

2. Schaltnetze

2.1 Boolesche Algebra

- 2.2 Logische Schaltungen (Schaltnetz, Schaltwerk)
- 2.3 Systematische Minimierungsverfahren
- 2.3 Ausgewählte Beispiele logischer Schaltungen
- 2.4 Digitale CMOS-Grundsaltungen

Boolesche Algebra (Bode 1854, Huntingdon 1904)	Schaltalgebra, logische Algebra (Sonderfall der B.A.) (Shannon 1938)
Gegeben eine Menge M mit Elementen $M = \{a_1, a_2, a_3, \dots\}$ Zwischen den Elementen sind zweistell. Operationen definiert	
ϕ, ψ	Elemente $a_1, a_2, a_3, \dots \in \{0, 1\}$ sind logische Variable Konjunktion: UND, $\wedge$ Disjunktion: ODER, $\vee$
Durch Anwendung dieser Operatoren entstehen wiederum eindeutig Elemente aus M. Es gelten die Axiome A1...A4.	

A1 Kommutativgesetze	
$a_1 \phi a_2 = a_2 \phi a_1$ $a_1 \psi a_2 = a_2 \psi a_1$	$a_1 \wedge a_2 = a_2 \wedge a_1$ $a_1 \vee a_2 = a_2 \vee a_1$
A2 Distributivgesetze	
$a_1 \phi (a_2 \psi a_3) = (a_1 \phi a_2) \psi (a_1 \phi a_3)$ $a_1 \psi (a_2 \phi a_3) = (a_1 \psi a_2) \phi (a_1 \psi a_3)$	$a_1 \wedge (a_2 \vee a_3) = (a_1 \wedge a_2) \vee (a_1 \wedge a_3)$ $= a_1 a_2 \vee a_1 a_3$ $a_1 \vee (a_2 \wedge a_3) = (a_1 \vee a_2) \wedge (a_1 \vee a_3)$ $= (a_1 \vee a_2) (a_1 \vee a_3)$
	UND $\wedge$ vor ODER $\vee$ $\wedge$ weglassen
2.1 Kurze Einführung der Logischen Algebra 2	
GIV II 2-3	

A3 Neutrales Element, Eins-Element Es existieren für jede der beiden Operationen jeweils	
$\phi, \psi : e, n$	$\wedge, \vee : 1, 0$
für welche die Verknüpfung mit einem beliebigen $a_i \in M$ wiederum a_i ergibt	
$a_i \phi e = a_i \quad \forall a_i \in M$ $a_i \psi n = a_i \quad \forall a_i \in M$	$a_i \wedge 1 = a_i$ $a_i \vee 0 = a_i$
A4 Komplement Es existiert zu jedem Element $a_i \in M$ auch das Komplement $\bar{a}_i \in M$	
$a_i \phi \bar{a}_i = n$ $a_i \psi \bar{a}_i = e$	$a_i \wedge \bar{a}_i = 0$ $a_i \vee \bar{a}_i = 1$
2.1 Kurze Einführung der Boolesche Algebra 3	
GIV II 2-4	

Beispiele

Mengenalgebra:

ϕ : Durchschnitt

ψ : Vereinigung

Aussagenalgebra:

1: wahr

0: falsch

$$(5 \langle 9 \rangle \vee (12 = \text{Primzahl})) = 1$$

$$(2 \langle 3 \rangle \vee (5 \langle 4 \rangle)) = 0$$

$$\overline{3 \langle 5 \rangle} = 1$$

Boolesche Ausdrücke (B.A.)

- (1) 0 und 1 sind Boolesche Ausdrücke
- (2) Jede Variable x_i ist Boolescher Ausdruck
- (3) Ist A ein Boolescher Ausdruck,
dann ist auch $\overline{A}$ ein Boolescher Ausdruck
- (4) A und B seien Boolesche Ausdrücke, dann
sind auch $(A \vee B)$ und $(A \wedge B)$ Boolesche Ausdrücke
- (5) Die Menge A_N der Booleschen Ausdrücke
in den Variablen $x_1, x_2, \dots, x_n$ wird
durch die Regeln (1) - (4) beschrieben.

Boolesche Funktionen (B.F.)

- Auswertung stets für Belegungen b_i (b_i sind Werte der Variablen x_i)

$$b_i \text{ mit } x_i = b_i \quad i = 1, 2, \dots, n \quad b_i \in \{0, 1\}$$

- für n Variable existieren
 2^n Belegungen (0,1-Muster)

n		
x_1	x_2	x_3
0	0	0
0	0	1
.	.	.
1	1	1

2^n

- n -stellige Boolesche (logische) Funktion definiert durch B.A. mit
 n Variablen als Abbildung f auf die Menge der Belegungen

$$y = f(x_1, x_2, \dots, x_n) \quad x_i, y \in \{0, 1\}$$

2.1 Boolesche Algebra

GIV II 2-7

Anzahl möglicher Boolescher Funktionen bei n Variablen

1 Variable: $2^{2^1} = 4$ Funktionen:

- f_0 : Nullfunktion
- f_1 : Negation
- f_2 : Identität
- f_3 : Einsfunktion

x	f_0	f_1	f_2	f_3
0	0	0	1	1
1	0	1	0	1

2 Variable: $2^{2^2} = 16$ Funktionen:

- f_0 : Nullfunktion
- f_1 : UND
- .
- f_{15} : Einsfunktion

2		2^{2^n}										
a	b	f_0	f_1	f_2		f_{15}						
0	0	0	0	.		1						
0	1	0	0	.		1						
1	0	0	0	.		1						
1	1	0	1	.		1						

2.1 Boolesche Algebra

GIV II 2-8

Werte- tabelle a 0101 b 0011	Boolesche Funktion Verknüpfungstabelle		Bezeichnung Aussagen- algebra	alternative Bezeichnung verbale Beschreibung	andere Ver- knüpfungs- symbole	Venn- diagramme a·b
0 0 0 0	$f_0 = 0$		Kontradiktion	Nullfunktion (Nie)		
0 0 0 1	$f_1 = a \wedge b$		Konjunktion	UND	$a \cdot b, a \& b$	
0 0 1 0	$f_2 = \bar{a} \wedge b$	$b \rightarrow a$	Inhibition	wenn b dann $\bar{a}$		
0 0 1 1	$f_3 = b$		Identität			
0 1 0 0	$f_4 = a \wedge \bar{b}$	$a \rightarrow b$	Inhibition	wenn a dann $\bar{b}$		
0 1 0 1	$f_5 = a$		Identität			
0 1 1 0	$f_6 = (a \wedge \bar{b}) \vee (\bar{a} \wedge b)$	$a \neq b$	Antivalenz	Exklusiv ODER	$a \oplus b$	

0 1 1 1	$f_7 = a \vee b$		Disjunktion	ODER	$a + b$	
1 0 0 0	$f_8 = \overline{a \vee b}$		Peircefunktion	NOR	$a \downarrow b, a * b$	
1 0 0 1	$f_9 = (a \vee \bar{b}) \wedge (\bar{a} \vee b)$	$a \equiv b$	Äquivalenz	gleichwertig	$a \leftrightarrow b$	
1 0 1 0	$f_{10} = \bar{a}$		Negation		- a	
1 0 1 1	$f_{11} = \bar{a} \vee b$	$a \rightarrow b$	Implikation	Wenn a dann b	$a \supset b$	
1 1 0 0	$f_{12} = \bar{b}$		Negation		- b	
1 1 0 1	$f_{13} = a \vee \bar{b}$	$b \rightarrow a$	Implikation	Wenn b dann a	$b \supset a$	
1 1 1 0	$f_{14} = \overline{a \wedge b}$		Shefferfunktio n	NAND	$a b$	
1 1 1 1	$f_{15} = 1$		Einsfunktion	Tautologie (immer)		

Rechenregeln, die Axiome selbst

Kommutativgesetze $A \wedge B = B \wedge A$
 $A \vee B = B \vee A$

Distributivgesetze $A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C) = A \wedge B \vee A \wedge C$
 $A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$

Einselement $A \wedge 1 = A$
 $A \vee 0 = A$

Komplementbildung $A \vee \bar{A} = 1$
 $A \wedge \bar{A} = 0$

Vereinbarungen:

"Negation vor Konjunktion
 vor Disjunktion"

$\wedge$ häufig weggelassen
 z.B. $A \wedge B = AB$

Rechenregeln, aus den Axiome selbst abgeleitet:

Assoziativgesetze $A \vee (B \vee C) = (A \vee B) \vee C$
 $A \wedge (B \wedge C) = (A \wedge B) \wedge C$

Gesetz von de Morgan $\overline{A \vee B} = \bar{A} \wedge \bar{B}$
 $\overline{A \wedge B} = \bar{A} \vee \bar{B}$

Absorptionsgesetze $A \wedge (A \vee B) = A$
 $A \vee (A \wedge B) = A$

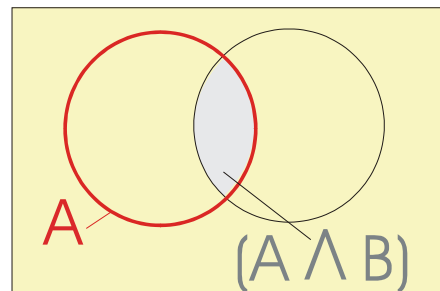
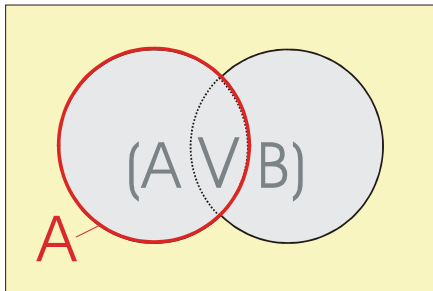
Idempotenzgesetze $A \vee A = A$
 $A \wedge A = A$

Reduktionsformeln $A \vee (\bar{A} \wedge B) = (A \vee \bar{A}) \wedge (A \vee B) = A \vee B$
 $A \wedge (\bar{A} \vee B) = (A \wedge \bar{A}) \vee (A \wedge B) = A \wedge B$

Zur Erläuterung der
Absorptionsgesetze

$$A \wedge (A \vee B) = A$$

$$A \vee (A \wedge B) = A$$



Minterme M_k^i	Maxterme M_k^i
Vollkonjunktion alle (n) Variablen (negiert oder nicht negiert) in konjunktiver Verknüpfung	Volldisjunktion alle (n) Variablen (negiert oder nicht negiert) in disjunktiver Verknüpfung
2^n Minterme	2^n Maxterme
Beispiel $\bar{a}_1 \wedge a_2 \wedge \bar{a}_3 \wedge a_4 \dots \wedge \bar{a}_n$	Beispiel $a_1 \vee a_2 \vee \bar{a}_3 \vee \bar{a}_4 \dots \vee a_n$

Minterm	a b c	Maxterm	Begriffs-Erläuterung
$\bar{a} \wedge \bar{b} \wedge \bar{c}$	0 0 0	$a \vee b \vee c$	<u>Minterm</u> : Boolesche Funktion nimmt für minimal wenige Terme den Wert 1 an, für nämlich genau einen; <u>Maxterm</u> : Boolesche Funktion nimmt für maximal viele Terme den Wert 1 an, nämlich für alle außer einem.
$\bar{a} \wedge \bar{b} \wedge c$	0 0 1	$a \vee b \vee \bar{c}$	
$\bar{a} \wedge b \wedge \bar{c}$	0 1 0	$a \vee \bar{b} \vee c$	
$\bar{a} \wedge b \wedge c$	0 1 1	$a \vee \bar{b} \vee \bar{c}$	
$a \wedge \bar{b} \wedge \bar{c}$	1 0 0	$\bar{a} \vee b \vee c$	
$a \wedge \bar{b} \wedge c$	1 0 1	$\bar{a} \vee b \vee \bar{c}$	
$a \wedge b \wedge \bar{c}$	1 1 0	$\bar{a} \vee \bar{b} \vee c$	
$a \wedge b \wedge c$	1 1 1	$\bar{a} \vee \bar{b} \vee \bar{c}$	

Formen Boolescher Funktionen (DF, KF)			
Disjunktive Form DF		Konjunktive Form KF	
<i>disjunktiv</i> verknüpfte <i>Konjunktionen</i> : $f(a_1, a_2, \dots, a_n) = K_1 \vee K_2 \vee \dots \vee K_m$		<i>konjunktiv</i> verknüpfte <i>Disjunktionen</i> : $f(a_1, a_2, \dots, a_n) = D_1 \wedge D_2 \wedge \dots \wedge D_m$	
K_i : mehrere konjunktiv verknüpfte Variable (negiert, nicht negiert)		D_i : mehrere disjunktiv verknüpfte Variable (negiert, nicht negiert)	

Formen Boolescher Funktionen (DNF, KNF)	
Disjunktive Normalform DNF	Disjunktive Normalform KNF
<i>disjunktiv verknüpfte Minterme:</i> $f(a_1, a_2, \dots, a_n) = M_1^i \vee M_2^i \vee \dots \vee M_k^i$	<i>konjunktiv verknüpfte Maxterme:</i> $f(a_1, a_2, \dots, a_n) = M_1^a \wedge M_2^a \wedge \dots \wedge M_k^a$
es treten in der DNF genau die Minterme auf, an denen die zugehörigen Belegungen den Funktionswert 1 liefern	es treten in der KNF genau die Maxterme auf, an denen die zugehörigen Belegungen den Funktionswert 0 liefern
jede B.F. lässt sich eindeutig durch ihre DNF oder KNF beschreiben	
2.1 Boolesche Algebra GIV II 2-17	

Die DF bzw. KF erhält man durch

- Ersetzen der Operatoren $\neq, \equiv, \dots$ durch $\wedge, \vee$ und Negation
- wiederholte Anwendung der Regeln von de Morgan bis Negationen nur über einzelnen Variablen auftreten
- wiederholte Anwendung des jeweiligen Distributivgesetzes

DNF aus DF:

- konjunktives Hinzufügen bei jeder Konjunktion aller nicht auftretender Variablen in der Form $a_i \vee \bar{a}_i = 1$
- anschließendes "ausmultiplizieren" (Distributivgesetz)

$$a_1 a_2 a_3 \vee a_1 \bar{a}_3 (a_2 \vee \bar{a}_2) = a_1 a_2 a_3 \vee a_1 a_2 \bar{a}_3 \vee a_1 \bar{a}_2 \bar{a}_3$$

KNF aus KF:

- disjunktives Hinzufügen bei jeder Disjunktion aller nicht auftretender Variablen in der Form
- anschließendes "ausaddieren" (Distributivgesetz)

$$(a_1 \vee a_2 \vee a_3) \wedge (a_1 \vee \bar{a}_3) (a_2 \wedge \bar{a}_2) = \\ = (a_1 \vee a_2 \vee a_3) \wedge (a_1 \vee a_2 \vee \bar{a}_3) \wedge (a_1 \vee \bar{a}_2 \vee \bar{a}_3)$$

2.1 Boolesche Algebra

GIV II 2-19

a ₁	a ₂	a ₃	f	Beispiel für eine KNF
0	0	0	0	
0	0	1	0	
0	1	0	1	
0	1	1	1	
1	0	0	0	
1	0	1	1	
1	1	0	0	
1	1	1	0	

2.1 Boolesche Algebra, Beispiel für eine KNF

GIV II 2-20

a_1	a_2	a_3	f	Beispiel für eine KNF
0	0	0	0	$f = (a_1 \vee a_2 \vee a_3) \wedge$
0	0	1	0	$(a_1 \vee a_2 \vee \bar{a}_3) \wedge$
0	1	0	1	Konjunktive Verknüpfung der Maxterme, an denen $f=0$
0	1	1	1	
1	0	0	0	$(\bar{a}_1 \vee a_2 \vee a_3) \wedge$
1	0	1	1	
1	1	0	0	$(\bar{a}_1 \vee \bar{a}_2 \vee a_3) \wedge$
1	1	1	0	$(\bar{a}_1 \vee \bar{a}_2 \vee \bar{a}_3)$

2.1 Boolesche Algebra, Beispiel für eine KNF

GIV II 2-21

Beispiel für eine DF

$$\begin{aligned}
 f(a_1, a_2, a_3, a_4) &= a_1 a_2 \bar{a}_4 \vee a_2 a_3 \bar{a}_4 \vee \underbrace{a_1 a_3 a_4 \vee a_3 a_4}_{=1} \\
 &= a_1 a_2 \bar{a}_4 \vee a_2 a_3 \bar{a}_4 \vee a_3 a_4 \underbrace{(a_1 \vee 1)}_{=1}
 \end{aligned}$$

2.1 Boolesche Algebra, Beispiel für eine DF

GIV II 2-22

a_1	a_2	a_3	a_4	f	a_1	a_2	a_3	a_4	f
0	0	0	0	0	1	0	0	0	0
0	0	0	1	0	1	0	0	1	0
0	0	1	0	0	1	0	1	0	0
0	0	1	1	1	1	0	1	1	1
0	1	0	0	0	1	1	0	0	1
0	1	0	1	0	1	1	0	1	0
0	1	1	0	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1

2.1 Boolesche Algebra, Beispiel für eine DNF

GIV II 2-23

a_1	a_2	a_3	a_4	f	$f_{DNF} =$	a_1	a_2	a_3	a_4	f	...
0	0	0	0	0	Disjunktive Verknüpfung der Minterme, an denen $f=1$	1	0	0	0	0	
0	0	0	1	0		1	0	0	1	0	
0	0	1	0	0		1	0	1	0	0	
0	0	1	1	1		1	0	1	1	1	$a_1 \bar{a}_2 a_3 a_4 \vee$
0	1	0	0	0		1	1	0	0	1	$a_1 a_2 \bar{a}_3 \bar{a}_4 \vee$
0	1	0	1	0		1	1	0	1	0	
0	1	1	0	1	$\bar{a}_1 a_2 a_3 \bar{a}_4 \vee$	1	1	1	0	1	$a_1 a_2 a_3 \bar{a}_4 \vee$
0	1	1	1	1	$\bar{a}_1 a_2 a_3 a_4 \vee$	1	1	1	1	1	$a_1 a_2 a_3 a_4 \cdot$

$$f_{DNF} = a_1 a_2 \bar{a}_4 \underbrace{(a_3 \vee \bar{a}_3)}_1 \vee a_2 a_3 \bar{a}_4 \underbrace{(a_1 \vee \bar{a}_1)}_1 \vee a_3 a_4 \underbrace{(a_1 \vee \bar{a}_1)}_1 \underbrace{(a_2 \vee \bar{a}_2)}_1 = f_{DNF}$$

2.1 Boolesche Algebra, Beispiel für eine DNF

GIV II 2-24

Partiell definierte Boolesche Funktionen

- haben für ausgewählte (nicht alle) Belegungen Funktionswerte
- besitzen nicht definierte Belegungen mit unerheblichen Funktionswerten: *"Don't-Cares"*, gekennzeichnet durch X
- benötigen Ergänzungen der Wahrheitstabellen !

Don't-Care-Plätzen kann man beliebige Funktionswerte zuweisen!

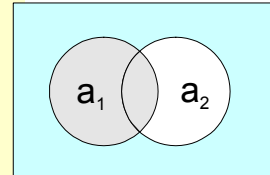
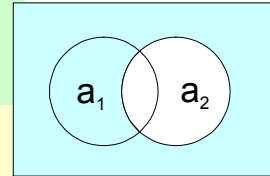
- geschickte Nutzung der Don't-Care-Plätze vereinfacht Funktion
- aber intuitives Einsetzen ist unbefriedigend, systematisches Vorgehen ist erforderlich !

a_1	a_2	a_2	x_1	x_2	
0	0	0	0	0	
0	0	1	0	0	
0	1	0	0	0	$x_1 = a_1 \bar{a}_2 \bar{a}_3 \vee a_1 a_2 \bar{a}_3$
0	1	1	0	0	$x_2 = a_1$
1	0	0	1	1	
1	0	1	X	1	
1	1	0	1	1	
1	1	1	X	1	

Beispiel 1 zu Don't-Cares

a ₁	a ₂	a ₂	f ₁	f ₂	f ₃	
0	0	1	1			$f_1 = \bar{a}_1 \bar{a}_2 a_3 \vee a_1 \bar{a}_2 \bar{a}_3$
0	1	0	0			
1	0	0	1			$f_3 = \bar{a}_2$
a ₁	a ₂	a ₃				
0	0	0	X	1	1	$f_2 = \bar{a}_1 \bar{a}_2 \bar{a}_3 \vee$
0	0	1	1	1	1	$\bar{a}_1 \bar{a}_2 a_3 \vee$ } $\bar{a}_1 \bar{a}_2$
0	1	0	0	0	0	
0	1	1	X	0	0	
1	0	0	1	1	1	$a_1 \bar{a}_2 \bar{a}_3 \vee$ } $a_1 \bar{a}_2$
1	0	1	X	1	1	$a_1 \bar{a}_2 a_3 \vee$ } a_1
1	1	0	X	1	0	$a_1 a_2 \bar{a}_3 \vee$ } $a_1 a_2$
1	1	1	X	1	0	$a_1 a_2 a_3$

$$f_3 = \bar{a}_2$$



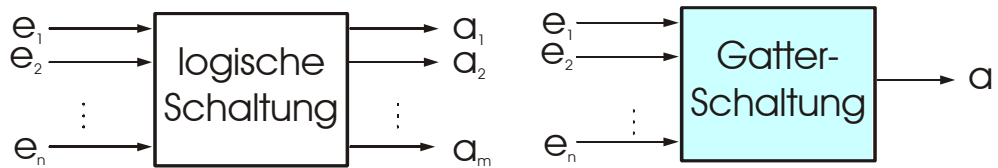
$$f_2 = a_1 \vee \bar{a}_1 \bar{a}_2$$

2.1 Boolesche Algebra, Beispiel 2 zu Don't-Cares

GIV II 2-27

2. Schaltnetze

- 2.1 Boolesche Algebra
- 2.2 Logische Schaltungen (Schaltnetz, Schaltwerk)
- 2.3 Systematische Minimierungsverfahren
- 2.3 Ausgewählte Beispiele logischer Schaltungen
- 2.4 Digitale CMOS-Grundsaltungen



Schaltwerk

- realisiert sequentielle Bool. Algorithmen
- besitzt Gedächtnis und Rückkopplung

Schaltnetz / Gatterschaltung

- realisiert Boolesche Funktion(en)
- bzw. Bündel Boolescher Funktionen

$$\underline{a} = f(\underline{e}) \quad \text{mit}$$

$$a_1 = f_1(e_1, e_2, \dots, e_n)$$

$$a_2 = f_2(e_1, e_2, \dots, e_n)$$

$$\vdots \quad \quad \quad \vdots$$

$$a_m = f_m(e_1, e_2, \dots, e_n)$$

2.2 Logische Schaltungen

GIV II 2-29

Bezeichnung	Schaltsymbole nach DIN 40700 seit 1976	früher	amerik. Symbole	Funktion	Bemerkung
UND	&			$e_1 \wedge e_2 \dots \wedge e_n$	$e_1 \oplus e_2 \dots \oplus e_n$ (EXOR) (EXNOR)
ODER	≥ 1			$e_1 \vee e_2 \dots \vee e_n$	
Anti- valenz	$= 1$			$e_1 \nabla e_2 \dots \nabla e_n$	
Äqui- valenz	$= 1$			$e_1 \oplus e_2 \dots \oplus e_n$	
NOR	≥ 1			$\overline{e_1 \vee e_2 \dots \vee e_n}$	
NAND	&			$\overline{e_1 \wedge e_2 \dots \wedge e_n}$	
Nega- tion	1			$\bar{e}$	
log. Schltg.	Gl.			$\bar{e}_1 \wedge \bar{e}_2 \wedge e_3$	

2.2 Logische Schaltungen

GIV II 2-30

vollständige Verknüpfungsmengen:

- Darstellbarkeit aller Booleschen Funktionen
- übliche Operatoren: $\{\vee, \wedge, \neg\}$
- Alternative z.B.: $\{\neq, \wedge, \neg\}$
- Vollständige Verknüpfungsmengen mit nur *einem* Operator sind **NOR** oder **NAND**. Letztere sind besonders zweckmäßig, da so komplexe Schaltungen mit einem Gattertyp realisierbar sind

2. Schaltnetze

- 2.1 Boolesche Algebra
- 2.2 Logische Schaltungen (Schaltnetz, Schaltwerk)
- 2.3 **Systematische Minimierungsverfahren**
- 2.3 Ausgewählte Beispiele logischer Schaltungen
- 2.4 Digitale CMOS-Grundsaltungen

Systematische Minimierungsverfahren

- Teil des Entwurfes logischer Schaltungen
- Aufgabenstellung großer Bedeutung (Aufwand !)
- Konzepte zur Minimierung
 - algebraische Verfahren
 - graphische Verfahren
 - tabellarische, algorithmische Verfahren
- Prinzipiell werden Gesetze der Booleschen Algebra angewandt:

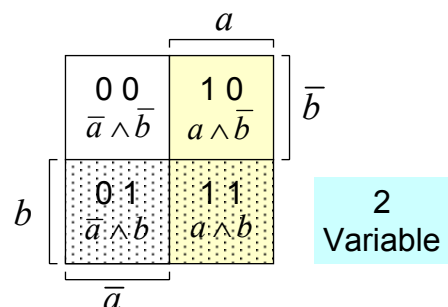
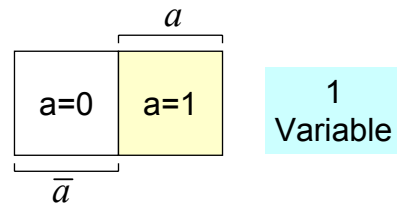
$$\begin{aligned}
 y &= \bar{a}\bar{b}c \vee a\bar{b}c \vee \bar{a}b\bar{c} \vee \bar{a}bc \vee a\bar{b}\bar{c} \\
 &= (\bar{a}\bar{b}c \vee \bar{a}bc) \vee (\bar{a}b\bar{c} \vee \bar{a}b\bar{c}) \vee (a\bar{b}c \vee a\bar{b}\bar{c}) \\
 &= (\bar{a}c)(b \vee \bar{b}) \vee (\bar{a}b)(c \vee \bar{c}) \vee (a\bar{b})(c \vee \bar{c}) \\
 &= (\bar{a}c) \vee (\bar{a}b) \vee (a\bar{b})
 \end{aligned}$$

2.3 Systematische Minimierungsverfahren

GIV II 2-33

KV-Diagramme 1

- grafische Darstellung Boolescher Funktionen
- von Karnaugh und Veitch (50er Jahre)
- n-stellige Boolesche Funktion:
 - 2^n Belegungen und 2^n Funktionswerte
 - 2^n quadratische Felder im Rechteckschema
 - jedem Feld ist eine Belegung zugeordnet
 - jedes Feld repräsentiert Minterm, bzw. Maxterm

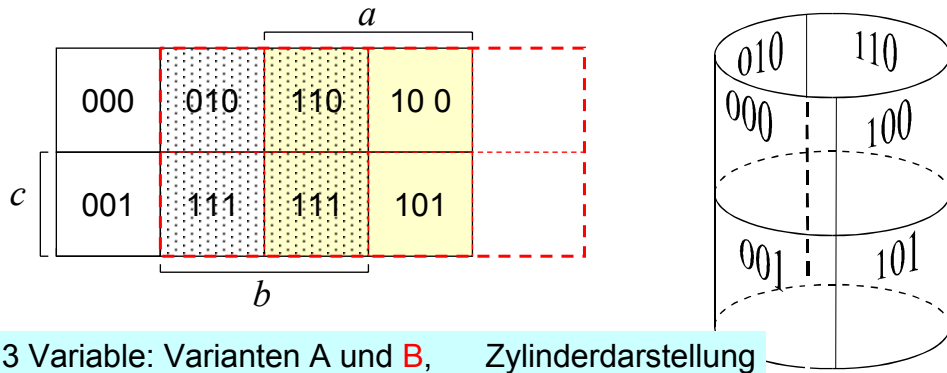


2.3 Systematische Minimierungsverfahren

GIV II 2-34

KV-Diagramme 2

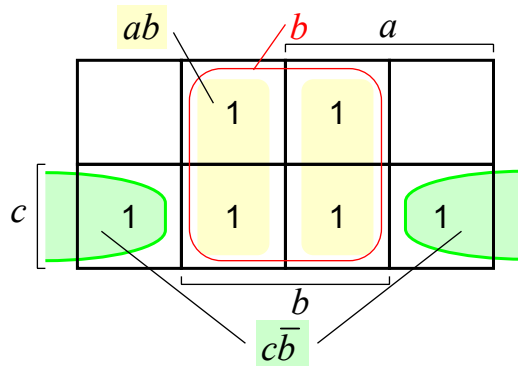
- Verdopplung für jede weitere Variable
- Varianten der Feldanordnung möglich
- math. Nachbarschaftsdefinition: Unterschied in einer Variablen
- Minimierung durch Zusammenfassen benachbarter Felder

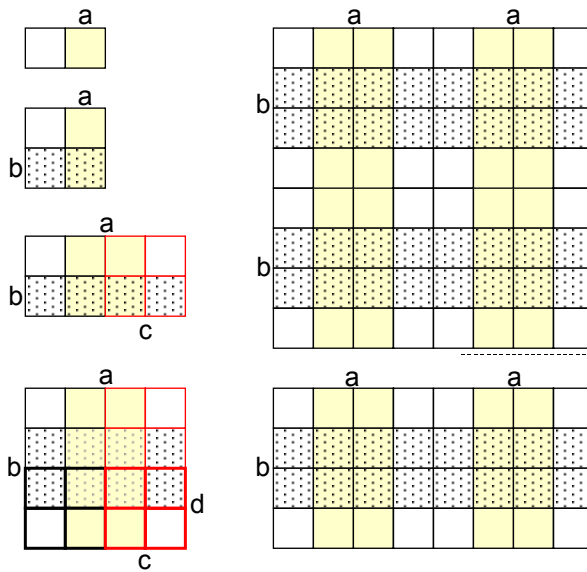


3 Variable: Varianten A und B, Zylinderdarstellung

KV-Diagramme 3

- algebraische Minimierung $abc \vee ab\bar{c} = ab \left(\underbrace{c \vee \bar{c}}_1 \right) = ab$
- entspricht einer Zusammenfassung von Nachbarschaften mit gleichen Werten 1 oder 0 $\Rightarrow$ die Zusammenfassung
- möglichst großer Blöcke von 2^n Feldern, die z.B. mit 1 besetzt sind:





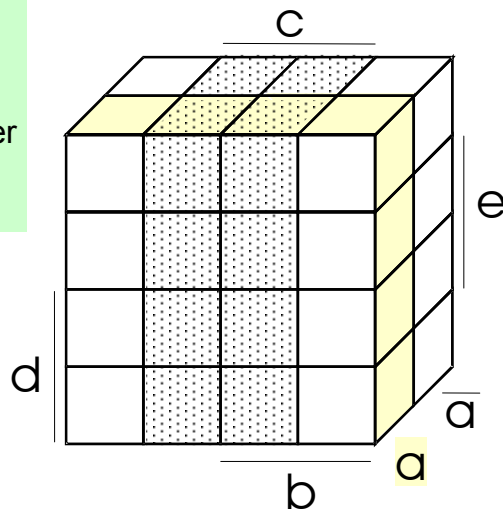
Entwicklungsschema
für KV-Diagramme

Achtung:
ab 5 Variablen sind
"benachbarte" Felder
örtlich getrennt

2.3 Systematische Minimierungsverfahren

GIV II 2-37

"Rettung" des örtlichen
Zusammenliegens
auseinanderfallender
mathematisch benachbarter
Bereiche durch räumliche
KV-Darstellung



2.3 Systematische Minimierungsverfahren

GIV II 2-38

Wertetabelle	Dezimale Feldbeschriftung	Eins - Muster	Null - Muster																								
n a b c f 0 0 0 0 0 1 0 0 1 1 2 0 1 0 1 3 0 1 1 0 4 1 0 0 X 5 1 0 1 0 6 1 1 0 0 7 1 1 1 1	a <table border="1" style="border-collapse: collapse; text-align: center;"> <tr><td>0</td><td>2</td><td>6</td><td>4</td></tr> <tr><td>1</td><td>3</td><td>7</td><td>5</td></tr> </table> b	0	2	6	4	1	3	7	5	a <table border="1" style="border-collapse: collapse; text-align: center;"> <tr><td></td><td>1</td><td></td><td>X</td></tr> <tr><td>1</td><td></td><td>1</td><td></td></tr> </table> b		1		X	1		1		a <table border="1" style="border-collapse: collapse; text-align: center;"> <tr><td>0</td><td></td><td>0</td><td>X</td></tr> <tr><td></td><td>0</td><td></td><td>0</td></tr> </table> b	0		0	X		0		0
0	2	6	4																								
1	3	7	5																								
	1		X																								
1		1																									
0		0	X																								
	0		0																								

Übertragen einer Wahrheitstabelle in ein KV-Diagramm:

- logische Auswertung jedes (Min/Max)-Termes
- einfach über dezimal beschriftete Felder

Boolesche Ausdrücke direkt über logische Auswertung alternativ über Erstellung einer Wahrheitstabelle

2.3 Systematische Minimierungsverfahren

GIV II 2-39

Gegeben DNF:

Übertragung
jedes Minterms
in *ein* Feld des
KV-Diagramms

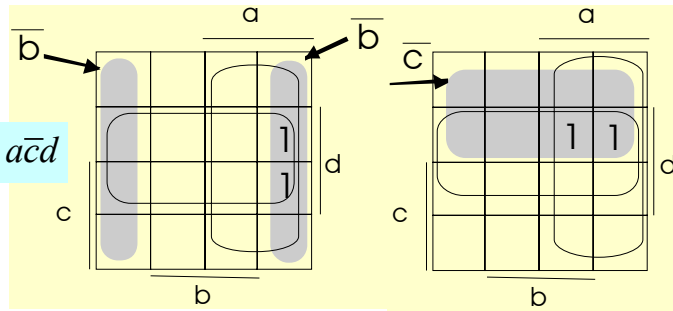
		<u>a</u>			
		0	2	6	4
				1	1
c		1	3	7	5
		<u>b</u>			

$$f(a, b, c) = ab\bar{c} \vee a\bar{b}\bar{c} \vee \bar{a}bc$$

2.3 Systematische Minimierungsverfahren

GIV II 2-40

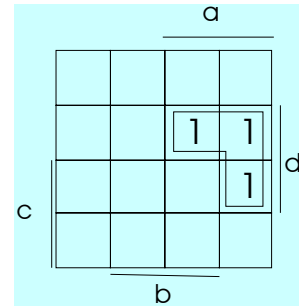
$$f(a, b, c, d) = a\bar{b}d \vee a\bar{c}d$$



Gegeben DF:

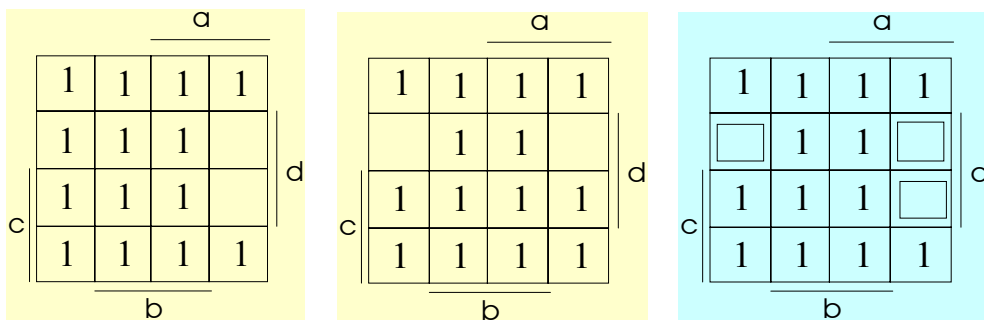
Jede Konjunktion kennzeichnet eine Gruppe von Feldern

Gesamtdarstellung durch
Zusammenfassung der Feldgruppen
(disjunktive Verknüpfung der Konjunktionen)



2.3 Systematische Minimierungsverfahren

GIV II 2-41



$$f(a, b, c, d) = (\bar{a} \vee b \vee \bar{d}) \wedge (b \vee c \vee \bar{d})$$

Gegeben KF, direkte KV-Diagramm-Erstellung:

Jede Disjunktion kennzeichnet eine Vereinigungsmenge von Feldern

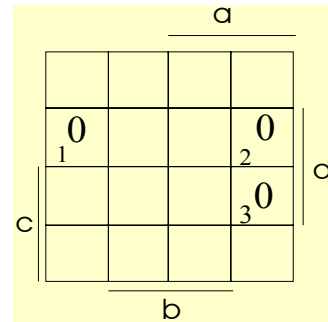
Gesamtdarstellung durch Schnittmengen der Feldergruppen
(konjunktive Verknüpfung der Disjunktionen)

2.3 Systematische Minimierungsverfahren

GIV II 2-42

$$f(a,b,c,d) = (\bar{a} \vee b \vee \bar{d}) \wedge (b \vee c \vee \bar{d})$$

$$\bar{f} = (\quad) \vee (\quad) = a\bar{b}d \vee \bar{b}\bar{c}d$$



Gegeben KF, KV-Diagramm über dualen Ausdruck:

Übersichtlicherer Weg: dualen Ausdruck bilden (Negation) $\Rightarrow$ DF, deren Minterme adressieren die *Nullen* der Booleschen Funktion

2.3 Systematische Minimierungsverfahren

GIV II 2-43

	x_2			
	0	2	6	4
x_0	1	3	7	5
	x_1			

Standard KV-Diagramme für diese Veranstaltung (x_0 LSB)

3 Variable

	x_2				x_2			
	0	4	12	8	24	28	20	16
	1	5	13	9	25	29	21	17
	3	7	15	11	27	31	23	19
	2	6	14	10	26	30	22	18
x_1	x_3				x_3			
	x_4				x_4			

5 Variable

2.3 Systematische Minimierungsverfahren

GIV II 2-44

	x_3			
	0	4	12	8
	1	5	13	9
x_1	3	7	15	11
	2	6	14	10
	x_2			

3 Variable

Standard KV-Diagramme
für diese Veranstaltung
(x_0 LSB)

6 Variable

x_3							
0	4	12	8	24	28	20	16
1	5	13	9	25	29	21	17
3	7	15	11	27	31	23	19
2	6	14	10	26	30	22	18
34	38	46	42	58	62	54	50
35	39	47	43	59	63	55	51
33	37	45	41	57	61	53	49
32	36	44	40	56	60	52	48
x_2				x_2			
x_4							
x_1	x_0						
	x_0						
	x_0						
	x_0						
	x_0						
x_0							
x_0							
x_0							

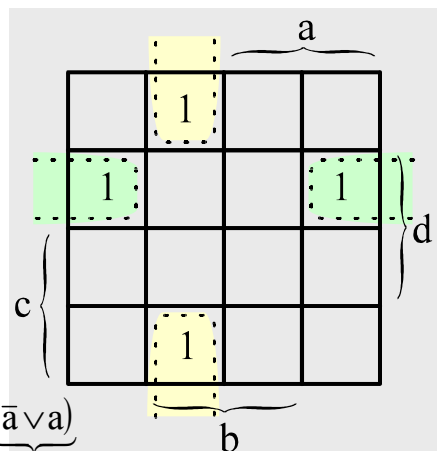
2.3 Systematische Minimierungsverfahren

GIV II 2-45

$$f = (\bar{a} \wedge b \wedge \bar{c} \wedge \bar{d}) \vee (\bar{a} \wedge \bar{b} \wedge \bar{c} \wedge d) \vee (a \wedge \bar{b} \wedge \bar{c} \wedge d) \vee (\bar{a} \wedge b \wedge c \wedge \bar{d})$$

Minimierung mit KV-Diagrammen:

Gruppen mit 2^n gleich besetzten
Feldern können zu einem
vereinfachten Ausdruck
zusammengefasst werden

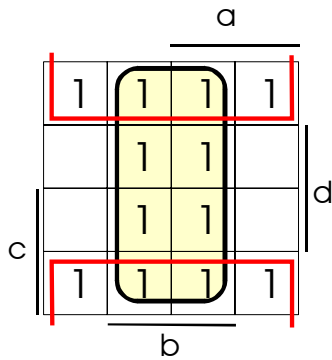


$$f = (\bar{a} \wedge b \wedge \bar{d}) \wedge \underbrace{(\bar{c} \vee c)}_1 \vee (\bar{b} \wedge \bar{c} \wedge d) \wedge \underbrace{(\bar{a} \vee a)}_1$$

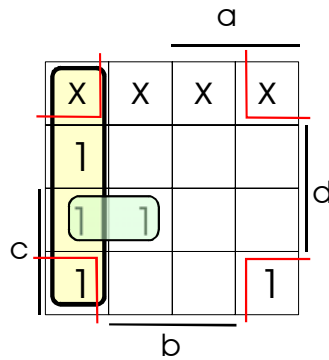
$$= \bar{a} \wedge b \wedge \bar{d} \vee \bar{b} \wedge \bar{c} \wedge d$$

2.3 Systematische Minimierungsverfahren

GIV II 2-46



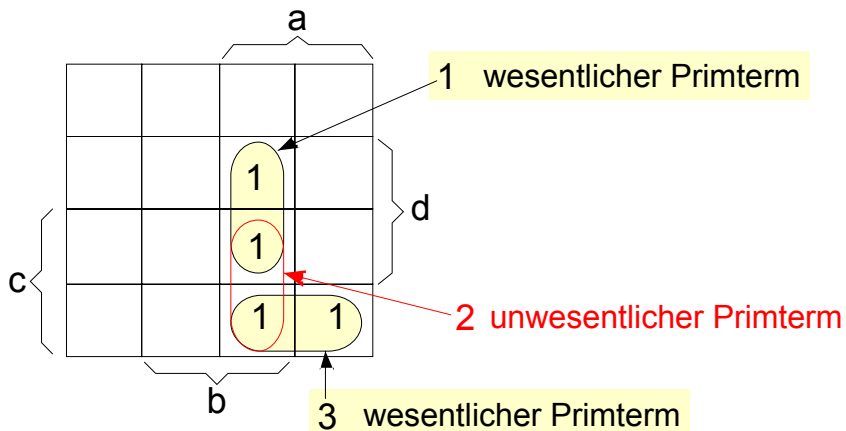
$$f(a,b,c,d) = \boxed{b} \vee \boxed{\bar{d}}$$



$$f(a,b,c,d) = \boxed{\bar{a}\bar{b}} \vee \boxed{\bar{a}cd} \vee \boxed{\bar{b}d}$$

Minimierung mit KV-Diagrammen:

- je größer die Blöcke, umso einfacher der beschreibende Ausdruck
- Don't -Care-Plätze sind so mit 1(0) zu besetzen, dass möglichst große zusammenfassbare 1(0) Blöcke entstehen



Algorithmische Minimierung nach Quine-McCluskey

- systematische "Primterm"-Suche für höhere Variablenanzahlen
- Primterm: nicht weiter zu vereinfachende Konjunktion
- Eliminierung unwesentlicher (durch andere abgedeckter) Primterme

Algorithmische Minimierung nach Quine-McCluskey 1

- Wahrheitstabelle bilden
- Mintermtabelle aufstellen
- Sortierung nach Anzahl von Einsen
- Vergleich der Terme benachbarter Gruppen und Erkennung von Termen mit nur einer unterschiedlichen Variablen
- Zusammenfassung durch Streichen dieser unterschiedlichen Variablen
- Neue Sortierung in Gruppen und Vergleich der Terme, Vereinfachung etc.
- bis keine Vereinfachung mehr geht (Primterme !)

Nr.	a	b	c	d	y
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	0

2.3 Systematische Minimierungsverfahren

GIV II 2-49

Nr.	a b c d	zusammengefasste Terme 1. Ordnung	a b c d	zusammengefasste Terme 2. Ordnung	a b c d
1	0 0 0 1	✓ 1,5	0 - 0 1 A	2,6;10,14	- - 1 0 B
2	0 0 1 0	✓ 2,6	0 - 1 0	✓ 2,10;6,14	- - 1 0 (doppelt)
4	0 1 0 0	✓ 2,10	- 0 1 0	✓ 4,5;12,13	- 1 0 - C
5	0 1 0 1	✓ 4,5	0 1 0 -	✓ 4,6;12,14	- 1 - 0 D
6	0 1 1 0	✓ 4,6	0 1 - 0	✓ 4,12;5,13	- 1 0 - (doppelt)
10	1 0 1 0	✓ 4,12	- 1 0 0	✓ 4,12;6,14	- 1 - 0 (doppelt)
12	1 1 0 0	✓ 5,13	- 1 0 1	✓	
13	1 1 0 1	✓ 6,14	- 1 1 0	✓	
14	1 1 1 0	✓ 10,14	1, 1 0	✓	
		12,13	1 1 0 -	✓	
		12,14	1 1 - 0	✓	

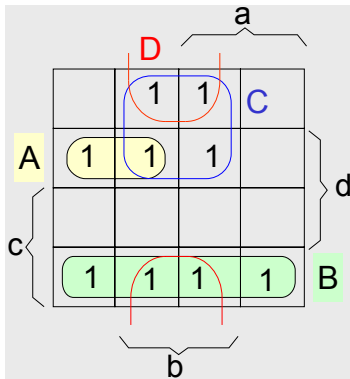
Algorithmische Minimierung nach Quine-McCluskey 2

2.3 Systematische Minimierungsverfahren

GIV II 2-50

Algorithmische Minimierung nach Quine-McCluskey 3

- Verschiedene Primterme können gleiche Minterme enthalten
- Primterme deren erzeugende Minterme alle anderweitig enthalten sind, können weggelassen werden



$$y = \bar{a}\bar{c}d \vee c\bar{d} \vee b\bar{c} \vee b\bar{d}$$

$$= \boxed{A} \vee \boxed{B} \vee C \vee \cancel{D}$$

Algorithmische Minimierung nach Quine-McCluskey 4

- Primterm-Minterm-Tabelle zur systematischen Prüfung aller Minterme bezüglich ihrer Abdeckung in den Primtermen
- Ermittlung und Streichung unwesentlicher Primterme

Minterme \ Primterme	√ 1	√ 2	√ 4	√ 5	√ 6	√ 10	√ 12	√ 13	√ 14
A: 1, 5	*			*					
2, 6; 10, 14		*			*	*			*
C: 4, 5; 12, 13			*	*			*	*	
D: 4, 6; 12, 14			*		*		*		*

2. Schaltnetze

- 2.1 Boolesche Algebra
- 2.2 Logische Schaltungen (Schaltnetz, Schaltwerk)
- 2.3 Systematische Minimierungsverfahren
- 2.3 Ausgewählte Beispiele logischer Schaltungen**
- 2.4 Digitale CMOS-Grundsaltungen

Zweistelliger Dualzahlenvergleich 1:

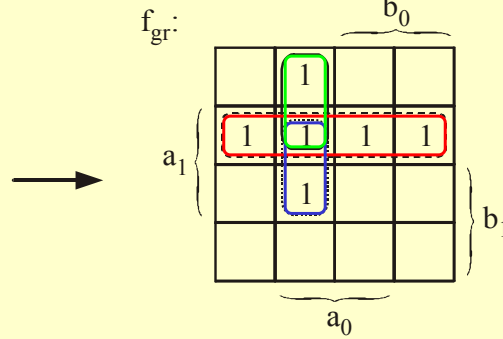
gegeben 2 natürliche Dualzahlen N_{2a} , N_{2b} :

$$N_2 = \sum_{k=0}^{K-1} z_k \cdot 2^k = z_{K-1} \cdot 2^{K-1} + \dots + z_1 \cdot 2^1 + z_0 \cdot 2^0$$

logische Aussage aus dem Vergleich:

$$\begin{array}{lll} & > & f_{gr} = (N_{2a} > N_{2b}) \\ N_{2a} & < & N_{2b} \quad \text{bzw.} \quad f_{kl} = (N_{2a} < N_{2b}) \\ & = & f_{gl} = (N_{2a} = N_{2b}) \end{array}$$

a_1	a_0	b_1	b_0	f_{gr}	f_{kl}	f_{gl}
0	0	0	0	0	0	1
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	0	1	0
0	1	0	0	1	0	0
0	1	0	1	0	0	1
0	1	1	0	0	1	0
0	1	1	1	0	1	0
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	0	1
1	0	1	1	0	1	0
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	0	1



Zweistelliger Dualzahlenvergleich 2:
Wahrheitstabelle und KV-Diagramm
für N_{2a} , N_{2b} bzw. $\{a_1, a_0\}$, $\{b_1, b_0\}$

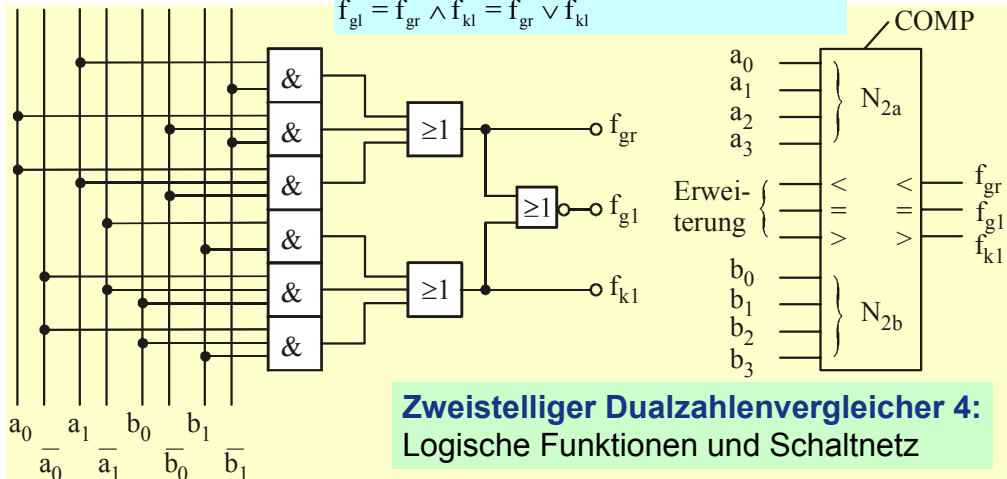
2.4 Ausgewählte Beispiele logischer Schaltungen

GIV II 2-55

$$f_{gr} = (a_1 \wedge \bar{b}_1) \vee (a_0 \wedge \bar{b}_1 \wedge \bar{b}_0) \vee (a_0 \wedge a_1 \wedge \bar{b}_0)$$

$$f_{kl} = (\bar{a}_1 \wedge b_1) \vee (b_0 \wedge \bar{a}_1 \wedge \bar{a}_0) \vee (b_0 \wedge b_1 \wedge \bar{a}_0)$$

$$f_{gl} = \bar{f}_{gr} \wedge \bar{f}_{kl} = \overline{f_{gr} \vee f_{kl}}$$



Zweistelliger Dualzahlenvergleich 4:
Logische Funktionen und Schaltnetz

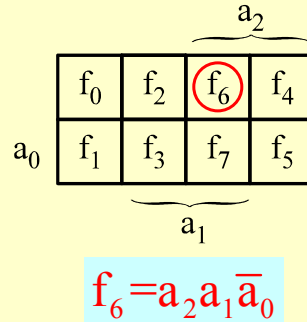
2.4 Ausgewählte Beispiele logischer Schaltungen

GIV II 2-56

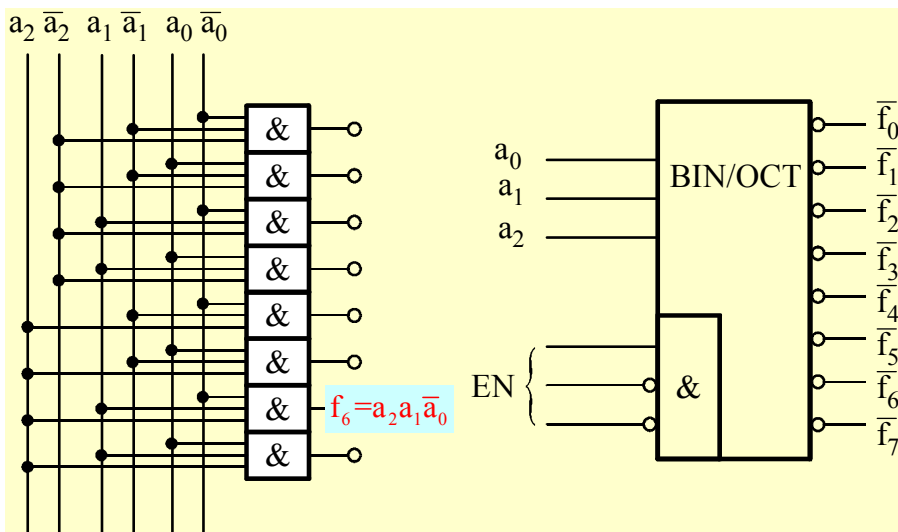
1-aus-N-Decoder 1 (z.B. für Adressierungszwecke):

- Code-Umsetzer ordnet Zeichen eines Code A den Zeichen eines Codes B zu
- 1-aus-N-Decoder hat N Ausgänge und ld N Eingänge
- Ausgang n liefert dann 1 wenn die eingegebene Dualzahl n ist

a_2	a_1	a_0	f_7	f_6	f_5	f_4	f_3	f_2	f_1	f_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0



1-aus-N-Decoder 2: Schaltnetz



Prioritäts-Encoder 1 (hier als Beispiel 1-aus-8):

- Umkehrung des 1-aus-N-Decoders
- Ausgänge liefern die Dualzahl n die der höchsten Eingangsnummer entspricht, an der 1 liegt (niedrigere gleichgültig !)

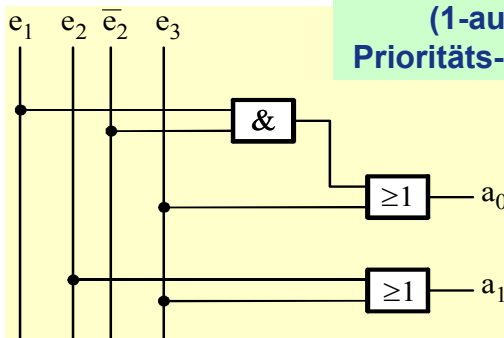
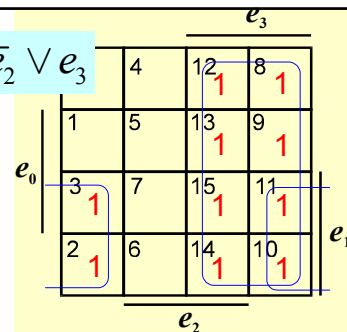
Dez	e_7	e_6	e_5	e_4	e_3	e_2	e_1	e_0	a_2	a_1	a_0
(0)–1	0	0	0	0	0	0	0	1	0	0	0
2–3	0	0	0	0	0	0	1	x	0	0	1
4–7	0	0	0	0	0	1	x	x	0	1	0
8–15	0	0	0	0	1	x	x	x	0	1	1
16–31	0	0	0	1	x	x	x	x	1	0	0
32–63	0	0	1	x	x	x	x	x	1	0	1
64–127	0	1	x	x	x	x	x	x	1	1	0
128–255	1	x	x	x	x	x	x	x	1	1	1

2.4 Ausgewählte Beispiele logischer Schaltungen

GIV II 2-59

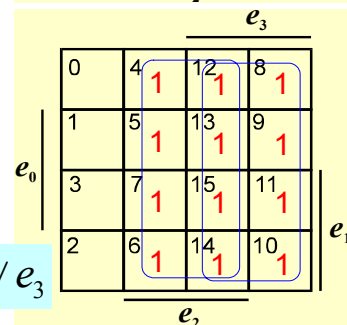
Dez	e_3	e_2	e_1	e_0	a_1	a_0
(0)–1	0	0	0	1	0	0
2–3	0	0	1	x	0	1
4–7	0	1	x	x	1	0
8–15	1	x	x	x	1	1

$$a_0 = e_1 \bar{e}_2 \vee e_3$$



(1-aus-4-) Prioritäts-Encoder 2

$$a_1 = e_2 \vee e_3$$

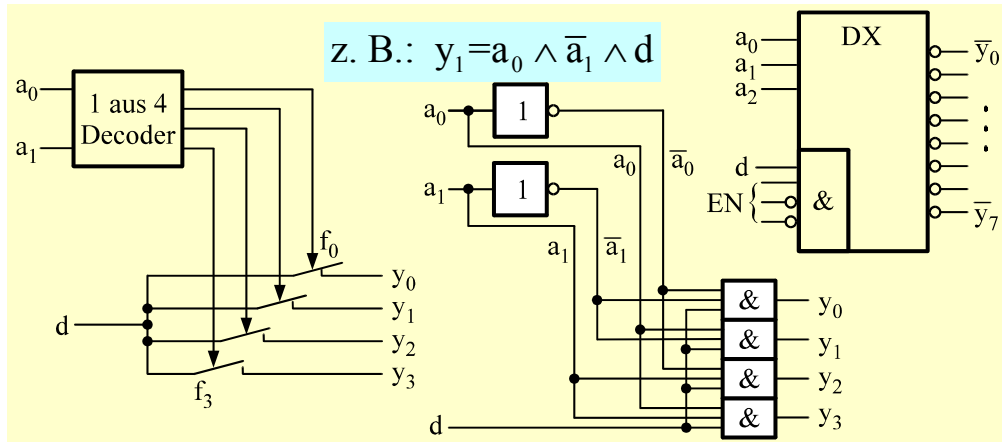


2.4 Ausgewählte Beispiele logischer Schaltungen

GIV II 2-60

Multiplexer

Eingangsdaten an wählbare Ausgänge verteilen, log. Funktion:
UND-Verknüpfung Adresse (1-aus-N-Decoder) mit Datum



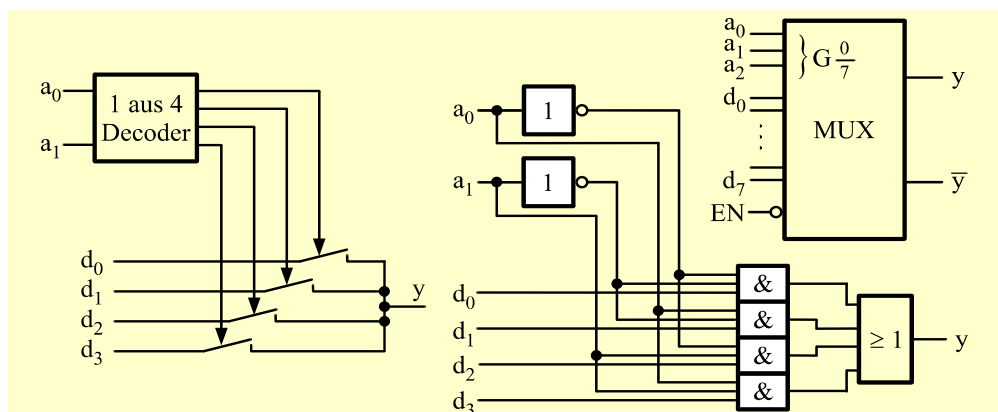
2.4 Ausgewählte Beispiele logischer Schaltungen

GIV II 2-61

Demultiplexer

Wählbare Eingangsdaten an einen Ausgang verteilen, Umkehrung
des Multiplexers, log. Funktion:

$$y = (\bar{a}_0 \wedge \bar{a}_1 \wedge d_0) \vee (a_0 \wedge \bar{a}_1 \wedge d_1) \vee (\bar{a}_0 \wedge a_1 \wedge d_2) \vee (a_0 \wedge a_1 \wedge d_3)$$



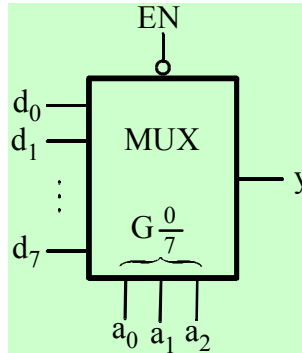
2.4 Ausgewählte Beispiele logischer Schaltungen

GIV II 2-62

Schaltnetze mit Multiplexern 1

- Technologisch bedeutsam (aufwandarme Schaltungsrealisierung)
- Multiplexereingänge erhalten Werte der logischen Funktion
- Adressierung der Eingänge liefert jeweiligen Wert am Ausgang

a_2	a_1	a_0	y
0	0	0	d_0
0	0	1	d_1
0	1	0	d_2
0	1	1	d_3
1	0	0	d_4
1	0	1	d_5
1	1	0	d_6
1	1	1	d_7



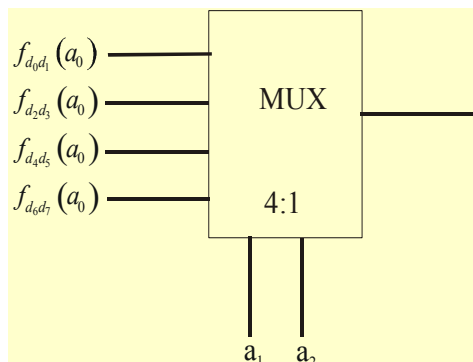
2.4 Ausgewählte Beispiele logischer Schaltungen

GIV II 2-63

Schaltnetze mit Multiplexern 2

- Vereinfachung durch Ersatz des N:1-Mux durch (N-1):1-Mux (d.h. ein Adressiereingang weniger und halb so viele Eingänge)
- an den halb so vielen Eingängen liegt dann der Funktionswert in Abhängigkeit einer Variablen
- die anderen Variablen verbleiben an den Adressiereingängen

a_2	a_1	a_0	y
0	0	0	d_0
0	0	1	d_1
0	1	0	d_2
0	1	1	d_3
1	0	0	d_4
1	0	1	d_5
1	1	0	d_6
1	1	1	d_7



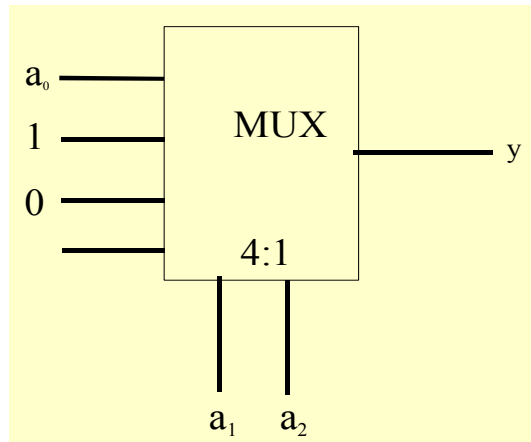
2.4 Ausgewählte Beispiele logischer Schaltungen

GIV II 2-64

Schaltnetze mit Multiplexern 3

- Beispiel zur Vereinfachung eines 8:1-Mux durch 4:1-Mux

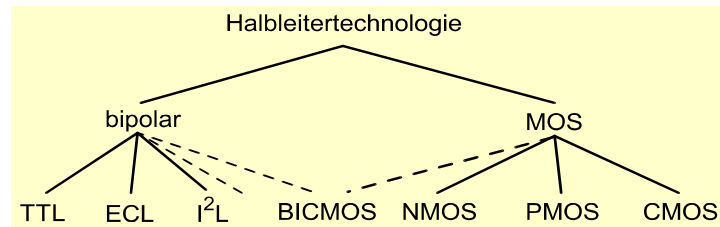
a_2	a_1	a_0	y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0



2. Schaltnetze

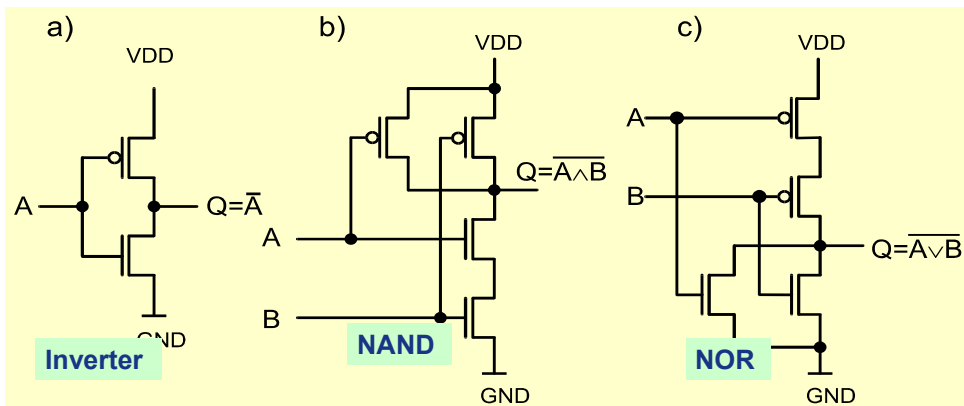
- 2.1 Boolesche Algebra
- 2.2 Logische Schaltungen (Schaltnetz, Schaltwerk)
- 2.3 Systematische Minimierungsverfahren
- 2.3 Ausgewählte Beispiele logischer Schaltungen
- 2.4 Digitale CMOS-Grundsaltungen

Überblick über Halbleitertechnologien

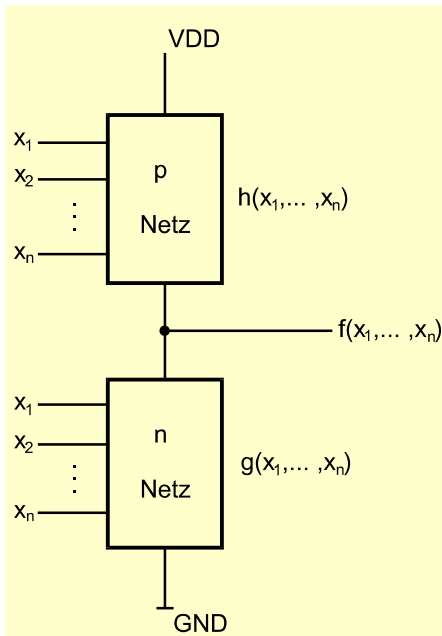


bipolar: hohe Verlustleistung / höchste Schaltgeschwindigkeit
CMOS: geringe Verlustleistung / höchster Integrationsgrad / hohe Schaltgeschwindigkeit

Inverter, NAND und NOR in CMOS-Technologie



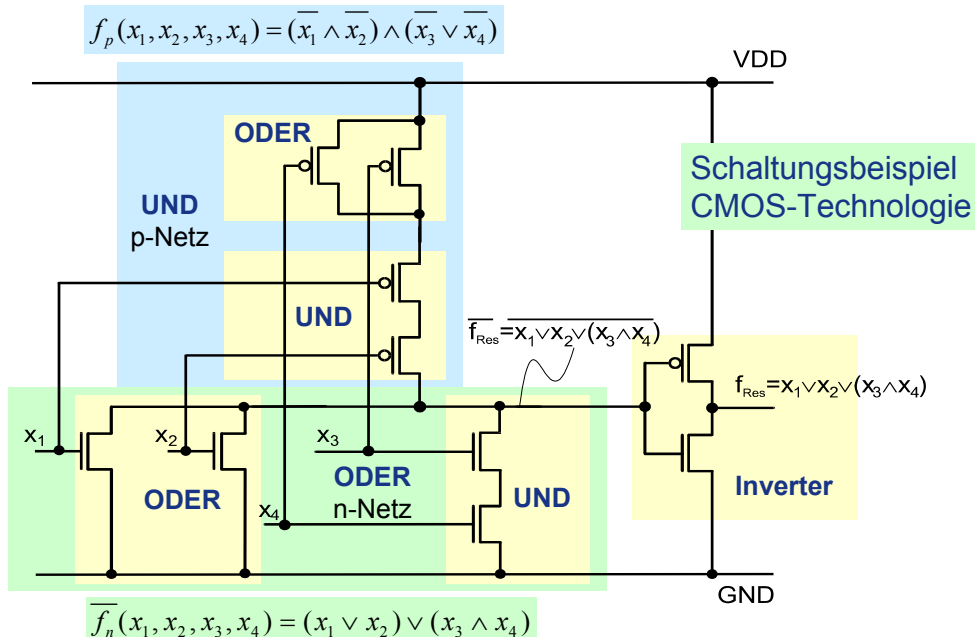
Stets 2 komplementäre Transistoren (n-, p- Kanal) in Serie simultan angesteuert: einer der beiden sperrt immer (statisch jedenfalls) und ein (minimaler Sperr-) Stromfluss resultiert (Verlustleistung!) „intelligenter Lastwiderstand“



Prinzipbild der Gatterlogik

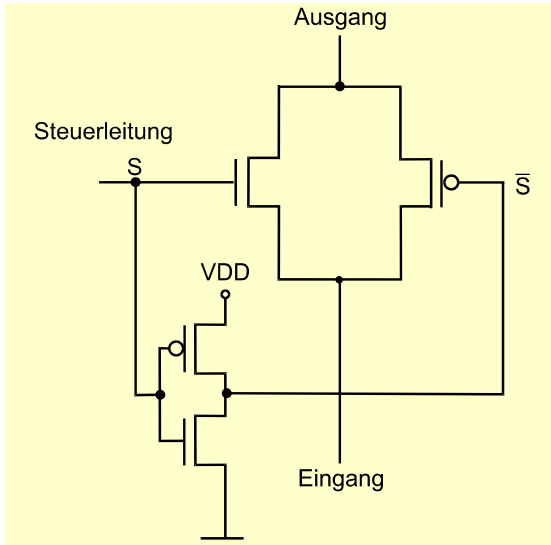
p-Netz und n-Netz: komplementäre Schaltnetze

es resultiert ein logischer Umschalter, der den Ausgang auf Versorgungs- oder Massepotential legt



Schaltungsbeispiel CMOS-Technologie

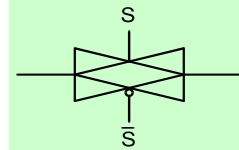
Aufbau und Logiksymbol eines Transmission-Gates



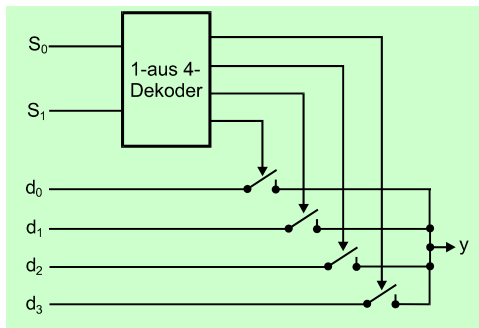
Transmission-Gate

- Parallelschaltung aus- und n-Kanal-Transistor
- gleichberechtigte Anschlüsse
- keine Pegelregenerierung, Störabstand sinkt bei Kette vom Transmission-Gates

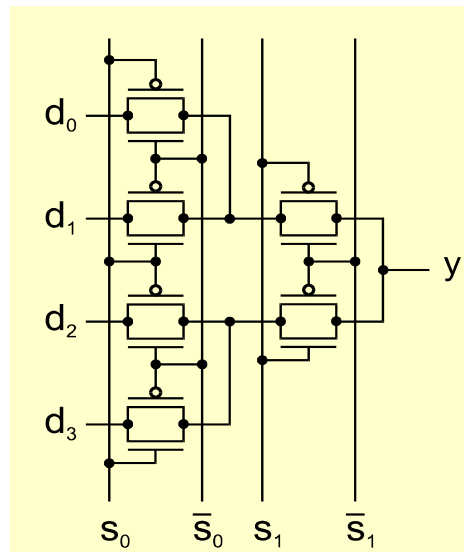
Symbol



4:1 Multiplexer aus Transmission-G.



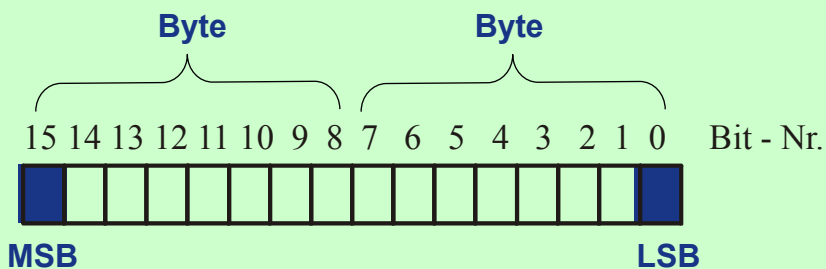
$$y = \overline{s_0} \overline{s_1} \cdot d_0 \vee s_0 \overline{s_1} \cdot d_1 \vee \overline{s_0} s_1 \cdot d_2 \vee s_0 s_1 \cdot d_3$$



3. Arithmetisch-Logische Einheiten

- 3.1 Zahlendarstellungen
- 3.2 Festkomma-Zahlenformate
- 3.3 Addierer-Schaltnetze
- 3.4 Schaltwerke zur binären Multiplikation

16 Bit Ganzzahlenformat:



$$-2^{15} \leq N \leq +2^{15} - 1$$

$$-32768 \leq N \leq 32767$$

einfache Genauigkeit

$$-2^{31} \leq N \leq +2^{31} - 1$$

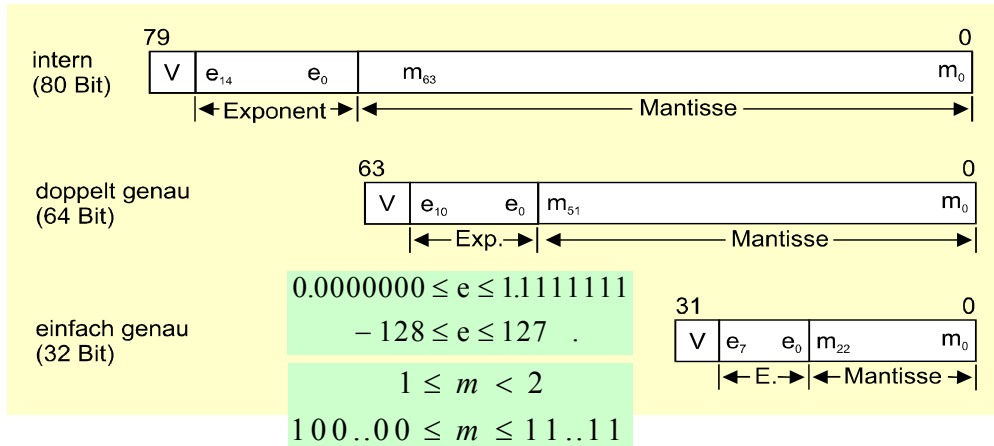
$$-2.147 \cdot 10^9 \leq N \leq +2.147 \cdot 10^9$$

Doppelwort-Genauigkeit

IEEE-Floating-Point-Standard:

$$Z_{GL} = (-1)^v \cdot m \cdot 2^e$$

- Exponent im Binary-Offset-Code
- Mantisse in Betrag-Vorzeichen-Darstellung, Mantisse auf 1 normiert (weglassen)



3.1 Zahlendarstellungen

GIV II 3-3

Gesichtspunkte der Auswahl von Zahlendarstellungen:

- Geschwindigkeit
- Genauigkeit
- Aufwand, Verfügbarkeit

Polyadische Zahlensysteme:

- ganzzahlige Basis, Position bestimmt Stellenbewertung
- von rechts nach links ansteigend
- z.B. dezimal-polyadisch oder dual-polyadisch

$$Z_n = \sum_{i=0}^{N-1} b_i \cdot 2^i = b_{N-1} \cdot 2^{N-1} + \dots + b_1 \cdot 2^1 + b_0 \cdot 2^0$$

Dualzahlendarstellungen:

- Betrag-Vorzeichen-Darstellung Z_{BV}
- Offset-Dual (Binary-Offset) -Darstellung Z_{BO}
- Zweierkomplement-Darstellung Z_{2K}
- Einer-Komplement-Darstellung Z_{1K}

3.2 Festkomma-Zahlenformate

GIV II 3-4

Dezimal- zahlwert	Betrag- Vorzeichen Z_{BV}	Offset- Binary Z_{OB}	Einer- Komplement Z_{IK}	Zweier- Komplement Z_{2K}
+7	0 1 1 1	1 1 1 1	0 1 1 1	0 1 1 1
+6	0 1 1 0	1 1 1 0	0 1 1 0	0 1 1 0
+5	0 1 0 1	1 1 0 1	0 1 0 1	0 1 0 1
+4	0 1 0 0	1 1 0 0	0 1 0 0	0 1 0 0
+3	0 0 1 1	1 0 1 1	0 0 1 1	0 0 1 1
+2	0 0 1 0	1 0 1 0	0 0 1 0	0 0 1 0
+1	0 0 0 1	1 0 0 1	0 0 0 1	0 0 0 1
+0	0 0 0 0	1 0 0 0	0 0 0 0	0 0 0 0
-0	1 0 0 0	-	1 1 1 1	-
-1	1 0 0 1	0 1 1 1	1 1 1 0	1 1 1 1
-2	1 0 1 0	0 1 1 0	1 1 0 1	1 1 1 0
-3	1 0 1 1	0 1 0 1	1 1 0 0	1 1 0 1
-4	1 1 0 0	0 1 0 0	1 0 1 1	1 1 0 0
-5	1 1 0 1	0 0 1 1	1 0 1 0	1 0 1 1
-6	1 1 1 0	0 0 1 0	1 0 0 1	1 0 1 0
-7	1 1 1 1	0 0 0 1	1 0 0 0	1 0 0 1
(-8)		(0 0 0 0)		(1 0 0 0)

3.2 Festkomma-Zahlenformate

GIV II 3-5

Vorzeichen-Betrag-Darstellung

$$b_N = 0 \quad \text{für} \quad Z_{Dez} \geq 0$$

$$b_N = 1 \quad \text{für} \quad Z_{Dez} < 0$$

$$Z_{BV} = (-1)^{b_N} \cdot \sum_{i=0}^{N-1} b_i \cdot 2^i$$

Beispiel:

Günstig: Multiplikation, Division

Ungünstig: Subtraktion (Steuerlogik für Vorzeichen, pos. neg. Null)

3.2 Festkomma-Zahlenformate

GIV II 3-6

Zweier-Komplement-Darstellung

$$b_N = 0 \quad \text{für} \quad Z_{\text{Dez}} \geq 0$$

$$b_N = 1 \quad \text{für} \quad Z_{\text{Dez}} < 0$$

$$Z_{2K} = -b_N \cdot 2^N + \sum_{i=0}^{N-1} b_i \cdot 2^i$$

Positive Zahl $Z_{N1} = +B_{N1}$

	2^N				2^0
	2^4	2^3	2^2	2^1	2^0
$Z_{N1}=+5$	0.	0	1	0	1
$B_{N1}=5$		0	1	0	1

Zweier-Komplement-Darstellung

Negative Zahl $Z_{N2} = -B_{N1}$: $B_{N2} = 2^N - B_{N1}$

2^N	1	0	0	0	0
$-B_{N1}=5$		0	1	0	1
B_{N2}		1	0	1	1
Z_{N2}	1.	1	0	1	1

Addition von B_{N1} und B_{N2} führt auf 2^N
Einbeziehung des Vorzeichenbits möglich

Zweier-Komplement-Darstellung

Negative Zahl $Z_{N2} = -B_{N1}$: $Z_{N2} = 2^{N+1} - Z_{N1}$

Überträge
weglassen!

	2^{N+1}	2^N		2^2	2^1	2^0
B_{N1}			0	1	0	1
$+B_{N2}$			1	0	1	1
2^N		1.	0	0	0	0
Z_{N1}		0	0	1	0	1
$+Z_{N2}$		1	1	0	1	1
2^N	1	0	0	0	0	0

3.2 Festkomma-Zahlenformate

GIV II 3-9

Zweier-Komplement-Darstellung

$b_N = 0$ für $Z_{Dez} \geq 0$
 $b_N = 1$ für $Z_{Dez} < 0$

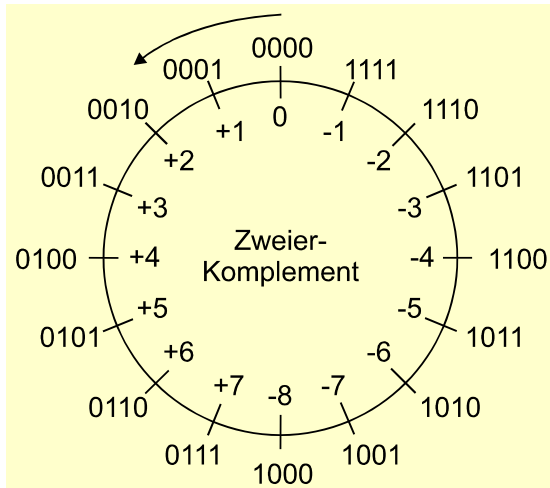
$$Z_{2K} = -b_N \cdot 2^N + \sum_{i=0}^{N-1} b_i \cdot 2^i$$

Zahlenbeispiel: $Z_{2K} = +101 = 64 + 32 + 4 + 1$, $Z_{2K} = -101 = 128 - 27$

	2^N	...	...			...	2^1	2^0
	b_N	...	...			...	b_1	b_0
	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
+101	0.	1	1	0	0	1	0	1
-101	1.	0	0	1	1	0	1	1

3.2 Festkomma-Zahlenformate

GIV II 3-10



Definitionsbereich:

$$-2^N \leq Z_{2K} \leq \sum_{i=0}^{N-1} 2^i$$

$$-2^N \leq Z_{2K} \leq 2^N - 1$$

Zweier-Komplement:

- zyklisches Zahlenformat
- bei Wortlänge N+1 zyklisch mit mod 2^{N+1}
- Zahlenring !

Komplementierung (Vorzeichenwechsel)

$$B_{N2} = 2^N - B_{N1} \quad \rightarrow \quad \sum_{i=0}^{N-1} \underbrace{b_{i2}}_{\text{gesuchte Bit-kombination}} \cdot 2^i = 2^N - \sum_{i=0}^{N-1} \underbrace{b_{i1}}_{\text{gegebene Bit-kombination}} \cdot 2^i$$

$$\text{wegen: } \sum_{i=0}^{N-1} b_{i1} \cdot 2^i + \sum_{i=0}^{N-1} \bar{b}_{i1} \cdot 2^i = \sum_{i=0}^{N-1} 2^i (\hat{=} 11111 \dots) = 2^N - 1$$

$$\sum_{i=0}^{N-1} b_{i1} \cdot 2^i = 2^N - 1 - \sum_{i=0}^{N-1} \bar{b}_{i1} \cdot 2^i$$

$$B_{N2} = \sum_{i=0}^{N-1} b_{i2} \cdot 2^i = 2^N - \sum_{i=0}^{N-1} b_{i1} \cdot 2^i = 2^N - 2^N + 1 + \sum_{i=0}^{N-1} \bar{b}_{i1} \cdot 2^i = \sum_{i=0}^{N-1} \bar{b}_{i1} \cdot 2^i + 1$$

Regel: Alle Bits negieren und eine $1=2^0$ addieren

Komplementierung (Vorzeichenwechsel)

B_{N1}		0	1	0	1
$\bar{B}_{N1}$		1	0	1	0
+1					1
B_{N2}		1	0	1	1
Z_{N2}	1.	1	0	1	1

Z_{N1}	0.	0	1	0	1
$\bar{Z}_{N1}$	1.	1	0	1	0
+1					1
Z_{N2}	1.	1	0	1	1

geg. *Betrag*

Negation

1 addieren

Ergebnis

Vorzeichen ergänzen

einfacher:

geg. *Zahl*

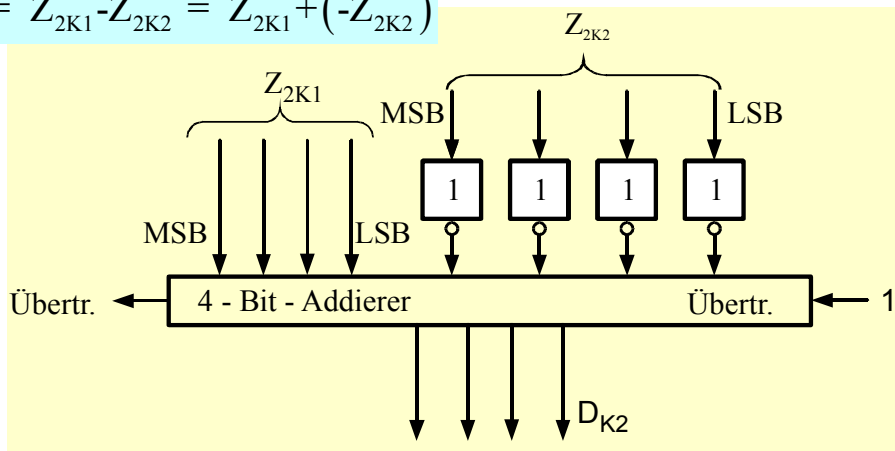
Negation

1 addieren

Ergebnis

Subtraktion: Addition komplementierter Zahlen (d.h. Vorzeichen wechseln und addieren)

$$D_{K2} = Z_{2K1} - Z_{2K2} = Z_{2K1} + (-Z_{2K2})$$



3. Arithmetisch-Logische Einheiten

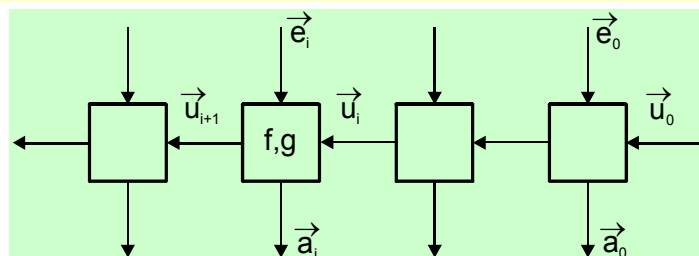
- 3.1 Zahlendarstellungen
- 3.2 Festkomma-Zahlenformate
- 3.3 **Addierer-Schaltnetze**
- 3.4 Schaltwerke zur binären Multiplikation

3.3 Addierer-Schaltnetze

GIV II 3-15

Schaltkette

- Kaskade gleichartiger Schaltnetze
- Basis für wortbezogene Operationen, die gleichartig für alle Bits angewendet werden sollen
- Einbeziehung von Übertragsverrechnungen
- Grundlage der logischen Realisierung arithmetischer Operationen
- Realisierung eines Systems von Booleschen Funktionsbündeln



$$\vec{u}_{i+1} = f(\vec{u}_i, \vec{e}_i) \quad \vec{a}_i = g(\vec{u}_i, \vec{e}_i)$$

3.3 Addierer-Schaltnetze

GIV II 3-16

A:	a_N	...	...	a_i	...	a_1	a_0	Dualzahl A
B:	b_N	...	...	b_i	...	b_1	b_0	Dualzahl B
C:	c_N	...	c_{i+1}	c_i	...	c_1	c_0	Überträge
D:	s_N	...	...	s_i	...	s_1	s_0	Summe

Aufgabe: Abbildung der angestrebten arithmetischen Operation auf ihr logisches Äquivalent

Sequentielle oder parallele Realisierungen der Teiloperationen in der Schaltkette

3.3 Addierer-Schaltnetze

GIV II 3-17

Eingangsbits			interne Bits			Ausgangsbits	
c_i	b_i	a_i	p_i	g_i	r_i	s_i	c_{i+1}
0	0	0	0	0	0	0	0
0	0	1	1	0	0	1	0
0	1	0	1	0	0	1	0
0	1	1	0	1	0	0	1
1	0	0	0	0	0	1	0
1	0	1	1	0	1	0	1
1	1	0	1	0	1	0	1
1	1	1	0	1	0	1	1

s_i : Summenbit
 c_i : Übertragsbit
 p_i : Propagate-Bit
 g_i : Generate-Bit

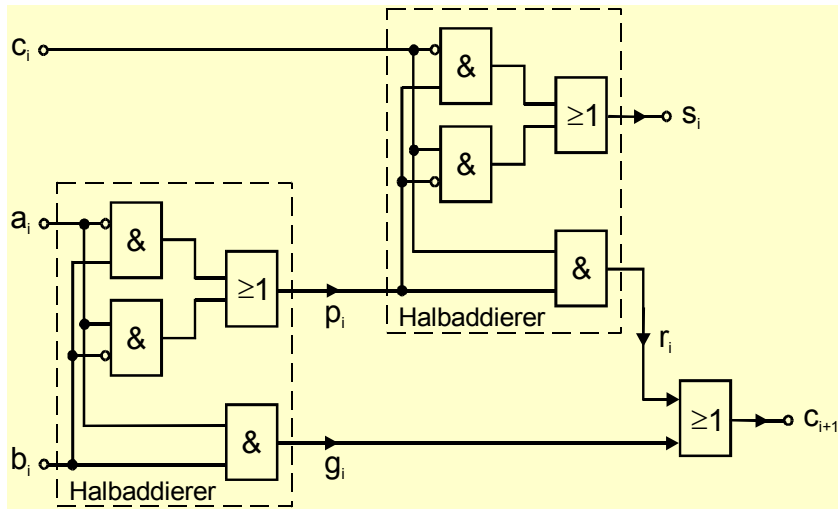
$$\begin{aligned}
 s_i &= LSB[a_i + b_i + c_i] = LSB[p_i + c_i] \\
 &= a_i \bar{b}_i \bar{c}_i \vee \bar{a}_i b_i \bar{c}_i \vee \bar{a}_i \bar{b}_i c_i \vee a_i b_i c_i \\
 &= \bar{p}_i c_i \vee p_i \bar{c}_i = c_i \oplus p_i
 \end{aligned}$$

$$\begin{aligned}
 p_i &= LSB[a_i + b_i] \\
 &= \bar{a}_i b_i \vee a_i \bar{b}_i = a_i \oplus b_i
 \end{aligned}$$

$$\begin{aligned}
 c_{i+1} &= a_i b_i \vee c_i p_i \\
 &= g_i \vee c_i p_i = a_i b_i \vee a_i c_i \vee b_i c_i
 \end{aligned}$$

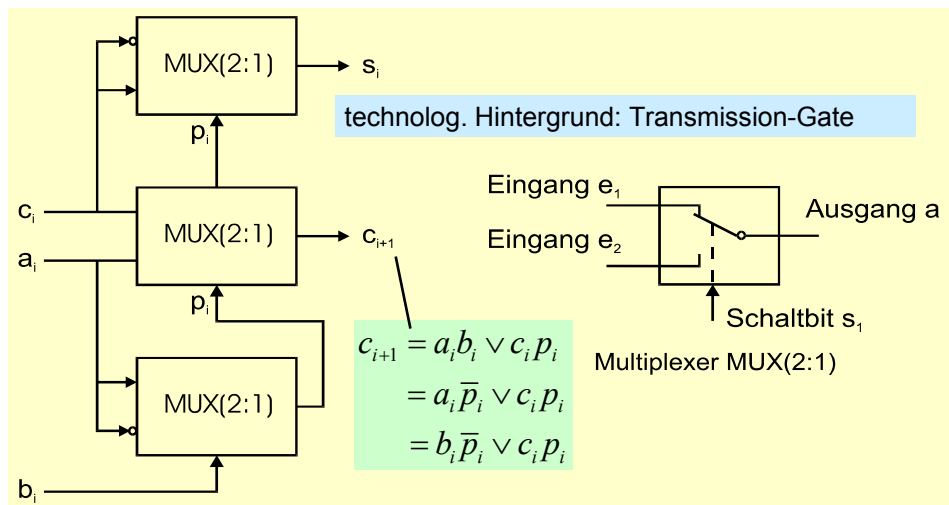
3.3 Addierer-Schaltnetze

GIV II 3-18



3.3 Addierer-Schaltnetze

GIV II 3-19

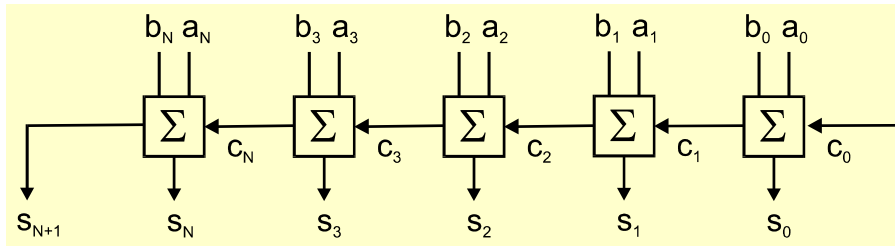


Begründung: lokales Übertragsbit durch
eins der p-Bits ausdrücken:

$$a_i b_i = a_i \bar{p}_i = b_i \bar{p}_i$$

3.3 Addierer-Schaltnetze

GIV II 3-20



Ripple-Carry-Addierer:

Nachteil Zeitbedarf ($N \times t_{\text{Add}}$)

3.3 Addierer-Schaltnetze

GIV II 3-21

$$c_{i+1} = g_i \vee (p_i c_i) = a_i b_i \vee c_i (\bar{a}_i b_i \vee a_i \bar{b}_i)$$

$$c_1 = g_0 \vee (p_0 c_0)$$

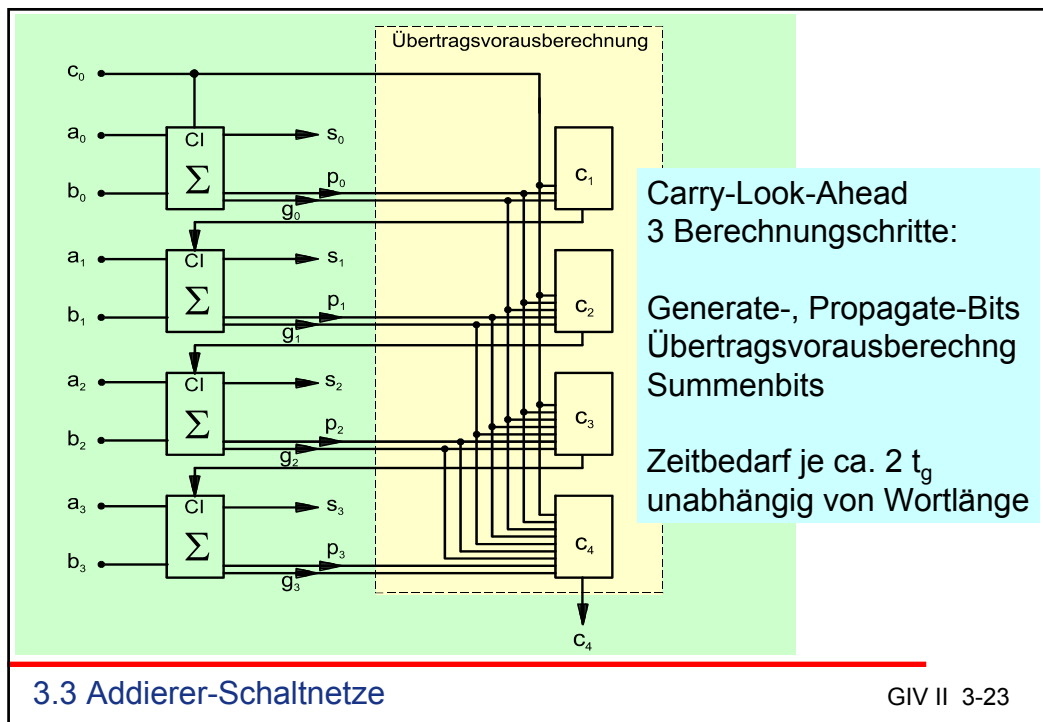
$$c_2 = g_1 \vee (p_1 c_1) = g_1 \vee (p_1 g_0) \vee (p_1 p_0 c_0)$$

$$\begin{aligned} c_3 &= g_2 \vee (p_2 c_2) = g_2 \vee (p_2 g_1) \vee (p_2 p_1 c_1) \\ &= g_2 \vee (p_2 g_1) \vee (p_2 p_1 g_0) \vee (p_2 p_1 p_0 c_0) \\ &\vdots \end{aligned}$$

Übertragsvorausberechnung aus Generate- und Propagate-Bits sowie äußerem c_0 durch sukzessives Einsetzen

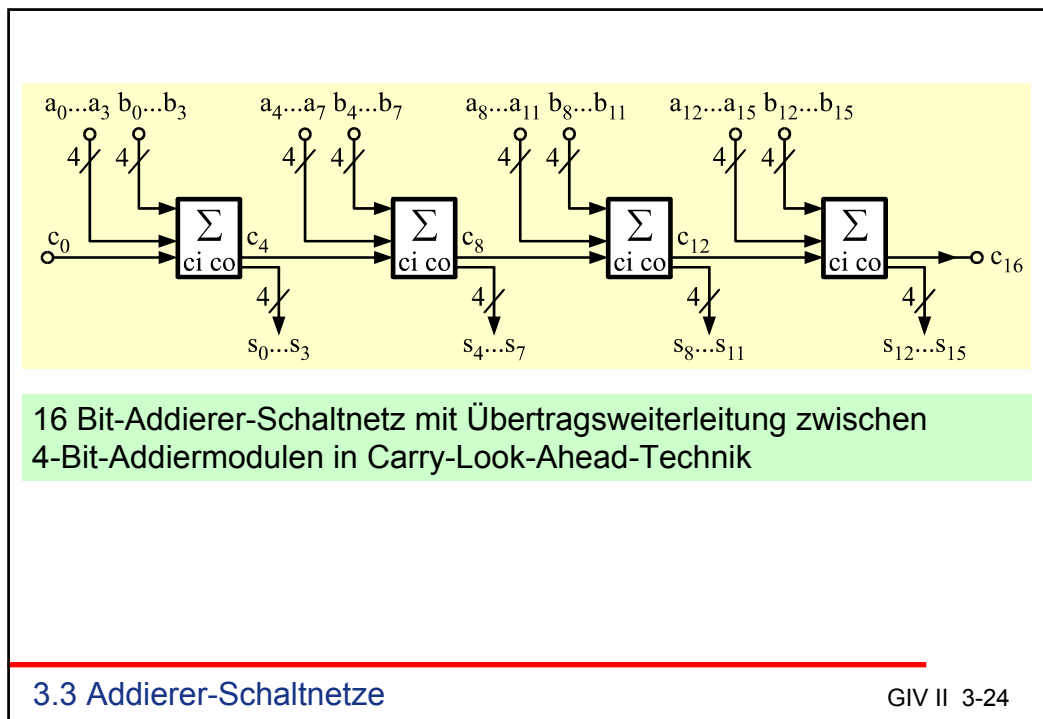
3.3 Addierer-Schaltnetze

GIV II 3-22



3.3 Addierer-Schaltnetze

GIV II 3-23



3.3 Addierer-Schaltnetze

GIV II 3-24

Hierarchische Carry-Look-Ahead-Addition

z.B. bei 16-Bit-Addierer: Übertragsvorausberechnung in zwei hierarchisch getrennten Ebenen

Gleichung für Übertrag c_{i+1} liefert für c_4 :

$$c_4 = \underbrace{g_3 \vee (p_3 \wedge g_2) \vee (p_3 \wedge p_2 \wedge g_1) \vee (p_3 \wedge p_2 \wedge p_1 \wedge g_0)}_{g_3^1} \vee \left(\underbrace{p_3 \wedge p_2 \wedge p_1 \wedge p_0}_{p_3^1} \wedge c_0 \right)$$

entspricht Berechnung von c_1 in unterer Hierarchiestufe

$$\text{Ebenso } c_8 = g_7^1 \vee p_7^1 c_4 = g_7^1 \vee p_7^1 (g_3^1 \vee p_3^1 c_0) = g_7^1 \vee p_7^1 g_3^1 \vee p_7^1 p_3^1 c_0$$

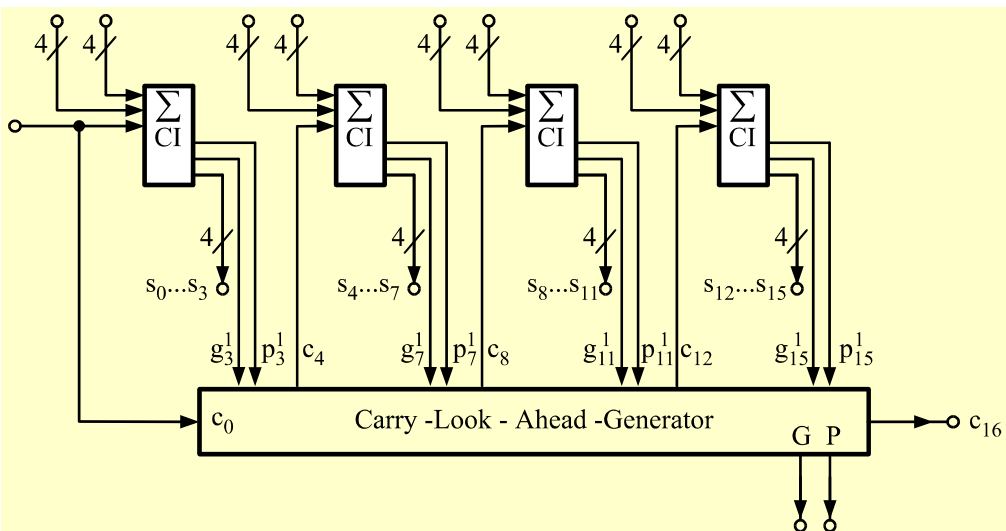
im Vergleich zu c_2

Ersichtlich: gleiche Operationen in unterschiedlichen Hierarchiestufen

3.3 Addierer-Schaltnetze

GIV II 3-25

Hierarchische Carry-Look-Ahead-Addition



3.3 Addierer-Schaltnetze

GIV II 3-26

Anwendungen der Akkumulation in der Signalverarbeitung

so genannte diskrete Faltung:
$$\sum_{m=1}^N s(m) \cdot h(n-m)$$

Korrelation:
$$\sum_{m=1}^N s_1(m) \cdot s_2(n-m)$$

digitale Filterung:
$$\sum_{i=1}^I a_i \cdot x_i$$

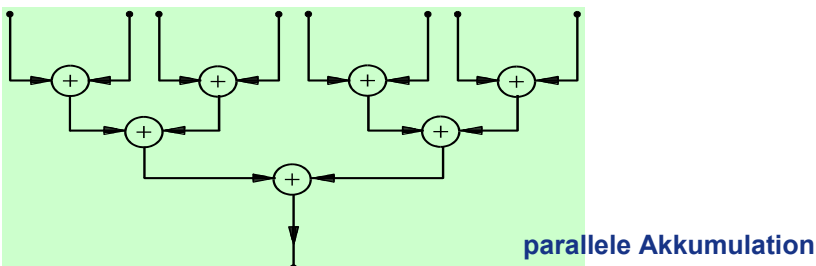
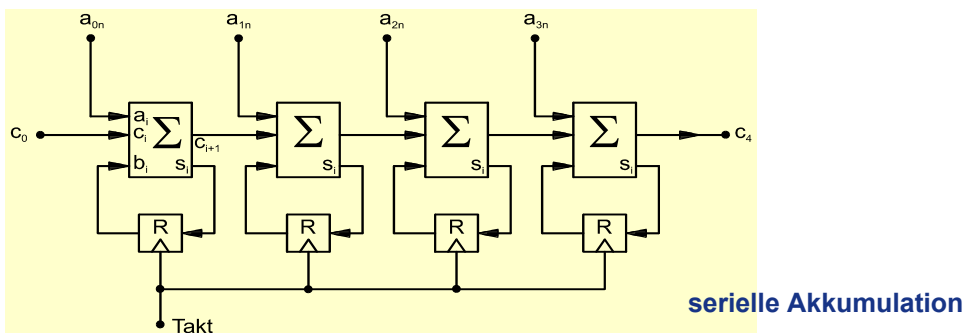
digitale Transformationen
(Fourier, Cosinus, Wavelet, ...):
$$\sum_{n=1}^N s(n) \cdot f_{Basis}(kn)$$

binäre Multiplikation:
$$\sum_{i=1}^I a_i \cdot 2^i \cdot \sum_{i=1}^I b_i \cdot 2^i$$

grundsätzlich: Matrix-Vektor-Operationen aller Art

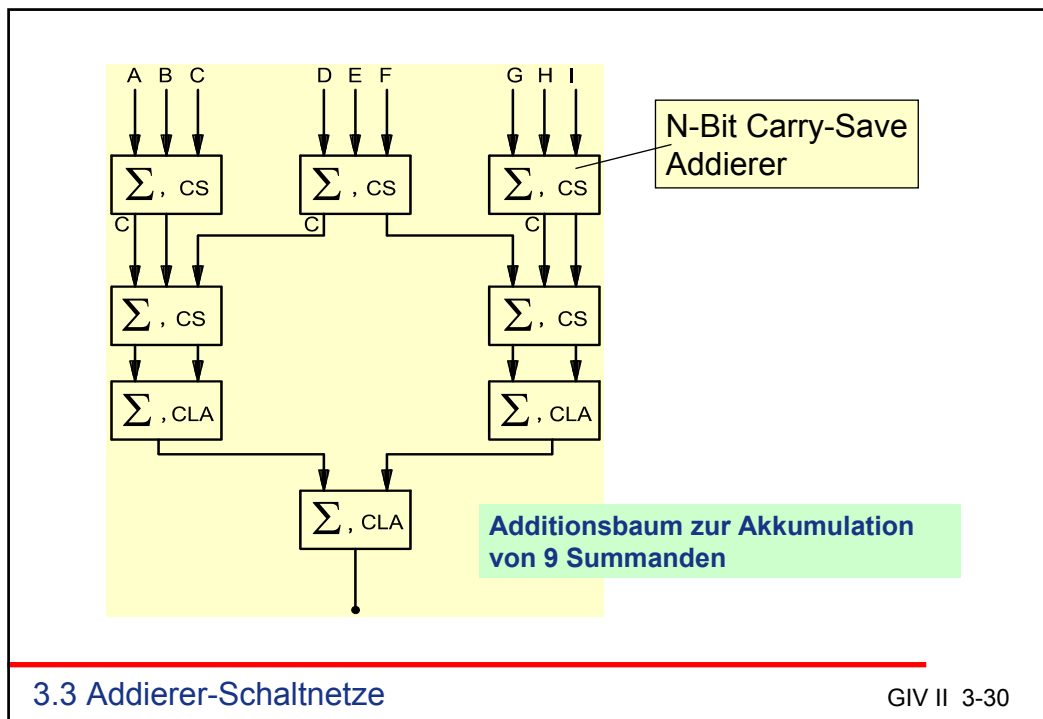
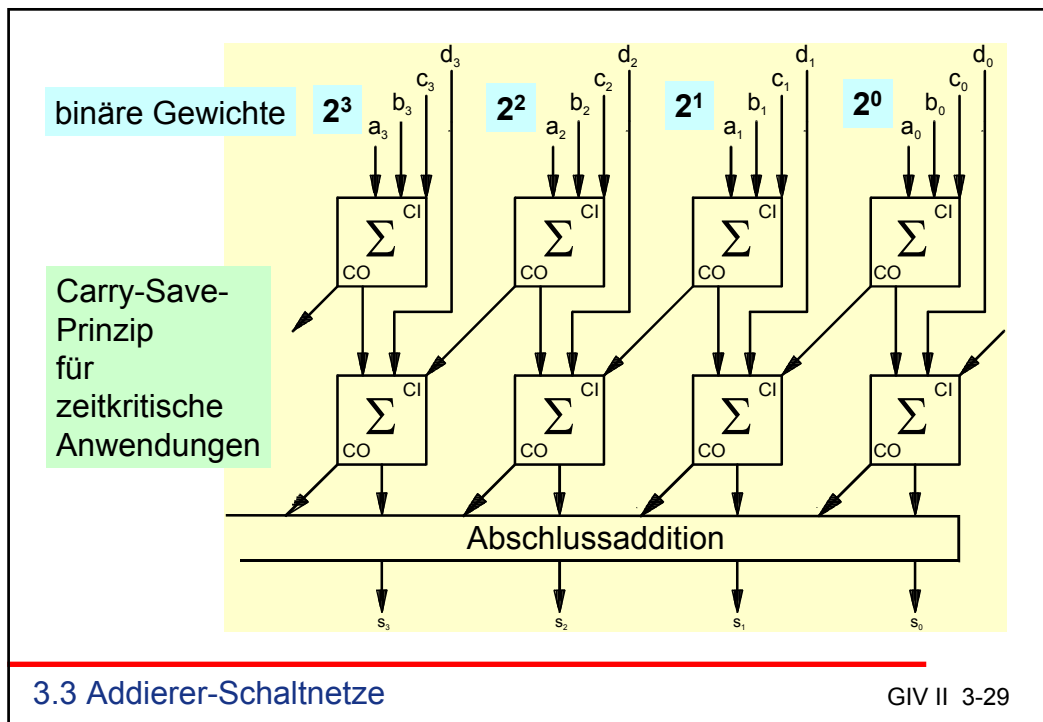
3.3 Addierer-Schaltnetze

GIV II 3-27



3.3 Addierer-Schaltnetze

GIV II 3-28



Rechenwerk zur Addition und Subtraktion von 2er-Komplementzahlen

Zweierkomplementzahlen A bzw. B.:
$$A = -a_N \cdot 2^N + \sum_{i=0}^{N-1} a_i \cdot 2^i$$

Addition A + B:
$$S = A + B = (-a_N - b_N) \cdot 2^N + \sum_{i=0}^{N-1} (a_i + b_i) \cdot 2^i$$

Ergebnis:
$$S = (-a_N - b_N) \cdot 2^N + c_N \cdot 2^N + \sum_{i=0}^{N-1} s_i \cdot 2^i$$

$$= (-a_N - b_N + c_N) \cdot 2^N + \sum_{i=0}^{N-1} s_i \cdot 2^i = -s_N \cdot 2^N + \sum_{i=0}^{N-1} s_i \cdot 2^i$$

Übertrag in 2^N ?

3.3 Addierer-Schaltnetze

GIV II 3-31

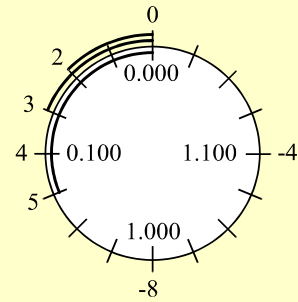
Rechenwerk zur Addition/Subtraktion von 2er-Komplementzahlen, Wahrheitstabelle und Überlaufverhalten

			Ausgang		Überlaufanzeige	Fall
a_N	b_N	c_N	S_N	c_{N+}	ü	Nr.
				1		
0	0	0	0	0	0	I
0	0	1	1	0	1 (Überlauf)	II
0	1	0	1	0	0	III
1	0	0	1	0	0	IV
0	1	1	0	1	0	V
1	0	1	0	1	0	VI
1	1	0	0	1	1 (Überlauf)	VII
1	1	1	1	1	0	VIII

3.3 Addierer-Schaltnetze

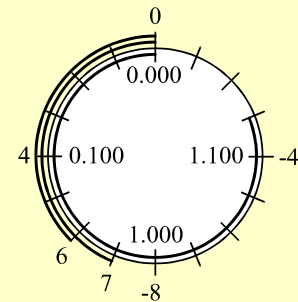
GIV II 3-32

$$\begin{array}{r}
 2 \\
 + 3 \\
 \hline
 5
 \end{array}
 \quad \text{I} \quad
 \begin{array}{r}
 (N+1) \quad N \quad 2 \quad 1 \quad 0 \quad i \\
 0. \quad 0 \quad 1 \quad 0 \quad a_i \\
 0. \quad 0 \quad 1 \quad 1 \quad b_i \\
 \quad \quad 1 \quad c_i \\
 \hline
 0. \quad 1 \quad 0 \quad 1 \quad s_i
 \end{array}$$



Addition von 2er-Komplementzahlen Fallbeispiele I, II

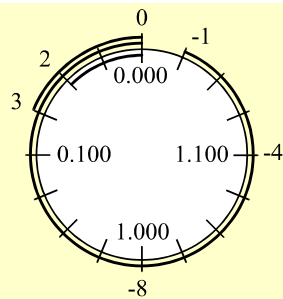
$$\begin{array}{r}
 7 \\
 + 6 \\
 \hline
 13
 \end{array}
 \quad \text{II} \quad
 \begin{array}{r}
 0. \quad 1 \quad 1 \quad 1 \quad a_i \\
 0. \quad 1 \quad 1 \quad 0 \quad b_i \\
 1 \quad 1 \quad c_i \\
 \hline
 1. \quad 1 \quad 0 \quad 1 \quad s_i
 \end{array}$$



3.3 Addierer-Schaltnetze

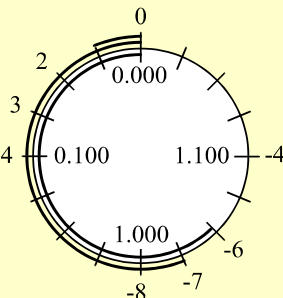
GIV II 3-33

$$\begin{array}{r}
 3 \\
 + (-1) \\
 \hline
 2
 \end{array}
 \quad \text{V,VI} \quad
 \begin{array}{r}
 0. \quad 0 \quad 1 \quad 1 \quad a_i \\
 1. \quad 1 \quad 1 \quad 1 \quad b_i \\
 1 \quad 1 \quad 1 \quad 1 \quad c_i \\
 \hline
 1 \quad 0. \quad 0 \quad 1 \quad 0 \quad s_i
 \end{array}$$



Addition von 2er-Komplementzahlen Fallbeispiele

$$\begin{array}{r}
 1 \\
 + (-7) \\
 \hline
 -6
 \end{array}
 \quad \text{III,IV} \quad
 \begin{array}{r}
 0. \quad 0 \quad 0 \quad 1 \quad a_i \\
 1. \quad 0 \quad 0 \quad 1 \quad b_i \\
 \quad \quad 1 \quad c_i \\
 \hline
 1. \quad 0 \quad 1 \quad 0 \quad s_i
 \end{array}$$



3.3 Addierer-Schaltnetze

GIV II 3-34

$$\begin{array}{r} -1 \\ + (-3) \\ \hline -4 \end{array}$$

VIII

$$\begin{array}{rcccc} 1 & 1 & 1 & 1 & a_i \\ 1 & 1 & 0 & 1 & b_i \\ 1 & 1 & 1 & 1 & c_i \\ \hline 1 & 1 & 1 & 0 & 0 & s_i \end{array}$$

$$\begin{array}{r} -6 \\ + (-7) \\ \hline -13 \end{array}$$

VII

~~$$\begin{array}{rcccc} 1 & 0 & 1 & 0 & a_i \\ 1 & 0 & 0 & 1 & b_i \\ 1 & & & & c_i \\ \hline 1 & 0 & 0 & 1 & 1 & s_i \end{array}$$~~

3.3 Addierer-Schaltnetze

GIV II 3-35

Rechenwerk zur Addition/Subtraktion von 2er-Komplementzahlen, Überlaufanzeige aus Wahrheitstabelle

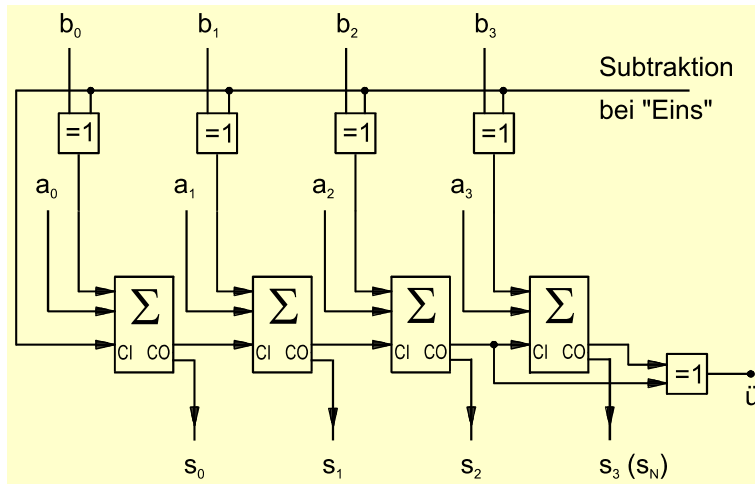
			Ausgang	Überlaufanzeige	Fall
a_N	b_N	c_N	$S_N \quad c_{N+}$	$\ddot{u}$	Nr.
			1		
0	0	0	0 0	0	I
0	0	1	1 0	1 (Überlauf)	II
0	1	0	1 0	0	III
1	0	0	1 0	0	IV
0	1	1	0 1	0	V
1	0	1	0 1	0	VI
1	1	0	0 1	1 (Überlauf)	VII
1	1	1	1 1	0	VIII

$$\ddot{u} = (a_N \equiv b_N) \wedge (c_{N+1} \neq s_N)$$

alternativ: $\ddot{u} = c_{N+1} \neq c_N$

3.3 Addierer-Schaltnetze

GIV II 3-36



$$\ddot{u} = (a_N \otimes b_N) \wedge (c_{N+1} \oplus s_N)$$

$$\ddot{u} = c_{N+1} \oplus c_N$$

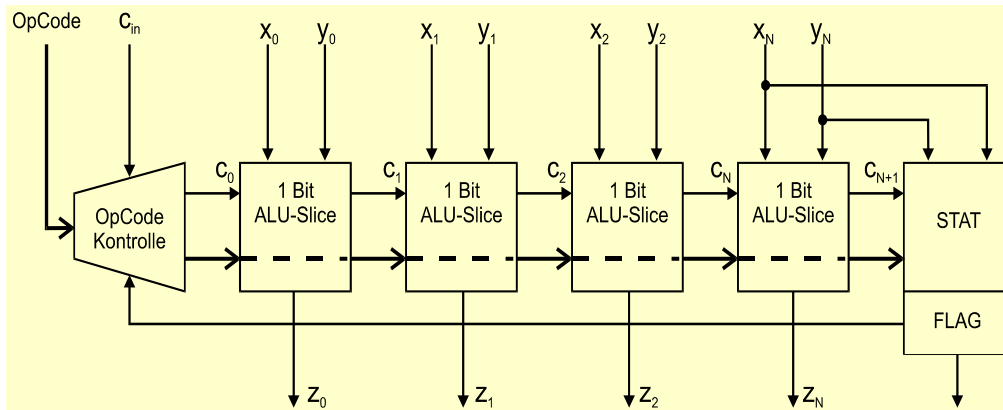
3.3 Addierer-Schaltnetze

GIV II 3-37

	Steuergrößen				logische Operation $s_4 = 1$	arithmetische Operation $s_4 = 0$	
	s_0	s_1	s_2	s_3		Übertrag $u_0 = 0$	Übertrag $u_0 = 1$
	s_0	s_1	s_2	s_3			
	0	0	0	0	$z = 0$	$z = -1$	$z = 0$
	0	0	0	1	$z = x \vee y$	$z = x \vee y$	$z = (x \vee y) + 1$
	0	0	1	0	$z = \bar{x} \wedge y$	$z = x \vee \bar{y}$	$z = (x \vee \bar{y}) + 1$
	0	0	1	1	$z = \bar{x}$	$z = x$	$z = x + 1$
	0	1	0	0	$z = x \wedge y$	$z = x \wedge \bar{y} - 1$	$z = x \wedge \bar{y}$
	0	1	0	1	$z = \bar{y}$	$z = (x \vee y) + (x \wedge \bar{y})$	$z = (x \vee y) + (x \wedge \bar{y}) + 1$
	0	1	1	0	$z = x \neq y$	$z = x - y - 1$	$z = x - y$
	0	1	1	1	$z = x \wedge y$	$z = (x + \bar{y}) + x$	$z = (x \wedge \bar{y}) + x + 1$
	1	0	0	0	$z = x \wedge y$	$z = (x \wedge y) - 1$	$z = x \wedge y$
	1	0	0	1	$z = x \equiv y$	$z = x + y$	$z = x + y + 1$
	1	0	1	0	$z = y$	$z = (x \vee \bar{y}) + (x \wedge y)$	$z = (x \vee \bar{y}) + (x \wedge y) + 1$
	1	0	1	1	$z = \bar{x} \vee y$	$z = (x \wedge y) + x$	$z = (x \wedge y) + x + 1$
	1	1	0	0	$z = x$	$z = x - 1$	$z = x$
	1	1	0	1	$z = x \vee \bar{y}$	$z = (x \vee y) + x$	$z = (x \vee y) + x + 1$
	1	1	1	0	$z = x \vee y$	$z = (x \vee \bar{y}) + x$	$z = (x \vee \bar{y}) + x + 1$
	1	1	1	1	$z = 1$	$z = x \cdot 2$	$z = x \cdot 2 + 1$

3.3 Addierer-Schaltnetze

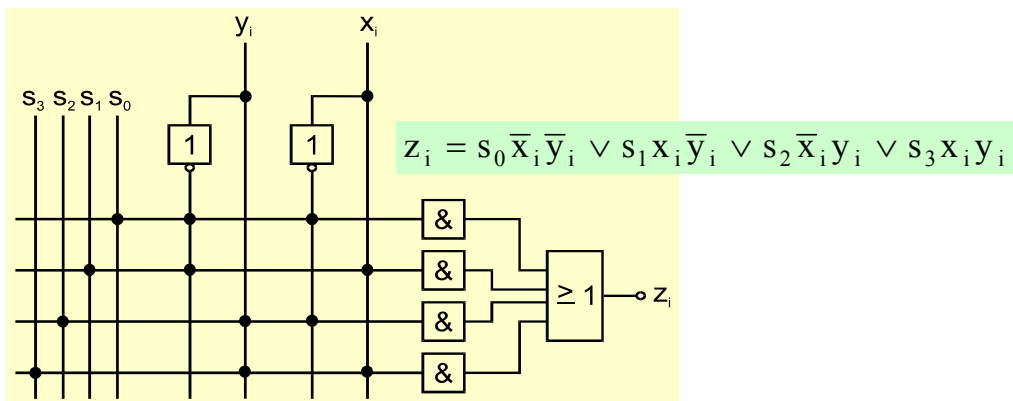
GIV II 3-38



ALU als Schaltkette bestehend aus 1 Bit-ALU-Slices

3.3 Addierer-Schaltnetze

GIV II 3-39



Universallogikbaustein (ULB) zur ALU-Realisierung:

- Bildung aller 16 möglichen zweistelligen booleschen Verknüpfungen von x_i, y_i gestattet auch die boolesche Realisierung aller möglichen arithmetischen Verknüpfungen dieser Variablen
- ≈ 3 ULBs für eine ALU: 2 ULB für z_i aus x_i, y_i, c_i , 1 ULB für c_{i+1}

3.3 Addierer-Schaltnetze

GIV II 3-40

3. Arithmetisch-Logische Einheiten

- 3.1 Zahlendarstellungen
- 3.2 Festkomma-Zahlenformate
- 3.3 Addierer-Schaltnetze
- 3.4 Schaltwerke zur binären Multiplikation

Multiplikation von Dualzahlen in Betrag-Vorzeichen-Darstellung

$$A = (-1)^{a_N} \cdot \sum_{i=0}^{N-1} a_i \cdot 2^i = (-1)^{a_N} \cdot |A|$$

a_N : Vorzeichenbit (= 0 für positive Zahlen)

$$P = A \cdot B = (-1)^{b_N} \cdot (-1)^{a_N} \cdot |A| \cdot |B| = (-1)^{a_N + b_N} \cdot |A| \cdot |B| = (-1)^{p_N} \cdot |P|$$

a_N	b_N	$a_N + b_N$	p_N
0	0	0	0
0	1	1	1
1	0	1	1
1	1	2	0

$$p_N = a_N \oplus b_N$$

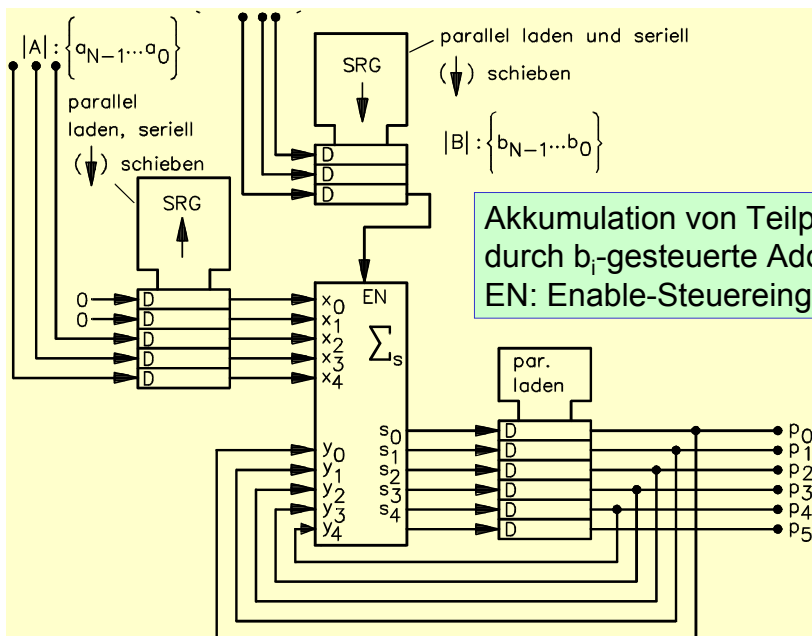
$$|P| = |A| \cdot |B| = |A| \cdot \sum_{i=0}^{N-1} b_i \cdot 2^i$$

$$= |A| \cdot b_0 + |A| \cdot b_1 \cdot 2 + |A| \cdot b_2 \cdot 2^2 + \dots + |A| \cdot b_{N-1} \cdot 2^{N-1}$$

	A				$b_{N-1} \dots$		b_1	b_0	
$ A \cdot B :$	1	0	1	1	.	1	1	0	1
$ A \cdot b_{N-1} \cdot 2^{N-1}$		1	0	1		1	0	0	0
$+ A \cdot b_{N-2} \cdot 2^{N-2}$	+		1	0		1	1	0	0
	1	0	0	0		0	1	0	0
$+ A \cdot b_1 \cdot 2$	+			0		0	0	0	0
	1	0	0	0		0	1	0	0
$+ A \cdot b_0$	+					1	0	1	1
	1	0	0	0		1	1	1	1

3.4 Schaltwerke zur binären Multiplikation

GIV II 3-43



3.4 Schaltwerke zur binären Multiplikation

GIV II 3-44

Multiplikation von Dualzahlen:

- Polynom-Multiplikation
- Faltungsgesetz zur Koeffizientenbestimmung
- Matrix-Vektor-Produkt

$$|P| = |A| \cdot |B| = \sum_{i=0}^{N-1} a_i \cdot 2^i \cdot \sum_{j=0}^{N-1} b_j \cdot 2^j \quad \text{mit } a_i, b_j \in \{0,1\}$$

$$= \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} a_i \cdot b_j \cdot 2^{i+j} \quad \text{mit } k = i + j$$

$$= \sum_{k=0}^{2N-1} \sum_{i=0}^{N-1} a_i \cdot b_{k-i} \cdot 2^k = \sum_{k=0}^{2N-2} q_k \cdot 2^k = \sum_{k=0}^{2N-1} p_k \cdot 2^k$$

$$|p| = a_0 \cdot b_0 + (a_0 \cdot b_1 + a_1 \cdot b_0) \cdot 2^1 + (a_0 \cdot b_2 + a_1 \cdot b_1 + a_2 \cdot b_0) \cdot 2^2 + (a_1 \cdot b_2 + a_2 \cdot b_1) \cdot 2^3 + a_2 \cdot b_2 \cdot 2^4 \quad \text{Beispiel: 3 Bit-Zahlen}$$

3.4 Schaltwerke zur binären Multiplikation

GIV II 3-45

$$\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ \vdots \\ \vdots \\ q_{N-1} \\ q_N \\ q_{N+1} \\ \vdots \\ \vdots \\ q_{2N-2} \end{bmatrix} = \begin{bmatrix} b_0 & 0 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ b_1 & b_0 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ b_2 & b_1 & b_0 & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & b_2 & b_1 & \cdot & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & b_2 & \cdot & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ b_{N-1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & b_0 \\ \cdot & b_{N-1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & b_1 \\ \cdot & \cdot & b_{N-1} & \cdot & \cdot & \cdot & \cdot & \cdot & b_2 \\ 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & b_{N-1} \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ a_{N-1} \end{bmatrix}$$

- Koeffizientenbestimmung bei der Dualzahlen – Multiplikation durch Matrix-Vektor-Produkt

3.4 Schaltwerke zur binären Multiplikation

GIV II 3-46

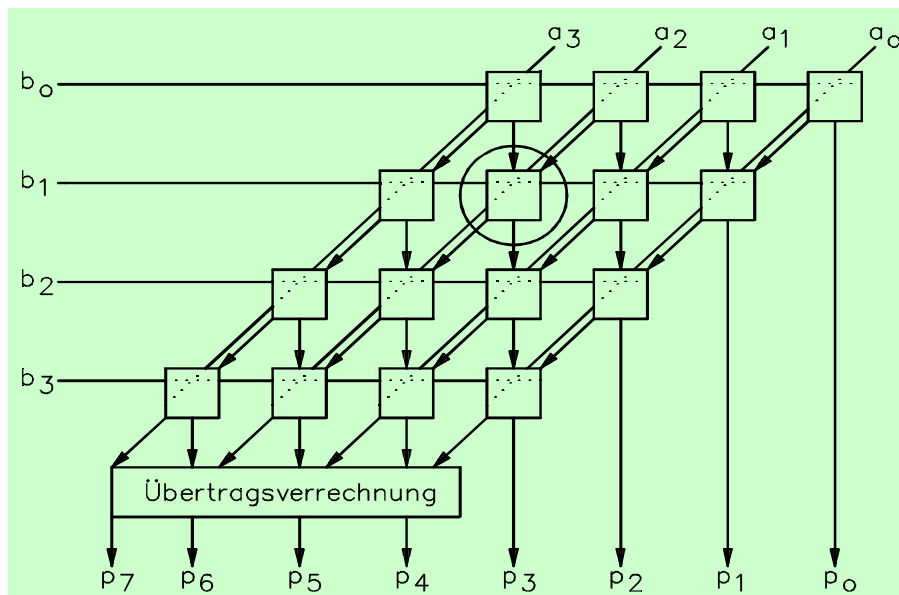
Beispiel: Multiplikation 3 Bit-Dualzahlen - Matrix-Vektor-Produkt

$$|p| = a_0 \cdot b_0 + (a_1 \cdot b_0 + a_0 \cdot b_1) \cdot 2^1 + (a_2 \cdot b_0 + a_1 \cdot b_1 + a_0 \cdot b_2) \cdot 2^2 + (a_2 \cdot b_1 + a_1 \cdot b_2) \cdot 2^3 + a_2 \cdot b_2 \cdot 2^4$$

$$\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} a_0 & 0 & 0 \\ a_1 & a_0 & 0 \\ a_2 & a_1 & a_0 \\ 0 & a_2 & a_1 \\ 0 & 0 & a_2 \end{bmatrix} \cdot \begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix}$$

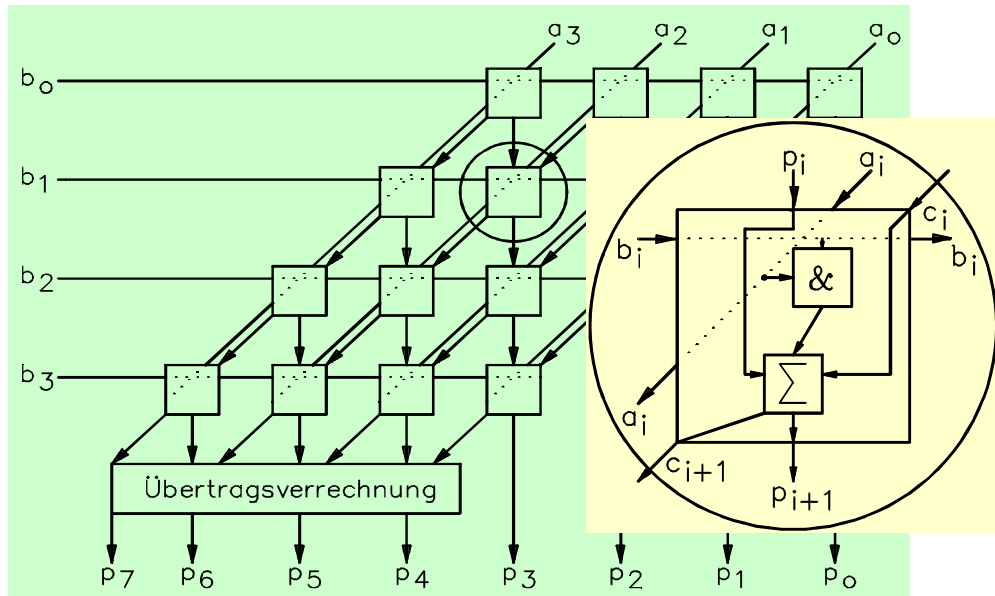
3.4 Schaltwerke zur binären Multiplikation

GIV II 3-47



3.4 Schaltwerke zur binären Multiplikation

GIV II 3-48

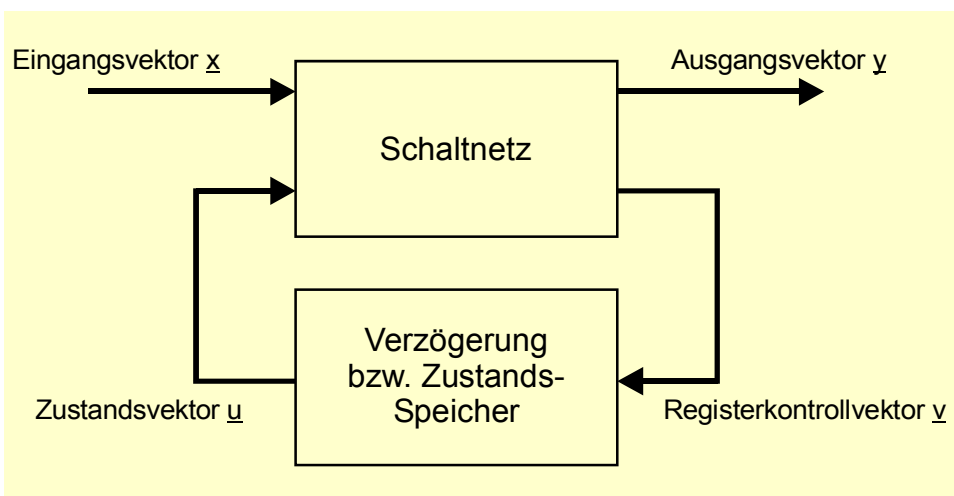


3.4 Schaltwerke zur binären Multiplikation

GIV II 3-49

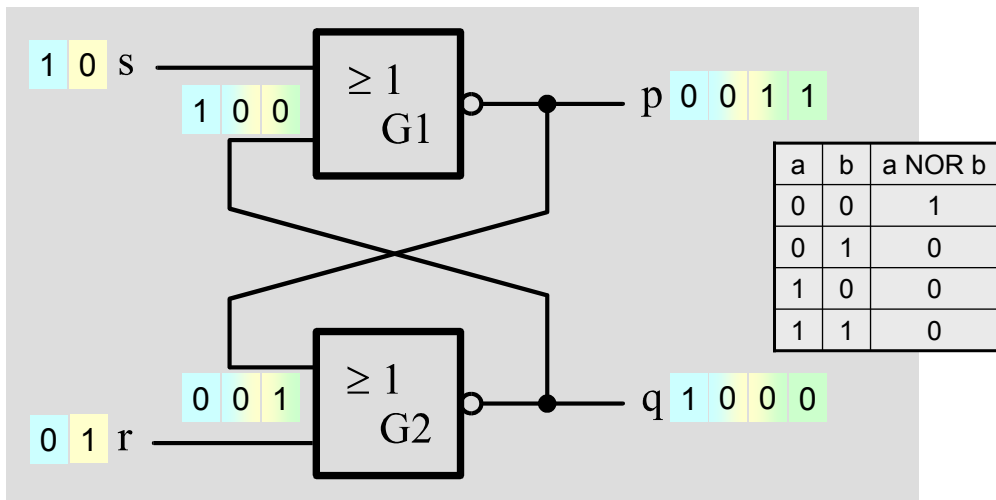
4. Schaltwerke

- 4.1 Begriff und Funktion des Schaltwerks
- 4.2 Elementare Schaltwerke
- 4.3 Formale Schaltwerkbeschreibung
- 4.4 Schaltwerkentwurf
- 4.5 Verbindungen zur Automatentheorie

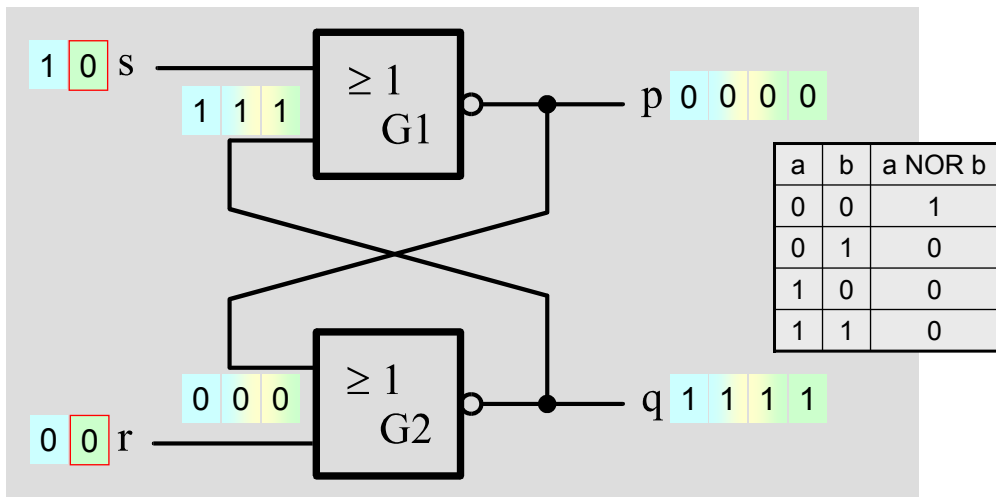


Eigenschaften:

- Speicher
- algorithmische Abläufe
- Rückkopplung
- sequentielle Algorithmen



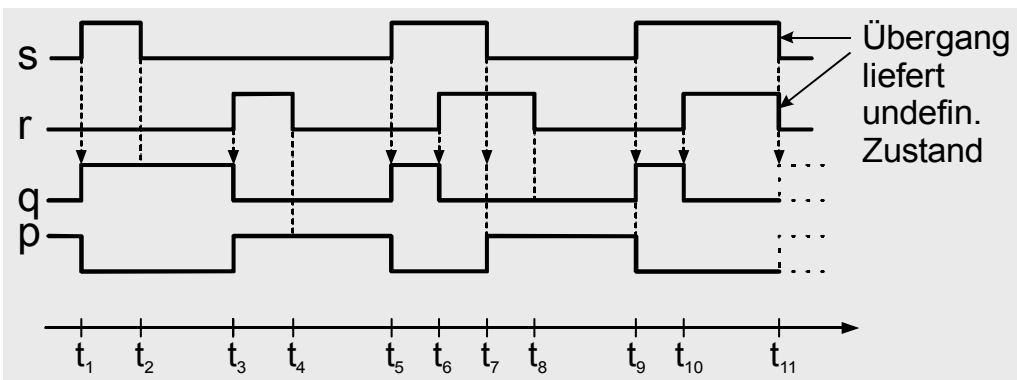
RS- (SR-) FlipFlop aus NOR-Gattern: Setzen



RS- (SR-) FlipFlop aus NOR-Gattern: Speichern

Eingangsvariablen		Zustandsvariablen		Ausgangsvariablen		Zustandsvariablen	
s	r	q^n	p^n	q^{n+1}	p^{n+1}	q^{n+1}	
0	0	0	1	0	1	q^n	Speichern
0	0	1	0	1	0	q^n	
0	1	0	1	0	1	0	Rücksetzen
0	1	1	0	0	1	0	
1	0	0	1	1	0	1	Setzen
1	0	1	0	1	0	1	
1	1	0	1	0	0	-	verboten
1	1	1	0	0	0	-	

RS- (SR-) FlipFlop aus NOR-Gattern: Wahrheitstabelle



RS- (SR-) FlipFlop aus NOR-Gattern: Signal-Zeit-Verlauf

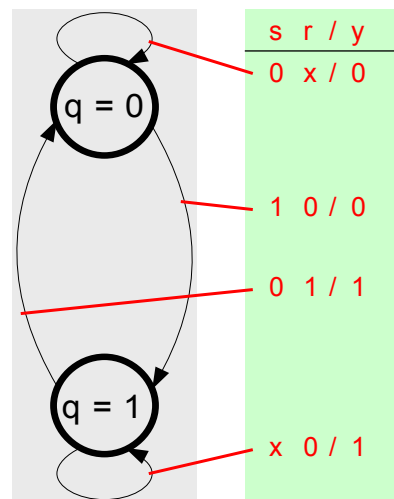
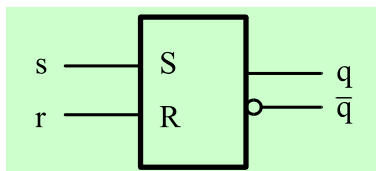
Zustandsgraph zur Charakterisierung von Schaltwerken:

- gerichteter (endlicher) Graph
- Knoten repräsentiert eindeutig stabilen Zustand (eintragen!)
- gerichtete Kante repräsentiert eindeutig Zustandsübergang
- Pfeile kennzeichnen Wirkungsübergang
- Kantenbeschriftung:
 - übergangsbewirkenden Eingangsvektor
 - durch Schrägstrich getrennt: Ausgangsvektor des alten Zustands
- zu jedem Knoten (Zustand) führt mindestens eine Kante
- alle Abläufe sind sequentiell, keine parallelen Läufe
- k Zustandsvariable $\Rightarrow$ höchstens 2^k Zustände (Knoten)
- m Eingangsvariable $\Rightarrow$ höchstens 2^m Übergänge (Kanten)

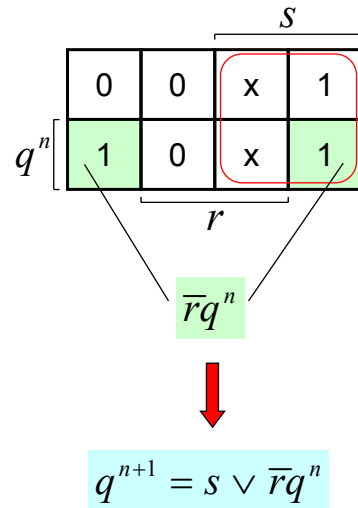
Zustandsgraph des SR - FlipFlop

Eingangsvektor $\underline{x} = (s, r)$

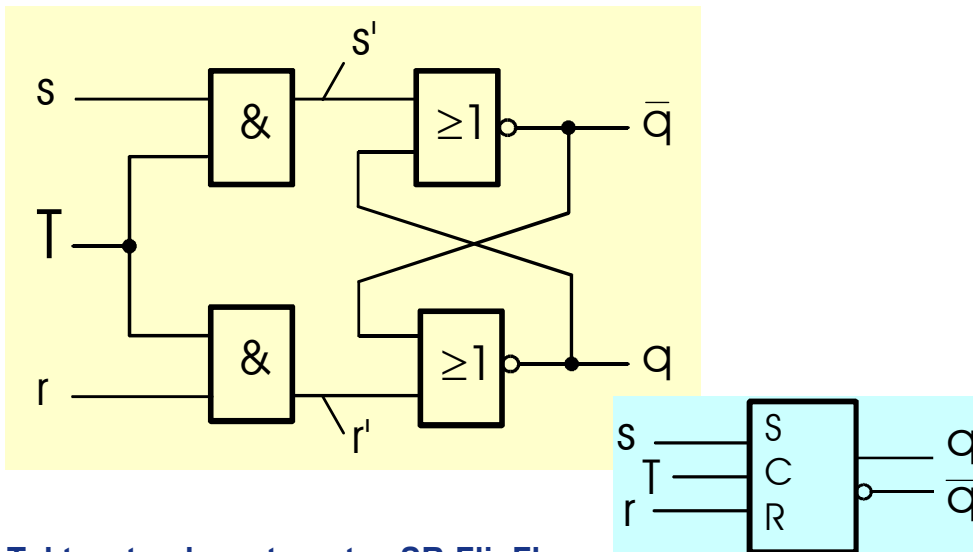
Ausgangsvariable $y = q$

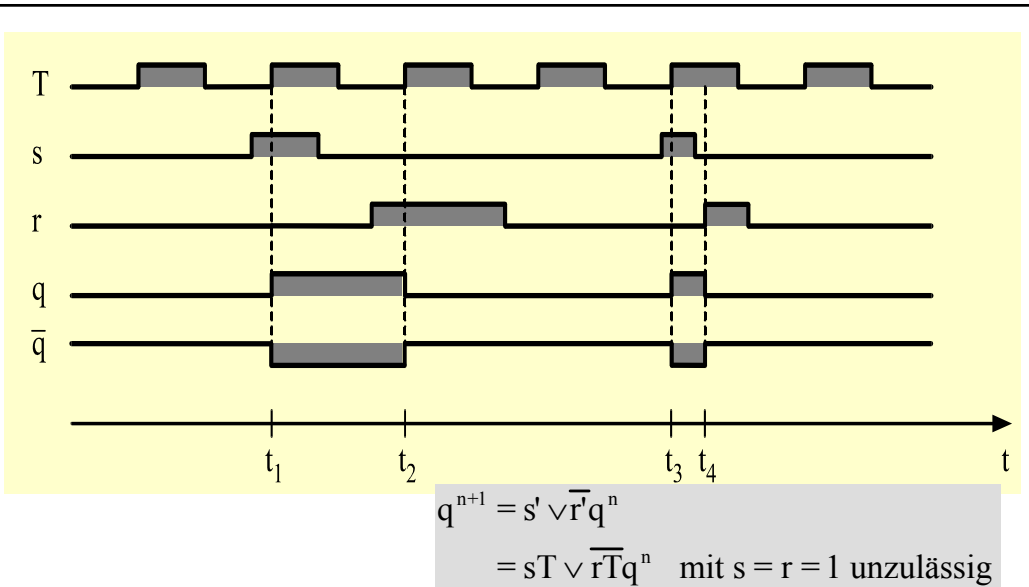


Eingangsvariablen		Zustand n	Zustand n+1	
s	r	q^n	q^{n+1}	
0	0	0	0	Speichern
0	0	1	1	
0	1	0	0	Rücksetzen
0	1	1	0	
1	0	0	1	Setzen
1	0	1	1	
1	1	0	x	verboten $\Rightarrow$ don't care
1	1	1	x	

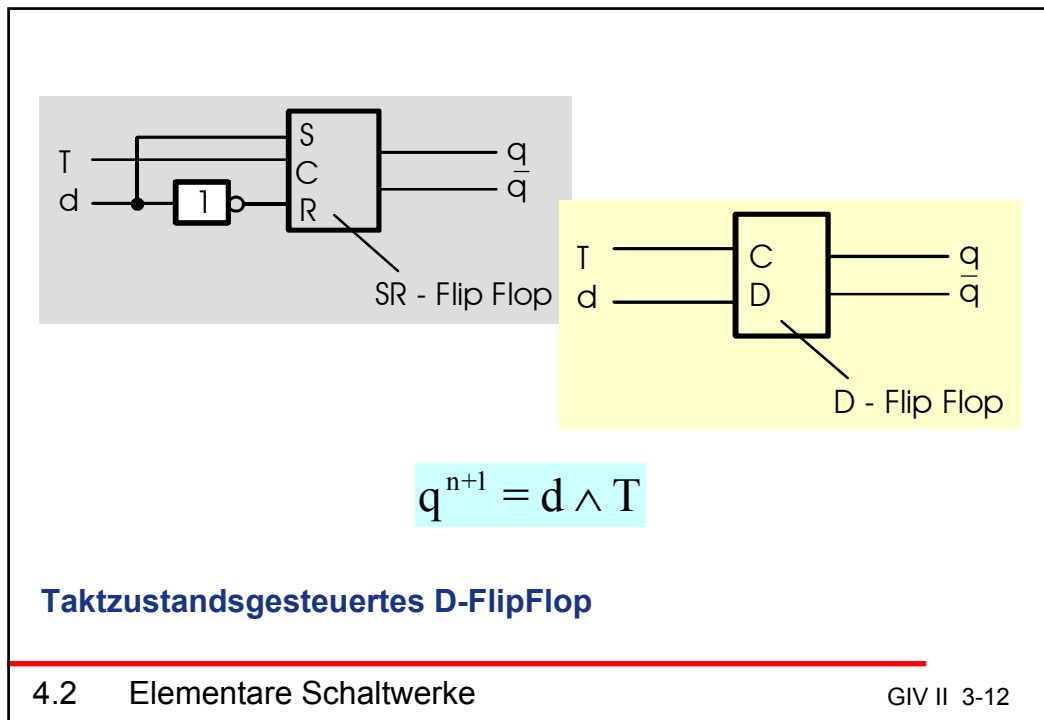


SR-FlipFlop: Übergangstabelle und -gleichung, KV-Diagramm

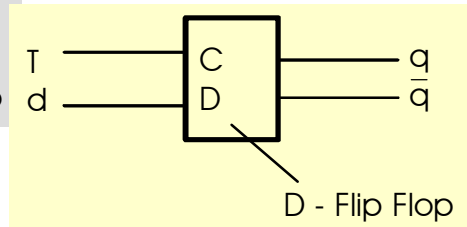
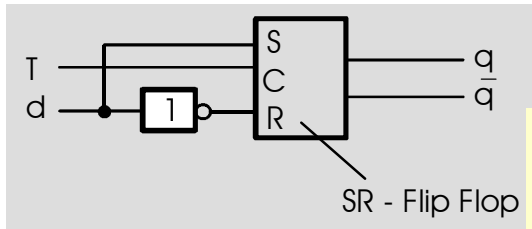




Taktzustandsgesteuertes SR-FlipFlop

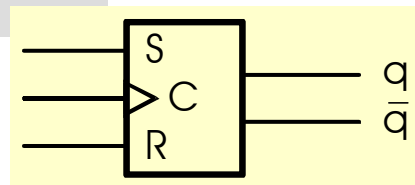
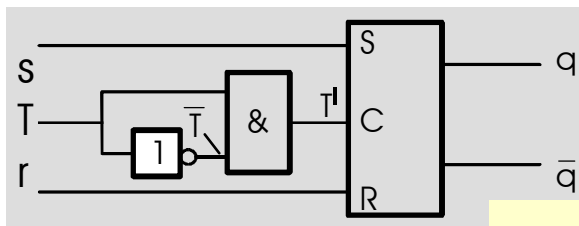


Taktzustandsgesteuertes D-FlipFlop

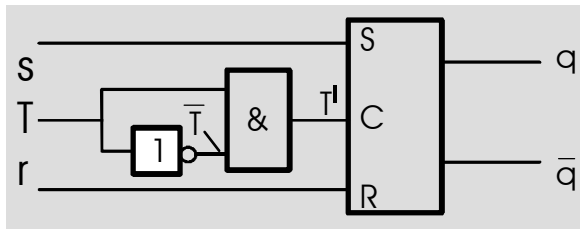


$$q^{n+1} = d \wedge T$$

Taktzustandsgesteuertes D-FlipFlop

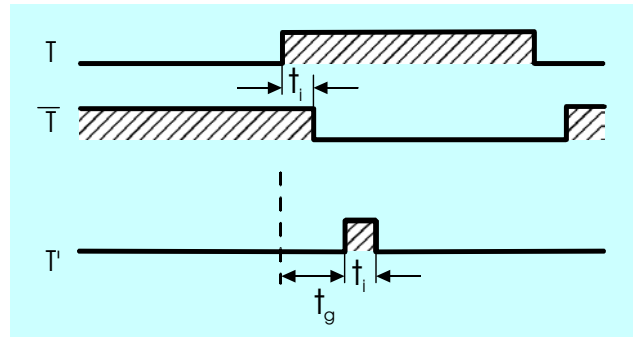


Taktflankengesteuertes SR-FlipFlop



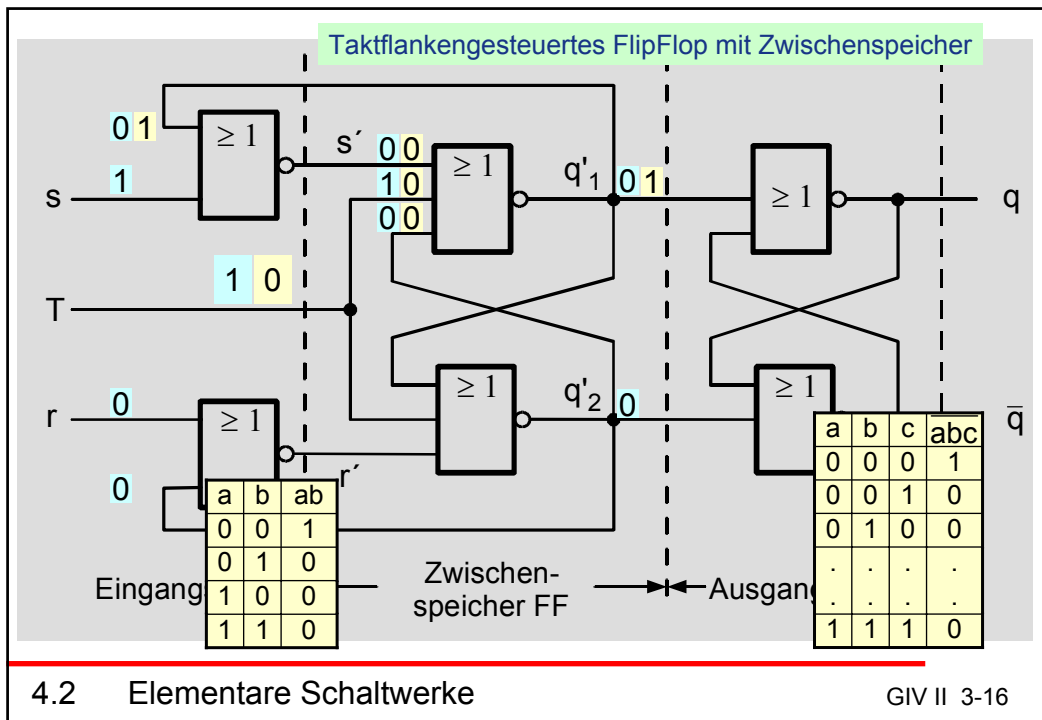
Taktflankengesteuertes SR-FlipFlop

t_i : Inverterlaufzeit
 t_g : & - Gatterlaufzeit



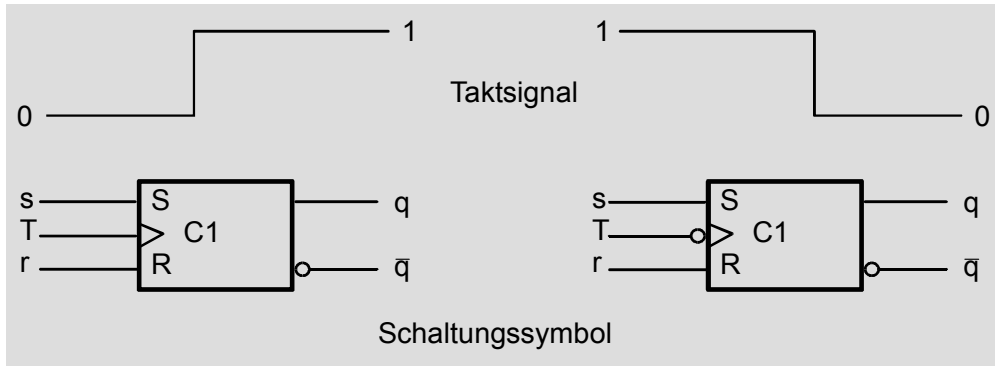
4.2 Elementare Schaltwerke

GIV II 3-15



4.2 Elementare Schaltwerke

GIV II 3-16



$$q^{n+1} = sT_n \vee \overline{rT_n}q^n$$

mit $srT_n = 0$

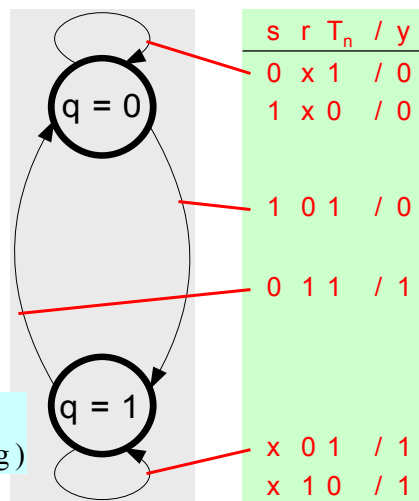
$$T_p = \begin{cases} 1 & \text{für } T = (0/1) \text{ (positive Flanke)} \\ 0 & \text{sonst} \end{cases}$$

$$T_n = \begin{cases} 1 & \text{für } T = (1/0) \text{ (negative Flanke)} \\ 0 & \text{sonst} \end{cases}$$

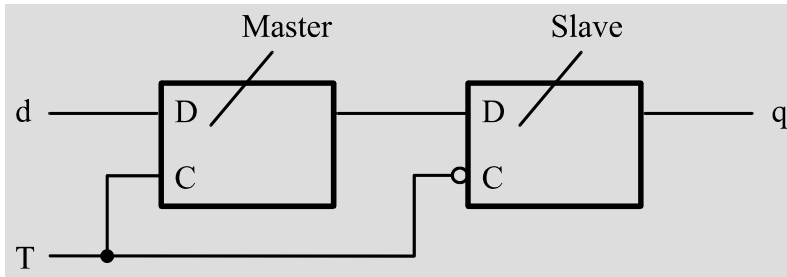
**Dynamische
(taktflankengesteuerte)
FlipFlops**

**Zustandsgraph des
negativ
taktflanken-gesteuerten
SR - FlipFlops**

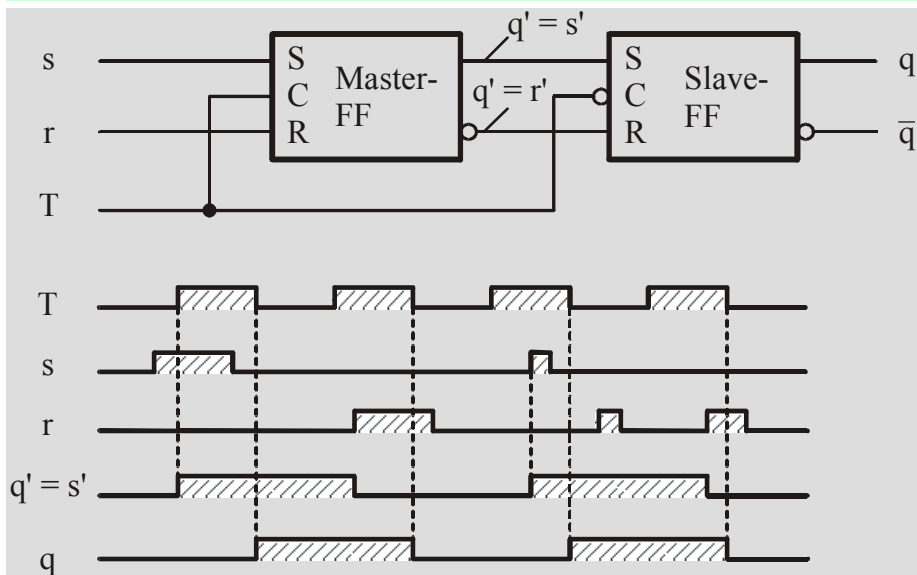
vereinfachte Gleichung:
 $q^{n+1} = s \vee \overline{r}q^n$ ($s = r = 1$ unzulässig)



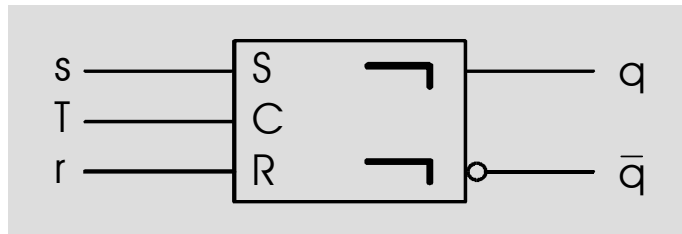
Master-Slave-FlipFlop (entkoppelte Übernahme und Weitergabe)



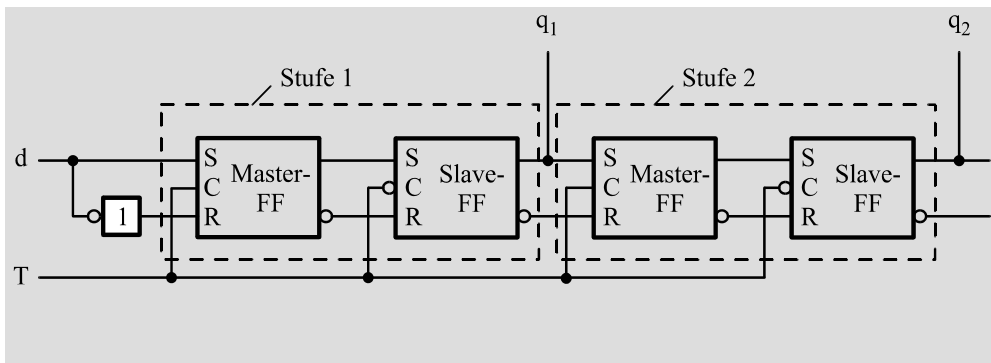
SR-Master-Slave-FlipFlop (2 taktzustandsgesteuerte SR-FF)

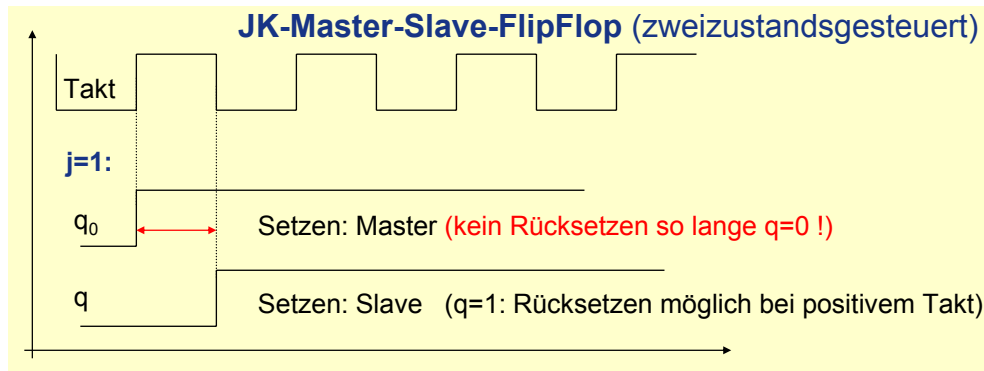


zweizustandsgesteuertes SR-Master-Slave-FlipFlop Schaltsymbol



Schieberegister aus SR-Master-Slave-FlipFlop





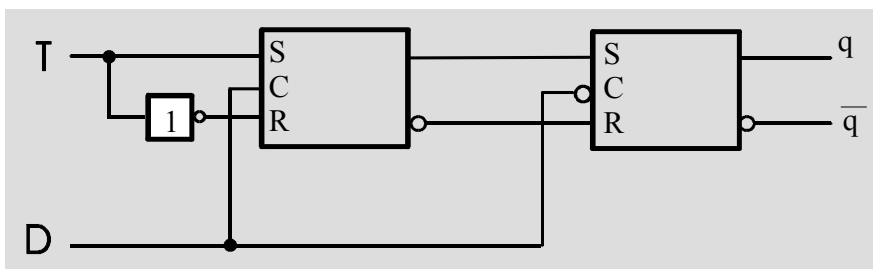
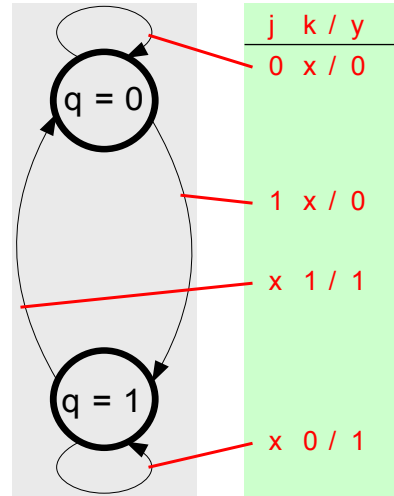
GIV II 3-23

JK-Master-Slave-FlipFlop (Übergangsverhalten)

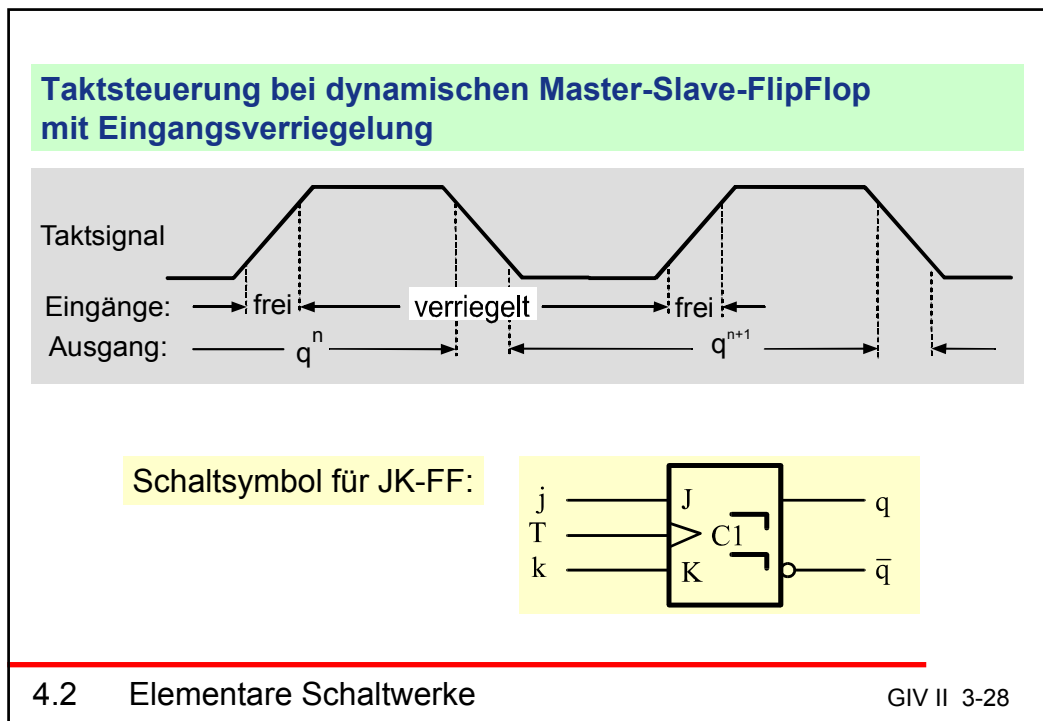
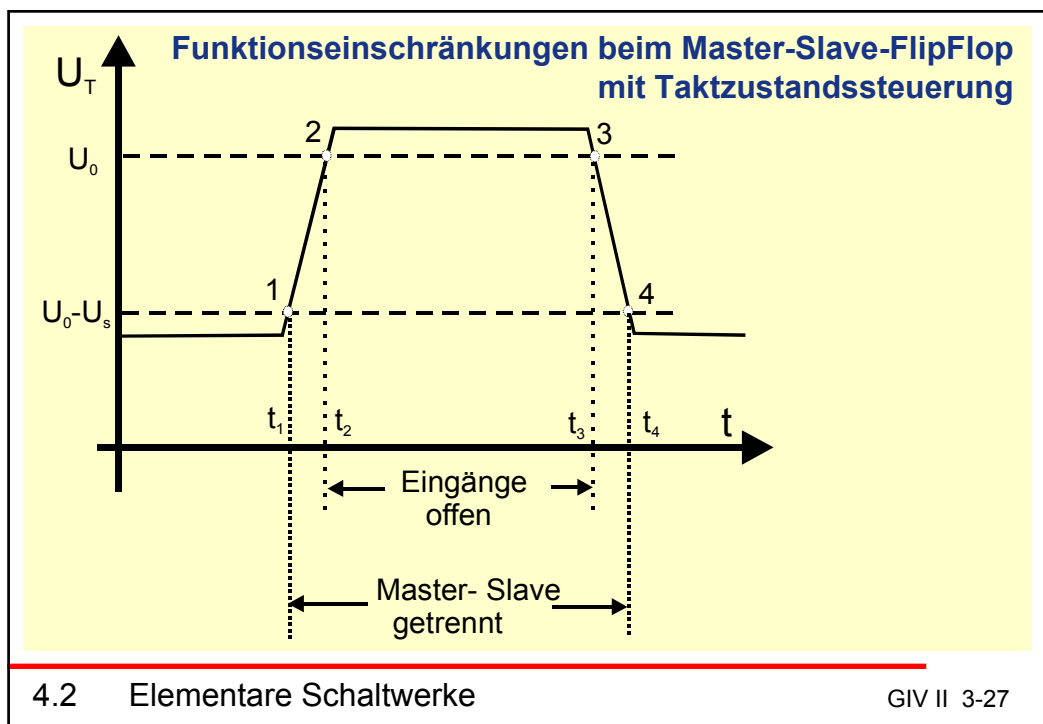
GIV II 3-24

Zustandsgraph des JK - FlipFlop

- (1) Z-wechsel $q^n = 0 \rightarrow q^{n+1} = 1$ bei $j=1, k=x$
- (2) Z-Wechsel $q^n = 1 \rightarrow q^{n+1} = 0$ bei $j=x, k=1$
- (3) Z-Wechsel $q^{n+1} = \overline{q^n}$ bei $j=1, k=1$
- (4) Z-Erhalt $q^{n+1} = q^n$ bei $j=0, k=0$



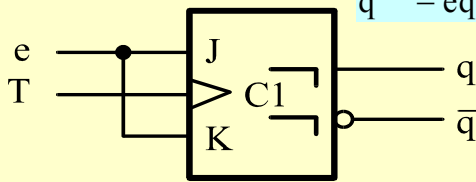
Funktionseinschränkungen beim Master-Slave-FlipFlop mit Taktzustandssteuerung



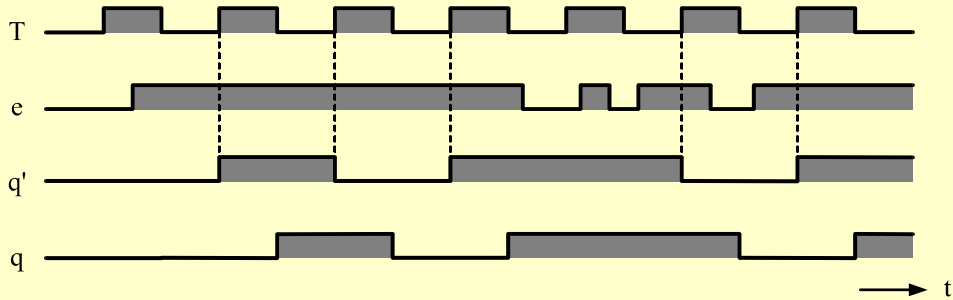
Toggle-MS-FlipFlop

$$q^{n+1} = j\bar{q}^n \vee \bar{k}q^n \quad \text{mit } j = k = e$$

$$q^{n+1} = e\bar{q}^n \vee \bar{e}q^n$$



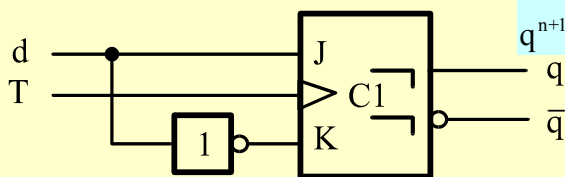
e	q^{n+1}
0	q^n
1	$\bar{q}^n$



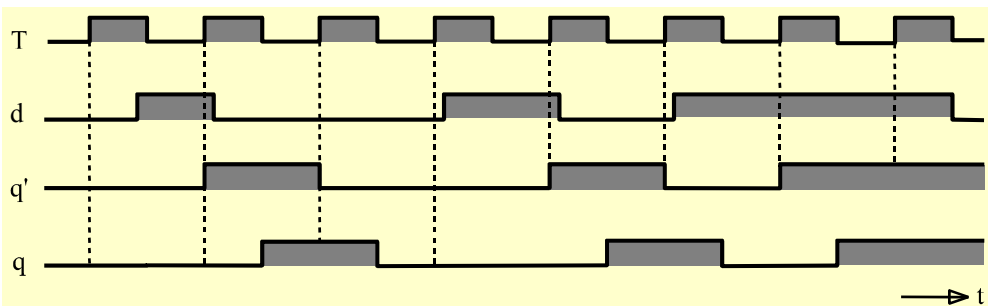
D-MS-FlipFlop

$$q^{n+1} = j\bar{q}^n \vee \bar{k}q^n \quad j = \bar{k} = d$$

$$q^{n+1} = d\bar{q}^n \vee d q^n = d$$

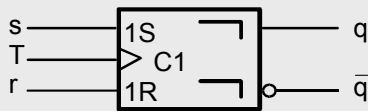


d	q^{n+1}
0	0
1	1



Typen zweiflankengesteuerter MS-FlipFlops (1)

SR - Flip Flop

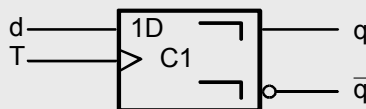


s	r	q^{n+1}
0	0	q^n
0	1	0
1	0	1
1	1	verboten

$$q^{n+1} = s \vee \bar{r} q^n$$

mit $s \wedge r = 0$

D - Flip Flop



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

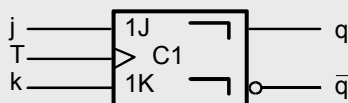
$$q^{n+1} = d$$

4.2 Elementare Schaltwerke

GIV II 3-31

Typen zweiflankengesteuerter MS-FlipFlops (2)

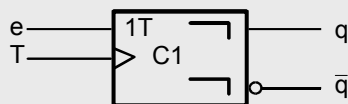
JK - Flip Flop



i	k	q^{n+1}
0	0	q^n
0	1	0
1	0	1
1	1	$\bar{q}^n$

$$q^{n+1} = j \bar{q}^n \vee \bar{k} q^n$$

Toggle-Flip Flop



e	q^{n+1}
0	q^n
1	$\bar{q}^n$

$$q^{n+1} = e \bar{q}^n \vee \bar{e} q^n$$

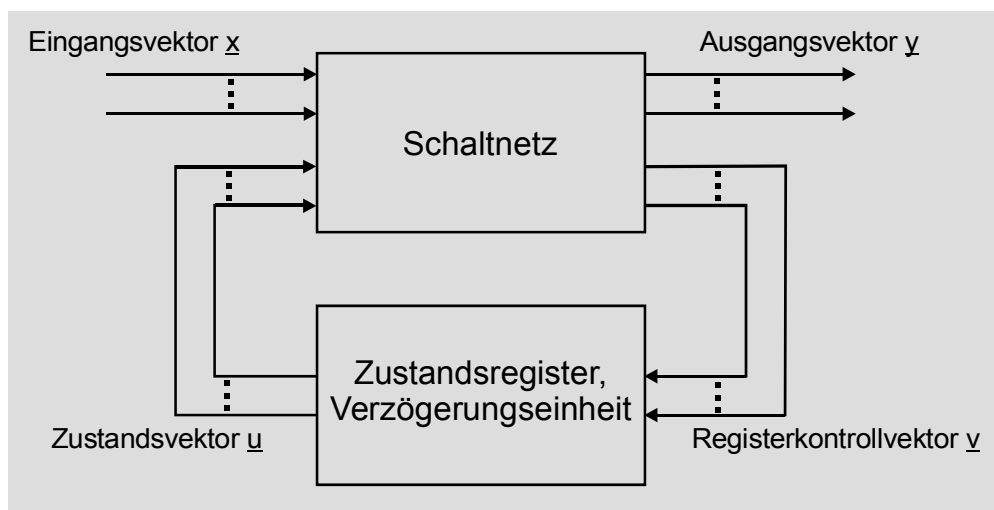
4.2 Elementare Schaltwerke

GIV II 3-32

4. Schaltwerke

- 4.1 Begriff und Funktion des Schaltwerks
- 4.2 Elementare Schaltwerke
- 4.3 **Formale Schaltwerkbeschreibung**
- 4.4 Schaltwerkentwurf
- 4.5 Verbindungen zur Automatentheorie

Schaltwerk, Blockschaltbild



Schaltwerk, Beschreibungsgrößen

- Eingangsvektor: $\underline{x}^n = \{x_1^n, x_2^n, \dots, x_M^n\}$
- Ausgangsvektor: $\underline{y}^n = \{y_1^n, y_2^n, \dots, y_L^n\} = f(\underline{x}^n, \underline{x}^{n-1}, \dots, \underline{x}^{n-r})$
- Zustandsvektor: $\underline{u}^n = \{u_1^n, u_2^n, \dots, u_K^n\}$
- Registerkontrollvektor: $\underline{v}^n = \{v_1^n, v_2^n, \dots, v_L^n\}$

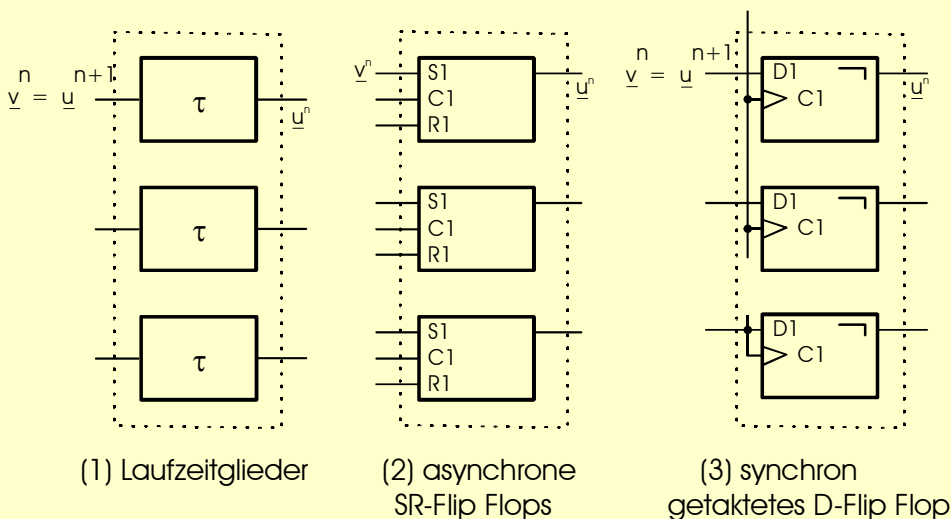
Schaltwerk, Kennfunktionen

- Ausgangsfunktion: $\underline{y}^n = f(\underline{x}^n, \underline{u}^n)$
- Registerkontrollfunktion: $\underline{v}^n = g_2(\underline{x}^n, \underline{u}^n)$
- Übergangsfunktion: $\underline{u}^{n+1} = g_1(\underline{v}^n)$
- Übergangsfunktion: $\underline{u}^{n+1} = g_1(g_2(\underline{x}^n, \underline{u}^n)) = g(\underline{x}^n, \underline{u}^n)$

4.3 Formale Schaltwerkbeschreibung

GIV II 3-35

Ausführungsvarianten für Schaltwerkregister bzw. -speicher



4.3 Formale Schaltwerkbeschreibung

GIV II 3-36

(A) Registerkontrollvektor gleich Zustandsvektor zum Zeitpunkt (n+1):

$$\underline{y}^n = \underline{u}^{n+1}$$

vereinfachte Beschreibungsgleichungen:

- Ausgangsfunktion: $\underline{y}^n = f(\underline{x}^n, \underline{u}^n)$
- Übergangsfunktion: $\underline{u}^{n+1} = g(\underline{x}^n, \underline{u}^n)$

**Vereinfachungen
bei Schaltwerken**

kein Registerkontrollvektor (z.B. D-FF, Verzögerung τ)

(B) Zustandsvektor zum Zeitpunkt (n+1) gleich Ausgangsvektor:

$\underline{u}^{n+1} = \underline{y}^n$ (z.B. bei Zähler):

- Ausgangsfunktion: $\underline{y}^n = f(\underline{x}^n, \underline{u}^n) = \underline{u}^{n+1}$

Übergangsfunktion gleich Ausgangsfunktion

(C) Schaltwerk von außen nicht beeinflussbar: **autonomes Schaltwerk**

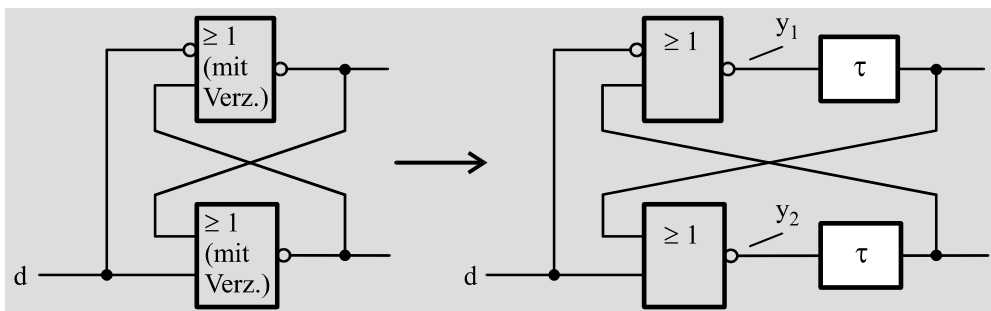
es gilt mit Eigenschaft (A):

- Ausgangsfunktion: $\underline{y}^n = f(\underline{u}^n)$
- Übergangsfunktion: $\underline{u}^{n+1} = g(\underline{u}^n)$

4.3 Formale Schaltwerkbeschreibung

GIV II 3-37

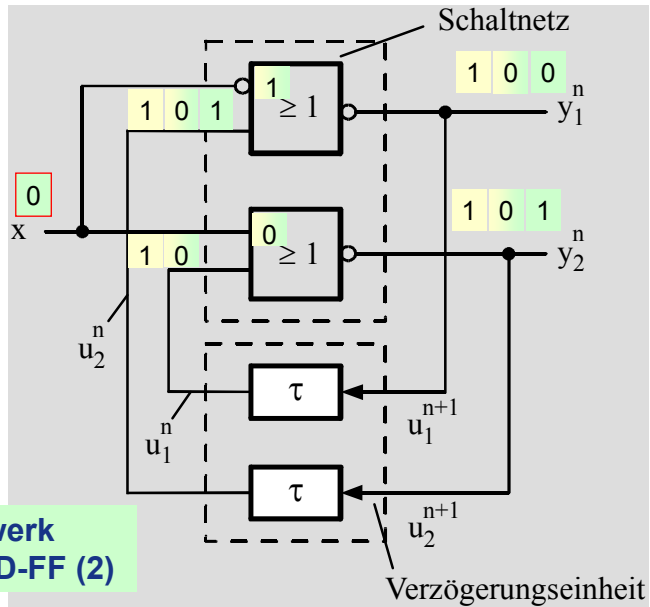
Asynchrones Schaltwerk mit nicht getaktetem D-FF (1)



4.3 Formale Schaltwerkbeschreibung

GIV II 3-38

a	b	a NOR b
0	0	1
0	1	0
1	0	0
1	1	0



Asynchrones Schaltwerk mit nicht getaktetem D-FF (2)

4.3 Formale Schaltwerkbeschreibung

GIV II 3-39

Asynchrones Schaltwerk mit nicht getaktetem D-FF (3)

x^n	u_1^n	u_2^n	Zustands-Übergänge	$u_1^{n+1} = y_1^n$	$u_2^{n+1} = y_2^n$
0	0	0	→	0	1
0	0	1	→	0	1
0	1	0	→	0	0
0	1	1	→	0	0
1	0	0	→	1	0
1	0	1	→	0	0
1	1	0	→	1	0
1	1	1	→	0	0

Durchlauf von Übergangsphasen ("Races"), Beschreibung:

$$y_1^n = x^n \overline{u_1^n} \overline{u_2^n} \vee x^n u_1^n \overline{u_2^n} = x^n \overline{u_2^n}$$

$$= x^n \vee u_2^n = u_1^{n+1}$$

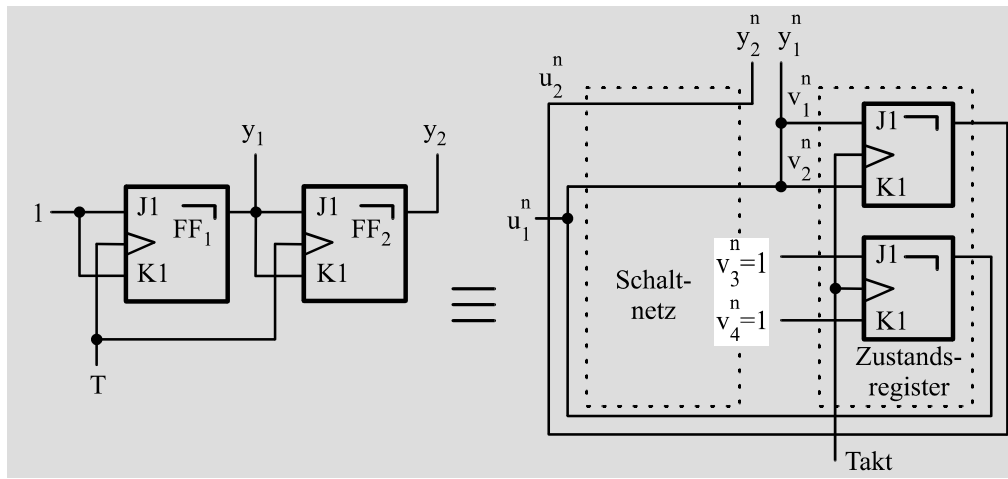
$$y_2^n = \overline{x^n} \overline{u_1^n} \overline{u_2^n} \vee x^n \overline{u_1^n} u_2^n = \overline{x^n} \overline{u_1^n}$$

$$= \overline{x^n} \vee u_1^n = u_2^{n+1}$$

4.3 Formale Schaltwerkbeschreibung

GIV II 3-40

Synchrones Schaltwerk: Modulo 4-Zähler (1)



4.3 Formale Schaltwerkbeschreibung

GIV II 3-41

Synchrones Schaltwerk: Modulo 4-Zähler (2)

u_2^n	u_1^n	v_2^n	v_1^n	$v_3^n = v_4^n$	u_2^{n+1}	u_1^{n+1}	y_2^n	y_1^n
0	0	0	0	1	0	1	0	0
0	1	1	1	1	1	0	0	1
1	0	0	0	1	1	1	1	0
1	1	1	1	1	0	0	1	1

e	q^{n+1}
0	q^n
1	$\overline{q^n}$

$$q^{n+1} = \overline{e}q^n \vee e\overline{q^n}$$

4.3 Formale Schaltwerkbeschreibung

GIV II 3-42

u_2^n	u_1^n	v_2^n	v_1^n	$v_3^n = v_4^n$	u_2^{n+1}	u_1^{n+1}	y_2^n	y_1^n
0	0	0	0	1	0	1	0	0
0	1	1	1	1	1	0	0	1
1	0	0	0	1	1	1	1	0
1	1	1	1	1	0	0	1	1

Ausgangsfunktion $\underline{y}^n = \underline{u}^n$ (Ausgangsvektor = Zustandsvektor)

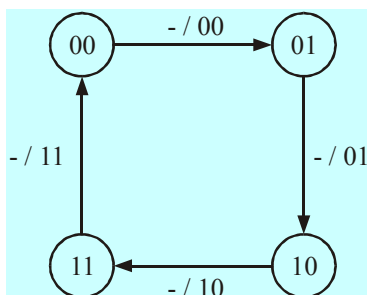
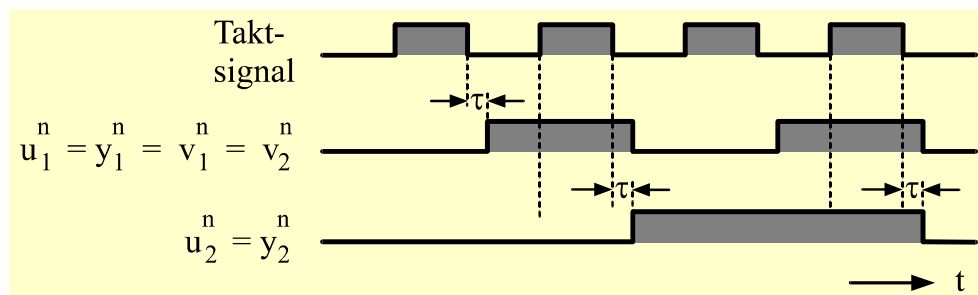
Registerkontrollfunktion $v_1^n = v_2^n = u_1^n$, $v_3^n = v_4^n = 1$

Übergangsfunktion $u_1^{n+1} = \overline{u_1^n}$
 $u_2^{n+1} = v_1^n \overline{u_2^n} \vee \overline{v_1^n} u_2^n = u_1^n \overline{u_2^n} \vee \overline{u_1^n} u_2^n$
 (Vergleiche JK-Flip Flop: $q^{n+1} = jq^n \vee \overline{k}q^n$)

Synchrones Schaltwerk: Modulo 4-Zähler (3)

4.3 Formale Schaltwerkbeschreibung

GIV II 3-43



**Synchrones Schaltwerk:
Modulo 4-Zähler (4)**
kein Eingangssignal: autonom

4.3 Formale Schaltwerkbeschreibung

GIV II 3-44

Modulo-2-Zählwerk, extern einstellbar (1)

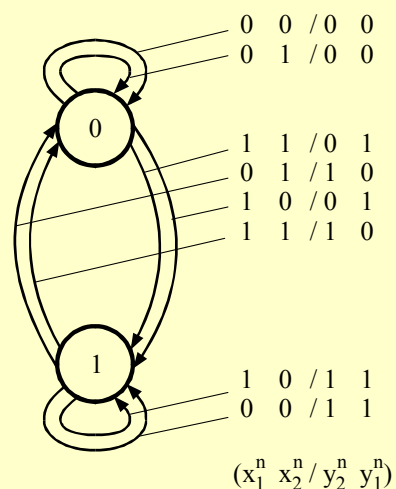
Einstellbare Funktionen:

- Stillstand mit Reset (Nullsetzen des Zustandsspeichers)
- Stillstand mit Speicherung des aktuellen Zustandes
- Stillstand mit Voreinstellung (Preset)
- Zählen

Ausgangssignale: aktuelle und folgende Zustandssignale

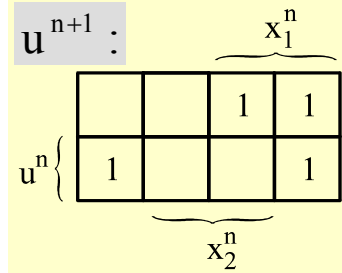
Modulo-2-Zählwerk, extern einstellbar (2)

eingestellte Funktion	x_1^n	x_2^n	u^n	u_1^{n+1}	y_2^n	y_1^n
Stillstand und Speichern	0	0	0	0	0	0
	0	0	1	1	1	1
Stillstand und Reset	0	1	0	0	0	0
	0	1	1	0	1	0
Stillstand und Voreinstellen	1	0	0	1	0	1
	1	0	1	1	1	1
Zählen	1	1	0	1	0	1
	1	1	1	0	1	0



Modulo-2-Zählwerk, extern einstellbar (3)

eingestellte Funktion	x_1^n	x_2^n	u^n	u_1^{n+1}	y_2^n	y_1^n
Stillstand und Speichern	0	0	0	0	0	0
	0	0	1	1	1	1
Stillstand und Reset	0	1	0	0	0	0
	0	1	1	0	1	0
Stillstand und Voreinstellen	1	0	0	1	0	1
	1	0	1	1	1	1
Zählen	1	1	0	1	0	1
	1	1	1	0	1	0



$$u^{n+1} = x_1^n \overline{u^n} \vee \overline{x_2^n} u^n$$

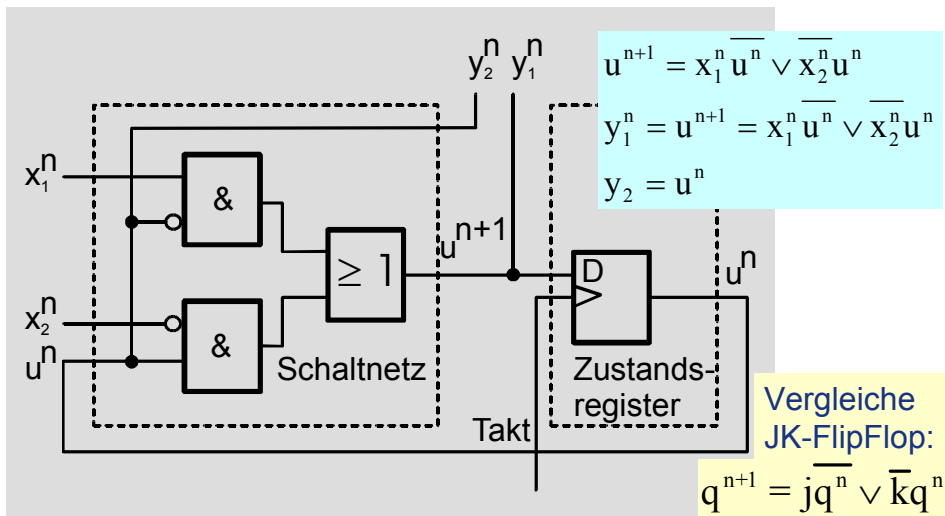
$$y_1^n = u^{n+1} = x_1^n \overline{u^n} \vee \overline{x_2^n} u^n$$

$$y_2^n = u^n$$

4.3 Formale Schaltwerkbeschreibung

GIV II 3-47

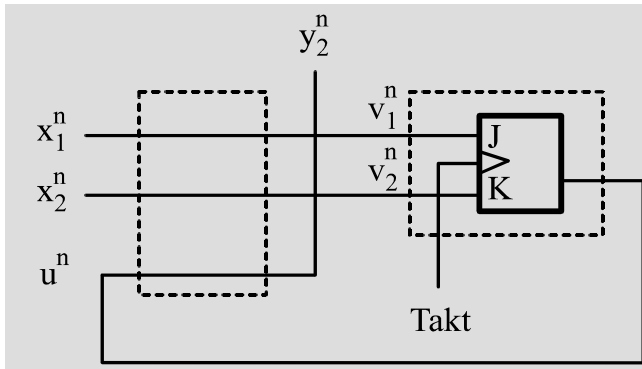
Modulo-2-Zählwerk, extern einstellbar (4)



4.3 Formale Schaltwerkbeschreibung

GIV II 3-48

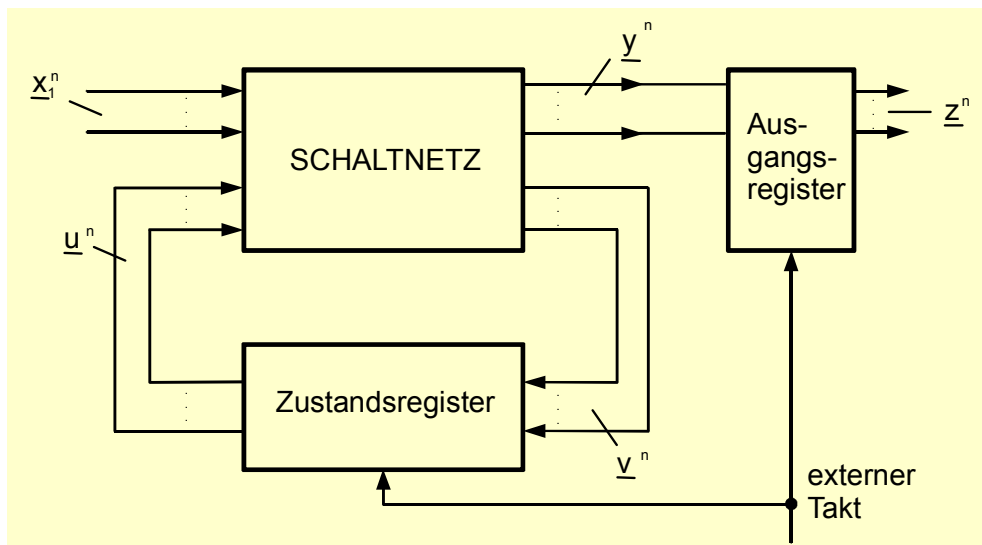
Modulo-2-Zählwerk, extern einstellbar (5)



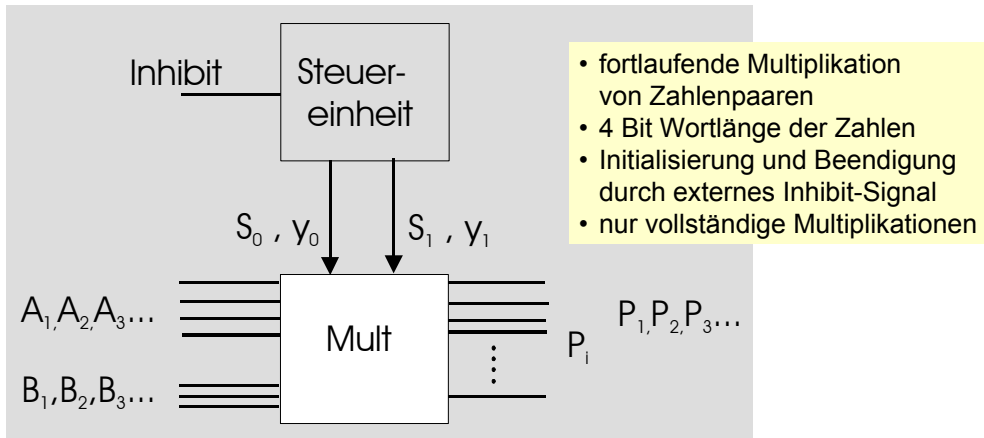
Vergleiche
JK-FlipFlop:

$$q^{n+1} = j\bar{q}^n \vee \bar{k}q^n$$

Schaltwerk mit getaktetem Ausgangsregister



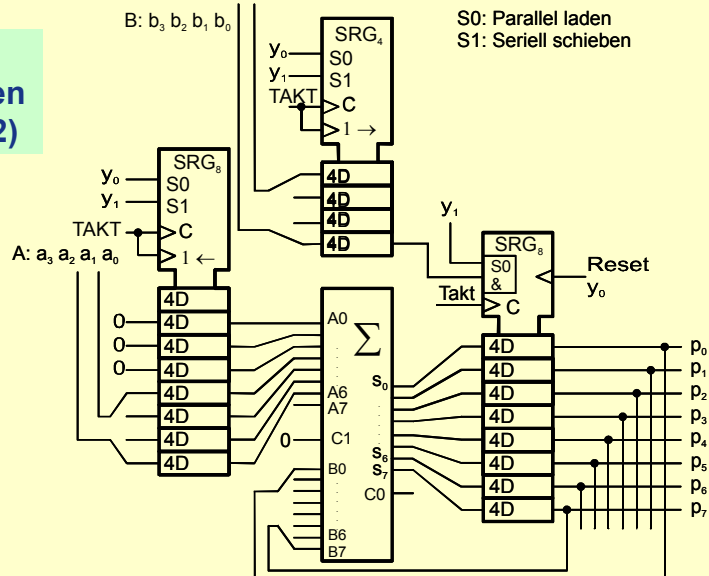
Steuerwerk für parallel-seriellen Multiplizierer(1)



4.3 Formale Schaltwerkbeschreibung

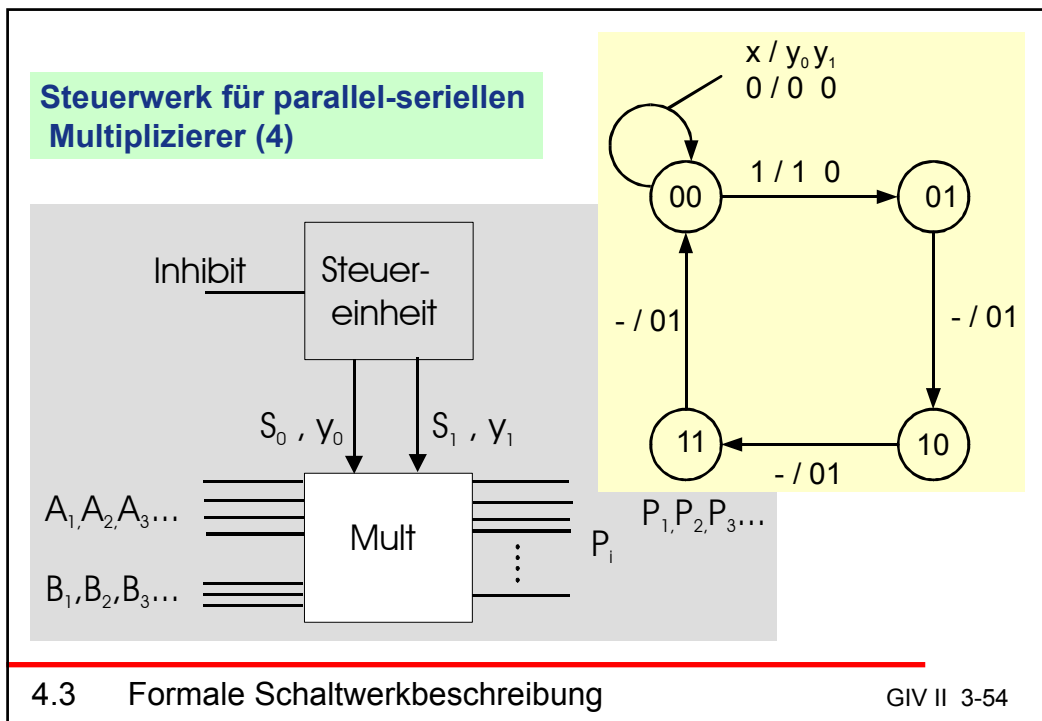
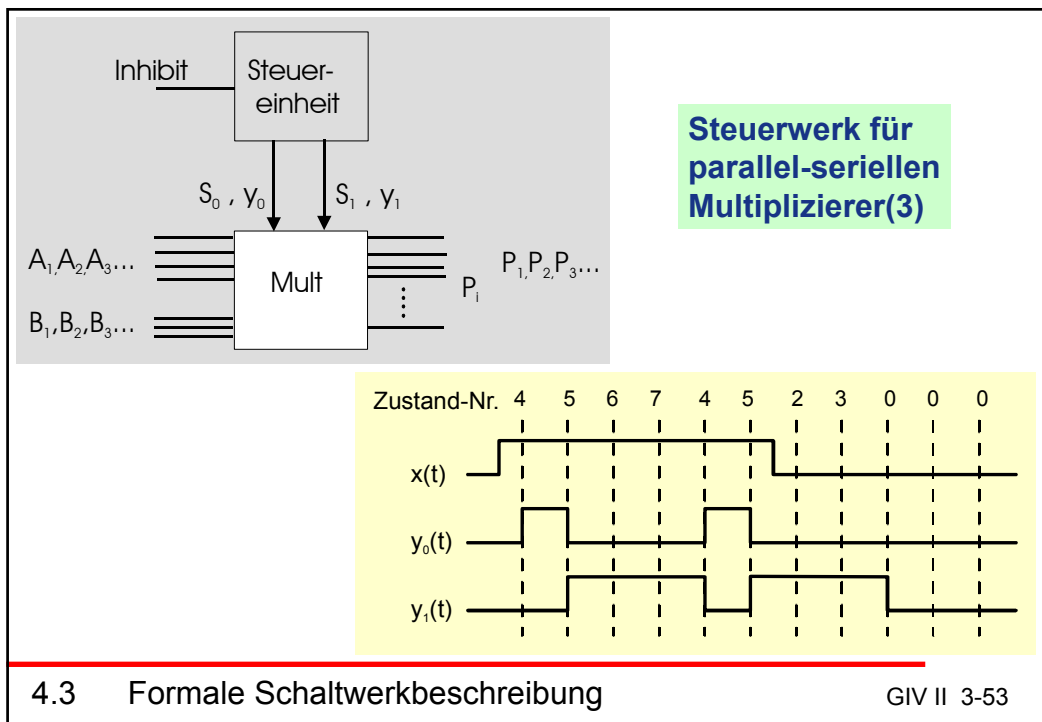
GIV II 3-51

Steuerwerk für parallel-seriellen Multiplizierer (2)



4.3 Formale Schaltwerkbeschreibung

GIV II 3-52



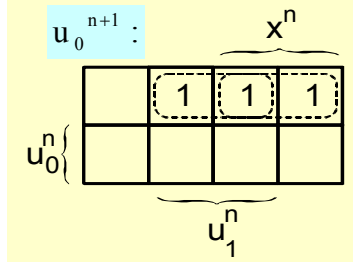
$$u_0^{n+1} = x^n \overline{u_0^n} \vee u_1^n \overline{u_0^n} = \overline{u_0^n} (x^n \vee u_1^n)$$

$$u_1^{n+1} = \overline{u_1^n} u_0^n \vee u_1^n \overline{u_0^n} = u_1^n \neq u_0^n$$

$$y_0^n = x^n u_1^n u_0^n$$

$$y_1^n = u_1^n \vee u_0^n$$

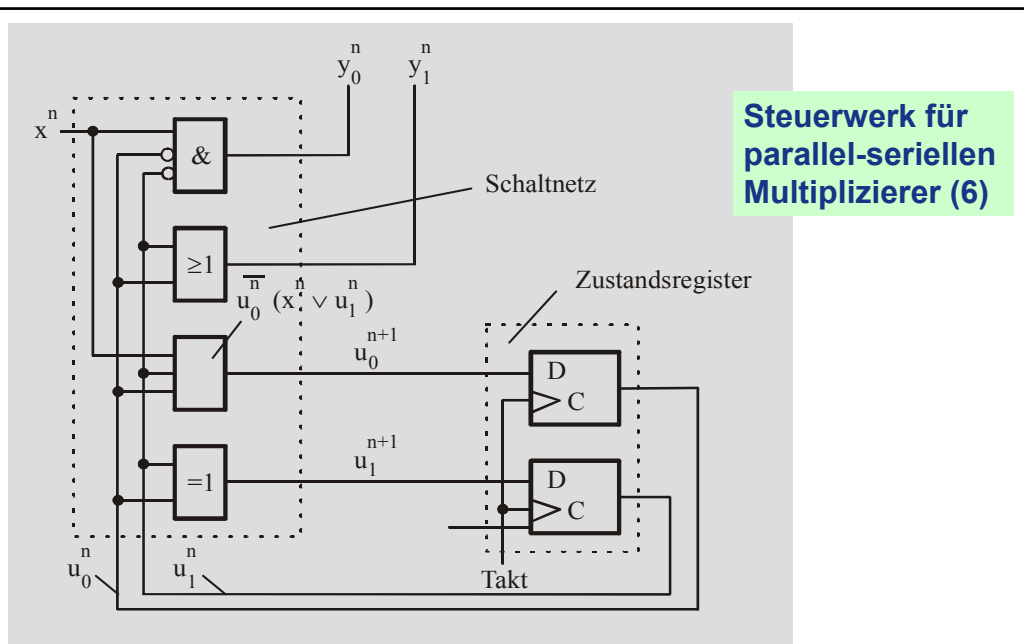
**Steuerwerk für
parallel-seriellen
Multiplikierer (5)**



Zustand-Nr	x^n	u_1^n	u_2^n		u_1^{n+1}	u_2^{n+1}	y_0^n	y_1^n
0	0	0	0	→	0	0	0	0
1	0	0	1	↘	1	0	0	1
2	0	1	0	→	1	1	0	1
3	0	1	1	↘	0	0	0	1
4	1	0	0	→	0	1	1	0
5	1	0	1	↘	1	0	0	1
6	1	1	0	→	1	1	0	1
7	1	1	1	↘	0	0	0	1

4.3 Formale Schaltwerkbeschreibung

GIV II 3-55

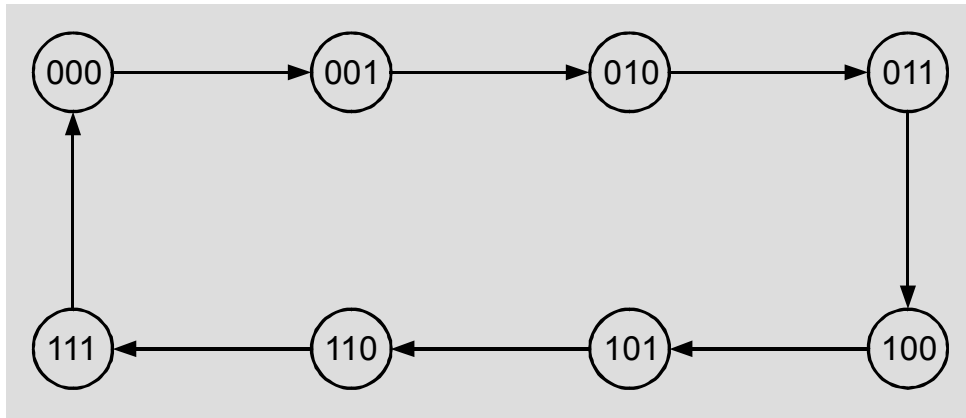


**Steuerwerk für
parallel-seriellen
Multiplikierer (6)**

4.3 Formale Schaltwerkbeschreibung

GIV II 3-56

Modulo-8-Zählwerk aus Toggle-FlipFlops (1)



4.3 Formale Schaltwerkbeschreibung

GIV II 3-57

Modulo-8-Zählwerk aus Toggle-FlipFlops (1)

u_2^n	u_1^n	u_0^n		u_2^{n+1}	u_1^{n+1}	u_0^{n+1}	v_2^n	v_1^n	v_0^n
0	0	0		0	0	1	0	0	1
0	0	1		0	1	0	0	1	1
0	1	0		0	1	1	0	0	1
0	1	1		1	0	0	1	1	1
1	0	0		1	0	1	0	0	1
1	0	1		1	1	0	0	1	1
1	1	0		1	1	1	0	0	1
1	1	1		0	0	0	1	1	1

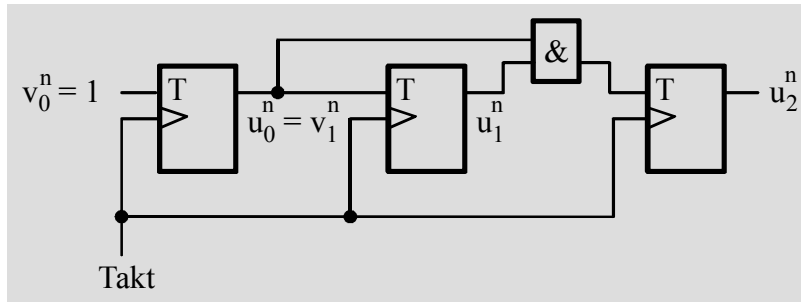
$$u_i^{n+1} = v_i^n \bar{u}_i^n \vee \bar{v}_i^n u_i^n \quad v_2^n = u_1^n \wedge u_0^n \quad v_1^n = u_0^n \quad v_0^n = 1$$

4.3 Formale Schaltwerkbeschreibung

GIV II 3-58

Modulo-8-Zählwerk aus Toggle-FlipFlops (2)

$$v_2^n = u_1^n \wedge u_0^n \quad v_1^n = u_0^n \quad v_0^n = 1$$



4.3 Formale Schaltwerkbeschreibung

GIV II 3-59

4. Schaltwerke

- 4.1 Begriff und Funktion des Schaltwerks
- 4.2 Elementare Schaltwerke
- 4.3 Formale Schaltwerkbeschreibung
- 4.4 Schaltwerkentwurf
- 4.5 Verbindungen zur Automatentheorie

3. Arithmetisch-Logische Einheiten

GIV II 3-60

Schritte beim Schaltwerkentwurf

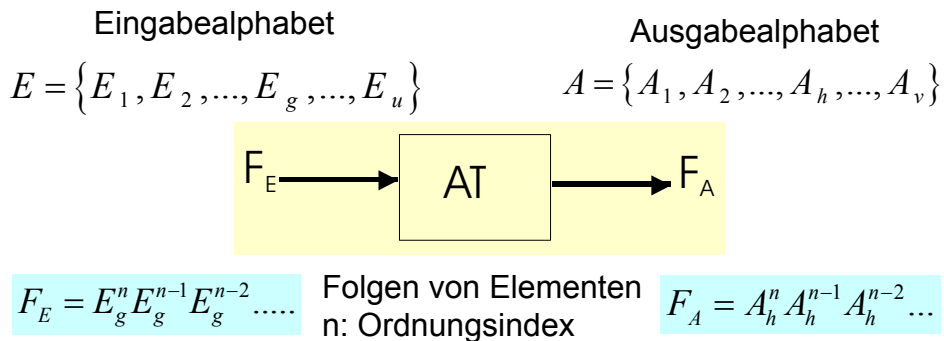
1. Problemanalyse
 - Zustände und Zustandsvariablen definieren
 - Übergänge und logische Übergangsbedingungen ermitteln
 - Ausgangsvariablen und Ausgangswerte ermitteln
 - Übergangstabelle, Zustandsgraph
2. Minimierung der Zustände (Kontrolle auf unwesentliche)
3. FF-Typ-Auswahl (D-FlipFlop nicht immer aufwand-optimal)
4. Schaltung des Schaltwerkes aus der Übergangstabelle

Automaten mit sequentiellen Algorithmen als Vereinfachung übergroßer Schaltnetze (1)

Beispiel: Fahrkartenautomat als kombinatorisches Schaltnetz?

- *gleichzeitige* Eingabe notwendig z.B. von:
Entfernung, Sondertarif, Kind/Erwachsener, Münzgröße/Schein
- oder Tastensatz nötig mit so vielen Tasten wie möglichen
Kombinationen von Eingabevarianten
- vernünftige Alternative:
Automaten mit sequentiellen (Eingabe-) Algorithmen

Automaten mit sequentiellen Algorithmen als Vereinfachung übergroßer Schaltnetze (2)



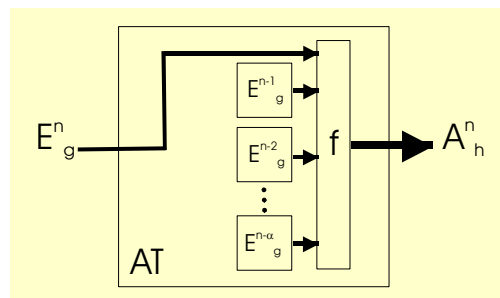
4.5 Verbindungen zur Automatentheorie (nach Lipp)

GIV II 3-63

Automaten mit sequentiellen Algorithmen als Vereinfachung übergroßer Schaltnetze (3)

$$A_h^n = f \{ E_g^n, E_g^{n-1}, E_g^{n-2}, \dots, E_g^{n-\alpha} \}$$

vgl. Schaltnetz: $A_h^n = f(E_g^n)$



α : zeitliche Tiefe der Verarbeitung

Problem: Speicherung von α Elementen bei großem Alphabet E

4.5 Verbindungen zur Automatentheorie (nach Lipp)

GIV II 3-64

Automaten mit sequentiellen Algorithmen als Vereinfachung übergroßer Schaltnetze (4)

Problem: Speicherung von α Elementen bei großem Alphabet E
u.U. viel zu große und redundante Speicherung

Lösung: Abbildung auf Zustandsalphabet $S = \{S_1, \dots, S_k, \dots, S_w\}$

Ausgabefunktion: $A_h^n = f(E_g^n, S_k^n)$

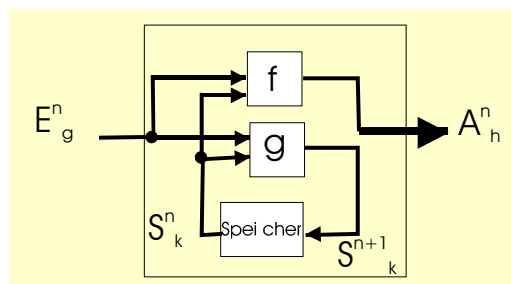
Überföhrungsfunktion: $S_k^{n+1} = g(E_g^n, S_k^n)$

(endlicher) Automat: $AT = (E, A, S, g, f)$

Automaten mit sequentiellen Algorithmen als Vereinfachung übergroßer Schaltnetze (5)

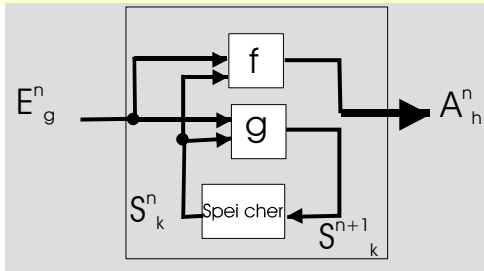
Ausgabefunktion: $A_h^n = f(E_g^n, S_k^n)$

Überföhrungsfunktion: $S_k^{n+1} = g(E_g^n, S_k^n)$

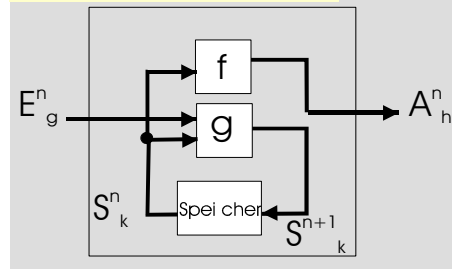


Automatentypen (1)

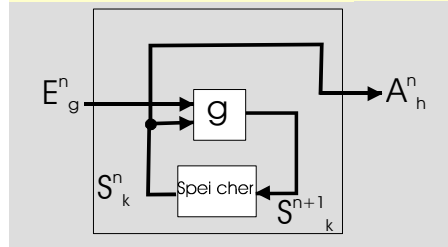
Mealey-Automat: $A_h^n = f(E_g^n, S_k^n)$



Moore-Automat: $A = \lambda(S_k^n)$



Medwedew-Automat: $A_h^n \equiv S_k^n$

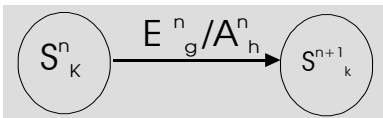


4.5 Verbindungen zur Automatentheorie (nach Lipp)

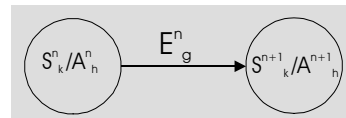
GIV II 3-67

Automatentypen (2)

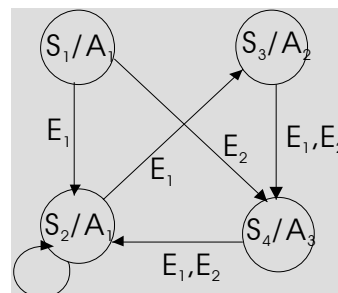
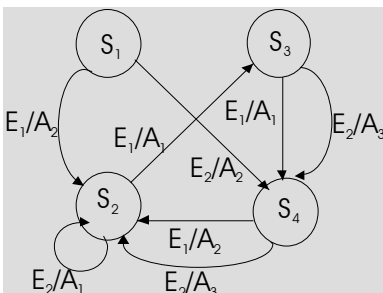
Mealey-Automat: $A_h^n = f(E_g^n, S_k^n)$



Moore-Automat: $A = \lambda(S_k^n)$



Beispiel: $E = \{E_1, E_2\}$, $A = \{A_1, A_2, A_3\}$, $S = \{S_1, S_2, S_3, S_4\}$



4.5 Verbindungen zur Automatentheorie (nach Lipp)

GIV II 3-68

Mealey-Automat: $A_h^n = f(E_g^n, S_k^n)$

S^n	S^{n+1}/A^n		
	E_1^n	...	E_m^n
...	...	...	...
S_k^n	...	S_k^{n+1}/A_h^n	...
...	...	...	...

Moore-Automat: $A = \lambda(S_k^n)$

S^n	S^{n+1}			A^n
	E_1^n	...	E_m^n	
...	...	...	...	...
S_k^n	...	S_k^{n+1}	...	A_h^n
...	...	...	...	...

Beispiel: $E = \{E_1, E_2\}$, $A = \{A_1, A_2, A_3\}$, $S = \{S_1, S_2, S_3, S_4\}$

S^n	S^{n+1}/A^n	
	E_1	E_2
S_1	S_2/A_2	S_4/A_2
S_2	S_3/A_1	S_2/A_1
S_3	S_4/A_1	S_4/A_3
S_4	S_2/A_2	S_2/A_3

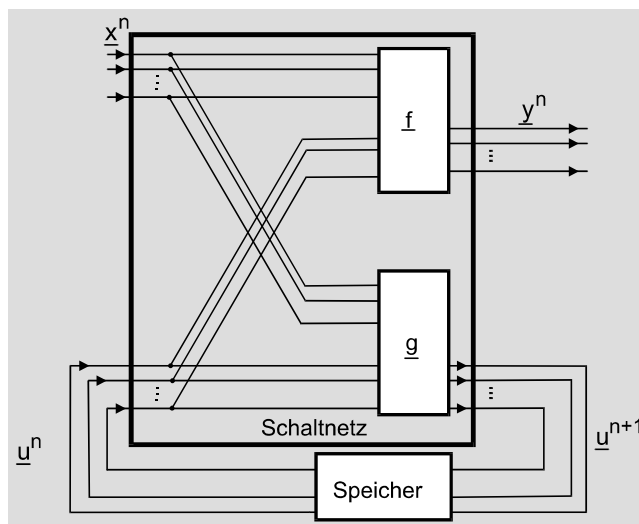
Automatentypen (3)

S^n	S^{n+1}		A^n
	E_1	E_2	
S_1	S_2	S_4	A_1
S_2	S_3	S_2	A_1
S_3	S_4	S_4	A_2
S_4	S_2	S_2	A_3

4.5 Verbindungen zur Automatentheorie (nach Lipp)

GIV II 3-69

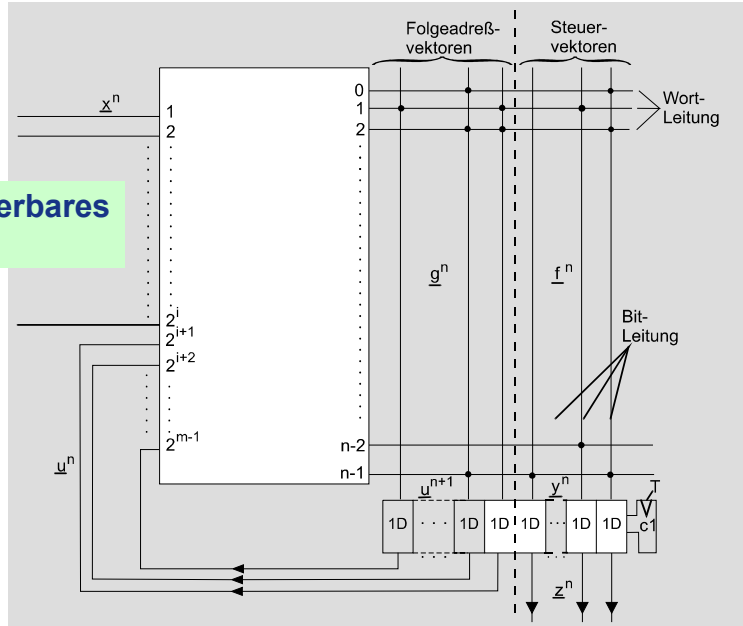
Synchrones Schaltwerk mit fester Logik



4.6 Prinzip der Mikroprogrammierung

GIV II 3-70

Mikroprogrammierbares Steuerwerk (1)



4.6 Prinzip der Mikroprogrammierung

GIV II 3-71

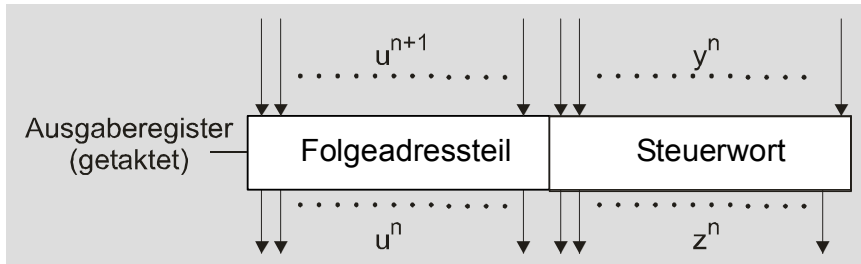
Mikroprogrammierbares Steuerwerk (2)

- Eingangs- und Zustandsvariablen (x^n , u^n) bilden Adresse und aktivieren Speicherwort
 - Speicherwort besteht aus Folgezustandsvektor und Ausgangsvektor
 - Speichermatrix enthält somit 2 Bereiche: zur Programmierung der Übergangs- und der Ausgangsfunktion
- Programmierung heißt Einschreiben der Wertetabelle

4.6 Prinzip der Mikroprogrammierung

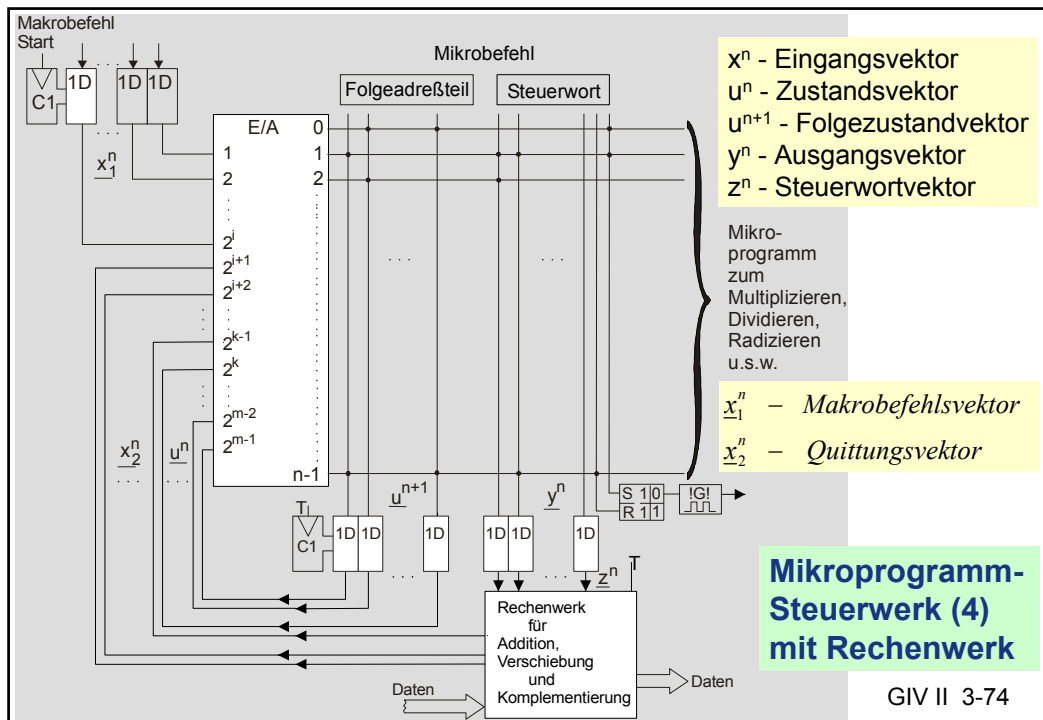
GIV II 3-72

Mikroprogrammierbares Steuerwerk (3): Mikrobefehl



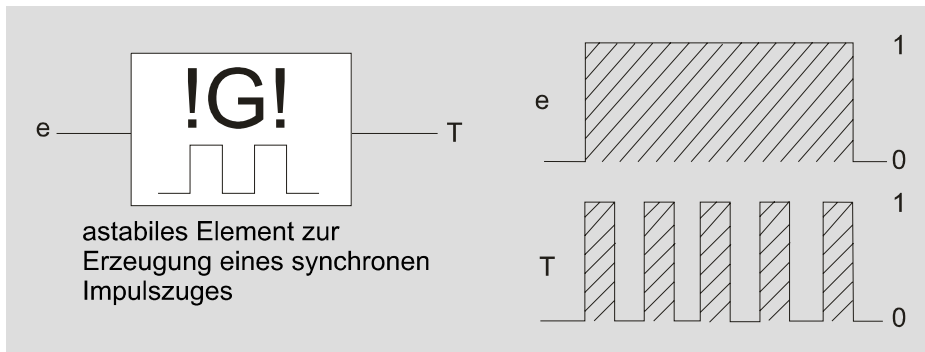
4.6 Prinzip der Mikroprogrammierung

GIV II 3-73



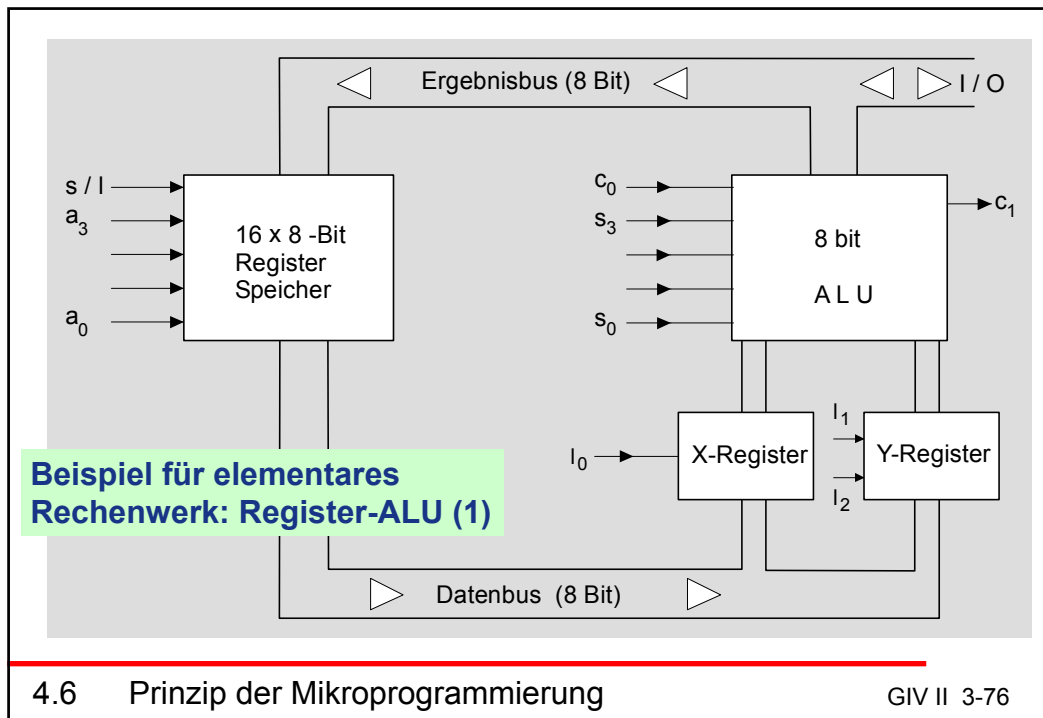
GIV II 3-74

Mikroprogramm-Steuerwerk mit Rechenwerk (5)



4.6 Prinzip der Mikroprogrammierung

GIV II 3-75



4.6 Prinzip der Mikroprogrammierung

GIV II 3-76

Beispiel für elementares Rechenwerk: Register-ALU (2)

a_0, a_1, a_2, a_3 : Registeradressen zur Adressierung der Operanden x,y
 s/l : Schreib-Lese-Steuerung für Registerspeicher
 s_0, s_1, s_2, s_3 : Steuersignale für Auswahl der ALU-Operation
 c_0 : Übertragseingang für ALU
 c_1 : Übertragsausgang für ALU
 l_0 : Übernahmesignal für X-Register
 l_1, l_2 : Übernahmesignale für Y-Register

4.6 Prinzip der Mikroprogrammierung

GIV II 3-77

Beispiel für elementares Rechenwerk: Register-ALU (3)

Steuerparam.			$s_3 = 1$	$s_3 = 0$	
			logische Operationen	Arithmetrische Operationen	
s_2	s_1	s_0		$c_0 = 0$	$c_0 = 1$
0	0	0	$\underline{z} = \underline{x} \cdot \underline{y}$	$[\underline{z}] = 1$	$[\underline{z}] = 0$
0	0	1	$\underline{z} = \underline{x} \oplus \underline{y}$	$[\underline{z}] = [\underline{y}] - [\underline{x}] - 1$	$[\underline{z}] = [\underline{y}] - [\underline{x}]$
0	1	0	$\underline{z} = \underline{x} \equiv \underline{y}$	$[\underline{z}] = [\underline{x}] - [\underline{y}] - 1$	$[\underline{z}] = [\underline{x}] - [\underline{y}]$
0	1	1	$\underline{z} = \overline{\underline{x}} \cdot \underline{y}$	$[\underline{z}] = [\underline{x}] + \underline{y}$	$[\underline{z}] = [\underline{x}] + [\underline{y}] + 1$
1	0	0	$\underline{z} = \underline{x} \cdot \overline{\underline{y}}$	$[\underline{z}] = [\underline{y}]$	$[\underline{z}] = [\underline{y}] + 1$
1	0	1	$\underline{z} = \underline{x} + \underline{y}$	$[\underline{z}] = [\overline{\underline{y}}]$	$[\underline{z}] = [\overline{\underline{y}}] + 1$
1	1	0	$\underline{z} = \overline{\underline{x}} + \underline{y}$	$[\underline{z}] = [\underline{x}]$	$[\underline{z}] = [\underline{x}] + 1$
1	1	1	$\underline{z} = \underline{x} + \overline{\underline{y}}$	$[\underline{z}] = [\overline{\underline{x}}]$	$[\underline{z}] = [\overline{\underline{x}}] + 1$

4.6 Prinzip der Mikroprogrammierung

GIV II 3-78

Beispiel für elementares Rechenwerk: Register-ALU (4)

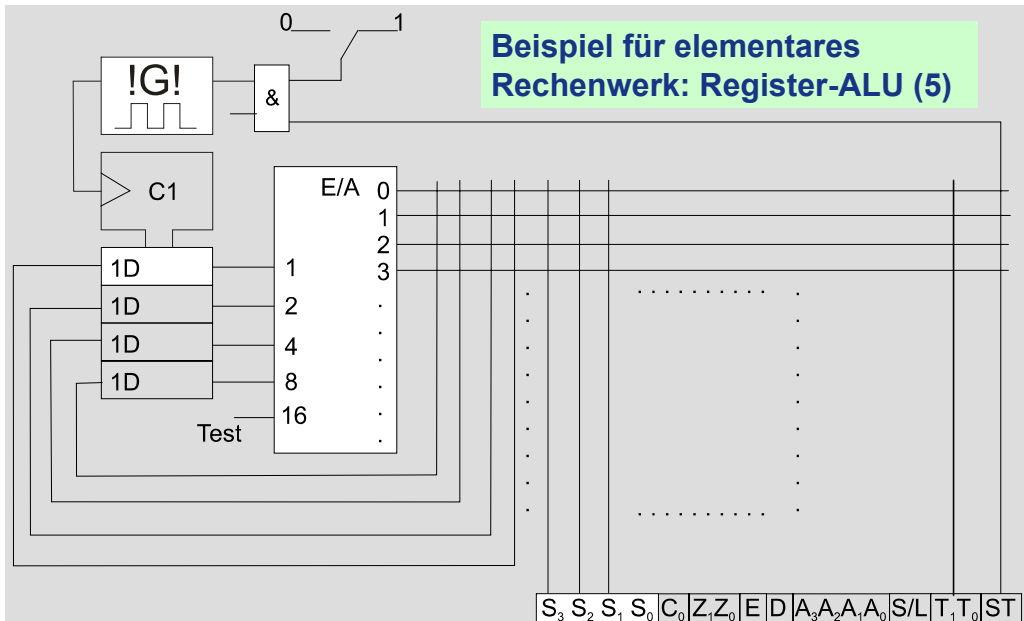
I_1	I_2	I_3	Funktion	s/l	Funktion
0	0	/	Laden der y- Register	0	Auslesen von Daten
0	1	/	Rechts-Shift um 1 Stelle		
1	0	/	Links-Shift um 1 Stelle		
1	1	/	Halten der Information	1	Einschreiben von Daten
/	/	0	Halten der Information		
/	/	1	Übernahme der Informa-tion in x-Register		

s_3	s_2	s_1	s_0	c_0	l_2	l_1	l_0	a_3	a_2	a_1	a_0	s/l
-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

4.6 Prinzip der Mikroprogrammierung

GIV II 3-79

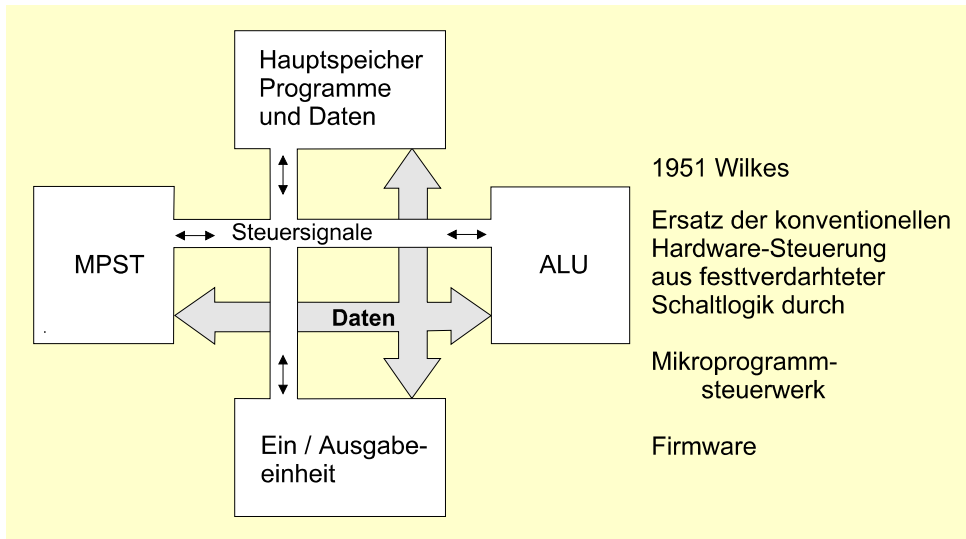
Beispiel für elementares Rechenwerk: Register-ALU (5)



4.6 Prinzip der Mikroprogrammierung

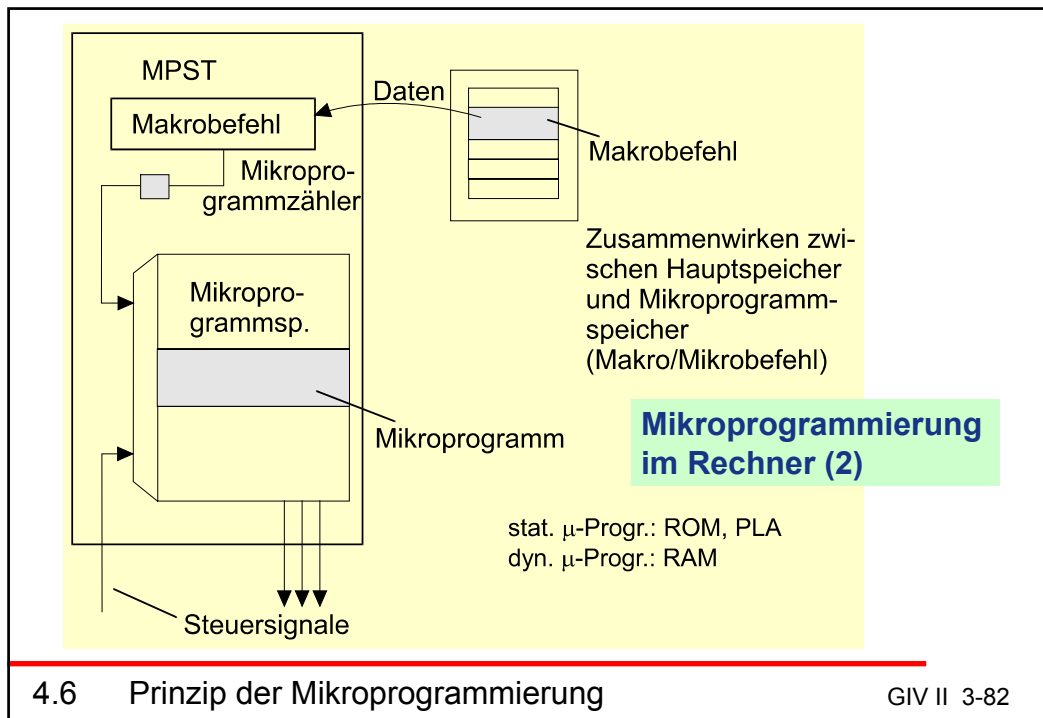
GIV II 3-80

Mikroprogrammierung im Rechner (1)



4.6 Prinzip der Mikroprogrammierung

GIV II 3-81



4.6 Prinzip der Mikroprogrammierung

GIV II 3-82

5. Speicherstrukturen und logische Felder

5.1 Halbleiterspeicher

- 5.2 Logische Beschreibung von Festwertspeichern
- 5.3 Programmierbare logische Felder
- 5.4 Anwenderprogrammierbare Schaltungen (Übersicht)
- 5.5 Field Programmable Logic Devices (FPLD)
- 5.6 Complex Programmable Logic Devices (CPLD)
- 5.7 Antifuse FPGA, Beispiel Actel-Bausteine
- 5.8 SRAM-basierte Bausteine, Beispiel Xilinx FPGA
- 5.9 Bemerkungen zum Entwurf programmierbarer Bausteine

5. Speicherstrukturen und logische Felder

GIV II 3-1

Begriffe und Abkürzungen

RAM (Random Access Memory): Schreib-Lese-Speicher

SRAM: statisches RAM

DRAM: dynamisches RAM

ROM (Read Only Memory): Festwertspeicher

ROM: maskenprogr. ROM

PROM: kundenprogramm. ROM

EPROM: löschbares PROM (E: erasable)

EEPROM: elektr. löschbares PROM

PLA: programmierbares logisches Feld (UND-, ODER-Feld programmierbar, **P**rogrammable **L**ogic **A**rray)

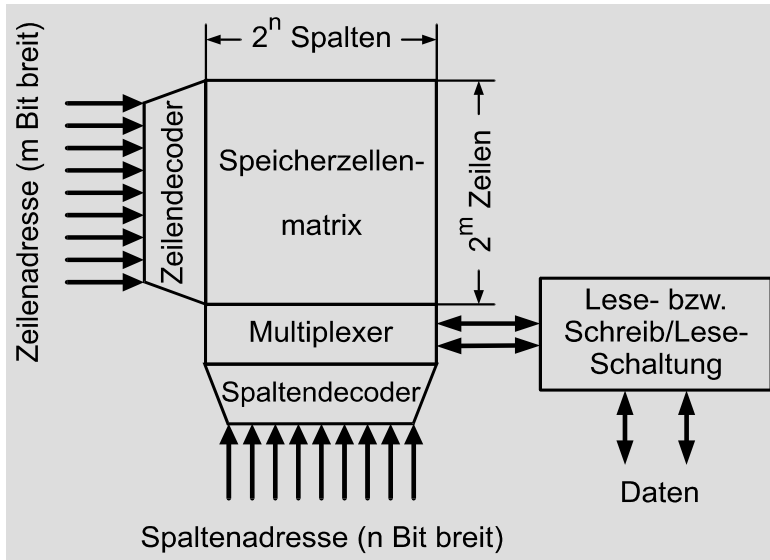
FPLA: kundenprogramm. PLA (F: Field)

PAL: programmierbares logisches Feld (nur UND-Feld programmierbar, **P**rogrammable **A**rray **L**ogic)

5.1 Halbleiterspeicher

GIV II 3-2

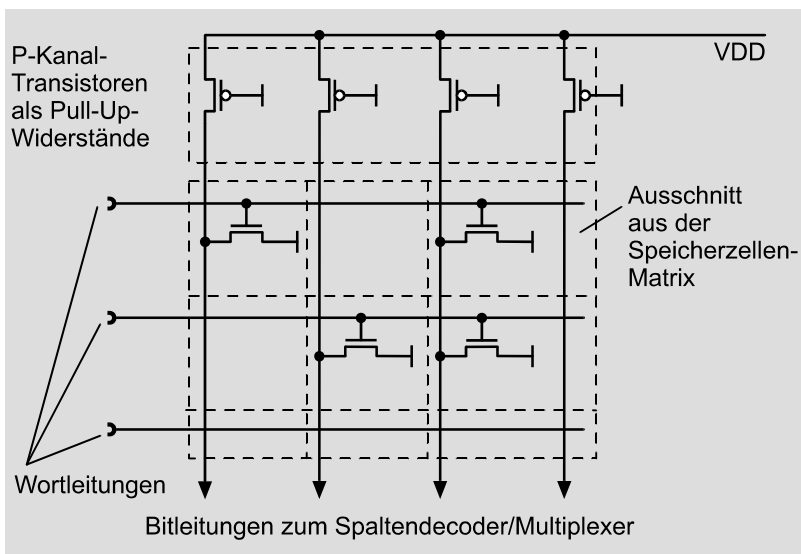
Allgemeine Struktur eines Speicherblockes



5.1 Halbleiterspeicher

GIV II 3-3

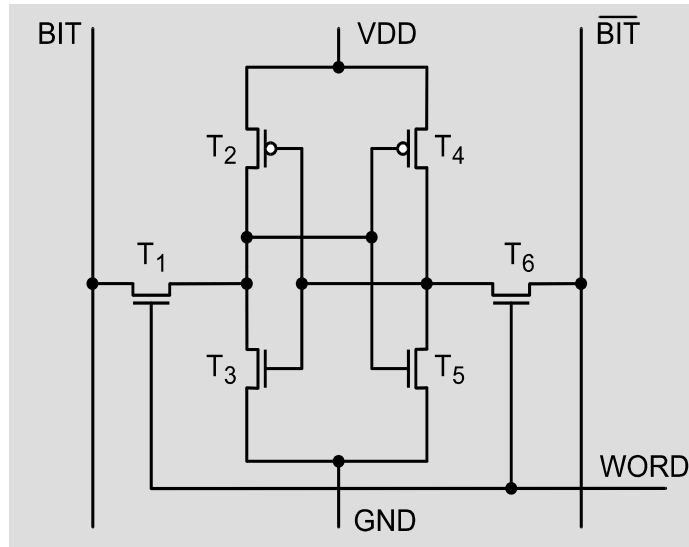
Speicherzellenmatrix eines ROM-Speichers



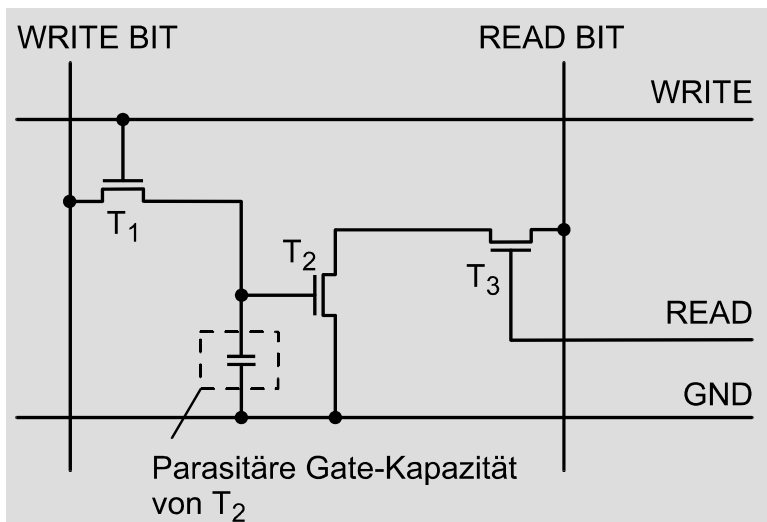
5.1 Halbleiterspeicher

GIV II 3-4

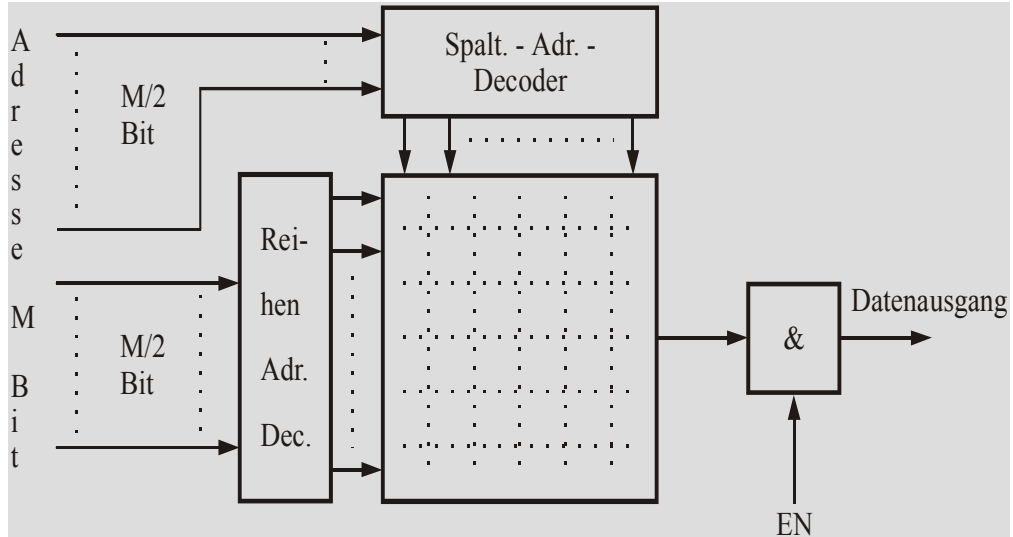
Statische 6-Transistor-RAM-Zelle



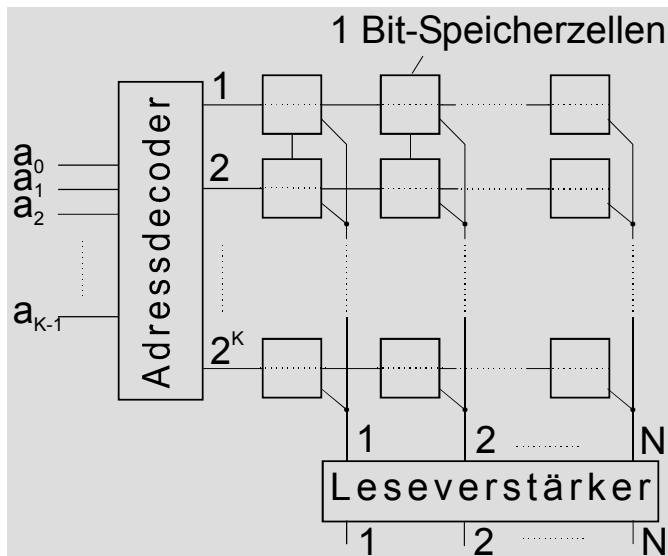
Dynamische 3-Transistor-RAM-Zelle



Koinzidenz-Adressierung

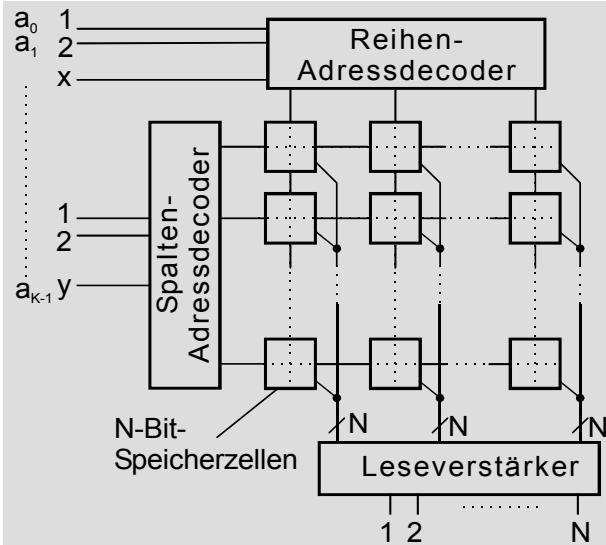


Eindimensionale Adressierung



- K Adreßbit
- 2 Adressleitungen
- 1 Bit-Speicherzellen
- nichtquadratische Speichermatrix (von Wortlänge abhängig)
- 1 Adressdecoder
- für große Datenwortbreiten

Zweidimensionale Adressierung

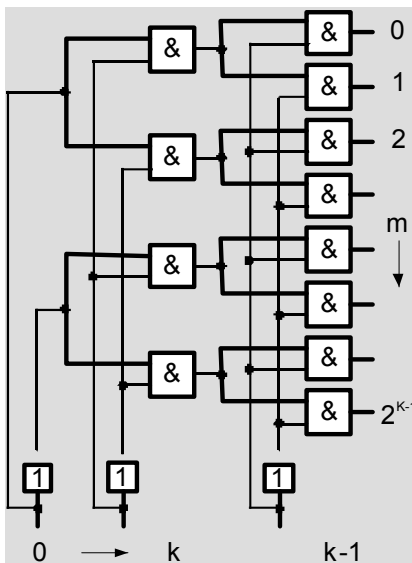


- $K = x + y$ Adressbit
- $2^x + 2^y$ Adressleitungen
- für $x = y = K/2 : 2^{K/2+1}$
- N-Bit-Speicherzellen
- 2 Adressdecoder
- für große Adresswortbreite

5.1 Halbleiterspeicher

GIV II 3-9

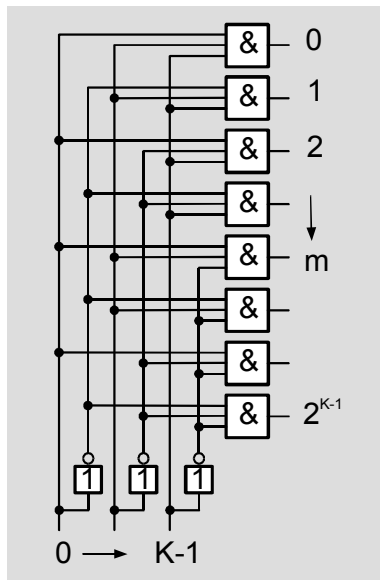
Schalterbaum - Dekodierung



- 1 aus 2^K Dekoder
- hierarchische Adresswegauswahl
- Durchlaufzeit:
Es müssen $K-1$ Gatter durchlauf werden bis zur vollständigen Adress-Dekodierung:
 $t_{co} = (K-1) \cdot t_0$
- ca. 2×2^K Gatter
a' 2 Eingänge

5.1 Halbleiterspeicher

GIV II 3-10

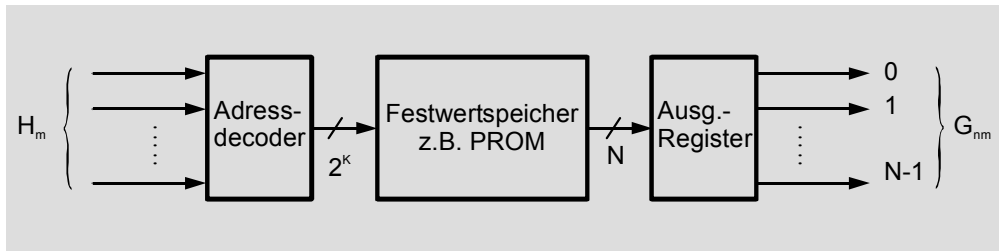


- 1 aus 2^k Dekoder
- 2^k Gatter mit K Eingängen (höherer Verdrahtungsaufwand)
- Durchlaufzeit: 1 Gatterlaufzeit t_0

UND- Feld - Dekodierung

5. Speicherstrukturen und logische Felder

- 5.1 Halbleiterspeicher
- 5.2 Logische Beschreibung von Festwertspeichern
- 5.3 Programmierbare logische Felder
- 5.4 Anwenderprogrammierbare Schaltungen (Übersicht)
- 5.5 Field Programmable Logic Devices (FPLD)
- 5.6 Complex Programmable Logic Devices (CPLD)
- 5.7 Antifuse FPGA, Beispiel Actel-Bausteine
- 5.8 SRAM-basierte Bausteine, Beispiel Xilinx FPGA
- 5.9 Bemerkungen zum Entwurf programmierbarer Bausteine



$$H_0 = \bar{h}_{00} \wedge \bar{h}_{10} \wedge \bar{h}_{20} \dots \bar{h}_{(K-1)0}$$

$$H_1 = h_{01} \wedge \bar{h}_{11} \wedge \bar{h}_{21} \dots \bar{h}_{(K-1)1}$$

$$H_2 = h_{02} \wedge h_{12} \wedge \bar{h}_{22} \dots \bar{h}_{(K-1)2}$$

⋮

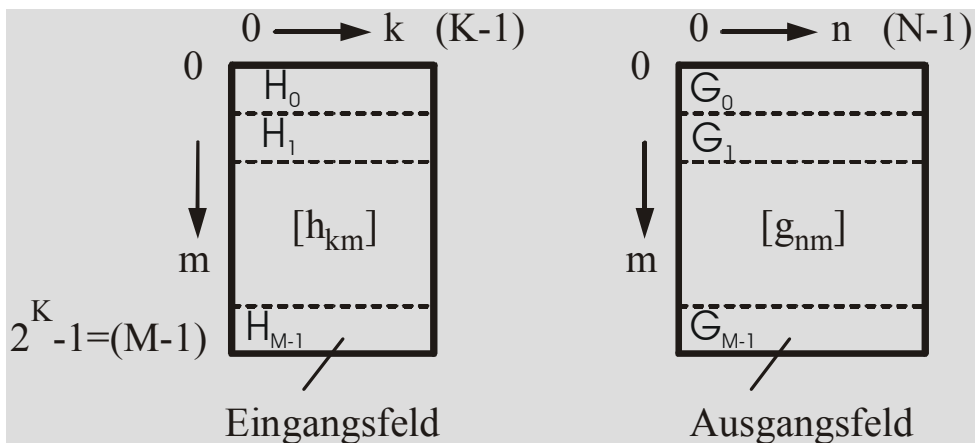
$$H_{M-1} = h_{0(M-1)} \wedge h_{1(M-1)} \wedge h_{2(M-1)} \dots h_{(K-1)(M-1)}$$

Beschreibungsmodell

5.2 Logische Beschreibung von Festwertspeichern

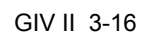
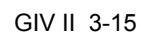
GIV II 3-13

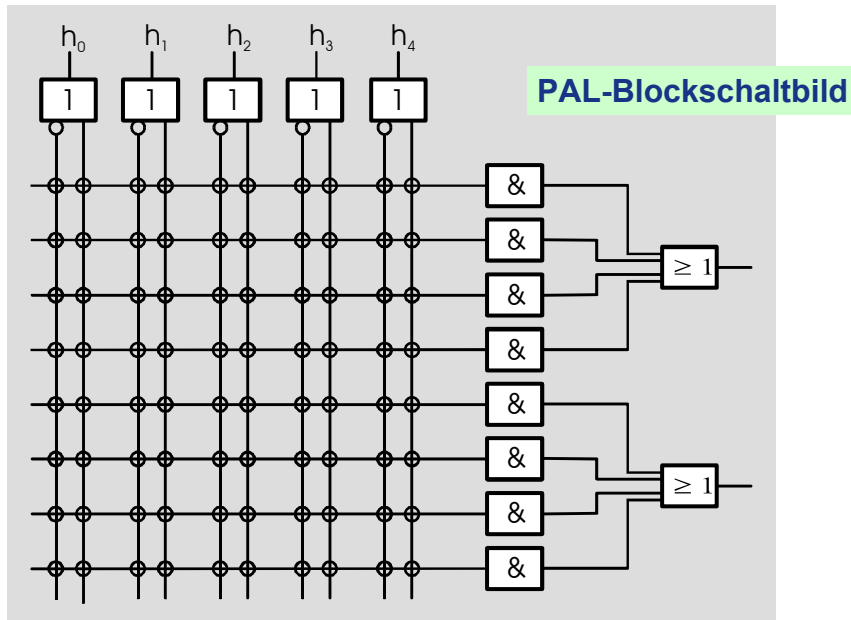
Eingangs- und Ausgangsfeld bei Festwertspeichern



5.2 Logische Beschreibung von Festwertspeichern

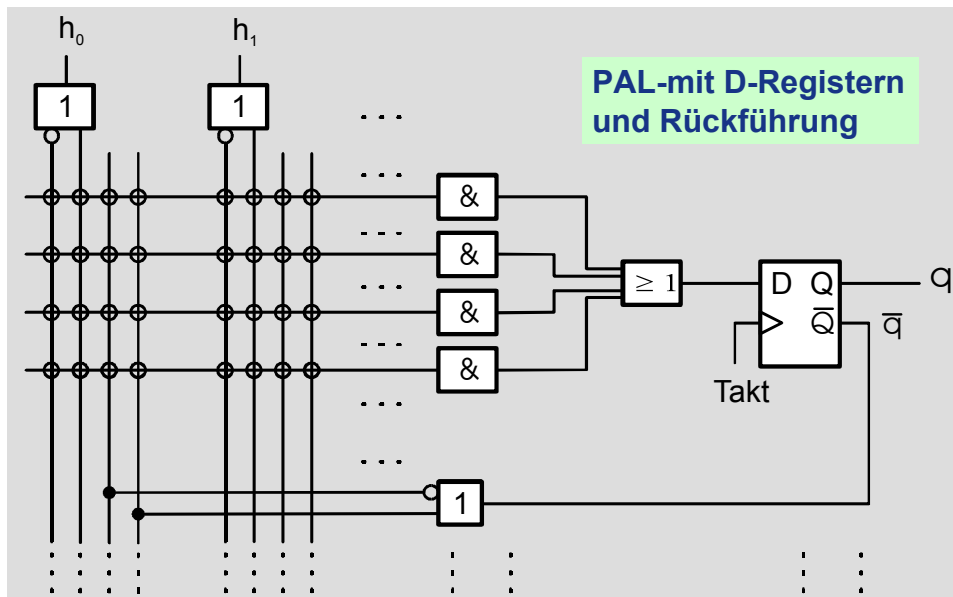
GIV II 3-14





5.3 Programmierbare logische Felder

GIV II 3-17

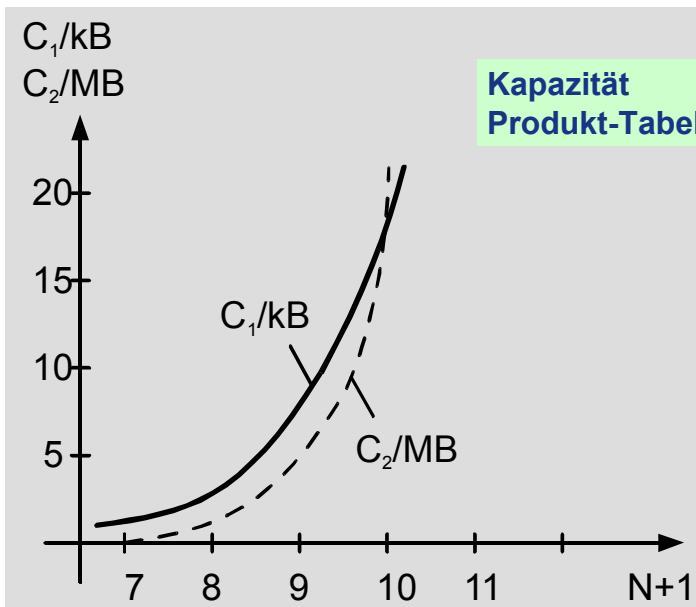


5.3 Programmierbare logische Felder

GIV II 3-18

Produkt im 2er Komplement

$$\begin{aligned}
 P &= A \cdot B = (-b_N \cdot 2^N + \underbrace{\sum_{j=0}^{N-1} b_j \cdot 2^j}_{|B_j|}) \cdot (-a_N \cdot 2^N + \underbrace{\sum_{k=0}^{N-1} a_k \cdot 2^k}_{|A_k|}) \\
 &= -p_{2N} \cdot 2^{2N} + |P_i| \\
 P_i &= |A_k| \cdot |B_j| + a_N \cdot b_N \cdot 2^{2N} - a_N \cdot 2^N \cdot |B_j| \\
 &\quad + p_{2N} \cdot 2^{2N} - b_N \cdot 2^N \cdot |A_k| \\
 p_{2N} &= a_N \neq b_N
 \end{aligned}$$

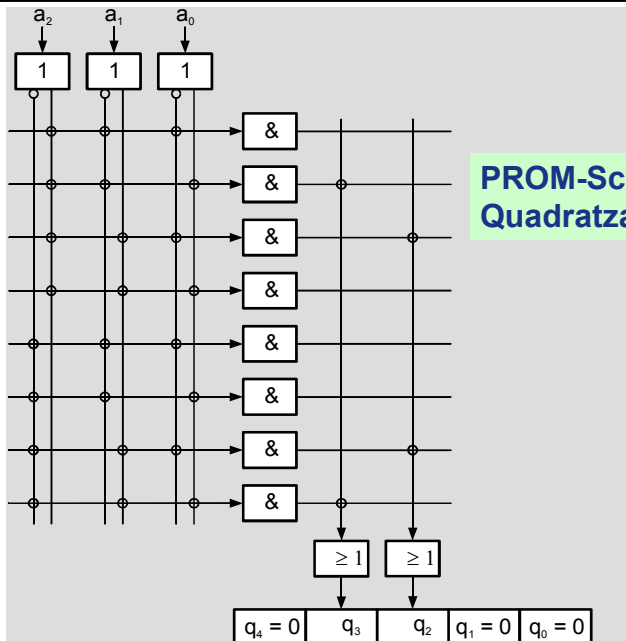


Produkt-Tabelle

A	$Q = A^2$	A (Adresse)			Q (Ergebnis)				
		a_1	a_2	a_3	q_4	q_3	q_2	q_1	q_0
-4	16	1.	0	0					
-3	9	1.	0	1	0.	1	0	0	1
-2	4	1.	1	0	0.	0	1	0	0
-1	1	1.	1	1	0.	0	0	0	1
0	0	0.	0	0	0.	0	0	0	0
1	1	0.	0	1	0.	0	0	0	1
2	4	0.	1	0	0.	0	1	0	0
3	9	0.	1	1	0.	1	0	0	1

5.3 Programmierbare logische Felder

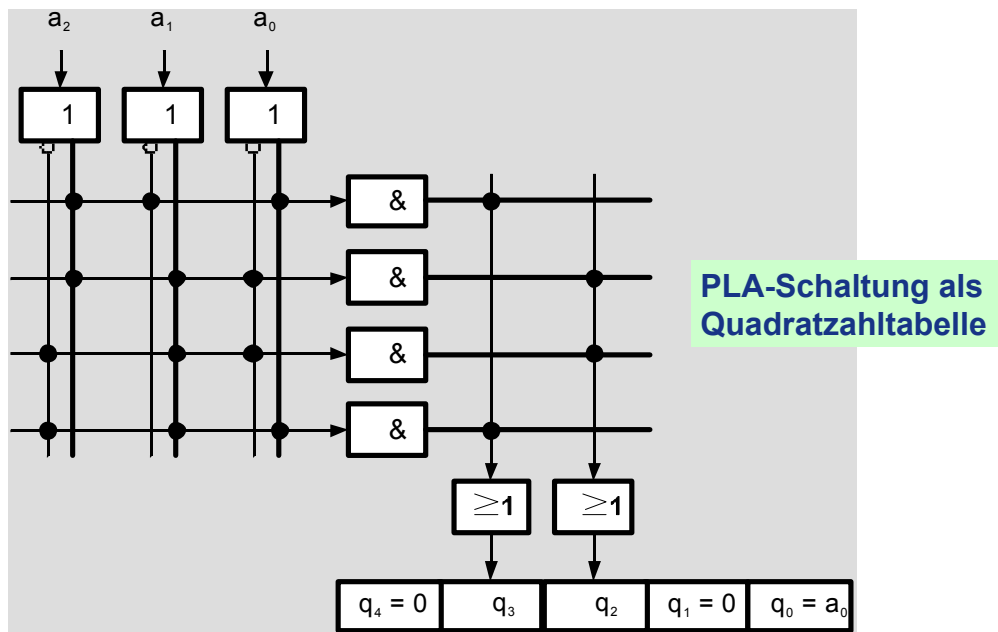
GIV II 3-21



PROM-Schaltung als Quadratzahltabelle

5.3 Programmierbare logische Felder

GIV II 3-22



5.3 Programmierbare logische Felder

GIV II 3-23

5. Speicherstrukturen und logische Felder

- 5.1 Halbleiterspeicher
- 5.2 Logische Beschreibung von Festwertspeichern
- 5.3 Programmierbare logische Felder
- 5.4 Anwenderprogrammierbare Schaltungen (Übersicht)
- 5.5 Field Programmable Logic Devices (FPLD)
- 5.6 Complex Programmable Logic Devices (CPLD)
- 5.7 Antifuse FPGA, Beispiel Actel-Bausteine
- 5.8 SRAM-basierte Bausteine, Beispiel Xilinx FPGA
- 5.9 Bemerkungen zum Entwurf programmierbarer Bausteine

5. Speicherstrukturen und logische Felder

GIV II 3-24

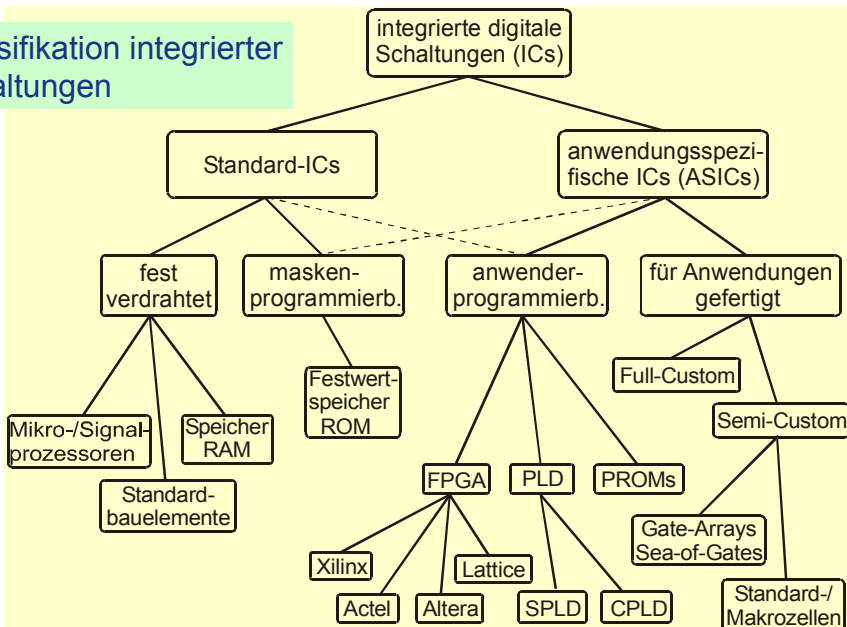
Begriffe und Abkürzungen für anwenderprogrammierbare Schaltungen:

• PLD	Programmable Logic Device
• CPLD	Complex PLD
• SPLD	Simple PLD (PAL, GAL)
• PROM	Programmable Read Only Memory
• EEPROM	Electrically Erasable PROM
• EPROM	Erasable PROM
• FPAD	Field Programmable Analog Device
• FPID	Field Programmable Interconnect Device
• GA	Gate Array
• FPGA	Field Programmable GA
• MPGA	Mask Programmable GA
• UPL	User Programmable Logic

5.4 Anwenderprogrammierbare Schaltungen (Übersicht)

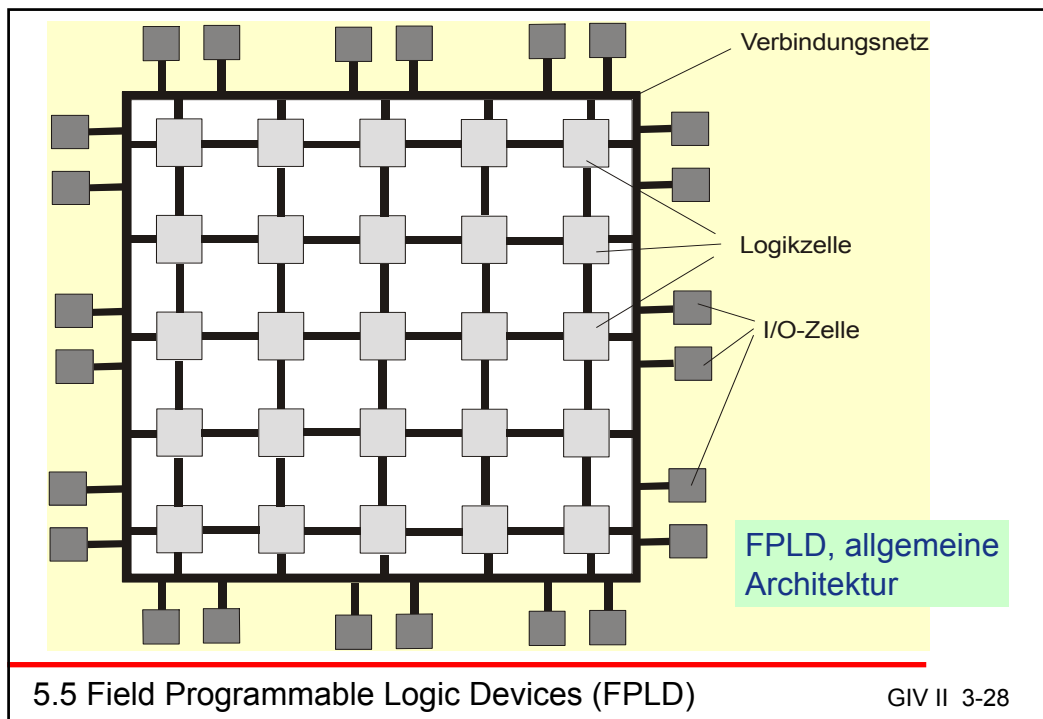
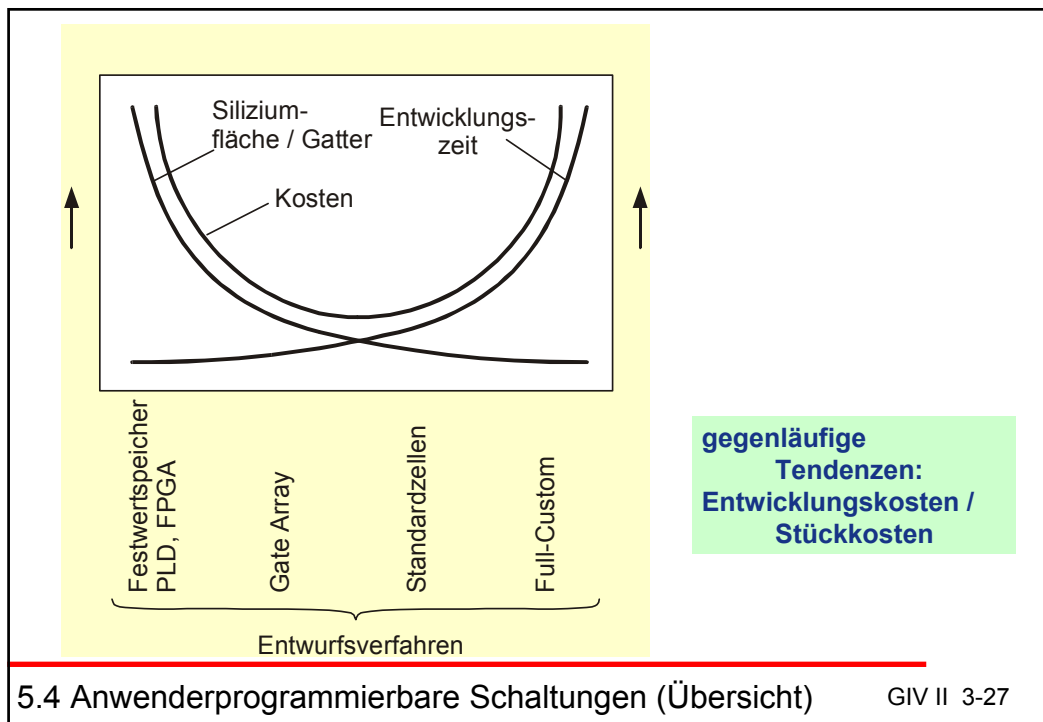
GIV II 3-25

Klassifikation integrierter Schaltungen



5.4 Anwenderprogrammierbare Schaltungen (Übersicht)

GIV II 3-26



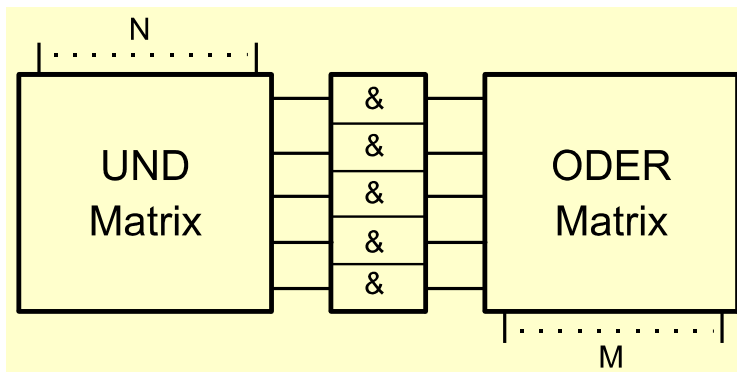
5. Speicherstrukturen und logische Felder

- 5.1 Halbleiterspeicher
- 5.2 Logische Beschreibung von Festwertspeichern
- 5.3 Programmierbare logische Felder
- 5.4 Anwenderprogrammierbare Schaltungen (Übersicht)
- 5.5 Field Programmable Logic Devices (FPLD)
- 5.6 **Complex Programmable Logic Devices (CPLD)**
- 5.7 Antifuse FPGA, Beispiel Actel-Bausteine
- 5.8 SRAM-basierte Bausteine, Beispiel Xilinx FPGA
- 5.9 Bemerkungen zum Entwurf programmierbarer Bausteine

5. Speicherstrukturen und logische Felder

GIV II 3-29

Grundstruktur eines PLDs (PAL, GAL etc.)

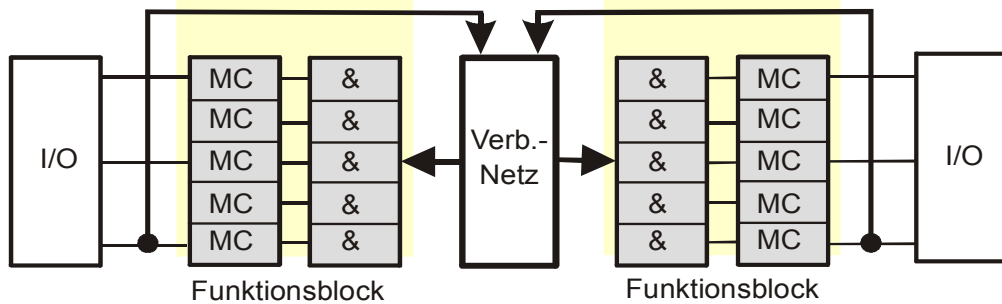


5.6 Complex Programmable Logic Devices (CPLD)

GIV II 3-30

Typische CPLD-Architektur (CPLD, EPLD)

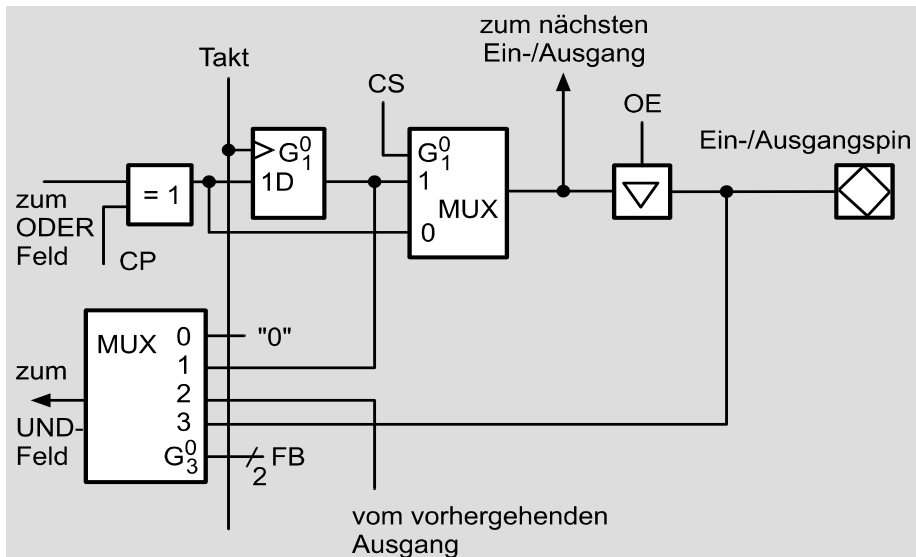
programmierbar: Verbindungsnetz und Makrozellen (MC)



5.6 Complex Programmable Logic Devices (CPLD)

GIV II 3-31

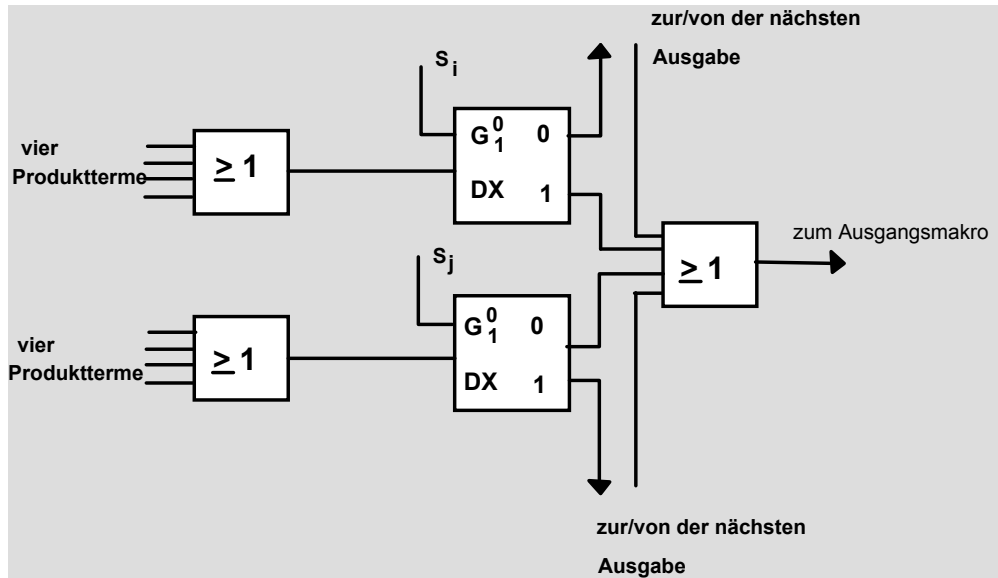
Vereinfachte Makrozelle (Programmierbares Ausgangs-Makro)



5.6 Complex Programmable Logic Devices (CPLD)

GIV II 3-32

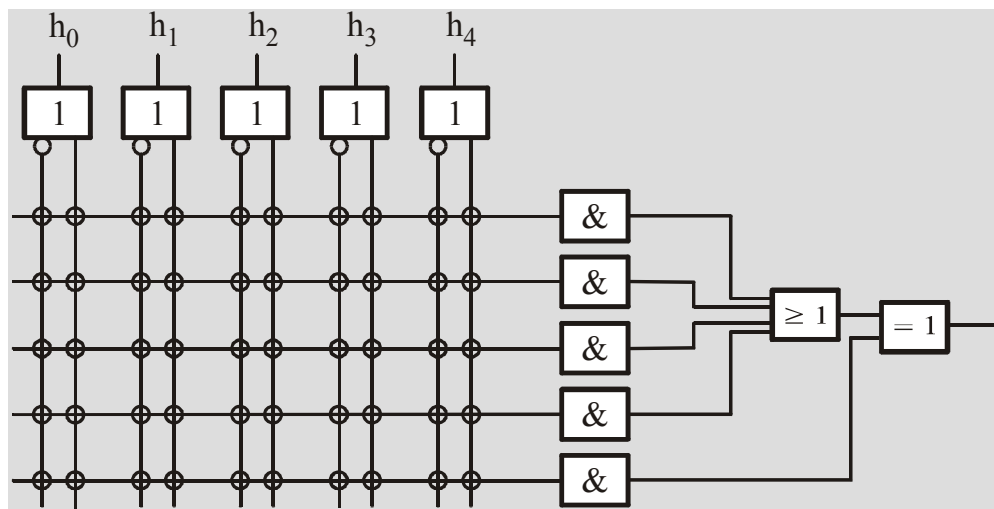
Flexible Produktterm-Allokation



5.6 Complex Programmable Logic Devices (CPLD)

GIV II 3-33

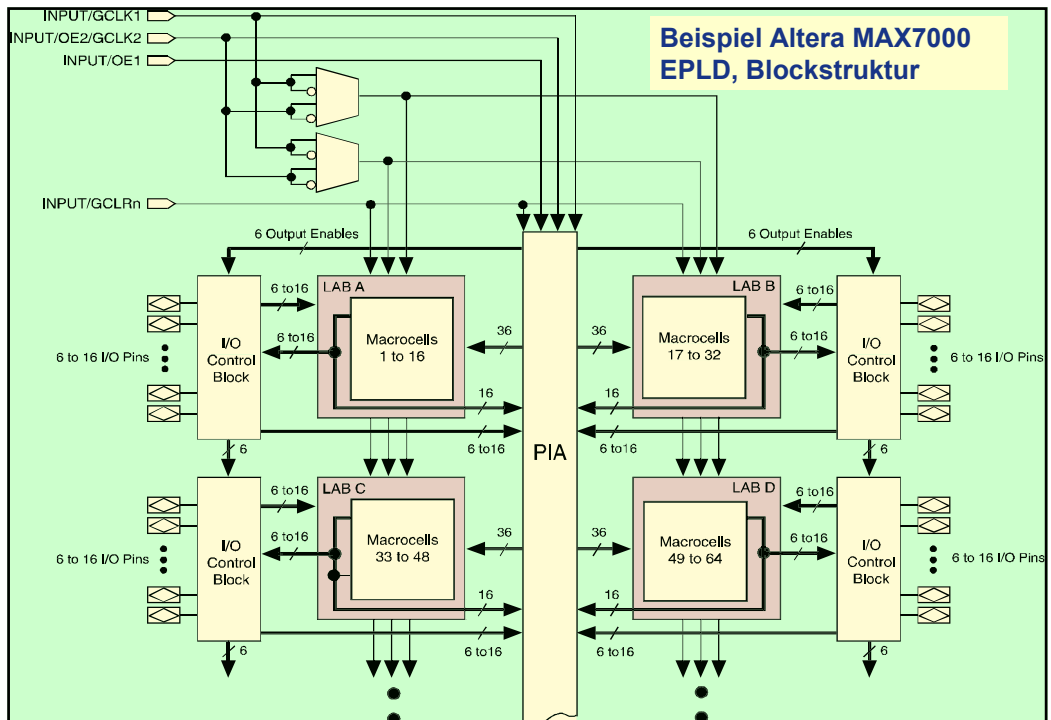
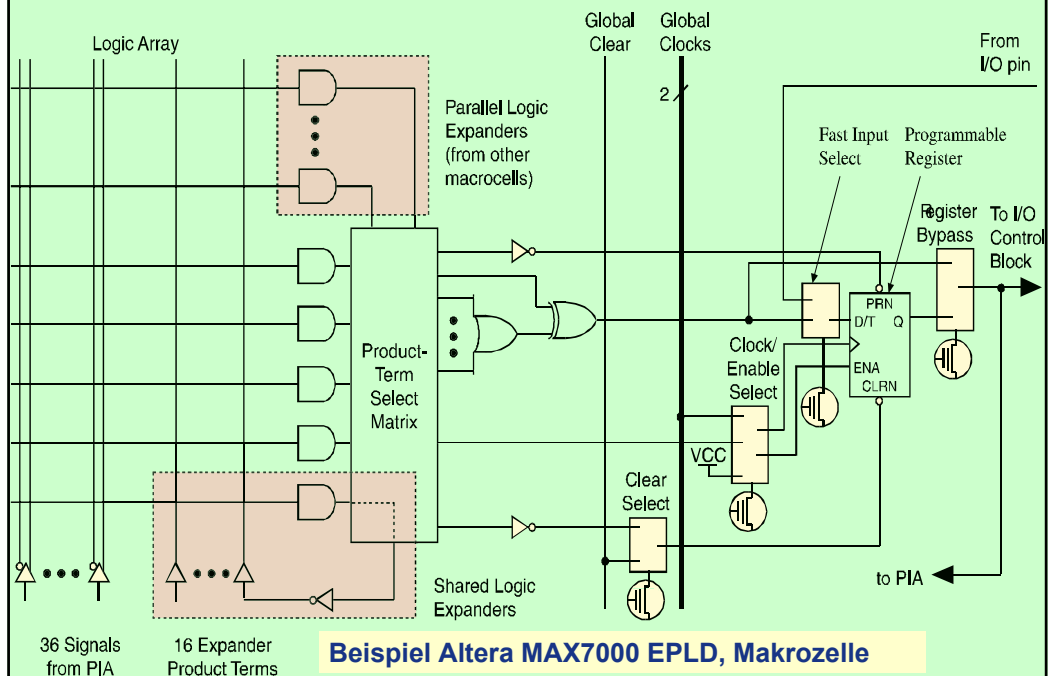
Logik-Block mit programmierbarer Negation

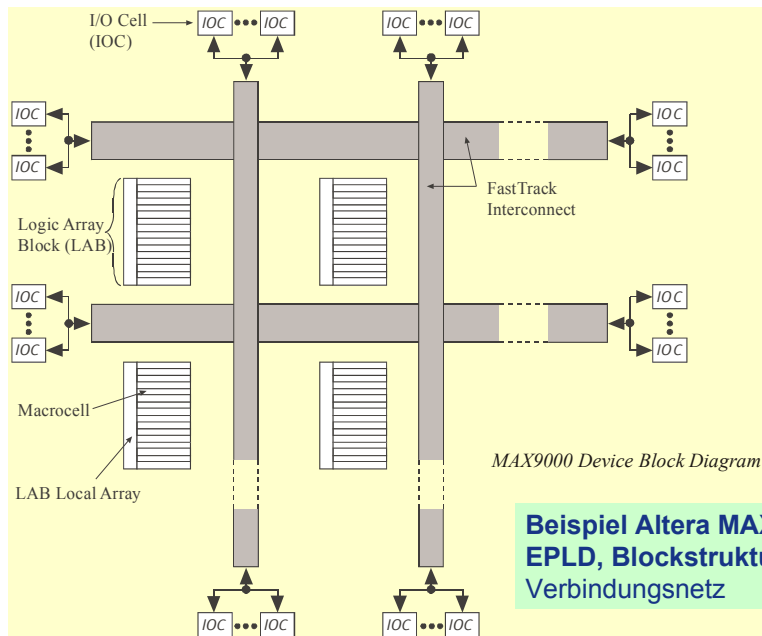


5.6 Complex Programmable Logic Devices (CPLD)

GIV II 3-34

EPM7032, EPM7064 & EPM7096 Device Macrocell





**Beispiel Altera MAX9000
EPLD, Blockstruktur
Verbindungsnetz**

5.6 Complex Programmable Logic Devices (CPLD)

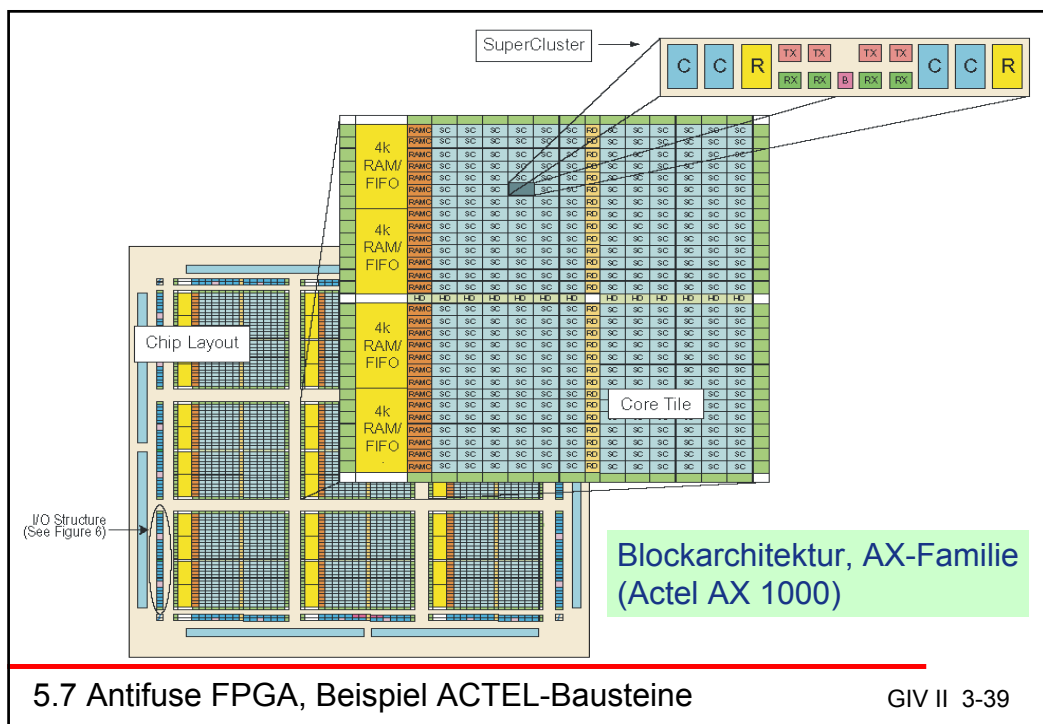
GIV II 3-37

5. Speicherstrukturen und logische Felder

- 5.1 Halbleiterspeicher
- 5.2 Logische Beschreibung von Festwertspeichern
- 5.3 Programmierbare logische Felder
- 5.4 Anwenderprogrammierbare Schaltungen (Übersicht)
- 5.5 Field Programmable Logic Devices (FPLD)
- 5.6 Complex Programmable Logic Devices (CPLD)
- 5.7 Antifuse FPGA, Beispiel Actel-Bausteine
- 5.8 SRAM-basierte Bausteine, Beispiel Xilinx FPGA
- 5.9 Bemerkungen zum Entwurf programmierbarer Bausteine

5. Speicherstrukturen und logische Felder

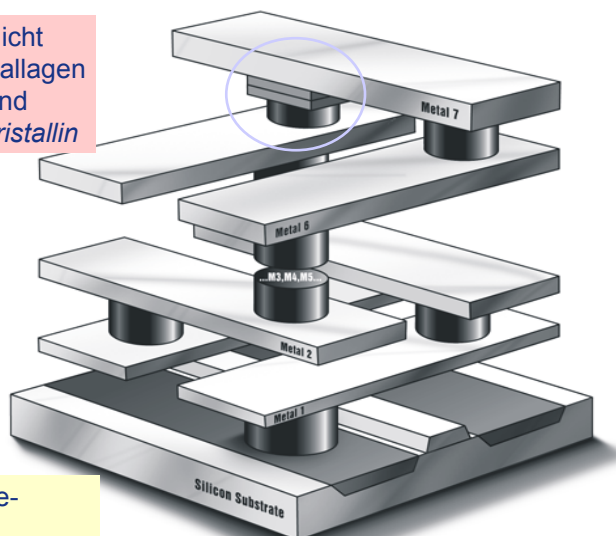
GIV II 3-38



5.7 Antifuse FPGA, Beispiel ACTEL-Bausteine

GIV II 3-39

dünne amorphe Siliziumschicht zwischen den oberen 2 Metallagen wird durch Brennstrom leitend
amorph: nicht kristallin

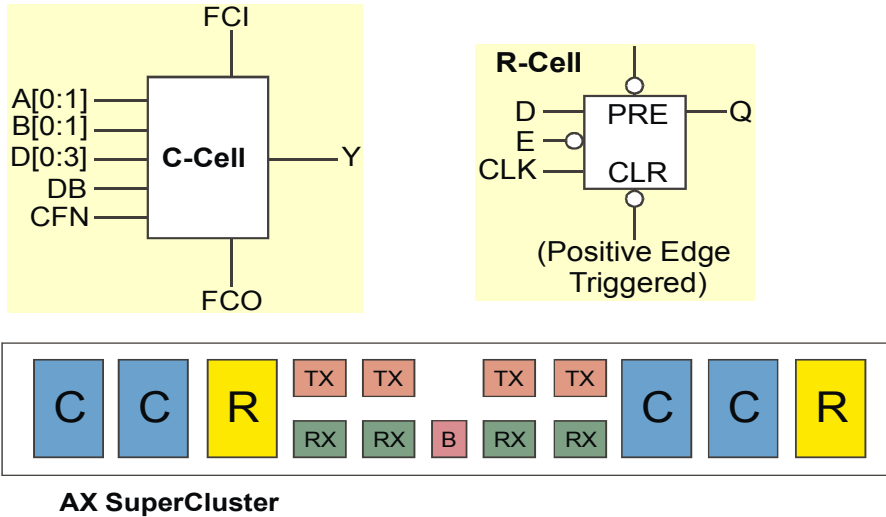


Programmierbare Antifuse-Metall-Metall-Verbindung
AX- (Axcelerator-) Familie (Actel)

5.7 Antifuse FPGA, Beispiel ACTEL-Bausteine

GIV II 3-40

Basiszellen, AX- (Axcelerator-) Familie (Actel)



5.7 Antifuse FPGA, Beispiel ACTEL-Bausteine

GIV II 3-41

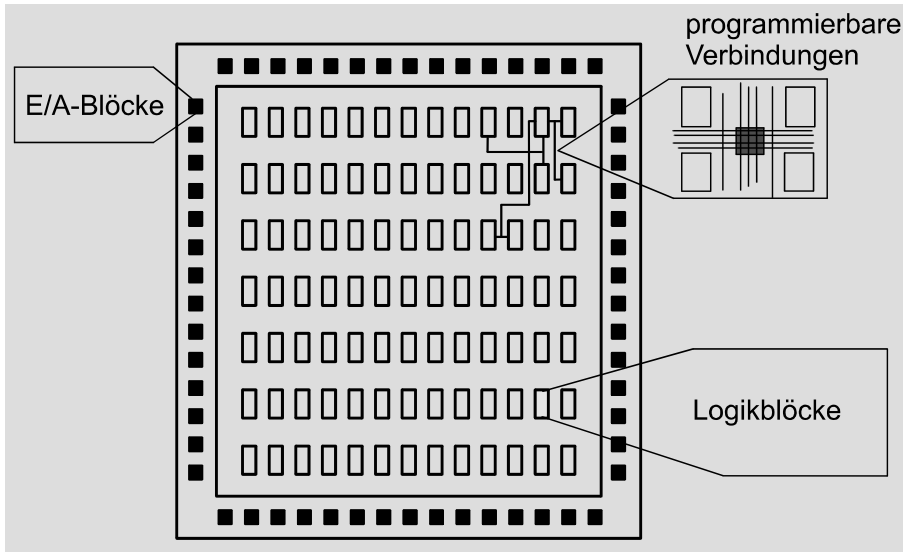
5. Speicherstrukturen und logische Felder

- 5.1 Halbleiterspeicher
- 5.2 Logische Beschreibung von Festwertspeichern
- 5.3 Programmierbare logische Felder
- 5.4 Anwenderprogrammierbare Schaltungen (Übersicht)
- 5.5 Field Programmable Logic Devices (FPLD)
- 5.6 Complex Programmable Logic Devices (CPLD)
- 5.7 Antifuse FPGA, Beispiel Actel-Bausteine
- 5.8 SRAM-basierte Bausteine, Beispiel Xilinx FPGA**
- 5.9 Bemerkungen zum Entwurf programmierbarer Bausteine

5. Speicherstrukturen und logische Felder

GIV II 3-42

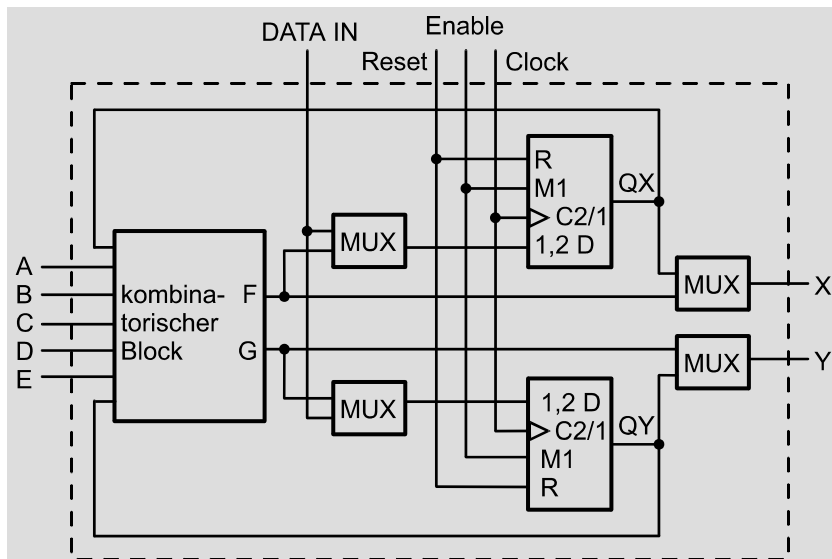
Grundstruktur Xilinx 3000/4000 FPGA



5.8 SRAM-basierte Bausteine, Beispiel Xilinx-FPGA

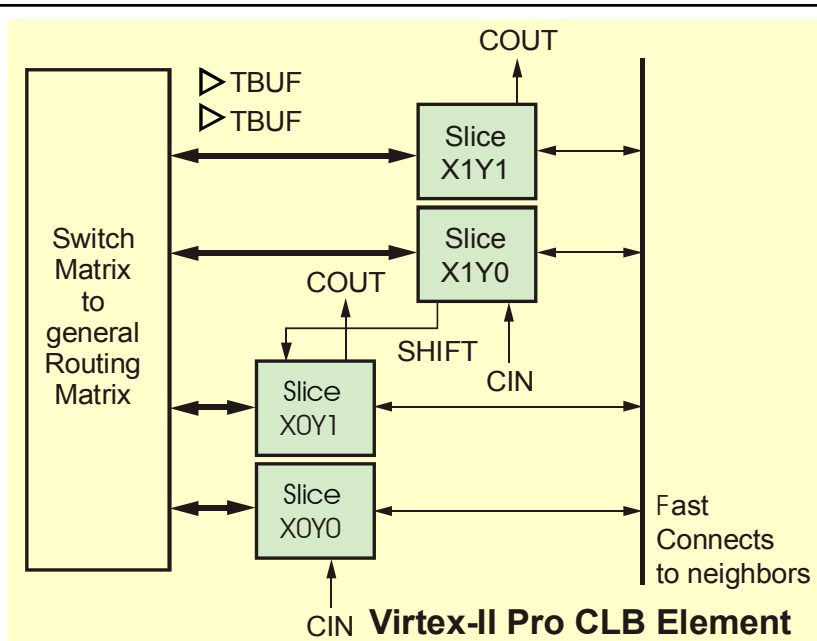
GIV II 3-43

Funktionszelle Xilinx 3000 FPGA



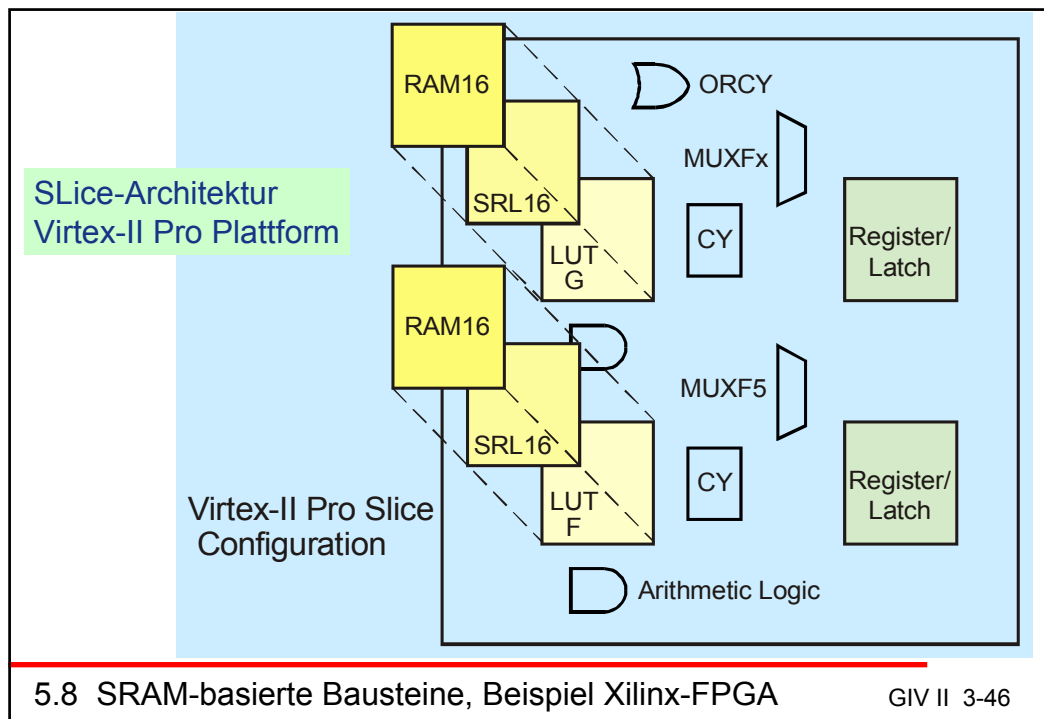
5.8 SRAM-basierte Bausteine, Beispiel Xilinx-FPGA

GIV II 3-44



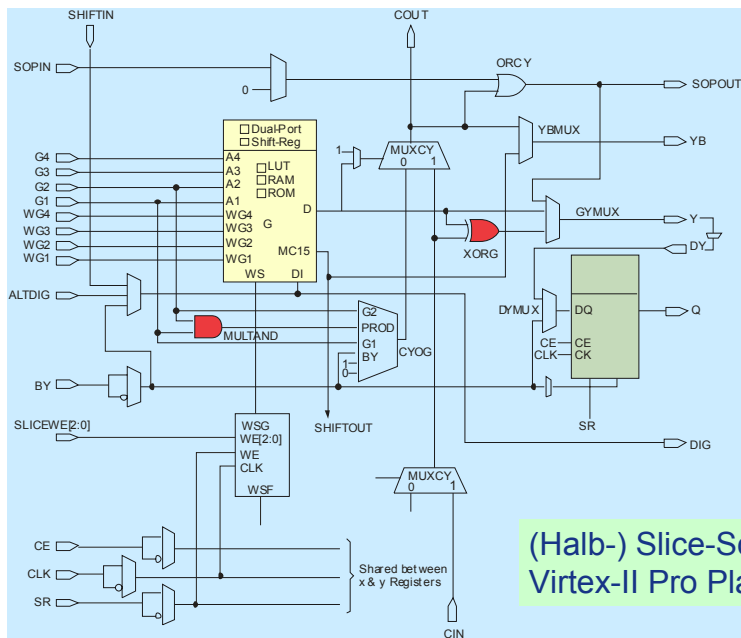
5.8 SRAM-basierte Bausteine, Beispiel Xilinx-FPGA

GIV II 3-45



5.8 SRAM-basierte Bausteine, Beispiel Xilinx-FPGA

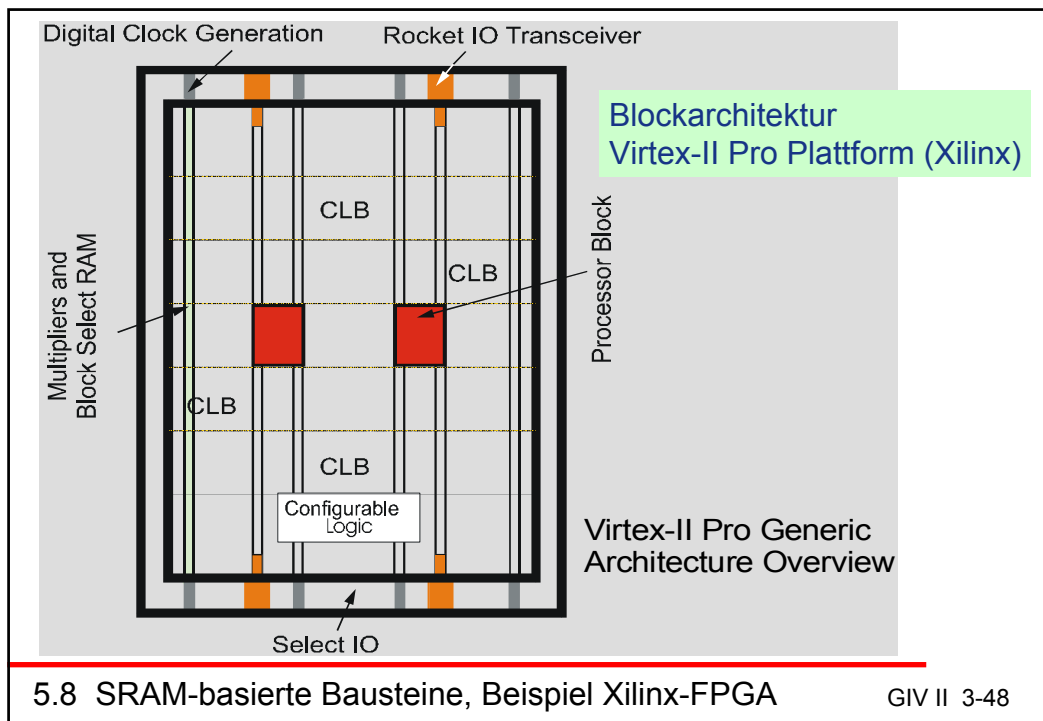
GIV II 3-46



(Halb-) Slice-Schaltung
Virtex-II Pro Plattform (Xilinx)

5.8 SRAM-basierte Bausteine, Beispiel Xilinx-FPGA)

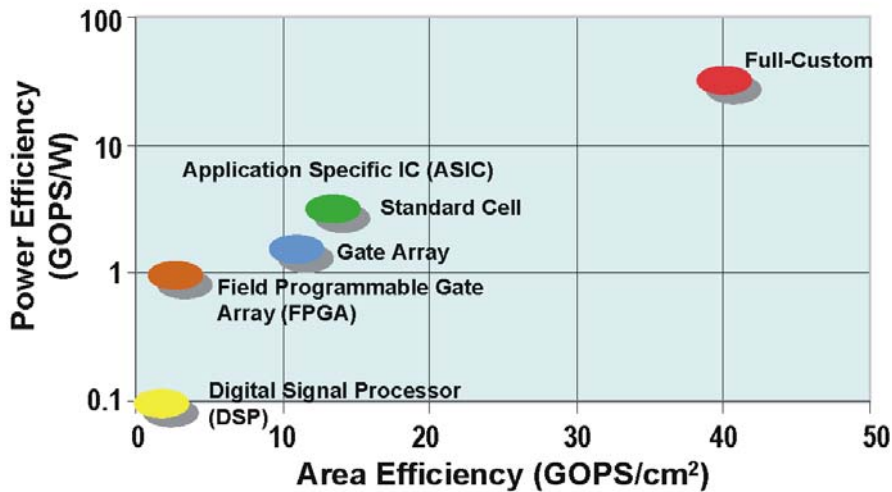
GIV II 3-47



5.8 SRAM-basierte Bausteine, Beispiel Xilinx-FPGA)

GIV II 3-48

DSP auf FPGA, Effizienzmaße (nach Vai)



5.8 SRAM-basierte Bausteine, Beispiel Xilinx-FPGA

GIV II 3-49

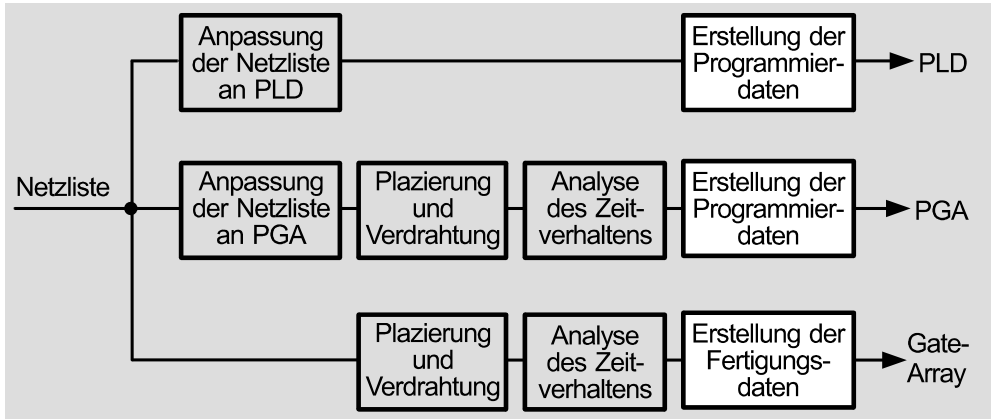
5. Speicherstrukturen und logische Felder

- 5.1 Halbleiterspeicher
- 5.2 Logische Beschreibung von Festwertspeichern
- 5.3 Programmierbare logische Felder
- 5.4 Anwenderprogrammierbare Schaltungen (Übersicht)
- 5.5 Field Programmable Logic Devices (FPLD)
- 5.6 Complex Programmable Logic Devices (CPLD)
- 5.7 Antifuse FPGA, Beispiel Actel-Bausteine
- 5.8 SRAM-basierte Bausteine, Beispiel Xilinx FPGA
- 5.9 Bemerkungen zum Entwurf programmierbarer Bausteine

5. Speicherstrukturen und logische Felder

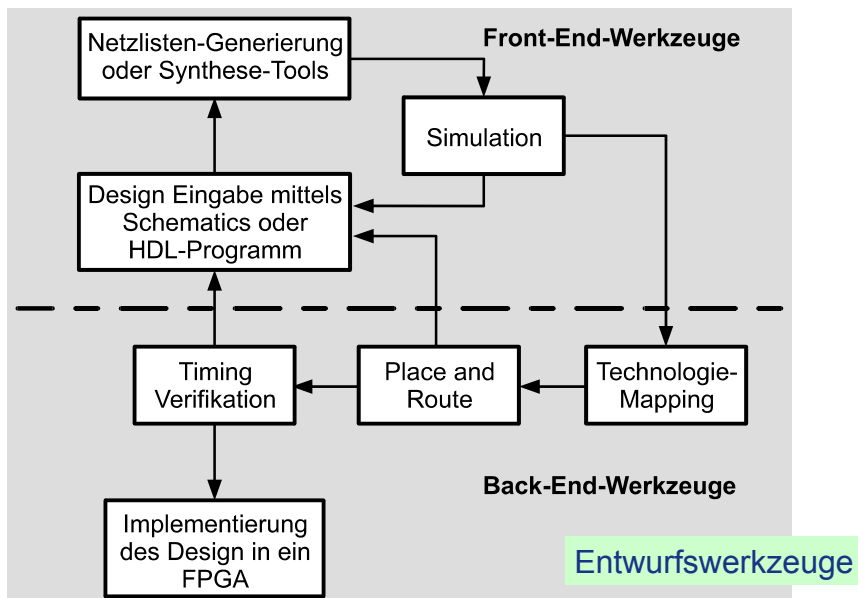
GIV II 3-50

Entwurfsabläufe bei FPLD, FPGA, GA



5.9 Bemerkungen zum Entwurf programmierb. Bausteine

GIV II 3-51



5.9 Bemerkungen zum Entwurf programmierb. Bausteine

GIV II 3-52

