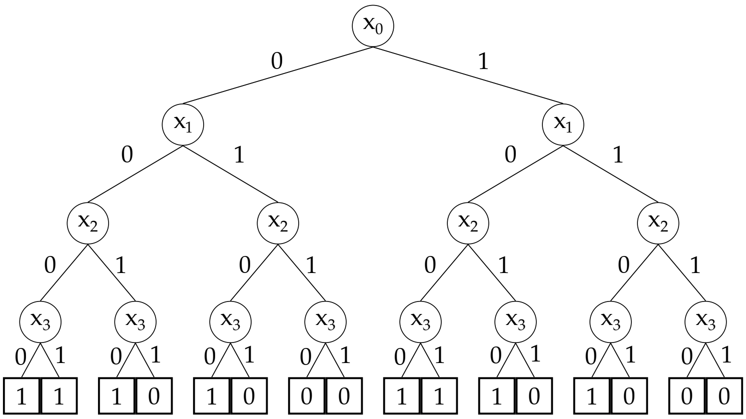
Grundlagen der Technischen Informatik 2
Sommersemester 25

Musterlösung Übungsblatt 3

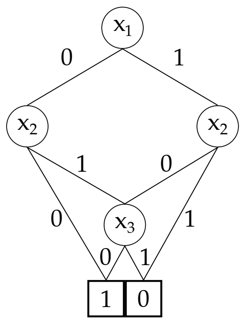
Aufgabe 1: Binäre Entscheidungsdiagramme

Gegeben sei die Funktion f durch das folgende geordnete binäre Entscheidungsdiagramm (OBDD):



1. Reduzieren Sie den OBDD so weit wie möglich und zeichnen Sie den rOBDD. Geben Sie bei jedem Schritt die angewandte Regel an.

   Regel 1: Eliminierung von Knoten mit gleichen Nachfolgern.
   Regel 2: Gemeinsame Nutzung gleicher Teilbäume.
   Lösung:
   3 Punkte: 2 für die Schritte bzw. den finalen Baum und 1 Punkte für die Angabe der Regeln. Nach Reduzierung erhält man folgenden rOBDD:



2. Leiten Sie aus dem rOBDD die Funktion f in disjunktiver Form ab.
   Lösung:
   Die Funktion erhält man, in dem man alle Pfade entlang zu den Einsen geht:

$$f' = \overline{x_2}\,\overline{x_1} \vee \overline{x_3} x_2 \overline{x_1} \vee \overline{x_3}\,\overline{x_2} x_1$$

3. Die minimierte Funktion von f lautet:

$$f_{min} = \overline{x_2}\,\overline{x_1} \vee \overline{x_3}\,\overline{x_1} \vee \overline{x_3}\,\overline{x_2}$$

Ist es somit möglich die Reduzierung des OBDDs als Minimierungsverfahren zu nutzen? Begründen Sie Ihre Entscheidung.

<u>*Lösung:*</u>
Vergleicht man f_{min} diese mit f erkennt man direkt, dass f nicht minimal ist. Somit kann man die Frage direkt mit nein beantworten.
Wir schauen beim Reduzieren immer nur auf einzelne Variablen bzw. eine kleine Menge von Funktionswerten; nicht auf alle Variablen bzw. die gesamte Funktion. Man betrachte folgende Minterme aus der DNF:

$$\overline{x_3}x_2\overline{x_1}x_0 \vee \overline{x_3}x_2\overline{x_1}\,\overline{x_0} \vee \overline{x_3}\,\overline{x_2}\,\overline{x_1}x_0$$

Betrachten wir alle drei zusammen, können wir folgenden Minimierung anwenden:

$$\overline{x_3}x_1\overline{x_1}(x_0 \vee \overline{x_0}) \vee \overline{x_3}\,\overline{x_1}x_0(x_2 \vee \overline{x_2}) = \overline{x_3}x_2\overline{x_1} \vee \overline{x_3}\,\overline{x_1}x_0$$

Arbeitet man schrittweise (wie bei der Reduktion):

$$\overline{x_3}x_2\overline{x_1}(x_0 \vee \overline{x_0}) \vee \overline{x_3}\,\overline{x_2}\,\overline{x_1}x_0 = \overline{x_3}x_2\overline{x_1} \vee \overline{x_3}\,\overline{x_2}\,\overline{x_1}x_0$$

Aufgabe 2: Maschinenzahlen

1. Wandeln Sie die folgenden Binärzahlen in Dezimalzahlen um.

   (a) 10001_2 (b) 1010111_2

   Lösung:

   Vorgehen: Man berechnet den Wert einer jeden Stelle der Binärzahl und addiert diese dann.
   Achtung: Die Stelle ganz rechts einer Zahl ist die mit dem kleinsten Wert.

   (a) $10001_2 = 16 + 0 + 0 + 0 + 1 = 17_{10}$

   (b) $64 + 0 + 16 + 0 + 1 + 1 + 1 = 87_{10}$

2. Wandeln Sie die folgenden Dezimalzahlen in Binärzahlen um.

   (a) 144_{10} (b) 413_{10}

   Lösung:
   Vorgehen: Man teilt die Dezimalzahl durch 2 und notiert den Rest. Diesen Schritt wiederholt man, bis die Zahl nicht mehr teilbar ist. Die Reste von unten nach oben gelesen und ergeben anschließend die Binärzahl.

   (a)

$$\begin{aligned} 144/2 &= 72 \quad R = 0 \\ 72/2 &= 36 \quad R = 0 \\ 36/2 &= 18 \quad R = 0 \\ 18/2 &= 9 \quad R = 0 \\ 9/2 &= 4 \quad R = 1 \\ 4/2 &= 2 \quad R = 0 \\ 2/2 &= 1 \quad R = 0 \\ 1/2 &= 0 \quad R = 1 \end{aligned}$$

   Daraus ergibt sich: 10010000_2

   (b) $413_{10} = 110011101_2$

3. Gegeben sei die Hexadezimalzahl $18A32D_{16}$. Wandeln Sie diese in eine Binärzahl um.

   Lösung:
   Um eine Hexadezimalzahl schnell in eine Binärzahl umzuwandeln, kann man die einzelnen Stellen der Hexadezimalzahl in Binär umwandeln und anschließend die Binärzahlen wieder zusammensetzen.
   Eine Hexadezimalstelle entspricht dabei 4 Binärstellen, da $2^4 = 16$
   z.B: $A_{16} = 10_{10} = 1010_2$
   Dieses wäre das 3. Segment der Binärzahl.
   Achtung: Dieses Verfahren funktioniert jedoch nicht für alle Umwandlungen.
   $18A32D_{16} = 1'1000'1010'0011'0010'1101'_2$

4. Gegeben sei die Binärzahl 11010101001001001111_2. Wandeln Sie diese in eine Hexadezimalzahl um.

   Lösung:
   Analog zu der vorherigen Aufgabe funktioniert das Verfahren auch umgekehrt.
   $'1101'0101'0010'0100'1111'_2 = D524F_{10}$

5. Berechnen Sie das Zweierkomplement der folgenden 8-Bit Integer.

   (a) `0x00001100` (b) `0x11111100`

   Lösung:
   Vorgehen: Es werden alle Stellen der Binärzahl invertiert und anschließend 1 addiert.

   (a) $0_b00001100 \rightarrow 11110011 \rightarrow 11110100$
   (b) $0_b11111100 \rightarrow 00000011 \rightarrow 00000100$

6. Wandeln Sie die folgenden Dezimalzahlen in IEEE754 16-bit half-precision floating-point Zahlen um.

   (a) 10000_{10} (b) 16.16_{10}

   Lösung:

   (a) 16 Bit: 1 Bit Vorzeichen, 5 Bit Exponent, 10 Bit Mantisse
   Vorzeichen: 0 *(da Zahl positiv)*
   $10000_{10} = 10011100010000{,}_2$
   Zählen wie weit das Komma geschoben werden muss, bis es hinter ersten 1 steht: 13 mal
   Exponent = $13 + 15$*(festgelegt durch Standard)* $= 28_{10} = 11100_2$
   Mantisse: von Binärzahl 10 Bit ohne 1 vor Komma notieren: 0011100010
   Zahl zusammensetzen: 0 11100 0011100010

   (b) Vorzeichen: 0
   $16,16_{10} = 10000,00101000...$
   Stellen hinter dem Komma werden wie folgt berechnet:

$$\begin{aligned}
0,16 * 2 &= 0,32 && \rightarrow 0\text{, da 0 vor Komma} \\
0,32 * 2 &= 0,64 && \rightarrow 0 \\
0,64 * 2 &= 1,28 && \rightarrow 1\text{, da 1 vor Komma anschließend } -1 \\
0,28 * 2 &= 0,56 && \rightarrow 0 \\
0,56 * 2 &= 1,12 && \rightarrow 1 \\
0,12 * 2 &= 0,24 && \rightarrow 0 \\
& ...
\end{aligned}$$

   Exponent: $4 + 15 = 19 = 10011$
   Mantisse: 0000001010
   Zahl zusammensetzen: 0 10011 0000001010

7. Wandeln Sie die folgende IEEE754 32-bit floating-point Zahlen in eine Dezimalzahl um.

   (a) `0 10001000 11110011100000000000000`
   Lösung:
   Vorzeichen: 0 → Zahl positiv
   Exponent: $10001000_2 = 136_{10} \rightarrow 136 - 127$ *(Ergibt sich aus Standard)* $= 9$
   Komma von Mantisse um 9 Stellen verschieben und 1 vor Zahl ergänzen: $1111100111,00000000000000$
   Binärzahl umwandeln: $1111100111_2 = 999_{10}$

Aufgabe 3: Schaltnetze

1. Seien $A =$`0b00010101` und $B =$`0b00111011` als zwei signed 8-bit Integer gegeben.
   (a) Berechnen Sie $A + B$. Führen Sie dafür binäre Addition durch. (Das Ergebnis soll ebenfalls ein signed 8-bit Integer sein.)
   (b) Berechnen Sie $B - A$. Führen Sie dafür binäre Subtraktion durch. (Das Ergebnis soll ebenfalls ein signed 8-bit Integer sein.)

   Lösung:

   (a)

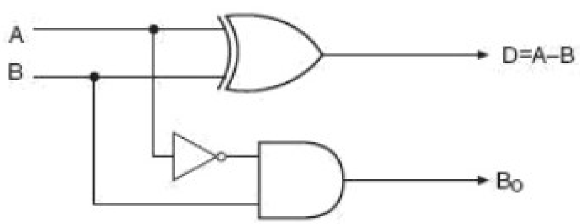
$$\begin{array}{r} 00010101 \\ +00111011 \\ ------ \\ 1010000 \end{array}$$

   (b)

$$\begin{array}{r} 00010101 \\ -00111011 \\ ------ \\ 00100110 \end{array}$$

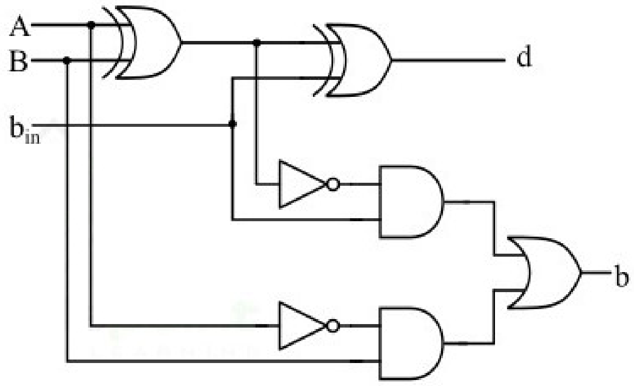
2. Konstruieren Sie analog zum Adder (siehe Vorlesung) einen Subtracter. (Ein Schaltnetz, welches die binäre Subtraktion durchführen kann.)
   (a) Entwerfen Sie einen Half-Subtracter. (Eine Schaltung, die zwei Bits subtrahieren kann.)
   Lösung:

A	*B*	*S*	*b*
0	*0*	*0*	*0*
0	*1*	*1*	*1*
1	*0*	*1*	*0*
1	*1*	*0*	*0*



   (b) Erweitern Sie diese Schaltung zu einem Full-Subtracter. (Eine Schaltung, die drei Bits subtrahieren kann.)
   Lösung:

A	*B*	*C*	*S*	*b*
0	*0*	*0*	*0*	*0*
0	*0*	*1*	*1*	*1*
0	*1*	*0*	*1*	*1*
0	*1*	*1*	*0*	*1*
1	*0*	*0*	*1*	*0*
1	*0*	*1*	*0*	*0*
1	*1*	*0*	*0*	*0*
1	*1*	*1*	*1*	*1*



(c) Wie kann eine solche Schaltung auf 8 Bit erweitert werden? Beschreiben Sie das theoretische Vorgehen.
Lösung:

3. Entwerfen Sie analog zum Multiplexer (siehe Vorlesung) eine Schaltung, welche einen Input e, abhängig vom Steuersignal s_0 an Output o_0 oder Output o_1 weiterleitet.
Lösung:

e	s_0	o_0	o_1
0	*0*	*0*	*0*
0	*1*	*0*	*0*
1	*0*	*1*	*0*
1	*1*	*0*	*1*