TI II

Sommersemester 2010

PD Dr. Katinka Wolter

2. Aufgabenblatt

Ausgabe	Abgabe
3.5.10	14.05.10

Kontakt bei Fragen
Matthias Dräger, Viet Do, Marco Jeschke, Uwe Kuehn, Markus Rudolph
{mdraeger/do/kuehn/mjeschke/rudolph}@mi.fu-berlin.de

Problem 1: (6*1 = 6)

Geben Sie für jede der folgenden Zahlen deren Ziffernschreibweisen im Dezimal-, Dual-, Oktal- und Hexadezimal-System an.

a) $(2748)_{10}$

b) $(1010011011)_2$

c) $(52056)_8$

d) $(D1)_{16}$

Rechnen Sie die folgende Zahl ins 10er-System um:

e) $(2161)_7$

Rechnen Sie die folgende Zahl ins 7er-System um:

f) $(2161)_{10}$

Problem 2: (14*1/2 + 1 = 8)

Im folgenden sei die Wortlänge gleich 8 (d. h.: es wird mit Bytes gearbeitet).

a) i) Wie ist die Darstellung von -50 im Zweier-Komplement?
ii) Wie ist die Darstellung von -62 im Einer-Komplement?
iii) Wie ist die Darstellung von $+44$ im Einer-Komplement?
iv) Berechnen Sie $76-44$ im Einer-Komplement.
v) Berechnen Sie $116-29$ im Zweier-Komplement.
vi) Welche Zahl wird durch 183 im Zweier-Komplement dargestellt?
vii) Welche Zahl wird durch 186 im Einer-Komplement dargestellt?
viii) Berechnen Sie $44-76$ im Zweier-Komplement.
ix) Berechnen Sie $-116+29$ im Einer-Komplement.
x) Berechnen Sie $-35-(-100)$ im Zweier-Komplement.
xi) Berechnen Sie $-18-(-100)$ im Einer-Komplement.
xii) Wie ist die Darstellung von -88 im Einer-Komplement?
xiii) Wie ist die Darstellung von -45 im Zweier-Komplement?
xiv) Berechnen Sie $-47-16$ im Zweier-Komplement.

b) Interpretieren Sie die Beträge[1] der Ergebnisse des vorigen Aufgabenteils der Reihe nach als ASCII-Codierungen mit ungerader Parität[2]. Korrigieren Sie eventuell falsche Paritätsbits.

[1] Für den Betrag $|z|$ einer Zahl z gilt: $|z| = z$, falls $z \geqslant 0$; $|z| = -z$, falls $z < 0$.

[2] „ASCII"=Code ungerade Parität" bedeutet: In einem Byte wird das Paritätsbit P jeweils so gewählt (0 oder 1), dass sich eine ungerade Anzahl von Einsen im Byte ergibt, z. B.: $P1000111 \Rightarrow P = 1$, $P1001001 \Rightarrow P = 0$.

Problem 3: (3*2 = 6)

Gegeben sei die Boolesche Funktion[3]

$$f(x_1, x_2, x_3) \;=\; (x_1 \nleftrightarrow x_2) \cdot (\overline{x}_1 + x_3)$$

a) Stellen Sie die Funktionstabelle auf, und bestimmen Sie die einschlägigen Indizes.

b) Bestimmen Sie die DNF und die KNF von f.

c) Welche der beiden Darstellungen ist „günstiger“? Begründen Sie Ihre Antwort.

Problem 4: Gleitkomma-Darstellung (2+2+2+2+2+2=12)

Gehen Sie bei der binären Gleitkommadarstellung von 2-Byte großen Zahlen aus. Der Charakteristik stehen 4 Bit zur Verfügung, der Mantisse 11 und ein Vorzeichenbit. Geben Sie folgende Zahlen in binärer Gleitkomma- und in der normalen dezimalen Festkommadarstellung an, bzw. berechnen Sie den nächsten Nachbarn, falls die Zahl selbst nicht darstellbar ist:

a) die kleinste in dieser binären Gleitkommadarstellung darstellbare Zahl.

b) die größte in dieser binären Gleitkommadarstellung darstellbare Zahl.

c) $11/10$

d) $17 + 2^{-3}$

e) 7^7

f) $0,F_{16}$

Problem 5: Gleitkommazahlen nach IEEE 754 (2+4+2=8)

Das IEEE-754-Format für Gleitkommazahlen definiert (unter anderem) zwei Darstellungen: Bei 32-Bit Gleitkommazahlen werden 1 Bit für das Vorzeichen, 8 Bits für den Exponenten und 23 Bits für die Mantisse verwendet; bei 64-Bit Gleitkommazahlen sind es 1 Bit für das Vorzeichen, 11 Bits für den Exponenten und 52 Bits für die Mantisse (jeweils in dieser Reihenfolge). Der Exponent wird dabei in der Excess-127 bzw. Excess-1023 Darstellung gespeichert, der Bias ist also 127 bzw. 1023. Betrachten wir die 32-Bit Gleitkommazahlen genauer: Ist die Bitfolge der Zahldarstellung durch $VEM \in B^{32}$ mit $V \in B, E \in B^8, M \in B^{23}$ gegeben, so ist der Zahlenwert durch

$$(VEM)_{IEEE} = (-1)^V \cdot 2^{(E)_2 - 127} \cdot (1,M)_2$$

gegeben. Das führende 1-Bit vor dem Komma in der normalisierten Darstellung ist also implizit und wird nicht gespeichert (‘hidden bit’). Beispiel:

$$\begin{aligned}(1\ 01111111\ 10000000000000000000000)_{IEEE} &= (-1)^1 \cdot 2^{127-127} \cdot (1{,}1)_2 \\ &= -1 \cdot 1 \cdot 1{,}5 \quad = \quad -1{,}5\end{aligned}$$

Dabei sind die Bitfolgen $E = 0\ldots0$ und $E = 1\ldots1$ jeweils für Spezialfälle reserviert (‘not a number’, unendlich, denormalisierte Darstellung, Null).

[3]Zur Erinnerung: $x \nleftrightarrow y = (x+y) \cdot \overline{(x \cdot y)} = (x+y) \cdot (\overline{x} + \overline{y}) = (x \cdot \overline{y}) + (\overline{x} \cdot y)$ („XOR“).

a) Stellen Sie die folgenden Zahlen im IEEE-32Bit-Format dar: $z_1 = 1{,}125$, $z_2 = 1{,}5$, $z_3 = 16777216{,}0$.

b) Berechnen Sie $(z_1 + z_2) + z_3$ und $z_1 + (z_2 + z_3)$ im IEEE-32Bit-Format, wobei nach jeder Operation gerundet wird. Runden Sie (i) zur nächsten darstellbaren Zahl; (ii) auf die betragsmäßig nächst kleinere darstellbare Zahl (Abschneiden nicht darstellbarer Stellen). Bei welcher der vier Rechnungen erhalten Sie das genaueste Ergebnis?

c) Wie viele Normalisierungsschritte sind bei der Berechnung von $16777218 - 16777216$ im IEEE-Format notwendig?

Problem 6: Primzahltest MMIX (8)

Schreiben Sie ein Programm, das den Benutzer dazu auffordert eine Zahl von 1 bis 10.000 einzugeben. Das Programm soll dann berechnen, ob es sich bei dieser Zahl um eine Primzahl handelt oder nicht und das Ergebnis dem Benutzer mitteilen.