Übung zur Vorlesung

Digitale Medien

Vorlesung: Heinrich Hußmann
Übung: Renate Häuslschmid, Hanna Schneider

Ludwig-Maximilians-Universität München
Wintersemester 2015/16

LZW-Komprimierung

Idee: Nicht einzelne Zeichen werden günstig kodiert, sondern ganze Zeichenketten *(Wörterbuch-Kompression*).

Beispiel: **abcdabcdabcdabcddd**

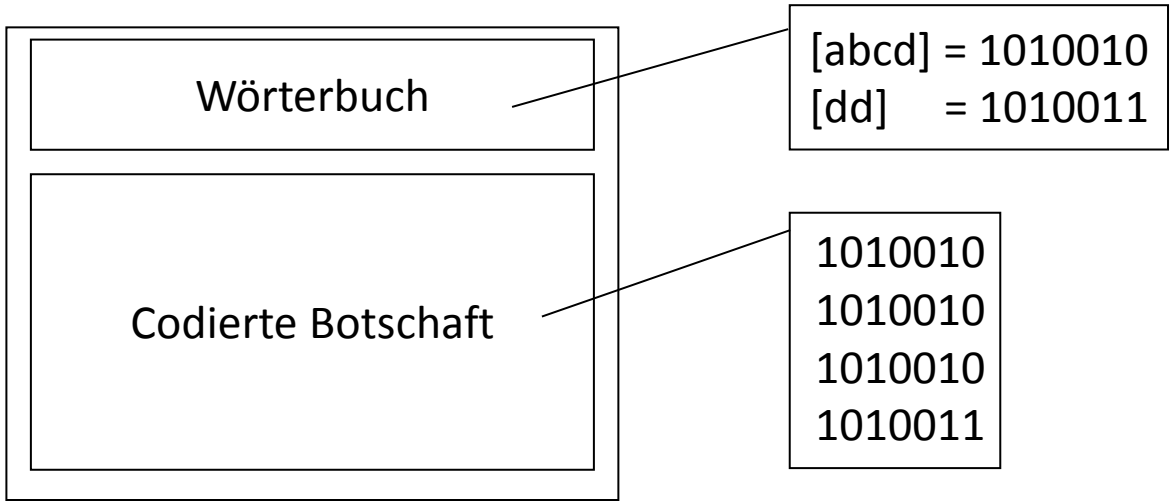
Huffman: [a][b][c][d][a][b][c][d][a][b][c][d][a][b][c][d][d][d]
→18 Symbole

Wörterbuch: [abcd][abcd][abcd][abcd][dd]
→ 5 Symbole

Nachteil: Mehr Symbole nötig (jeder Eintrag ins Wörterbuch ist ein Symbol)

LZW-Komprimierung

Weiterer Nachteil: Empfänger muss Wörterbuch kennen, um die Nachricht dekodieren zu können.



Aber: Muss das Wörterbuch wirklich mit übertragen werden?

Bei der LZW-Komprimierung wird das Wörterbuch *während* der Dekodierung aufgebaut!

LZW-Komprimierung

Ausgegangen wird von einem Grundwörterbuch (z.B. ASCII-Code)

a	97	h	104	o	111	v	118
b	98	i	105	p	112	w	119
c	99	j	106	q	113	x	120
d	100	k	107	r	114	y	121
e	101	l	108	s	115	z	122
f	102	m	109	t	116		
g	103	n	110	u	117		

Alogrithmus:

SeqChar p = < NächstesEingabezeichen >;
Char k = NächstesEingabezeichen;
Wiederhole:
    Falls p & < k > in Tabelle enthalten
        dann p = p & < k >
        sonst trage p & <k> neu in Tabelle ein
            (und erzeuge neuen Index dafür);
            Schreibe Tabellenindex von p auf Ausgabe;
            p = < k >;
    Ende Fallunterscheidung;
    k = NächstesEingabezeichen;
solange bis Eingabeende

Schreibe Tabellenindex von p auf Ausgabe;

LZW-Komprimierung

Beispiel: **labala**

Algorithmus

SeqChar p = < NächstesEingabezeichen >;
Char k = NächstesEingabezeichen;
Wiederhole:
    Falls p & < k > in Tabelle enthalten
        dann p = p & < k >
        sonst trage p & <k> neu in Tabelle ein
            (und erzeuge neuen Index dafür);
            Schreibe Tabellenindex von p auf Ausgabe;
            p = < k >;
    Ende Fallunterscheidung;
    k = NächstesEingabezeichen;
solange bis Eingabeende
Schreibe Tabellenindex von p auf Ausgabe;

Lesen (k)	Codetabelle schreiben (p & <k>)	Ausgabe	Puffer füllen (p)
			<l>
a	<la>, 256	108 (l)	<a>
b	<ab>, 257	97 (a)	<b>
a	<ba>, 258	98 (b)	<a>
l	<al>, 259	97 (a)	<l>
a			<la>
		256 (la)	<>

a	97	h	104	o	111	v	118
b	98	i	105	p	112	w	119
c	99	j	106	q	113	x	120
d	100	k	107	r	114	y	121
e	101	l	108	s	115	z	122
f	102	m	109	t	116		
g	103	n	110	u	117		

LZW-Komprimierung

Beispiel: **ballaballala**

Lesen (k)	Codetabelle schreiben (p & <k>)	Ausgabe	Puffer füllen (p)

SeqChar p = < NächstesEingabezeichen >;
Char k = NächstesEingabezeichen;
Wiederhole:
    Falls p & < k > in Tabelle enthalten
        dann p = p & < k >
        sonst trage p & <k> neu in Tabelle ein
            (und erzeuge neuen Index dafür);
            Schreibe Tabellenindex von p auf Ausgabe;
            p = < k >;
    Ende Fallunterscheidung;
    k = NächstesEingabezeichen;
solange bis Eingabeende
Schreibe Tabellenindex von p auf Ausgabe;

a	97	h	104	o	111	v	118
b	98	i	105	p	112	w	119
c	99	j	106	q	113	x	120
d	100	k	107	r	114	y	121
e	101	l	108	s	115	z	122
f	102	m	109	t	116		
g	103	n	110	u	117		

LZW-Komprimierung

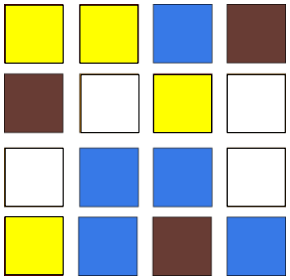
Beispiel: **ballaballala**

Lesen (k)	Codetabelle schreiben (p & <k>)	Ausgabe	Puffer füllen (p)
			<b>
a	<ba>, 256	98 (b)	<a>
l	<al>, 257	97 (a)	<l>
l	<ll>, 258	108 (l)	<l>
a	<la>, 259	108 (l)	<a>
b	<ab>, 260	97 (a)	<b>
a			<ba>
l	<bal>, 261	256 (ba)	<l>
l			<ll>
a	<lla>, 262	258 (ll)	<a>
l			<al>
a	<ala>, 263	257 (al)	<a>
		97 (a)	<>

→ 98 – 97 – 108 – 108 – 97 – 256 – 258 – 257 - 97

Aufgabe 1

Folgendes Bild mit einer Auflösung von 4 mal 4 Bildpunkten ist vorgegeben:



	Dezimal	Huffman
Gelb (ge)	0	
Blau (bl)	1	
Braun (br)	2	
Weiß (we)	3	

Die vorhandenen Farben sollen mit der obenstehenden Tabelle nach den Werten die in der Spalte „Dezimal“ angegeben sind codiert werden. Das Bild wird zeilenweise von links nach rechts verarbeitet, d.h. nachdem die vier Pixel in Zeile 1 verarbeitet wurden, wird mit dem ersten Pixel in Zeile 2 weitergemacht.

a) Codieren Sie das komplette Bild. Gehen Sie dabei entsprechend dem in der Vorlesung behandelten Algorithmus vor. Verwenden Sie zur Darstellung der Zwischenschritte des Algorithmus eine Tabelle mit den Spalten: „Lesen“, „Codetabelle schreiben“, „Ausgabe“ und „Puffer füllen“. Der erste neue Tabelleneintrag ist bei Index 4.

b) Codieren Sie das Bild dann in Huffman-Code Pixel für Pixel.

c) Konvertieren Sie das Ergebnis aus Aufgabe a) ins Binärformat (für LZW gilt diesmal eine 4-Bit Darstellung) und bestimmen Sie, welches der beiden Codierungsverfahren für dieses spezielle Bild besser geeignet ist (d.h. den kürzeren Code liefert).

Lösung zu Aufgabe 1a)

Lesen	Codetabelle schreiben	Ausgabe	Puffer füllen
			<ge>
ge	<gege>, 4	0 (ge)	<ge>
bl	<gebl>, 5	0 (ge)	<bl>
br	<blbr>, 6	1 (bl)	
br	<brbr>, 7	2 (br)	
we	<brwe>, 8	2 (br)	<we>
ge	<wege>, 9	3 (we)	<ge>
we	<gewe>, 10	0 (ge)	<we>
we	<wewe>, 11	3 (we)	<we>
bl	<webl>, 12	3 (we)	<bl>
bl	<blbl>, 13	1 (bl)	<bl>
we	<blwe>, 14	1 (bl)	<we>
ge			<wege>
bl	<wegebl>, 15	9 (wege)	<bl>
br			<blbr>
bl	<blbrbl>, 16	6 (blbr)	<bl>
EOF		1 (bl)	<>

gegeblbr

brwegewe

weblblwe

geblbrbl

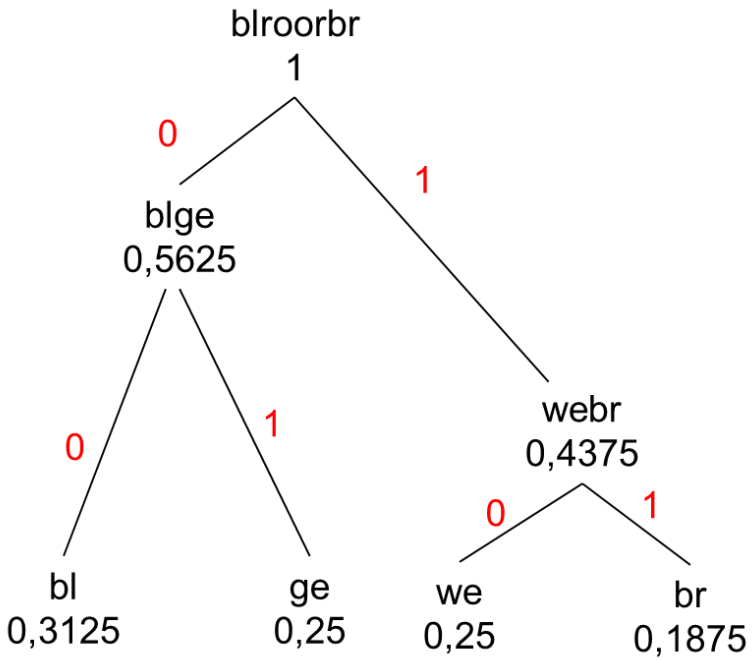
Lösung zu Aufgabe 1b)

Zeichen	ro	bl	br	or
Häufigkeit	0,25	0,3125	0,1875	0,25

Ge	01
Ge	01
Bl	00
Br	11
Br	11
We	10
Ge	01
We	10
We	10
Bl	00
Bl	00
We	10
Ge	01
Bl	00
Br	11
Bl	00

01 01 00 11 11 10 01 10 10 00 00 10 01 00 11 00

Insgesamt: 32 Bit



Lösung zu Aufgabe 1c)

LZW: 0 0 1 2 2 3 0 3 3 1 1 9 6 1

0	0	1	2	2	3	0	3	3	1	1	9	6	1
0000	0000	0001	0010	0010	0011	0000	0011	0011	0001	0001	1001	0110	0001

Insgesamt: 4 x 14 = 56 Bit

Huffman: 01 01 00 11 11 10 01 10 10 00 00 10 01 00 11 00

Insgesamt: 32 Bit

→ Huffman kürzer und damit besser! (Gilt allerdings nur für dieses Beispiel)

Aufgabe 2

Gegeben sei die Codetabelle des ASCII-Zeichensatzes (siehe Seite 3). Ab Index 256 können neue Einträge erfolgen. Weiterhin gegeben sei folgende Nachricht: **bobobobowebewe**

a) Codieren Sie die Nachricht mittels LZW-Codierung. Gehen Sie dabei entsprechend dem in der Vorlesung behandelten Algorithmus vor. Verwenden Sie zur Darstellung der Zwischenschritte des Algorithmus eine Tabelle mit den Spalten: „Lesen“, „Codetabelle schreiben“, „Ausgabe“ und „Puffer füllen“. Das Wörterbuch wird mit der oben angegebenen Codetabelle initialisiert.

b) Codieren Sie die Nachricht zusätzlich im unkomprimierten ASCII-Format. Wandeln Sie dann beide Codierungsformen (LZW und ASCII) in die jeweiligen Binärdarstellungen. Benutzen Sie für den ASCII-Code 8-Bit Darstellungen für jedes Zeichen (siehe Tabelle im Anhang) und für LZW 9-Bit Darstellungen (da Zahlen > 255). Um wie viel Prozent schrumpft die Nachricht durch Verwendung von LZW?

Lösung zu Aufgabe 2a)

Lesen	Codetabelle schreiben	Ausgabe	Puffer füllen
			<b>
o	<bo>, 256	98 (b)	<o>
b	<ob>, 257	111 (o)	<b>
o			<bo>
b	<bob>, 258	256 (bo)	<b>
o			<bo>
b			<bob>
o	<bobo>, 259	258 (bob)	<o>
w	<ow>, 260	111 (0)	<w>
e	<we>, 261	119 (w)	<e>
b	<eb>, 262	101 (e)	<b>
e	<be>, 263	98 (b)	<e>
w	<ew>, 264	101 (e)	<w>
e			<we>
		261 (we)	<>

Lösung zu Aufgabe 2b)

Konvertieren nach ASCII:

b	o	b	o	b	o	b	o	w	e	b	e	w	e
98	111	98	111	98	111	98	111	119	101	98	101	119	101
01 10 00 10	01 10 11 11	01 10 00 10	01 10 11 11	01 10 00 10	01 10 11 11	01 10 00 10	01 10 11 11	01 11 01 11	01 10 01 01	01 10 00 10	01 10 01 01	01 11 01 11	01 10 01 01

01100010 01101111 01100010 01101111 01100010 01101111 01100010
01101111 01110111 01100101 01100010 01100101 01110111 01100101

Insgesamt: 14 x 8 = 112 Bit

Lösung zu Aufgabe 2b)

98	0 0110 0010
111	0 0110 1111
256	1 0000 0000
258	1 0000 0010
111	0 0110 1111
119	0 0111 0111
101	0 0110 0101
98	0 0110 0010
101	0 0110 0101
261	1 0000 0101

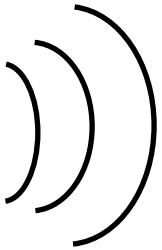
001100010 001101111 100000000
100000010 001101111 001110111
001100101 001100010 001100101
100000101

Insgesamt:
10 x 9 Bit = 90 Bit

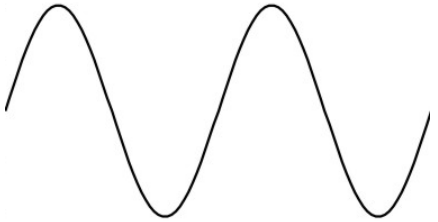
Ersparnis:
112 - 90 = 22 Bit
22 / 112 = 19,6%

Digitalisierung

physikalische Signale



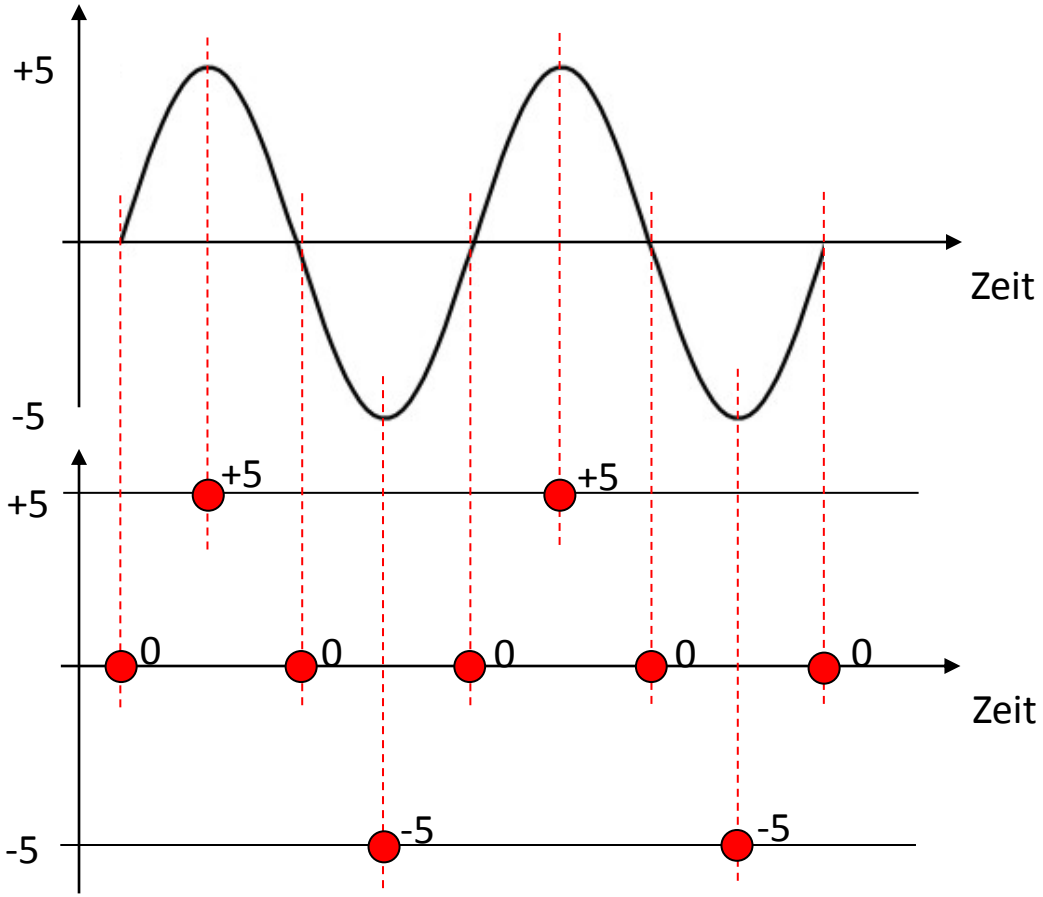
elektrische Signale



digitale Signale

10100100101111010111101

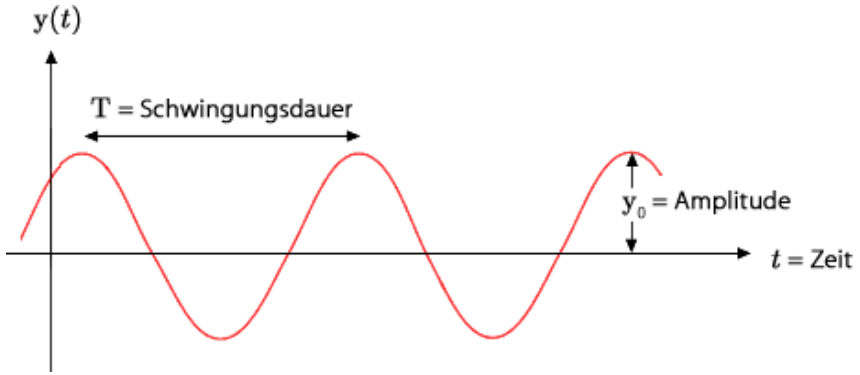
Digitalisierung



Wellen

Eine Welle wird beschrieben durch:

A	Amplitude (z.B. Lautstärke dB)
c	Ausbreitungsgeschwindigkeit
f	Frequenz (f = 1/T)
	bzw.
λ	Wellenlänge (*λ = c / f*)



Frequenz *f* wird in Hertz (*Hz*) gemessen.

1 Hz = 1 Schwingung / Sekunde

Schwingungsdauer T ist die Zeit bis sich das Wellenmuster wiederholt.

Umrechnung

…von Schwingungsdauer und Frequenz.

- Welle mit Schwingungsdauer **T = 2 ms**. Wie hoch ist die Frequenz f ?

T = 2 ms = 2 * 10^{-3} s = 0,002 s

f = 1 / T = 1 / 0,002 = **500 Hz**

- Welle mit Frequenz **f = 500 MHz**. Bestimme Schwingungsdauer T.

f = 500 MHz = 500.000.000 Hz

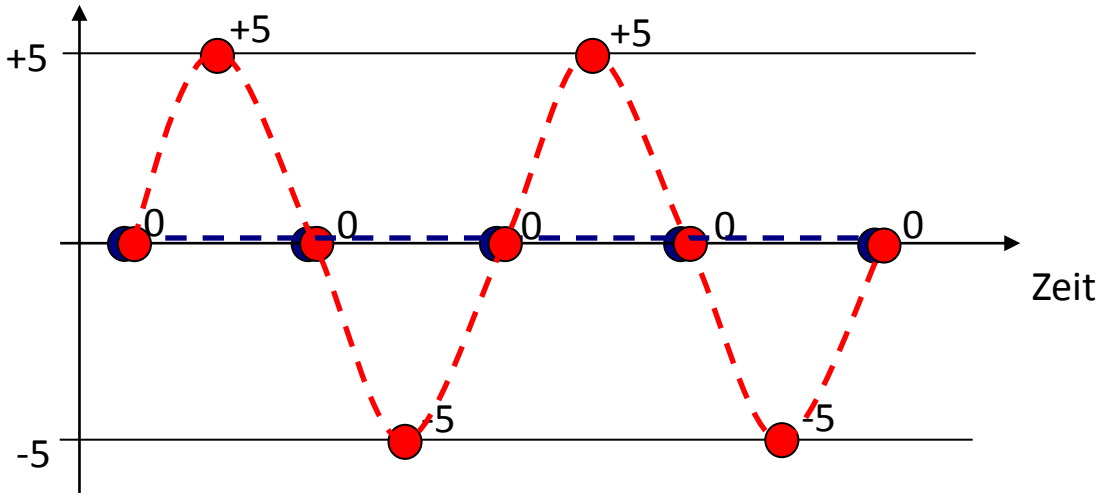
f = 1 / T → T = 1 / f

T = 1 / 500.000.000 Hz

= 0,000000002 s = 0,000002 ms = **0,002 µs**

1 ms = 1 * 10^{-3} s = 0,001 s
1 µs = 1 * 10^{-6} s = 0,000001 s

Nyquist Theorem



Aus den Samples wird versucht, das Originalsignal zu rekonstruieren. Je höher die Abtastrate desto besser die Rekonstruktion.

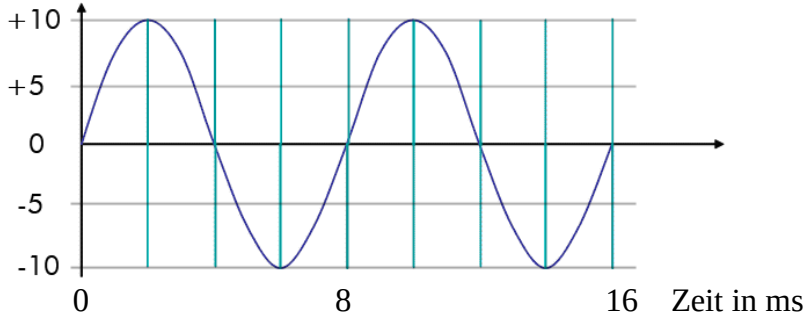
Doppelte Frequenz der Welle (blau) reicht nicht! Abtastrate f_A muss größer als die doppelte Frequenz f sein:

$$f_A > 2 * f \qquad \text{(Nyquist-Theorem)}$$

z.B. Audio-CD: Frequenz f maximal 20 kHz, Abtastrate 44,1 kHz

Aufgabe 3

In folgender Abbildung ist ein 16 Millisekunden langer Ausschnitt eines periodischen Signals dargestellt. Die linke Skala bezeichnet die Amplitude des Signals:



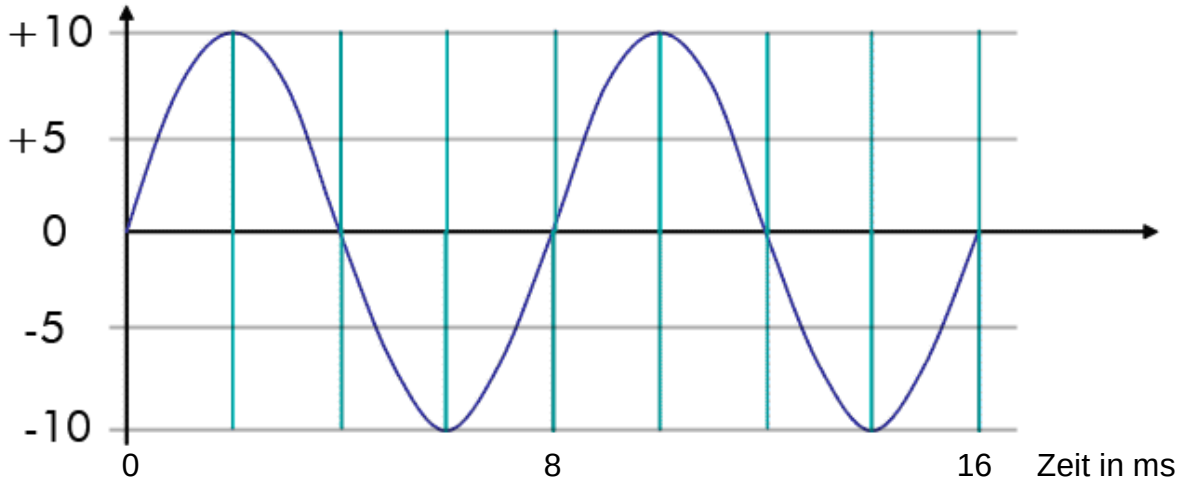
a) Welche Frequenz in Hertz hat das Signal?

b) Das Signal wird mit einer Rate von 250 Hz abgetastet. Geben Sie für jedes Sample, das dadurch innerhalb der 16ms erstellt wird, den Zeitpunkt der Messung in ms und die gemessene Amplitude an. Die Messungen beginnen zum Zeitpunkt 0.

c) Die Abtastrate wird jetzt auf 0,5 kHz erhöht. Geben Sie wiederum die Anzahl der Samples, deren Zeitpunkte und gemessene Amplituden an.

d) Mit welcher der beiden Abtastraten kann das Signal wieder rekonstruiert werden? Wie groß muss die Abtastrate mindestens sein (mit Begründung)?

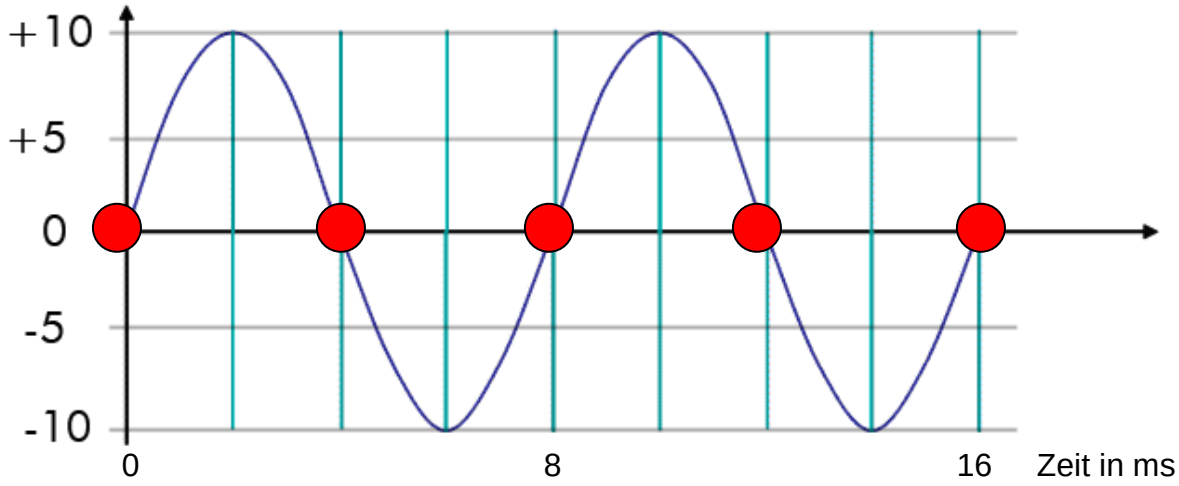
Lösung zu Aufgabe 3a)



Signal wiederholt sich alle 8 ms.
Frage: Wie oft wiederholt sich das Signal innerhalb einer Sekunde?

8 ms * *x* Hertz = 1000 ms (eine Sekunde)
x = (1000 / 8) Hz
=> Signal hat eine Frequenz von 125 Hertz (125 Schwingungen pro Sekunde)

Lösung zu Aufgabe 3b)

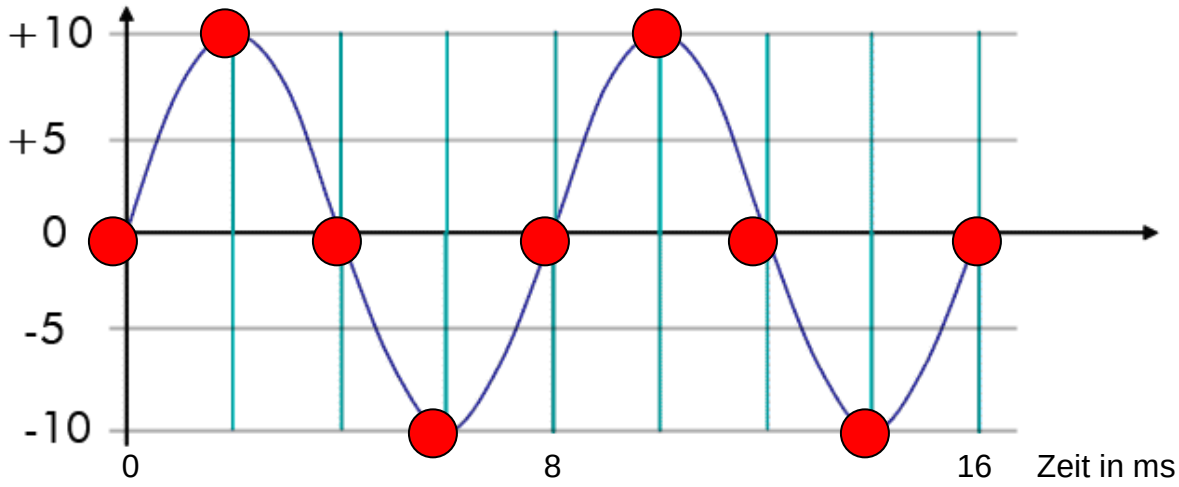


Messungen mit 250Hz

T´= 1/250 s = 0,004 s = 4ms, also eine Messung pro 4 ms.

Messungszeitpunkt (ms)	0	4	8	12	16
Gemessene Amplitude	0	0	0	0	0

Lösung zu Aufgabe 3c)



Messungen mit 0,5 kHz = 500 Hz

x ms * 500 Hz = 1000 ms

x = (1000 / 500) ms

=> Abtastung erfolgt alle 2 ms

Messungszeitpunkt (ms)	0	2	6	6	8	10	12	14	16
Gemessene Amplitude	0	+10	0	-10	0	+10	0	-10	0

Lösung zu Aufabge 3d)

Mit der ersten Abtastung ist eine Rekonstruktion nicht möglich,
mit der zweiten schon.

Die Abtastrate muss **größer** als die doppelte Frequenz des Signals sein,
um eine Rekonstruktion zu ermöglichen (Nyquist-Theorem).
In unserem Beispiel: f = 125 Hz
Damit muss die Abtastrate größer als 250 Hz sein.

Übungsblatt 3

- Übungsblatt 3:
  https://www.medien.ifi.lmu.de/lehre/ws1516/dm/

- Abgabe bis Freitag den 13.11.2015, 09:00 Uhr morgens in UniWorX

Beispiel Labala

LZW KOMPRIMIERUNG SCHRITT FÜR SCHRITT

LZW-Komprimierung

labala

Lesen (k)	Codetabelle schreiben (p & <k>)	Ausgabe	Puffer füllen (p)

SeqChar p = < NächstesEingabezeichen >;
Char k = NächstesEingabezeichen;
Wiederhole:
    Falls p & < k > in Tabelle enthalten
        dann p = p & < k >
        sonst trage p & <k> neu in Tabelle ein
            (und erzeuge neuen Index dafür);
            Schreibe Tabellenindex von p auf Ausgabe;
            p = < k >;
    Ende Fallunterscheidung;
    k = NächstesEingabezeichen;
solange bis Eingabeende
Schreibe Tabellenindex von p auf Ausgabe;

LZW-Komprimierung

labala

Lesen (k)	Codetabelle schreiben (p & <k>)	Ausgabe	Puffer füllen (p)
			<l>
a			

SeqChar p = < NächstesEingabezeichen >;
Char k = NächstesEingabezeichen;
Wiederhole:
    Falls p & < k > in Tabelle enthalten
        dann p = p & < k >
        sonst trage p & <k> neu in Tabelle ein
            (und erzeuge neuen Index dafür);
            Schreibe Tabellenindex von p auf Ausgabe;
            p = < k >;
    Ende Fallunterscheidung;
    k = NächstesEingabezeichen;
solange bis Eingabeende
Schreibe Tabellenindex von p auf Ausgabe;

LZW-Komprimierung

labala

Lesen (k)	Codetabelle schreiben (p & <k>)	Ausgabe	Puffer füllen (p)
			<l>
a	LEER, (p(l)&k(a)) als la nicht enthalten		

SeqChar p = < NächstesEingabezeichen >;
Char k = NächstesEingabezeichen;
Wiederhole:
    Falls p & < k > in Tabelle enthalten
        dann p = p & < k >
        sonst trage p & <k> neu in Tabelle ein
            (und erzeuge neuen Index dafür);
            Schreibe Tabellenindex von p auf Ausgabe;
            p = < k >;
    Ende Fallunterscheidung;
    k = NächstesEingabezeichen;
solange bis Eingabeende
Schreibe Tabellenindex von p auf Ausgabe;

LZW-Komprimierung

labala

Lesen (k)	Codetabelle schreiben (p & <k>)	Ausgabe	Puffer füllen (p)
			<l>
a	**<la>, 256**	**108 (l)**	**<a>**

SeqChar p = < NächstesEingabezeichen >;
Char k = NächstesEingabezeichen;
Wiederhole:
    Falls p & < k > in Tabelle enthalten
        dann p = p & < k >
        sonst trage p & <k> neu in Tabelle ein
            (und erzeuge neuen Index dafür);
            Schreibe Tabellenindex von p auf Ausgabe;
            p = < k >;
    Ende Fallunterscheidung;
    k = NächstesEingabezeichen;
solange bis Eingabeende
Schreibe Tabellenindex von p auf Ausgabe;

LZW-Komprimierung

labala

Lesen (k)	Codetabelle schreiben (p & <k>)	Ausgabe	Puffer füllen (p)
			<l>
a	<la>, 256	108 (l)	<a>
b			

SeqChar p = < NächstesEingabezeichen >;
Char k = NächstesEingabezeichen;
Wiederhole:
    Falls p & < k > in Tabelle enthalten
        dann p = p & < k >
        sonst trage p & <k> neu in Tabelle ein
            (und erzeuge neuen Index dafür);
            Schreibe Tabellenindex von p auf Ausgabe;
            p = < k >;
    Ende Fallunterscheidung;
    k = NächstesEingabezeichen;
solange bis Eingabeende
Schreibe Tabellenindex von p auf Ausgabe;

LZW-Komprimierung

labala

Lesen (k)	Codetabelle schreiben (p & <k>)	Ausgabe	Puffer füllen (p)
			<l>
a	<la>, 256	108 (l)	<a>
b	<ab>, 257	97 (a)	<b>
a	<ba>, 258	98 (b)	<a>
l	<al>, 259	97 (a)	<l>
a			<la>
		256 (la)	<>

```
SeqChar p = < NächstesEingabezeichen >;
Char k = NächstesEingabezeichen;
Wiederhole:
        Falls p & < k > in Tabelle enthalten
                dann p = p & < k >
                sonst trage p & <k> neu in Tabelle ein
                        (und erzeuge neuen Index dafür);
                        Schreibe Tabellenindex von p auf Ausgabe;
                        p = < k >;
        Ende Fallunterscheidung;
        k = NächstesEingabezeichen;
solange bis Eingabeende
Schreibe Tabellenindex von p auf Ausgabe;
```

LZW-Komprimierung

labala

Lesen (k)	Codetabelle schreiben (p & <k>)	Ausgabe	Puffer füllen (p)
			<l>
a	<la>, 256	108 (l)	<a>
b	<ab>, 257	97 (a)	<b>
a	<ba>, 258	98 (b)	<a>
l	<al>, 259	97 (a)	<l>
a			<la>
		256 (la)	<>

```
SeqChar p = < NächstesEingabezeichen >;
Char k = NächstesEingabezeichen;
Wiederhole:
        Falls p & < k > in Tabelle enthalten
                dann p = p & < k >
                sonst trage p & <k> neu in Tabelle ein
                        (und erzeuge neuen Index dafür);
                        Schreibe Tabellenindex von p auf Ausgabe;
                        p = < k >;
        Ende Fallunterscheidung;
        k = NächstesEingabezeichen;
solange bis Eingabeende
Schreibe Tabellenindex von p auf Ausgabe;
```