

Die Mathematik in der CD

Gerhard Hiß
Lehrstuhl D für Mathematik
RWTH Aachen

Lehrstuhl D für Mathematik
RWTH Aachen

Kreisgymnasium Heinsberg
15.05.(Kalte Sophie)2006

Ein **Code** ist eine künstliche Sprache zum Speichern oder Übertragen von Daten.

Codes: Definition und Aufgaben

Ein **Code** ist eine künstliche Sprache zum Speichern oder Übertragen von Daten.

[Ein Code dient **nicht** der Verschlüsselung (Geheimhaltung) von Nachrichten.]

Ein **Code** ist eine künstliche Sprache zum Speichern oder Übertragen von Daten.

[Ein Code dient **nicht** der Verschlüsselung (Geheimhaltung) von Nachrichten.]

Aufgaben von Codes: **Erkennen** und **Korrigieren** von Übertragungsfehlern.

Codes: Definition und Aufgaben

Ein **Code** ist eine künstliche Sprache zum Speichern oder Übertragen von Daten.

[Ein Code dient **nicht** der Verschlüsselung (Geheimhaltung) von Nachrichten.]

Aufgaben von Codes: **Erkennen** und **Korrigieren** von Übertragungsfehlern.

Anwendungen: CDs, Satelliten-Kommunikation, Internet, etc.

Ein **Code** ist eine künstliche Sprache zum Speichern oder Übertragen von Daten.

[Ein Code dient **nicht** der Verschlüsselung (Geheimhaltung) von Nachrichten.]

Aufgaben von Codes: **Erkennen** und **Korrigieren** von Übertragungsfehlern.

Anwendungen: CDs, Satelliten-Kommunikation, Internet, etc.

Methode der Fehlererkennung und -korrektur: Übertragung **redundanter** Zeichen.

Fleiheit

Korrektur möglich

Fleiheit

Korrektur möglich

Schile

Korrektur nicht möglich

Alphabet A : endliche Menge von Symbolen

Alphabet A : endliche Menge von Symbolen

Beispiele:

$\{0, 1\}$ (binäres Alphabet)

$\{a, b, \dots, z, A, B, \dots, Z, ?, +, \dots\}$

Alphabet A : endliche Menge von Symbolen

Beispiele:

$\{0, 1\}$ (binäres Alphabet)

$\{a, b, \dots, z, A, B, \dots, Z, ?, +, \dots\}$

Code über A : Menge von Wörtern mit Alphabet A

Alphabet A : endliche Menge von Symbolen

Beispiele:

$\{0, 1\}$ (binäres Alphabet)

$\{a, b, \dots, z, A, B, \dots, Z, ?, +, \dots\}$

Code über A : Menge von Wörtern mit Alphabet A

I. A . gehören nicht alle denkbaren Wörter zum Code.

Alphabet A : endliche Menge von Symbolen

Beispiele:

$\{0, 1\}$ (binäres Alphabet)

$\{a, b, \dots, z, A, B, \dots, Z, ?, +, \dots\}$

Code über A : Menge von Wörtern mit Alphabet A

I. A . gehören nicht alle denkbaren Wörter zum Code.

Beschreibung:

Sprache: Wörterbuch

Codierungstheorie: andere, kompaktere Beschreibung

Alphabet A : endliche Menge von Symbolen

Beispiele:

$\{0, 1\}$ (binäres Alphabet)

$\{a, b, \dots, z, A, B, \dots, Z, ?, +, \dots\}$

Code über A : Menge von Wörtern mit Alphabet A

I. A . gehören nicht alle denkbaren Wörter zum Code.

Beschreibung:

Sprache: Wörterbuch

Codierungstheorie: andere, kompaktere Beschreibung

Block-Code: alle Codewörter sind gleich lang

Beispiel: Der ISBN-Code

ISBN-Code (International Standard Book Number):

$$A = \{0, 1, \dots, 9, X\} \quad (X \hat{=} 10)$$

Beispiel: Der ISBN-Code

ISBN-Code (International Standard Book Number):

$$A = \{0, 1, \dots, 9, X\} \quad (X \cong 10)$$

Ein Wort des ISBN-Codes besteht aus 10 Symbolen (aus A), $z_1 z_2 \cdots z_9 z_{10}$, wobei die ersten neun aus $\{0, 1, \dots, 9\}$ sind.

Beispiel: Der ISBN-Code

ISBN-Code (International Standard Book Number):

$$A = \{0, 1, \dots, 9, X\} \quad (X \cong 10)$$

Ein Wort des ISBN-Codes besteht aus 10 Symbolen (aus A), $z_1 z_2 \cdots z_9 z_{10}$, wobei die ersten neun aus $\{0, 1, \dots, 9\}$ sind.

Die zehnte, z_{10} , wird so gewählt, dass

$$10z_1 + 9z_2 + \dots + 2z_9 + 1z_{10}$$

durch 11 teilbar ist.

Beispiel: Der ISBN-Code

ISBN-Code (International Standard Book Number):

$$A = \{0, 1, \dots, 9, X\} \quad (X \cong 10)$$

Ein Wort des ISBN-Codes besteht aus 10 Symbolen (aus A), $z_1 z_2 \cdots z_9 z_{10}$, wobei die ersten neun aus $\{0, 1, \dots, 9\}$ sind.

Die zehnte, z_{10} , wird so gewählt, dass

$$10z_1 + 9z_2 + \dots + 2z_9 + 1z_{10}$$

durch 11 teilbar ist.

Beispiel: 3 411 15193 5

Beispiel: Der ISBN-Code

ISBN-Code (International Standard Book Number):

$$A = \{0, 1, \dots, 9, X\} \quad (X \cong 10)$$

Ein Wort des ISBN-Codes besteht aus 10 Symbolen (aus A), $z_1 z_2 \cdots z_9 z_{10}$, wobei die ersten neun aus $\{0, 1, \dots, 9\}$ sind.

Die zehnte, z_{10} , wird so gewählt, dass

$$10z_1 + 9z_2 + \dots + 2z_9 + 1z_{10}$$

durch 11 teilbar ist.

Beispiel: 3 411 15193 5

Der ISBN-Code erkennt **einen** Fehler und eine **Vertauschung** benachbarter Ziffern (Hausaufgabe).

Beispiel: Der $[7, 4; \{0, 1\}]$ -Hamming-Code

$$A = \{0, 1\}$$

$16 = 2^4$ Informationseinheiten sollen codiert werden:

→ Wörter der Länge 4 über A

Hänge drei Kontrollsymbole nach folgendem Schema an:

$$z_1 + z_3 + z_4 + z_5 = 0$$

$$z_1 + z_2 + z_4 + z_6 = 0$$

$$z_2 + z_3 + z_4 + z_7 = 0$$

Boolesche Arithmetik oder „exclusive or“ = „XOR“ ($1 + 1 = 0$)

Beispiel: Der $[7, 4; \{0, 1\}]$ -Hamming-Code

$$A = \{0, 1\}$$

$16 = 2^4$ Informationseinheiten sollen codiert werden:

→ Wörter der Länge 4 über A

Hänge drei Kontrollsymbole nach folgendem Schema an:

$$z_1 + z_3 + z_4 + z_5 = 0$$

$$z_1 + z_2 + z_4 + z_6 = 0$$

$$z_2 + z_3 + z_4 + z_7 = 0$$

Boolesche Arithmetik oder „exclusive or“ = „XOR“ ($1 + 1 = 0$)

→ Wörter der Länge 7 über A : $[7, 4; A]$ -Code

Beispiel: Der $[7, 4; \{0, 1\}]$ -Hamming-Code

$$A = \{0, 1\}$$

$16 = 2^4$ Informationseinheiten sollen codiert werden:

→ Wörter der Länge 4 über A

Hänge drei Kontrollsymbole nach folgendem Schema an:

$$z_1 + z_3 + z_4 + z_5 = 0$$

$$z_1 + z_2 + z_4 + z_6 = 0$$

$$z_2 + z_3 + z_4 + z_7 = 0$$

Boolesche Arithmetik oder „exclusive or“ = „XOR“ ($1 + 1 = 0$)

→ Wörter der Länge 7 über A : $[7, 4; A]$ -Code

Rate: $4/7$ (eines Wortes ist Information)

Allgemein: Ein $[n, k; A]$ -Code \mathcal{C} ist ein Block-Code der Länge n über dem Alphabet A mit $|A|^k$ Wörtern. **Rate** von \mathcal{C} : k/n

Allgemein: Ein $[n, k; A]$ -Code \mathcal{C} ist ein Block-Code der Länge n über dem Alphabet A mit $|A|^k$ Wörtern. **Rate** von \mathcal{C} : k/n

Für zwei Codewörter w_1 und w_2 sei $d(w_1, w_2)$ die Anzahl der Stellen, an denen sich w_1 und w_2 unterscheiden.

(Beispiel: $d(1000110, 0010101) = 4$)

Allgemein: Ein $[n, k; A]$ -Code \mathcal{C} ist ein Block-Code der Länge n über dem Alphabet A mit $|A|^k$ Wörtern. **Rate** von \mathcal{C} : k/n

Für zwei Codewörter w_1 und w_2 sei $d(w_1, w_2)$ die Anzahl der Stellen, an denen sich w_1 und w_2 unterscheiden.
(Beispiel: $d(1000110, 0010101) = 4$)

Dieser **Hamming-Abstand** hat ähnliche Eigenschaften wie der euklidische Abstand in der Ebene. (Metrik)

Allgemein: Ein $[n, k; A]$ -Code \mathcal{C} ist ein Block-Code der Länge n über dem Alphabet A mit $|A|^k$ Wörtern. **Rate** von \mathcal{C} : k/n

Für zwei Codewörter w_1 und w_2 sei $d(w_1, w_2)$ die Anzahl der Stellen, an denen sich w_1 und w_2 unterscheiden.
(Beispiel: $d(1000110, 0010101) = 4$)

Dieser **Hamming-Abstand** hat ähnliche Eigenschaften wie der euklidische Abstand in der Ebene. (Metrik)

Für den Hamming-Code gilt: Zwei Codewörter unterscheiden sich immer an mindestens 3 Stellen.

Allgemein: Ein $[n, k; A]$ -Code \mathcal{C} ist ein Block-Code der Länge n über dem Alphabet A mit $|A|^k$ Wörtern. **Rate** von \mathcal{C} : k/n

Für zwei Codewörter w_1 und w_2 sei $d(w_1, w_2)$ die Anzahl der Stellen, an denen sich w_1 und w_2 unterscheiden.
(Beispiel: $d(1000110, 0010101) = 4$)

Dieser **Hamming-Abstand** hat ähnliche Eigenschaften wie der euklidische Abstand in der Ebene. (Metrik)

Für den Hamming-Code gilt: Zwei Codewörter unterscheiden sich immer an mindestens 3 Stellen.

$d(\mathcal{C})$, der kürzeste Abstand zweier Codewörter, heißt die **Minimaldistanz** von \mathcal{C} .

Ist $d(c) = 3$, dann kann c **einen** Fehler korrigieren: Es gibt nur ein einziges Codewort, das sich von dem fehlerhaften Wort an einer Stelle unterscheidet.

(Beispiel: gesendet 1000110, empfangen 1001110)

Ist $d(C) = 3$, dann kann C **einen** Fehler korrigieren: Es gibt nur ein einziges Codewort, das sich von dem fehlerhaften Wort an einer Stelle unterscheidet.

(Beispiel: gesendet 1000110, empfangen 1001110)

Analog: Unterscheiden sich zwei Codewörter an mindestens 5 ($2e + 1$) Stellen, dann können bis zu **zwei** (e) Fehler korrigiert werden.

Ist $d(c) = 3$, dann kann c **einen** Fehler korrigieren: Es gibt nur ein einziges Codewort, das sich von dem fehlerhaften Wort an einer Stelle unterscheidet.

(Beispiel: gesendet 1000110, empfangen 1001110)

Analog: Unterscheiden sich zwei Codewörter an mindestens 5 ($2e + 1$) Stellen, dann können bis zu **zwei** (e) Fehler korrigiert werden.

Singleton Schranke: Ist c ein $[n, k; A]$ -Code mit $d = d(c)$, dann ist $d + k \leq n + 1$.

Ist $d(c) = 3$, dann kann c **einen** Fehler korrigieren: Es gibt nur ein einziges Codewort, das sich von dem fehlerhaften Wort an einer Stelle unterscheidet.

(Beispiel: gesendet 1000110, empfangen 1001110)

Analog: Unterscheiden sich zwei Codewörter an mindestens 5 ($2e + 1$) Stellen, dann können bis zu **zwei** (e) Fehler korrigiert werden.

Singleton Schranke: Ist c ein $[n, k; A]$ -Code mit $d = d(c)$, dann ist $d + k \leq n + 1$.

Also: große Rate kleine Minimaldistanz, und große Minimaldistanz kleine Rate.

Suche **gute** Codes mit **hoher** Rate und **größtmöglicher** Minimaldistanz.

Suche **gute** Codes mit **hoher** Rate und **größtmöglicher** Minimaldistanz.

Fragen:

- Wie beschreibt man Codes (ohne Aufzählung der einzelnen Codewörter)?
- Wie erkennt man **schnell** die Codewörter?
- Wie korrigiert man **schnell** die eventuellen Fehler?

Suche **gute** Codes mit **hoher** Rate und **größtmöglicher** Minimaldistanz.

Fragen:

- Wie beschreibt man Codes (ohne Aufzählung der einzelnen Codewörter)?
- Wie erkennt man **schnell** die Codewörter?
- Wie korrigiert man **schnell** die eventuellen Fehler?

Eine erste Antwort:

- Durch Einführen und Ausnutzen zusätzlicher Strukturen auf dem Alphabet A und dem Code \mathcal{C} über A .
- Auf A kann eine z.B. Addition und Multiplikation eingeführt werden, die A zu einem **Körper** macht.
- Dann kann \mathcal{C} als Lösungsmenge eines linearen Gleichungssystems beschrieben werden.

Ein Körper mit 256 Elementen

Beim CD-Spieler wird ein Code über einem Alphabet A mit 256 Elementen benutzt.

Elemente von A sind die Bytes (also Bit-Folgen der Länge 8).

Ein Körper mit 256 Elementen

Beim CD-Spieler wird ein Code über einem Alphabet A mit 256 Elementen benutzt.

Elemente von A sind die Bytes (also Bit-Folgen der Länge 8).

Addition auf A : Komponentenweise binäre Addition.

(Beispiel: $01101010 + 11001010 = 10100000$)

Ein Körper mit 256 Elementen

Beim CD-Spieler wird ein Code über einem Alphabet A mit 256 Elementen benutzt.

Elemente von A sind die Bytes (also Bit-Folgen der Länge 8).

Addition auf A : Komponentenweise binäre Addition.

(Beispiel: $01101010 + 11001010 = 10100000$)

Multiplikation: Ein Byte $b_7b_6b_5b_4b_3b_2b_1b_0$ wird als Polynom

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$$

interpretiert. Zwei Bytes werden multipliziert, indem die zugehörigen Polynome multipliziert werden, und durch

$$x^8 + x^4 + x^3 + x^2 + 1$$

geteilt werden. Der Rest ist das Ergebnis der Multiplikation.

Ein Körper mit 256 Elementen

Beim CD-Spieler wird ein Code über einem Alphabet A mit 256 Elementen benutzt.

Elemente von A sind die Bytes (also Bit-Folgen der Länge 8).

Addition auf A : Komponentenweise binäre Addition.

(Beispiel: $01101010 + 11001010 = 10100000$)

Multiplikation: Ein Byte $b_7b_6b_5b_4b_3b_2b_1b_0$ wird als Polynom

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$$

interpretiert. Zwei Bytes werden multipliziert, indem die zugehörigen Polynome multipliziert werden, und durch

$$x^8 + x^4 + x^3 + x^2 + 1$$

geteilt werden. Der Rest ist das Ergebnis der Multiplikation.

Dadurch wird A zu einem Körper.

Alphabet des CD-Codes: $256 = 2^8$ Symbole (Symbol $\hat{=}$ 8 Bits)

Audiodaten vor der Codierung

Alphabet des CD-Codes: $256 = 2^8$ Symbole (Symbol $\hat{=}$ 8 Bits)

Schallsignal (analog) \rightsquigarrow Folge von Nullen und Einsen (digital)

Audiodaten vor der Codierung

Alphabet des CD-Codes: $256 = 2^8$ Symbole (Symbol $\hat{=}$ 8 Bits)

Schallsignal (analog) \rightsquigarrow Folge von Nullen und Einsen (digital)

44100 Schallsignale/sec werden gemessen (Samples) \implies
Frequenzen bis zu 22,5 kHz rekonstruierbar (Abtasttheorem)

Audiodaten vor der Codierung

Alphabet des CD-Codes: $256 = 2^8$ Symbole (Symbol $\hat{=}$ 8 Bits)

Schallsignal (analog) \rightsquigarrow Folge von Nullen und Einsen (digital)

44100 Schallsignale/sec werden gemessen (Samples) \implies
Frequenzen bis zu 22,5 kHz rekonstruierbar (Abtasttheorem)

Sample (Amplitude) $\mapsto a$ ($0 \leq a \leq 2^{16} - 1$),
d.h. a benötigt 2 Symbole aus A

Audiodaten vor der Codierung

Alphabet des CD-Codes: $256 = 2^8$ Symbole (Symbol $\hat{=}$ 8 Bits)

Schallsignal (analog) \rightsquigarrow Folge von Nullen und Einsen (digital)

44100 Schallsignale/sec werden gemessen (Samples) \implies
Frequenzen bis zu 22,5 kHz rekonstruierbar (Abtasttheorem)

Sample (Amplitude) $\mapsto a$ ($0 \leq a \leq 2^{16} - 1$),
d.h. a benötigt 2 Symbole aus A

2 Kanäle, 1 Wert pro Kanal: 1 Sample $\hat{=}$ 4 Symbolen $\hat{=}$ 32 Bits

Audiodaten vor der Codierung

Alphabet des CD-Codes: $256 = 2^8$ Symbole (Symbol $\hat{=}$ 8 Bits)

Schallsignal (analog) \rightsquigarrow Folge von Nullen und Einsen (digital)

44100 Schallsignale/sec werden gemessen (Samples) \implies
Frequenzen bis zu 22,5 kHz rekonstruierbar (Abtasttheorem)

Sample (Amplitude) $\mapsto a$ ($0 \leq a \leq 2^{16} - 1$),
d.h. a benötigt 2 Symbole aus A

2 Kanäle, 1 Wert pro Kanal: 1 Sample $\hat{=}$ 4 Symbolen $\hat{=}$ 32 Bits
 \rightsquigarrow 1 411 200 Bits/sec (vor Codierung)

Audiodaten vor der Codierung

Alphabet des CD-Codes: $256 = 2^8$ Symbole (Symbol $\hat{=}$ 8 Bits)

Schallsignal (analog) \rightsquigarrow Folge von Nullen und Einsen (digital)

44100 Schallsignale/sec werden gemessen (Samples) \implies
Frequenzen bis zu 22,5 kHz rekonstruierbar (Abtasttheorem)

Sample (Amplitude) $\mapsto a$ ($0 \leq a \leq 2^{16} - 1$),
d.h. a benötigt 2 Symbole aus A

2 Kanäle, 1 Wert pro Kanal: 1 Sample $\hat{=}$ 4 Symbolen $\hat{=}$ 32 Bits
 \rightsquigarrow 1 411 200 Bits/sec (vor Codierung)

75 Minuten Musik auf CD entsprechen

$75 * 60 * 1\,411\,200 / 8 \text{ B} = 793\,800\,000 \text{ B} \approx 800 \text{ MB}$

Sample vor der Codierung: 32 Bits, danach: 98 Bits

Audiodaten nach der Codierung

Sample vor der Codierung: 32 Bits, danach: 98 Bits

Nach der Codierung: $44100 * 98 = 4\,321\,800$ Bits/sec

Audiodaten nach der Codierung

Sample vor der Codierung: 32 Bits, danach: 98 Bits

Nach der Codierung: $44100 * 98 = 4\,321\,800$ Bits/sec

Ist (nur) jedes 10 000. Bit falsch, dann sind das mehr als 400 Fehler/sec

Audiodaten nach der Codierung

Sample vor der Codierung: 32 Bits, danach: 98 Bits

Nach der Codierung: $44100 * 98 = 4\,321\,800$ Bits/sec

Ist (nur) jedes 10 000. Bit falsch, dann sind das mehr als 400 Fehler/sec

75 Minuten Musik auf CD benötigt nach der Codierung

$75 * 60 * 4\,321\,800 / 8 \text{ B} = 2\,431\,012\,500 \text{ B} \approx 2,4 \text{ GB}$

Die Verarbeitung erfolgt in **Frames** (= Blöcken) à 588 Bits.
Diese enthalten die Audiodaten von 6 Samples.

Frames, I

Die Verarbeitung erfolgt in **Frames** (= Blöcken) à 588 Bits.
Diese enthalten die Audiodaten von 6 Samples.

Aufbau eines Frames: 6 Samples

→ Wort der Länge 24 über A

Die Verarbeitung erfolgt in **Frames** (= Blöcken) à 588 Bits.
Diese enthalten die Audiodaten von 6 Samples.

Aufbau eines Frames: 6 Samples

→ Wort der Länge 24 über A

→ $[28, 24; A]$ -Code → Wort der Länge 28 über A

Die Verarbeitung erfolgt in **Frames** (= Blöcken) à 588 Bits.
Diese enthalten die Audiodaten von 6 Samples.

Aufbau eines Frames: 6 Samples

→ Wort der Länge 24 über A

→ $[28, 24; A]$ -Code → Wort der Länge 28 über A

→ $[32, 28; A]$ -Code → Wort der Länge 32 über A

Die Verarbeitung erfolgt in **Frames** (= Blöcken) à 588 Bits.
Diese enthalten die Audiodaten von 6 Samples.

Aufbau eines Frames: 6 Samples

→ Wort der Länge 24 über A

→ $[28, 24; A]$ -Code → Wort der Länge 28 über A

→ $[32, 28; A]$ -Code → Wort der Länge 32 über A

Der zweite Code dient der Korrektur von Fehlerbündeln (durch Cross-Interleaving):

Bis zu 8 000 aufeinander folgende Bits $\hat{=}$ 2,5 mm Spur können korrigiert werden.

Pro Wort: 8 Bits für Steuerung \rightarrow Wort der Länge 33 über A

Pro Wort: 8 Bits für Steuerung \rightarrow Wort der Länge 33 über A

Technische Anforderung (Lese- und Schreibtechnik der CD):

$$1 \dots 100001 \underbrace{0 \dots 0}_{\text{min } 2, \text{max } 10} 1000001 \dots 00 \quad (1)$$

Dazu Ersetzung nach Tabelle

$$a \in A \text{ (8 Bits)} \longleftrightarrow a' \text{ (14 Bits)}$$

(etwas Kombinatorik, 14 Bits kleinstmöglich)

Pro Wort: 8 Bits für Steuerung \rightarrow Wort der Länge 33 über A

Technische Anforderung (Lese- und Schreibtechnik der CD):

$$1 \dots 100001 \underbrace{0 \dots 0}_{\text{min } 2, \text{max } 10} 1000001 \dots 00 \quad (1)$$

Dazu Ersetzung nach Tabelle

$$a \in A \text{ (8 Bits)} \longleftrightarrow a' \text{ (14 Bits)}$$

(etwas Kombinatorik, 14 Bits kleinstmöglich)

Wegen (1) kommen noch 3 Bits zwischen je zwei 14-Bit-Folgen, dazu 27 Bits zwischen je zwei Frames zur Synchronisation.

Pro Wort: 8 Bits für Steuerung \rightarrow Wort der Länge 33 über A

Technische Anforderung (Lese- und Schreibtechnik der CD):

$$1 \dots 100001 \underbrace{0 \dots 0}_{\text{min } 2, \text{max } 10} 1000001 \dots 00 \quad (1)$$

Dazu Ersetzung nach Tabelle

$$a \in A \text{ (8 Bits)} \longleftrightarrow a' \text{ (14 Bits)}$$

(etwas Kombinatorik, 14 Bits kleinstmöglich)

Wegen (1) kommen noch 3 Bits zwischen je zwei 14-Bit-Folgen, dazu 27 Bits zwischen je zwei Frames zur Synchronisation.

Pro Frame: $33 * (14 + 3) + 27 = 588$ Bits

Pits und Lands

Speicherung auf der CD durch

Rillen (Pits) und Nicht-Rillen (Lands)

Pits und Lands

Speicherung auf der CD durch

Rillen (Pits) und Nicht-Rillen (Lands)

Übergänge

[Pit \rightarrow Land] oder [Land \rightarrow Pit]

werden als 1 interpretiert.

Pits und Lands

Speicherung auf der CD durch

Rillen (Pits) und Nicht-Rillen (Lands)

Übergänge

[Pit \rightarrow Land] oder [Land \rightarrow Pit]

werden als 1 interpretiert.

Lands und Pits werden als Folgen von Nullen interpretiert.

Pits und Lands

Speicherung auf der CD durch

Rillen (Pits) und Nicht-Rillen (Lands)

Übergänge

[Pit \rightarrow Land] oder [Land \rightarrow Pit]

werden als 1 interpretiert.

Lands und Pits werden als Folgen von Nullen interpretiert.

Länge eines Lands oder Pits:

maximal $3,05 * 10^{-6} \text{m} \Rightarrow \leq 10$ Nullen

minimal $0,83 * 10^{-6} \text{m} \Rightarrow \geq 2$ Nullen

Pits und Lands

Speicherung auf der CD durch

Rillen (Pits) und Nicht-Rillen (Lands)

Übergänge

[Pit \rightarrow Land] oder [Land \rightarrow Pit]

werden als 1 interpretiert.

Lands und Pits werden als Folgen von Nullen interpretiert.

Länge eines Lands oder Pits:

maximal $3,05 * 10^{-6} \text{m} \Rightarrow \leq 10$ Nullen

minimal $0,83 * 10^{-6} \text{m} \Rightarrow \geq 2$ Nullen

Drehzahl: innen: 486/min, außen: 196/min

Pits und Lands

Speicherung auf der CD durch

Rillen (Pits) und Nicht-Rillen (Lands)

Übergänge

[Pit \rightarrow Land] oder [Land \rightarrow Pit]

werden als 1 interpretiert.

Lands und Pits werden als Folgen von Nullen interpretiert.

Länge eines Lands oder Pits:

maximal $3,05 * 10^{-6} \text{m} \Rightarrow \leq 10$ Nullen

minimal $0,83 * 10^{-6} \text{m} \Rightarrow \geq 2$ Nullen

Drehzahl: innen: 486/min, außen: 196/min

1 Bit $\hat{=}$ $0,3 * 10^{-6} \text{m} \rightarrow 6,5 \text{km Spur}$