Klausur Informatik 1
3. März 2003

Jähnichen / Flik
Lechtchinsky / Neunast

Name: ....

Vorname: ....

Matr.-Nr.: ....

Bearbeitungszeit: 120 Minuten

- Als Hilfsmittel sind nur ein nicht programmierbarer Taschenrechner und ein doppelseitig, **in eigener Handschrift** beschriebenes DIN-A4-Blatt zugelassen.
- Benutzen Sie für die Lösung der Aufgaben nur das mit diesem Deckblatt ausgeteilte Papier. **Lösungen, die auf anderem Papier geschrieben werden, können nicht gewertet werden!**
- Schreiben Sie Ihre Lösungen auf das Aufgabenblatt der jeweiligen Aufgabe. Verwenden Sie auch die Rückseiten. **Schreiben Sie keine Lösungen einer Aufgabe auf ein Blatt, das nicht zu dieser Aufgabe gehört!** Wenn Sie zusätzliche, von uns ausgegebene Blätter verwenden, geben Sie unbedingt an, zu welcher Aufgabe die Lösung gehört!
- Schreiben Sie deutlich! Doppelte, unleserliche oder mehrdeutige Lösungen werden nicht gewertet!
- Schreiben Sie nur in **blau** oder **schwarz**. Lösungen, die mit Bleistift geschrieben sind, können nicht gewertet werden!
- Erscheint Ihnen eine Aufgabe mehrdeutig, wenden Sie sich an die Betreuer.
- Tragen Sie jetzt (vor Beginn der Bearbeitungszeit) auf *allen* Blättern Ihren Namen und Ihre Matrikelnummer ein.

Aufgabe	Punkte	erreicht
1	15	
2	18	
3	2	
4	6	
5	5	
6	3	

1. Aufgabe (15 Punkte): Funktionen höherer Ordnung

In dieser Aufgabe werden totale zweistellige boolsche Funktionen betrachtet, also Funktionen mit der Funktionalität `bool ** bool -> bool`, die für alle möglichen Werte der Argumente ein wohldefiniertes Ergebnis liefern. Eine Wertetabelle für eine solche Funktion läßt sich als eine Sequenz von Tripeln darstellen. Jedes Tripel speichert dabei die Werte der beiden Argumente sowie das Ergebnis. Ein solches Tripel kann durch den folgenden Datentyp repräsentiert werden:

```
DATA tripel == tripel(arg1 : bool, arg2 : bool, erg : bool)
```

Hier entsprechen `arg1` und `arg2` den beiden Argumenten und `erg` dem Ergebnis der Anwendug der Funktion auf diese Argumente. Eine mögliche Wertetabelle für die Funktion `and` wäre demnach

```
   tripel(false, false, false)        -- false and false = false
:: tripel(false, true, false)         -- false and true  = false
:: tripel(true, false, false)         -- true  and false = false
:: tripel(true, true, true)           -- true  and true  = true
:: <>
```

Beachten Sie allerdings, daß nach dieser Definition zu einer Funktion unendlich viele Wertetabellen gehören, da die Reihenfolge der Tripel nicht festgelegt ist und diese auch mehrfach in der Wertetabelle vorkommen können.

Hinweis: Beachten Sie bei der Bearbeitung dieser Aufgabe die Hinweise zur Bearbeitung von OPAL-Aufgaben auf Seite 6.

1.1. Wertetabellen (2 Punkte) Deklarieren und definieren Sie die Funktion `tabelle`, die eine Wertetabelle für eine zweistellige boolsche Funktion liefert.

1.2. Überprüfung von Tripeln (2 Punkte) Deklarieren und definieren Sie die Funktion `passt?`, die überprüft, ob ein Tripel zu eine Funktion paßt. So paßt `tripel(true, false, true)` zu der Funktion `or`, da `or(true, false) = true` ist. Dieses Tripel paßt allerdings nicht zu der Funktion `and`, da `and(true, false) = false` ist.

1.3. Überprüfung von Wertetabellen (4 Punkte) Deklarieren und definieren Sie die Funktion `passtTabelle?`, die überprüft, ob eine Wertetabelle zu einer Funktion paßt. Sie können davon ausgehen, daß die Wertetabelle alle möglichen Kombinationen von Argumenten abdeckt.

1.4. Bestimmung von Funktionen aus Wertetabellen (4 Punkte) Deklarieren und definieren Sie die Funktion `funktion`, die eine Wertetabelle als Argument bekommt und eine zweistellige boolsche Funktion als Ergebnis liefert, die dieser Wertetabelle entspricht. Sie können davon ausgehen, daß die Wertetabelle alle möglichen Kombinationen von Argumenten abdeckt.

1.5. Gleichheit von Funktionen (3 Punkte) Deklarieren und definieren Sie die Funktion `gleich?`, die überprüft, ob zwei zweistellige boolsche Funktionen gleich sind. Zwei Funktionen sind genau dann gleich, wenn sie für jede Kombination von Argumenten das gleiche Ergebnis liefern.

2. Aufgabe (18 Punkte): Datentypen

Eine elektronische Schaltung, die nur aus NOR-Gattern besteht, kann durch den folgenden Datentyp repräsentiert werden:

```
DATA schaltung == null
                  eins
                  eingang( name : char )
                  nor( links : schaltung,
                       rechts : schaltung )
```

Hierbei repräsentieren `null` und `eins` entsprechende festverdrahtete Eingangswerte, `eingang` einen benannten Eingang und `nor` ein NOR-Gatter, bei dem die Eingänge mit den Ausgängen der Schaltungen `links` und `rechts` verbunden sind.

Hinweis: Der Ausgang eines NOR-Gatters ist genau dann eins, wenn beide Eingänge null sind. Beachten Sie bei der Bearbeitung dieser Aufgabe die Hinweise zur Bearbeitung von Opal-Aufgaben auf Seite 6.

2.1. Induzierte Signatur (3 Punkte) Welche Signatur induziert die obige Datentypdefinition? Benennen Sie die einzelnen Komponenten. Welche der induzierten Funktionen sind nicht total? Begründen Sie Ihre Anwort.

2.2. Eingänge bestimmen (4 Punkte) Deklarieren und definieren Sie die Funktion `eingaenge`, die die Namen aller Eingänge einer gegebenen Schaltung bestimmt und als Sequenz liefert. Die Namen können in der Ergebnissequenz doppelt vorkommen.

2.3. Eingänge umbenennen (4 Punkte) Deklarieren und definieren Sie die Funktion `umbenennen`, die in einer gegebenen Schaltung alle Eingänge umbenennt. Diese Funktion erhält eine Funktion vom Typ `char -> char` als Argument und bestimmt den neuen Namen eines Eingangs durch die Anwendung dieser Funktion auf den alten Namen.

2.4. Schaltungen vereinfachen (5 Punkte) Deklarieren und definieren Sie eine Funktion `vereinfachen`, die eine gegebene Schaltung so weit wie möglich vereinfacht. Beachten Sie dabei, daß der Ausgang eines NOR-Gatters nicht in allen Fällen von beiden Eingängen abhängt. So ist der Ausgang null, wenn einer der beiden Eingänge eins ist.

2.5. Erweiterung des Datentyps `schaltung` (2 Punkte) Der Datentyp `schaltung` kann bislang nur Schaltungen repräsentieren, die ausschließlich aus NOR-Gattern bestehen. Verändern Sie diesen Datentyp so, daß Schaltungen beliebige Gatter mit zwei Eingängen enthalten können. Die Art des Gatters wird dabei durch den Aufzählungstyp `art` repräsentiert. Eine mögliche Definition von `art` ist z.B.

```
DATA art = nor nand or and xor
```

Hinweise zur Bearbeitung der Opal-Aufgaben

Bei der Bearbeitung der Opal-Aufgaben können alle Funktionen aus der Opal-Standardbibliothek verwendet werden. Auf `IMPORT`-, `SIGNATURE`- und `IMPLEMENTATION`-Deklarationen kann verzichtet werden. Eine nicht vollständige Liste der Strukturen und Funktionen aus der Opal-Standardbibliothek ist im folgenden angegeben.

```
SIGNATURE Bool
FUN true false :                 bool
FUN not        : bool         -> bool
FUN and or     : bool ** bool -> bool
FUN =          : bool ** bool -> bool

SIGNATURE Seq[data]
SORT data
TYPE seq == <>
            ::(ft : data, rt : seq)
FUN ++ : seq ** seq -> seq
FUN =  : (data ** data -> bool) -> seq ** seq -> bool

SIGNATURE SeqMap[from,to]
SORT from to
FUN map : (from -> to) -> seq[from] -> seq[to]

SIGNATURE SeqFilter[data]
SORT data
FUN filter : (data -> bool) -> seq[data] -> seq[data]

SIGNATURE SeqReduce[from,to]
SORT from to
FUN reduce : (from ** to -> to) ** to -> seq[from] -> to

SIGNATURE SeqZip[from1,from2,to]
SORT from1 from2 to
FUN zip : (from1 ** from2 -> to) -> seq[from1] ** seq[from2] -> seq[to]
```

3. Aufgabe (2 Punkte): Codesicherung

Ermitteln Sie für den folgenden Datenstrom die CRC-Prüfbits (Generatorpolynom $x^6 + x^4 + 1$):

$$0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 1 \quad 1 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1$$

Geben Sie auch den zu versendenden Datenstrom an.

4. Aufgabe (6 Punkte): Zahlendarstellungen

4.1. (1 Punkt) Wandeln Sie -726_{10} in eine 12-Bit-2-Komplementzahl um. Die Rechenschritte müssen erkennbar sein!

4.2. (2 Punkte) Multiplizieren Sie 0101.0011_2 mit 010.11100_2. Kennzeichnen Sie deutlich an welcher Stelle der Punkt im Ergebnis steht.

Hinweis: Für die nächsten beiden Teilaufgaben ist das aus der Vorlesung bekannte Gleitpunktformat gegeben: 1-Bit-Vorzeichen, 4-Bit-Exponent, 11-Bit-Mantisse, Bias=7.

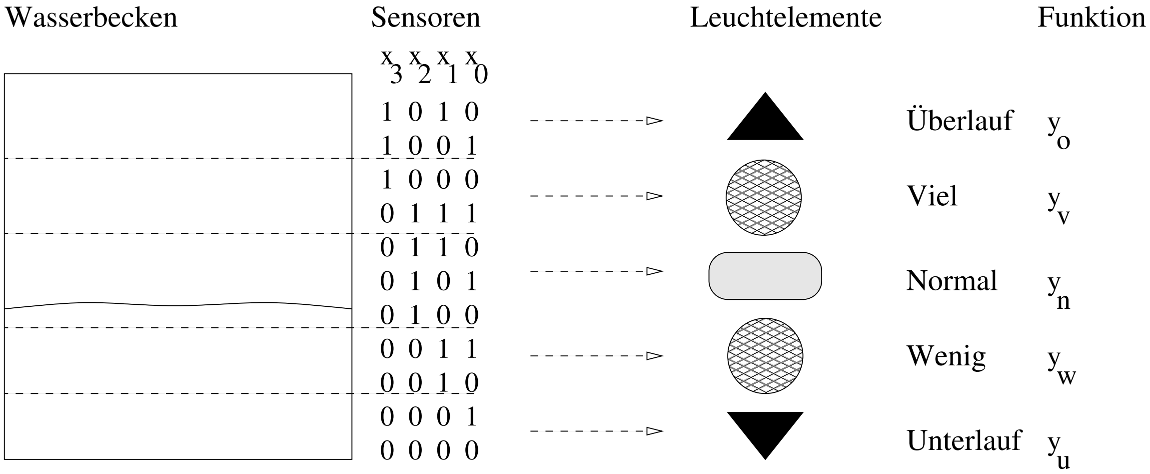
4.3. (1 Punkt) Stellen Sie 0101.100101_2 im Gleitpunktformat dar.

4.4. (2 Punkte) Addieren Sie die im Gleitpunktformat dargestellten Zahlen 0101100101011011 und 0010101010000001 und geben Sie das Ergebnis wieder im Gleitpunktformat an.

5. Aufgabe (5 Punkte): Minimierung boolescher Funktionen

Es soll eine Wasserstandskontrolle mit Anzeige durch Leuchtelemente entwickelt werden. Die Zuordnungen der einzelnen Leuchtelemente zu den vom Sensor übertragenen Bitmustern wird durch die untenstehende Abbildung visualisiert. Da die Anzeige aus nur 5 Elementen besteht, werden jeweils mehrere Bitmuster auf ein Element abgebildet.

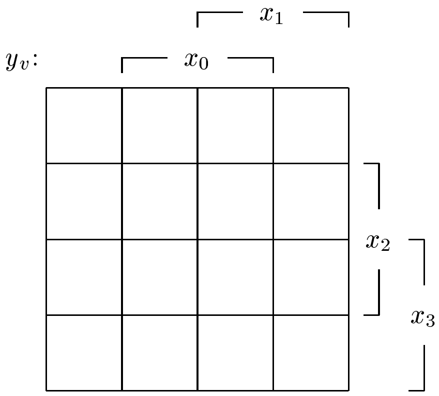
Hinweis: Die Teilaufgaben sind voneinander unabhängig lösbar.

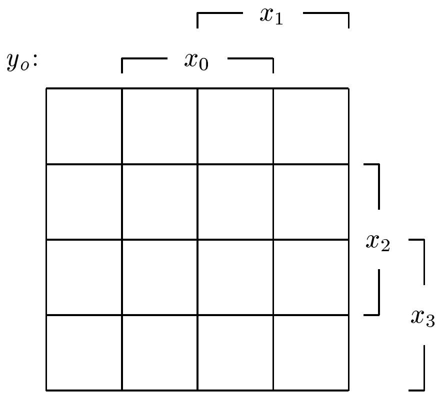


5.1. Wertetabelle (1 Punkt) Ergänzen Sie die nachstehende Wertetabelle um die Funktionswerte y_u und y_w.

5.2. KV-Tafel (1 Punkt) Übertragen Sie die Funktionswerte für y_v und y_o in die nachstehenden KV-Tafeln.

x_3	x_2	x_1	x_0	y_u	y_w	y_v	y_o
0	0	0	0			0	0
0	0	0	1			0	0
0	0	1	0			0	0
0	0	1	1			0	0
0	1	0	0			0	0
0	1	0	1			0	0
0	1	1	0			0	0
0	1	1	1			1	0
1	0	0	0			1	0
1	0	0	1			0	1
1	0	1	0			0	1
1	0	1	1			*	*
1	1	0	0			*	*
1	1	0	1			*	*
1	1	1	0			*	*
1	1	1	1			*	*





Name: ....

Matr.-Nr. ....

A5

5.3. Zusammenfassen (1 Punkt) Markieren Sie für y_n in der linken KV-Tafel die Bereiche, die zu einer minimalen *disjunktiven* Normalform zusammengefaßt werden müssen sowie in der rechten KV-Tafel die Bereiche, die zu einer minimalen *konjunktiven* Normalform zusammengefaßt werden müssen.

x_1 x_0 y_n

	0	0	0	0
	1	1	0	1
	*	*	*	*
	0	0	*	0

x_2 x_3

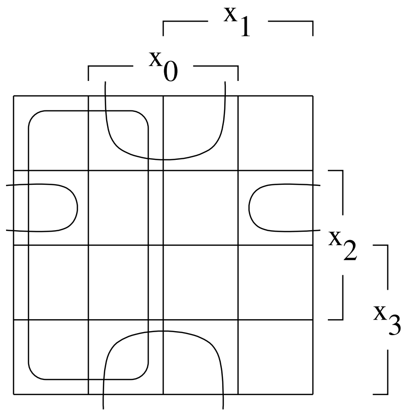
x_1 x_0 y_n

	0	0	0	0
	1	1	0	1
	*	*	*	*
	0	0	*	0

x_2 x_3

5.4. Auslesen (2 Punkte) Beschreiben Sie die in den nachstehenden KV-Tafeln markierten Bereiche durch eine *disjunktive* Normalform für die durch die linke KV-Tafel gegebene Funktion z und durch eine *konjunktive* Normalform für y_w.

z



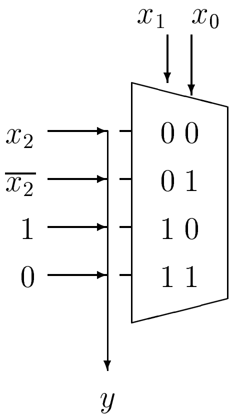
x_1 x_0 y_w

0	0	1	1
0	0	0	0
*	*	*	*
0	0	*	0

x_2 x_3

6. Aufgabe (3 Punkte): Schaltnetze

6.1. Multiplexer (1 Punkt) Vervollständigen Sie die vorgegebene Wertetabelle, indem Sie für die gegebenen Eingangswerte die korrekten Ausgaben des Multiplexer eintragen.



x_2	x_1	x_0	y
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

6.2. PAL (2 Punkte) Die Funktionen y und z sollen durch einen PAL-Baustein realisiert werden. Tragen Sie dazu in der Und-Matrix die erforderlichen Verbindungen ein. Quadrate stehen für vorgegebene, Kreise für programmierbare Verbindungen. Beachten Sie, daß die Anzahl der Zeilen für jede Ausgangsfunktion begrenzt ist.

x_1	x_2	x_3	y	z
0	0	0	1	1
0	0	1	0	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	0
1	0	1	1	1
1	1	0	0	1
1	1	1	0	0