Karlsruhe Institute of Technology

Institut für Technik der Informationsverarbeitung

Dipl.-Ing. J. Heißwolf

heisswolf@kit.edu

Karlsruher Institut für Technologie (**KIT**)

Digitaltechnik

Fragestunde

Entwicklungssatz

- $f\big|_{x_i=1}$ und $f\big|_{x_i=0}$ werden als **Restfunktionen** oder **Kofaktoren** bezeichnet
- der nach x_i entwickelte Ausdruck lässt sich nach den weiteren Variablen entwickeln

Beispiel einer Funktion mit zwei Variablen:

$$y = f(x_2, x_1) = x_1 \,\&\, f(x_2,1) \vee \overline{x_1} \,\&\, f(x_2,0)$$

Entwicklung nach x_1

$$= x_2 \,\&\, [x_1 \,\&\, f(1,1) \vee \overline{x_1} \,\&\, f(1,0)] \vee \overline{x_2} \,\&\, [x_1 \,\&\, f(0,1) \vee \overline{x_1} \,\&\, f(0,0)]$$

Entwicklung nach x_2

$$= \underbrace{x_2 \,\&\, x_1}_{m_3} \,\&\, f(1,1) \vee x_2 \,\&\, \overline{x_1} \,\&\, f(1,0) \vee \overline{x_2} \,\&\, x_1 \,\&\, f(0,1) \vee \overline{x_2} \,\&\, \overline{x_1} \,\&\, f(0,0)]$$

$$= m_3 \,\&\, f_3 \vee m_2 \,\&\, f_2 \vee m_1 \,\&\, f_1 \vee m_0 \,\&\, f_0$$

Minterme

$$= \bigvee_{j=0}^{3} (m_i \,\&\, f_i)$$

DNF

Entwicklungssatz

Klausuraufgabe SS2008

Gegeben ist folgende Schaltfunktion F:

$$F(d,b,c,a) = (\bar{c}b\bar{a}) \vee (cb) \vee (d\bar{c}\bar{b}a) \vee (dc\bar{b}) \vee (c\bar{b}\bar{a})$$

Realisieren Sie diese Schaltfunktion mit Hilfe eines Multiplexer Schaltnetzes.

A) Entwickeln Sie Hierzu die gegebene Funktion F nacheinander nach den Variablen b, c, und d. Geben Sie dazu alle Zwischenschritte an.

Entwicklungssatz

$$F(d,b,c,a) = (\bar{c}b\bar{a}) \vee (cb) \vee (d\bar{c}\bar{b}a) \vee (dc\bar{b}) \vee (c\bar{b}\bar{a})$$

A) Entwickeln Sie Hierzu die gegebene Funktion F nacheinander nach den Variablen b, c, und d. Geben Sie dazu alle Zwischenschritte an.

Entwicklung nach b:

$f(d,c,b,a) = b \cdot f_b(d,c,1,a) + \bar{b} \cdot f_{\bar{b}}(d,c,0,a)$

$= b \cdot (\bar{c}\bar{a} + c) + \bar{b} \cdot (a\bar{c}d + cd + c\bar{a})$

$f_b(d,c,1,a) = (\bar{c}\bar{a} + c); \qquad f_{\bar{b}}(d,c,0,a) = (a\bar{c}d + cd + c\bar{a})$

Entwicklung nach *c*:

$f_b(d,c,b,a) = c(1) + \bar{c}(\bar{a}) \Rightarrow f_{bc}(d,1,1,a) = 1; \qquad f_{b\bar{c}}(d,0,1,a) = \bar{a}$

$f_{\bar{b}}(d,c,b,a) = c(d+\bar{a}) + \bar{c}(ad) \Rightarrow f_{\bar{b}c}(d,1,0,a) = (d+\bar{a}); \qquad f_{\bar{b}\bar{c}}(d,0,0,a) = (ad)$

Entwicklung nach d:

$f_{\bar{b}c}(d,1,0,a) = d(1) + \bar{d}(\bar{a}) \Rightarrow f_{d\bar{b}c}(1,1,0,a) = 1; \qquad f_{\bar{d}\bar{b}c}(0,1,0,a) = \bar{a}$

$f_{\bar{b}\bar{c}}(d,0,0,a) = d(a) + \bar{d}(0) \Rightarrow f_{d\bar{b}\bar{c}}(1,0,0,a) = a; \qquad f_{\bar{d}\bar{b}\bar{c}}(0,0,0,a) = 0$

Entwicklungssatz

$$F(d,b,c,a)=(\bar{c}b\bar{a})\vee(cb)\vee(d\bar{c}\bar{b}a)\vee(dc\bar{b})\vee(c\bar{b}\bar{a})$$

A) Entwickeln Sie Hierzu die gegebene Funktion F nacheinander nach den Variablen b, c, und d. Geben Sie dazu alle Zwischenschritte an.

Alternative Entwicklung:

Entwicklung nach b:

$$\boldsymbol{F(d,c,b,a)=(\bar{c}b\bar{a})\vee(cb)\vee(d\bar{c}\bar{b}a)\vee(dc\bar{b})\vee(c\bar{b}\bar{a})}$$

$$\boldsymbol{F(d,c,b,a)=b[(\bar{c}\bar{a})\vee(c)]\vee\bar{b}[(d\bar{c}a)\vee(dc)\vee(c\bar{a})]}$$

Entwicklung nach *c*:

$$\boldsymbol{F(d,c,b,a)=b[c(1)\vee\bar{c}(\bar{a})]\vee\bar{b}[c(d\vee\bar{a})\vee\bar{c}(da)]}$$

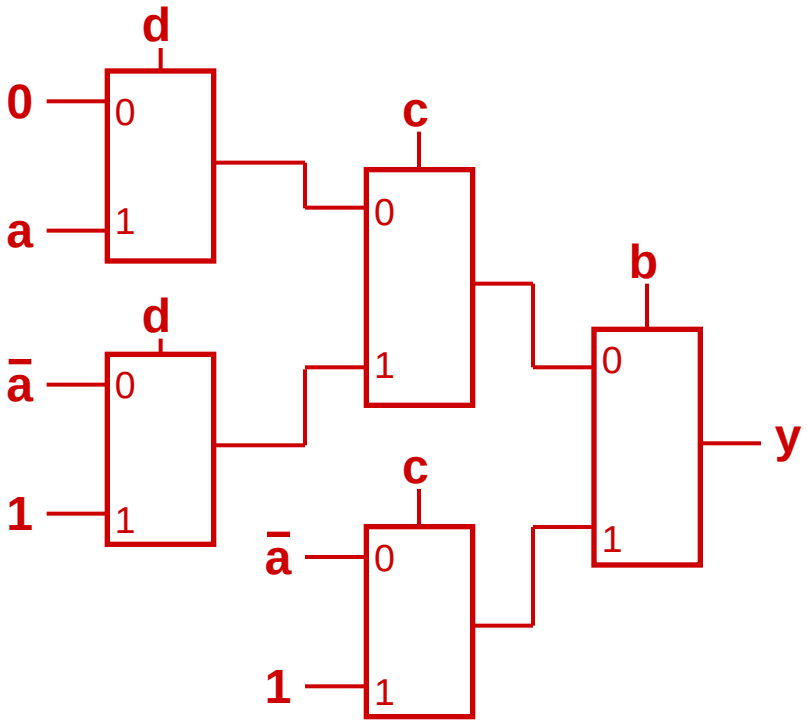
Entwicklung nach d:

$$\boldsymbol{F(d,c,b,a)=b[c(1)\vee\bar{c}(\bar{a})]\vee\bar{b}[c\big(d(1)\vee\bar{d}(\bar{a})\big)\vee\bar{c}\big(d(a)\vee\bar{d}(0)\big)]}$$

Entwicklungssatz

B) Zeichnen Sie das zugehörige Multiplexerschaltnetz in drei Stufen. Setzen sie hierzu zunächst nur 2:1 Multiplexer ein.

$$F(d,c,b,a) = b[c(1) \vee \bar{c}(\bar{a})] \vee \bar{b}[c\big(d(1) \vee \bar{d}(\bar{a})\big) \vee \bar{c}\big(d(a) \vee \bar{d}(0)\big)]$$



Entwicklungssatz

Übertragen der Ergebnisse einer Entwicklung in ein Symmetriediagram, DNF oder KNF

Entwicklung nach b:

$f(d,c,b,a) = b \cdot f_b(d,c,1,a) + \bar{b} \cdot f_{\bar{b}}(d,c,0,a)$

$= b \cdot (\bar{c}\bar{a} + c) + \bar{b} \cdot (a\bar{c}d + cd + c\bar{a})$

$f_b(d,c,1,a) = (\bar{c}\bar{a} + c); \qquad f_{\bar{b}}(d,c,0,a) = (a\bar{c}d + cd + c\bar{a})$

Entwicklung nach *c*:

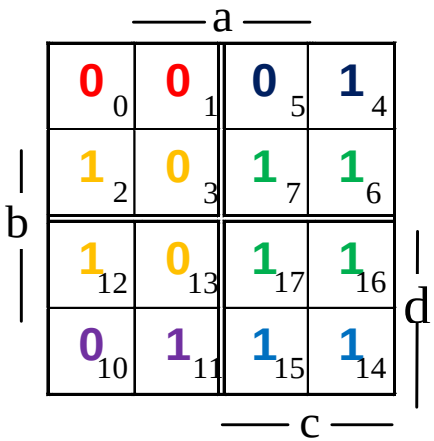
$f_b(d,c,b,a) = c(1) + \bar{c}(\bar{a}) \Rightarrow f_{bc}(d,1,1,a) = 1;$ $\quad f_{b\bar{c}}(d,0,1,a) = \bar{a}$

$f_{\bar{b}}(d,c,b,a) = c(d+\bar{a}) + \bar{c}(ad) \Rightarrow f_{\bar{b}c}(d,1,0,a) = (d+\bar{a});$ $\quad f_{\bar{b}\bar{c}}(d,0,0,a) = (ad)$

Entwicklung nach d:

$f_{\bar{b}c}(d,1,0,a) = d(1) + \bar{d}(\bar{a}) \Rightarrow f_{d\bar{b}c}(1,1,0,a) = 1;$ $\quad f_{\bar{d}\bar{b}c}(0,1,0,a) = \bar{a}$

$f_{\bar{b}\bar{c}}(d,0,0,a) = d(a) + \bar{d}(0) \Rightarrow f_{d\bar{b}\bar{c}}(1,0,0,a) = a;$ $\quad f_{\bar{d}\bar{b}\bar{c}}(0,0,0,a) = 0$



Fließkommazahlen

Wertigkeit der einzelnen stellen der Mantisse:

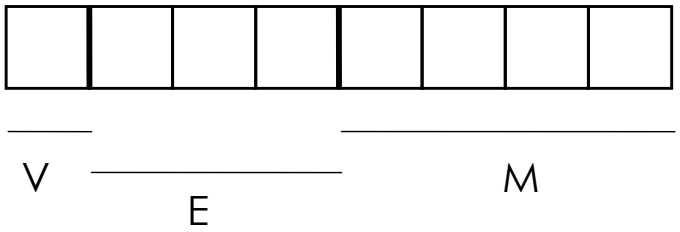
16-Bit-Exponentialzahl = $-1^{V} \cdot 2^{(E - 31)} \cdot (1,M)$

V	E5	E4	E3	E2	E1	E0	M8	M7	M6	M5	M4	M3	M2	M1	M0
							2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}	2^{-8}	2^{-9}
							0,5	0,25	,125	,0625	....				

Fließkommazahlen

Klausuraufgabe SS2009

Zur Verwendung in einem Microcontroller wurde eine platzsparende Darstellung von Fließkommazahlen in einem einzigen Byte entwickelt. Das höchstwertige Bit stellt das Vorzeichen V dar, die vier niederwertigsten Bits die Mantisse M und die drei Bits in der Mitte den Exponenten E (siehe Abbildung).

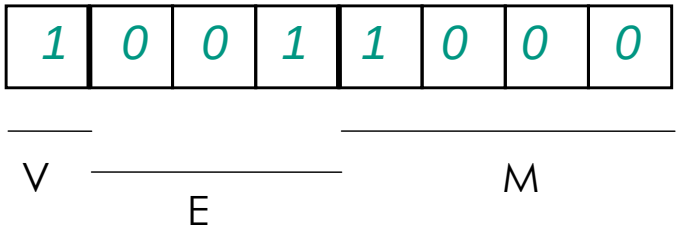


Für alle möglichen binären Belegungen ergibt sich der Dezimalwert Z aus nachstehender Formel (vgl. IEEE-Fließkommazahl):

$$Z_D = (-1)^V \cdot 2^{E-3} \cdot (1, M)$$

Fließkommazahlen

A) Berechnen Sie den Dezimalwert der Belegung 10011000.



$$Z_D = (-1)^V \cdot 2^{E-3} \cdot (1,M)$$

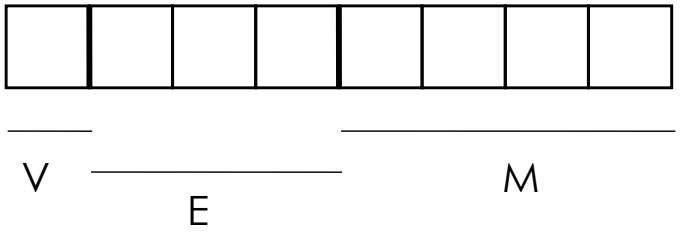
$Z = (-1) \cdot 2^{(1-3)} \cdot 1,5 = (-1) \cdot 2^{(-2)} \cdot 1,5$

$= (-1) \cdot 0,25 \cdot 1,5$

$= - 0,375$

Fließkommazahlen

B) Konvertieren Sie die Zahl **3,625$_D$** aus dem Dezimalsystem in Fließkomma. (keine *Klausuraufgabe)*



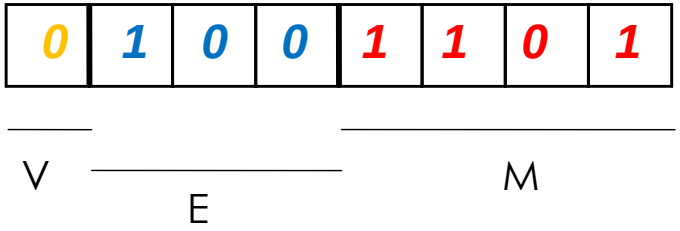
$$Z_D = (-1)^V \cdot 2^{E-3} \cdot (1,M)$$

Umwandeln in Dualzahl: $+3{,}625_D = +11{,}101_B$

Normalisieren: $11{,}101_B = 1{,}1101_B \bullet 2^1$

Exponent berechnen: $2^{E-3} = 2^1 \Rightarrow E = 3+1 = 4$

Fließkommazahl angeben:



3. Aufgabe, ÜB3

Ein 16 bit Mikroprozessor verwendet zum Speichern großer Zahlen eine 16-Bit-Fließkommadarstellung.
Die Zahlendarstellung erfolgt mit 1-Bit-Vorzeichen, 6-Bit-Exponent und 9-Bit-Mantisse.
Berechnen Sie die Summe der beiden Fließkommazahlen, indem Sie beide Zahlen auf den selben Exponenten umrechnen und dann die Mantissen addieren.
Geben Sie das Ergebnis in 16-Bit-Fließkommadarstellung an und geben Sie auch alle Zwischenschritte an.

16-Bit-Exponentialzahl = $-1^{V} \cdot 2^{(E - 31)} \cdot (1,M)$

	V	E5	E4	E3	E2	E1	E0	M8	M7	M6	M5	M4	M3	M2	M1	M0
Zahl 1: $47E1_H$																
+																
Zahl 2: 4261_H																

3. Aufgabe, ÜB3

V	E5	E4	E3	E2	E1	E0	M8	M7	M6	M5	M4	M3	M2	M1	M0

Zahl 1: $47E1_H$

0	1	0	0	0	1	1	1	1	1	1	0	0	0	0	1

\+

Zahl 2: 4261_H

0	1	0	0	0	0	1	0	0	1	1	0	0	0	0	1

Zahl 1: $+ 2^{(35-31)} \cdot 1,111100001$

Zahl 2: $+ 2^{(33-31)} \cdot 1,001100001 = + 2^{(35-31)} \cdot 0,01001100001$

Addition der Mantissen:

$1,111100001_B$

$+ \; 0,01001100001_B$

$= 10,00111100101_B = 2^1 \cdot 1,000111100_B$

(Mantisse auf 9 Stellen gekürzt!)

Ergebnis: $\Rightarrow \quad + 2^{(35-31)} \cdot 2^1 \cdot 1,000111100$

$= + 2^{(36-31)} \cdot 1,000111100$

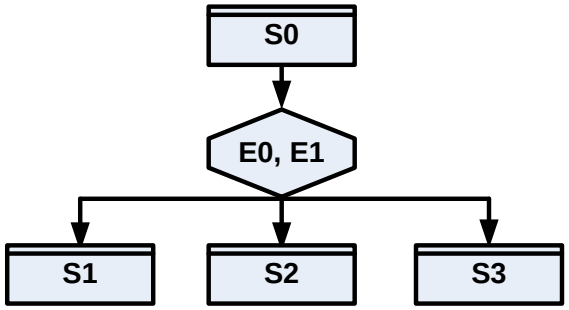
0	1	0	0	1	0	0	0	0	0	1	1	1	1	0	0

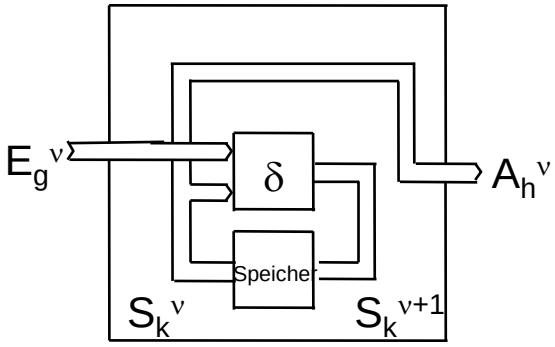
Automaten

Es gibt drei Typen von Automaten:

Medwedew-Automat

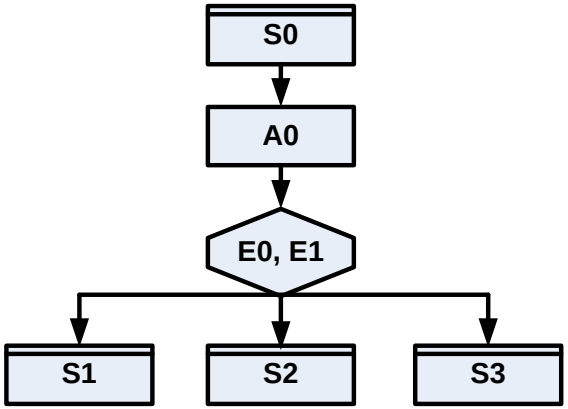
$A_h^{\nu} = S_k^{\nu}$

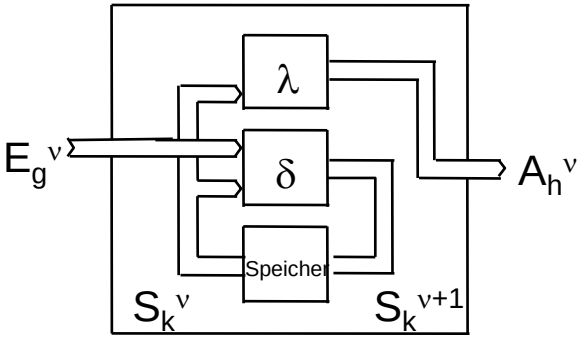




Moore-Automat

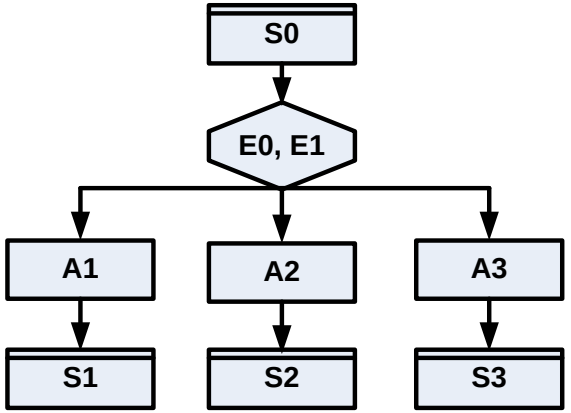
$A_h^{\nu} = \lambda (S_k^{\nu})$

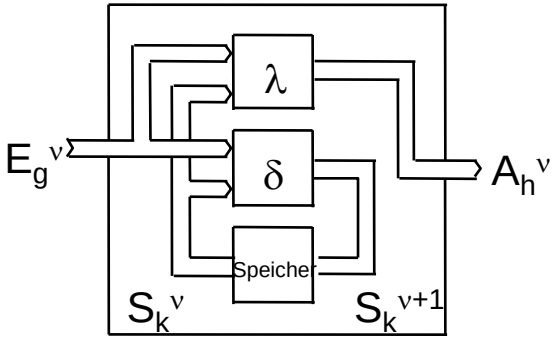




Mealy-Automat

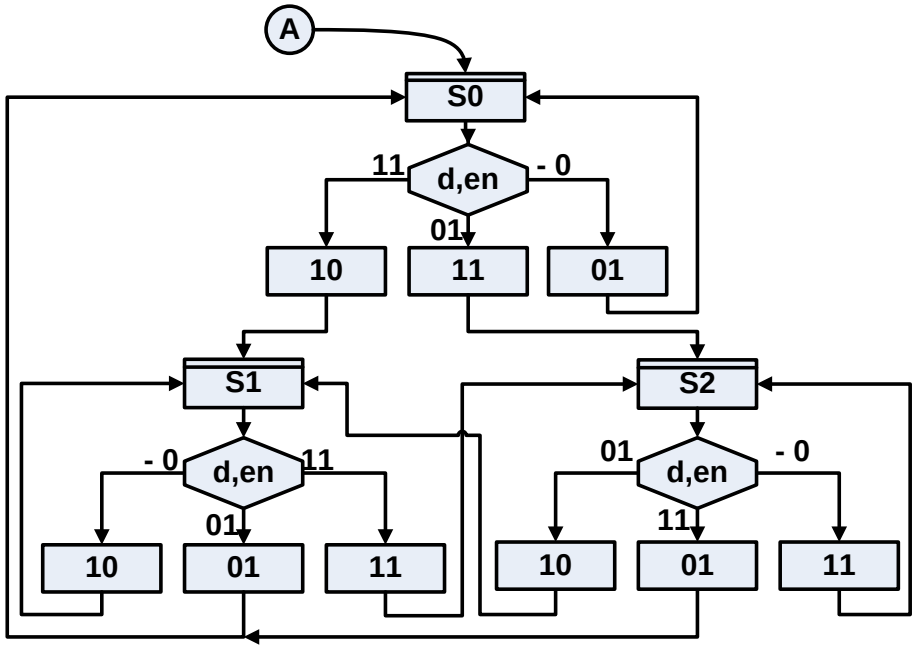
$A_h^{\nu} = \lambda (E_g^{\nu}, S_k^{\nu})$





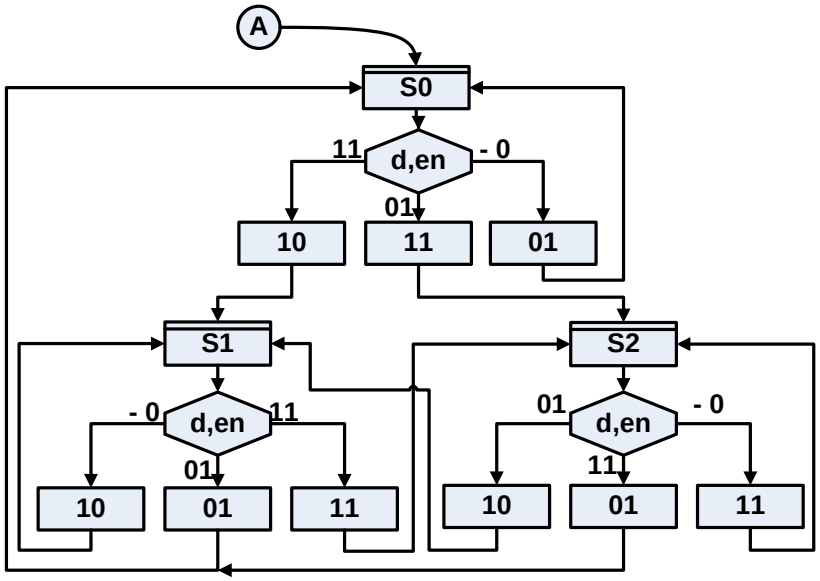
Automaten

Klausuraufgabe WS2009/2010



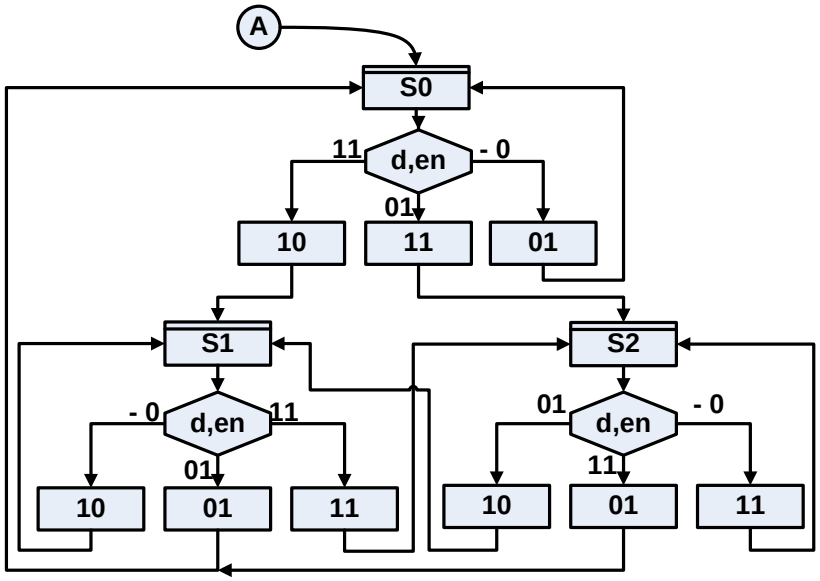
A) Das Ablaufdiagramm soll in eine Ablauftabelle übertragen werden. Vervollständigen Sie die Felder in der Tabelle und streichen Sie eventuell nicht notwendige Spalten (Q: Zustandskodierung, X: Eingangsgröße, Y: Ausgangssignale, Zustandsspeicherung über Toggle Flipflops). Verwenden Sie eine minimale Anzahl von Flip Flops.

Automaten



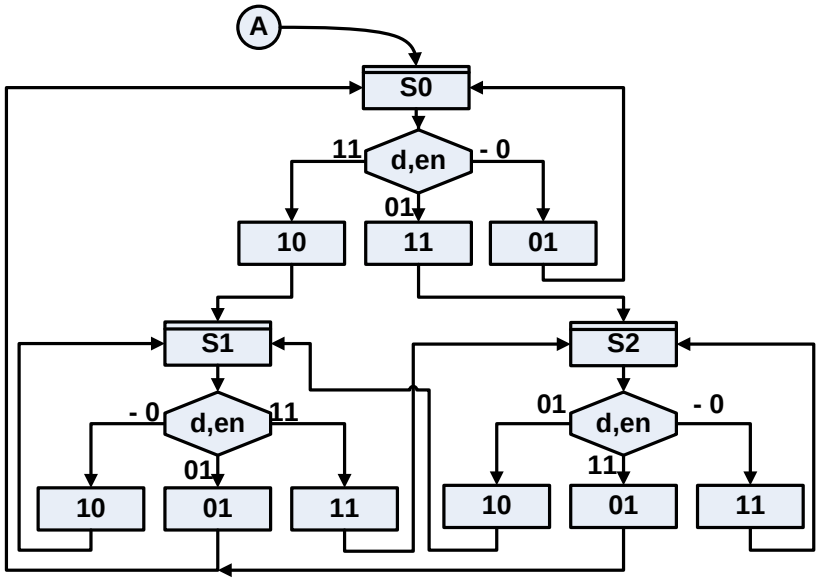
	Q^t			X^t		Q^{t+1}			T-FF			Y^t		
	q_2	q_1	q_0	d	en	q_2	q_1	q_0	t_2	t_1	t_0	y_2	y_1	y_0
		1	0	-	0									
				0	1		0	1						
				1	1									
		0	1	-	0									
S1				0	1		0	0						
				1	1									
				-	0									
				0	1		1	0						
				1	1									

Automaten



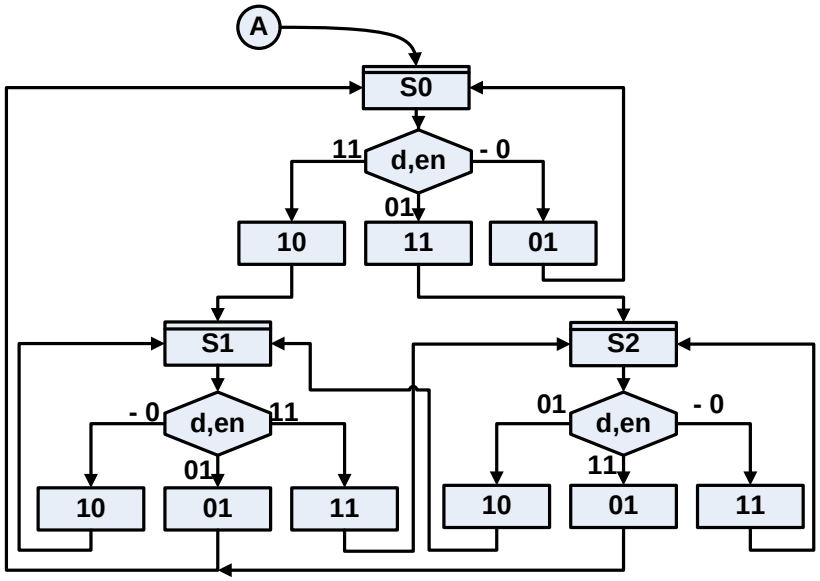
	Q^t			X^t		Q^{t+1}			T-FF			Y^t		
	q_2	q_1	q_0	d	en	q_2	q_1	q_0	t_2	t_1	t_0	y_2	y_1	y_0
		1	0	-	0									
				0	1		0	1						
				1	1									
		0	1	-	0									
S1				0	1		0	0						
				1	1									
		0	0	-	0									
				0	1		1	0						
				1	1									

Automaten



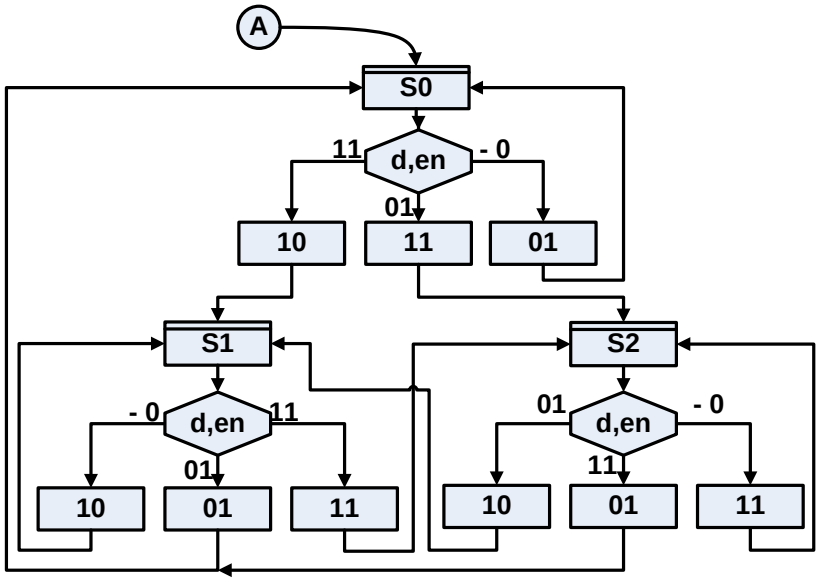
	Q^t			X^t		Q^{t+1}			T-FF			Y^t		
	q_2	q_1	q_0	d	en	q_2	q_1	q_0	t_2	t_1	t_0	y_2	y_1	y_0
		1	0	-	0									
				0	1		0	1						
				1	1									
		0	1	-	0									
S1				0	1		0	0						
				1	1									
		0	**0**	-	0									
S0				0	1		1	0						
				1	1									

Automaten



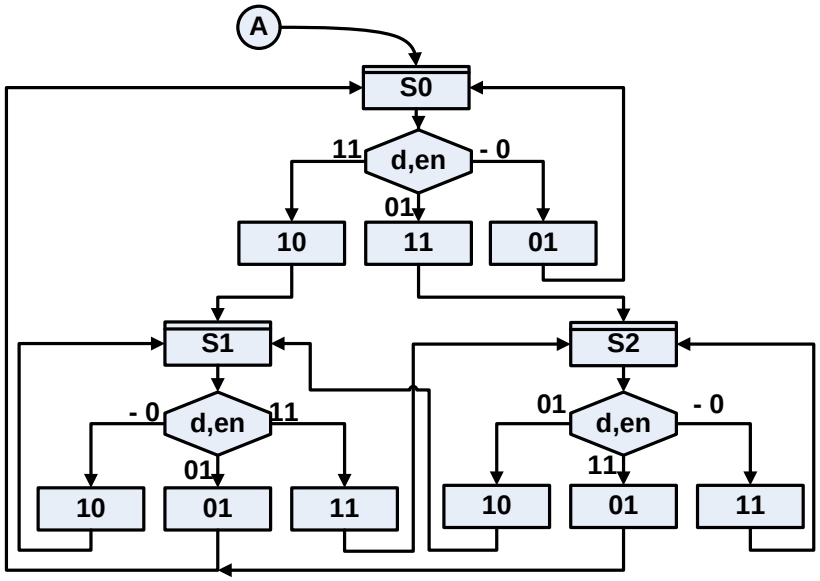
	Q^t			X^t		Q^{t+1}			T-FF			Y^t		
	q_2	q_1	q_0	d	en	q_2	q_1	q_0	t_2	t_1	t_0	y_2	y_1	y_0
S2		1	0	-	0									
				0	1		0	1						
				1	1									
S1		0	1	-	0									
				0	1		0	0						
				1	1									
S0		**0**	**0**	-	0									
				0	1		1	0						
				1	1									

Automaten



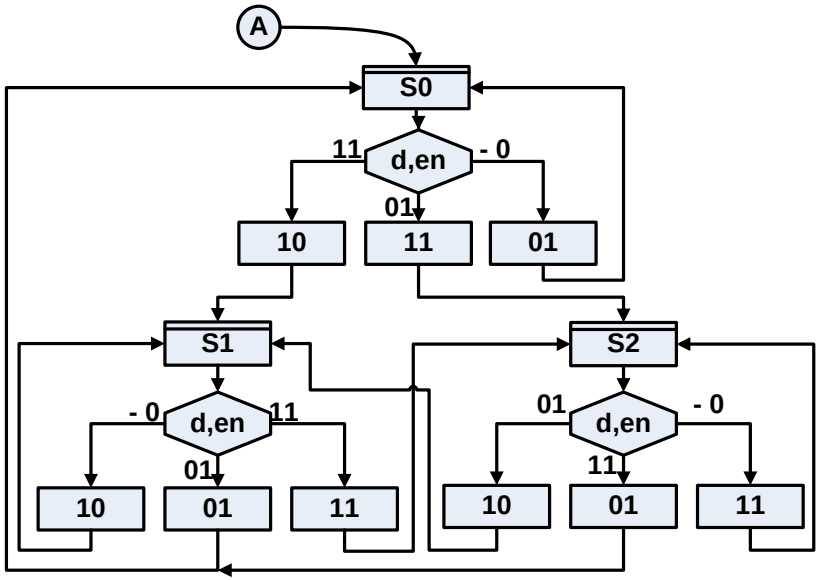
	Q^t			X^t		Q^{t+1}			T-FF			Y^t		
	q_2	q_1	q_0	d	en	q_2	q_1	q_0	t_2	t_1	t_0	y_2	y_1	y_0
S2		1	0	-	0		1	0					1	1
				0	1		0	1						
				1	1									
S1		0	1	-	0									
				0	1		0	0						
				1	1									
S0		**0**	**0**	-	0									
				0	1		1	0						
				1	1									

Automaten



	Q^t			X^t		Q^{t+1}			T-FF			Y^t		
	q_2	q_1	q_0	d	en	q_2	q_1	q_0	t_2	t_1	t_0	y_2	y_1	y_0
S2		1	0	-	0		1	0					1	1
				0	1		0	1					1	0
				1	1		0	0					0	1
S1		0	1	-	0		0	1					1	0
				0	1		0	0					0	1
				1	1		1	0					1	1
S0		**0**	**0**	-	0		0	0					0	1
				0	1		1	0					1	1
				1	1		0	1					1	0

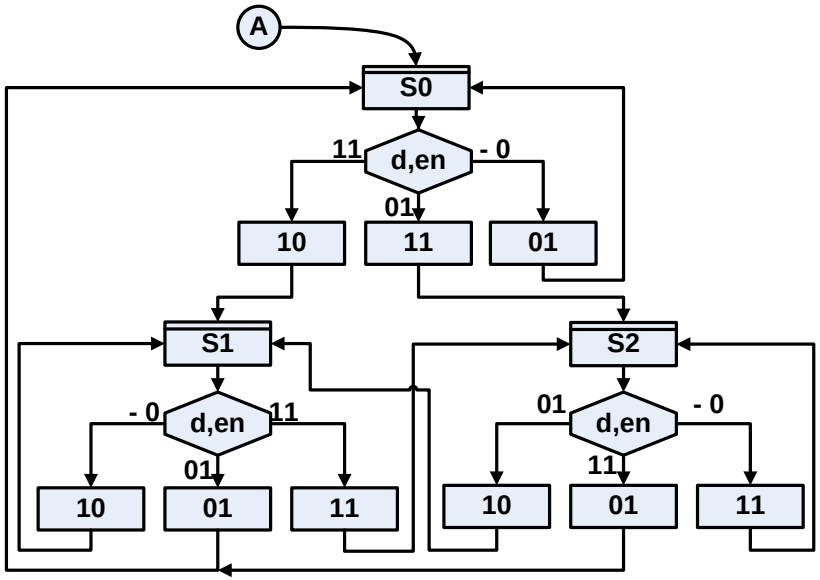
Automaten



Q	T	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

	Q^t			X^t		Q^{t+1}			T-FF			Y^t		
	q_2	q_1	q_0	d	en	q_2	q_1	q_0	t_2	t_1	t_0	y_2	y_1	y_0
S2		1	0	-	0		1	0		0	0		1	1
				0	1		0	1					1	0
				1	1		0	0					0	1
S1		0	1	-	0		0	1					1	0
				0	1		0	0					0	1
				1	1		1	0					1	1
S0		**0**	**0**	-	0		0	0					0	1
				0	1		1	0					1	1
				1	1		0	1					1	0

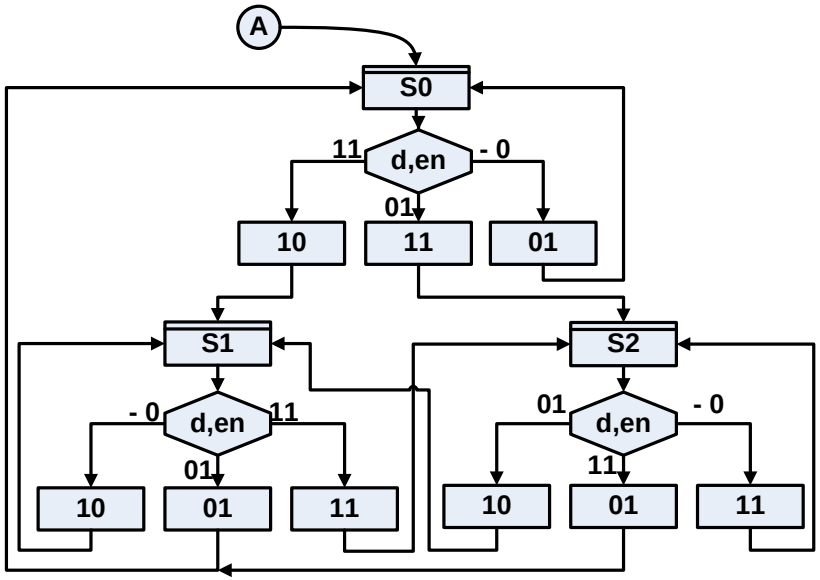
Automaten



Q	T	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

	Q^t			X^t		Q^{t+1}			T-FF			Y^t		
	q_2	q_1	q_0	d	en	q_2	q_1	q_0	t_2	t_1	t_0	y_2	y_1	y_0
S2		1	0	-	0		1	0		0	0		1	1
				0	1		0	1		1	1		1	0
				1	1		0	0					0	1
S1		0	1	-	0		0	1					1	0
				0	1		0	0					0	1
				1	1		1	0					1	1
S0		**0**	**0**	-	0		0	0					0	1
				0	1		1	0					1	1
				1	1		0	1					1	0

Automaten

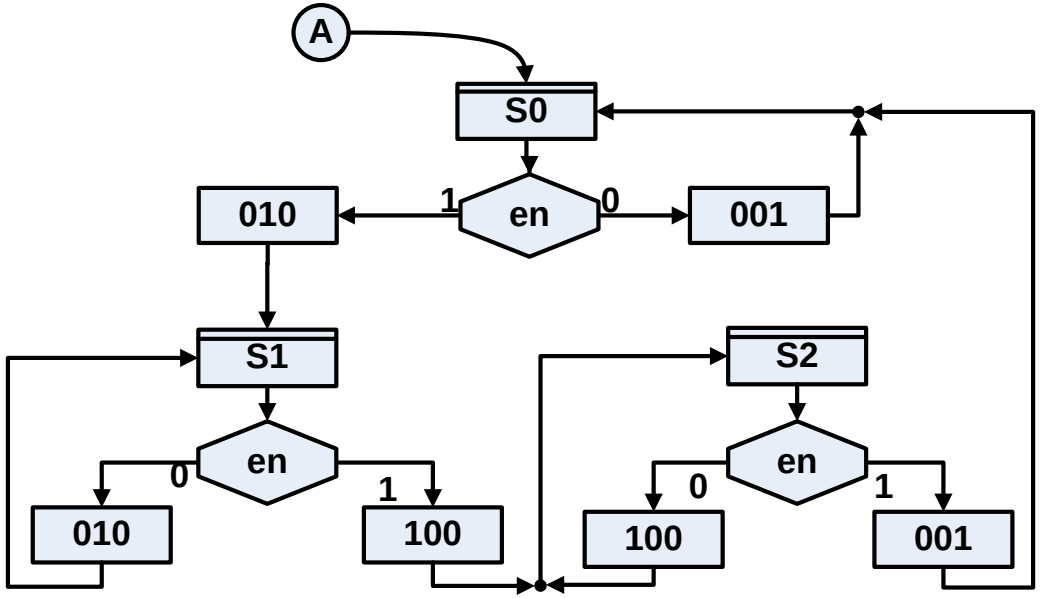


Q	T	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

	Q^t			X^t		Q^{t+1}			T-FF			Y^t		
	q_2	q_1	q_0	d	en	q_2	q_1	q_0	t_2	t_1	t_0	y_2	y_1	y_0
S2		1	0	-	0		1	0		0	0		1	1
				0	1		0	1		1	1		1	0
				1	1		0	0		1	0		0	1
S1		0	1	-	0		0	1		0	0		1	0
				0	1		0	0		0	1		0	1
				1	1		1	0		1	1		1	1
S0		**0**	**0**	-	0		0	0		0	0		0	1
				0	1		1	0		1	0		1	1
				1	1		0	1		0	1		1	0

Automaten

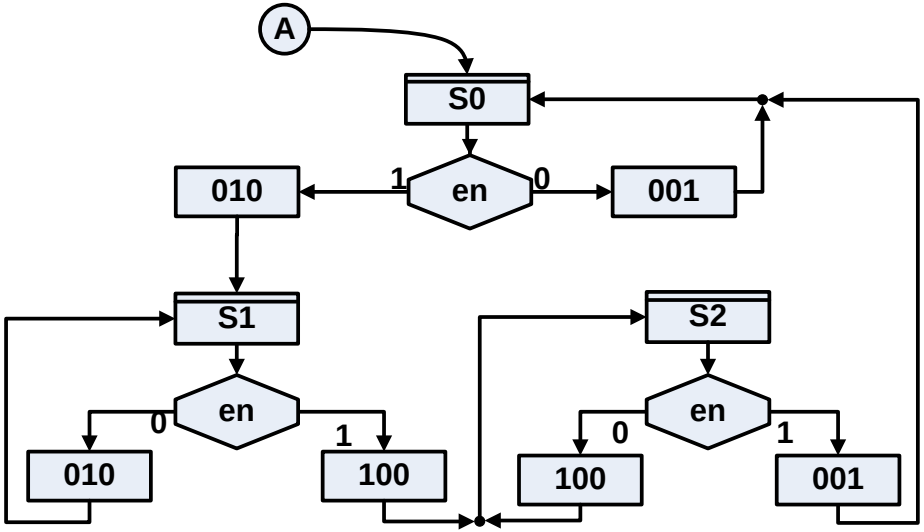
Klausuraufgabe WS2009/2010



B) In der Abbildung ist eine vereinfachte Version eines Ablaufdiagramms gegeben. Die darin beschriebene Funktionalität soll als Medwedewautomat realisiert werden. Geben Sie die Kodierung der minimal notwendigen Zustände an wie sie sich aus dem Ablaufdiagramm oben ableiten lässt.

Automaten

B) In der Abbildung ist eine vereinfachte Version eines Ablaufdiagramms gegeben. Die darin beschriebene Funktionalität soll als Medwedewautomat realisiert werden. Geben Sie die Kodierung der minimal notwendigen Zustände an wie sie sich aus dem Ablaufdiagramm oben ableiten lässt.



Kodierung für Medwedewautomat:

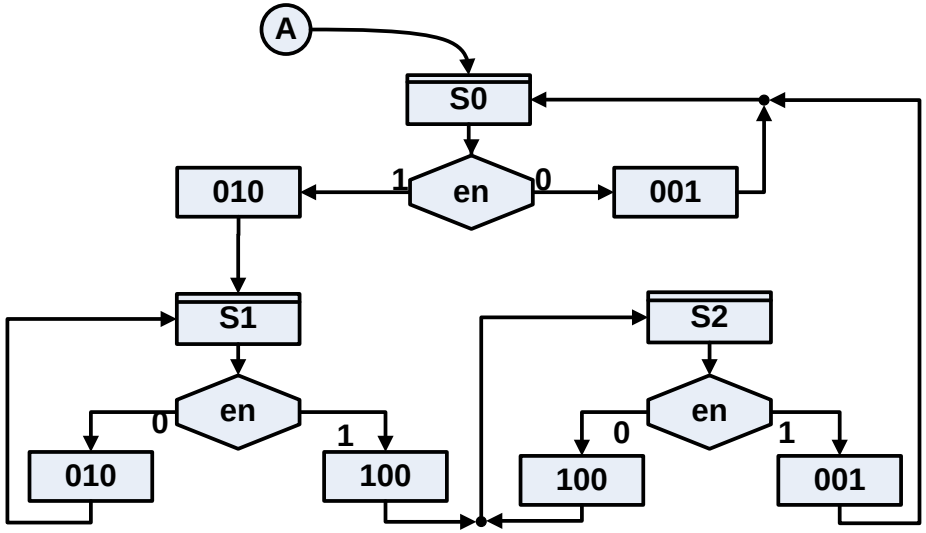
S0: 001

S1: 010

S2: 100

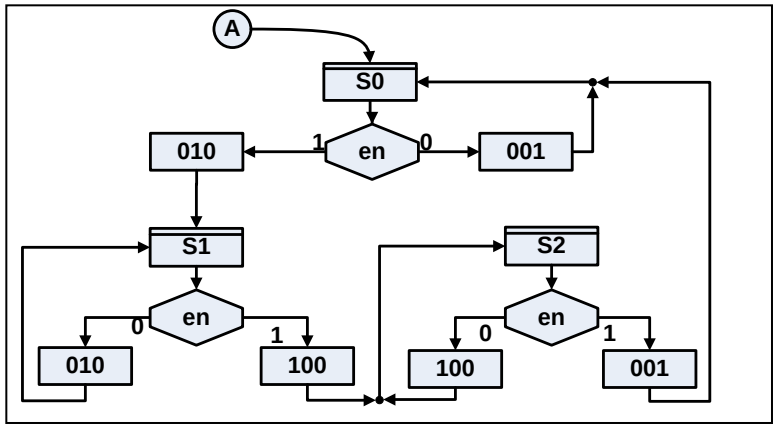
Automaten

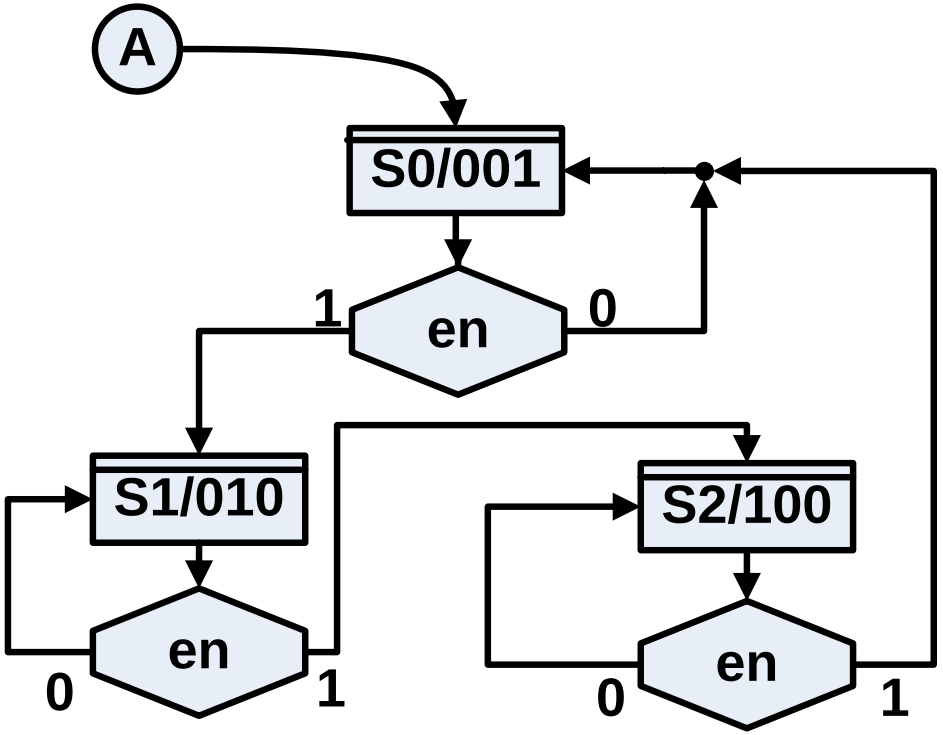
C) Zeichnen Sie den vereinfachten Automaten als Medwedewautomat. Verwenden Sie dabei eine minimale Anzahl von Zuständen



Automaten

C) Zeichnen Sie den vereinfachten Automaten als Medwedewautomat. Verwenden Sie dabei eine minimale Anzahl von Zuständen

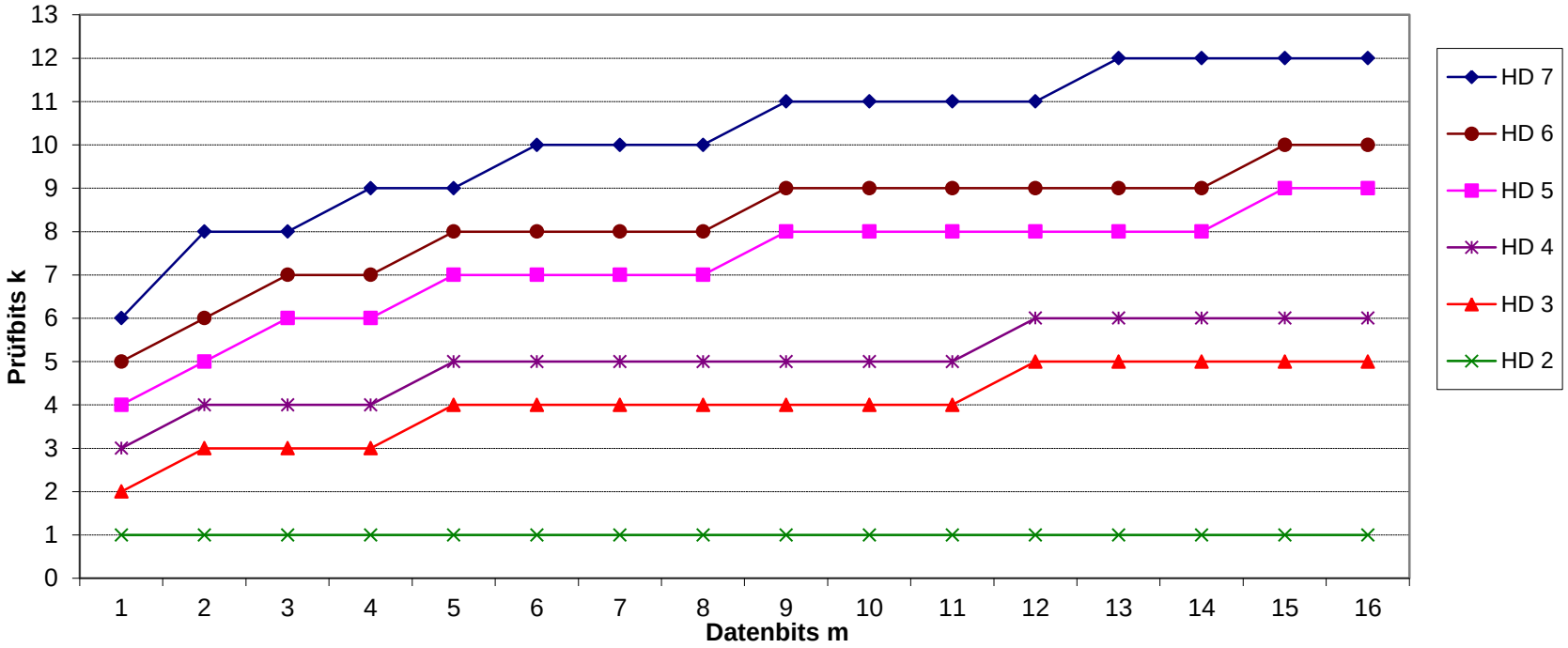




Hamming-Codes

Prüfbare und korrigierbare Codes:

→ Systematische Konstruktion



→ Notwendige Anzahl von **Prüfstellen k**
in Abhängigkeit von der Anzahl der **Informationsstellen m**
um **minimale Hamming-Distanz HD_{min} = d** zu erhalten

→ Systematische **Codes** nach **Hamming**

Hamming-Codes

Systematik: Konstruktion von Hamming-Codes

Aufbau eines 1-F-korrigierbaren Codes:

→ Einfachfehler sind korrigierbar: $HD_{min} = 3$

1. Schritt	lfd. Nr. duale Kennzahl	1 001	2 010	3 011	4 100	5 101	6 110	7 111
2. Schritt	1. Stellenbelegung Bestimmung von y1	y1 0		x1 0		x2 1		x3 1
3. Schritt	2. Stellenbelegung Bestimmung von y2		y2 1	x1 0			X4 0	x3 1
4. Schritt	3. Stellenbelegung Bestimmung von y3				y3 0	x2 1	x4 0	x3 1

→ Zuordnung der geprüften Stellen zu den Prüfstellen:

Beispiel: **1. Prüfstelle y_1** überprüft **alle Binärstellen,** die in der **1. Stelle (im 1. Bit)** der **Dualzahlendarstellungen** der entsprechenden **Binärstellenpositionen ("duale Kennzahlen")** eine **"1"** aufweisen

Hamming-Codes

Klausuraufgabe WS2009/2010

Konstruieren Sie den 1-F-korrigierbaren Hammingcode. Gegeben ist dazu ein fehlerfreier Datenstrom unter Verwendung gerader Parität: ***0100101 0001111 0010110 1000011*** Die Reihenfolge der übertragenen Bits entspricht der Abbildung.

X3	X2	X1	X0	Y2	Y1	Y0

A) Ergänzen Sie die untenstehende Tabelle. Bestimmen Sie die korrekte Zuordnung von Prüf- und Datenbits entsprechend der in der Tabelle gegebenen dualen Kennzahlen. Tragen Sie in der folgenden Tabelle ein, welche Datenbits x0, x1, x2, x3 durch welche Prüfbits y0, y1, y2 geschützt werden.

lfd. Nr. duale Kennzahl	1 001	2 010	3 011	4 100	5 101	6 110	7 111
1. Stelle	y0						
2. Stelle		y1					
3. Stelle				y2			

Hamming-Codes

Klausuraufgabe WS2009/2010

Konstruieren Sie den 1-F-korrigierbaren Hammingcode. Gegeben ist dazu ein fehlerfreier Datenstrom unter Verwendung gerader Parität: ***0100101 0001111 0010110 1000011*** Die Reihenfolge der übertragenen Bits entspricht der Abbildung.

X3	X2	X1	X0	Y2	Y1	Y0

A) Ergänzen Sie die untenstehende Tabelle. Bestimmen Sie die korrekte Zuordnung von Prüf- und Datenbits entsprechend der in der Tabelle gegebenen dualen Kennzahlen. Tragen Sie in der folgenden Tabelle ein, welche Datenbits x0, x1, x2, x3 durch welche Prüfbits y0, y1, y2 geschützt werden.

lfd. Nr. duale Kennzahl	1 001	2 010	3 011	4 100	5 101	6 110	7 111
1. Stelle	y0				x2		
2. Stelle		y1					
3. Stelle				y2	x2		

Hamming-Codes

Klausuraufgabe WS2009/2010

Konstruieren Sie den 1-F-korrigierbaren Hammingcode. Gegeben ist dazu ein fehlerfreier Datenstrom unter Verwendung gerader Parität: ***0100101 0001111 0010110 1000011*** Die Reihenfolge der übertragenen Bits entspricht der Abbildung.

X3	X2	X1	X0	Y2	Y1	Y0

A) Ergänzen Sie die untenstehende Tabelle. Bestimmen Sie die korrekte Zuordnung von Prüf- und Datenbits entsprechend der in der Tabelle gegebenen dualen Kennzahlen. Tragen Sie in der folgenden Tabelle ein, welche Datenbits x0, x1, x2, x3 durch welche Prüfbits y0, y1, y2 geschützt werden.

lfd. Nr. duale Kennzahl	1 001	2 010	3 011	4 100	5 101	6 110	7 111
1. Stelle	y0				x2		x0
2. Stelle		y1					x0
3. Stelle				y2	x2		x0

Hamming-Codes

Klausuraufgabe WS2009/2010

Konstruieren Sie den 1-F-korrigierbaren Hammingcode. Gegeben ist dazu ein fehlerfreier Datenstrom unter Verwendung gerader Parität: ***0100101 0001111 0010110 1000011*** Die Reihenfolge der übertragenen Bits entspricht der Abbildung.

X3	X2	X1	X0	Y2	Y1	Y0

A) Ergänzen Sie die untenstehende Tabelle. Bestimmen Sie die korrekte Zuordnung von Prüf- und Datenbits entsprechend der in der Tabelle gegebenen dualen Kennzahlen. Tragen Sie in der folgenden Tabelle ein, welche Datenbits x0, x1, x2, x3 durch welche Prüfbits y0, y1, y2 geschützt werden.

lfd. Nr. duale Kennzahl	1 001	2 010	3 011	4 100	5 101	6 110	7 111
1. Stelle	y0				x2		x0
2. Stelle		y1				x1	x0
3. Stelle				y2	x2	x1	x0

Hamming-Codes

Klausuraufgabe WS2009/2010

Konstruieren Sie den 1-F-korrigierbaren Hammingcode. Gegeben ist dazu ein fehlerfreier Datenstrom unter Verwendung gerader Parität: ***0100101 0001111 0010110 1000011*** Die Reihenfolge der übertragenen Bits entspricht der Abbildung.

X3	X2	X1	X0	Y2	Y1	Y0

A) Ergänzen Sie die untenstehende Tabelle. Bestimmen Sie die korrekte Zuordnung von Prüf- und Datenbits entsprechend der in der Tabelle gegebenen dualen Kennzahlen. Tragen Sie in der folgenden Tabelle ein, welche Datenbits x0, x1, x2, x3 durch welche Prüfbits y0, y1, y2 geschützt werden.

lfd. Nr. duale Kennzahl	1 001	2 010	3 011	4 100	5 101	6 110	7 111
1. Stelle	y0		x3		x2		x0
2. Stelle		y1	x3			x1	x0
3. Stelle				y2	x2	x1	x0

Hamming-Codes

Klausuraufgabe WS2009/2010

lfd. Nr. duale Kennzahl	1 001	2 010	3 011	4 100	5 101	6 110	7 111
1. Stelle	y0		x3		x2		x0
2. Stelle		y1	x3			x1	x0
3. Stelle				y2	x2	x1	x0

B) Es wurde eine zweite Folge aus Codewörtern empfangen. Es kam die gleiche Codierung wie in Aufgabenteil A) gefordert ist zum Einsatz. Wieviele Fehler sind bei der Übertragung mindestens aufgetreten? Markieren Sie die korrigierbaren Bits in der unten gegebenen Folge.

Codewort Folge: 0100011 1110000 0001000

Hamming-Codes

- Codewort 0100 011 in Diagramm einsetzen und Fehlerhafte Bitposition stelle ermitteln.

X3	X2	X1	X0	Y2	Y1	Y0

lfd. Nr. duale Kennzahl	1 001	2 010	3 011	4 100	5 101	6 110	7 111
1. Stelle	1		0		1		0
2. Stelle		1	0			0	0
3. Stelle				0	1	0	0

- Für welche Prüfbits liegt gerade Parität vor?
  - Y1 und Y2 weißen die falsche Parität auf
  - Duale Kennzahl der fehlerhaften stelle 110
  - X1 ist fehlerhaft, da X1 die duale Kennzahl 6 zugeordnet wurde

Hamming-Codes

- Nächstes Codewort (1110 000) in Diagramm einsetzen und Fehlerhafte Stelle ermitteln.

X3	X2	X1	X0	Y2	Y1	Y0

lfd. Nr. duale Kennzahl	1 001	2 010	3 011	4 100	5 101	6 110	7 111
1. Stelle	0		1		1		0
2. Stelle		0	1			1	0
3. Stelle				0	1	1	0

- Für welche Prüfbits liegt gerade Parität vor?
  - Für alle Prüfbits liegt gerade Parität vor
  - Es sind keine Übertragungsfehler aufgetreten

Hamming-Codes

- Leztes Codewort (0001 000) in Diagramm einsetzen und Fehlerhaftes Bit ermitteln.

X3	X2	X1	X0	Y2	Y1	Y0

lfd. Nr. duale Kennzahl	1 001	2 010	3 011	4 100	5 101	6 110	7 111
1. Stelle	0		0		0		1
2. Stelle		0	0			0	1
3. Stelle				0	0	0	1

- Für welche Prüfbits liegt gerade Parität vor?
  - Für keines der Prüfbits liegt gerade Parität vor.
  - Duale Kennzahl der fehlerhaften Stelle ist daher 111
  - Die Duale Kennzahl 7 ist X0 zugeordnet => X0 muss korrigiert werden

Fehlererkorrektur durch Blocksicherung

Beispiel: **Fehlerlokalisierung durch Blocksicherungsverfahren**

Ziffer	Codewörter mit gerader Parität				
5	0	1	0	1	0
4	0	1	0	0	1
1	0	0	1	1	1
3	0	0	1	1	0
9	1	0	0	1	0
8	1	0	0	0	1
Prüfwort	0	0	1	0	1

Zeile mit Fehler
(ungerade Parität)

Spalte mit Fehler
(ungerade Parität)

- ***2 Fehler erkennbar***
- ***1 Fehler korrigierbar***

Universell nutzbare Hardwarestrukturen

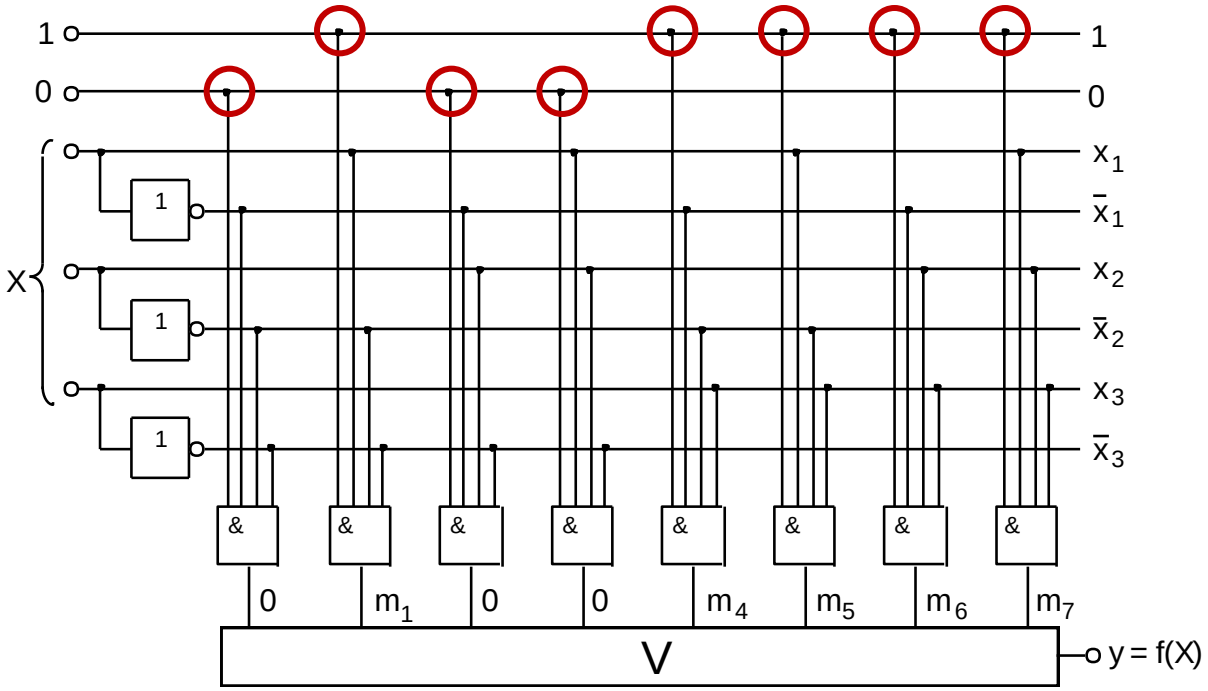
- **Prinzipielle Unterscheidung in folgende beiden zweistufigen Klassen:**
  - **Minterm- bzw. Maxterm-orientiert (basierend auf DNF bzw. KNF)**
  - **blockorientiert (basierend auf DMF bzw. KMF**
- **Universelle Realisierungsmöglichkeit:**
  - **durch eine Matrixstruktur**

Normalformorientierte Strukturen: DNF

Beispiel: ULA (Universal Logic Array)

- besteht aus 2^n **UND-Schaltgliedern**, zur Realisierung **jedes Minterms** in der **1. Stufe**
- einem oder mehreren **ODER-Schaltgliedern** in der **2. Stufe**
- **Personalisierung:** in 1. Stufe durch konjunktive Verknüpfung der Minterme mit 1 oder 0

Beispiel: dargestellte Funktion $y = m_1 \vee m_4 \vee m_5 \vee m_6 \vee m_7$

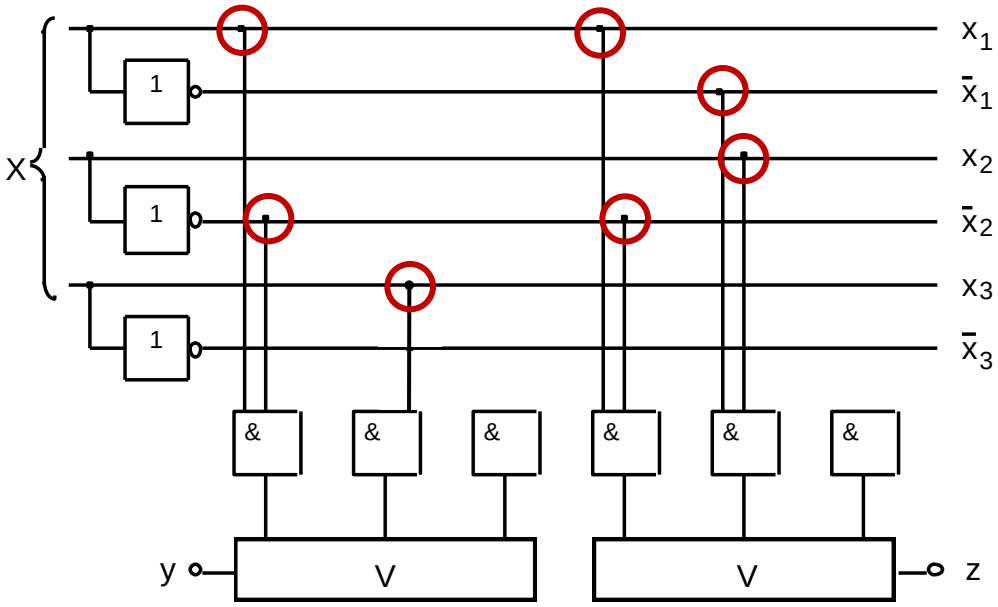


Blockorientierte Strukturen: DMF

Beispiel: PAL (Programmable Array Logic):

- besteht i.a. aus weniger als 2^n UND-Schaltgliedern in der 1. Stufe
- einem oder mehreren ODER-Schaltgliedern in der 2. Stufe
- die Personalisierung erfolgt in der 1. Stufe durch Festlegung der Primterme

Beispiel: dargestellte Funktion $y = (\overline{x_2} \,\&\, x_1) \vee (x_3), \quad z = (\overline{x_2} \,\&\, x_1) \vee (x_2 \,\&\, \overline{x_1})$



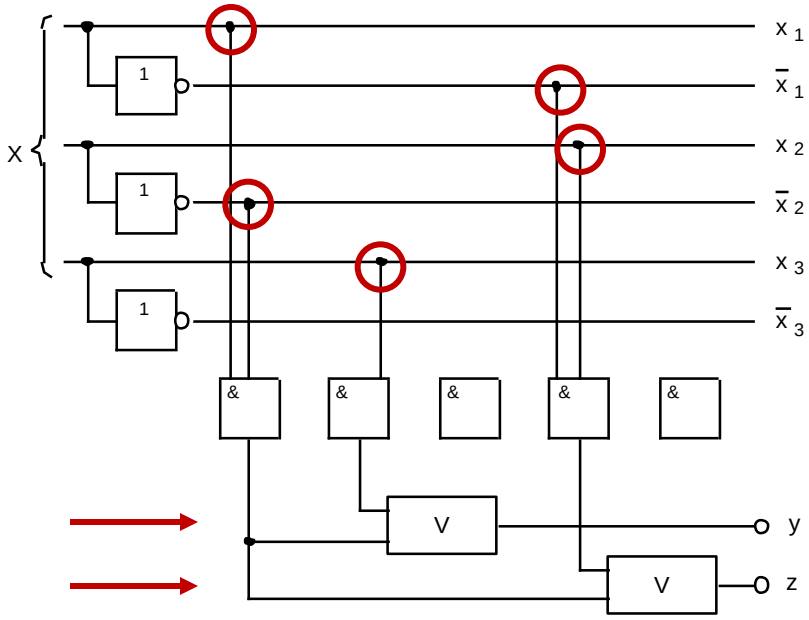
Blockorientierte Strukturen: DMF

Beispiel: PLA (Programmable Logic Array)

- Aufbau **ähnlich** zu **PALs**, jedoch **flexibler** in der **2. Stufe**
- **beide Matrizen** sind **programmierbar** (*personalisierbar*)

Vorteil: **mehrfache Ausnutzung** von **Primtermen**

Beispiel: dargestellte Funktion $y = (\overline{x_2} \,\&\, x_1) \vee (x_3), \quad z = (\overline{x_2} \,\&\, x_1) \vee (x_2 \,\&\, \overline{x_1})$



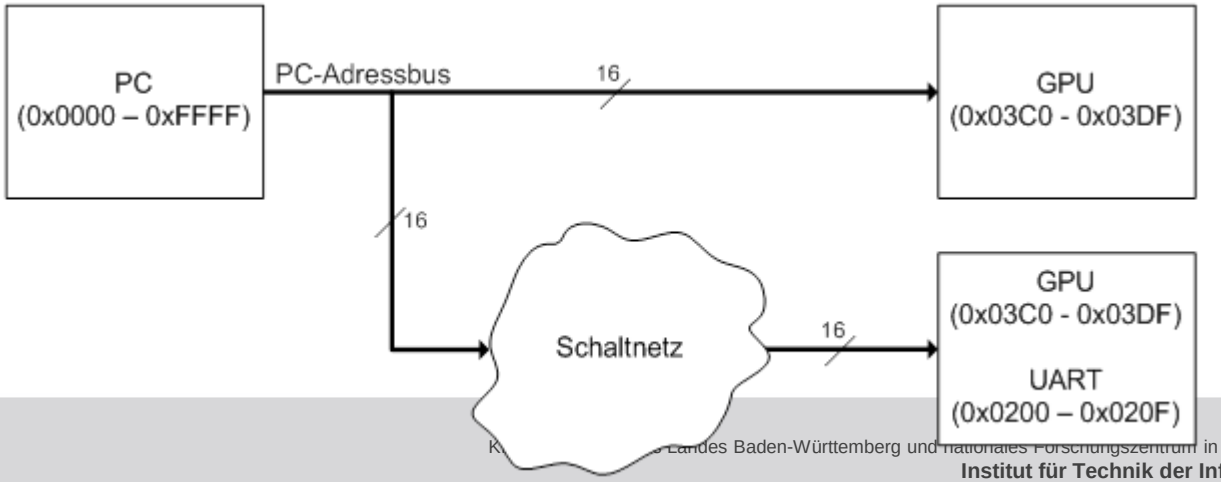
ISA-Bus (Anwendungsaufgabe)

Klausuraufgabe WS2009/2010

Der Steuer-PC einer Industrieanlage soll um ein zusätzliches Touchscreen Display erweitert werden. Das neue Display wird, parallel zum alten Display des Steuer-PCs, über den vorhandenen ISA Bus angesprochen. Die 16 Adressleitungen des ISA Bus(A15 – A0) stellen Adressen in binärer Form dar.

Beide Displays reagieren auf die Adressen 0x03C0-0x03DF, so dass ein einfaches Einbauen der neuen Hardware zu einem Adresskonflikt zwischen den beiden Displays führen würde. Das neue Display soll nun vom PC aus mit den Adressen 0x11C0 - 0x11DF angesprochen werden.

Dazu wird ein Schaltnetz vor das neue Display geschaltet. Diese kodiert den Adressbereich 0x11C0 - 0x11DF auf den Bereich 0x03C0-0x03DF um.



ISA-Bus (Anwendungsaufgabe)

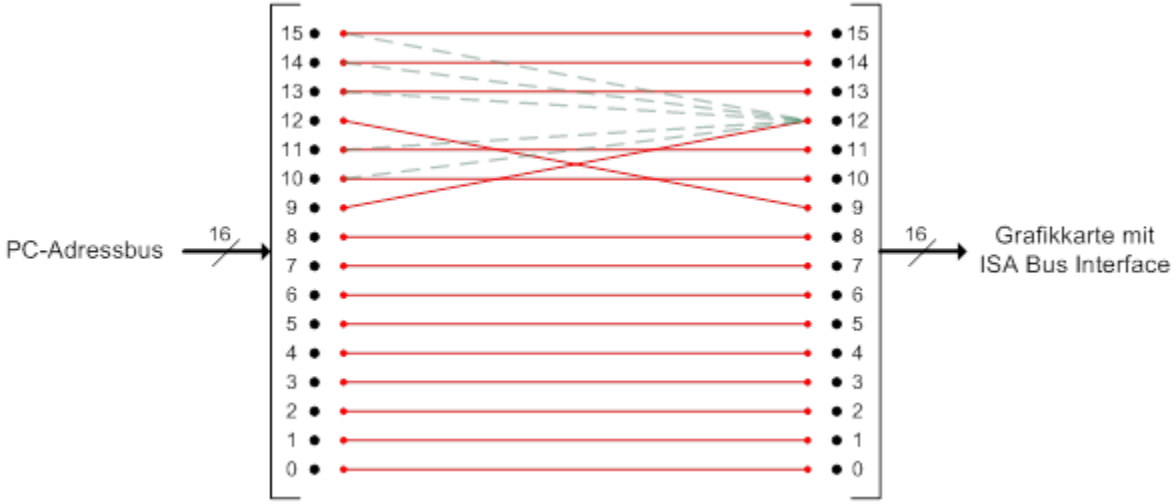
A) Geben Sie die Adressen 0x11C0, 0x11DF, 0x03C0, 0x03DF in binärer Darstellung an.

0x11C0 = 0001 0001 1100 0000
0x11DF = 0001 0001 1101 1111
0x03C0 = 0000 0011 1100 0000
0x03DF = 0000 0011 1101 1111

ISA-Bus (Anwendungsaufgabe)

B) Geben Sie die minimal mögliche Umverdrahtung der Adressleitungen an, mit der die Adressen von 0x11C0 - 0x11DF in den Bereich 0x03C0-0x03DF überführt werden können.

(Bemerkung: Sie können davon ausgehen, dass die Karte Zugriffe außerhalb ihres Adressbereichs ignoriert.)

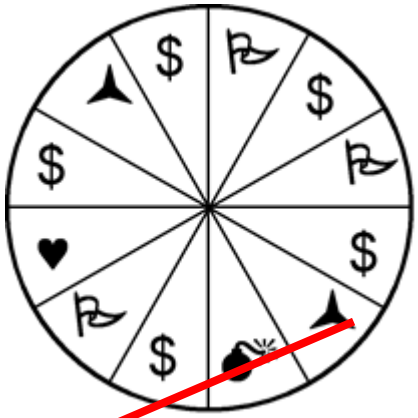


0x11C0 = 0001 0001 1100 0000
0x03C0 = 0000 0011 1100 0000

0x11DF = 0001 0001 1101 1111
0x03DF = 0000 0011 1101 1111

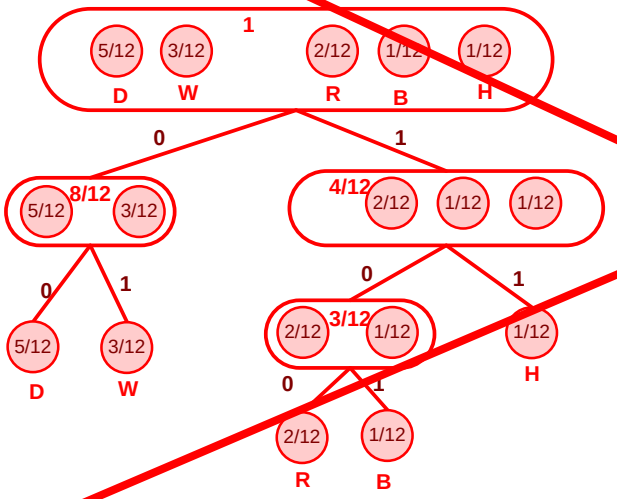
Spielautomat (SHANNON-FANØ)

Klausuraufgabe SS2009



1.2 C) Entwickeln Sie jetzt eine SHANNON-FANØ-Codierung und tragen Sie diese in die Tabelle ein.

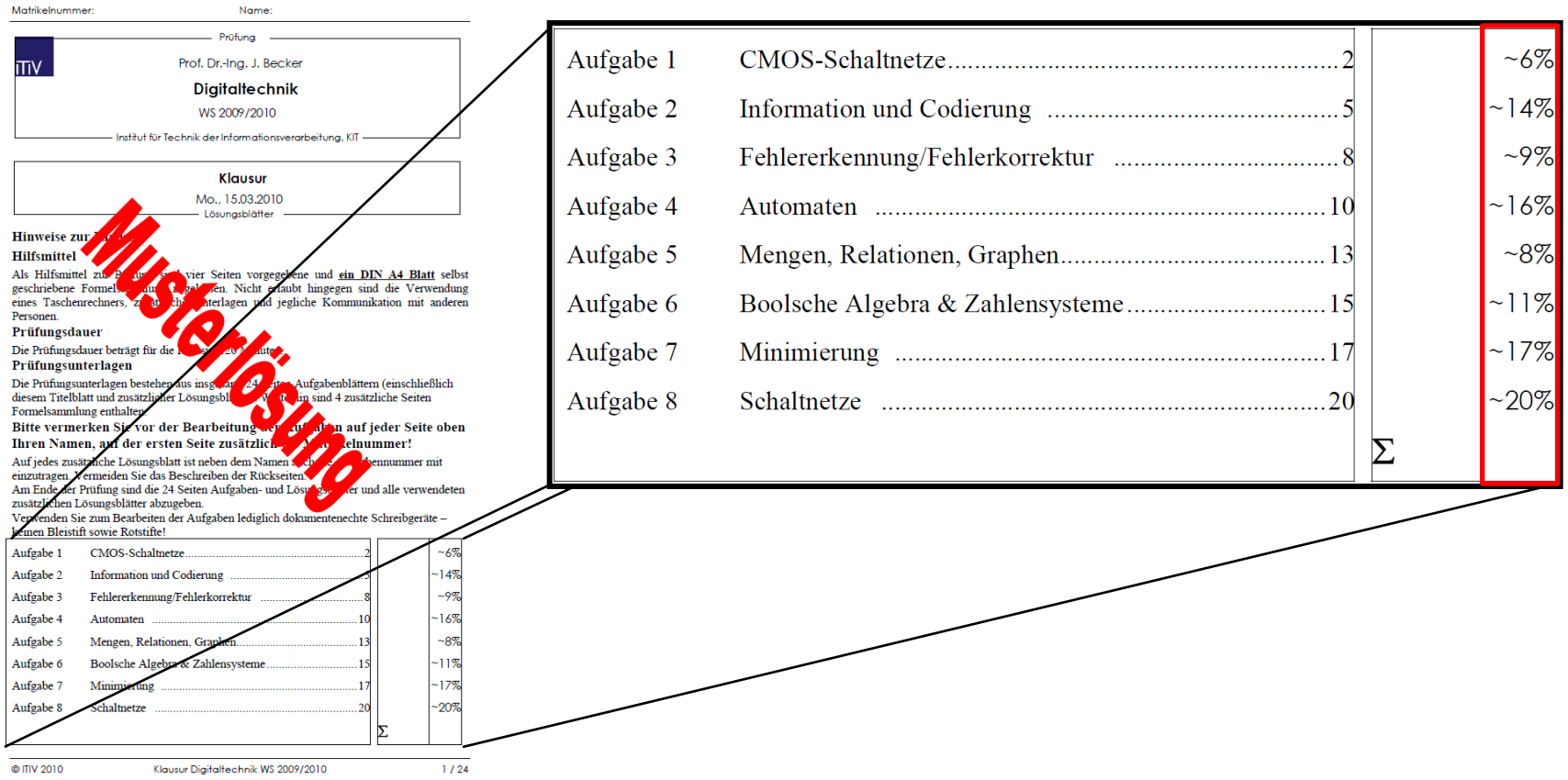
Falsch!!!



Symbol	Abk	Auftrittswahrscheinlichkeit	Ermittelte Codierung
	≙ B	1/12	101
	≙ D	5/12	00
	≙ H	1/12	11
	≙ R	2/12	100
	≙ W	3/12	01

Punkteverteilung in der Klausur

- Auf dem **Deckblatt** der Klausur wird **die Gewichtung** der **einzelnen Aufgaben** angegeben:



Matrikelnummer: Name:

Prüfung

ITIV

Prof. Dr.-Ing. J. Becker

Digitaltechnik

WS 2009/2010

Institut für Technik der Informationsverarbeitung, KIT

Klausur

Mo., 15.03.2010

Lösungsblätter

Hinweise zur [illegible]

Hilfsmittel

Als Hilfsmittel zu[illegible] sind vier Seiten vorgegebene und **ein DIN A4 Blatt** selbst geschriebene Formel[illegible] zugelassen. Nicht erlaubt hingegen sind die Verwendung eines Taschenrechners, [illegible]terlagen und jegliche Kommunikation mit anderen Personen.

Prüfungsdauer

Die Prüfungsdauer beträgt für die [illegible] 120 Minuten.

Prüfungsunterlagen

Die Prüfungsunterlagen bestehen aus insg[illegible] 24 [illegible] Aufgabenblättern (einschließlich diesem Titelblatt und zusätzlicher Lösungsbl[illegible]) [illegible]n sind 4 zusätzliche Seiten Formelsammlung enthalten.

Bitte vermerken Sie vor der Bearbeitung [illegible] auf jeder Seite oben Ihren Namen, auf der ersten Seite zusätzlich [illegible]elnummer!

Auf jedes zusätzliche Lösungsblatt ist neben dem Namen [illegible]benummer mit einzutragen. Vermeiden Sie das Beschreiben der Rückseiten.

Am Ende der Prüfung sind die 24 Seiten Aufgaben- und Lösungs[illegible]ter und alle verwendeten zusätzlichen Lösungsblätter abzugeben.

Verwenden Sie zum Bearbeiten der Aufgaben lediglich dokumentenechte Schreibgeräte – keinen Bleistift sowie Rotstifte!

Musterlösung

Aufgabe 1	CMOS-Schaltnetze	2	~6%
Aufgabe 2	Information und Codierung	5	~14%
Aufgabe 3	Fehlererkennung/Fehlerkorrektur	8	~9%
Aufgabe 4	Automaten	10	~16%
Aufgabe 5	Mengen, Relationen, Graphen	13	~8%
Aufgabe 6	Boolsche Algebra & Zahlensysteme	15	~11%
Aufgabe 7	Minimierung	17	~17%
Aufgabe 8	Schaltnetze	20	~20%
		Σ	

© ITIV 2010 Klausur Digitaltechnik WS 2009/2010 1 / 24

Klausurrelevante Themen

- **Alle Themengebiete und Aufgaben aus Übung und Tutorien!!!**
- **Aus der Vorlesung:**
  - 1. Funktion und Struktur
  - 2. Nachricht und Signal
  - 3. Informationsgehalt
  - 4. Codierung
  - 5. Optimale Codes *(JPEG-Verfahren nicht Klausurrelevant)*
  - 6. Hamming Codes
  - 7. Zahlensysteme
  - 8. Zahlendarstellung und Komplement
  - 9. Codewandlung
  - 10. Mathematische Grundlagen - Mengen
  - 11. Mathematische Grundlagen - Relationen
  - 12. Mathematische Grundlagen - Graphen
  - 13. Petrinetze *(nicht behandelt => nicht Klausurrelevant)*

Klausurrelevante Themen

- **Aus der Vorlesung** (Teil2)**:**
  - 14. Boolesche Algebra
  - 15. Schaltfunktionen
  - 16. Entwicklungssatz
  - 17. Minimierung Teil 1
  - 18. Minimierung Teil 2
  - 19. Schaltnetze Teil 1 *(nur Halb- und Volladdierer (1 Bit))*
  - 20. Schaltnetze Teil 2
  - 21. Schaltnetze Teil 3
  - 22. Automaten/FlipFlops
  - 23. FlipFlop Schaltungen *(FlipFlop-Typen und deren Funktionstabellen)*
  - 24. Funktionseinheiten Teil 1 *(nur Multiplexer und Zähler)*
  - 25. Funktionseinheiten Teil 2 *(nicht relevant)*
  - 26. Schalter & Gatter *(nur CMOS-Prinzip verstehen => komplementäre Netze, Tafelanschrieb relevant => Wohldefiniertheit, Kurzschluss, Undefiniert)*
  - 27. CMOS Technologie *(nur Folien 13, pull-up- und pull-down-Netz)*