

RS 05 (HA) zum 19.11.2012

Paul Bienkowski, Hans Ole Hatzel

20. November 2012

1. Die einzelnen Werte sollen nach folgendem Schema übersetzt werden:

a1	a2	a3	a4
b1	b2	b3	

Um dies zu erreichen, werden die Eingabebits verschoben und mit ODER kombiniert. Um nur die ersten 8 bits zu verwenden, wird der Modulo 256 gebildet.

```
int b1 = (a1 << 2 | a2 >>> 4) % 256;
int b2 = (a2 << 4 | a3 >>> 2) % 256;
int b3 = (a3 << 6 | a4)      % 256;
```

2. a) Der Code hat $\frac{360^\circ}{15^\circ} = 24$ Codewörter.
b) **Schritt 1** (2 Codewörter):

0 1

Schritt 2 (4 Codewörter):

00 01
11 10

Schritt 3 (8 Codewörter):

000 001 011 [010]
[110] 111 101 100

Schritt 4 (6 Codewörter): Die Wörter in [Klammern] werden weggelassen, um aus 6 Codewörtern innerhalb von 2 Schritten $6 \cdot 2^2 = 24$ Codewörter zu erhalten.

000 001 011
111 101 100

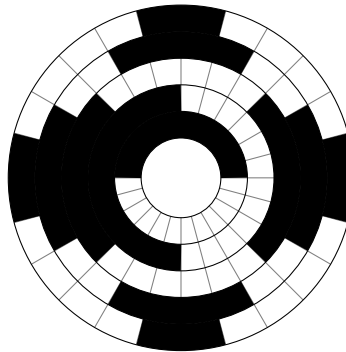
Schritt 5 (12 Codewörter):

0000 0001 0011 0111 0101 0100
1100 1101 1111 1011 1001 1100

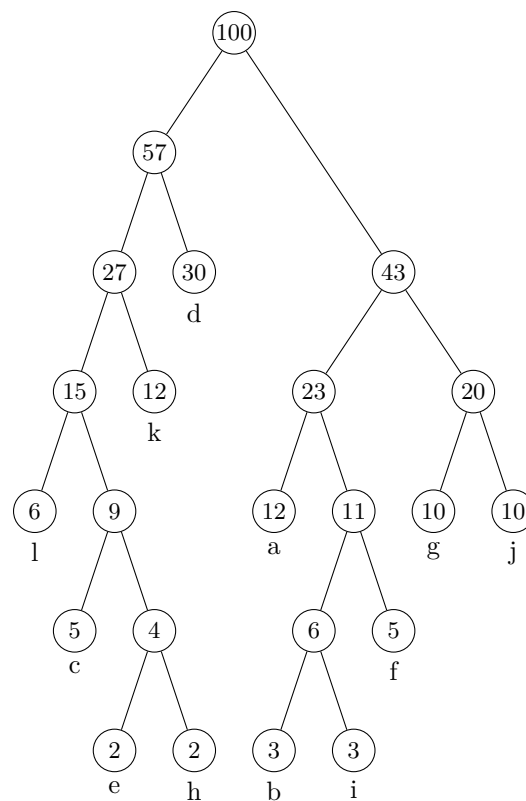
Schritt 6 (24 Codewörter):

00000 00001 00011 00111 00101 00100 01100 01101 01111 01011 01001 01100
11100 11001 11011 11111 11101 11100 10100 10101 10111 10011 10001 10000

Die folgende Winkelcodierscheibe stellt diesen Code dar:



3. a) Der entstehende Huffman-Baum kann so dargestellt werden (Wahrscheinlichkeits-Angaben werden zur Übersicht in Prozent notiert):



Legt man den jeweils linken Zweig als 0 und den jeweils rechten Zweig als 1 fest, ergibt sich folgende Codierung:

a	100	b	10100	c	00010
d	01	e	000110	f	1011
g	110	h	000111	i	10101
j	111	k	001	l	0000

b)

$$\begin{aligned}
 H &= -2 \cdot 0.02 \cdot \log_2(0.02) - 2 \cdot 0.3 \cdot \log_2(0.3) - 2 \cdot 0.05 \cdot \log_2(0.05) - 0.06 \cdot \log_2(0.06) \\
 &\quad - 2 \cdot 0.10 \cdot \log_2(0.10) - 2 \cdot 0.12 \cdot \log_2(0.12) - 0.30 \cdot \log_2(0.30) \\
 &\approx 3.12
 \end{aligned}$$

4. a) Eine unsystematischer Beispiel-Code:

0	1	2	3	4	5	6	7	8	9
0110	0010	0000	0100	0001	1000	1100	1001	0011	1111

$$H_0 = 10 \cdot 0.1 \cdot \log_2(2^4) = \log_2(16) = 4 \text{ Bit}$$

$$H = -10 \cdot 0.1 \cdot \log_2(0.1) = -\log_2(0.1) \approx 3.322 \text{ Bit}$$

$$R = 4 \text{ Bit} - 3.322 \text{ Bit} = 0.678 \text{ Bit}$$

- b) Es werden mindestens 7 bits (128 mögliche Codewörter) benötigt. Ein möglicher Code wäre die binäre Repräsentation, in der das $(7 - n)$ -te bit den Wert 2^n erhält.

0000000, 0000001, 0000010, 0000011, 0000100, ..., 1100011

$$H_0 = 100 \cdot \frac{1}{100} \cdot \log_2(2^8) = 8 \text{ Bit}$$

$$H = -100 \cdot \frac{1}{100} \cdot \log_2\left(\frac{1}{100}\right) \approx 6,6438526 \text{ Bit}$$

$$R = H_0 - H \approx 1,35615 \text{ Bit}$$

Redundanz pro Dezimalziffer:

$$\frac{1}{100} \cdot \log_2(2^8) - \frac{1}{100} \cdot \log_2\left(\frac{1}{100}\right) \approx 0,14643856$$

c)

d)