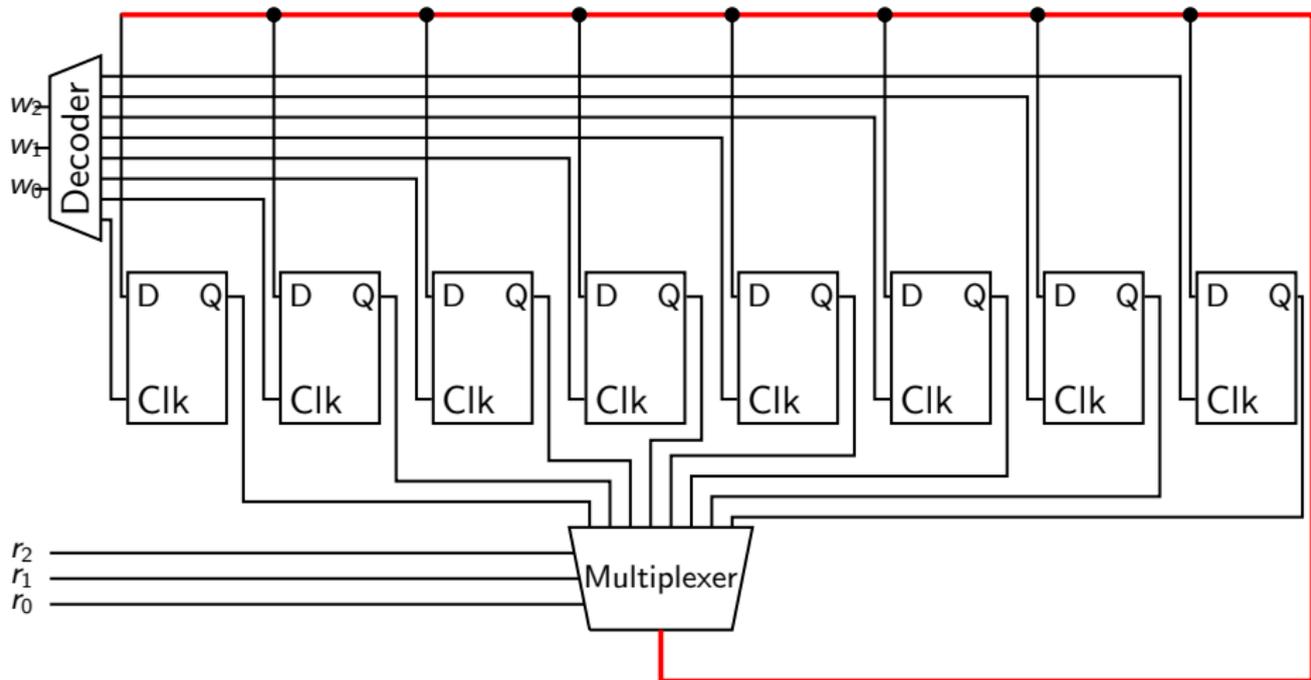
- ▶ immer nur ein Paar aktiv, also **langsam**
- ▶ Steuerlogik erforderlich, **nicht ganz einfach**

Name Bus

Anmerkungen

- ▶ häufig verwendet
- ▶ Engpass durch mehrere Busse entschärfbar

Bus für acht 1-Bit-Register



Verallgemeinerung

Verallgemeinerung auf mehr Register

- ▶ mehr Adressleitungen zum Adressieren des Quell- und Zielregisters
- ▶ größere Multiplexer und Decoder

Verallgemeinerung auf breitere Register

- ▶ je Register entsprechend mehr Flip-Flops (je Bit ein Flip-Flop)
- ▶ entsprechend mehr Multiplexer (je Bit ein Multiplexer)
- ▶ entsprechend mehr Bus-Leitungen (je Bit eine Bus-Leitung)

Taktung von Digitalrechnern

Gibt es in einem Rechner eigentlich **den** Takt?

jein oft mehrere Takte
aber nur eine „Uhr“

Wie funktioniert das?

Einsicht schneller geht **nicht**, verschoben oder langsamer schon

Langsamer – Wie geht das?

klar Takt(e) auslassen

klar dazu Zähler ausreichend

Beobachtung Zähler wie jedes Schaltwerk realisierbar

Taktreduktion mit Zählern

Idee: Zähle Taktimpulse \rightarrow erzeuge in jedem n -ten Takt einen *neuen* Impuls für den **langsameren** abgeleiteten Takt

Erforderlich: Zähler modulo n , d.h. $z' \leftarrow (z + 1) \bmod n$

Realisierung als synchrones Schaltwerk

- ▶ Zustand $\hat{=}$ Zählerstand, d.h.: $Q = \{0, 1, \dots, n - 1\}$
(speicherbar in $\log_2 n$ Bits / Flip-Flops)
- ▶ Ausgabe, z.B. wenn Zählerstand 0 erreicht \rightarrow **nicht betrachtet**
- ▶ Eingabe? ... eigentlich nicht zwingend erforderlich,
aber Umschaltung der Zählrichtung möglich!
 $\Rightarrow \Sigma = \{0, 1\}$ (d.h. 0 vorwärts, 1 rückwärts zählen)

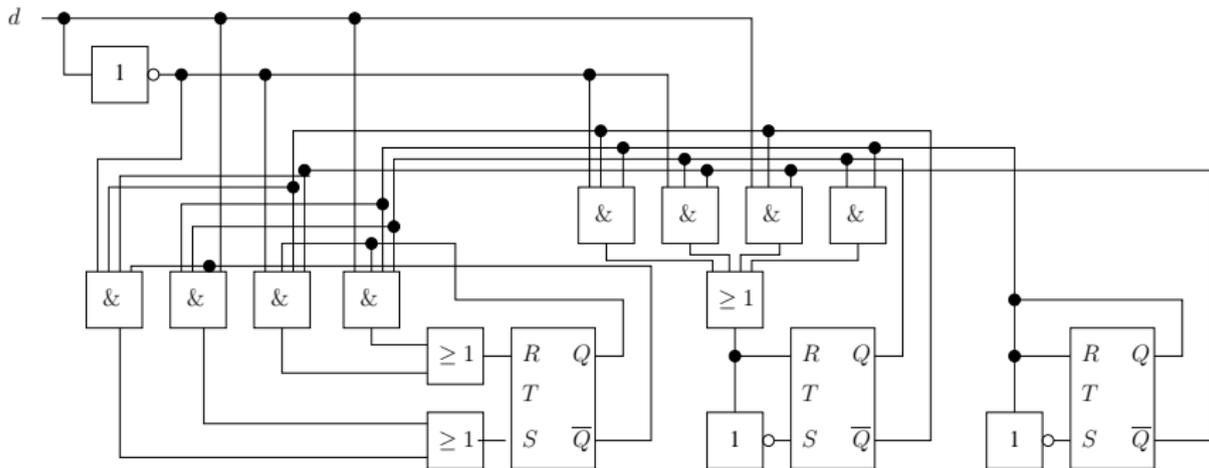
Beispiel: Zähler modulo 8

Zustandsüberföhrungsfunktion: $\delta(q, d) = (q + (-1)^d) \bmod 8$

Achtung bei Implementierung!

d	q			q_{neu}			R_l	S_l	R_m	S_m	R_r	S_r
0	0	0	0	1	1	1	0	1	0	1	0	1
0	0	0	1	0	0	0	*	0	*	0	1	0
0	0	1	0	0	0	1	*	0	1	0	0	1
0	0	1	1	0	1	0	*	0	0	*	1	0
0	1	0	0	0	1	1	1	0	0	1	0	1
0	1	0	1	1	0	0	0	*	*	0	1	0
0	1	1	0	1	0	1	0	*	1	0	0	1
0	1	1	1	1	1	0	0	*	0	*	1	0
1	0	0	0	0	0	1	*	0	*	0	0	1
1	0	0	1	0	1	0	*	0	0	1	1	0
1	0	1	0	0	1	1	*	0	0	*	0	1
1	0	1	1	1	0	0	0	1	1	0	1	0
1	1	0	0	1	0	1	0	◀▶*	0	0	0	1

Schaltwerk des Zählers modulo 8

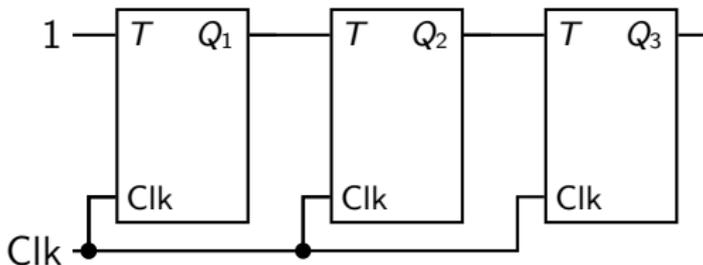


Nachteil: aufwendig!

Gibt es einfachere Möglichkeiten der Taktreduktion?

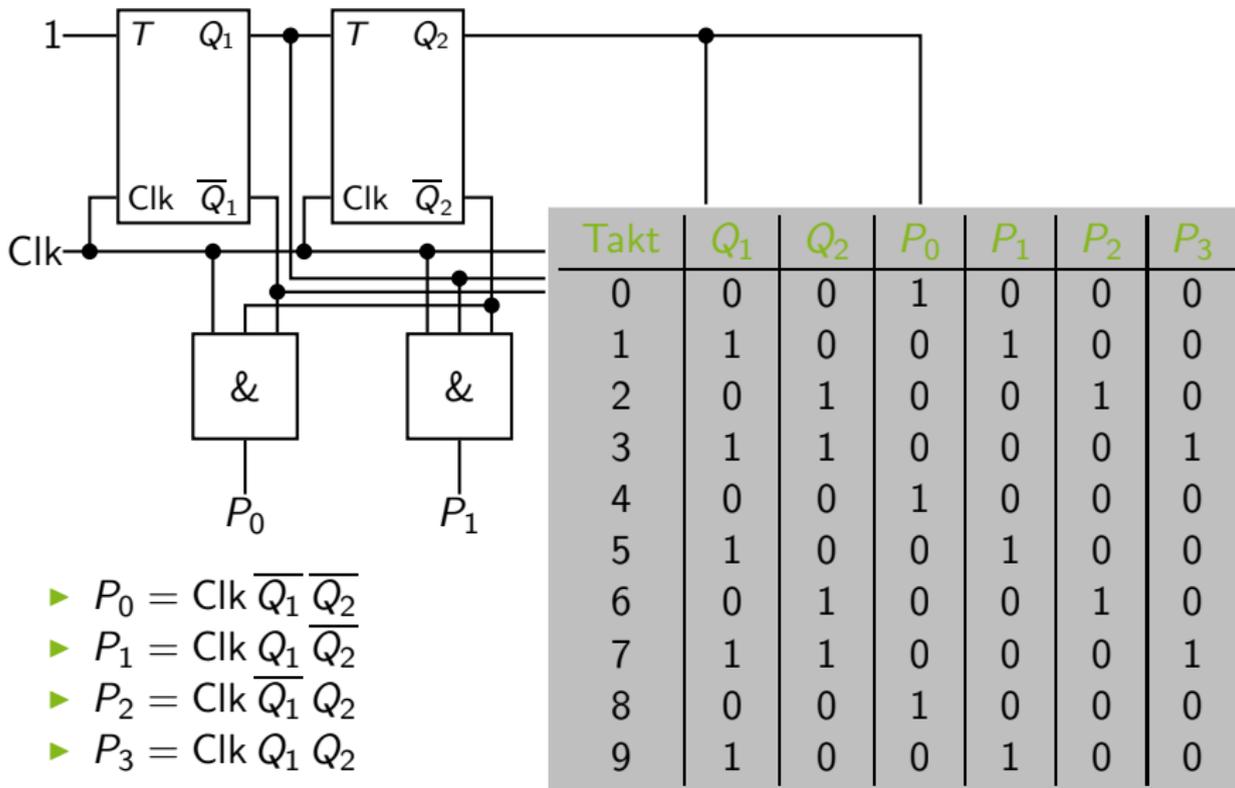
Taktleitung

Betrachte eine "Kette" von T -Flip-Flops



Takt	Q_1	Q_2	Q_3
0	0	0	0
1	1	0	0
2	0	1	0
3	1	1	1
4	0	0	0
5	1	0	0
6	0	1	0
7	1	1	1
8	0	0	0

Phasenableitung



Zusammenfassung

Zusammenfassung

- ▶ Repräsentation von Daten
 - ▶ natürliche Zahlen, ganze Zahlen
 - ▶ rationale Zahlen (IEEE 754)
 - ▶ *auch*: Texte, andere Daten
- ▶ Boolesche Funktionen und Schaltnetze
 - ▶ Boolesche Funktionen
 - ▶ Boolesche Algebra, Repräsentation boolescher Funktionen
 - ▶ Normalformen (DNF, KNF, RNF)
 - ▶ Repräsentation boolescher Funktionen mit OBDDs
 - ▶ Schaltnetze & Rechnerarithmetik
 - ▶ Addition (Ripple-Carry Addierer, Carry-Look-Ahead-Addierer)
 - ▶ Multiplikation (Carry-Save Addierer & Wallace-Tree)
 - ▶ Subtraktion $\hat{=}$ Addition negativer Zahlen
 - ▶ Gleitkommazahlen: Multiplikation, Addition
 - ▶ ...

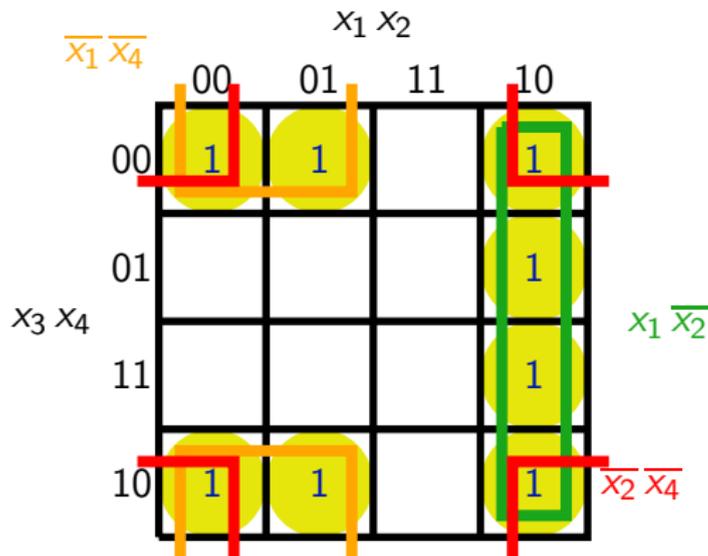
Zusammenfassung II

- ▶ Boolesche Funktionen und Schaltnetze (cont.)
 - ▶ ...
 - ▶ Strukturierter Schaltnetz-Entwurf (insbes. Multiplexer)
 - ▶ Optimierung von Schaltnetzen
 - ▶ KV-Diagramme: Minimalpolynome
 - ▶ Algorithmus von Quine und McCluskey: PI-Tafel
 - ▶ Unvollständig definierte Funktionen
 - ▶ Hazards (insbes. Schaltungshazards)
 - ▶ Programmierbare Bausteine: PLAs
- ▶ Sequenzielle Schaltungen
 - ▶ Modellierung mit Automaten
 - ▶ Synchrone Schaltwerke: Flip-Flops
 - ▶ Schaltwerk-Entwurf: von Neumann-Addierwerk
 - ▶ Speicher (insbes. Adressierung, Demultiplexer)
 - ▶ Register: Schieberegister, Datenbus
 - ▶ Taktung von Digitalrechnern (Taktleitung)

Schaltnetzentwurf: Begriffe

Begriff	Definition	Beispiele
Variable		x_1, x_7, x_{42}
Literal	(negierte) Variable	$x_1, \overline{x_4}, x_9$
Monom	Konjunktion von Literalen	$x_2 \overline{x_7}, x_3 x_4, x_{42}$
Polyonom	Disjunktion von Monomen	$x_2 \overline{x_7} \vee x_3 x_4, x_{42}$
Implikant	„Implikant(x)=1 \Rightarrow $f(x) = 1$ “	Monom eines Polynoms
Verkürzung echte	„enthält Teilmenge der Literale“	$x_3 x_4$ von $x_3 x_4 x_7$, x_1 von x_1
Verkürzung	„enthält echte Teilmenge der L.“	$x_3 x_4$ von $x_3 x_4 x_7$
Prim- implikant	Implikant, keine echte Verkürzung auch Implikant	
Minimal- polynom	Polynom mit minimalen Kosten	
Erinnerung	Minimalpolynome enthalten nur Primimplikanten	
Erinnerung	zweischrittiges Vorgehen	
	① Berechne Menge aller PI.	
	② Berechne minimale überdeckende Auswahl.	

Primimplikantenbestimmung mit KV-Diagramm



Minimalpolynom $x_1 \overline{x_2} \vee \overline{x_1} \overline{x_4}$

Algorithmus von Quine/McCluskey: PI-Tafel erstellen

bekannt $f: \{0, 1\}^4 \rightarrow \{0, 1\}$

x_1 x_2 x_3 x_4	$f(x_1, x_2, x_3, x_4)$
0000	0
0001	0
0010	0
0011	1
0100	0
0101	0
0110	0
0111	0
1000	0
1001	0
1010	0
1011	1
1100	0
1101	0
1110	0
1111	1

PI-Tafel

	0011	1011	1111
x_1 x_3 x_4		1	1
$\overline{x_2}$ x_3 x_4	1	1	

Algorithmus von Quine/McCluskey: PI-Tafel verkleinern

$x_1 x_2 x_4$								1	1		1	1
$x_1 x_3 x_4$						1	1				1	1
$\overline{x_2} x_3 x_4$			1	1				1	1			
$x_1 x_2 x_5$								1		1	1	1
$x_1 x_3 x_5$							1	1			1	1
$\overline{x_2} x_3 x_5$		1		1				1	1			
$\overline{x_4} x_5$	1	1			1	1	1	1		1		

► **Streichung von Kernimplikanten-Zeilen**

Spalte s mit nur einer 1: Wähle Primimplikant der korrespondierenden Zeile, streiche diese Zeile und alle von ihr überdeckten Spalten.

► **Streichung überdeckender Spalten**

Spalten s, s' mit $s \geq s'$: Streiche Spalte s .

► **Streichung überdeckter Zeilen**

Zeilen z, z' mit $z \geq z'$: Streiche Zeile z' .

IEEE 754-1985 — Addition

x		1	1000	0110	010	1001	0000	0000	0000	0000
y	+	0	1000	0101	111	0000	0000	0000	0000	0000

Zahl mit größerem Exponenten ist x

also vorläufiges Vorzeichenbit 1
 vorläufiger Exponent 1000 0110

Differenz der Exp. 1, also in y Komma um 1 Stelle nach links

aus x		1	,	010	1001	0000	0000	0000	0000
aus y	+	0	,	111	1000	0000	0000	0000	0000

unterschiedliche Vorzeichen, also Zweierkomplement

aus x		01	,	010	1001	0000	0000	0000	0000
aus y	+	11	,	000	1000	0000	0000	0000	0000
		1	00	,	011	0001	0000	0000	0000

zum Normalisieren Komma 2 Stellen nach rechts, also Exp. -2

1 1000 0100 100 0100 0000 0000 0000 0000