RWTH RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

LEHRSTUHL INFORMATIK XI SOFTWARE FÜR EINGEBETTETE SYSTEME

Professor Dr.-Ing. Stefan Kowalewski
Dipl.-Inform. Andreas Polzer
Dipl.-Inform. Ralf Mitsching

Aachen, 07. März 2007
SWS: V2/Ü2, ECTS: 4

Einführung in die Technische Informatik

WS 2006/2007

Bachelor-Klausur

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	6,0	
2	5,0	
3	6,0	
4	5,0	
5	11,0	
6	4,0	
7	6,0	
8	8,0	
9	13,0	
10	7,0	
11	10,0	
12	5,0	
13	11,0	
14	8,0	
15	5,0	
16	10,0	
Gesamt:	120,0	

Note: ....

Vorname, Name:	Matr.-Nr.:

____
Unterschrift

Aufgabe 1 : Zahlensysteme (6 Punkte)

Konvertieren Sie bitte die folgenden Zahlen in das angegebene Zahlensystem. Folgendes Alphabet steht Ihnen zur Verfügung: $A = \{1, ..., 9, A, ..., F\}$. Die Elemente der Menge sind in aufsteigender Reihenfolge sortiert.

a) $(280)_{10} = (\quad)_2$ (2 Punkte)

b) $(1220430)_5 = (\quad)_{25}$ (2 Punkte)

c) $(10010110)_2 = (\quad)_8$ (2 Punkte)

Aufgabe 2 : Dualdarstellung im Rechner (5 Punkte)

Interpretieren Sie das Byte

$\boxed{1\ 1\ 0\ 1\ 1\ 1\ 0\ 0}$

a) als vorzeichenlose Dualzahl und konvertieren Sie in das Dezimalsystem.

b) als vorzeichenlose Dualzahl und konvertieren Sie in das **Hexadezimalsystem**:

c) als Dualzahl in Vorzeichen-/Betrag-Darstellung und konvertieren Sie in das Dezimalsystem.

d) als Dualzahl im Einerkomplement und konvertieren Sie in das Dezimalsystem.

e) als Dualzahl im Zweierkomplement und konvertieren Sie in das Dezimalsystem.

Aufgabe 3 : Funktionale Vollständigkeit (6 Punkte)

Beantworten Sie folgende Fragen mit Begründung!

a) Wann ist ein System $B = \{f_1, ..., f_n\}$ Boolescher Funktionen funktional vollständig?

(2 Punkte)

b) Ist das System $\{f_1(x, y), f_2(x), f_3(x, y, z)\}$ Boolescher Funktionen funktional vollständig?

(4 Punkte)

Folgenden Definitionen der Booleschen Funktionen sind gegeben:

$f_1(x, y)$:

x	y	f_1
0	0	1
0	1	1
1	0	0
1	1	0

$f_2(x)$:

x	f_2
0	0
1	0

$f_3(x, y, z)$:

x	y	z	f_3
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Begründen Sie Ihre Antwort!

Aufgabe 4 : Boolesche Funktionen (5 Punkte)

Gegeben sind die folgenden einschlägigen Indizes 0, 3, 5, 7 einer Booleschen Funktion $f : B^3 \to B$. Kreuzen Sie in der nebenstehenden Liste genau die Funktionen an, die mit der gegebenen Funktion $f(x_1, x_2, x_3)$ übereinstimmen. D. h. die Funktionen $f_i(x_1, x_2, x_3)$, für die gilt:

$$f_i(x_1, x_2, x_3) = f(x_1, x_2, x_3), \forall i \in \{1, ..., 5\}.$$

a) $f_1(x_1, x_2, x_3) = \overline{x}_1\overline{x}_2\overline{x}_3 + (\overline{x}_1 x_2 x_3) + x_1\overline{x}_2 x_3 + x_1 x_2 x_3$ — ja ☐ nein ☐ weiß nicht ☐

b) $f_2(x_1, x_2, x_3) = x_1 x_2 x_3 + x_1\overline{x}_2\overline{x}_3 + \overline{x}_1 x_2\overline{x}_3 + \overline{x}_1\overline{x}_2\overline{x}_3$ — ja ☐ nein ☐ weiß nicht ☐

c) $f_3(x_1, x_2, x_3) = \overline{x}_1\overline{x}_2\overline{x}_3 + x_3$ — ja ☐ nein ☐ weiß nicht ☐

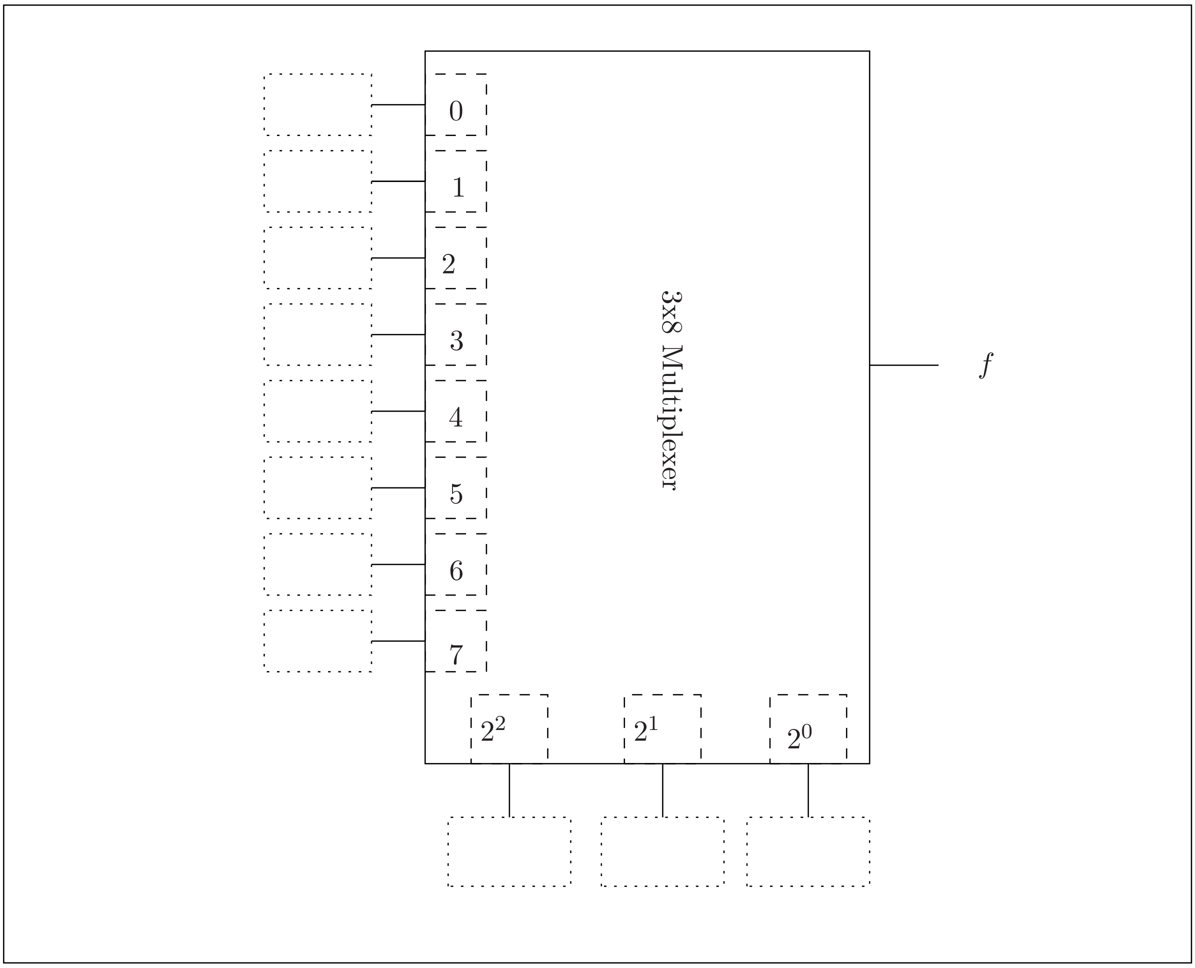
d) $f_4(x_1, x_2, x_3) = (x_1 + x_2 + x_3) + (x_1 \to x_2)x_3$ — ja ☐ nein ☐ weiß nicht ☐

e) $f_5(x_1, x_2, x_3) = ((x_1 + x_2 + x_3) \to x_1 x_3) + \overline{x}_1 x_2 x_3$ — ja ☐ nein ☐ weiß nicht ☐

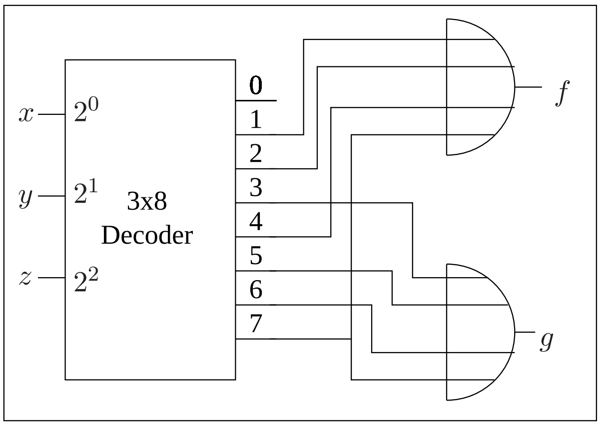
Aufgabe 5 : Multiplexer (11 Punkte)

a) Gegeben ist eine Boolesche Funktion $f(x_1, x_2, x_3, x_4)$, die genau dann eins ist, wenn die Dualzahl x_1, x_2, x_3, x_4 eine Primzahl darstellt. Beispiel: $f(0, 0, 1, 1) = 1$, da gilt $(0011)_2 = (3)_{10}$ und 3 eine Primzahl ist.

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)



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		

Welcher aus der Vorlesung bekannte Baustein wird mit dieser Schaltung realisiert? (1 Punkt)

Aufgabe 6 : Karnaugh-Diagramm (4 Punkte)

Sei $f : B^4 \rightarrow B$ die Boolesche Funktion mit den einschlägigen Indizes 0, 2, 4, 8, 10, 12.

Minimieren Sie f mit dem folgenden Karnaugh-Diagramm als Hilfsmittel. (3 Punkte)

		x_1x_2			
		00	01	11	10
	00				
x_3x_4	01				
	11				
	10				

Bestimmen Sie das (eindeutige) Minimalpolynom! (1 Punkt)

Aufgabe 7 : Karnaugh-Diagramm (6 Punkte)

Minimieren Sie die folgende Funktion!

$$
\begin{aligned}
f(x_1, x_2, x_3, x_4) \quad = \quad & \overline{x}_3\overline{x}_4 + \\
& x_3\overline{x}_4 + \\
& x_1x_2x_3x_4 + \\
& \overline{x}_1\overline{x}_2x_4 + \\
& \overline{x}_1\overline{x}_3x_4
\end{aligned}
$$

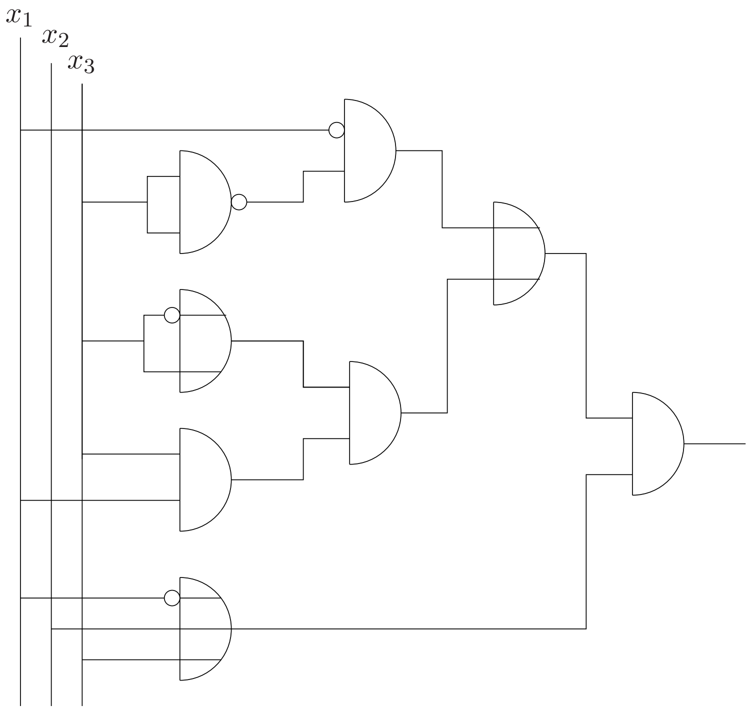
Tragen Sie dazu in das folgende Karnaugh-Diagramm die entsprechenden Werte ein! (4 Punkte)

		x_1x_2			
		00	01	11	10
x_3x_4	00				
	01				
	11				
	10				

Bestimmen Sie das (eindeutige) Minimalpolynom für die Funktion $f(x_1, x_2, x_3, x_4)$! (2 Punkte)

Aufgabe 8 : Disjunktive Normalform und Konjunktive Normalform (8 Punkte)

Gegeben ist die Boolesche Funktion $f(x_1, x_2, x_3)$ durch das folgende Schaltbild:



a) Geben Sie die Boolesche Funktion f in disjunktiver Normalform (DNF) an. (4 Punkte)

b) Geben Sie die Boolesche Funktion f in konjunktiver Normalform (KNF) an. (4 Punkte)

Aufgabe 9 : Quine-McCluskey-Verfahren (13 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. (5 Punkte)

0. Iteration:

Gruppe	Implikant	Index	Minterm-Nummern
0	$x_1x_2x_3x_4$	1111	15
1	$x_1\overline{x}_2x_3x_4$	1011	11
2	$\overline{x}_1\overline{x}_2x_3x_4$ $\overline{x}_1x_2x_3\overline{x}_4$ $x_1x_2\overline{x}_3\overline{x}_4$	0011 0110 1100	3 6 12
3	$\overline{x}_1\overline{x}_2\overline{x}_3x_4$ $\overline{x}_1\overline{x}_2x_3\overline{x}_4$	0001 0010	1 2

1. Iteration:

Gruppe	Implikant	Index	Minterm-Nummern
0			
1			
2			
4			

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

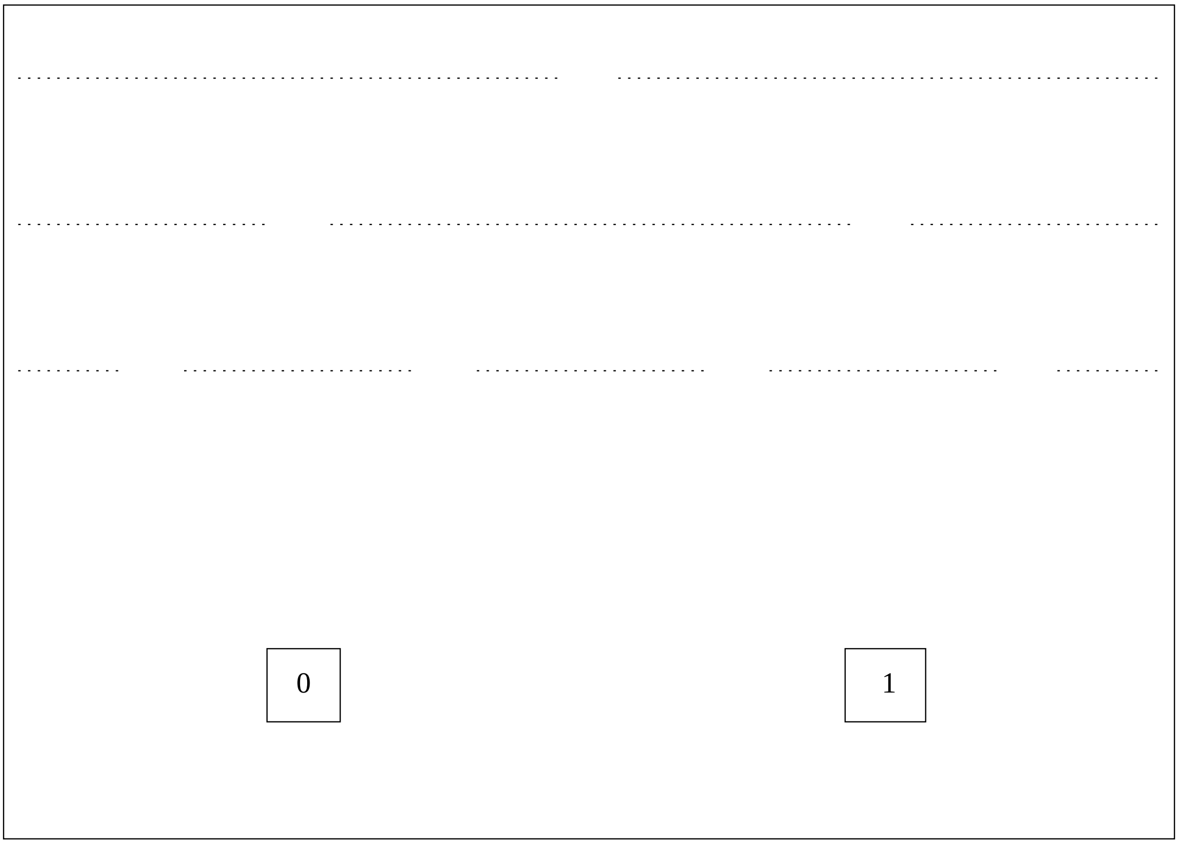
Minterm / Primimplikant	4	9	10	11	13	14	15	23	24	25	26	27	28	30
$\overline{x}_1x_2x_5$	0	1	0	1	1	0	1	0	0	0	0	0	0	0
$x_1x_2\overline{x}_5$	0	0	0	0	0	0	0	0	1	0	1	0	1	1
$x_2\overline{x}_3x_5$	0	1	0	1	0	0	0	0	0	1	0	1	0	0
$x_1x_2\overline{x}_3x_5$	0	0	0	0	0	0	0	0	0	1	0	1	0	0
$x_1\overline{x}_2x_3x_4x_5$	0	0	0	0	0	0	0	1	0	0	0	0	0	0
$x_1x_2\overline{x}_3$	0	0	0	0	0	0	0	0	1	1	1	1	0	0
$x_1x_2\overline{x}_3\overline{x}_5$	0	0	0	0	0	0	0	0	1	0	1	0	0	0
$\overline{x}_1x_2x_4$	0	0	1	1	0	1	1	0	0	0	0	0	0	0
$\overline{x}_1\overline{x}_2x_3\overline{x}_4\overline{x}_5$	1	0	0	0	0	0	0	0	0	0	0	0	0	0
$x_2x_4\overline{x}_5$	0	0	1	0	0	1	0	0	0	0	1	0	0	1

Bestimmen Sie das Minimalpolynom der Funktion f! (6 Punkte)

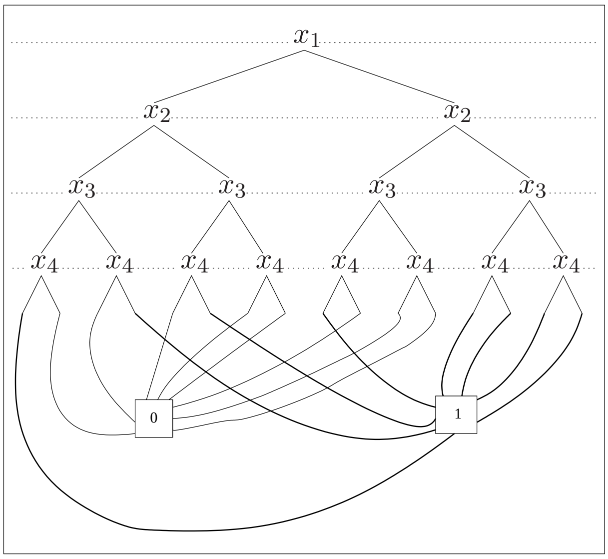
Ist das Minimalpolynom eindeutig? Begründen Sie Ihre Antwort! (2 Punkte)

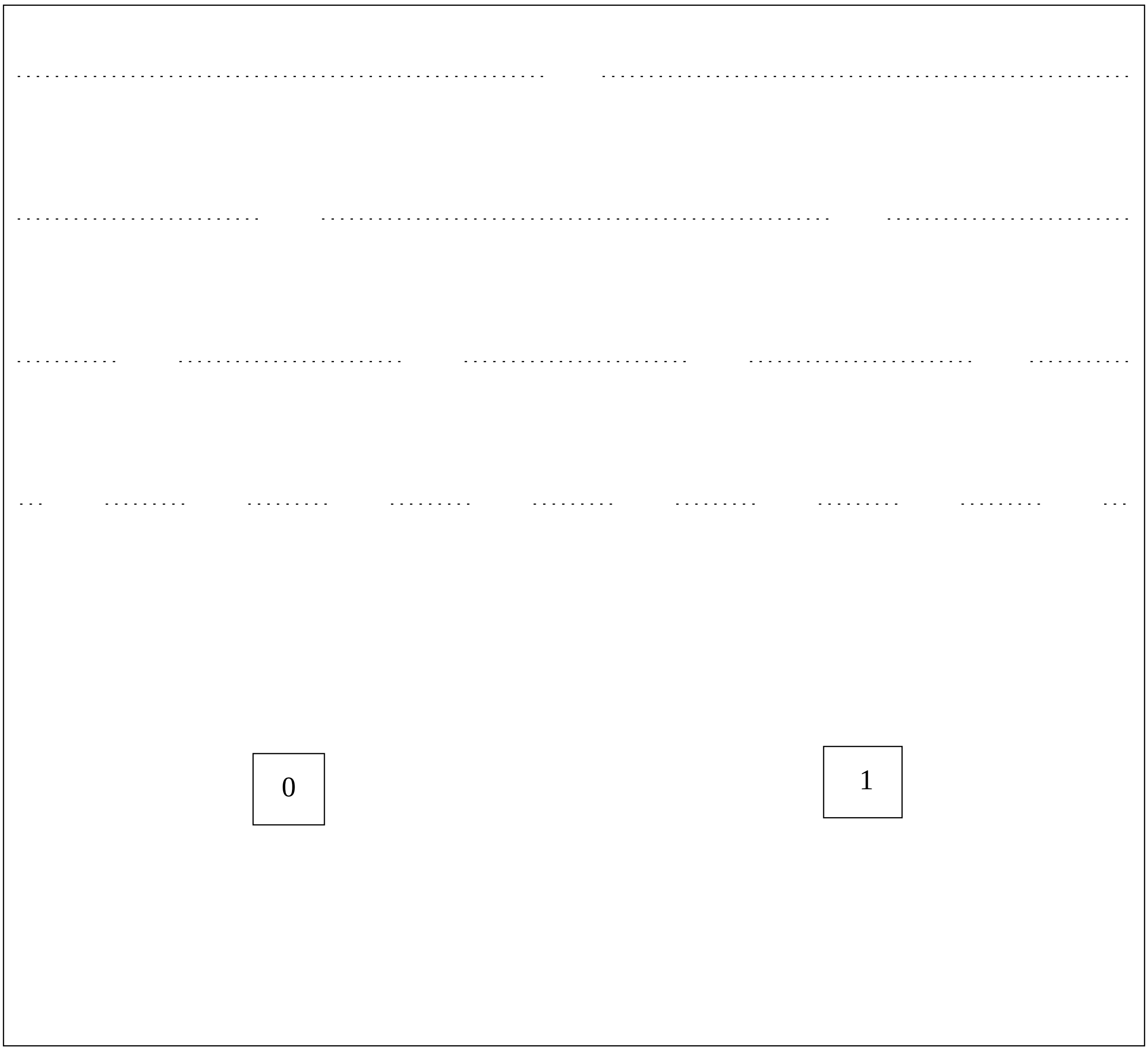
Aufgabe 10 : Ordered Binary Decision Diagram (7 Punkte)

a) Gegeben ist die dreistellige Boolesche Funktion $f(x_1, x_2, x_3)$ durch ihre **nicht**einschlägigen Indizes 0, 1, 2, 4, 6. Stellen Sie im folgenden Bild, in der Variablenreihenfolge $x_1 < x_2 < x_3$, die Funktion f als OBDD dar! Als Konvention gilt, dass die **linke** Kante ein **Null**-Kante ist und die **rechte** Kante eine **Eins**-Kante ist. Verwenden Sie auf jeden Fall die Hilfslinien als Anhaltspunkte für die Ebenen und die entsprechenden Lücken für die entsprechenden Variablen. Die unterste Ebene ist entsprechend der Vorgaben schon gemäß der Verschmelzungsregel zu einem **Null**-Knoten und einem **Eins**-Knoten zusammengefasst. (3 Punkte)



b) Minimieren Sie das nebenstehende OBDD gemäß des Algorithmus aus der Vorlesung! Wenden Sie den Algorithmus aber nur auf die zwei untersten Schichten an (Schicht mit `Null`- und `Eins`-Knoten und Variablenschicht x_4)! Es gelten die gleichen Konventionen wie in a). Zusätzlich soll zuerst die Eliminationsregel von links nach rechts und anschließend die Verschmelzungsregel von links nach rechts angewendet werden. (4 Punkte)

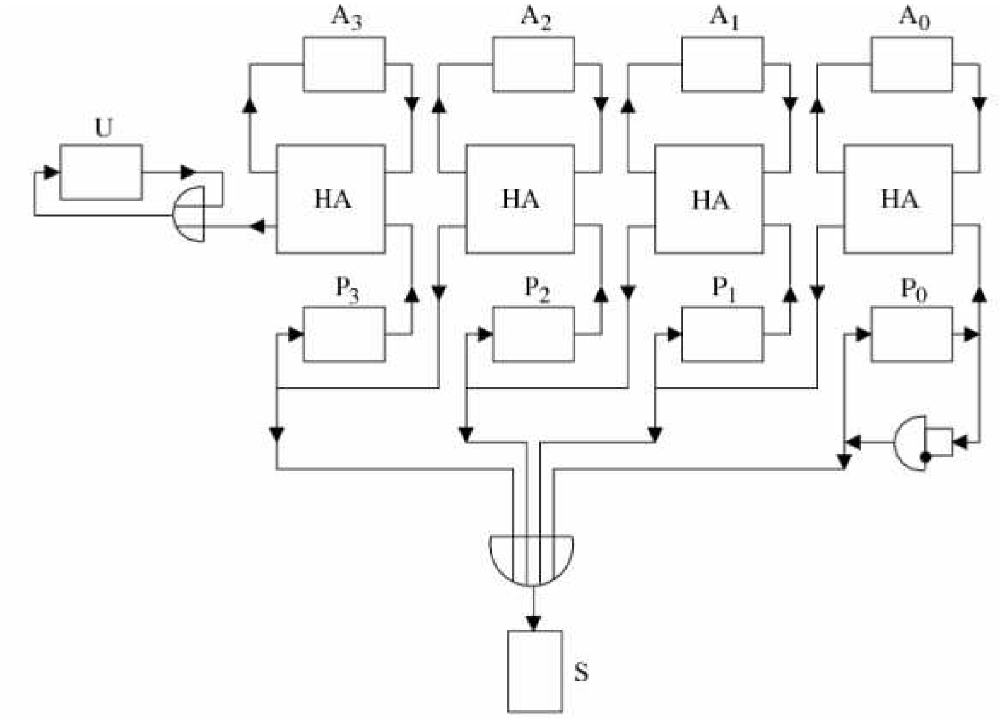




Aufgabe 11 : Addierwerke (10 Punkte)

1. Beantworten Sie die folgenden Frage:
   Geben Sie die kleinste obere Schranke für die Anzahl der Rechenschritte eines n-Bit-Von-Neumann-Addierwerkes an. (2 Punkte)

2. Betrachten Sie das Von-Neumann-Addierwerk in der folgenden Abbildung.



(a) Simulieren Sie die Arbeitsweise eines Von-Neumann-Addierwerkes indem Sie die nachfolgende Tabelle ausfüllen. Die zu addierenden Werte 11 und 13 stehen bereits im Akku bzw. im Puffer. (4 Punkte)

Zeile	S	U	$P_3P_2P_1P_0$	$A_3A_2A_1A_0$
1	0	0	0000	0000
2	1	0	1101	1011
3				
4				
5				
6				

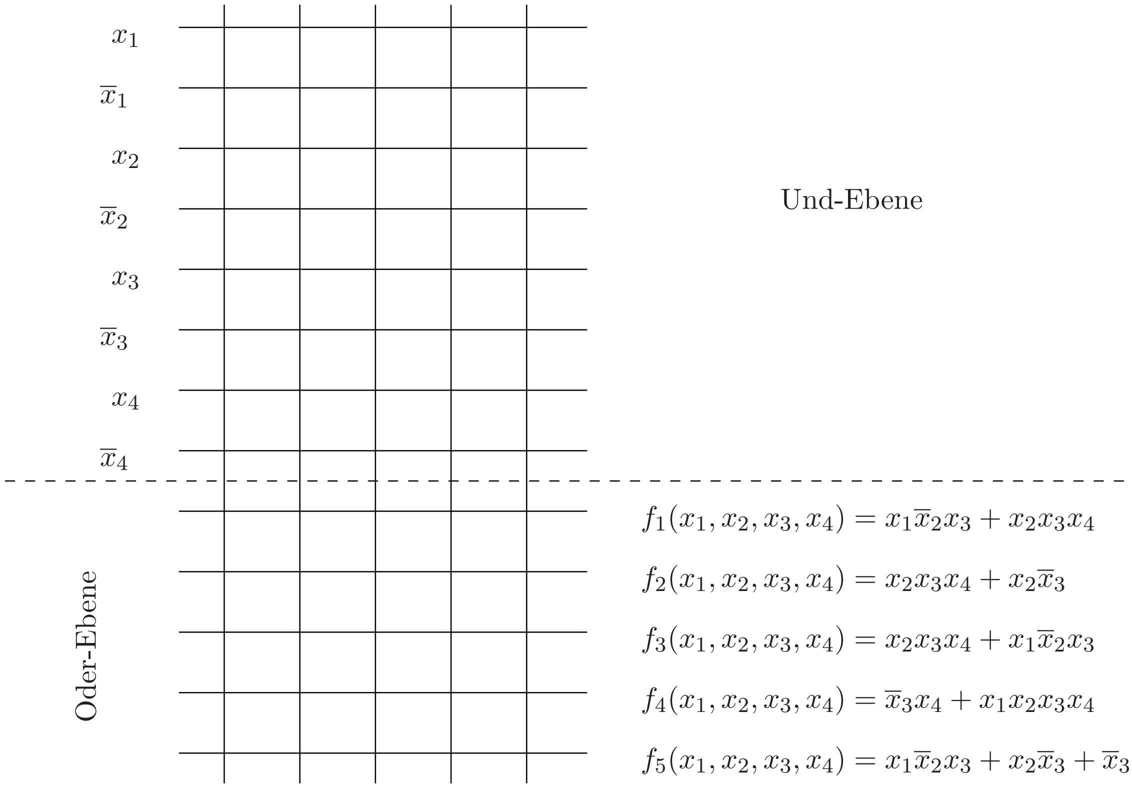
(b) Addieren Sie in der nächsten Tabelle den Wert 15. Diese Werte sind in der Tabelle bereits im Akku bzw. im Puffer eingetragen. (2 Punkte)

Zeile	S	U	$P_3P_2P_1P_0$	$A_3A_2A_1A_0$
1	0	0	0000	0000
2	1	0	1111	1111
3				
4				

(c) Erklären Sie, warum im zweiten Fall deutlich weniger Schritte zur Berechnung benötigt werden. (2 Punkte)

Aufgabe 12 : Programable Logic Array (5 Punkte)

Skizzieren Sie, wie sich die unten angegebenen Booleschen Funktionen f_1, f_2, f_3, f_4, f_5 mittels eines PLA realisieren lassen, indem Sie die entsprechenden Punkte in das folgende punktorientierte PLA eintragen.



$f_1(x_1, x_2, x_3, x_4) = x_1\overline{x}_2x_3 + x_2x_3x_4$

$f_2(x_1, x_2, x_3, x_4) = x_2x_3x_4 + x_2\overline{x}_3$

$f_3(x_1, x_2, x_3, x_4) = x_2x_3x_4 + x_1\overline{x}_2x_3$

$f_4(x_1, x_2, x_3, x_4) = \overline{x}_3x_4 + x_1x_2x_3x_4$

$f_5(x_1, x_2, x_3, x_4) = x_1\overline{x}_2x_3 + x_2\overline{x}_3 + \overline{x}_3x_4$

Aufgabe 13 : Gleitkommadarstellung IEEE (11 Punkte)

In den folgenden Teilaufgaben werden Gleitkommazahlen nach IEEE 754 verwendet.

a) Konvertieren Sie die Zahl $-16,3125$ in Dezimaldarstellung in eine Gleitkommazahl gemäß IEEE 754. Berechnung Sie dazu die Dualzahl (Mantisse), den Exponenten und das Vorzeichen. (5 Punkte)

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

b) Berechnen Sie das Produkt der folgenden 32-Bit-IEEE-Gleitkomma-Zahlen: (6 Punkte)

11000101101011000000000000000000
11000001101010000000000000000000

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

Aufgabe 14 : Von-Neumann-Rechner (8 Punkte)

Gegeben ist ein Von-Neumann-Rechner. Welche der folgenden Aussagen sind richtig?

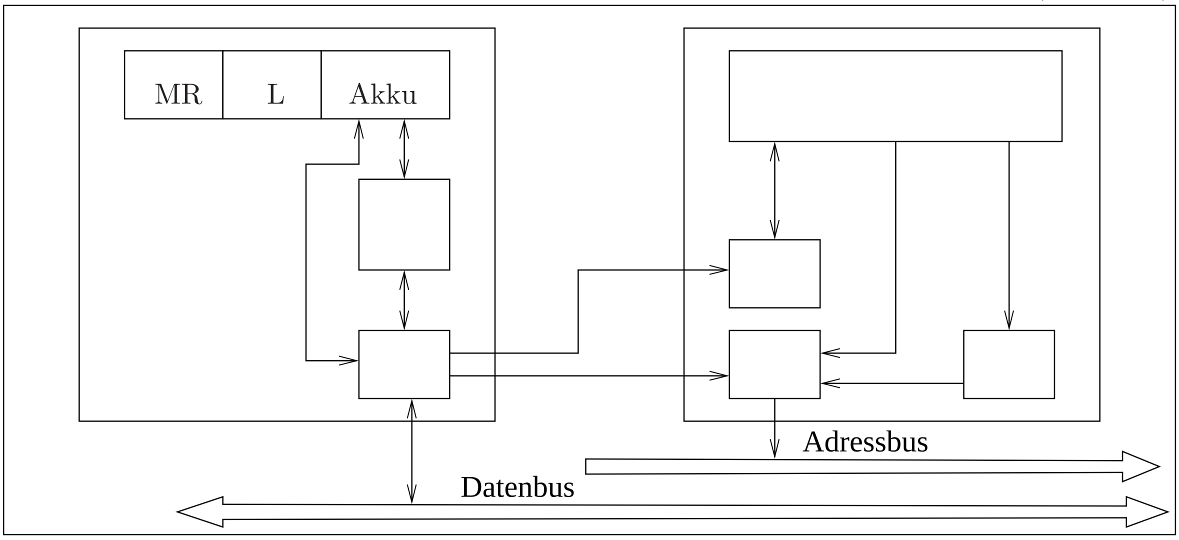
1. Allgemeine Fragen zum Von-Neumann-Rechner: (2,5 Punkte)

	ja	nein	weiß nicht
(a) Beim interrupt-gesteuerten I/O ist die CPU nicht beteiligt.	☐	☐	☐
(b) Die Hauptbestandteile eines Von-Neumann-Rechners sind CPU, Speicher, I/O und die verschiedenen Busse.	☐	☐	☐
(c) Die Aufgabe des Datenprozessors besteht darin, Befehle zu entschlüsseln und deren Ausführung zu steuern.	☐	☐	☐
(d) Der charakteristische Von-Neumann-Rechner arbeitet nach dem SIMD-Prinzip.	☐	☐	☐
(e) Die jeweilige Verwendung eines Datums im Speicher richtet sich nach dem momentanen Kontext.	☐	☐	☐

2. Fragen zur CPU eines Von-Neumann-Rechners: (2,5 Punkte)

	ja	nein	weiß nicht
(a) Der von-Neumannsche Flaschenhals bezeichnet die Kommunikation zwischen CPU und Speicher.	☐	☐	☐
(b) Die ALU dient der Berechnung von Operationen im Datenprozessor.	☐	☐	☐
(c) Der Akku ist ein Universalregister.	☐	☐	☐
(d) Der PC wird stets um die Anzahl der Bytes, welche zur Speicherung des gerade ausgeführten Befehls benötigt werden, erhöht.	☐	☐	☐
(e) Jeder Befehl eines Von-Neumann-Rechners muss einen Operanden haben.	☐	☐	☐

3. Beschriften Sie im folgenden Bild die fehlenden Komponenten! (3 Punkte)



Aufgabe 15 : CISC & RISC Rechner (5 Punkte)

a) Sie haben CISC und RISC Prozessoren kennengelernt. Was ist der Unterschied zwischen den beiden Prozessorenarten in Bezug auf ihre Befehlssätze? (1 Punkt)

b) Warum haben RISC Prozessoren deutlich mehr Register als CISC Prozessoren? (2 Punkte)

c) Warum wird im Kontext von RISC Prozessoren auch von einer Load- und Store-Architektur gesprochen? (2 Punkte)

Aufgabe 16 : MMIX-Programmierung (10 Punkte)

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

a) Die folgenden Programmzeilen stammen aus der Simulation eines MMIX-Befehls.

```
mmix>
          9. 000000000000010c: 20fcfcfb (ADD) $252=g[252] = 36 + 9 = 45
   43 instructions, 0 mems, 43 oops; 9 good guesses, 0 bad
   (now at location #0000000000000110)
```

Welcher Befehl wurde ausgeführt? (1 Punkt)

An welcher Speicherstelle steht der Befehl? (1 Punkt)

Wieviel Speicher braucht der Befehl? (1 Punkt)

Welche Register werden für den Befehl genutzt (Nummer des Registers)? (1 Punkt)

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

```
x = 1;
y = 20;

FOR i = 20 DOWNTO 0 STEP 2 DO
    IF (i > 9) THEN
        x = x + i;
    ELSE
        y = y - i;
    ENDIF
ENDFOR
```

Benutzen Sie nebenstehende Box zur Erstellung Ihres Programms. Es müssen nicht alle vorgegeben Zeilen genutzt werden.(6 Punkte)

```
% MMIX Program
        LOC #100
x       GREG
y       GREG
i       GREG
hilf    GREG
....    ........................
....    ........................
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