Fachbereich Informatik

Lehrgebiet Technische Informatik I

Kurs 1707 „Technische Informatik I“

Lösungsvorschläge zu den Aufgaben

der Nachklausur im SS 2002

(am 21. September 2002)

Aufgabe 1 Boolesche Algebra (15 Punkte)

Die Boolesche Algebra begegnet Ihnen nicht nur im Umfeld der Technischen Informatik, sondern u.a. auch beim Programmieren mit imperativen Programmiersprachen wie C, Cobol oder Pascal.
Im Rahmen dieser Aufgabe sollen Sie gegebene Schleifen- und Verzweigungsbedingungen mit den im Kurs kennengelernten Methoden nach den jeweiligen Vorgaben umformen und/oder vereinfachen. Als Referenzsprache haben wir hierfür die Sprache Pascal verwendet. Sofern Sie mit Pascal nicht vertraut sind, gibt Ihnen der folgende Abschnitt eine Einführung in die für die Lösung dieser Aufgabe benötigten Sprachelemente.
Hinweis: In der Praxis hat es sich bewährt, komplexe Bedingungen während der Berechnung durch einfache Konstanten abzukürzen.

Im Rahmen dieser Aufgabe benötigen Sie die folgenden Sprachelemente:

1. Blöcke
   Befehle, die stets zusammen ausgeführt werden sollen, werden zu Blöcken zusammengefaßt. Enthält ein Block mehrere Befehle, wird er in Pascal mit BEGIN und END eingefasst. Bei nur aus einem einzigen Befehl bestehenden Blöcken kann auf diese Einfassung verzichtet werden.
2. Entscheidungen
   Die IF/THEN-Anweisung wird zur Realisierung von Entscheidungen verwendet. Dem Befehlswort IF folgt die Bedingung, die erfüllt werden muß, damit der auf das THEN nachfolgende Block ausgeführt wird. Sollte die Bedingung nicht erfüllt werden, kann über einen ELSE-Zweig ein alternativer Block angesprungen werden.
3. Schleifen
   Wie die meisten anderen imperativen Programmiersprachen verfügt auch Pascal über kopf- und fußgesteuerte Schleifen. Bei den kopfgesteuerten Schleifen wird vor Eintritt in den Schleifenrumpf geprüft, ob die gestellte Bedingung erfüllt ist. Sollte diese Bedingung nicht erfüllt sein, wird auf den nächsten Befehl hinter der Schleife gesprungen, andernfalls kommt der aus einem Block bestehende Schleifenrumpf so lange zur Ausführung, bis die Bedingung nicht mehr erfüllt ist. Diese Schleifenform hat das Aussehen

   ```
   WHILE (<Bedingung>) DO
         <Block>
   ```

   und kann mit „Führe <Block> aus, so lange <Bedingung> erfüllt ist !“ übersetzt werden. Im Gegensatz hierzu gilt für fußgesteuerte Schleifen, dass sie solange ausgeführt werden, bis die am Schleifenende positionierte (Abbruch-) Bedingung erfüllt ist. Die Notation in Pascal lautet

   ```
   REPEAT
         <Block>
   UNTIL <Bedingung>;
   ```

und kann mit „Führe <Block> solange aus, bis <Bedingung> erfüllt ist" gleichgesetzt werden.

4. Bedingungen, Ausdrücke, Operatoren und Zuweisung
   Bedingungen bestehen aus Booleschen Ausdrücken. Hinsichtlich der Formulierung von Ausdrücken gelten die gleichen Gesetzmäßigkeiten, wie Sie sie im Kurs kennengelernt haben. Einzig für die Operatoren werden andere Symbole verwendet:

Operator	Operator-Symbol in Pascal
NOT	NOT
AND	AND
OR	OR
Gleichheit	=
Zuweisung	:=

Zwecks Unabhängigkeit von den geltenden Vorrangbeziehungen zwischen den Operatoren haben wir sämtliche Ausdrücke sehr ausführlich mit Klammern ausgestattet.

a) Überführen Sie die Bedingung der folgenden if-Anweisung

IF
(NOT ((NOT ((X = 1) OR ((Y < 4) AND (K > 7))) AND (X = 1)) OR NOT (K > 7)))
THEN

durch die im Kurs kennen gelernten Booleschen Umformungen in eine minimale DNF. Geben Sie das Endergebnis wieder als if-Statement in gültiger Pascal-Syntax an. Bei dieser Teilaufgabe werden Lösungen mit Wahrheitstabellen oder Karnaugh-Diagrammen nicht anerkannt.

Es seien

$A := (X = 1)$, $B := (Y < 4)$ und $C := (K > 7)$.

Dann gilt:

(NOT ((NOT ((X = 1) OR ((Y < 4) AND (K > 7))) AND (X = 1)) OR NOT(K > 7)))

$= \overline{\overline{(A \vee BC)}\, A \vee \overline{C}}$

$= \overline{\overline{(A \vee BC)}\, A} \wedge C$

$= ((A \vee BC) \vee \overline{A}) \wedge C$

$= (1 \vee BC) \wedge C$

$= 1 \wedge C = C$

Das gesuchte minimierte Ergebnis lautet:

IF (K > 7) THEN

(6)

b) Weshalb lässt sich der Programmtext

```
IF (X = 1) THEN
        X := 0;
ELSE
        X := 1;
```

zu X := 1 – X vereinfachen ? Verwenden Sie bei Ihrer Argumentation die Gesetzmäßigkeiten der Zweierkomplementdarstellung !

Für das Zweierkomplement (ZK) einer n-stelligen Dualzahl X gilt:

$ZK(X) = 2^n - X.$

Es gilt ebenfalls: $ZK(X) = \overline{X} + 1.$

Daraus folgt: $2^n - X = \overline{X} + 1$

Da in unserem Fall ausschließlich einstellige Dualzahlen betrachtet werden, ist n = 1. Es gilt:

$2 - X = \overline{X} + 1 \quad \Leftrightarrow$

$\overline{X} = 1 - X$ (3)

c) Gegeben Sei die folgende kopfgesteuerte Schleife:

```
WHILE
      (((NOT((i < 5) AND NOT (j > 7)) OR (k = 2)) AND NOT(i < 5)) OR
      NOT(j  > 7))
DO
                        ...
```

Gesucht wird eine fußgesteuerte Schleife, deren Abbruchbedingung zu der Eintrittsbedingung der oben gezeigten kopfgesteuerten Schleife logisch äquivalent und minimal ist. Das unterschiedliche Verhalten der beiden Schleifentypen hinsichtlich des ersten Schleifendurchlaufs soll dabei nicht berücksichtigt werden.

Es seien

A := (i < 5), B := (j > 7) und C := (k = = 2).

Die Umwandlung erfolgt durch einfache Inversion der Bedingung:

NOT (((NOT((i < 5) AND NOT (j > 7)) OR (k = 2)) AND NOT(i < 5)) OR NOT(j > 7))

$= \overline{((\overline{A\overline{B}} \vee C) \wedge \overline{A}) \vee \overline{B}}$

$= \overline{((\overline{A\overline{B}} \vee C) \wedge \overline{A})} \wedge B$

$= ((\overline{A\overline{B}} \vee C) \vee A) \wedge B$

$= (A\overline{B}\overline{C} \vee A) \wedge B \qquad = \qquad AB$

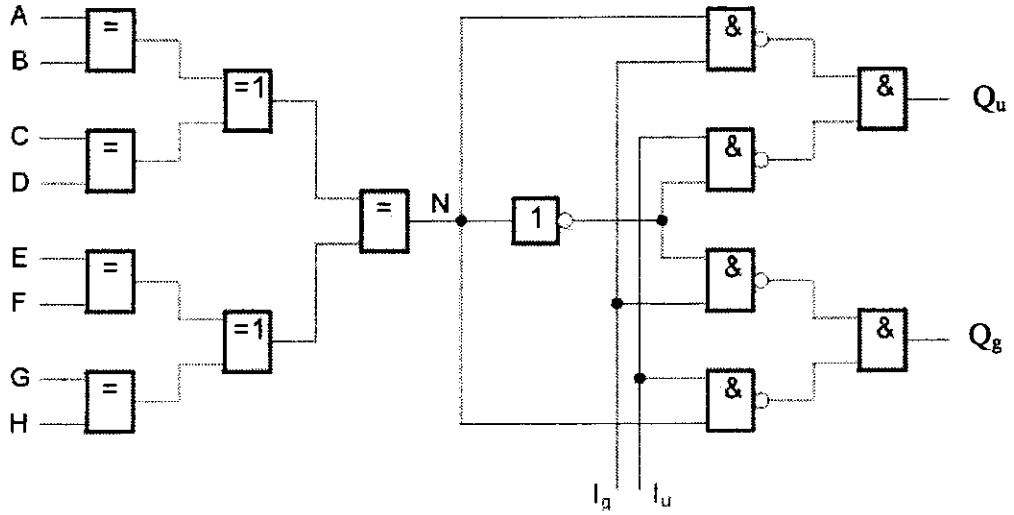
Das gesuchte Ergebnis lautet also:

```
REPEAT
...
UNTIL ((i < 5) AND (j > 7))
```

(6)

Aufgabe 2 Analyse eines TTL-Bausteins (15 Punkte)

Analysieren Sie das in der folgenden Abbildung gezeigte Schaltnetz des TTL-Bausteins SN74180 !

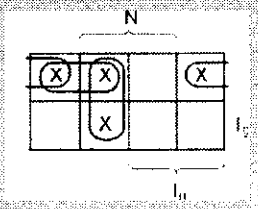


a) Minimieren Sie die Funktionsgleichungen der Ausgänge Q_g und Q_u unter Verwendung von Karnaugh-Diagrammen

Aus dem Schaltbild folgt für Q_g:

$Q_g = \overline{N\, I_u} \wedge \overline{\overline{N}\, I_g} = (\overline{N} \vee \overline{I_u}) \wedge (N \vee \overline{I_g}) = \overline{N}\,\overline{I_g} \vee N\,\overline{I_u} \vee \overline{I_u}\,\overline{I_g}$

Die Übertragung in das Karnaugh-Diagramm ergibt:

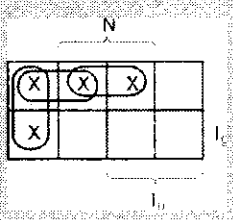


Und wir erhalten: $Q_g = \overline{N}\,\overline{I_g} \vee N\,\overline{I_u}$

Aus dem Schaltbild folgt für Q_u:

$Q_u = \overline{N\, I_g} \wedge \overline{\overline{N}\, I_u} = (\overline{N} \vee \overline{I_g}) \wedge (N \vee \overline{I_u}) = \overline{N}\,\overline{I_u} \vee N\,\overline{I_g} \vee \overline{I_g}\,\overline{I_u}$

Die Übertragung in das Karnaugh-Diagramm ergibt:



Und wir erhalten: $Q_u = \overline{N}\,\overline{I_u} \vee N\,\overline{I_g}$ (5)

Name: Vorname: Matr.-Nr.:

b) Stellen Sie das boolesche (logische) Verhalten in der vorgegebenen Tabelle dar !

N der 1-Signale von A bis H	I_g	I_u	Q_g	Q_u
0	0	0	1	1
0	0	1	1	0
0	1	0	0	1
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	1	0
1	1	1	0	0

(8)

c) Welche Aufgabe erfüllt dieser Baustein ?

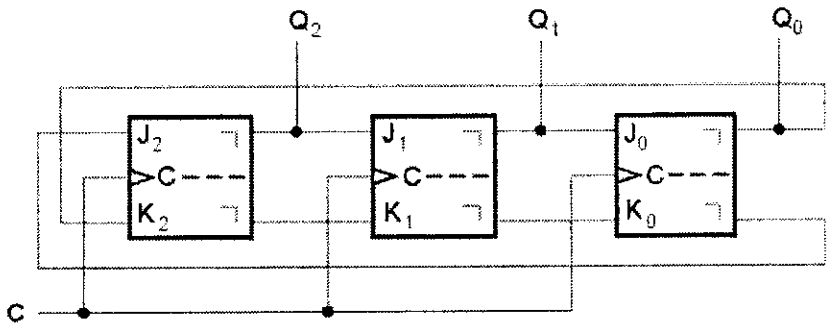
Bei diesem Baustein handelt es sich um einen 9-Bit Paritätsgenerator bzw. 8-Bit Paritätsprüfer.

(2)

Name: Vorname: Matr.-Nr.:

Aufgabe 3 Johnsonzähler (20 Punkte)

Analysieren Sie den sogenannten Johnsonzähler aus nachfolgend gezeigter Abbildung.



a) Bestimmen Sie ausgehend vom Startzustand (000) den Zählzyklus. Geben Sie eine Folgezustandstabelle an !

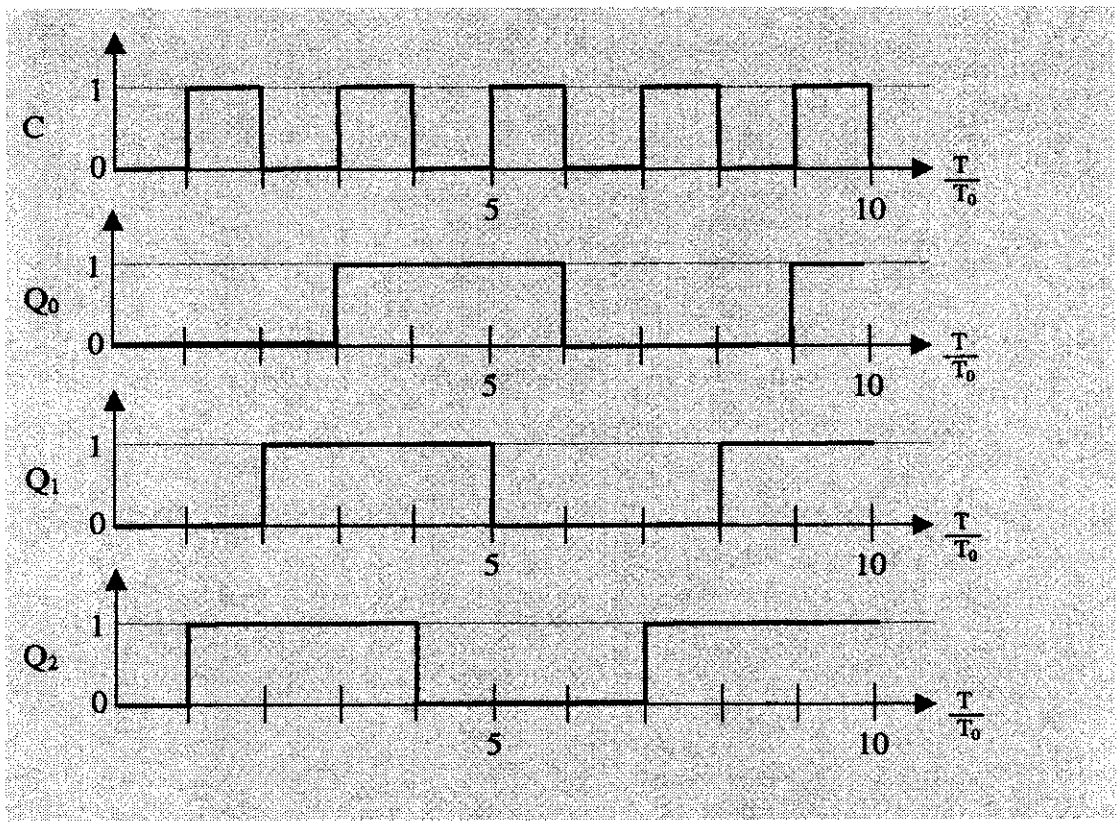
			t_n						t_{n+1}		
Q_0	Q_0	Q_2	J_0	K_0	J_1	K_1	J_2	K_2	Q_0	Q_1	Q_2
0	0	0	0	1	0	1	1	0	0	0	1
0	0	1	0	1	1	0	1	0	0	1	1
0	1	1	1	0	1	0	1	0	1	1	1
1	1	1	1	0	1	0	0	1	1	1	0
1	1	0	1	0	0	1	0	1	1	0	0
1	0	0	0	1	0	1	0	1	0	0	0

Der Zählzyklus lautet : 0, 1, 3, 7, 6, 4, 0, ...

(12)

Name: Vorname: Matr.-Nr.:

b) Vervollständigen Sie das folgende Impulsdiagramm !

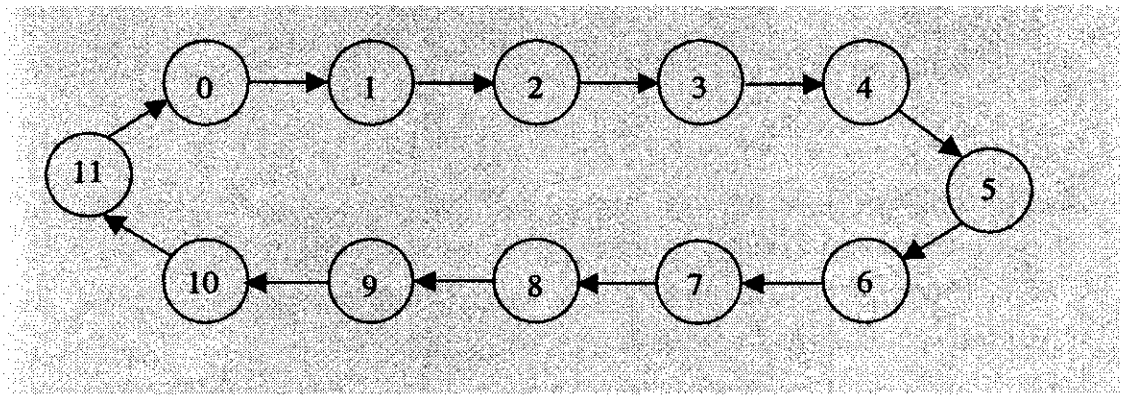


(8)

Aufgabe 4 Duodezimalsystem-Zähler (25 Punkte)

Im Kurs 1707 wurden neben Zahlen in Dezimaldarstellung auch Binär- und Hexadezimalzahlen verwendet. Auch wenn in der täglichen Praxis überwiegend diese 3 Zahlensysteme zum Einsatz kommen, kann nach dem gleichen Schema zu jeder natürlichen Zahl ein Zahlensystem gebildet werden. In dieser Aufgabe betrachten wir das Duodezimalsystem, welches die Zahl 12 zur Basis hat. Entwickeln Sie einen Duodezimalzähler, also eine Schaltung, die zyklisch die Ziffern von 0 bis 11 durchzählt, indem Sie die folgenden Teilaufgaben lösen:

a) Zeichnen Sie den Zustandsgraphen



(2)

b) Erstellen Sie die Zustandsfolgetabelle !

Q_3	Q_2	Q_1	Q_0 (t_n)	D_3	D_2	D_1	D_0	Q_3	Q_2 (t_{n+1})	Q_1	Q_0
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	0
0	0	1	0	0	0	1	1	0	0	1	1
0	0	1	1	0	1	0	0	0	1	0	0
0	1	0	0	0	1	0	1	0	1	0	1
0	1	0	1	0	1	1	0	0	1	1	0
0	1	1	0	0	1	1	1	0	1	1	1
0	1	1	1	1	0	0	0	1	0	0	0
1	0	0	0	1	0	0	1	1	0	0	1
1	0	0	1	1	0	1	0	1	0	1	0
1	0	1	0	1	0	1	1	1	0	1	1
1	0	1	1	0	0	0	0	0	0	0	0

(6)

c) Ein 16-Bit-Indexregister IR verfüge über die Möglichkeiten der **automatischen Modifikation** durch Pre-/Post-Inkrementierung bzw. Pre-/Post-Dekrementierung um einen wählbaren Offset n sowie der **Skalierung** mit einem Faktor m. Das Register besitze den momentanen (Hexadezimal-)Wert $0A88. Vom Befehl werde eine skalierte Adressierung mit automatischer Modifikation ausgeführt. Geben Sie an, welche Form der automatischen Modifikation und welche Werte für die Größen m und n gewählt werden müssen, wenn durch den Befehl die Speicheradresse $1500 ausgegeben werden und das Indexregister auf den Wert IR=$0A80 gesetzt werden soll. (Rechnung muß angegeben werden!)

Modifikation: Predekrementierung

m = 2 , n = 8

Rechnung: ($0A88-8)*2= $1500 (3)

d) Die 8-bit-Register R0, R1, R2 eines Mikroprozessors werden mit den folgenden Werten initialisiert: R0 := $1A, R1 := $80, R2 := $08.
Der Prozessor führe danach die nachstehende Operationenfolge solange aus, bis im Register R3 ein Wert ≠ $00 steht:

R3 := R0 ^ R1	*Logische UND-Verknüpfung*
R2 := R2 – 1	*Dekrementieren um 1*
R1 := >> R1	*Logisches Rechtsschieben um 1 bit*

Welche Werte stehen danach in R0 bis R3 ?

R0: $1A R1: $10 R2: $04 R3: $10 (2)

Welche Bedeutung hat der Wert in R2 nach Beendigung aller Operationen ?

R2 enthält ... den Index des höchstwertigen ‚1'-Bits der Zahl in R0. (1)