Fachbereich Informatik

Praktikum und Werkstatt Technische Informatik (PWTI)

Lösungsvorschläge zu den Aufgaben

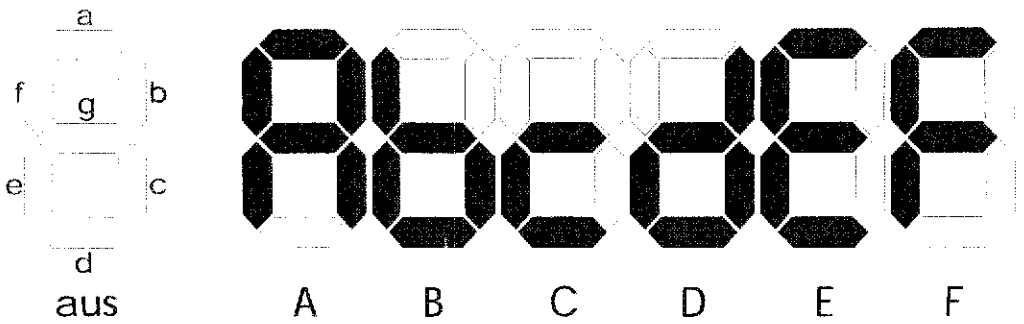
der Nachklausur im SS 2001

(am 15. September 2001)

Kurs 1707 "Technische Informatik I"

Zu Aufgabe 1: **(15 Punkte)**

Gesucht ist ein Codierer zur Ansteuerung einer 7-Segment-Anzeige, mit dem nur die Darstellung der hexadezimalen Ziffern A bis F ermöglicht werden soll. Zusätzlich soll die Anzeige auch abgeschaltet werden können. Das folgende Bild zeigt den Aufbau einer 7-Segment-Anzeige, die Benennung der Segmente mit a,..,g sowie die Darstellung der hexadezimalen Ziffern A,..,F:



a) Ergänzen Sie die folgende Wertetabelle:

Darstellung	Anzeigesegmente							BCD-Code[1]			
	a	b	c	d	e	f	g	X3	X2	X1	X0
aus	0	0	0	0	0	0	0	0	0	0	0
A	1	1	1	0	1	1	1	1	0	1	0
B	0	0	1	1	1	1	1	1	0	1	1
C	0	0	0	1	1	0	1	1	1	0	0
D	0	1	1	1	1	0	1	1	1	0	1
E	1	0	0	1	1	1	1	1	1	1	0
F	1	0	0	0	1	1	1	1	1	1	1

b) Stellen Sie die Funktionsgleichungen für die Ausgangsvariablen a, b und c auf und wandeln Sie sie durch algebraische Umformungen in eine möglichst kurze DNF um. Benennen Sie jeden Umformungsschritt.

$a = (X3 \wedge \overline{X2} \wedge X1 \wedge \overline{X0}) \vee (X3 \wedge X2 \wedge X1 \wedge \overline{X0}) \vee (X3 \wedge X2 \wedge$ Verschmelzung

$a = (X3 \wedge \overline{X2} \wedge X1 \wedge \overline{X0}) \vee (X3 \wedge X2 \wedge X1)$ Ausklammern, Absorption

$a = X3 \wedge X1 \wedge ((\overline{X2} \wedge \overline{X0}) \vee X2) = X3 \wedge X1 \wedge (\overline{X0} \vee X2)$ Zusammenfassen

$a = (X3 \wedge X1 \wedge \overline{X0}) \vee (X3 \wedge X2 \wedge X1) = X3 \wedge X1 \wedge (\overline{X0} \vee X2)$

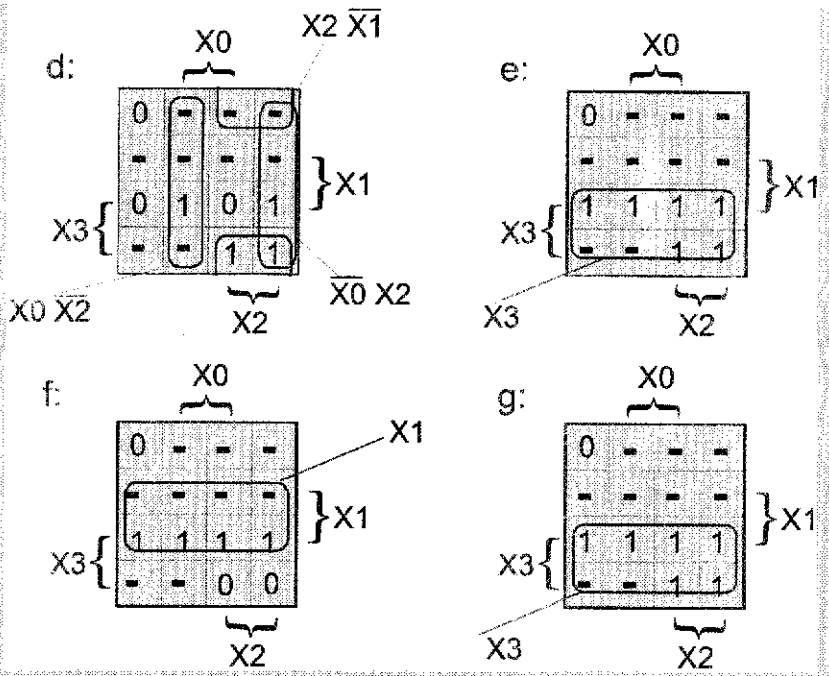
$b = (X3 \wedge \overline{X2} \wedge X1 \wedge \overline{X0}) \vee (X3 \wedge X2 \wedge \overline{X1} \wedge X0)$ Verschmelzung

$c = (X3 \wedge \overline{X2} \wedge X1 \wedge \overline{X0}) \vee (X3 \wedge \overline{X2} \wedge X1 \wedge X0) \vee (X3 \wedge X2 \wedge \overline{X1} \wedge X0)$

$c = (X3 \wedge \overline{X2} \wedge X1) \vee (X3 \wedge X2 \wedge \overline{X1} \wedge X0)$

[1] Die BCD-Ziffer ‚0' (0000) wird hier zum Ausschalten der Anzeige benutzt – anders als in gewöhnlichen 7-Segment-Anzeigen..

c) Stellen Sie jeweils mit Hilfe eines Karnaugh-Diagramms die Funktionsgleichungen der Ausgangsvariablen d, e, f und g in Form einer DNF auf. Markieren Sie dabei in den Karnaugh-Diagrammen die „don't care"-Terme mit einem „-" und lassen Sie diese in die Funktionsgleichung mit einfliessen, soweit dies eine Vereinfachung ergibt:



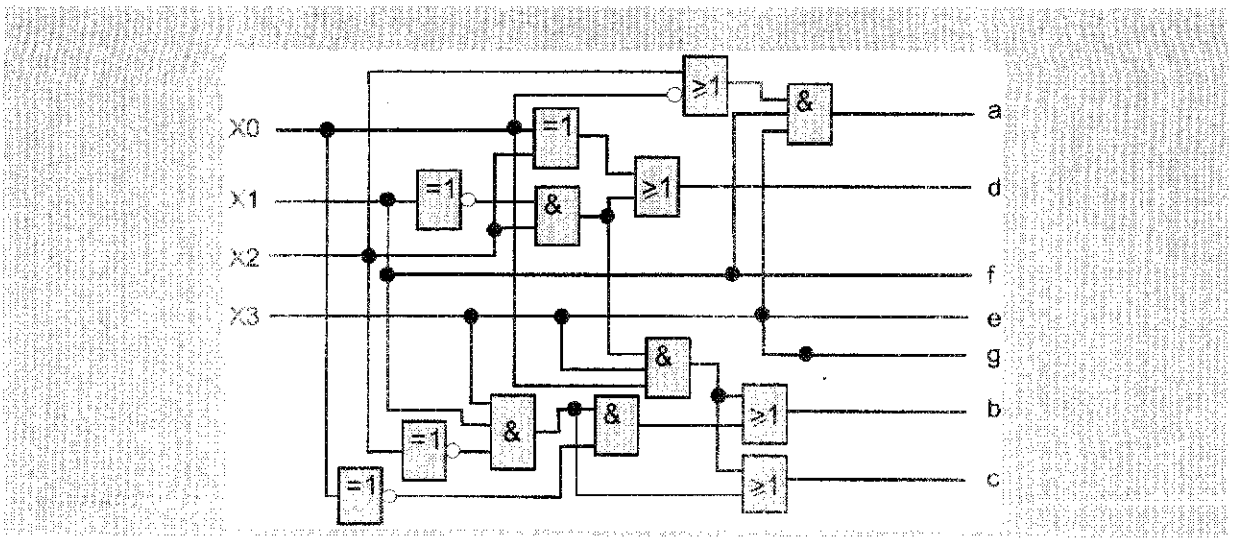
$$d = (\overline{X0} \wedge X2) \vee (X0 \wedge \overline{X2}) \vee (X2 \wedge \overline{X1}) = (X0 \langle + \rangle X2) \vee (X2 \wedge \overline{X1})$$

$$e = X3$$

$$f = X1$$

$$g = X3 = e$$

d) Zeichnen Sie zu den von Ihnen gewonnenen Schaltfunktionen das Schaltnetz des gesuchten Codierers.



Zu Aufgabe 2: **(15 Punkte)**

Gegeben seien die Funktionen F und G mit

$$F = A <+> \left(B \wedge C \vee \overline{B} \wedge \overline{C}\right) <+> A \wedge \overline{B \wedge \overline{C} \vee \overline{B} \wedge C}$$
$$G = A \vee \left(B <-> C\right)$$

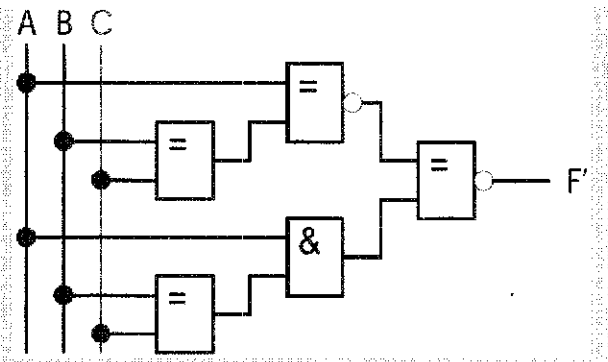
a) Stellen Sie durch algebraische Umformungen der Funktion F die Gleichheit zwischen den gegebenen Funktionen F und G fest. Wenden Sie pro Umformungsschritt jeweils nur eine Regel bzw. Formel an.

$$\begin{aligned} F &= A <+> \left(B \wedge C \vee \overline{B} \wedge \overline{C}\right) <+> A \wedge \overline{B \wedge \overline{C} \vee \overline{B} \wedge C} \\ &= A <+> \left(B \wedge C \vee \overline{B} \wedge \overline{C}\right) <+> A \wedge \overline{B <+> C} \\ &= A <+> \left(B \wedge C \vee \overline{B} \wedge \overline{C}\right) <+> A \wedge \left(B <-> C\right) \\ &= A <+> \left(B <-> C\right) <+> A \wedge \left(B <-> C\right) \\ &= A \vee \left(B <-> C\right) = G \end{aligned}$$

b) Wandeln Sie die Funktion F so um, daß sie ausschließlich aus Äquivalenz- und UND-Verknüpfungen sowie Negationen besteht. (Sie können mit einer bereits in Aufgabe 2 a) ermittelten Form der Funktion F beginnen.)

$$\begin{aligned} F &= A <+> \left(B <-> C\right) <+> A \wedge \left(B <-> C\right) \\ &= \overline{A <-> \left(B <-> C\right)} <+> A \wedge \left(B <-> C\right) \\ &= \overline{\overline{A <-> \left(B <-> C\right)} <-> A \wedge \left(B <-> C\right)} \end{aligned}$$

c) Zeichnen Sie die von Ihnen in Aufgabe 2 b) gewonnene Schaltfunktion F.



Zu Aufgabe 3: **(20 Punkte)**

Gegeben sei die Realisierung eines synchronen 4-bit-Dualzählers aus JK-Master-Slave-Flipflops und logischen Grundschaltungen (UND, ODER, NICHT, Antivalenz,...). Dieser Zähler soll so erweitert werden, daß er in Abhängigkeit von zwei Steuersignalen S1, S0 die folgenden Funktionen erfüllt:

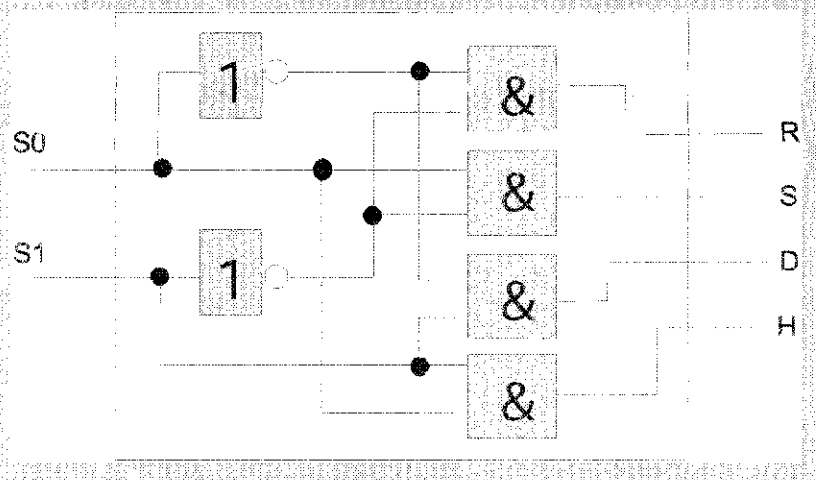
S1	S0	AS	Funktion
0	0	R	Rücksetzen auf den Zustand ‚0' (0000)
0	1	S	Speichern des Zustands, keine Zustandsänderung
1	0	D	Dezimalzähler 0,...,9 (0000,...,1001)
1	1	H	Dualzähler 0,...,F (0000,...,1111)

Dabei sollen die folgenden Restriktionen eingehalten werden:

- die JK-Flipflops verfügen über keine statischen Setz/Rücksetzeingänge,
- der Takt wird ohne jede Veränderung direkt auf die Takteingänge der Flipflops gelegt,
- beim Wechsel vom Dualzählermodus in den Dezimalzählermodus soll in den Zustand ‚0' (0000) gesprungen werden, wenn der Zähler vor dem Wechsel keine Dezimalzahl ausgegeben hat, also in einem der Zustände A,...,F (1010,...,1111) war. Sonst soll der folgende Dezimalwert ausgegeben werden.

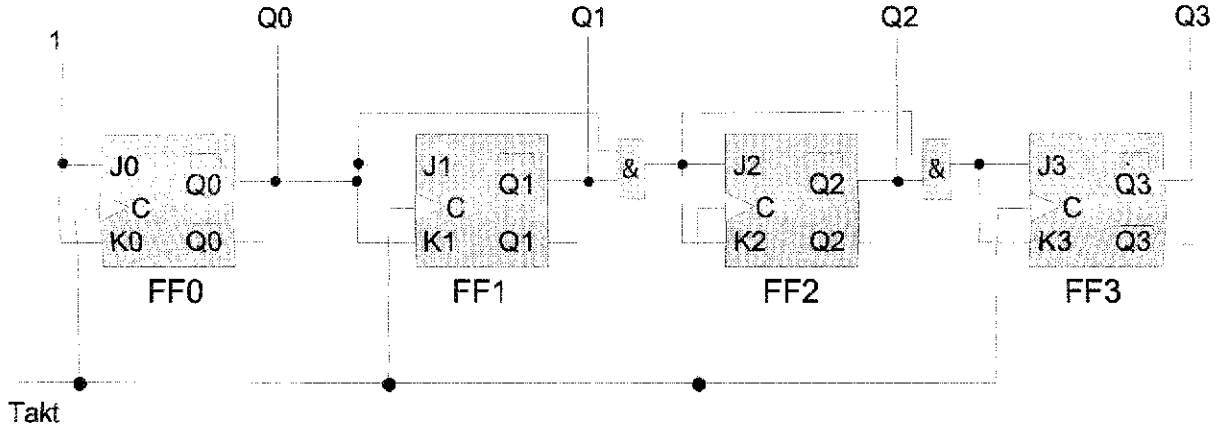
a) Geben Sie zunächst eine Schaltung zur Decodierung der Steuereingänge in die vier in der oben stehenden Tabelle angegebenen Auswahlsignale (R, S, D, H) an.

Es handelt sich um einen einfachen 'Adreßdecoder', der - in Abhängigkeit von S1, S0 - eine 1-aus-4-Auswahl trifft.

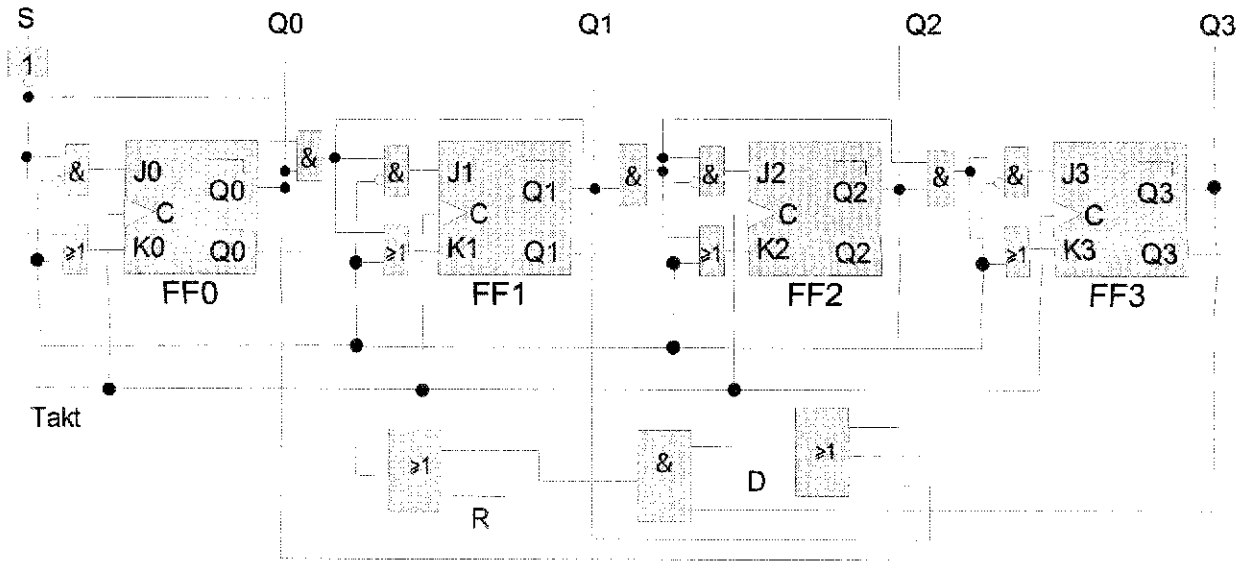


b) Ergänzen Sie die folgende Schaltung zu einem synchronen 4-bit-Dualzähler.

Die Zustandstabelle zeigt, daß ein Flipflop genau dann seinen Zustand ändert, wenn alle links von ihm liegenden Flipflopausgänge Qi den Wert '1' haben. Dies wird durch die UND-Gatter abgefragt. J=K=1 führt beim JK-Flipflop zu der gewünschten Zustandsänderung.



c) Ergänzen Sie Ihre Schaltung aus b) so, daß sie die geforderten zusätzlichen Funktionen (R, S, D) erfüllt. Verwenden Sie dazu die Signale aus Teil a). Begründen Sie Ihre Lösung.



Herleitungen:

Funktion R (Reset):

Das Rücksetzen aller Flipflops mit dem nächsten Takt (!) erreicht man, indem man für R=1 an die J-Eingänge eine '0' über ein UND-Gatter mit negiertem Eingang und in die K-Eingänge über ein ODER-Gatter eine '1' anlegt. (R wird über eine ODER-Schaltung zugeführt, da auch im Dezimalzählermodus die Flipflops gelöscht werden müssen, s.u. Das Oder-Gatter vor K0 von FF0 kann übrigens noch entfallen, wenn man S invertiert direkt mit K0 verbindet.)

Funktion S (Speichern):

Das Speichern des augenblicklichen Zustands eines Flipflops über das Signal S kann erreicht werden, wenn der Zustand seines 'linken' Nachbarn nicht mehr auf seine Eingänge J, K durchgeschaltet und stattdessen beide Eingänge auf den Wert J=K=0 gelegt werden. Im Bild ist dazu das negierte Signal S verwendet worden, das über ein UND-Gatter das Ausgangssignal Q0 des FF0 und über die folgende Kette von UND-Gattern auch alle anderen Ausgänge Qi sperrt.

Funktion D (Dezimalzähler):

Aus den Zuständen 9,...,F (1001,...,1111) muß in den Zustand '0' gesprungen werden. Diese Zustände werden durch die Funktion

$$Q3 \wedge (Q2 \vee Q1 \vee Q0)$$

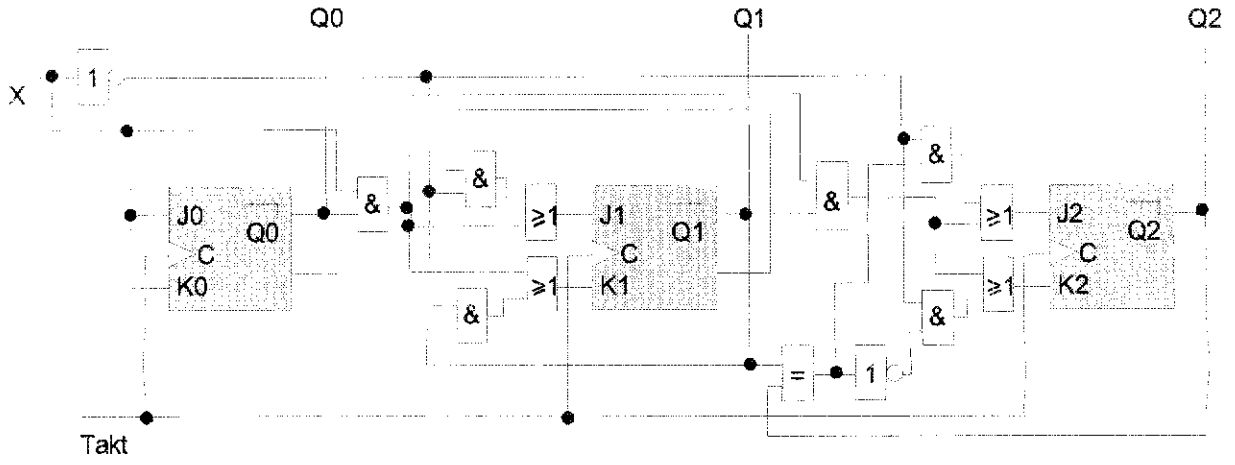
angezeigt. Diese Funktion wird durch das Dreifach-ODER und Dreifach-UND (an das auch das Auswahlsignal D gelegt ist) am unteren Rand der Zeichnung erzeugt. Ein Anliegen eines der genannten Zustände führt beim nächsten Taktsignal zum oben beschriebenen Rücksetzen aller Flipflops.

Funktion H ('Hexadezimalzähler'):

Dies ist die Grundfunktion des synchronen Dualzählers, für die keine besonderen Schaltungsmaßnahmen ergriffen werden müssen. (Natürlich sind auch Lösungen möglich, bei denen das Signal H explizit verwendet wird.)

Zu Aufgabe 4: (20 Punkte)

Gegeben sei das folgende synchrone Schaltwerk mit dem Eingangssignal X und dem Ausgangsvektor $\underline{Y}$ = (Q2, Q1, Q0).



a) Um welchen Typ eines Schaltwerks handelt es sich ? (Moore, Mealy, ...) Begründen Sie Ihre Antwort.

Es handelt sich um einen Moore-Automat, da die Ausgangsfunktion $\underline{Y}$ = (Q2, Q1, Q0) nur vom Zustand der Schaltung, nicht direkt vom Eingangssignal X abhängt.

b) Ermitteln Sie aus der Zeichnung die Ansteuerfunktionen für die Flipflops:

$J0 = X$, $K0 = J0 = X$

$J1 = X \wedge Q0 \vee \overline{X} \wedge \overline{Q1}$, $K1 = X \wedge Q0 \vee \overline{X} \wedge Q1$

$J2 = X \wedge Q0 \wedge Q1 \vee \overline{X} \wedge (Q1 <-> Q2)$

$K2 = X \wedge Q0 \wedge Q1 \vee \overline{X} \wedge \overline{(Q1 <-> Q2)}$

c) Tragen Sie die Ansteuerfunktionen Ji, Ki in die Zustandstabelle auf dem folgenden Lösungsblatt ein und ermitteln Sie die Folgezustände $Q2^+$, $Q1^+$, $Q1^+$. s.u.

d) Ergänzen Sie den Zustandsgraphen auf dem Lösungblatt durch die Übergänge zwischen den Zuständen und den jeweiligen Wert des Ausgangsvektors $\underline{Y}$. s.u.

e) Beschreiben Sie kurz, welche Funktion die Schaltung ausführt.

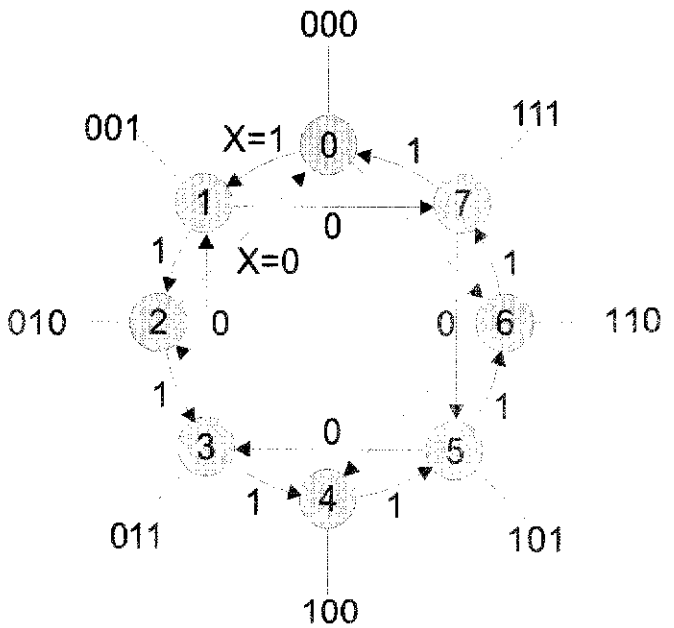
Die Schaltung stellt für X = 1 einen synchronen 3-bit-Dualzähler dar. Für X = 0 zählt sie jeweils in Zweier-Schritten rückwärts, also in den Zyklen 0-6-4-2-0... oder 7-5-3-1-7.... .

Lösungsblatt zu Aufgabe 4:

Zu c) Zustandstabelle

Nr.	X	Q2	Q1	Q0	$Q2^+$	$Q1^+$	$Q0^+$	J2	K2	J1	K1	J0	K0
0	0	0	0	0	1	1	0	1	0	1	0	0	0
1	0	0	0	1	1	1	1	1	0	1	0	0	0
2	0	0	1	0	0	0	0	0	1	0	1	0	0
3	0	0	1	1	0	0	1	0	1	0	1	0	0
4	0	1	0	0	0	1	0	0	1	1	0	0	0
5	0	1	0	1	0	1	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1	0	0	1	0	0
7	0	1	1	1	1	0	1	1	0	0	1	0	0
8	1	0	0	0	0	0	1	0	0	0	0	1	1
9	1	0	0	1	0	1	0	0	0	1	1	1	1
10	1	0	1	0	0	1	1	0	0	0	0	1	1
11	1	0	1	1	1	0	0	1	1	1	1	1	1
12	1	1	0	0	1	0	1	0	0	0	0	1	1
13	1	1	0	1	1	1	0	0	0	1	1	1	1
14	1	1	1	0	1	1	1	0	0	0	0	1	1
15	1	1	1	1	0	0	0	1	1	1	1	1	1

Zu d) Zustandsgraph



Zu Aufgabe 5: (10 Punkte)

Welche der folgenden Aussagen sind richtig ?

	ja	nein
a) Ein (semi-)synchroner Systembus verlangt von allen angeschlossenen Komponenten die Einhaltung derselben strengen Anforderungen an ihre Zugriffsgeschwindigkeit. (vgl. READY-Signal)	☐	X
b) Die Assemblerbefehle ‚logisches Linksschieben' (LSL) und ‚arithmetisches Linksschieben' (ASL) können in denselben Maschinenbefehl übersetzt werden. *Beide Befehle bewirken dieselbe Operation des Rechenwerks.*	X	☐
c) Ein Registersatz unterscheidet sich von einem kleinen Schreib-Lesespeicher (RAM) nur dadurch, daß er mit auf dem Prozessorchip integriert ist. Viele Sonderfunktionen möglich: Rücksetzen, Mehrfachzugriff, Zählen, gleichzeitiger Schreib/Lesezugriff, etc.	☐	X

d) Ein **8-bit-Indexregister IR** verfüge über die Möglichkeiten der automatischen Modifikation ‚autoinkrement / autodekrement' und Skalierung. Welcher Wert W wird zur Adreßberechnung verwendet, wenn IR=$1E, Skalierungsfaktor m = 8 und predekrement mit n = 2 gewählt wird ?

W = ($1E-2)*8 = $E0 ($1C 3 mal linksschieben)

e) Das **IEEE-754-Format** zur Berechnung von Gleitpunktzahlen unterscheidet sich hauptsächlich durch die beiden folgenden Eigenschaften von der ‚normalen' Darstellung von (binären) Gleitpunktzahlen durch Vorzeichen, Mantisse und Exponent:

1. *Anstelle des Exponenten wird die Charakteristik berechnet, die aus der Verschiebung des Exponenten in den positiven Zahlenbereich entsteht.*
2. *Die höchstwertige '1' der Mantisse wird vor den Dezimalpunkt verschoben und nicht dargestellt.*

f) Die Verwaltung großer Registersätze in Form von **überlappenden Registerbänken** bietet insbesondere die folgenden beiden Vorteile:

1. *Für die Parameterübergabe zwischen aufrufendem und aufgerufenem Programmteil müssen keine Registerinhalte transferiert werden.*
2. *Die Register in einer Bank können durch eine kurze relative Adresse selektiert werden.*

Zu Aufgabe 6: (20 Punkte)

a) i)

Die zwei vorzeichenbehafteten 8-bit-Zahlen A_8 = \$63, B_8 = \$A7 (im Zweierkomplement) sollen so in 16-bit-Zahlen A_{16}, B_{16} umgewandelt werden, daß sich ihr Wert nicht ändert. Geben Sie A_{16}, B_{16} an und begründen Sie Ihre Lösung !

A_{16} = \$0063

B_{16} = \$FFA7

Begründung:

Das Vorzeichen der Zahl muß erhalten bleiben. Im Zweierkomplement müssen dazu alle höherwertigen Bits mit dem Vorzeichenbit gefüllt werden (Sign Extension).

ii)

Stellen Sie die Zahl Z = - 464 im 32-bit-Gleitpunktformat des IEEE-754-Standards in hexadezimaler Form dar (Rechnung angeben !). Tragen Sie sie in den folgenden Zahlenrahmen ein und kennzeichnen Sie die einzelnen Bitfelder !

$$Z = (-1)^{VZ} * 2^{VZ*E} * 1.M_{10} = (-1)^{1} * 2^{+8} * 1.8125.....10$$

$$\Rightarrow Z_{32} = (-1)^{1} * 2^{127+8} * (1.110100.....)_2 = (-1)^{1} * 2^{128+7} * (1.110100.....)_2$$

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

VZ Charakteristik Mantisse

Z_{32} = \$ C3E8 0000

iii)

Stellen Sie die Zahl Z = - 464 im 64-bit-Gleitpunktformat des IEEE-754-Standards in hexadezimaler Form dar (Rechnung angeben !).

$$Z_{64} = (-1)^{1} * 2^{1023+8} * (1.110100.....)_2$$

$$= (-1)^{1} * 2^{1024+7} * (1.110100.....)_2$$

$$\Rightarrow Z_{64} = \$\ C07D\ 0000\ 0000\ 0000$$

b) Ein Mikroprozessor besitze einen 8-bit-Datenbus und einen 16-bit-Adreßbus. Adressen werden im Speicher im Little-Endian-Format abgelegt. Der Bezugspunkt für die Berechnung Programmzähler-relativer Adressen bei Sprung- und Verzweigungsbefehlen ist die Adresse des OpCodes des nachfolgenden Befehls. Geben Sie für die unten stehende Befehlsfolge jeweils den Wert des Programmzählers PC nach der Ausführung der Sprung- und Verzweigungsbefehle an. Gehen Sie dabei aus von der Registerbelegung:

R0 =$A3, R1 = $6F, R2 = $97, R3 =$ BF

und dem Speicherausschnitt[2]:

A460	A461	A462	A463	A464	A465	A466	A467	A468	A469	A46A	A46B	A46C	A46D	A46E	A46F
7F	30	AB	D0	34	FF	00	40	C1	5F	45	20	B6	ED	CC	32
				4.				2.				5.			

Befehlsfolge[3]:

Nr.	Adresse	Befehl	gesuchter PC	Begründung
1	$A400	JMP $A300	$A300	Neuer PC direkt als absolute Adresse im Befehl angegeben
2	$A403	JMP ($A467)	$C140	Indirekter Sprung. In den Speicherstellen $A467, $A468 steht die Adresse $C140
3	$A406	CMP R0, R1	$A408	Vergleichsergebnis hängt von Interpretation der Zahlen ab (vorzeichenbehaftet/vorzeichenlos)
4	$A408	BHI $5A (PC)	$A464	Vorzeichenlos: R0=$A3 > R1=$6F, daher wird BHI ausgeführt: $A40A + $5A
5	$A40A	CMP R0, R1	$A40C	Vergleichsergebnis hängt von Interpretation der Zahlen ab (vorzeichenbehaftet/vorzeichenlos)
6	$A40C	BLT ($5D(PC))	$B620	vorzeichenbehaftet: R0=$A3 < R1=$6F, BLT ausgeführt: ($A40E + $5D)=($A46B)
7	$A40E	CMP R2, R3	$A410	Vergleichsergebnis hängt von Interpretation der Zahlen ab (vorzeichenbehaftet/vorzeichenlos)
8	$A410	BSRGT $5E (PC)	$A412	Vorzeichenbehaftet: R2=$97 < R3=$BF, BSRGT nicht ausgeführt: PC=$A412
9	$A412	NOP	$A413	

[2] Zur Vereinfachung werden hexadezimale Adressen und Daten ohne ‚$' geschrieben.
[3] Das ‚Programm' ist für sich sinnlos. Es geht hier nur um die Adreßberechnungen.

c) Ein MMX-Rechenwerk stellt die parallele vorzeichenlose Vergleichsoperation ‚>' (größer als - *gt*, nach Bild 6.8-13 in KE6) sowie die bitweise, parallel ausgeführten logischen Operationen ∧ (und, PAND), ∨ (oder, POR), <+> (Antivalenz, PXOR) zur Verfügung. Die ebenfalls bitweise ausgeführte logische Operation -∧ (Inhibition, PANDN) invertiert zunächst den „linken" Operanden und bildet dann die Und-Verknüpfung mit dem „rechten" Operanden. Die Operationen (‚op') werden im Drei-Adreßformat auf den Inhalten der 64-bit-MMX-Register Rn ausgeführt:

Ri := Rj op Rk.

Durch den Zweiadreß-Befehl MOVQ Ri,<Datum> kann ein Register mit einem Speicherwort, einem Registerinhalt oder einem unmittelbaren Wert (#$....) geladen und insbesondere eine Kopie eines Registerinhalts angefertigt werden.

Geben Sie für die Anfangsbelegung der Register R0, R1 mit jeweils vier 16-bit-Wörtern

R0 = ($073F,$A0A0,$6324,$370A), R1 = ($A37F,$827A,$2F3E,$8520)

eine Folge von logischen Befehlen und MOV-Befehlen an, durch die in einem Register

- in den beiden höherwertigen (‚linken') 16-bit-Wörtern die Maxima,
- in den beiden niederwertigen (‚rechten') 16-bit-Wörtern Minima

von R0 und R1 abgelegt werden, wobei Maxima und Minima elementeweise über R0, R1 ermittelt werden.

Nr.	Befehl	Bemerkung
1	MOVQ R4, R1	Kopie von R1 anlegen
2	MOVQ R3, R0	Kopie von R0 anlegen
3	PCMPGTW R3, R4	R0, R1 wortweise auf größer vergleichen
4	MOVQ R2,#$00000000 FFFFFFFF	Maske zum Invertieren des Vergleichsergebnisses in den unteren beiden Wörtern
5	PXOR R3, R2	Vergleichsergebnis in unteren Stellen inv.
6	PAND R0, R3	In R0 in den höheren Wörtern die Maxima, in den niedrigen die Minima bestimmen
7	PANDN R3, R1	In R1 in den höheren Wörtern die Maxima, in den niedrigen die Minima bestimmen
8	POR R0, R3	Beide Register ver-'odern', Ergebnis in R0
9		
10		