Professor Dr.-Ing. Stefan Kowalewski
Dipl.-Inform. Philipp Kranen
Dipl.-Inform. Andreas Polzer

Aachen, 05. März 2008
SWS: V4/Ü2, ECTS: 7

Einführung in die Technische Informatik

WS 2007/2008

Bachelor-Klausur (TI)

Hinweise

Bitte sorgfältig durchlesen.

- Tragen Sie auf allen Blättern Ihren Namen und Ihre Matrikelnummer ein.
- Benutzen Sie den Platz auf den Aufgabenblättern und berücksichtigen Sie, dass auch auf den Rückseiten Aufgaben stehen.
- Falsche Antworten in Ankreuzteilen führen zu Punktabzug (jedoch nie zu negativen Punkten bei einer Aufgabe). Nicht-Beantwortung bzw. *weiß nicht*-Antworten führen nicht zu Punktabzug.
- Sollte Ihnen der Platz nicht ausreichen, so können Sie Papier von der Aufsicht bekommen.
- Jeder Punkt entspricht ungefähr einer Bearbeitungszeit von einer Minute. Die Bearbeitungszeit für die gesamte Klausur beträgt 120 Minuten.
- Schreiben Sie nur mit dokumentenechten Stiften wie z. B. Kugel- oder Tintenschreiber in blauer oder schwarzer Farbe. Lösungen mit Bleistift werden nicht bewertet.
- Legen Sie Ihren Studierendenausweis und einen amtlichen Lichtbildausweis bereit.
- Es sind **keine Hilfsmittel** erlaubt (außer Schreibzeug). Mit Ihrer Unterschrift versichern Sie Eides statt, dass die Prüfungsleistung von Ihnen ohne unzulässige fremde Hilfe erbracht worden ist.

Auswertung

Aufgabe	Punkte	Ergebnis
1	12,0	
2	12,0	
3	12,0	
4	10,0	
5	10,0	
6	10,0	
7	10,0	
8	10,0	
9	10,0	
10	12,0	
11	12,0	
Gesamt:	120,0	

Note: . . . .

Vorname, Name:	Matr.-Nr.:

Unterschrift

Aufgabe 1 : Zahlendarstellung (12 Punkte)

a) Konvertieren Sie bitte die folgenden Zahlen in das angegebene Zahlensystem. Folgendes Alphabet steht Ihnen zur Verfügung: $A = \{0, ..., 9, A, ..., F\}$. Die Elemente der Menge sind in aufsteigender Reihenfolge sortiert.

$(234)_{10} = (\quad\quad\quad)_2$ (1 Punkt)

$(124)_6 = (\quad\quad\quad)_2$ (2 Punkte)

$(3477)_8 = (\quad\quad\quad)_{16}$ (2 Punkte)

b) Berechnen Sie die folgenden Aufgaben durch binäre **Addition** mit 5 Bit zur Zahlendarstellung. (6 Punkte)

$7 + 3$:

+

$(13) + (-4)$ durch Verwendung des Einerkomplements:

+

$(-10) + (-7)$ durch Verwendung des Zweikomplements:

+

c) Interpretieren Sie die Ergebnisse aus Teil b). Beantworten Sie dabei die Frage warum bei einer Beschränkung auf fünf Bit nicht bei allen Berechnungen das Ergebnis stimmt! (1 Punkt)

Aufgabe 2 : Karnaugh-Diagramm (12 Punkte)

Minimieren Sie die folgende Funktion!

$$\begin{aligned} f(x_1, x_2, x_3, x_4, x_5) \quad = \quad & \overline{x}_1\overline{x}_2\overline{x}_3\overline{x}_4\overline{x}_5 + \\ & \overline{x}_2\overline{x}_3\overline{x}_4\overline{x}_5 + \\ & \overline{x}_2\overline{x}_3\overline{x}_4 x_5 + \\ & \overline{x}_1\overline{x}_2\overline{x}_4 + \\ & x_1 x_2 x_3 x_4 \overline{x}_5 + \\ & x_1\overline{x}_2\overline{x}_4\overline{x}_5 + \\ & x_2\overline{x}_3 x_4 x_5 + \\ & x_1\overline{x}_2\overline{x}_3 x_5 + \end{aligned}$$

Tragen Sie dazu in das folgende Karnaugh-Diagramm die entsprechenden Werte ein! (8 Punkte)

$x_5 = 0$:

x_3x_4 \ x_1x_2	00	01	11	10
00				
01				
11				
10				

$x_5 = 1$:

x_3x_4 \ x_1x_2	00	01	11	10
00				
01				
11				
10				

Bestimmen Sie das (eindeutige) Minimalpolynom für die Funktion $f(x_1, x_2, x_3, x_4, x_5)$! (4 Punkte)

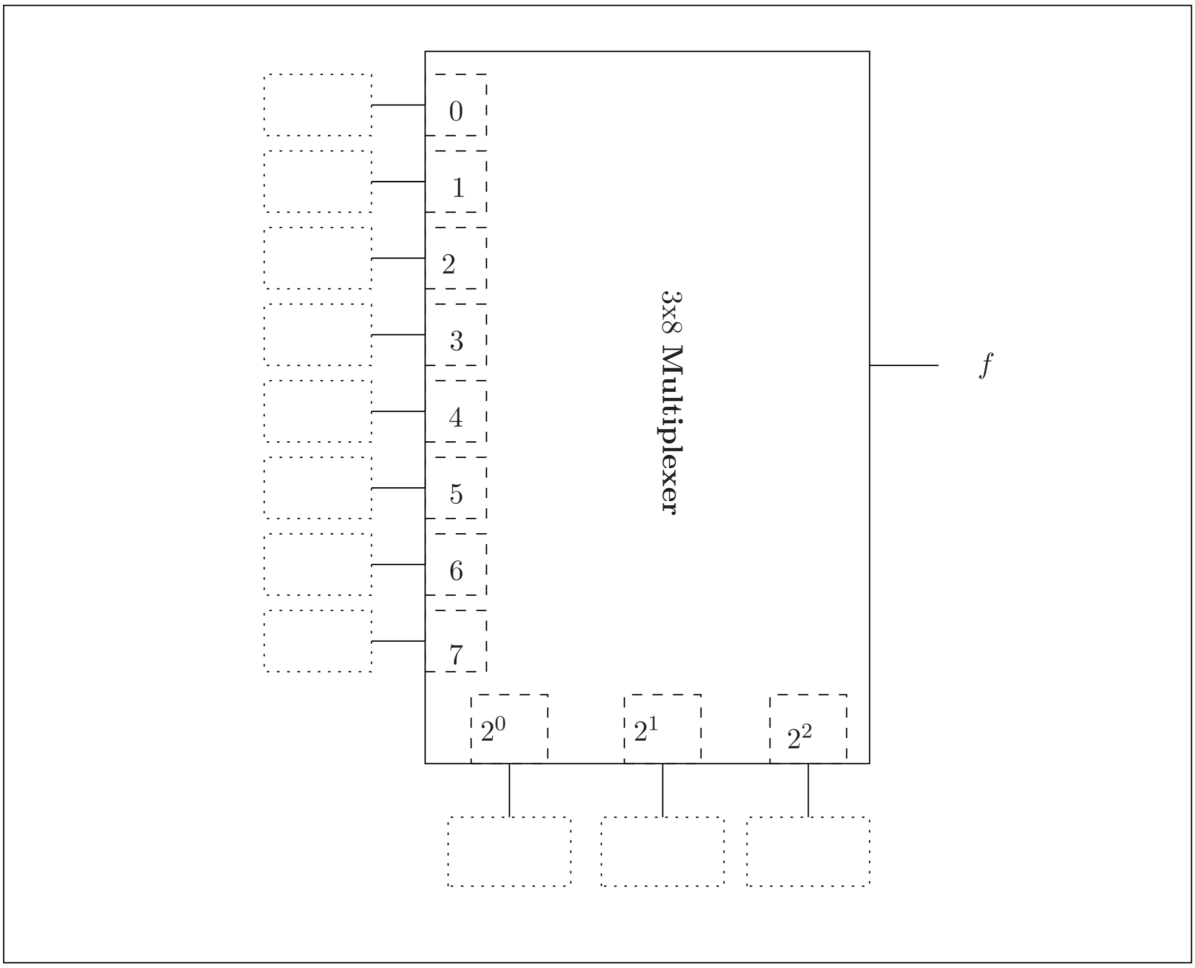
Aufgabe 3 : Multiplexer (12 Punkte)

a) Gegeben ist eine Boolesche Funktion $f(x_1, x_2, x_3, x_4)$, durch die folgende Wertetabelle:

x_1	x_2	x_3	x_4	f
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0

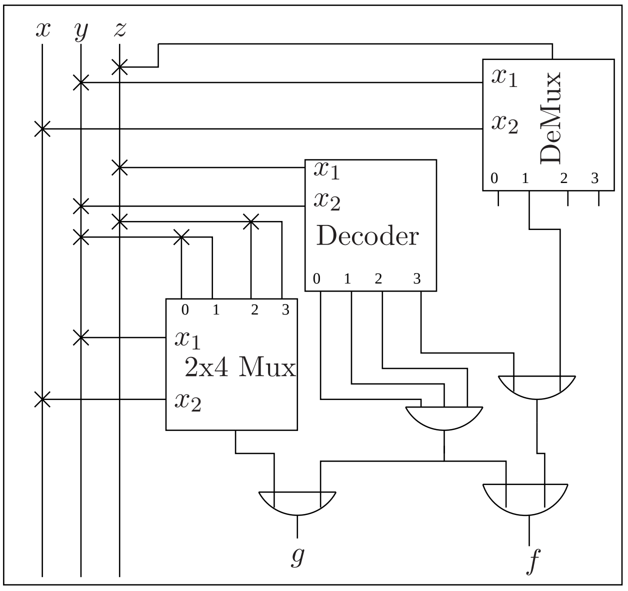
x_1	x_2	x_3	x_4	f
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Beschriften Sie die Eingänge des Multiplexers so, dass er die Funktion f realisiert! Benutzen Sie dazu die **gepunkteten** Boxen. (6 Punkte)



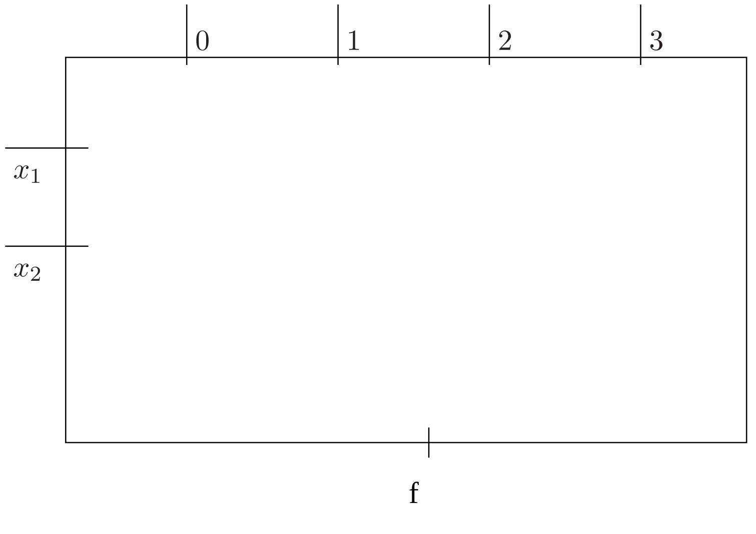
b) Gegeben ist der Decoder in der unten gezeigten Verschaltung. Vervollständigen Sie die Funktionstabellen für f und g! (4 Punkte)

Hinweis: Es gilt, dass die waagrechten Ein- und Ausgänge des **DeMux**, des **2x4 Mux** und des **Decoder** von links nach rechts aufsteigend beschaltet beschaltet sind. Die Steuereingänge sind als Dualzahl der Form $(x_1x_2)_2$ aufzufassen.



x	y	z	f	g
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

c) Skizzieren Sie die innere Schaltung eines 2x4 Multiplexers mit Hilfe von *UND*, *ODER* und Negations Bausteinen. Die Steuereingänge sind als Dualzahl der Form $(x_1x_2)_2$ aufzufassen. (2 Punkte)



Aufgabe 4 : Quine-McCluskey-Verfahren (10 Punkte)

a) Die folgende Tabelle enthält alle Minterme einer Booleschen Funktion $f : B^4 \to B$. Bestimmen Sie alle Implikanten, die sich in der 1. Iteration (und nur in der 1. Iteration) des Quine-McCluskey-Verfahrens ergeben, und tragen Sie diese in der zweiten Tabelle ein. (4 Punkte)

0. Iteration:

Gruppe	Implikant	Index	Minterm-Nummern
0	$x_1x_2x_3x_4$	1111	15
1	$\overline{x}_1x_2x_3x_4$ $x_1x_2x_3\overline{x}_4$	0111 1110	7 14
2	$\overline{x}_1\overline{x}_2x_3x_4$ $x_1\overline{x}_2\overline{x}_3x_4$ $x_1x_2\overline{x}_3\overline{x}_4$	0011 1001 1100	3 9 12
3			
4	$\overline{x}_1\overline{x}_2\overline{x}_3\overline{x}_4$	0000	0

1. Iteration:

Gruppe	Implikant	Index	Minterm-Nummern
0			
1			
2			
3			
4			

b) Angenommen folgende Iterationtabelle ergibt sich nach dem ersten Schritt. Stellen Sie dazu die entsprechende Primimplikationstabelle auf! (3 Punkte)
n. Iteration:

Gruppe	Implikant	Index	Minterm-Nummern
0	$x_2x_3x_4$	*111	7,15
1			
2	$x_1\overline{x}_3\overline{x}_4$	1*00	8,12
	$\overline{x}_2\overline{x}_4$	*0*0	0,2,8,10

Minterm Primimplikant							

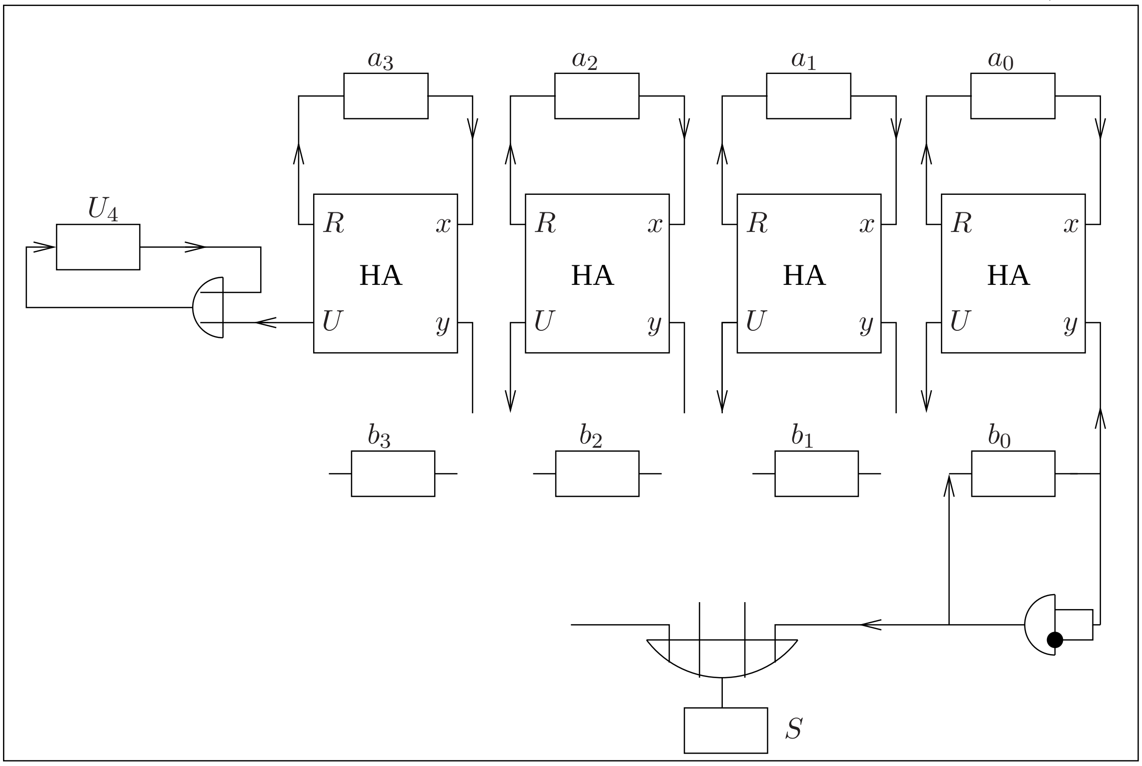
c) Angenommen, die untenstehende Implikationsmatrix ist während der Anwendung des Quine-McCluskey-Verfahrens auf eine vierstellige Boolesche Funktion f erstellt worden.

Minterm Primimplikant	1	3	5	6	7	8	9	11	13	14	15
$x_1\overline{x}_2\overline{x}_3$	0	0	0	0	0	1	1	0	0	0	0
$x_1x_2\overline{x}_3x_4$	0	0	0	0	0	0	0	0	1	0	0
$\overline{x}_1x_4$	1	1	1	0	1	0	0	0	0	0	0
x_4	1	1	1	0	1	0	1	1	1	1	1
x_2x_3	0	0	0	1	1	0	0	0	0	1	1
$x_2x_3\overline{x}_4$	0	0	0	1	0	0	0	0	0	1	0
$x_1x_2x_3$	0	0	0	1	0	0	0	0	0	1	1

Bestimmen Sie ein Minimalpolynom der Funktion f! (3 Punkte)

Aufgabe 5 : Von Neumann Addierwerk (10 Punkte)
Betrachten Sie im folgenden ein vier Bit von Neumann Addierwerk.

a) Zeichnen Sie die fehlende Verdrahtung in der folgenden Abbildung ein! (3 Punkte)



b) Welche Bedeutung hat der Baustein S in der Schaltung? (1 Punkt)

c) Demonstrieren Sie die Funktionsweise dieses Addierwerks anhand der Addition der beiden Zahlen 14 und 2, indem Sie Inhalte der Delays nach Ablauf des Taktes in die folgende Tabelle eintragen. Benutzen Sie den *Takt 1* um die Delays zu laden. (6 Punkte)

Takt	S	U_4	a_3	a_2	a_1	a_0	b_3	b_2	b_1	b_0
1										
2										
3										
4										
5										

Aufgabe 6 : Gleitkommadarstellung IEEE (10 Punkte)
In den folgenden Teilaufgaben werden Gleitkommazahlen nach IEEE 754 verwendet.

a) Konvertieren Sie die Zahl -25.3125 in Dezimaldarstellung in eine Gleitkommazahl gemäß IE-EE 754. Berechenen Sie dazu zuerst die entsprechende binäre Kommazahl in der Form:

$$[x/-](b_n \cdots b_1, b_{-1} \cdots b_{-m})$$ (2 Punkte)

Bringen Sie die binäre Kommazahl in eine normalisierte Darstellung der Form:

$$(1, b_{-1} \cdots b_{-m})_2 \cdot 2^{(e_1 \cdots e_n)_{10}}$$ (1 Punkt)

Tragen Sie das 32-Bit-IEEE-Gleitkomma-Ergebnis hier ein: (3 Punkte)

b) Quadrieren Sie die folgende 32-Bit-IEEE-Gleitkomma-Zahl: (4 Punkte)

1100000111010000000000000000000

Berechnen Sie dazu im folgenden die Mantisse, den Exponenten, und das Vorzeichen:

Tragen Sie das 32-Bit-IEEE-Gleitkomma-Ergebnis hier ein:

Aufgabe 7 : Disjunktive Normalform und Konjunktive Normalform (10 Punkte)

a) Die konjunktive Normalform (KNF) ist im Sinne des *Darstellungssatzes für Boolesche Funktionen* definiert als Konjunktion der Maxterme der nichteinschlägigen Indizes I der darzustellenden Funktion $f(x_1, ..., x_n)$. Weisen Sie die Korrektheit der auf diese Weise konstruierten KNF nach, indem Sie die möglichen Werte der Funktion f betrachten und mit Hilfe der Eigenschaften der Maxterme und der nichteinschlägigen Indizes argumentieren.

Hinweis: Sie können als bewiesen annehmen, dass der Maxterm $J = (j_1, \cdots, j_n)_2$ nur bei der Eingabe von $(j_1, \cdots, j_n)_2$ den Wert *Null* annimmt und bei alle anderen Eingaben den Wert *Eins*.

i) Zeigen Sie, dass die Konstruktion der KNF für den Fall $f(x_1 = j_1, ..., x_n = j_n) = 1$ mit der Eingabe $J = (j_1, ..., j_n)_2$ korrekt ist. (4 Punkte)

ii) Zeigen Sie, dass die Konstruktion der KNF für den Fall $f(x_1 = j_1, ..., x_n = j_n) = 0$ mit der Eingabe $J = (j_1, ..., j_n)_2$ korrekt ist. (4 Punkte)

b) Gegeben sei die sechsstellige Boolsche Funktion $h(x_1, x_2, x_3, x_4, x_5, x_6)$ durch ihre folgenden *nicht*einschlägigen Indizes 0, 10, 50. Stellen Sie die Funktion h **entweder** in DNF oder in KNF dar! (2 Punkte)

Aufgabe 8 : Funktionale Vollständigkeit (10 Punkte)

Sind die folgenden Mengen Boolescher Operatoren funktional vollständig? Begründen Sie Ihre Antwort! (3 Punkte je Frage)

a) $B = \{\rightarrow, 0\}$

ja ☐ nein ☐ weiß nicht ☐

b) $B = \{\nleftrightarrow, 0\}$

ja ☐ nein ☐ weiß nicht ☐

c) $B = \{f(x, y, z), g(x, y)\}$

ja ☐ nein ☐ weiß nicht ☐

x	y	z	$f(x, y, z)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1

x	y	z	$f(x.y.z)$
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

x	y	$g(x, y)$
0	0	1
0	1	1
1	0	1
1	1	0

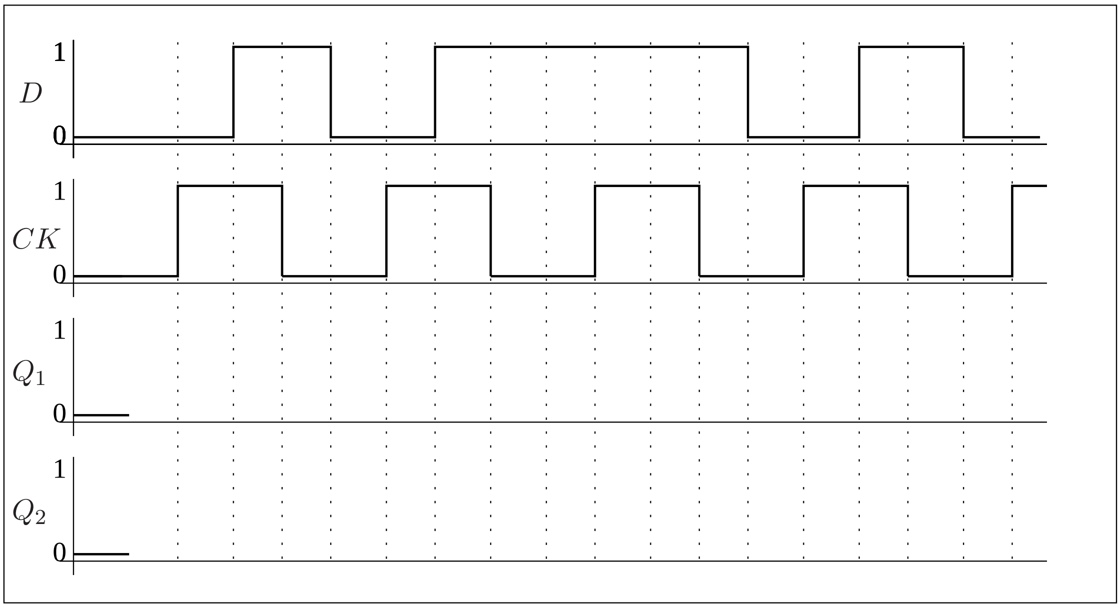
d) Aufgrund welchen Satzes genügt es zum Nachweis der *funktionalen Vollständigkeit* zu zeigen, dass die Funktionen $\{\wedge, \vee, \neg\}$ existieren? (1 Punkt)

		0	$\wedge$	$\nrightarrow$	x	$\nleftarrow$	y	$\nleftrightarrow$	$\vee$	$\downarrow$	$\leftrightarrow$	$\neg y$	$\leftarrow$	$\neg x$	$\rightarrow$	$\uparrow$	1
x	y	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

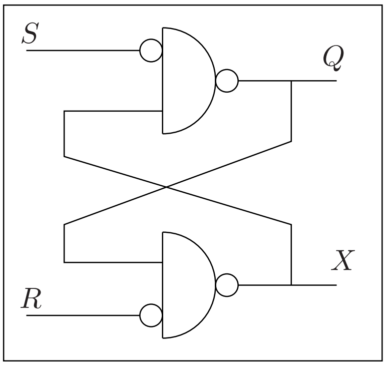
Aufgabe 9 : Latches (10 Punkte)

a) In den folgenden Abbildungen sind verschiedene Latches dargestellt.

Zeichnen Sie das Verhalten der Ausgänge Q der Latches i) bis ii) in das folgende Diagramm ein! (2 Punkte)

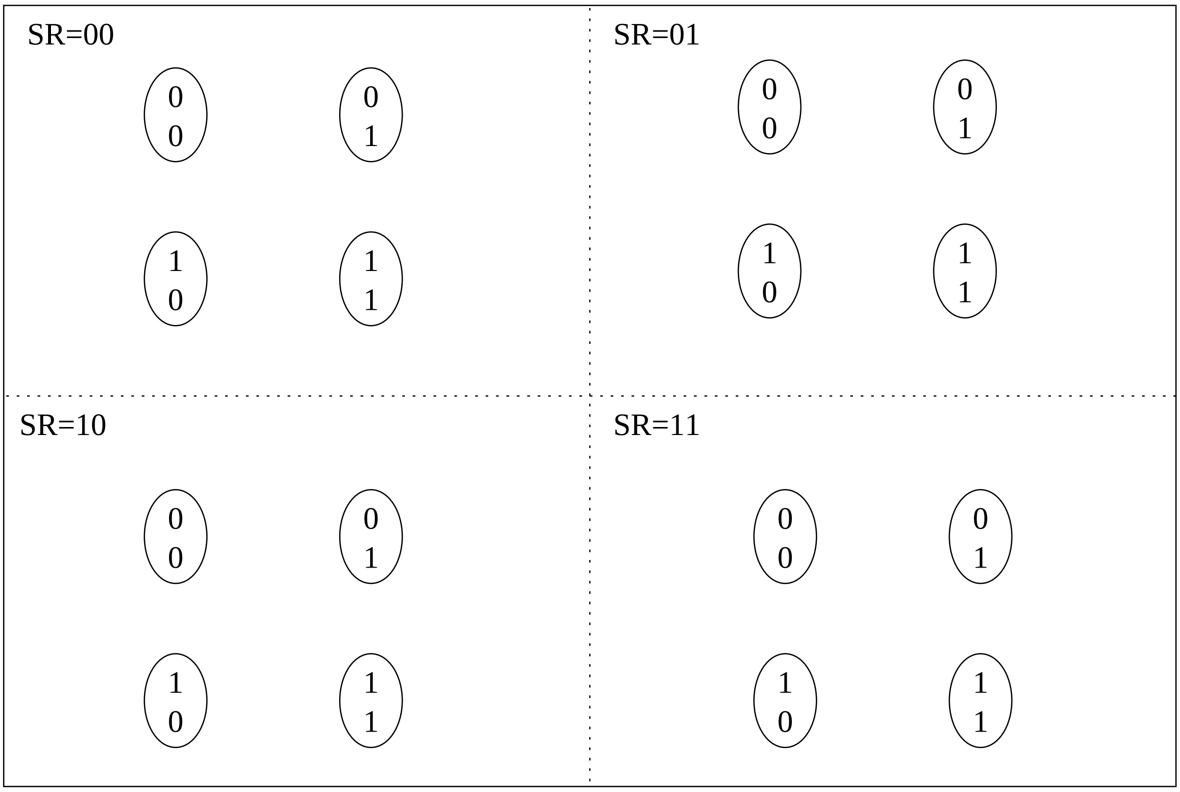


b) In der folgenden Abbildung ist ein *NAND* Latch gegeben.



In jedem der Graphen werden die vier möglichen Kombinationen an den beiden Ausgängen Q und X durch die vier Knoten $\binom{Q}{X}$ repräsentiert. Die Kanten repräsentieren, wie sich die Ausgänge durch genau einmaliges Schalten der beiden *NAND*-Gatter ändern. Vervollständigen Sie dafür die gegebenen Zustandsübergangdiagramme mit dem Kanten für die Übergänge.

(8 Punkte)



Aufgabe 10 : Schaltwerke (12 Punkte)

a) Entwickeln Sie einen Vier-Bit-Zähler in dem Sie die Funktionstabelle für die Booleschen Funktionen f_1, f_2, f_3, f_4 ausfüllen. Die Ergebnisse der Funktionen f_1, f_2, f_3, f_4 ergeben binär betrachtet die nächste Zahl. Beispiel: Der Zähler zählt von 0000 auf 0001. Das bedeutet für die Funktionen: $f_1(0,0,0,0) = 0$, $f_2(0,0,0,0) = 0$, $f_3(0,0,0,0) = 0$ und $F_4(0,0,0,0) = 1$.
Der Zähler soll folgende Ziffernfolge erzeugen:

0000 → 0001 → 0010 → 0100 → 1000 →
1100 → 1010 → 1001 → 0000 → ...

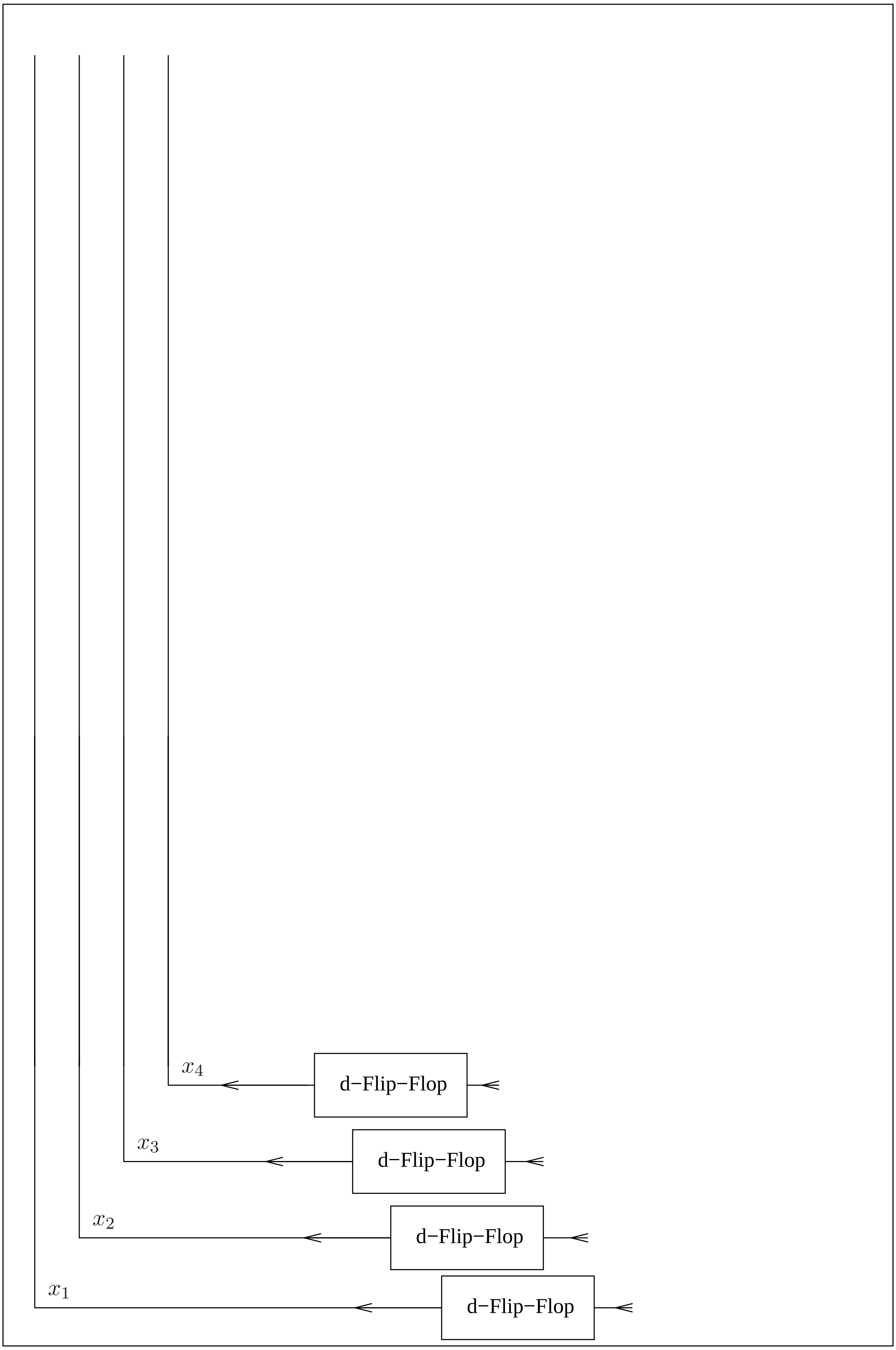
Füllen Sie nebenstehende Funktionstabelle aus! Nicht verwendete Funktionswerte können als beliebig angesehen werden.
(8 Punkte)

x_1	x_2	x_3	x_4	f_1	f_2	f_3	f_4
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0				
1	1	0	1				
1	1	1	0				
1	1	1	1				

b) Nehmen Sie an, nach einer Minimierung sind die folgenden Funktionen zur Realisierung des Zählers gegeben:

$$\begin{aligned}
f_1(x_1, x_2, x_3, x_4) &= x_3 + \overline{x}_1 x_2 + x_1 \overline{x}_2 \overline{x}_4 \\
f_2(x_1, x_2, x_3, x_4) &= x_1 x_2 + \overline{x}_1 x_4 + x_1 x_3 \\
f_3(x_1, x_2, x_3, x_4) &= x_1 x_3 + \overline{x}_1 \overline{x}_2 \overline{x}_3 \\
f_4(x_1, x_2, x_3, x_4) &= x_2 + x_1 \overline{x}_4
\end{aligned}$$

Stellen Sie die Funktionen f_1, f_2, f_3 und f_4 als Schaltung dar. Ergänzen Sie dazu in der folgenden Skizze die entsprechenden NEGATIONEN, UND- und ODER- Bausteine! Verbinden Sie die Ausgänge mit den d-Flip-Flop um die in a) beschriebene Zählerfunktionalität zu erhalten.
(6 Punkte)


x_4
d−Flip−Flop
x_3
d−Flip−Flop
x_2
d−Flip−Flop
x_1
d−Flip−Flop

Aufgabe 11 : MMIX (12 Punkte)

Bearbeiten Sie folgende Aufgaben zur Programmierung des MMIX. Beachten Sie, dass am Ende dieser Klausur eine Befehlsreferenz angehängt ist.

a) Folgenden ist ein MMIX Programm gegeben.

```
1              LOC  #100
2  x           GREG
3  y           GREG
4  i           GREG
5  h           GREG
6  Main        SET  x,1
7              SET  y,0
8              SET  i,10
9  lbl_1       BN   i,lbl_4
10             SUB  h,i,5
11             BN   h,lbl_2
12             ADD  y,y,i
13             JMP  lbl_3
14 lbl_2       MUL  x,x,i
15 lbl_3       SWYM % do nothing
16             SUB  i,i,1
17             JMP  lbl_1
18 lbl_4       SWYM % do nothing
19             TRAP 0,Halt,0
```

Welche Bedeutung hat die Einrückung der Befehle im MMIX Programm? (1 Punkt)

Welche Kontrollstruktur wurde durch die Zeilen 10 bis 15 realisiert? (2 Punkte)

Das Programm realisiert eine Schleife. Welche Zeile ist die erste und letzte Zeile der Schleife? (2 Punkte)

Um welche Art von Schleife handelt es sich? (1 Punkt)

b) Folgendes Programm ist in einer generischen Hochsprache gegeben. Schreiben Sie ein Programm für den MMIX, welches semantisch die gleiche Bedeutung hat und die Kontrollstrukturen umsetzt:

```
x = 1;
y = 20;

FOR i= 1 TO 20 STEP 2 DO
      y = y + x + i;
      x = i * 2;
ENDFOR
```

(Hinweis: `STEP 2` bedeutet ein Erhöhen von i jeweils um 2, also $1, 3, 5, \ldots$) Benutzen Sie nebenstehende Box zur Erstellung Ihres Programms. Es müssen nicht alle vorgegeben Zeilen genutzt werden. (6 Punkte)

```
% MMIX Program
          LOC #100
x         IS  $10
y         IS  $11
i         IS  $12
hilf      IS  $13
....      ..........................
....      ..........................
Main      ..........................
....      ..........................
....      ..........................
....      ..........................
....      ..........................
....      ..........................
....      ..........................
....      ..........................
....      ..........................
....      ..........................
....      ..........................
....      ..........................
....      ..........................
....      ..........................
....      ..........................
....      ..........................
....      TRAP 0,Halt,0
```

MMIX-Standardbefehle

*`U`: Unsigned, Register werden als unsigned ints betrachtet
*`I`: Immediate (Parameter $X,$Y,Z), Z direkt im Opcode angegeben, erkennt der Assembler automatisch

Arithmetische Befehle

`ADD[U][I] $X,$Y,$Z`	add (signed)	$X=$Y + $Z	rA
`SUB[U][I] $X,$Y,$Z`	subtract (signed)	$X=$Y - $Z	rA
`MUL[U][I] $X,$Y,$Z`	multiply (signed)	$X=$Y * $Z	rA (U: rH)
`DIV[U][I] $X,$Y,$Z`	divide (signed)	$X=int($Y/$Z)	rA,rR
`NEG[U][I] $X,[Y,]$Z`	negate (signed)	$X=[Y]-$Z	rA
`S[R|L][U][I]`	shift right/left	$X=$Y>>$Z, $X=$Y<<$Z	rA
`FADD/FSUB/FMUL/FDIV`	float operations	$X = $Y op $Z	rA
`FSQRT $X, 0, $Z`	float square root	$\$X = \sqrt{\$Z}$	rA
`FINT $X, 0, $Z`	floating integerize	$X = int($Z)	rA

Logik

`[N]AND[I] $X,$Y,$Z`	bitwise AND/NAND	$X = $Y [N]AND $Z
`[N]OR[I] $X,$Y,$Z`	bitwise OR/NOR	$X=$Y [N]OR $Z
`[N]XOR[I] $X,$Y,$Z`	bitwise XOR/NXOR	$X = $Y [N]XOR $Z

Laden und Speichern

`GET $X,rx`	get from special register	$X=rx
`GETA $X, Var`	get adress of variable	$X=adr(Var)
`PUT rx, $X`	write to special register	rx = $X
`SET $X,$Y`	set register	$X = $Y
`SET $X,Y`	set register	$X = Y
`SET[L|ML|MH|H] $X, YZ`	set low/med low/med high/high wyde	$\$X = YZ*2^{0,16,32,48}$
`INC[L|ML|MH|H] $X, YZ`	increase by wyde	$\$X = \$X + YZ*2^{0,16,32,48}$
`LD[B|W|T|O][U][I] $X,$Y,$Z`	load byte / wyde / tetra / octa	$X = <$Y+$Z>
`ST[B|W|T|O][U][I] $X,$Y,$Z`	store byte / wyde / tetra / octa	<$Y+$Z> = $X

Vergleiche, Sprünge, Prozeduren

`[P]B[N|Z|P] $X, marke/adresse`	[probable] branch if negative / zero / positive	$X<0/$X==0/$X>0 ⇒ marke
`[P]BN[N|Z|P] $X, marke/adresse`	[probable] branch if not neg. / zero / positive	$X≥0/$X≠ 0/$X≤ 0 ⇒ marke
`[P]B[OD|EV] $X, marke/adresse`	[probable] branch if odd / even	$X mod 2 = 1/0 ⇒ marke
`CMP[U][I] $X,$Y,$Z`	compare	$X=$Y<=>$Z
`FCMP[E] $X,$Y,$Z`	floating compare (with respect to ε)	$X=$Y<=>$Z
`JMP marke/adresse`	jump	
`GO $X,$Y,$Z`	go to location($Y,$Z)	$X = address of next instruction
`PUSHJ $X,marke/adresse`	push(X) and jump	rJ = address of next instruction
`PUSHGO $X,$Y,$Z`	push(X) and go to location($Y,$Z)	rJ = address of next instruction
`POP X,YZ`	pop(X) and jump to rJ + 4 * YZ	

Systemaufrufe

`TRAP 0,Halt,0`	end program		rBB/WW/XX/YY/ZZ
`TRAP 0,Fgets,StdIn`	get from stdin	buf←stdin (vorher: $255=adr(buf),size)	rBB/WW/XX/YY/ZZ
`TRAP 0,Fputs,StdOut`	write on stdout	buf→stdout (vorher: $255=adr(buf))	rBB/WW/XX/YY/ZZ