Musterlösung INCO SEP FS 2017

Aufgabe 1a: **1 1 1 0 1 0 1**

Aufgabe 1b: **$495D_H$**

Aufgabe 1c: **$1000\ 1001_{BCD}$**

Aufgabe 1d: **$\ldots 110000.0001_2$**

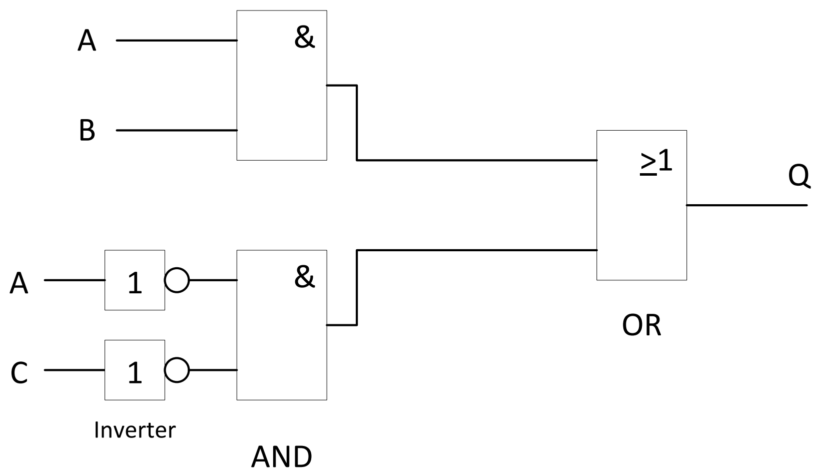
Aufgabe 2a: **$Q = (\neg A \wedge \neg B \wedge \neg C) \vee (\neg A \wedge B \wedge \neg C) \vee (A \wedge B \wedge \neg C) \vee (A \wedge B \wedge C)$**

Die Klammern sind unnötig, aber sie verbessern die Übersicht

Aufgabe 2b: **$Q = (A \wedge B) \vee (\neg A \wedge \neg C)$**

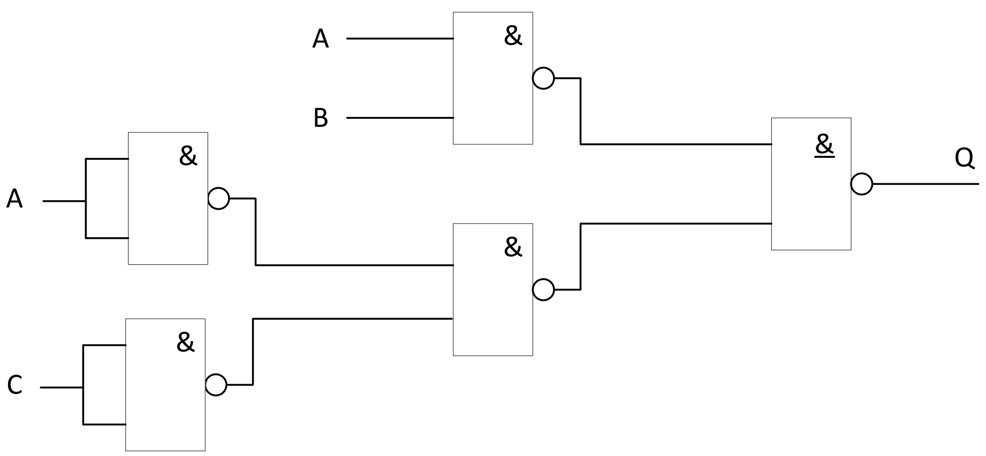
Aufgabe 2c:

Bool'sche Funktion als Gatterschaltung:



Aufgabe 2d:

Falls nur NAND Gatter verwendet sind:



Aufgabe 3a: **I(4) = 2.415 bit**

Aufgabe 3b: **I(5) ➔ unendlich**.

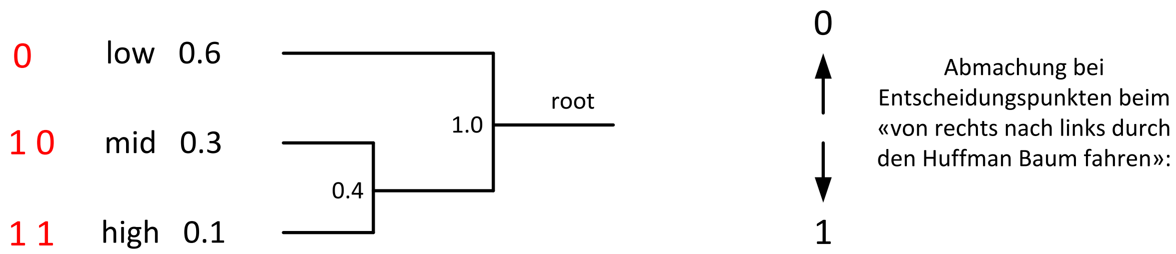
Die „5“ kann nie entstehen. *Theoretisch* ist der Informationsgehalt unendlich.

Aufgabe 3c: **H = 3.078 bit pro Symbol**

Aufgabe 3d: **R = 4 – H(X) = 0.922 bit pro Symbol**

Aufgabe 4a:

Huffman Baum zur Entwicklung des Huffman Codes:



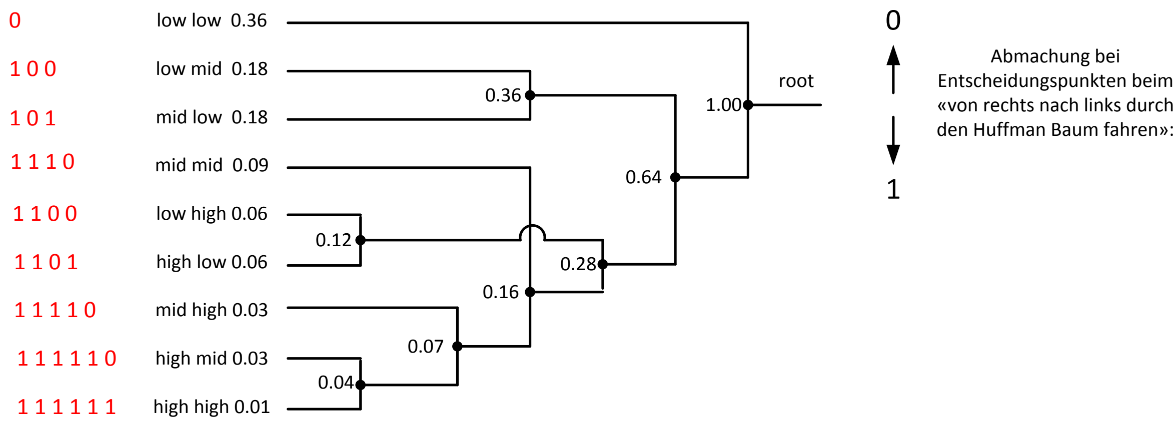
Entropie: **H = 1.295 bit pro Symbol**

Mittlere CW-Länge: **E[L] = 1.400 bit pro Symbol**

Redundanz: **R = E[L] – H = 0.105 bit pro Symbol**

Aufgabe 4b: Hierzu **Doppelsymbole** verwenden und dementsprechend einen Huffman tree für Doppelsymbole entwerfen:

Huffman Baum für Doppelsymbole:



Mittlere CW-Länge: **E[L] = 2.67 bit pro Doppel-Symbol; also 1.335 bit pro Symbol**

Redundanz **R = E[L] – H**

R = 1.335 bit/Symbol – 1.295 bit/Symbol = 0.04 bit/Symbol.

Aufgabe 4c/a: Die Entropie bleibt **gleich** gross.

Aufgabe 4c/b: Die mittlere CW-Länge wird **grösser**

Aufgabe 4c/c: Die Redundanz wird **grösser**

Aufgabe 5a: **LZ77-Token = (0,0,A) (1,1,B) (1,1,A) (5,4,B) (6,3,A) (7,3,B) …**

Aufgabe 5b: **LZ78 Codierung**:

Wörterbuch	Token		Wörterbuch	Token
0 null	---		7 AAA	4,A
1 A	0,A		8 BB	3,B
2 AB	1,B		9 BB ?	8,? (ignorierbar)
3 B	0,B			
4 AA	1,A			
5 ABB	2,B			
6 BA	3,A			

Aufgabe 5c:
Kompressionsrate R_a für LZ77: Ursprünglich 18 Zeichen, also 18 Byte = 144 bit
Anzahl LZ77-Token: 6
Pro Token (4 + 3 + 8) bit = 15 bit pro Token
Die 4 bit für den search buffer: 0000_2 …1111_2
Die 3 bit für den preview buffer: 000_2 …111_2
Die 8 bit fürs darauffolgende ASCII-Zeichen (1 Byte)

Total also 6 Token * 15 bit/Token = 90 bit; R_a = 90/144 = 0.625 < 1
Also ist bei LZ77 eine Kompression erreicht.

Kompressionsrate R_b für LZ78: Ursprünglich 18 Zeichen, also 18 Byte = 144 bit
Anzahl LZ78-Token: 8 (das neunte Token am Textende ist nicht bestimmbar)

Pro Token (8 + 8) bit = 16 bit pro Token
Die ersten 8 bit fürs Adressieren der 255 Wörterbucheinträge: 00000000_2 …11111111_2
Die 8 zweiten 8 bit fürs darauffolgende ASCII-Zeichen (1 Byte)
Total also 8 * 16 bit = 128 bit; R_b = 128/144 = 0.889 < 1

Also ist auch hier bei LZ78 eine Kompression erreicht.

Aufgabe 6a:

Zuerst die Kanalkapazität des BSC berechnen: $\mathbf{C_{BSC} = 1 - h(\varepsilon)}$

$\mathbf{h(\varepsilon) = -\varepsilon \cdot \log_2(\varepsilon) + (-1) \cdot (1- \varepsilon) \cdot \log_2(1- \varepsilon) = 0.045415}$

$\mathbf{C_{BSC} = 1 - h(\varepsilon) = 0.9546}$

Nun Bedingung: **Coderate = (K/N) = R < C_{BSC} = 0.9546**

Falls K = 250: $(250/N) < 0.9546$

$250 < N \cdot 0.9546$

$(250/0.9546) = 261.88 < N$

⇨ **N = 262 oder grösser: N muss also mindestens einen Wert von 262 haben.**

Aufgabe 6b

Mit dem (23,17, t = 2) code:

$P_{\text{null Fehler pro CW}} = 0.891109$

$P_{\text{ein Fehler pro CW}} = 0.102993$

$P_{\text{zwei Fehler pro CW}} = 0.005693$

Total: 0.999795 ➔ **P = 0.9998**

Aufgabe 7a: **K = 1; N = 5**

Linear? Falls u = 1: $CW_1 = [11111]$; falls u = 0: $CW_0 = [00000]$

Nun bitweise EXOR verknüpfen:
CW_1 exor CW_1 = [00000] = CW_0
CW_0 exor CW_0 = [00000] = CW_0
CW_0 exor CW_1 = [11111] = CW_1
Also: **Ja, der Code ist linear**, da immer ein CW entsteht bei Modulo zwei Verknüpfung.
Systematisch? **Ja**.
Zyklisch? **Ja**.

Aufgabe 7b:**dmin = 5**
Anzahl erkennbare Fehler = dmin - 1 = 4
Anzahl korrigierbare Fehler = $[(dmin – 1)/2]_{Floor}$ = 2
Aufgabe 7c: CW = $\underline{u} \cdot G$
G = [11111]
Aufgabe 7d: Parity Check Matrix H ist gesucht:
Generatormatrix $G = [P\ I_K]$ wobei G = [1111 1]; N=5; K=1 ➔ $I_k = 1$
$H = [I_{N\text{-}K}\ P^T]$

$$I_{N\text{-}K} = \begin{bmatrix} 1\,0\,0\,0 \\ 0\,1\,0\,0 \\ 0\,0\,1\,0 \\ 0\,0\,0\,1 \end{bmatrix} \qquad P^T = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \qquad H = \begin{bmatrix} 1\,0\,0\,0\,1 \\ 0\,1\,0\,0\,1 \\ 0\,0\,1\,0\,1 \\ 0\,0\,0\,1\,1 \end{bmatrix}$$

==============

Aufgabe 7e:

Fürs Aufstellen der Syndromtabelle: Es gilt $\underline{S} = \underline{e} \cdot H^T$

Tabelle für **null** Bitfehler oder **einen** Bitfehler (vgl. Fehlervektor)

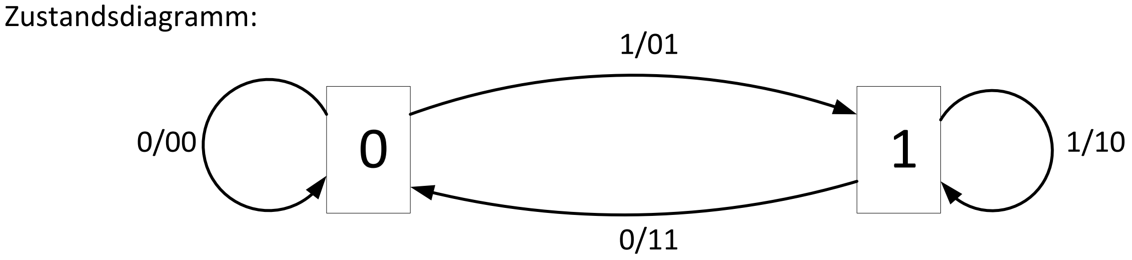
$\underline{S}$	Fehlervektor $\underline{e}$
0 0 0 0	0 0 0 0 0
1 0 0 0	1 0 0 0 0
0 1 0 0	0 1 0 0 0
0 0 1 0	0 0 1 0 0
0 0 0 1	0 0 0 1 0
1 1 1 1	0 0 0 0 1

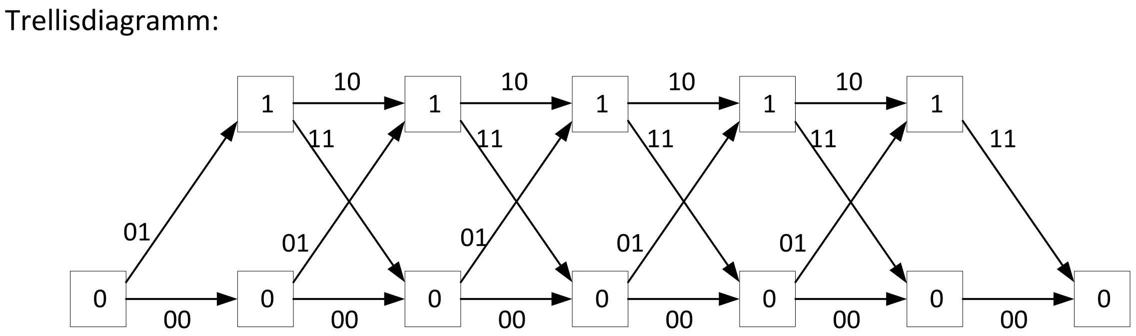
Tabelle für **zwei** Bitfehler (vgl. Fehlervektor)

$\underline{S}$	Fehlervektor $\underline{e}$
1 1 0 0	1 1 0 0 0
1 0 1 0	1 0 1 0 0
1 0 0 1	1 0 0 1 0
0 1 1 1	1 0 0 0 1
0 1 1 0	0 1 1 0 0
0 1 0 1	0 1 0 1 0
1 0 1 1	0 1 0 0 1
0 0 1 1	0 0 1 1 0
1 1 0 1	0 0 1 0 1
1 1 1 0	0 0 0 1 1

Aufgaben 8a und 8b:

Zustandsdiagramm:





Aufgabe 8c: **d_{free} = 3** Es sind drei Einsen bei einem Minimalumweg, der vom Pfad 00-00-00-00-00-00 abweicht.

Aufgabe 8d: **CW = 01 11 01 10 11 00** (vom tail bit stammend)