Informatik 5bi Schuljahr 2018/2019 als PDF exportieren

Informatik 5. Klasse - Schuljahr 2018/19

Lehrinhalte

- Lehrplaninhalte

Remote-Zugriff auf Schulserver

Kapitel

- 1) Zahlensysteme in der Informatik
- 2) Schaltalgebra
- 3) Zeichencodierung
- 4) Hardware
- 5) Präsentationstechniken
- 6) Bildbearbeitung
- 7) Grundlagen zur Programmierung
- 8) Textverarbeitung
- 9) Audiobearbeitung
- 10) Videobearbeitung
- 11) Betriebssysteme
- 12) Web-Techniken (HTML)

Leistungsbeurteilung

- **Test (SA)**
  - 2x Test (1h) pro Semester
- **Mitarbeit (MA)**
  - Aktive Mitarbeit im Unterricht (aMA)
  - Mündliche Stundenwiederholungen (mMA)
  - Schriftliche Stundenwiederholungen (sMA)
- **Praktische Arbeiten (PA)**
  - 1x praktischer Arbeitsauftrag pro Woche

- Aktueller Leistungsstand

Stoff für den 2. Test in Informatik – 5BI – ??.??.????

Kapitel 4.3 bis ??

Stoff für den 1. Test in Informatik – 5BI – 17.10.2018

Kapitel 1 bis Kapitel 4.2

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819

Last update: **2018/10/23 15:54**

Was wird in der 5. Klasse gemacht?

Gesellschaftliche Aspekte der Informationstechnologie

Geschichte der Informatik

- Einen Überblick hinsichtlich der wichtigsten Entwicklungsstufen der Informatik gewinnen

Kommunikation und Kooperation

- Digitale Systeme zum Informationsaustausch, zur Unterstützung der Unterrichtsorganisation und zum Lernen auch in kommunikativen und kooperativen Formen verwenden können.
- Informationsmanagement und Lernorganisation für die eigene Lernarbeit und Weiterbildung mit geeigneter Software in der Praxis umsetzen und dabei vorhandene Informationsquellen erschließen und unterschiedliche Informationsdarstellungen ausgehend von den Vorkenntnissen anwenden

Verantwortung, Datenschutz und Datensicherheit

- Wesentliche Maßnahmen und rechtliche Grundlagen im Zusammenhang mit Datensicherheit, Datenschutz und Urheberrecht kennen lernen sowie die Auswirkungen des Technikeinsatzes auf die Einzelnen und die Gesellschaft nachvollziehen

Berufliches Spektrum der Informatik

- Einsatzmöglichkeiten der Informatik in verschiedenen Berufsfeldern kennen lernen und somit in ihrer Berufsorientierung Unterstützung finden

Informatiksysteme – Hardware, Betriebssysteme und Vernetzung

Technische Grundlagen und Funktionsweisen (Hardware)

- Den Aufbau von digitalen Endgeräten beschreiben und erklären können.
- Die Funktionsweise von Informatiksystemen erklären können.
- Die Komponenten eines Rechners und ihr Zusammenspiel kennen
- Die verschiedenen Formen von Datenträgern kennen und bewerten können
- Speichermedien konfigurieren können, Partitionierung von Festplatten erstellen und bearbeiten können

Betriebssysteme und Software (DOS und Windows, Netzwerk)

- Grundlagen von Betriebssystemen erklären, eine graphische Oberfläche und Dienstprogramme benutzen können.
- Grundlegende Konsolenkommandos kennen und anwenden können
- Mit graphische Oberflächen (Desktop, Modern UI-Oberfläche) umgehen können
- Batches und System-Scripts erstellen können
- Wesentliche Konfigurationsmöglichkeiten von Windows kennen
- Ein vernetztes Informationssystem für die individuelle Arbeit aufbauen und nutzen können

Algorithmik und Programmierung

Zahlensysteme und Schaltalgebra

- Die Zahlensysteme der Informatik kennen und damit umgehen können
- Rechenregeln der Schaltalgebra kennen und anwenden können
- Grundschaltungen und sowie Halbaddierer-, Volladdierer-Schaltung kennen und Schaltungssimulationen erstellen können

Algorithmen und Datenstrukturen

- Grundprinzipien von Automaten, Algorithmen, Datenstrukturen und Programmen erklären können
- Grundlegende Datentypen kennen
- Einfache Algorithmen erklären, entwerfen, darstellen können.
- Die Codierung einfacher Algorithmen kennen und an einfachen Beispielen anwenden können
- Wichtige und bekannte Algorithmen aus Mathematik und Informatik (z.B. aus Teilbarkeitslehre, Reihenfolgeprobleme, ...) kennen
- Möglichkeiten der Visualisierungen von Algorithmen kennen

Programmierung (Objektorientierte Programmiersprache)

- Algorithmen in einer Programmiersprache implementieren können
- Kontrollstrukturen (Verzweigungen, Schleifen) kennen und einsetzen können
- Unterprogramme kennen und einsetzen können

Mathematische Arbeitsumgebungen

- Mit mathematischen Arbeitsumgebungen umgehen können

Angewandte Informatik, Datenbanksysteme und Internet

Allgemein

- Einblicke in wesentliche Begriffe und Methoden der Informatik, ihre typischen Denk-und Arbeitsweisen, ihre historische Entwicklung sowie ihre technischen und theoretischen Grundlagen gewinnen und Grundprinzipien von Automaten, Algorithmen und Programmen kennen lernen
- Begriffe und Konzepte der Informatik verstehen und Methoden und Arbeitsweisen anwenden können

Grundlagen der Bild-, Ton- und Videobearbeitung

- Standardsoftware zur Bildbearbeitung kennen und einsetzen können.
- Standardsoftware zur Audiobearbeitung kennen und einsetzen können.
- Standardsoftware zur Videobearbeitung kennen und einsetzen können.

Textverarbeitungs- und Satzsysteme

- Grundlagen der Text- und Seitengestaltung kennen und anwenden können
- Techniken zur Bearbeitung großer Dokumente (wie z.B. VWA) kennen und anwenden können
- den sicheren Umgang mit Standardsoftware zur schriftlichen Korrespondenz, zur Dokumentation, zur Publikation von Arbeiten erreichen

Präsentationssysteme und Visualisierung

- Standardsoftware zur Kommunikation und Dokumentation sowie zur Erstellung, Publikation und multimedialen Präsentation eigener Arbeiten einsetzen können.
- den sicheren Umgang mit Standardsoftware zur multimedialen Präsentation sowie zur Kommunikation erreichen
- Inhalte systematisieren und strukturieren sowie Arbeitsergebnisse zusammenstellen und multimedial präsentieren können

Web-Techniken

- Grundlagen in HTML (Grundformatierung, Tabellen, Container, Einbau von Bild-, Ton- und Videoelementen) kennen und anwenden können
- Das Editieren in einem Content-Management-System beherrschen

---

```
=> 5. Klasse (3 Stunden,  2-3 Tests pro Semester)
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:0_lehrplaninhalte

Last update: **2018/09/10 14:30**

1) Zahlensysteme in der Informatik

- Kurzüberblick - Geschichtliche Entwicklung des Zählens
- Skriptum zur Einführung
- Informationseinheiten in der Informatik

- 1.01) Dualsystem
- 1.02) Hexadezimalsystem
- 1.03) Oktalsystem
- 1.04) Umrechnungen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:1_zahlensysteme

Last update: **2018/09/14 04:27**

1.01) Dualsystem

Das **duale Zahlensystem - auch Dualsystem oder Binärsystem** genannt - besteht aus **2 Ziffern**, gekennzeichnet durch 0 und 1. Man benötigt dieses Zahlensystem in der Informatik, da sich mit technischen Bauteilen sehr leicht die Zustände AN und AUS erzeugen lassen können. Diese Zahlen können entsprechend unserem „normalen“ Dezimalsystem verwendet werden. Man kann sie addieren, subtrahieren, multiplizieren und dividieren. Da sie sich also kaum vom „normalen“ Rechnen unterscheiden, eignen sie sich hervorragend, um in der EDV eingesetzt zu werden.

Zählen im Dualsystem

Auch hier beginnen wir mit 0 und zählen dann 1. Leider haben wir nur 2 Zahlen, also gehen uns hier die Zahlen schnell aus. Wir machen es jetzt aber genau wie im Dezimalsystem und nehmen eine Stelle dazu. Nach 0 und 1 kommen dann also 10 und 11. Wieder reichen die Stellen nicht! Also noch eine dazu: 100, 101, 110, 111, usw.

Zählen von 0 bis 15 im Dezimal- und Dualsystem

Dezimal	Dual
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

Eine andere Schreibweise

Man kann Zahlen auch anhand ihrer Basis darstellen. Im Dezimalsystem haben wir 10 Zahlen zur Verfügung, von 0 bis 9. Mit 2 Stellen können wir also 10 * 10 = 100 Zahlen darstellen. 100 Zahlen? Aber 100 hat doch drei Stellen! Dieser Einwand stimmt. Da wir jedoch mit der Zahl 0 beginnen, ist 0 die 1. Zahl, 1 die 2. Zahl, … 98 die 99. Zahl und 99 die 100. Zahl.

Mit 3 Stellen können wir 10 * 10 * 10 = 1000 Zahlen darstellen. Jede Stelle entspricht einer 10-er Potenz.

An einem einfachen Beispiel versuche ich diesen Sachverhalt zu erklären.

Wir nehmen dazu die Zahl 372 und schreiben sie als kleine Rechnung auf:

$372 = 3*100 + 7*10 + 2*1$.

Das kann man jetzt noch anders darstellen als:

$3*10^2 + 7*10^1 + 2*10^0$.

Auf diese Weise kann man jetzt alle anderen Zahlen auch darstellen:

$6574 = 6*10^3 + 5*10^2 + 7*10^1 + 4*10^0$
$12032 = 1*10^4 + 2*10^3 + 0*10^2 + 3*10^1 + 2*10^0$

Verwendung der Potenzschreibweise bei binären Zahlen

Wendet man die Potenzschreibweise bei binären Zahlen an, so muss man eine andere Basis wählen. Es gibt ja nur 2 verschiedene Ziffern, 0 und 1. Also nehmen wir als Basis 2.

Die Zahl 1011 schreibt sich dann als

$1*2^3 + 0*2^2 + 1*2^1 + 1*2^0$

Die Umrechnung von dezimalen in binäre Zahlen

Bei der Umrechnung der Dezimalzahlen verwenden wir die „Division mit Rest“ aus der Grundschule. Wir teilen die Zahl solange durch 2, bis als Ergebnis 0 herauskommt und merken uns dabei den Rest. Als Beispiel sollen die Zahlen 13 und 14 dienen.

13 / 2 = 6 Rest 1
6 / 2 = 3 Rest 0
3 / 2 = 1 Rest 1
1 / 2 = 0 Rest 1

Die Reste von unten nach oben aneinander gereiht ergeben dann die Dualzahl 1101.

14 / 2 = 7 Rest 0
7 / 2 = 3 Rest 1
3 / 2 = 1 Rest 1
1 / 2 = 0 Rest 1

Hieraus ergibt sich dann die Dualzahl 1110.

Addition von Dualzahlen

Einige erinnern sich vielleicht noch an die Addition von Zahlen wie wir sie in der Grundschule gelernt haben. Wir schreiben die Zahlen untereinander und addieren sie Stelle für Stelle. Dabei beginnen wir mit der letzten Stelle und arbeiten uns langsam nach vorne durch. Die gleiche Vorgehensweise

benutzen wir nun auch wenn wir Dualzahlen addieren wollen.

Die Addition funktioniert wie bei der Addition von Dezimalzahlen:

```
 0111
+0100
 ====
 1011
```

Addition mit Überlauf:

```
 1111
+0100
 ====
10011
```

Es gelten die Regeln 0+0=0, 1+0=1, 0+1=1, 1+1=0 Übertrag 1. Im Prinzip also nichts neues. Addiert man im Dezimalsystem 2 Zahlen so kommt bei 5+5 auch 0 Übertrag 1 heraus. Der Übertrag wird bei beiden System jeweils voran gestellt. Also gilt im Dualsystem 1+1=10.

Vorsicht Überlauf!

Die Addition ist im Prinzip problemlos. Es gibt allerdings einen Haken an der Sache: Die Addition funktioniert nur innerhalb eines bestimmten Wertebereiches. Woran liegt das? In der Realität können wir beliebig grosse Zahlen darstellen. Das geht leider nicht in der Informatik. Wir haben nur einen begrenzten Raum bzw. Speicherplatz zur Verfügung. Wir müssen aus diesem Grund den Speicherbereich einschränken (siehe was ist ein Byte), in unserem Beispiel nehmen wir als Speicherplatz ein Byte. Normalerweise verwendet man zur Addition von ganzen Zahlen, einen deutlich größeren Wertebereich, aber um einen überschaubaren Rahmen zu haben, begrenzen wir uns absichtlich auf ein Byte. Unsere größte darstellbare Zahl ist die 11111111, also dezimal 255. Was passiert nun, wenn wir eine 00000001, also 1, addieren? Das verrückte ist: es kommt 00000000, also 0 heraus. Da bekanntlich 1+1=0 Übertrag 1 gibt, bekommt man als Ergebnis in Wirklichkeit nicht 00000000 sondern 00000000 Übertrag 1. Dieser letzte Übertrag kann jedoch nicht mehr gespeichert werden und wird deshalb einfach ersatzlos gestrichen. Es ergibt sich daraus jedoch auch eine gewisse Logik. Durch meinen beschränkten Wertebereich komme ich irgendwann an meine obere Grenze. Bei der Addition von 1 fängt dann jedoch der Wertebereich wieder von vorne an, ich bin jetzt an der unteren Grenze, man durchläuft dann wieder den Bereich bis zur oberen Grenze, usw. Wenn wir also immer und immer wieder 1 addieren zählen wir unendlich oft von 0 bis 255, dann wieder 0 bis 255 usw.

Subktraktion von Dualzahlen

Drei Schritte zu Subtraktion

Hier kommen wir mit unserer normalen Schulmathematik nicht mehr weiter. Bevor wir uns mit dem komplizierten „Warum ist das denn so?“ beschäftigen, merken wir uns erst einmal den Mechanismus. Die Subtraktion von binären Zahlen wird durch die **Addition des Zweierkomplementes**

durchgeführt. Zur Erklärung beginnen wir im ersten Schritt mit dem **Einerkomplement**, dann schauen wir im zweiten Schritt was das **Zweierkomplement** ist und dann kommen wir im letzten Schritt zur **Subtraktion**.

Das Einerkomplement

Was ist das Komplement von Dualzahlen? Man bildet das sogenannte Einerkomplement, indem man jede Zahl durch ihr Gegenteil ersetzt, also die 0 durch die 1 und die 1 durch die 0.

01011010 wird zu 10100101 11101101 wird zu 00010010

Das Zweierkomplement

Das Zweierkomplement entspricht dem Einerkomplement, nur wird zusätzlich noch 00000001 addiert.

01011010 wird im Einerkomplement zu 10100101 im Zweierkomplement zu 10100110 11101101 wird im Einerkomplement zu 00010010 im Zweierkomplement zu 00010011

Die Subtraktion von Dualzahlen

Der Satz lautet: Die Subtraktion von 2 Zahlen erfolgt durch die Addition des Zweierkomplementes. Als konkretes Beispiel nehmen wir dazu die Rechnung 14-9=5.

!!WICHTIG:!!
Die restlichen Ziffern müssen immer mit 0 aufgefüllt werden.

9 ist im Dualsystem 00001001.

Das Einerkomplement zu 00001001 ist 11110110.

Das Zweierkomplement 11110111.

Dies addieren wir nun zu 14 also 00001110.

```
 00001110
+11110111
 ========
 00000101
```

Auch hier wäre die richtige Zahl eigentlich 00000101 Übertrag 1, da wir den Übertrag jedoch nicht speichern können, bleiben wir bei 00000101 was ja der Dezimalzahl 5 entspricht.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:1_zahlensysteme:1_01

Last update: **2018/09/11 15:00**

1.02) Hexadezimalsystem

Besonders **wichtig** ist in der **Informatik und Digitaltechnik** neben dem Binärsystem auch das **Hexadezimalsystem (Sedezimalsystem)**. Das Hexadezimalsystem verwendet die **Basis 16**, d.h. es gibt **16 verschiedene Ziffern**, **0 bis 9** und zusätzlich die Buchstaben **A bis F** (sog. Zahlzeichen; können auch als klein geschrieben werden: a-f).

Mit dem Hexadezimalsystem können auf einfachere und kürzere Weise Binärzahlen notiert werden. Mit einer 4-stelligen Binärzahl (auch als Halbbyte oder Nibble bezeichnet) lassen sich 16 (24 = 16) verschiedene Zahlen darstellen, und zwar 0 bis 15 (die Null zählt mit!). Da das Hexadezimalsystem die Basis 16 (= 24) verwendet, reicht eine (!) Hexadezimalzahl aus, um vier Bits (Binärziffern) darzustellen. Mit zwei Hexadezimalzahlen kann ein Byte (8 Bits) angeschrieben werden.

Gegenüberstellung Hexadezimal-, Binär- und Dezimalsystem

Hex	Binär	Dezimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

Um eindeutig darauf hinzuweisen, dass es sich um eine Hexadezimalzahl handelt, kann ebenso wie in anderen Zahlensystemen die Basis tiefgestellt dazu geschrieben werden, z.B. $\mathbf{3F_{16}}$ (= 63_{10} dezimal) oder $\mathbf{93_{16}}$ (= 147_{10} dezimal). Es sind aber auch andere Schreibweisen üblich:

a) Vorangestelltes **0x (Prefix), z.B. 0x93**. Diese Notation wird in Programmiersprachen mit C-ähnlicher-Syntax verwendet.

b) Nachgestelltes **h** (Postfix), z.B. **93h**. Letztere Schreibweise ist besonders in der Technik gebräuchlich.

Umrechnung vom Dezimal- ins Hexadezimalsystem

Die Umrechnung funktioniert ähnlich der Umrechnung von Dezimal- zu Binärzahlen (s.o.). Nun muss aber, statt durch 2, durch 16 dividiert werden. Die Reste werden genauso von rechts nach links angeschrieben und geben, wenn das Ergebnis der Ganzzahlendivision 0 ist, das Endergebnis.

Beispiel: Die Dezimalzahl **304_{10}** soll in eine Hexadezimalzahl umgewandelt werden.

```
304 / 16 = 19 => 0 Rest
 19 / 16 =  1 => 3 Rest
  1 / 16 =  0 => 1 Rest
```

Für das Endergebnis werden jetzt die Reste **von unten nach oben gelesen**.
Somit ergibt sich ein Endergebnis von 130_{16}, das entspricht der Dezimalzahl **304_{10}**.

Umrechnung vom Hexadezimal- ins Dezimalsystem

Die Umrechnung vom Hexadezimal- ins Dezimalsystem kann genauso wie oben von Binär→Dezimal demonstriert, erfolgen. Die einzelnen Ziffern werden mit dem jeweiligen Stellenwert (16^n, wobei n = 0, 1, 2, ...) multipliziert und die jeweiligen Ergebnisse aufsummiert. Das folgende Beispiel demonstriert dies anhand der Hexadezimalzahl 130_{16}:

```
0 * 16^0  =    0
3 * 16^1  =   48
1 * 16^2  =  256
---------------
          =  304
```

Als Ergebnis erhalten wir 304 dezimal, womit die Probe - zur vorigen Rechnung in die umgekehrte Richtung - erfolgreich war. 304_{10} entspricht 130_{16}. Diese Antwort hätte in der Praxis natürlich auch ein wissenschaftlicher Taschenrechner geliefert. 😀 Es reicht dazu sogar der **Windows-Rechner** (den Sie nur auf die wissenschaftliche Ansicht umstellen müssen) oder unter Linux Programme wie z.B. KCalc.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:1_zahlensysteme:1_02

Last update: **2018/09/14 04:24**

1.03) Oktalsystem

Das Oktalsystem, auch Achtersystem genannt, verwendet die **Basis 8 (acht)**. Um Zahlen darzustellen, stehen die **Ziffern 0 bis 7** zur Verfügung. Die Bedeutung in der Informatik/Digitaltechnik ergibt sich dadurch, dass sich mit einer **Oktalzahl drei Bits** darstellen lassen. 23 ist 8, somit lassen sich mit 3 Bits 8 verschiedene Möglichkeiten darstellen. Eine Oktalzahl reicht, um diese Information wiederzugeben.

Das Oktalsystem wird hier insbesondere deshalb erwähnt, weil in vielen Programmiersprachen Zahlen auch in Oktalform angegeben werden können. Meist, z.B. in PHP, wird dazu eine 0 (Null) vorangestellt, z.B. 077 für 77_8 (= 63_{10}).

Umrechnungen erfolgen genauso wie beim Hexadezimalsystem gezeigt. Um die Oktalzahl auszurechnen, die einer best. Dezimalzahl entspricht, dividieren Sie die Dezimalzahl fortlaufend durch 8 und schreiben die Reste von rechts nach links an. In umgekehrter Richtung - von Oktal nach Dezimal - multiplizieren Sie die einzelnen Ziffern mit dem Stellenwert (8^n für n = 0, 1, 2, ...) und addieren die Teilergebnisse.

Umrechnung vom Dezimal- ins Oktalsystem

Die Umrechnung funktioniert ähnlich der Umrechnung von Dezimal- zu Binärzahlen (s.o.). Nun muss aber, statt durch 2, durch 8 dividiert werden. Die Reste werden genauso von rechts nach links angeschrieben und geben, wenn das Ergebnis der Ganzzahlendivision 0 ist, das Endergebnis.

Beispiel: Die Dezimalzahl **304_{10}** soll in eine Hexadezimalzahl umgewandelt werden.

```
304 / 8 = 38 => 0 Rest
 38 / 8 =  4 => 6 Rest
  4 / 8 =  0 => 4 Rest
```

Für das Endergebnis werden jetzt die Reste **von unten nach oben gelesen**.
Somit ergibt sich ein Endergebnis von 460_8, das entspricht der Dezimalzahl **304_{10}**.

Umrechnung vom Oktal- ins Dezimalsystem

Die Umrechnung vom Oktal- ins Dezimalsystem kann genauso wie oben von Binär→Dezimal demonstriert, erfolgen. Die einzelnen Ziffern werden mit dem jeweiligen Stellenwert (8^n, wobei n = 0, 1, 2, ...) multipliziert und die jeweiligen Ergebnisse aufsummiert. Das folgende Beispiel demonstriert dies anhand der Oktalzahl 460_8:

```
0 * 8^0  =    0
6 * 8^1  =   48
4 * 8^2  =  256
---------------
         =  304
```

Als Ergebnis erhalten wir 304 dezimal, womit die Probe - zur vorigen Rechnung in die umgekehrte Richtung - erfolgreich war. 304_{10} entspricht 460_{8}. Diese Antwort hätte in der Praxis natürlich auch ein wissenschaftlicher Taschenrechner geliefert. Es reicht dazu sogar der **Windows-Rechner** (den Sie nur auf die wissenschaftliche Ansicht umstellen müssen) oder unter Linux Programme wie z.B. KCalc.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:1_zahlensysteme:1_03

Last update: **2018/09/14 04:24**

Umrechnungen

Prinzipiell kann man jede Zahl von einem Zahlensystem in ein anderes Zahlensystem umrechnen. Dazu muss man normalerweise die Zahl immer zuerst in das Dezimalsystem und dann in das Ziel-Zahlensystem umrechnen. Ausnahmen gibt es bei Umrechnungen vom Binärsystem in ein anderes Zahlensystem, hier braucht man nicht unbedingt das Dezimalsystem.

Binärsystem ⇔ Oktalsystem

Binärsystem ⇒ Oktalsystem

Wir wissen ja bereits, dass eine Ziffer im Oktalsystem die Ziffern 0-7 annehmen kann und dass man 3 Bits im Dualsystem braucht um 8 verschiedene Zahlenwerte darzustellen.

⇒ Immer 3 Stellen im Binärsystem ergeben eine Ziffer/Stelle im Oktalsystem

Beispiel

Umwandlung der Zahl 010110011111_2 ins Hexadezimalsystem:

```
010 110 011 111
 ⇓   ⇓   ⇓   ⇓
 2   6   3   7
```

Die Zahl 010110011111_2 entspricht somit der Zahl 2637_8.

Oktalsystem ⇒ Binärsystem

Selbiges gilt auch für die Umrechnung vom Oktalsystem in das Binärsystem.

⇒ Eine Stelle im Oktalsystem entspricht 3 Stellen im Binärsytem

Beispiel

Umwandlung der Zahl 672_8 ins Binärsystem:

```
 6   7   2
 ⇓   ⇓   ⇓
110 111 010
```

Die Zahl 672_8 entspricht somit der Zahl 110111010_2.

Binärsystem ⇔ Hexadezimalsystem

Binärsystem ⇒ Hexadezimalsystem

Wir wissen ja bereits, dass eine Ziffer im Hexadezimalsystem die Werte 0-15 annehmen kann und dass man 4 Bits im Dualsystem braucht um 16 verschiedene Zahlenwerte darzustellen.

⇒ Immer 4 Stellen im Binärsystem ergeben eine Ziffer/Stelle im Oktalsystem

Beispiel

Umwandlung der Zahl 010110011111_2 ins Oktalsystem:

```
0101 1001 1111
 ⇓    ⇓    ⇓
 5    9    F
```

Die Zahl 010110011111_2 entspricht somit der Zahl $59F_{16}$.

Hexadezimalsystem ⇒ Binärsystem

Selbiges gilt auch für die Umrechnung vom Hexadezimalsystem in das Binärsystem.

⇒ Eine Stelle im Hexadezimalsystem entspricht 4 Stellen im Binärsytem

Beispiel

Umwandlung der Zahl 672_{16} ins Binärsystem:

```
 6    7    2
 ⇓    ⇓    ⇓
0110 0111 0010
```

Die Zahl 672_{16} entspricht somit der Zahl 11001110010_2.

Oktalsystem ⇒ Hexadezimalsystem

Will man vom Oktalsystem ins Hexadezimalsystem umrechnen, rechnet man am besten zuerst ins Binärsystem und dann ins Hexadezimalsystem um.

Hexadezimalsystem ⇒ Oktalsystem

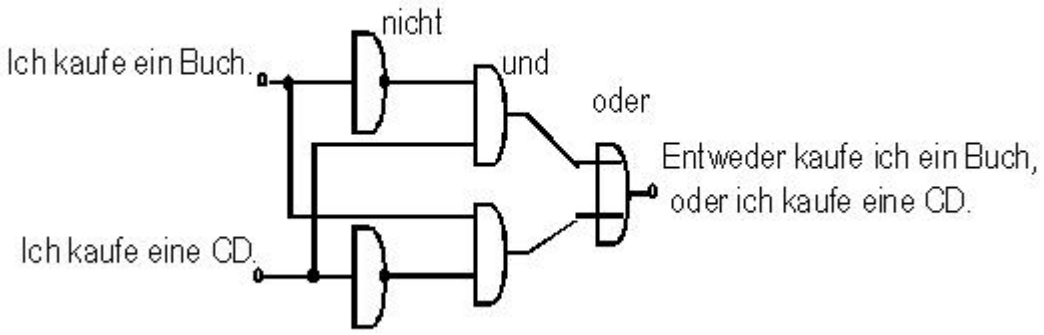
Will man vom Hexadezimalsystem ins Oktalsystem umrechnen, rechnet man am besten zuerst ins Binärsystem und dann ins Oktalsystem um.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:1_zahlensysteme:1_04

Last update: **2018/09/24 08:44**

2) Schaltalgebra



- 2.01) Grundlagen
- 2.02) Grundschaltungen
- 2.03) Zusammengesetzte Schaltungen
- 2.04) Gesetze der Schaltalgebra
- 2.05) Digitale Rechenschaltungen
- 2.06) Übungen

https://elektroniktutor.de/swfdatei/gatter.swf

Das Adobe Flash Plugin wird benötigt, um diesen Inhalt anzuzeigen.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:2

Last update: **2018/10/07 11:31**

2.01) Grundlagen

In der Digitaltechnik sind die Eingangsvariablen so miteinander zu verknüpfen, dass die Ausgangsvariable einen definierten Zustand annimmt, um damit einen Prozess zu steuern. So soll ein Fahrstuhl nur dann fahren, wenn eine Zieletage gewählt wurde, in der er zurzeit nicht steht, seine Tür vollständig geschlossen ist und von ihr nichts eingeklemmt wurde und die Kabine nicht überlastet ist. Man kann durch Kombinieren der beschriebenen digitalen Gatter und Ausprobieren bei einfacheren Aufgaben die eine oder andere Lösung finden. Das eigentliche Ziel ist es, eine wirtschaftliche Digitalschaltung zu entwickeln, die mit einem Minimum gleicher Gattertypen zum Ziel führt.

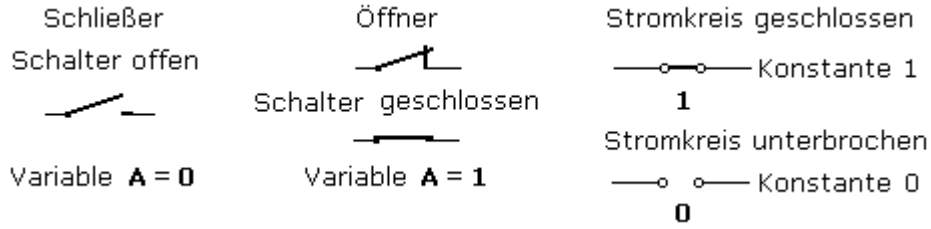
Der englische Mathematiker Georg Boole entwickelte eine Mengenalgebra, die in angepasster Weise als Boolesche Schaltalgebra bei der Problemlösung hilfreich ist. In der Schaltalgebra gibt es Variablen und Konstanten. Die binäre Digitaltechnik kommt mit zwei definierten logischen Zuständen, der 0 und 1 aus.

Konstante

In der Schaltalgebra gibt es nur die zwei konstanten Größen 0 und 1. In der elektronischen Schaltung entspricht eine dauerhaft geschaltete Leitung der Konstanten mit dem Zustand 1. Die dauerhafte Unterbrechung eines Stromkreises steht für die Konstante mit dem Wert 0.

Variable

Veränderbare, schaltbare Eingangsgrößen und davon abhängige veränderliche Ausgangswerte werden als Variable bezeichnet. In der binären Digitaltechnik nehmen sie entweder den Zustand 0 oder 1 an. In elektronischen Schaltungen können sie mit einem Schalter verglichen werden. Der geöffnete Schalter entspricht einer Variablen mit dem Wert 0. Bei geschlossenem Schalter nimmt die Variable den Wert 1 an.



Wir wissen, dass alle Grundrechnungsarten auf eine Addition zurückgeführt werden können. Deshalb ist das Ziel dieses Kapitels eine Maschine simulieren zu können, die addieren kann.

Informationseinheiten

BIT

Die Ziffern 0 und 1 werden physikalischen Zuständen zugeordnet:

```
0   Strom fließt nicht
1   Strom fließt
```

0 und 1 werden in der Informatik mit Bit bezeichnet. Ein Bit stellt die kleinste Informationseinheit dar (nur 2 Zustände – 0 und 1).

1 Bit ⇒ 2^1 ⇒ 2 Möglichkeiten
2 Bit ⇒ 2^2 ⇒ 4 Möglichkeiten
3 Bit ⇒ 2^3 ⇒ 8 Möglichkeiten
8 Bit = 1 Byte ⇒ 2^8 ⇒ 256 Möglichkeiten

BYTES

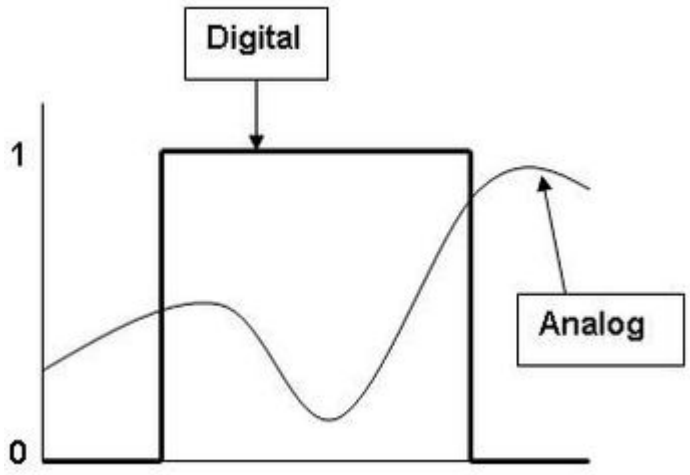
8 Bits sind ein Byte!

1 KB (Kilobyte) = 2^{10} = 1024 Byte
1 MB (Megabyte) = 2^{10} KB = 2^{10} x 2^{10} Byte = 2^{20} Byte = 1048576 Byte
1 GB (Gigabyte) = 2^{10} Byte
1 TB (Terrabyte) = 2^{10} Byte

Darstellung von 0 und 1

Ein Computer verarbeitet systembedingt Digitalsignale, da nur ausgewertet werden kann, ob eine Spannung anliegt oder nicht bzw. innerhalb eines Definitionsbereichs einen Wert über- oder unterschreitet. Diese zwei Zustände werden auch häufig bezeichnet als:

HIGH und **LOW**
TRUE und **FALSE**
AN und **AUS**
1 und **0**



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:2:2_01

Last update: **2018/09/23 16:51**

2.02) Grundschaltungen

UND-Verknüpfung (=Serienschaltung)

Bei der UND-Verknüpfung führt der Ausgang das Signal 1, wenn an beiden Eingängen (a und b) das Signal 1 anliegt.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)	Zeichen (Mathematik)
UND	AND	Konjunktion	X=A∧B	X=A*B

Schaltwerttabelle

a	b	a∧b
0	0	0
0	1	0
1	0	0
1	1	1

Schaltfunktion

f(a,b)=a∧b

Schaltsymbol

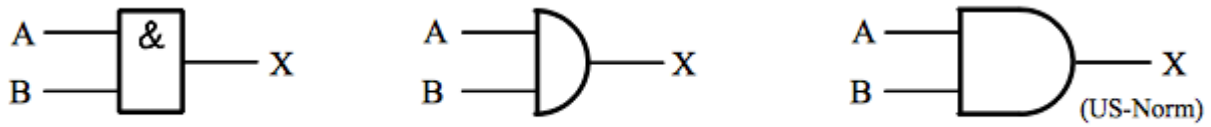


Bild 5-5: Schaltzeichen für UND-Gatter

elektronische Schaltung

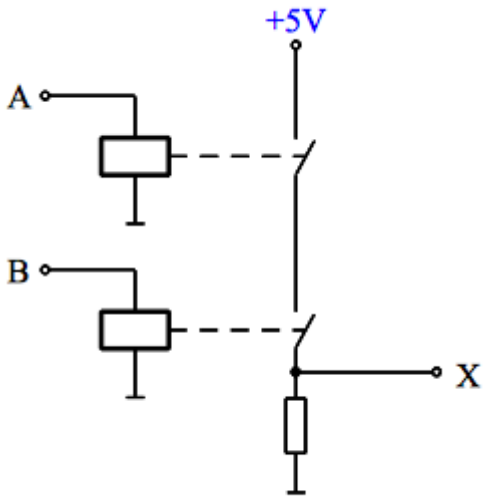


Bild 5-4: UND-Verknüpfung mit Relais

ODER-Verknüpfung (=Parallelschaltung)

Bei der ODER-Verknüpfung führt der Ausgang das Signal 1, wenn an einem der beiden Eingänge das Signal 1 anliegt.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)	Zeichen (Mathematik)
ODER	OR	Disjunktion	X=A∨B	X=A+B

Schaltwerttabelle

a	b	a∧b
0	0	0
0	1	1
1	0	1
1	1	1

Schaltfunktion

f(a,b)=a∨b

Schaltsymbol

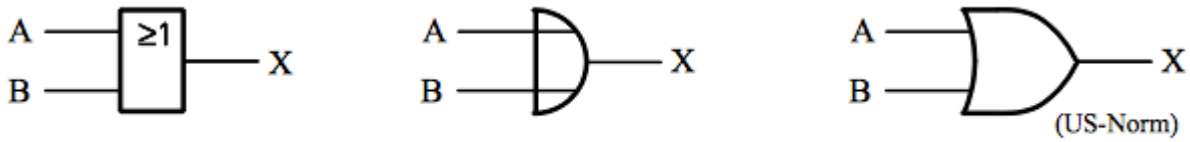


Bild 5-8: Schaltzeichen für ODER-Gatter

elektronische Schaltung

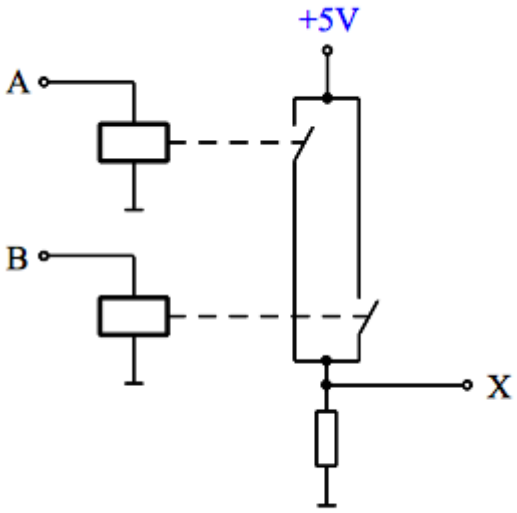


Bild 5-7: ODER-Verknüpfung mit Relais

NICHT-Verknüpfung (=NOT-Schaltung)

Bei einer NICHT-Verknüpfung wird der Ausgang logisch 1, wenn der Eingang logisch 0 ist, und umgekehrt.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)	Zeichen (Mathematik)
NICHT	NOT	Negation	X=¬A	X=Ā

Schaltwerttabelle

a	¬a
0	1
1	0

Schaltfunktion

f(a)=¬a

Schaltsymbol

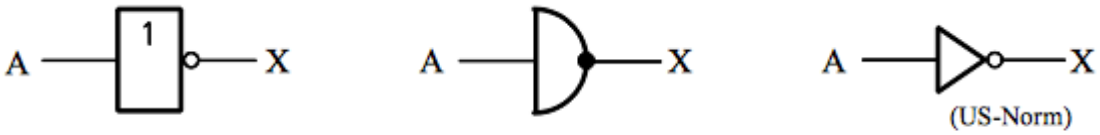


Bild 5-11: Schaltzeichen für NICHT-Gatter

elektronische Schaltung

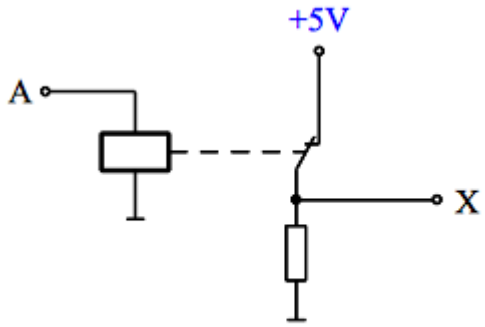


Bild 5-10: NICHT-Verknüpfung mit Relais

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:2:2_02

Last update: **2018/09/23 14:44**

2.03) Zusammengesetzte Schaltungen

Die **Grundverknüpfungen** werden in der Regel miteinander **kombiniert**, sodass die Logik der Schaltung verändert wird. Beispielsweise kann man eine UND- mit einer NICHT-Verknüpfung kombinieren und erhält dadurch eine sogenannte NAND-Verknüpfung (NICHT-UND). Da solch zusammengesetzte Schaltfunktionen sehr häufig verwendet werden, haben sie **eigene Schaltzeichen** bekommen. Mit den folgenden Schaltfunktionen werden die Grundverknüpfungen erweitert.

- NAND
- NOR
- XOR
- XNOR

NAND-Verknüpfung

Eine Kombination aus einem **UND-Gatter mit nachfolgendem NICHT-Gatter ergibt ein NAND-Gatter**. Die Ausgangsvariable Z des UND-Gatters wird durch das NICHT-Gatter negiert und erzeugt die Ausgangsvariable X des NAND-Glieds. Viele logische Funktionen lassen sich durch den Einsatz von NAND-Gattern lösen. Oft steigt dadurch die Anzahl der notwendigen Gatter, mit dem wirtschaftlichen Vorteil nur einen Gattertyp zu verwenden.

Der Ausgangszustand eines NAND-Glieds ergibt 1, wenn nicht alle Eingangszustände 1 sind.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)
NEGIERTES UND	NOT AND	Negative Konjunktion	$Z = \overline{A \wedge B}$

Schaltwerttabelle

a	b	a⊼b
0	0	1
0	1	1
1	0	1
1	1	0

Schaltfunktion

f(a,b)=a⊼b

Schaltsymbol

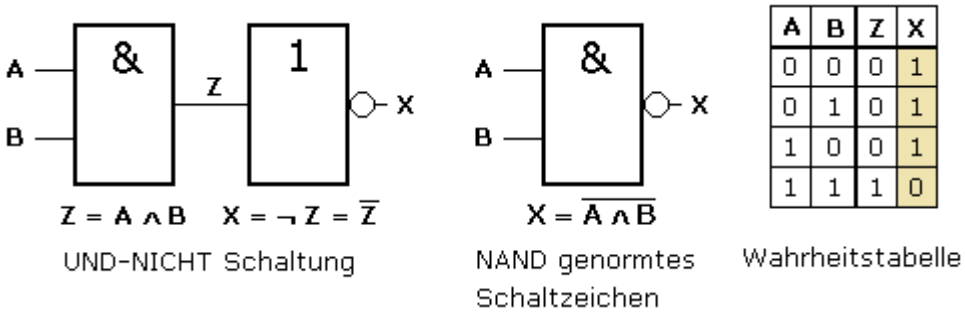


Bild 5-13: Schaltzeichen für NAND-Gatter

NOR-Verknüpfung

Eine Kombination aus einem **ODER-Gatter mit nachfolgendem NICHT-Gatter ergibt ein NOR-Gatter**. Die Ausgangsvariable Z des ODER-Gatters wird durch das NICHT-Gatter negiert und erzeugt die Ausgangsvariable X des NOR-Glieds. NOR-Glieder haben in logischen Schaltungen die gleiche wichtige Bedeutung wie NAND-Glieder.

Der Ausgangszustand eines NOR-Glieds ergibt 1, wenn alle Eingangszustände 0 sind.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)
NEGIERTES ODER	NOT OR	Negative Disjunktion	$Z = \overline{A \vee B}$

Schaltwerttabelle

a	b	a⊽b
0	0	1
0	1	0
1	0	0
1	1	0

Schaltfunktion

f(a,b)=a⊽b

Schaltsymbol

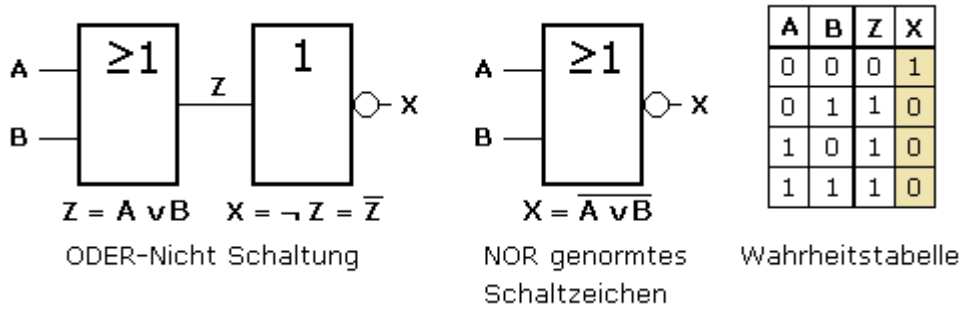


Bild 5-15: Schaltzeichen für NOR-Gatter

XOR-Verknüpfung

Die logische Verknüpfung des XOR-Gatters mit zwei Eingangsvariablen kann mit einem **'entweder - oder'** umschrieben werden. Die **Ausgangsvariable wird immer dann 'true' liefern, wenn die Eingangsvariablen unterschiedliche Zustände haben**. Die Wahrheitstabelle des XOR-Gatters entspricht dem ODER-Gatter mit dem Ausschluss gleicher Eingangszustände, also exklusiv der Äquivalenz. Dieses Verhalten wird als Antivalenz bezeichnet.

Der Ausgangszustand eines XOR-Glieds ergibt 1, wenn eine ungerade Zahl der Eingangszustände 1 und alle anderen Eingangszustände 0 sind.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)
Exklusives ODER (Antivalenz)	Exclusiv OR	Exklusive Disjunktion	Z=a⊻b

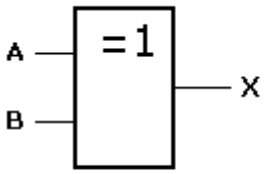
Schaltwerttabelle

a	b	a⊻b
0	0	0
0	1	1
1	0	1
1	1	0

Schaltfunktion

f(a,b)=a⊻b

Schaltsymbol



Exklusiv ODER
XOR – Antivalenz

$X = (A \wedge \overline{B}) \vee (\overline{A} \wedge B)$
$X = A \underline{\vee} B$
$X = A \oplus B$

Funktionsgleichung

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Wahrheitstabelle

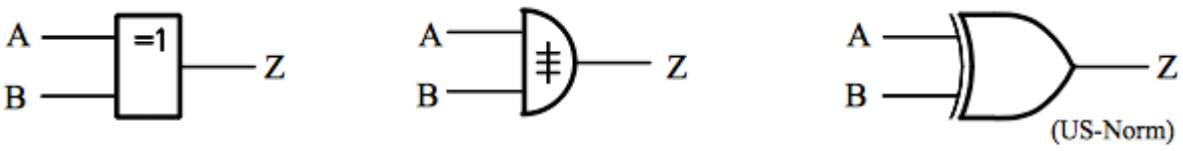


Bild 5-21: Schaltzeichen für ANTIVALENZ-Gatter

XNOR-Verknüpfung

Die Bezeichnung **Äquivalenz bedeutet Gleichwertigkeit**. Beim XNOR-Gatter mit zwei Eingangsvariablen ist der **Zustand der Ausgangsvariable 1, wenn beide Eingangsvariablen den gleichen Zustand 0 oder 1 haben**. Die Funktion kann durch eine Schaltung mit vier NOR-Gattern erreicht werden. Es sind zwei unterschiedliche Schaltzeichen zu finden.

Der Ausgangszustand eines XNOR-Glieds ergibt 1, wenn eine gerade Zahl der Eingangzustände 1 und alle anderen Eingangszustände 0 aufweisen oder alle 0 sind.

Zeichen

Deutsch	Englisch	Fachausdruck	Zeichen (boolsche Algebra)
Exklusives NICHT ODER (Äquivalenz)	Exclusiv NOT OR	Biimplikation	Z=a≡b

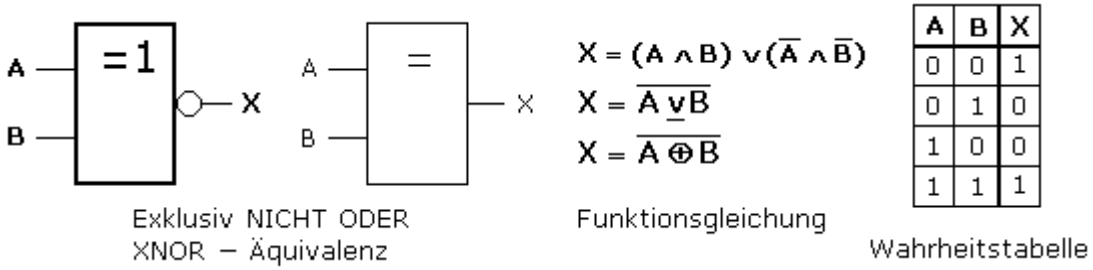
Schaltwerttabelle

a	b	a≡b
0	0	1
0	1	0
1	0	0
1	1	1

Schaltfunktion

f(a,b)=a≡b

Schaltsymbol



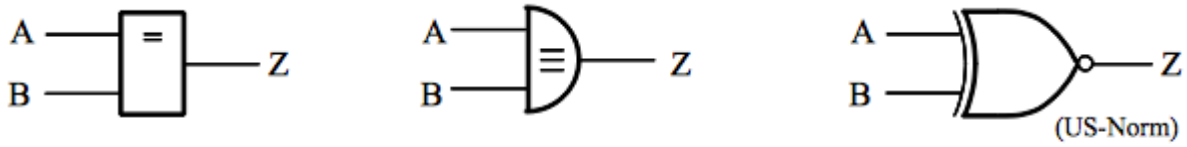


Bild 5-19: Schaltzeichen für ÄQUIVALENZ-Gatter

xor_nand.swf

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:2:2_03

Last update: **2018/09/23 16:10**

Gesetze der Schaltalgebra

Vorrang- und Klammerregel

Wie in der Algebra ist auch in der Digitaltechnik auf eine bestimmte Reihenfolge der Operationen zu beachten. Die Negation sollte vor der Konjunktion (UND) und diese vor der Disjunktion (ODER) ausgeführt werden. Das ist vergleichbar mit der Aussage, dass die Multiplikation Vorrang vor der Addition hat.

Bei mehreren Variablen und unterschiedlichen Verknüpfungen kann eine Klammersetzung notwendig sein. Beachtet man die Vorrangregel, kann man auf viele Klammern verzichten. Ein Setzen zeigt sogleich eindeutig, welche der Variablen wie zu verknüpfen ist. Bei ODER sollte immer geklammert werden, ebenfalls beim Anwenden der De Morganschen Gesetze auf NAND- und NOR-Verknüpfungen.

$X = A \vee B \wedge C$

nach der Vorrangregel gilt

$\boxed{X = A \vee (B \wedge C)}$

festgelegte Abfolge der Verknüpfungen

$\boxed{Z = (A \vee B) \wedge C}$

A	B	C	B∧C	A∨(B∧C)	A∨B	(A∨B)∧C
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	1	0
0	1	1	1	1	1	1
1	0	0	0	1	1	0
1	0	1	0	1	1	1
1	1	0	0	1	1	0
1	1	1	1	1	1	1

$A \vee (B \wedge C) \neq (A \vee B) \wedge C$

Bei drei Eingangsvariablen und binären Zuständen gibt es 2^3 = 8 Eingangskombinationen. Die Wahrheitstabelle zeigt bei definierter Klammersetzung oder Beachtung der Vorrangregel UND vor ODER unterschiedliche Ergebnisse.

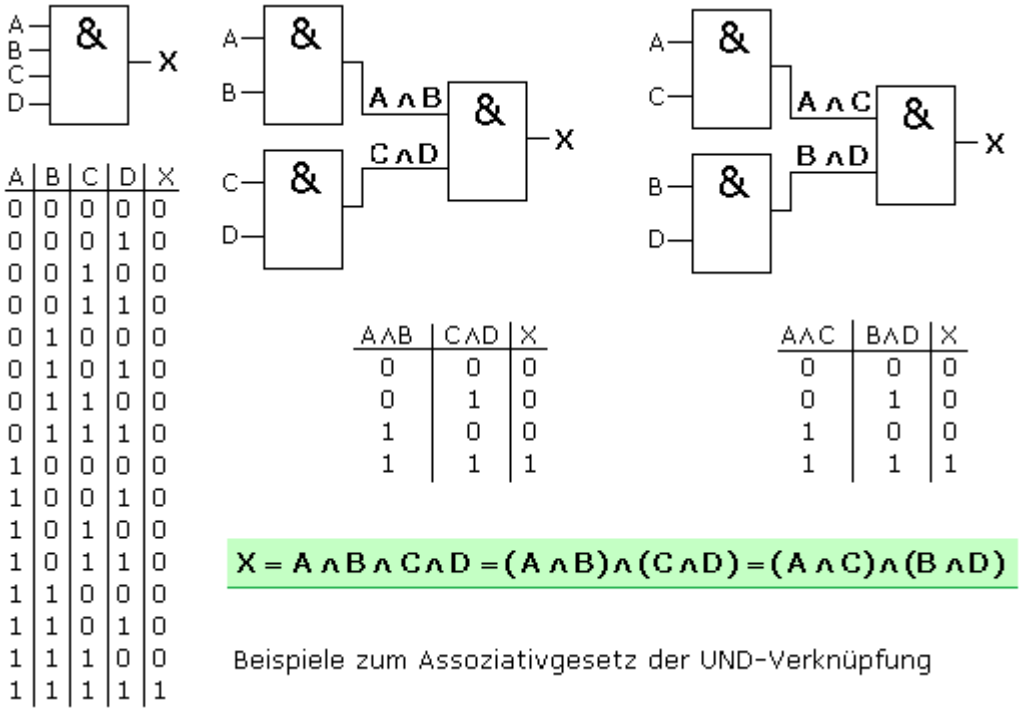
Kommutativgesetz

Das Kommutativ- oder Vertauschungsgesetz besagt, dass bei der UND sowie ODER Verknüpfung auch bei beliebiger Vertauschung der Reihenfolge der Eingangsvariablen das Ergebnis gleich bleibt.

UND	$X = A \wedge B \wedge C = C \wedge A \wedge B = B \wedge C \wedge A$
ODER	$X = A \vee B \vee C = C \vee A \vee B = B \vee C \vee A$

Assoziativgesetz

Das Assoziativ- oder Verbindungsgesetz besagt, dass bei einer UND- beziehungsweise ODER-Verknüpfung mit mehr als zwei Schaltvariablen die Verknüpfung auch stufenweise nacheinander in beliebiger Reihenfolge erfolgen kann.



A	B	C	D	X
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

A∧B	C∧D	X
0	0	0
0	1	0
1	0	0
1	1	1

A∧C	B∧D	X
0	0	0
0	1	0
1	0	0
1	1	1

$$X = A \wedge B \wedge C \wedge D = (A \wedge B) \wedge (C \wedge D) = (A \wedge C) \wedge (B \wedge D)$$

Beispiele zum Assoziativgesetz der UND-Verknüpfung

Anstelle des UND-Gatters mit vier Eingangsvariablen lassen sich zwei UND-Gatter mit zwei Eingängen verwenden. Die Variablen A, B, C, D werden einzeln beliebig mit den Eingängen verbunden. Die Ausgangsvariable der beiden UND-Gatter kann den Wert 0 oder 1 annehmen. Beide Ausgangsvariablen sind nochmals mit einem UND-Gatter zu verknüpfen, wobei sich wieder vier Eingangskombinationen ergeben. Nur wenn alle Eingangsvariablen 1 sind, wird der Ausgangszustand ebenfalls 1.

Das Assoziativgesetz gilt gleichermaßen für die ODER-Verknüpfung. Die Klammersetzung ist nicht notwendig und soll nur verschiedene Verteilungen besser erkennbar machen.

Distributivgesetz

Das Distributiv- oder Verteilungsgesetz wird zur Vereinfachung von Verknüpfungsgleichung angewendet. Es ist vergleichbar mit dem Ausmultiplizieren und Ausklammern von Variablen der normalen Algebra. Da die logische UND-Verknüpfung der algebraischen Multiplikation entspricht, während die ODER-Verknüpfung mit der Addition vergleichbar ist, gibt es zwei unterschiedliche Distributivgesetze.

Konjunktives Distributivgesetz

Eine Variable wird mit UND verknüpft und auf den Folgeausdruck verteilt. Die Vorgehensweise entspricht dem aus der Algebra bekannten Ausmultiplizieren eines Klammerausdrucks mit einem Faktor. Im umgekehrten Fall kann eine Variable, die mit mehreren anderen Variablen verknüpft ist, ausgeklammert werden. Das bedeutet für den Schaltungsaufbau eine Einsparung an Gattern.

$$X = A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$$

⇨ Ausmultiplizieren
⇦ Ausklammern

$$X = A \wedge (A \vee B)$$
$$X = (A \wedge A) \vee (A \wedge B)$$
$$X = A \vee (A \wedge B) = A$$
$$X = A$$

		UND	ODER	
A	B	A∧B	A	X
0	0	0	0	0
0	1	0	0	0
1	0	0	1	1
1	1	1	1	1

Beim Ausklammern wird die Variable mit ihrem Verknüpfungszeichen, hier UND, vor die Klammer gesetzt. Die in der Klammer stehenden Variablen werden mit dem zuvor zwischen den Klammern stehenden ODER verknüpft.

Disjunktives Distributivgesetz

Eine Variable kann durch ein vergleichbares Ausmultiplizieren auf andere Ausdrücke verteilt werden oder bei mehrfachem Auftreten ausgeklammert werden. Beim Ausklammern wird das ODER Verknüpfungszeichen mit der Variablen vor die Klammer geschrieben. disjunktives Distributivgesetz

$$X = A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$$

⇨ Ausmultiplizieren
⇦ Ausklammern

$$X = A \vee (A \wedge B)$$
$$X = (A \vee A) \wedge (A \vee B)$$
$$X = A \wedge (A \vee B)$$
$$X = A$$

		ODER	UND	
A	B	A∨B	A	X
0	0	0	0	0
0	1	1	0	0
1	0	1	1	1
1	1	1	1	1

De Morgansche Gesetze

Die boolesche Algebra wurde vom englischen Mathematiker De Morgan weiter entwickelt. Für die Schaltalgebra gibt es zwei De Morgansche Gesetze. Sie machen Aussagen zur Negation einer Verknüpfung und der Umkehr von Verknüpfungszeichen. Mit den De Morganschen Gesetzen lassen sich bei der Entwicklung von Digitalschaltungen Gatter gegeneinander austauschen, Schaltungen verkleinern oder mit nur einem Gattertyp verwirklichen. Das erste Gesetz ist für die NAND-Verknüpfung und das zweite Gesetz entsprechend für die NOR-Verknüpfung definiert.

$$X = \overline{A \wedge B} = \overline{A} \vee \overline{B}$$

1. De Morgansche Gesetz

			ODER		
A	B	$\overline{A \wedge B}$	$\overline{A}$	$\overline{B}$	X
0	0	1	1	1	1
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	0	0	0

$$X = \overline{A \vee B} = \overline{A} \wedge \overline{B}$$

2. De Morgansche Gesetz

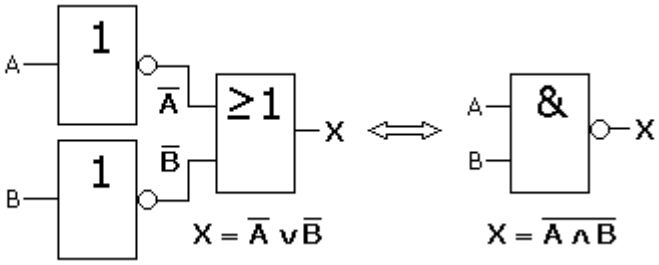
			UND		
A	B	$\overline{A \vee B}$	$\overline{A}$	$\overline{B}$	X
0	0	1	1	1	1
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	0

Ist eine Verknüpfung insgesamt negiert, so ist ihre Ausgangsvariable negiert. Die Negation kann auf die Einzelglieder aufgeteilt werden, wobei aus der Grundverknüpfung UND ein ODER beziehungsweise aus ODER ein UND wird. Eine doppelte Negation hebt sich auf. Getrennte Negationsstriche über

Variablen bedeuten, dass die Eingangsvariablen negiert sind.

Die oben zum 1. De Morganschen Gesetz gezeigte Verknüpfung kann schaltungstechnisch mit zwei NICHT- und einem ODER-Gatter verwirklicht werden. Wie zu erkennen ist, folgt das gleiche Ergebnis mit nur einem NAND-Gatter. Für das 2. De Morgansche Gesetz gilt die entsprechende Aussage. Zwei NICHT- und ein UND-Gatter reduzieren sich auf den Einsatz eines NOR-Gatters. Mit der doppelten Negation und den De Morganschen Gesetzen kann bei einer gegebenen Funktionsgleichung auf das Bestehen einer derartige Vereinfachung geprüft werden. Treten dabei mehrfache Negationen auf, so lassen sie sich von innen nach außen auflösen.

$$X = \overline{A} \vee \overline{B}$$
$$\overline{\overline{X}} = \overline{\overline{A} \vee \overline{B}} \quad \overline{v} \rightarrow \wedge$$
$$\overline{\overline{X}} = \overline{\overline{A} \wedge \overline{B}} \quad \overline{\overline{A}} = A$$
$$X = \overline{A \wedge B}$$
$$\boxed{X = \overline{A \wedge B} = \overline{A} \vee \overline{B}}$$



From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:2:2_04

Last update: **2018/09/30 11:26**

Digitale Rechenschaltungen

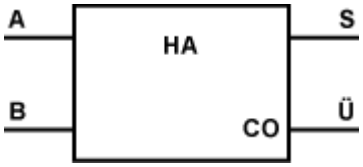
Aus logischen Verknüpfungen lassen sich digitale Schaltungen zusammenbauen, mit denen man Rechenvorgänge durchführen kann. Das heißt, diese Schaltungen haben zwischen ihren Eingängen eine Kombination aus logischen Verknüpfungen, die einem Rechenvorgang entspricht. In der Digitaltechnik kennt man Rechenschaltungen hauptsächlich für das duale Zahlensystem und den BCD-Code. Im Prinzip kann für jedes Zahlensystem eine Rechenschaltung aufgebaut werden.

Einige Rechenschaltungen der Digitaltechnik

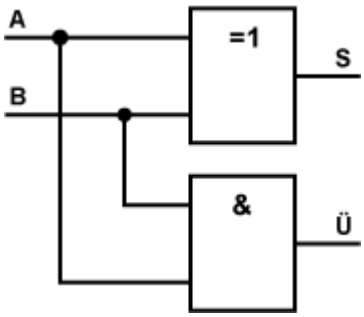
- Addiererschaltungen
- Subtrahiererschaltungen
- Addier-Subtrahier-Werke
- Multiplikationsschaltungen
- Arithmetisch-logische Einheit (ALU)

Stellvertretend für alle Rechenschaltungen dienen die folgenden Ausführungen zum Halbaddierer und dem Volladdierer.

Halbaddierer



Ein Halbaddierer ist die einfachste Rechenschaltung und kann zwei einstellige Dualziffern addieren. Der Eingang A des Halbaddierers ist der Summand A, der Eingang B ist der Summand B. Die Schaltung hat zwei Ausgänge. Der Ausgang S als Summenausgang (2^0) und der Ausgang Ü als Übertrag (2^1) für die nächsthöhere Stelle im dualen Zahlensystem.



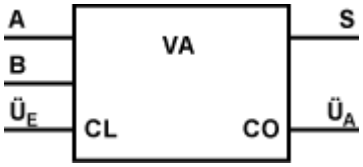
B	A	Ü	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Der Halbaddierer ist eine Schaltung aus den Verknüpfungsgliedern XOR und UND. Das XOR ist die Addier-Verknüpfung. Das UND stellt fest, ob ein Übertrag für die nächsthöhere Stelle vorgenommen

werden muss. Aus der Tabelle ist ersichtlich, dass das Ergebnis aus der Spalte Summe (S) einer Exklusiv-ODER-Verknüpfung (Antivalenz, XOR) entspricht. Das Ergebnis der Spalte Übertrag (Ü) entspricht einer UND-Verknüpfung. Die so entstandene Schaltung wird als Halbaddierer bezeichnet. Sie ist in der Lage zwei 1-Bit Summanden (einstellig) zu addieren.

Volladdierer

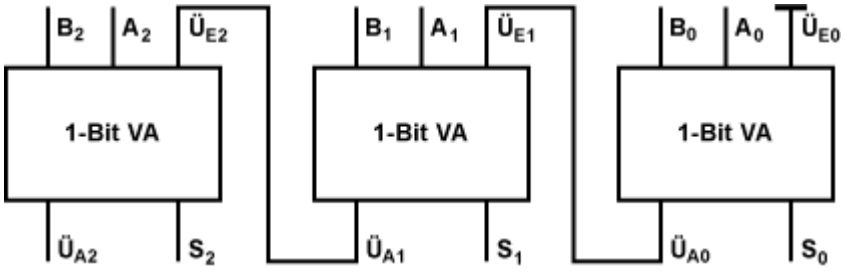
Um **mehrstellige Dualzahlen** addieren zu können benötigt man Schaltungen die auch einen Übertrag einer niederwertigen Stelle berücksichtigen. Man spricht vom **Übertragseingang ÜE**. Die Schaltung bezeichnet man als **Volladdierer (VA)**. Ein Volladdierer kann drei Dualzahlen addieren. Bzw. zwei Dualzahlen addieren und den Übertrag aus einer niederwertigen Stelle berücksichtigen.



Folgende Wahrheitstabelle ergibt sich:

$Ü_E$	B	A	$Ü_A$	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

3-Bit-Volladdierer



Folgende Schaltung zeigt einen 3-Bit-Volladdierer (VA), der aus drei 1-Bit-Volladdierer (VA) realisiert wurde. Damit lassen sich zwei dreistellige Dualzahlen addieren. Der Eingang $Ü_{E0}$ liegt an 0 V (Masse), weil in der niederwertigsten Stelle kein Übertrag berücksichtigt werden muss.

Beispiel

```
 010 = A
+111 = B
```

```
---
1001 = S
```

0. Stelle

$Ü_{E0}$	A_0	B_0	$Ü_{A0}$	S_0
0	0	1	0	1

1. Stelle

$Ü_{E1}$	A_1	B_1	$Ü_{A1}$	S_1
0	1	1	1	0

2. Stelle

$Ü_{E2}$	A_2	B_2	$Ü_{A2}$	S_2
1	0	1	1	0

Ergebnis

$Ü_{A2}$ S_2 S_1 S_0

```
1  0  0  1
```

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:2:2_05

Last update: **2018/10/07 08:15**

2.07) Übungen

1) Um ihren Swimmingpool schneller füllen zu können, hat Fam. Huber zwei Wasserleitungen (beide verfügen über einen Absperrhahn B und C verlegt, die direkt in den Pool führen. Diese Leitungen zweigen beide von einer Hauptleitung ab, die ebenfalls über einen Absperrhahn A verfügt. Demnach kann Wasser nur in den Pool fließen, wenn der Hauptabsperrhahn A und zumindest einer der anderen Absperrhähne offen sind.
Stelle die Situation mittels einer Wahrheitstabelle und einer Schaltfunktion (LogikSim, …) dar.

2) Frau Müller sagt zu ihrem Mann: „Den ganzen Sonntag hockst du auf dem Sofa, trinkst Bier und stopfst dir Kartoffelchips in den Schlund! Du solltest mal joggen!
Ihr Gatte erwidert: „Ich verspreche dir für nächsten Sonntag Folgendes:

- Wenn ich jogge, werde ich sogar auf Bier oder auf Chips verzichten.
- Wenn ich keine Chips esse, trinke ich kein Bier oder ich jogge nicht.
- Wenn ich aber jogge, brauche ich unbedingt hinterher ein Bier!
- Allerdings: Auch falls ich nicht jogge, werde ich auf jeden Fall Bier oder Chips zu mir nehmen.“

Falls Herr Müller seine Versprechen einhält, wie könnten seine Aktivitäten am nächsten Sonntag aussehen? Erstelle einen schaltalgebraischen Ausdruck und die Wahrheitstabelle zu diesen Aussagen.

3) Zeichne für folgende schaltalgebraischen Ausdrücke die zugehörige Schaltfunktion bzw. erstelle die jeweilige Wahrheitstabelle!

- $f(A,B) = (\neg A \vee \neg B)$
- $f(A,B,C) = (\neg A \wedge \neg B) \wedge C$
- $f(A,B) = (\neg A \vee \neg B) \vee (\neg A \wedge B)$

4) Überprüfe das Gesetz von De Morgan mit Hilfe einer Leitwerttabelle und stelle die Schaltung im Digitalsimulator dar.

5) Überprüfe das Distributivgesetz mittels einer Leitwerttabelle und stelle die Schaltung im Digitalsimulator dar.

6) Vereinfache die Schaltfunktion $f(a,b)= a \wedge (\neg a \vee b)$ und baue die Schaltung im Digitalsimulator auf.

7) Vereinfache die Schaltfunktion $f(a,b)= (a \vee b) \wedge (a \vee \neg b) \wedge (\neg a \vee b)$

8) Vereinfache die Schaltfunktion $f(a,b)= (a \wedge b) \vee (a \wedge \neg b) \vee (\neg a \wedge b)$

9) Zur automatischen Brandbekämpfung wurden drei Sensoren in einem Raum angebracht. Melden mindestens zwei der drei Sensoren eine Rauchentwicklung, so schaltet sich die Sprinkleranlage ein. Entwirf eine Leitwerttabelle, eine Schaltfunktion und stelle diese mittels Digitalsimulator dar!

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:2:2_07

Last update: **2018/10/07 08:20**

Zeichencodierung

Eine **Zeichenkodierung (englisch Character encoding, kurz Encoding)** erlaubt die eindeutige Zuordnung von Schriftzeichen (i. A. Buchstaben oder Ziffern) und Symbolen innerhalb eines Zeichensatzes. In der elektronischen Datenverarbeitung werden Zeichen über einen Zahlenwert kodiert, um sie zu übertragen oder zu speichern. Der deutsche Umlaut Ü wird zum Beispiel im ISO-8859-1-Zeichensatz mit dem Dezimalwert 220 kodiert. Im EBCDIC-Zeichensatz kodiert derselbe Wert 220 die geschweifte Klammer }. Zur richtigen Darstellung eines Zeichens muss also die Zeichenkodierung bekannt sein; der Zahlenwert allein reicht nicht aus.

Zahlenwerte aus Zeichenkodierungen lassen sich auf verschiedene Art speichern oder übertragen, z. B. als Morsezeichen, verschieden hohe Töne (Faxgerät), verschieden hohe Spannungen.

Geschichte des Character Encoding

Mit der Entwicklung des Computers begann die Umsetzung der im Grunde schon seit dem Baudot-Code verwendeten binären Zeichenkodierung in Bit-Folgen, bzw. intern meist in verschiedene elektrische Spannungswerte als Unterscheidungskriterium, ganz analog zu der bisher zur Unterscheidung der Signalwerte genutzten Tonhöhe oder Signaldauer.

Um diesen Bit-Folgen darstellbare Zeichen zuzuordnen, mussten Übersetzungstabellen, sogenannte Zeichensätze, engl. Charsets, festgelegt werden. 1963 wurde eine erste **7-Bit-Version des ASCII-Codes durch die ASA (American Standards Association)** definiert, um eine **Vereinheitlichung der Zeichenkodierung** zu erreichen. Obwohl IBM an der Definition mitgearbeitet hatte, führte man 1964 einen eigenen **8-Bit-Zeichencode EBCDIC** ein. Beide finden bis heute in der Computertechnik Verwendung.

Da für viele Sprachen jeweils unterschiedliche diakritische Zeichen benötigt werden, mit denen Buchstaben des lateinischen Schriftsystems modifiziert werden, gibt es für viele Sprachgruppen jeweils eigene Zeichensätze. Die **ISO** hat mit der **Normenreihe ISO 8859 Zeichenkodierungen für alle europäischen Sprachen** (einschließlich Türkisch) und Arabisch, Hebräisch sowie Thai standardisiert.

Das **Unicode Consortium** schließlich veröffentlichte 1991 eine erste Fassung des gleichnamigen Standards, der es sich zum Ziel gesetzt hat, alle Zeichen aller Sprachen in Codeform zu definieren. **Unicode** ist gleichzeitig die **internationale Norm ISO 10646**.

Bevor ein Text elektronisch verarbeitet wird, muss der verwendete Zeichensatz und die Zeichenkodierung festgelegt werden. Dazu dienen beispielsweise folgende Angaben:

Definition des Zeichensatzes in einer HTML-Seite

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
```

Definition des Zeichensatzes in den Kopfzeilen (Headern) einer E-Mail oder eines HTTP-Pakets

```
Content-Type: text/plain; charset="ISO-8859-1"
```

ASCII - American Standard Code for Information Interchange

Der **American Standard Code for Information Interchange (ASCII, deutsch „Amerikanischer Standard-Code für den Informationsaustausch“)** ist eine **7-Bit-Zeichenkodierung**; sie entspricht der US-Variante von ISO 646 und dient als Grundlage für spätere, auf mehr Bits basierende Kodierungen für Zeichensätze.

Der ASCII-Code wurde zuerst am 17. Juni 1963 von der **American Standards Association (ASA) als Standard** ASA X3.4-1963 gebilligt und 1967/1968 wesentlich sowie zuletzt im Jahr 1986 von ihren Nachfolgeinstitutionen aktualisiert. Die Zeichenkodierung definiert **128 Zeichen**, bestehend aus **33 nicht druckbaren** sowie **95 druckbaren Zeichen**.

ASCII-Code Tabelle

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

www.VirtualUniversity.ch

Fakten

- sehr verbreiteter Standard, u.a. in PCs (Hardware-Ebene)
- von ISO genormt

- ursprünglich 7-Bit-Code, also 128 Zeichen
- davon einige nach nationalem Bedarf abgewandelt z.B. deutsche Umlaute statt [ ] { } \ |
- unterschiedliche 8-Bit-Erweiterungen mit zusätzlichen Zeichen im Bereich 128 – 255
- Erweiterungen oft problematisch bei älteren Rechnern, im Internet1) etc. → deshalb E-Mail, HTTP etc. auf 7 Bit beschränkt (→ 2. Sem.)

UTF8 - UCS Transformation Format

UTF-8 (Abk. für 8-Bit UCS Transformation Format, wobei UCS wiederum Universal Character Set abkürzt) ist die am weitesten verbreitete Kodierung für Unicode-Zeichen (Unicode und UCS sind praktisch identisch). Die Kodierung wurde im September 1992 von Ken Thompson und Rob Pike bei Arbeiten am Plan-9-Betriebssystem festgelegt.

UTF-8 ist in den **ersten 128 Zeichen (Indizes 0–127) deckungsgleich mit ASCII** und eignet sich mit in der Regel nur **einem Byte Speicherbedarf** für Zeichen vieler westlicher Sprachen besonders für die Kodierung englischsprachiger Texte, die sich im Regelfall ohne Modifikation daher sogar mit nicht-UTF-8-fähigen Texteditoren ohne Beeinträchtigung bearbeiten lassen, was einen der Gründe für den Status als **De-facto-Standard-Zeichenkodierung des Internets** und damit verbundener Dokumenttypen darstellt. Im Oktober 2017 verwendeten **89,9 % aller Websites UTF-8**

In anderen Sprachen ist der Speicherbedarf in Byte pro Zeichen größer, wenn diese vom ASCII-Zeichensatz abweichen: Bereits die **deutschen Umlaute erfordern zwei Byte**, ebenso griechische oder kyrillische Zeichen. Zeichen fernöstlicher Sprachen und von Sprachen aus dem afrikanischen Raum belegen dagegen bis zu 4 Byte je Zeichen. Da die Verarbeitung von UTF-8 als Multibyte-Zeichenfolge wegen der notwendigen Analyse jedes Bytes im Vergleich zu Zeichenkodierungen mit fester Byteanzahl je Zeichen mehr Rechenaufwand und für bestimmte Sprachen auch mehr Speicherplatz erfordert, werden abhängig vom Einsatzszenario auch andere UTF-Kodierungen zur Abbildung von Unicode-Zeichensätzen verwendet: Microsoft Windows als meistgenutztes Desktop-Betriebssystem verwendet intern als Kompromiss zwischen UTF-8 und UTF-32 etwa UTF-16 Little Endian

Fakten Unicode (UTF-8, UTF-16, UTF-32)

- neuere Kodierung, ebenfalls ISO-standardisiert 1993 und heutzutage Standard
- Ziel: Berücksichtigung möglichst vieler Sprachen und ihrer Eigenheiten
- Buchstaben-, Silben-, und Ideogrammsprachen
- Schreibrichtungen (links-rechts, rechts-links, oben-unten)
- außerdem diverse Sonderzeichen, mathematisch-technische Symbole, Diakritika, geometrische Formen, Pfeile, Piktogramme u.v.m.
- dafür 16-Bit-Darstellung (erlaubt 65.536 Zeichen)
- Erweiterung auf 32 Bit für künftigen Bedarf
- Standard auf unixoiden Betriebssystemen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:3

Last update: **2018/10/09 13:50**

Hardware

Hardware ist der Oberbegriff für die **physischen Komponenten (die elektronischen und mechanischen Bestandteile)** eines datenverarbeitenden Systems, als **Gegenteil zu Software (den Programmen und Daten)**

Wortherkunft

„Hardware“ kommt ursprünglich aus dem Englischen und bedeutet übersetzt Eisenwaren.

Hardware vs. Software

- Hardware = sind alle greifbaren/sichtbaren Elemente eines PCs
- Software = Programme und Daten, also nicht greifbare Elemente eines PCs

Überblick

- 4.1) EVA-Prinzip
- 4.2) Von-Neumann-Architektur
- 4.3) HW-Komponenten
- 4.4) Schnittstellen

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

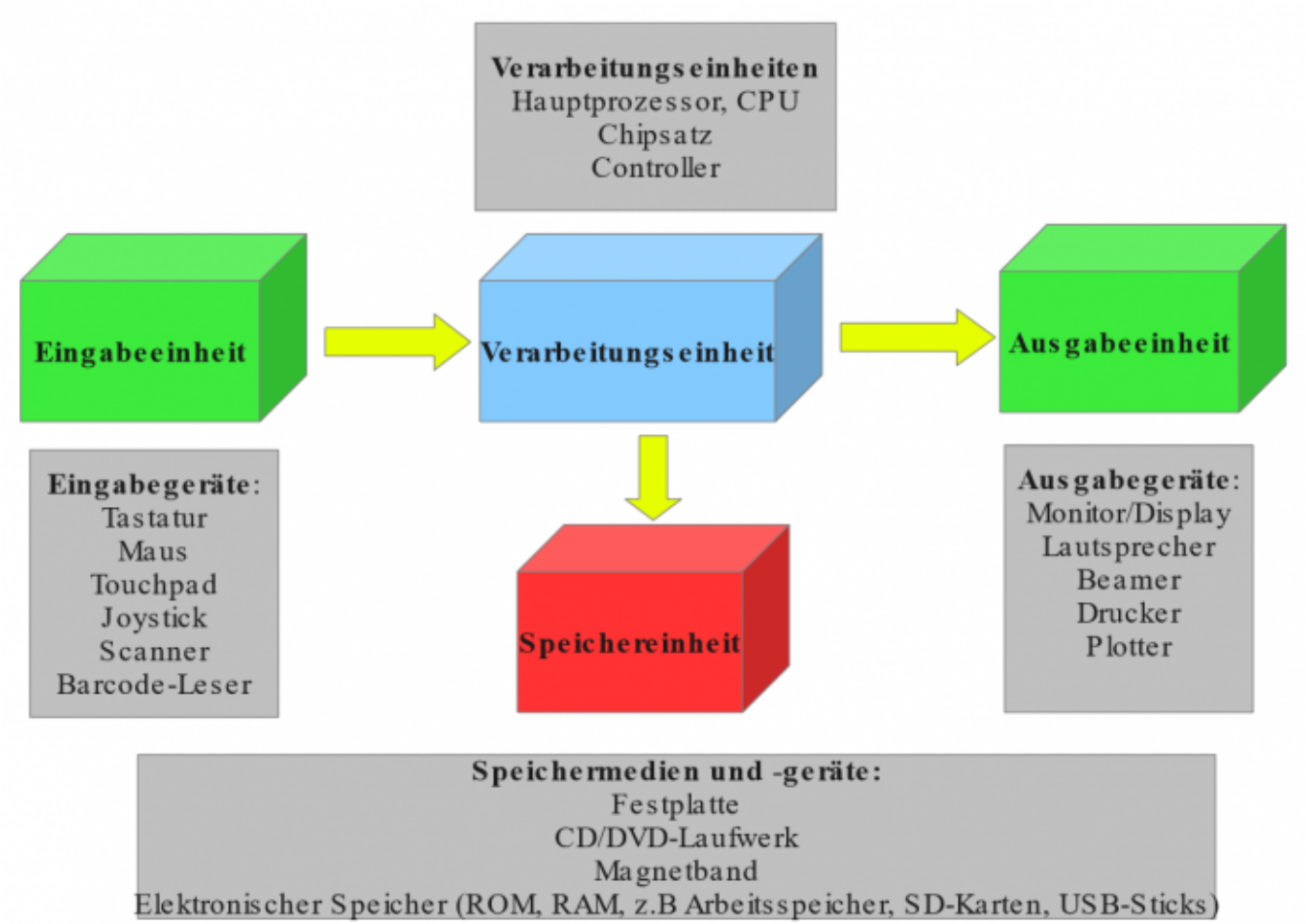
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:4

Last update: **2018/10/23 15:38**

EVA/IPO Prinzip

Das **EVA**-Prinzip ist ein Grundprinzip für die Datenverarbeitung und beschreibt die Reihenfolge in der Daten verarbeitet werden.

Eingabe (**I**nput) - **V**erarbeitung (**P**rocess) - **A**usgabe (**O**utput)



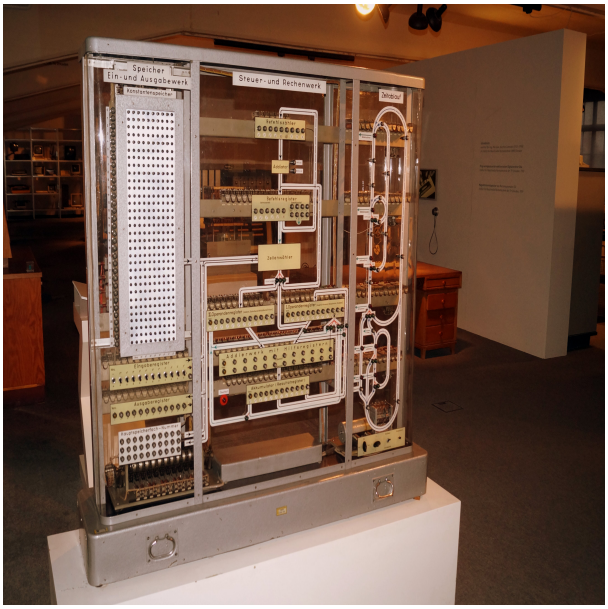
From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:4:4_01

Last update: **2018/10/09 14:33**

Von-Neumann-Architektur (VNA)

Die **V**on-**N**eumann-**A**rchitektur ist ein Modell für Computer und bietet die Grundlage für alle heutigen Computer. Das Modell wurde von Johann von Neumann im Jahr 1945 entwickelt. Heute ist Johann von Neumann unter seinem amerikanischen Namen John von Neumann bekannt.

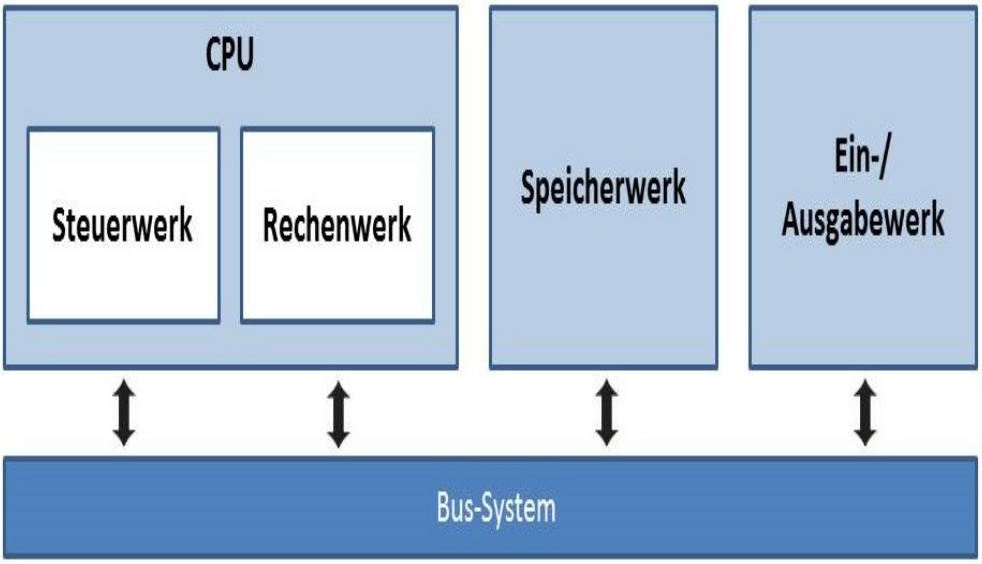


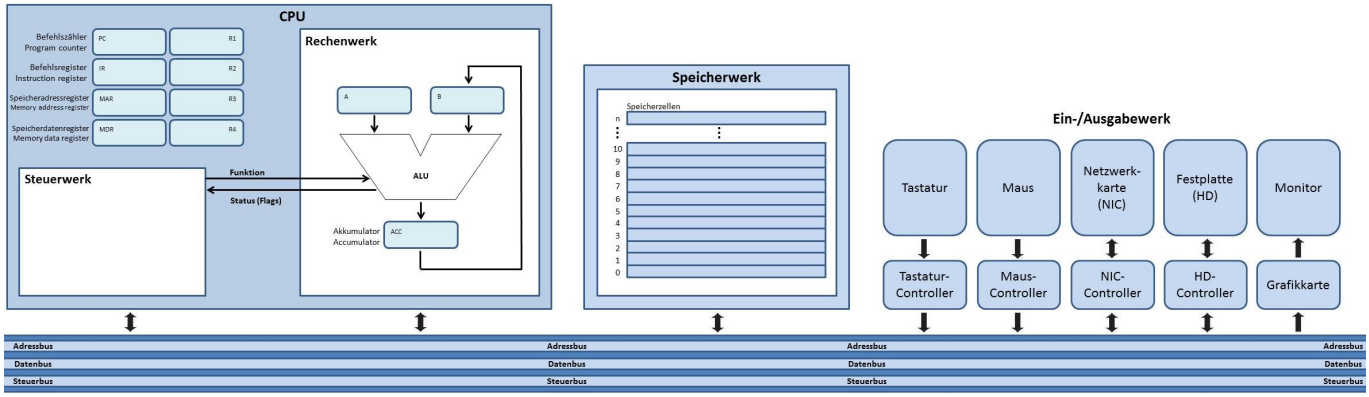
Er revolutionierte die bisherigen Computer, da durch seine Architektur verschiedene Programme auf derselben Hardware laufen konnten.

Komponenten

Ein Von-Neumann-Rechner beruht auf folgenden Komponenten, die bis heute in Computern verwendet werden:

- ALU (Arithmetic Logic Unit) – Rechenwerk, selten auch Zentraleinheit oder Prozessor genannt, führt Rechenoperationen und logische Verknüpfungen durch. (Die Begriffe Zentraleinheit und Prozessor werden im Allgemeinen in anderer Bedeutung verwendet.)
- Control Unit – Steuerwerk oder Leitwerk, interpretiert die Anweisungen eines Programms und verschaltet dementsprechend Datenquelle, -senke und notwendige ALU-Komponenten; das Steuerwerk regelt auch die Befehlsabfolge.
- Bussystem: Dient zur Kommunikation zwischen den einzelnen Komponenten (Steuerbus, Adressbus, Datenbus)
- Memory – Speicherwerk speichert sowohl Programme als auch Daten, welche für das Rechenwerk zugänglich sind.
- I/O Unit – Eingabe-/Ausgabewerk steuert die Ein- und Ausgabe von Daten, zum Anwender (Tastatur, Bildschirm) oder zu anderen Systemen (Schnittstellen).





Register

- **Befehlszähler (Program Counter - PC)**

Enthält die Speicheradresse vom Speicherwerk des aktuellen Befehls. (Startadresse = 0x0000). Wird nach jedem Befehl um 1 erhöht.

- **Befehlsregister (Instruction Register - IR)**

Speichert den vom Speicherwerk zurückbekommenen Befehl.

- **Speicheradressregister (Memory Address Register - MAR)**

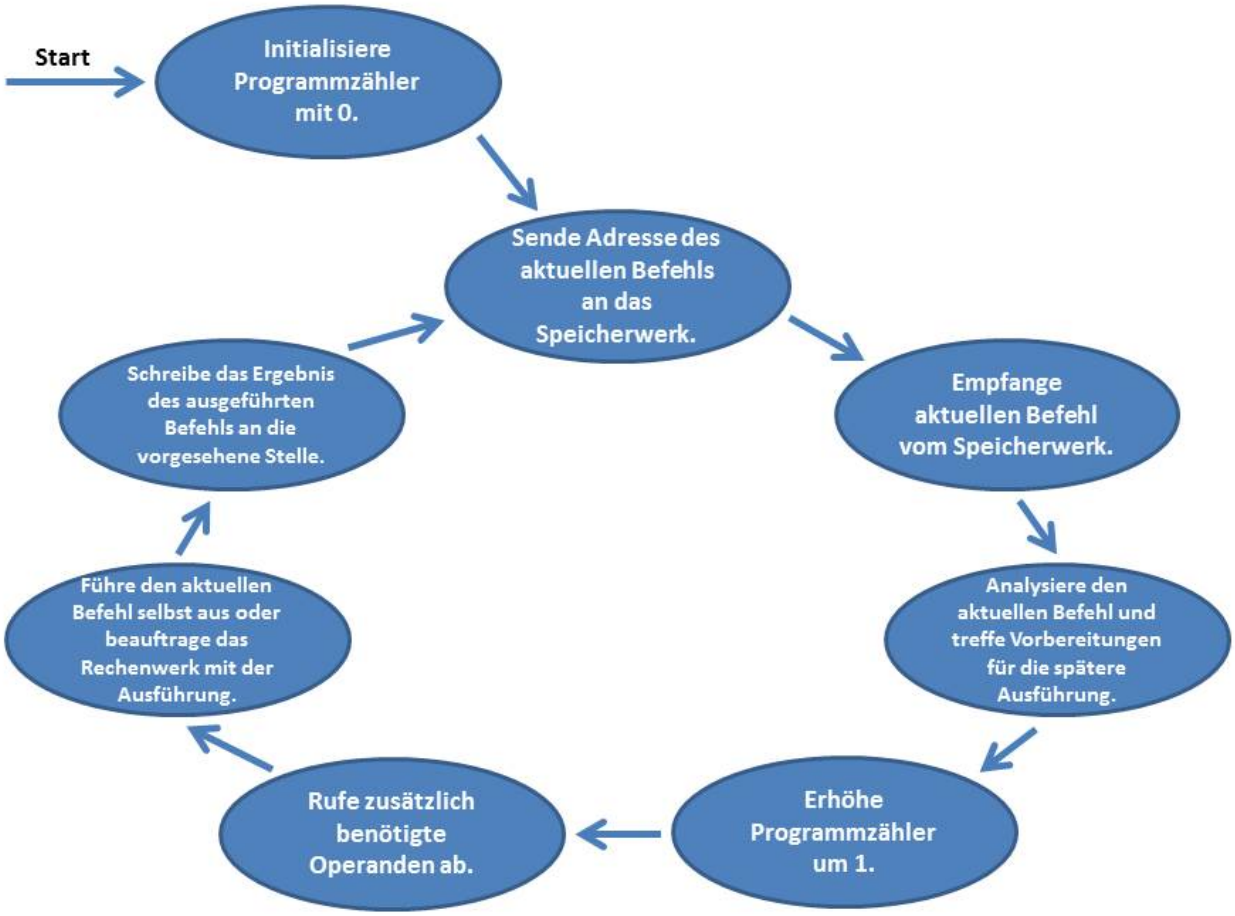
Ausschließlich für die Kommunikation zwischen Steuerwerk und Rechenwerk. Im MAR legt das Steuerwerk jeweils die Adresse ab, welche im Speicherwerk angesprochen werden soll.

- **Speicherdatenregister (Memory Data Register - MDR)**

Ausschließlich für die Kommunikation zwischen Steuerwerk und Rechenwerk. Bei einem Lesezugriff auf die Speicherzelle wird der vom Speicherwerk über den Datenbus bereitgestellte Wert im MDR abgelegt und kann von hier aus weiter verarbeitet werden. Bei einem Schreibzugriff muss sich im MDR der zu schreibende Wert befinden, so dass er über den Datenbus an das Speicherwerk übermittelt werden kann.

Prozesszyklus

1. Initialisiere das Befehlszählerregister (Program Counter- PC) mit 0 (Start)
2. Sende Adresse des aktuellen Befehls zum Speicherwerk
3. Empfange aktuellen Befehl vom Speicherwerk und speichere diesen in das Befehlsregister (Instruction Register - IR)
4. Analysiere aktuellen Befehl und treffe Vorbereitungen für die spätere Ausführung (Welcher Befehl und was ist dazu notwendig?)
5. Erhöhe den Befehlszähler (PC) um 1
6. Rufe zusätzlich benötigte Operanden ab (z.B.: Befehl ADD OP1 OP2)
7. Führe den Befehl selbst (Steuerwerk) aus oder beauftrage das Rechenwerk für die Ausführung
8. Schreibe das Ergebnis des ausgeführten Befehls an die vorgesehene Stelle



Arbeitsweise des Steuer- und Rechenwerks

Arbeitsweise des Speicherwerks

Befehlszähler und Befehlsregister im Zusammenspiel mit dem Bus-System

Arbeitsweise des Steuerwerks

Die 7 Prinzipien der Von-Neumann-Architektur

- Rechner besteht aus fünf Funktionseinheiten
- Struktur des Rechners ist unabhängig vom zu bearbeitenden Problem. Zur Lösung eines Problems muss Programm im Speicher abgelegt werden.
- Programme, Daten und Ergebnisse werden im selben Speicher abgelegt.
- Der Speicher ist in fortlaufenden nummerierten Zellen unterteilt. Über die Adresse einer Speicherzelle kann auf den Inhalt zugegriffen werden.
- Aufeinanderfolgende Befehle eines Programms werden in aufeinanderfolgende Speicherzellen abgelegt.
- Durch Sprungbefehle kann von der gespeicherten Befehlsreihenfolge abgewichen werden.
- Es gibt zumindest
    - arithmetische Befehle (Addition, Subtraktion, Multiplikation)
    - logische Befehle (EQUAL, NOR, AND, OR)
    - Transportbefehle, z.B. von Speicher zu Rechenwerk und für Ein- und Ausgabe
- Alle Daten (Befehle, Adressen, usw.) werden binär kodiert

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:4:4_02

Last update: **2018/10/09 14:32**

Hardware-Komponenten

Grundbestandteile

- Stromversorgung
- 4.3.1) Motherboard/Mainboard - Hauptplatine
- 4.3.2) Central Processing Unit (CPU) - zentrale Recheneinheit (Prozessor)
- 4.3.3) Random Access Memory (RAM) - Arbeitsspeicher

Massenspeicher

- 4.3.4) Festplatte (SSD, SATA)
- 4.3.5) Laufwerke (CD, DVD, Band,..)


SAMSUNG
Solid State Drive
SAMSUNG
Solid State Drive


ASUS
QuieTrack

Erweiterungskarten

- 4.3.6) Grafikkarte
- 4.3.7) Soundkarte
- 4.3.8) Netzwerkkarte

Peripheriegeräte

Eingabegeräte

- 4.3.9) Maus & Tastatur
- 4.3.10) Scanner

Ausgabegeräte

- 4.3.11) Bildschirm
- 4.3.12) Drucker

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

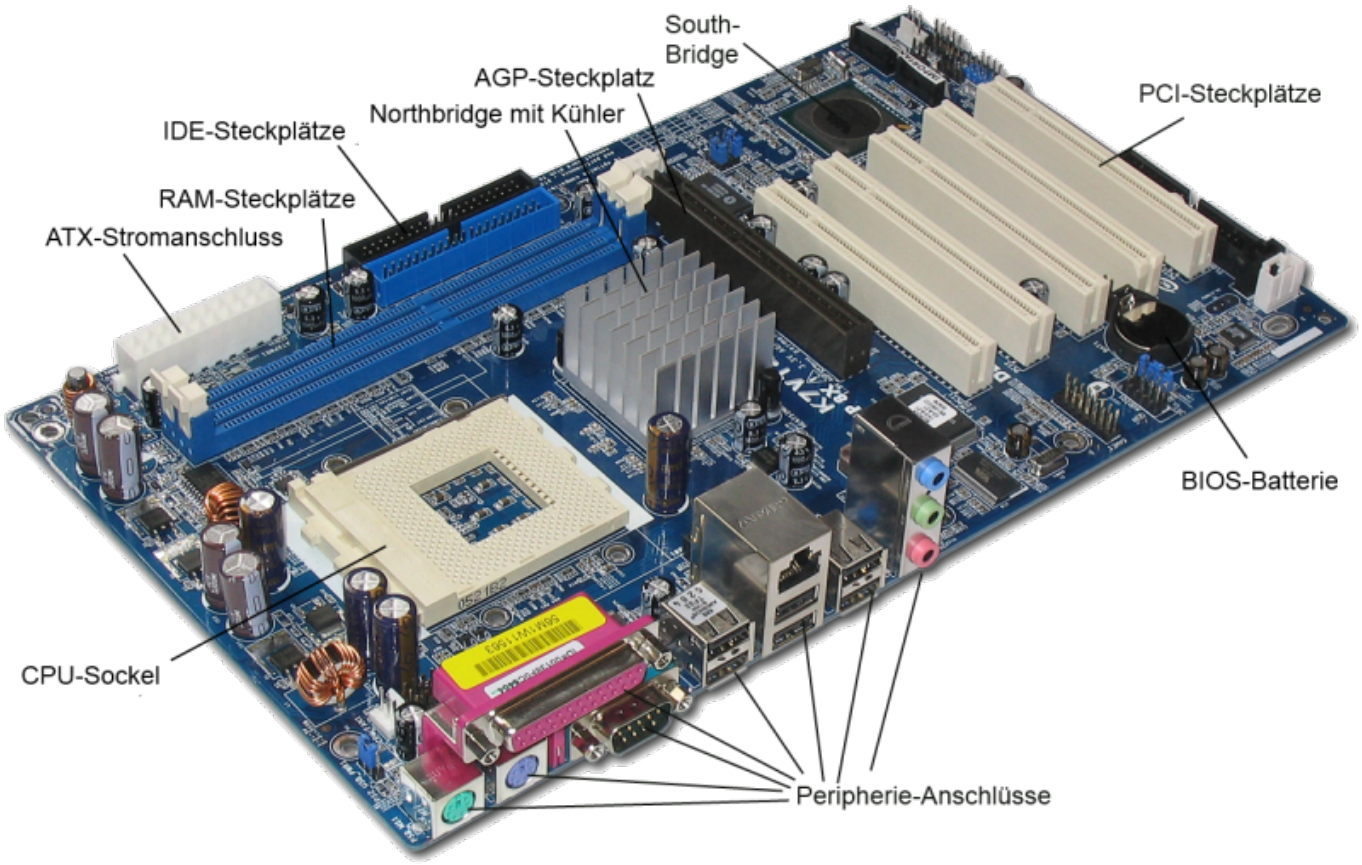
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:4:4_03

Last update: **2018/10/23 15:40**

Motherboard / Mainboard / Hauptplatine

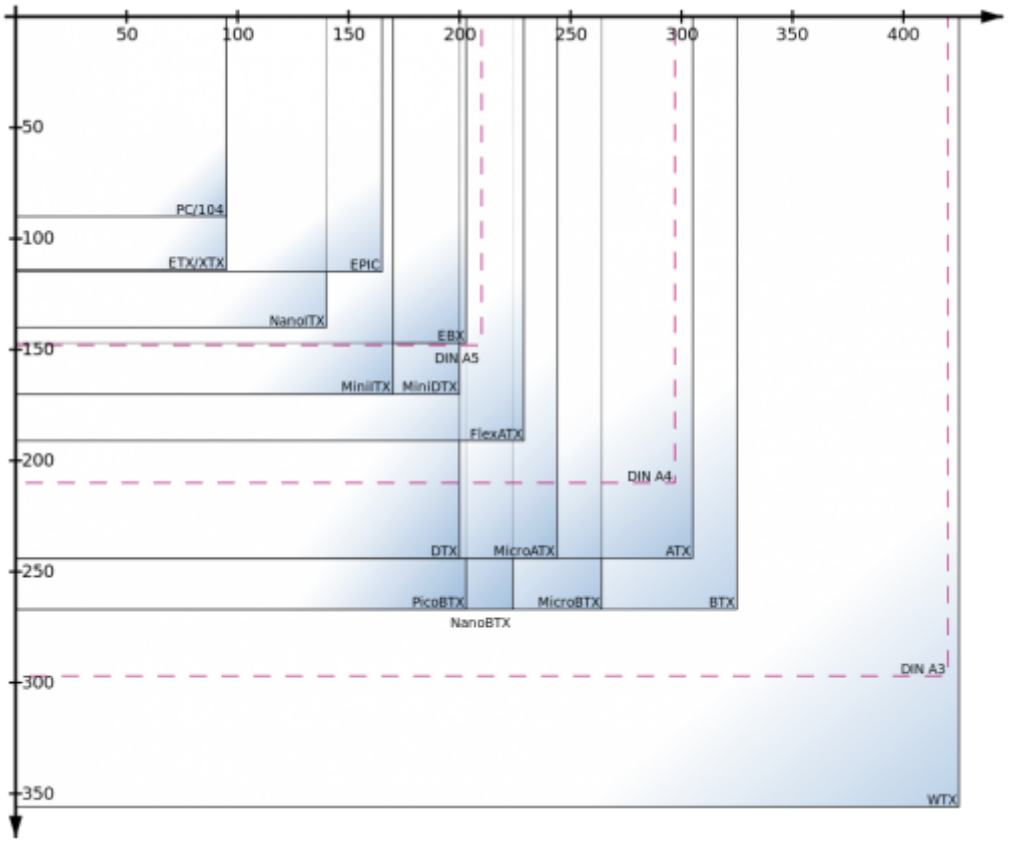
Das Motherboard ist der **Grundbaustein eines Computers**. Das Motherboard ist die Platine, auf der alle Systemkomponenten eines Computers eine **physikalische und logische Verbindung** erhalten. Die wichtigsten und einige leistungsbeeinflussenden Bauteile sind fest auf dieser Platine miteinander verbunden.

Die **Ausstattung des Motherboards bestimmt die System-Leistung, Erweiterbarkeit und Zukunftsfähigkeit eines Computersystems**. Die meisten Motherboards sind auf eine bestimmte Anwendung mit einem bestimmten Prozessor zugeschnitten. Man kann also nicht jeden beliebigen Prozessor auf jedem Motherboard verwenden. Der **Einsatz eines Prozessors hängt vom Motherboard bzw. vom Prozessorsockel und dem Chipsatz ab**.



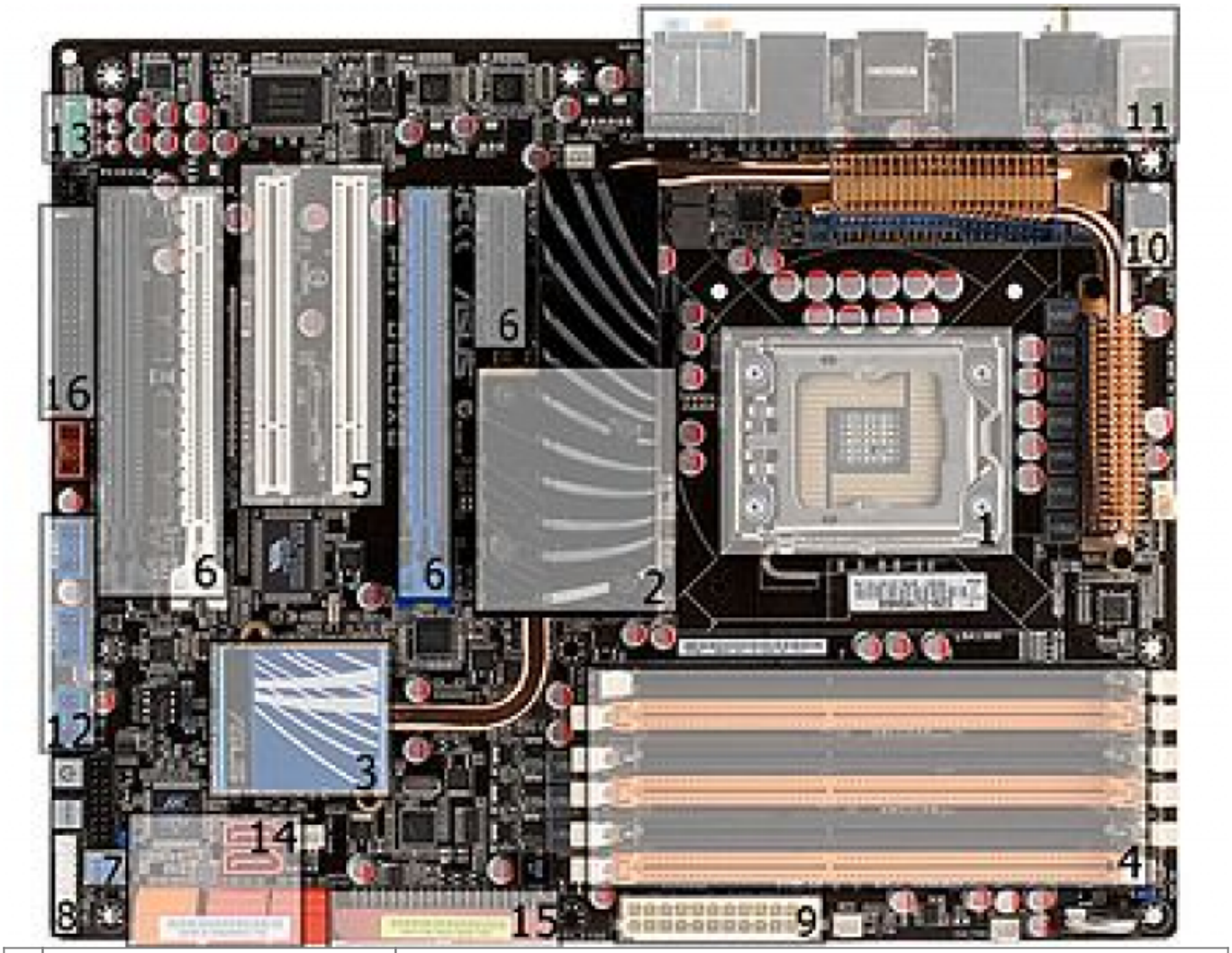
Bauformen

Das Format der Hauptplatinen wird nach dem **Formfaktor** unterschieden. Seit 1995 ist das **ATX-Format aktuell**. Es löste das AT-Format ab und brachte diverse Umstellungen an Gehäusen und Netzteilen mit sich. Es existieren diverse Variationen von ATX und AT, um auch kompaktere Geräte ohne proprietäre Formate bestücken zu können, etwa Baby AT oder µATX. Mini-ITX, Flex-ATX und Micro-ATX passen in ATX-Gehäuse. **Steckplätze, Schrauben** und das **Fenster für I/O-Shield** befinden sich an einheitlichen oder **derselben Position**.



Komponenten eines Motherboards

Auf einem Motherboard können je nach Bauform, Ausstattung und Integrationsdichte unterschiedliche Systemkomponenten zu finden sein.



Nr	Komponente	Beschreibung
1	**Prozessorsockel**	Der Prozessor, der auch als Hauptprozessor bezeichnet wird, hat auf dem Motherboard einen eigenen Steckplatz bzw. Sockel (z.B.: LGA775 oder LGA1155).
2	**Chipsatz - Northbridge**	Der Chipsatz ist das Bindeglied zwischen den einzelnen Systemkomponenten eines Computers. Egal was in einem Computer passiert, der Chipsatz hat immer damit zu tun. Er sorgt dafür, dass alle Komponenten über unterschiedliche Schnittstellen miteinander kommunizieren können. Dabei werden unterschiedliche Spannungspegel, Taktfrequenzen und Protokolle berücksichtigt und untereinander umgewandelt.
3	**Chipsatz - Southbridge**	
4	**RAM-Steckplatz**	Steckplätze für den RAM (z.B.: hier für DDR3)
5	**PCI-Steckplatz**	für Erweiterungskarten (z.B.: Audiokarte)
6	**PCI-Express-Steckplatz**	drei PCIe-x16 Slots (Grafikkarte) & 1 PCIe-x1-Slot (Erweiterungskarten)
7	**Jumper**	Kurzschlussstecker zur Aktivierung und Deaktivierung von Einstellungen (z.B. Übertakten)
8	**Anschlüsse Frontblende**	Hier werden die Schalter (Power, Reset, Lampen für FP) von vorne an das Mainboard angebunden
9	**24-poliger ATX-Connector**	Haupstromversorgung für das Mainboard
10	**8-poliger-ATX-Connector**	Anschluss für die Stromversorgung der CPU
11	**Externe Anschlüsse**	Anschlüsse ragen aus dem Gehäuse nach hinten hinaus und dienen als Anschluss für die Peripheriegeräte (Maus, Tastatur,..)
12	**Interne USB-Anschlüsse**	USB-Anschlüsse für die Vorderseite des Gehäuses

Nr	Komponente	Beschreibung
13	**AAFP-Soundanschluss**	An diesem Anschluss werden die Sound Ein- und Ausgänge der Front des Gehäuses angeschlossen
14	**Serial-ATA-Anschlüsse**	Ermöglicht die Verbindung von Festplatten oder optischen Laufwerken mit der Hauptplatine
15	**IDE-Anschluss**	Ermöglicht die Verbindung von Festplatten oder optischen Laufwerken mit der Hauptplatine
16	**Floppy-Disk-Stecker**	Anschluss des Diskettenlaufwerks

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:4:4_03:4_03_01

Last update: **2018/11/01 09:49**

CPU - Central Processing Unit

Der Prozessor, Hauptprozessor oder die CPU ist heutzutage das **Herzstück eines jeden elektronischen Geräts**. Er wird in Smartphones, Taschenrechnern und in Computern, für die er eigentlich erfunden wurde, eingesetzt. Eine Welt ohne diese Rechengenies ist undenkbar.
Die bekanntesten Prozessoren stammen von den Unternehmen **Intel und AMD**.

In einem **Computersystem** kann es **mehrere Prozessoren** geben. Wenn man vom Prozessor spricht, dann ist damit in der Regel immer der Hauptprozessor gemeint. Wegen seiner **zentralen Stellung** wird die Bezeichnung **„Zentrale Verarbeitungseinheit“** verwendet.
Der Hauptprozessor ist die **Funktionseinheit** in einem Computer, der **die eigentliche Verarbeitungsleistung erbringt**. Der Hauptprozessor ist für die **Informationsverarbeitung und die Steuerung der Verarbeitungsabläufe zuständig**. Dazu holt sich der Prozessor **aus dem Speicher nacheinander die Befehle** und **veranlasst die Informationsverarbeitung**.
Neben dem Hauptprozessor gibt es noch **weitere Prozessoren**, die den Hauptprozessor von der Arbeit entlasten. Der **Grafikprozessor (GPU)** ist ein solcher Prozessor.

Prozessor-Hersteller

Es gibt 2 marktführende Prozessore Hersteller, **AMD** und **INTEL**.
Seit jeher gilt der Prozessor-Hersteller AMD als ewiger Zweiter im Kampf um die Marktanteile im PC-Prozessor-Markt. Dazu kommt, dass AMD lange Zeit von dem am stärksten wachsenden Markt, den Notebooks, abgekoppelt war. Mit wenigen Ausnahmen, waren die meisten AMD-Prozessoren in Low-Cost-PCs für Privatkunden verbaut. In den Bereichen, in denen richtig Geld verdient werden konnte, war Intel immer besser unterwegs.

Das Problem von AMD und INTEL ist, dass die Prozessoren von den beiden Herstellern jeweils komplett andere Sockel gebrauchen und somit ein Umstieg von einem Hersteller zum anderen auch meistens mit einem neuen Motherboard-Kauf einhergeht.

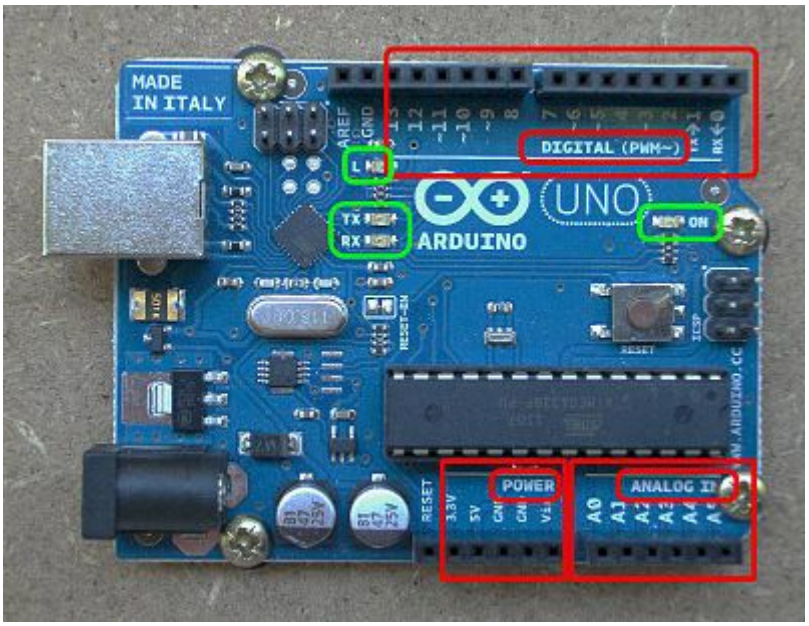
Mikroprozessor

Der Mikroprozessor ist ein Prozessor, der vollständig in einem einzigen Schaltkreis untergebracht ist. Der Prozessor im Personal Computer ist ein solcher Mikroprozessor.

⇒ Ein **Mikroprozessor ist die Vereinigung von Rechenwerk und Steuerwerk eines Computers mit den dazugehörigen Daten-, Adress- und Steuerleitungen auf einem Chip**

Mikrocontroller

Ein Mikrocontroller ist ein **Prozessor**, der über **zusätzliche digitale Ein- und Ausgänge** verfügt und für Steuerungsaufgaben vorgesehen ist. Er wird bereits **als vollständiger Computer** angesehen, der im Embedded-Bereich eingeordnet wird.



Moderne Prozessortechnik

Die **Entwicklung der Prozessoren** verlief lange Zeit nach einem scheinbar einfachen Gesetz. Man **optimierte den internen Aufbau**, **verkleinerte die Strukturen**, **senkte die Spannung**, **erhöhte die Taktfrequenz** oder verbesserte den Herstellungsprozess.
Schon war die **nächste Prozessor-Generation** geboren. Doch diesem Spiel sind enge Grenzen gesetzt. So muss mit **zunehmender Chipgröße das Taktsignal immer längere Wege** zurücklegen. Damit der Zeitunterschied der Taktflanken im akzeptablen Bereich liegt, muss der **Takttreiber immer leistungsstärker** werden. Dadurch **erhöht er die Verlustleistung** des Prozessors. Deshalb wird die **Chipfläche regelmäßig verkleinert**. Dabei werden zwischen den Schaltelementen **immer dünnere Verbindungen** eingesetzt. Dadurch **steigt der Widerstand der Verbindungen** und die Signale bewegen sich immer langsamer. Das führt unter Umständen dazu, dass die Signallaufzeit unter der Verarbeitungszeit der Gatter liegt. Die **physikalischen Grenzen zeigten sehr schnell**, dass **insbesondere die Taktfrequenz nicht unendlich weit gesteigert werden konnte**. Die Taktfrequenz bestimmt unter anderem die entstehende Verlustleistung und damit **die Lebensdauer des Prozessors**. Deshalb wurden sehr bald andere leistungssteigernde Techniken entwickelt.

Während vor Jahren die Geschwindigkeit eines Prozessors immens wichtig war, ist heute ein **ausgewogenes System aus Prozessor, Arbeitsspeicher und Chipsatz** das Maß **für einen schnellen Computer**. Immer weniger Anwendungen benötigen die volle Rechenleistung eines aktuellen Prozessors. Aus diesem Grund bieten die heutigen Prozessoren viel mehr als nur reine Rechengeschwindigkeit. Sie haben **mehrere Kerne, nutzen Befehlssatzerweiterungen, intelligente Zwischenspeicher, verfügen über Virtualisierungstechnik und Grafikfunktionen**.

Multi-Core-Prozessoren

Multi-Core bedeutet, dass **in einem Prozessor mehrere Prozessor-Kerne** eingebaut sind. Man bezeichnet diese Prozessoren als Multi-Core- oder Mehrkern-Prozessoren. Rein **äußerlich unterscheiden sich** Multi-Core-CPUs **nicht von Single-Core-CPUs**. Der Rechenkern ist bei Multi-Core-CPUs einfach mehrfach vorhanden. Innerhalb des **Betriebssystems** wird der **Multi-Core-Prozessor wie mehrere Recheneinheiten** behandelt.

Je nach Anzahl der Kerne gibt es abgewandelte Bezeichnungen, die darauf hindeuten, wie viele Kerne im Prozessor integriert sind.

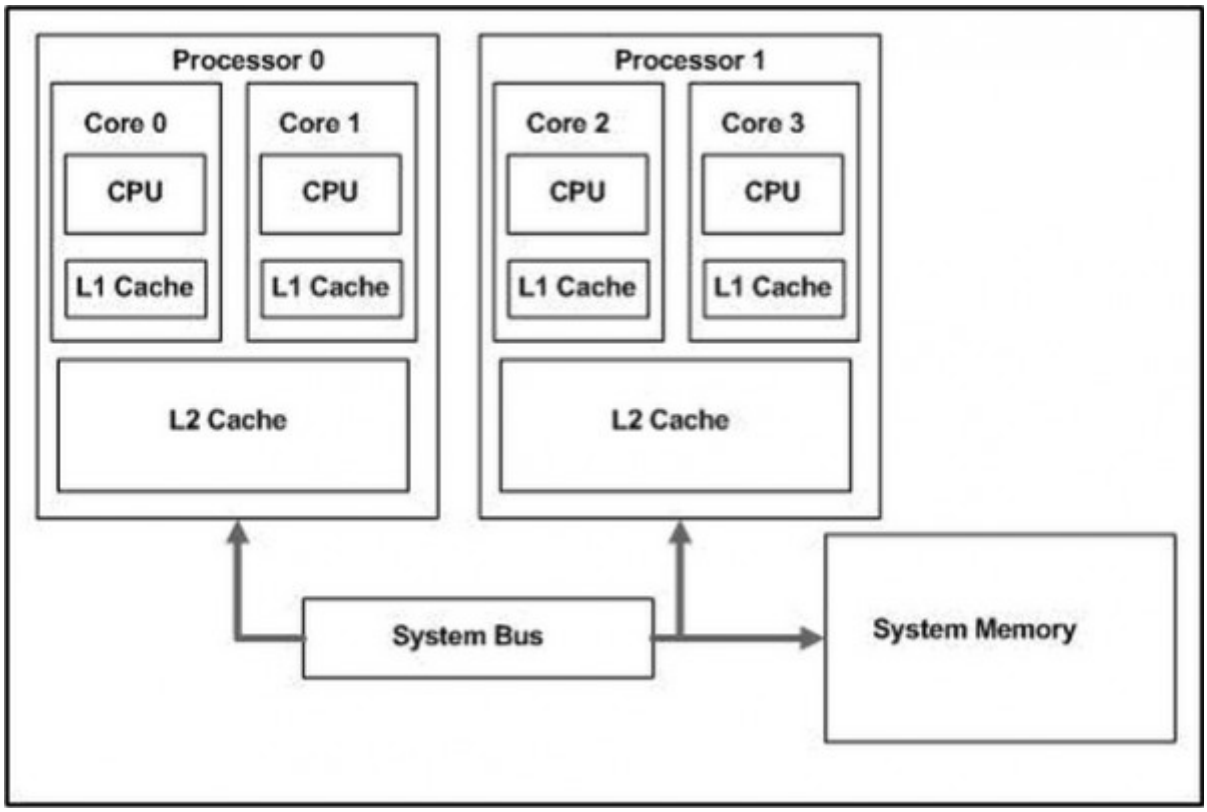
- Dual-Core (2)
- Triple-Core (3)
- Quad-Core (4)
- Hexa-Core (6)
- Octa-Core (8)
- ....

Unterschied Mehrprozessorsysteme vs. Mehrkernprozessor

Ein Mehrprozessorsystem ist ein Computersystem, das **mehr als einen Prozessorsockel** auf der

Multi-Prozessor-Hauptplatine besitzt, und in dem mehr als einer dieser Sockel auch bestückt ist.

Ein Mehrkernprozessor ist ein Prozessor mit mehreren Kernen.



In der obigen Abbildung ist erkennbar, dass dies 2 Prozessoren mit jeweils 2 Kernen sind. Also für das Betriebssystem insgesamt 4 Kerne zur Verfügung stehen!

Vom Takt-orientierten Prozessor zum Mehr-Kern-Prozessor

Um einen Prozessor schneller zu machen war die **Taktfrequenz lange Zeit der maßgebliche Faktor**, um **mehr Rechenleistung** aus einem Prozessor heraus zu bekommen. Leider hat die **Erhöhung der Taktfrequenz auch Nachteile**, die zu Folgeproblemen führen, die nur sehr schwer zu lösen sind.

- höhere Leistungsaufnahme
- höhere Wärmeentwicklung
- umfangreichere Kühlmaßnahmen
- lautere Computer durch aktive Kühlung

Schon eine **minimale Leistungssteigerung führt zu einem dramatisch höheren Energieverbrauch**. Die damit verbundene Leistungsaufnahme verhält sich proportional zur Taktfrequenz. Zusätzliche Transistoren und kleinere Halbleiterstrukturen erhöhen zusätzlich die Wärmeentwicklung. Die **Anforderungen an die Kühlung** sind mit den herkömmlichen Mitteln **nicht mehr zu leisten**. Gleichzeitig **verringert sich die Stabilität und Lebensdauer des Prozessors**.

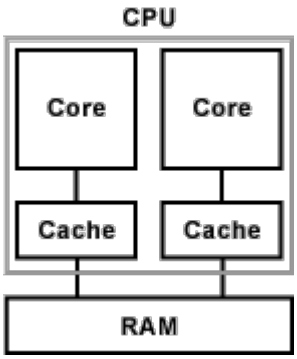
Eine **Alternative** zu immer höheren Taktraten sind **mehrere Rechenkerne**. Das bedeutet **mehr Leistung bei gleichzeitig geringerer Leistungsaufnahme**. Dabei werden die **einzelnen Kerne weit geringer getaktet**, als ein einziger Rechenkern. Insgesamt **steigt jedoch die**

Leistungsfähigkeit des Prozessors mit jedem weiteren Rechenkern.

Problematik

Grundsätzlich kann man die **Rechenleistung mehrerer Kerne nicht „addieren“**. Das würde voraussetzen, dass sich die vorliegenden Rechenaufgaben parallelisieren lassen. Die effektive Nutzung mehrerer Kerne erfordert es, dass die **Software so geschrieben ist, dass sie Daten parallel verarbeitet und so zum Beispiel mehrere Kerne gleichzeitig genutzt werden**. Das heißt, ein Problem bzw. die **Ausführung eines Programms** muss **in mehrere kleine Teilaufgaben zerlegt werden**, damit diese auf mehrere Kerne verteilt werden können. Doch leider sind die üblichen Anwendungen nicht auf parallele Ausführung ausgelegt und meist auch nicht notwendig.

Auswirkungen auf die Prozessor- und Computer-Architektur



In einer **Multi-Core-Architektur müssen** sich mehrere Rechenkerne die **vorhandenen Resourcen teilen**. Zum Beispiel **Arbeitsspeicher, Schnittstellen** usw. **Je mehr Kerne** ein Prozessor hat, **desto mehr Speicher und mehr Bandbreite zum Speicher ist erforderlich**. Aus diesem Grund ist der **Speicher-Controller nicht mehr im Chipsatz, sondern im Prozessor verankert**.

Arbeitsspeicher / Hauptspeicher

Gleichzeitig besteht das Problem, dass sich die **Rechenkerne darüber abstimmen** müssen, **wer gerade welche Daten im Cache** hält. In der Regel arbeiten die Rechenkerne mit einer **hierarchischen Cache-Struktur**. Dabei bekommt **jeder Kern einen eigenen L1- und L2-Cache**. Den **L3-Cache müssen sich die Kerne teilen**. Ein Cache-Kohärenz-Protokoll sorgt dafür, dass sich die Prozessorkerne bei der Nutzung des L3-Caches nicht in die Quere kommen.

Typische Anwendungen für Single-Core-Prozessoren

- Textverarbeitung
- Browser
- E-Mail
- Instant-Messaging
- Virenscanner
- MP3-Player

- einfache Bildbearbeitung (Schneiden, Skalieren, Farbkorrektur)

Typische Anwendungen für Multi-Core-Prozessoren

Gut parallelisierbare Anwendungen sind Schneiden von 4K-Videos, Computergrafiken mit Raytracing erzeugen oder aufwendige Softwareprojekte kompiliert. In diesen Fällen können im Prozessor nicht genug Kerne enthalten sein.

- CAD
- Simulation
- HD-Video
- Compiler
- 3D-Rendering
- professionelle Audio-Bearbeitung
- professionelle Bildbearbeitung
- Videoschnittprogramme

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

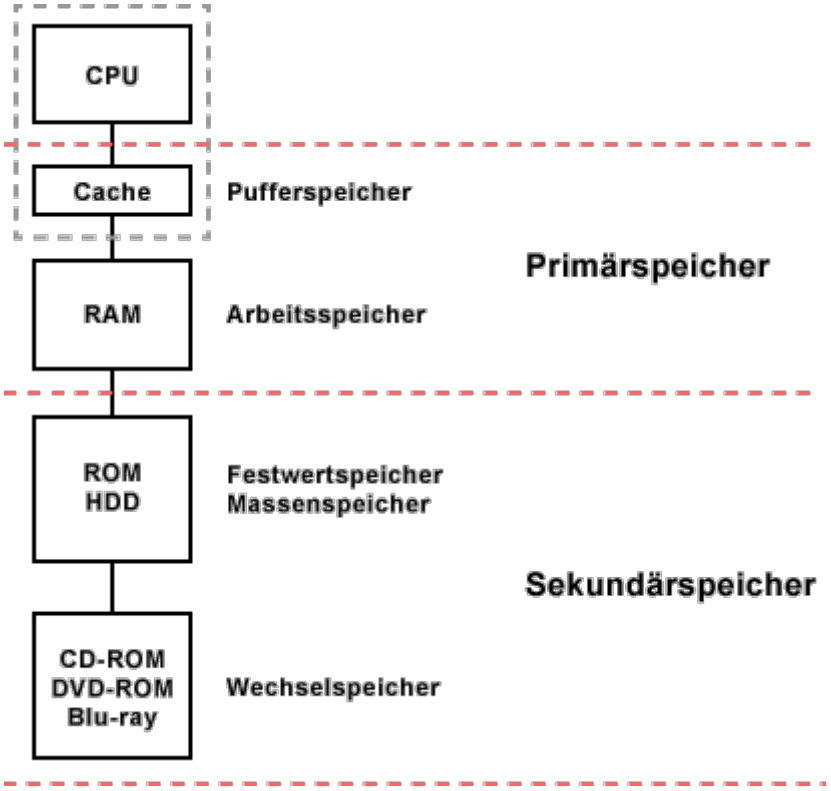
Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:4:4_03:4_03_02

Last update: **2018/11/01 10:49**

RAM (Random Access Memory)

RAM bezeichnet einen **Speichertyp dessen Speicherzellen über ihre Speicheradressen direkt angesprochen** werden können. In diesem Zusammenhang wird auch von **„wahlfrei“** gesprochen, was sich auf **„random“** bezieht und **nichts mit „Zufall“ oder „zufälligem Zugriff“ zu tun** hat. In diesem Zusammenhang spricht man von **„Speicher mit wahlfreiem Zugriff“** oder „Direktzugriffsspeicher“. Auf **andere Speicherarten (zum Beispiel Flash) kann man nur blockweise zugreifen**. **RAM erlaubt den Zugriff auf jede einzelne Speicherzelle**. Bei ROM (Read-Only-Memory, Nur-Lese-Speicher) funktioniert das genauso. Bei RAM funktioniert es lesend, wie auch schreibend. **Doch wird die Stromversorgung abgeschaltet gehen die Daten im RAM verloren**.

RAM wird in **Computer- und Mikrocontroller-Systemen als Arbeitsspeicher** eingesetzt. Weil in der Regel nur RAM als Arbeitsspeicher verwendet wird, wird **RAM gerne als Abkürzung für Arbeitsspeicher** verwendet. In diesem **Arbeitsspeicher werden Programme und Daten von externen Speicherträgern und Festplatten geladen**. Zur schnellen Verarbeitung kann der Prozessor darauf zugreifen, verarbeiten und danach wieder in den Arbeitsspeicher schreiben.



Prinzipiell **unterscheidet** man zwischen **statischem RAM und dynamischem RAM**. SRAM und DRAM sind flüchtige Halbleiterspeicher. Sie verlieren nach dem Ausschalten ihren Speicherinhalt.

- **SRAM - Static Random Access Memory** (Statisches RAM)
- **DRAM - Dynamic Random Access Memory** (Dynamisches RAM)

SRAM - Static Random Access Memory

SRAM ist ein statischer Halbleiterspeicher, was bedeutet, dass der **Speicherinhalt mittels Flip-Flops gespeichert wird und so nach dem Abruf des Speicherinhaltes erhalten bleibt**. Dadurch ist der **Stromverbrauch sehr hoch**, was aber zu einem **schnellen Arbeiten innerhalb des Speichers** führt. Aufgrund seines **hohen Preises** und des **großen Stromverbrauchs** wird SRAM **nur als Cache- oder Pufferspeicher** mit geringen Kapazitäten verwendet.

- Speicherung erfolgt in Flip-Flops
- sehr schnell
- kein Refresh nötig
- hoher Stromverbrauch
- Einsatz als L1-, L2- und L3-Cache

DRAM - Dynamic Random Access Memory

DRAM ist der **einfachste und billigste Speicher**. Im Computer-Bereich ist **DRAM als Arbeitsspeicher bzw. Hauptspeicher der bevorzugte Halbleiterspeicher**, den es in verschiedensten Varianten und Weiterentwicklungen gibt. Heute ist der SDRAM der am weitesten verbreitete Halbleiterspeicher in der Computertechnik. Im Gegensatz zum SRAM muss der **Speicherinhalt beim DRAM zyklisch aufgefrischt werden (Refresh)**. Dies ist normalerweise in Abständen von **einigen zig Millisekunden** erforderlich. Das Auffrischen des Speichers erfolgt zeilenweise.

- Kondensator als Speicherelement
- Speicherhaltung durch Refresh der Speicherzellen
- langsam
- geringer Stromverbrauch
- Einsatz als Arbeitsspeicher oder Hauptspeicher

Eine DRAM-Speicherzelle besteht aus einem Transistor und einem Kondensator (1T1C), der das eigentliche Speicherelement ist. In einer DRAM-Speicherzelle wird ein Bit durch die Ladung des Kondensators gespeichert. Die Messung der Spannung am Speicherkondensator (Lesen) und dessen anschließende Aufladung (Schreiben) benötigen eine gewisse Zeit.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:4:4_03:4_03_03

Last update: **2018/11/01 11:02**

Schnittstellen (PC)

Eine Schnittstelle **verbindet Systeme, die unterschiedliche physikalische, elektrische und mechanische Eigenschaften** besitzen. Die **Definition oder Spezifikation einer Schnittstelle enthält gemeinsame Eigenschaften**. Dazu gehört auch ein **Protokoll** für die **Kommunikation** und den **Datenaustausch**. Schnittstellen befinden sich überall dort, wo unterschiedliche Systeme miteinander verbunden werden müssen. Die Schnittstellen bilden den **Übergang von einem System in ein anderes System**. Dieser Übergang kann zur Kommunikation oder dem Datenaustausch verwendet werden.

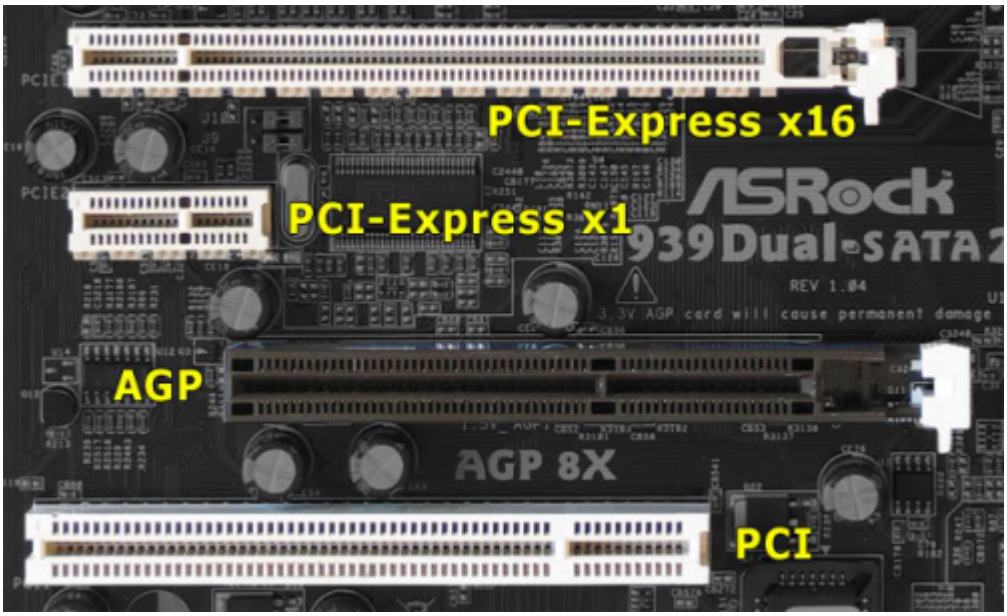
Ein Computer hat **interne** Schnittstellen, die sich **im Computer-Gehäuse** befinden und **externe** Schnittstellen, die **aus dem Computer-Gehäuse herausgeführt** sind.

Interne Schnittstellen

Interne Schnittstellen verbinden Systeme innerhalb eines Computers. Diese Schnittstellen werden meist auf dem **Motherboard als Sockel oder Slot** herausgeführt. Dort werden dann Erweiterungskarten direkt oder interne Laufwerke über Kabel angeschlossen. Einige andere Schnittstellen werden nicht herausgeführt, sondern befinden sich auf der Hauptplatine zwischen den einzelnen Controllern.

Interne Schnittstellen für Erweiterungskarten

AGP (Accelerated Graphics Port)



Der braun gefärbte AGP- Steckplatz wurde als **Schnittstelle für Grafikkarten** entwickelt und ist **doppelt so schnell, wie der PCI-Bus**. Mit der letzten Spezifikation „AGB 3.0“ kann eine **maximale Taktrate von 2133 MHz** erreicht werden. Aufgrund der **Nachfrage nach höher getakteten Grafikkarten** - vor allem für die Spieleindustrie - wurde dieser Slot **von der PCI Express**

Schnittstelle abgelöst.

PCI - Peripheral Component Interconnect

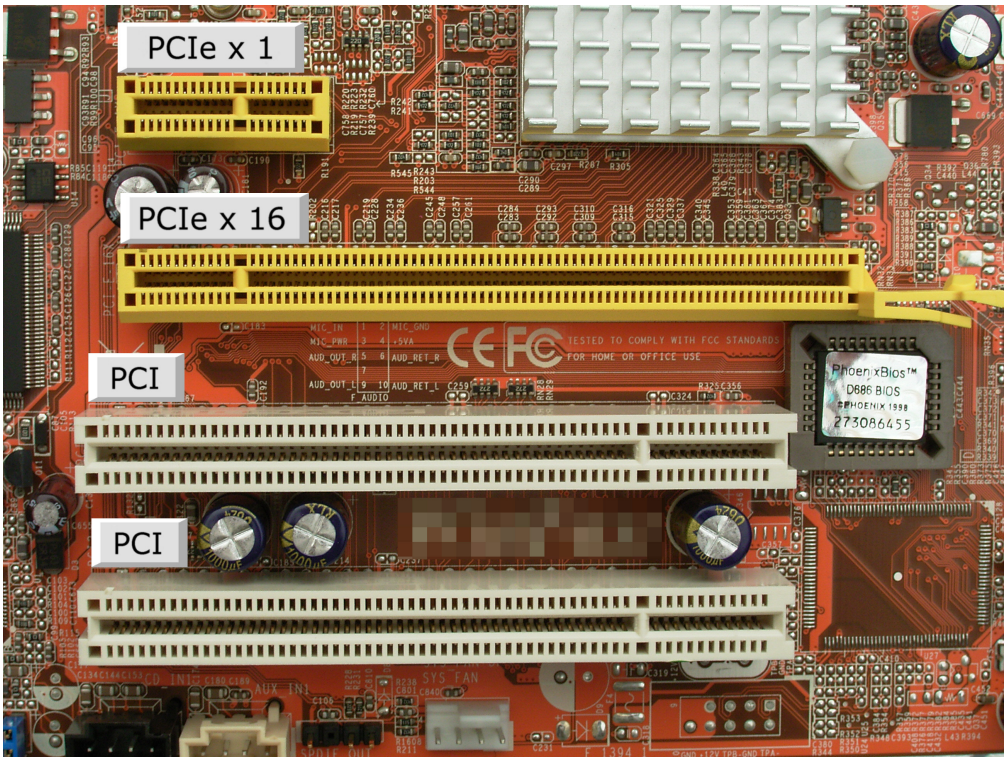
Der **PCI-Bus ist Industriestandard** und war viele Jahre fester Bestandteil von IBM-kompatiblen PCs, Macs von Apple und Alpha-Workstations von Digital. Das von Intel im Jahr 1993 entwickelte Bus-System ist bis ins Detail normiert, so dass andere Computerhersteller den PCI-Bus nachbauen können.

Der PCI-Bus kann **32 oder 64 Signalleitungen** haben. Der PCI-Bus mit 32 Bit teilen sich Adress- und Datenbus die Signalleitungen. Die Signalleitungen werden im Multiplex-Betrieb genutzt. Das bedeutet, mit einem Takt wird zuerst die Adresse und in einem zweiten Takt das Datenwort gesendet. So werden 32 Signalleitungen eingespart. In Servern kommt der PCI-Bus mit 64 Bit zum Einsatz. dort stehen jeweils 32 Adress- und Datenleitungen zur Verfügung.

Sie sind an ihrer weißen Färbung zu erkennen. Eine Erweiterung des Computers mittels Soundkarten, Netzwerkkarten, TV-Karten o.ä. ist über diese Slots möglich.

Spezifikation	Bus-Breite	Taktfrequenz	Datentransferrate	Signalspannung	Geräte pro Bus	Einführung
PCI 2.0	32 Bit	8 bis 33 MHz	0,12 GByte/s	5 V	6	1993
PCI 2.3	32 Bit	33 MHz	0,133 GByte/s	5V	6	2002
PCI 2.3	64 Bit	33 MHz	0,266 GByte/s	5V	6	2002
PCI 2.3	32 Bit	66 MHz	0,266 GByte/s	3,3V	3	2002
PCI 2.3	64 Bit	66 MHz	0,533 GByte/s	3,3V	3	2002
PCI-X 1.0	64 Bit	66 MHz	0,533 GByte/s	3,3V	4	1999
PCI-X 1.0	64 Bit	100 MHz	0,800 GByte/s	3,3V	2	1999
PCI-X 1.0	64 Bit	133 MHz	1,066 GByte/s	3,3V	1	1999
PCI-X 266 (2.0)	64 Bit	133 DDR	2,133 GByte/s	1,5V	1	2003
PCI-X 533 (2.0)	64 Bit	133 QDR	4,266 GByte/s	1,5V	1	2003

PCI-E(xpress)



PCI Express (PCIe) ist eine schnelle interne Schnittstelle für Erweiterungskarten in Computer-Systemen. Mit der **Einführung von PCIe im Jahr 2004 wurde dem AGP als Grafikkarten-Schnittstelle ein Ende gesetzt** und auch der **PCI als internes Computer-Bussystem abgelöst**.

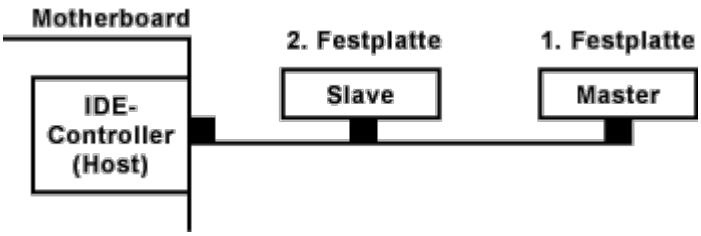
Die **Übertragungsgeschwindigkeit** bei PCIe **orientiert sich an der Version und der Anzahl der Links bzw. Lanes**. Je höher die Version und je mehr Links, desto höher die Bandbreite und desto höher ist die Übertragungsgeschwindigkeit. Die Bandbreite gibt dabei an, wie viel Kapazität für die Datenübertragung theoretisch bzw. maximal zur Verfügung steht. Die tatsächliche Datenrate liegt jedoch darunter.

PCIe	Bandbreite pro Link		PCIe x1	PCIe x4	PCIe x8	PCIe x16	Kodierung/Balast	Verfügbar seit
1.0	2,5 GT/s	2,5 GBit/s	250 MByte/s	1 GByte/s	2 GByte/s	4 GByte/s	8b10b / 20%	2004
2.0	5 GT/s	5 GBit/s	500 MByte/s	2 GByte/s	4 GByte/s	8 GByte/s	8b10b / 20%	2008
3.0	8 GT/s	10 GBit/s	0,9846 GByte/s	3,938 GByte/s	7,877 GByte/s	15,754 GByte/s	128b/130b / <2%	2011
4.0	16 GT/s	20 GBit/s	1,969 GByte/s	7,877 GByte/s	15,754 GByte/s	31,508 GByte/s	128b/130b / <2%	2017
5.0	32 GT/s		3,9 GByte/s	15,8 GByte/s	31,5 GByte/s	63 GByte/s	128b/130b / <2%	?

Interne Schnittstellen für Massenspeicher

IDE - Integrated Drive Electronics

IDE bedeutet offiziell **Integrated Device Elektronics**. Bei IDE handelt sich um eine **alte Festplatten-Schnittstelle**, bei der die Steuerungselektronik bzw. der Controller in das Festplattengehäuse integriert ist. Nachfolger von IDE ist Enhanced IDE (EIDE), was auch als ATA (PATA) bezeichnet wird, eine deutlich höhere Übertragungsgeschwindigkeit hat und den Anschluss von CD-ROM- und DVD-Laufwerken ermöglicht. IDE-Controller war Anfangs eine ISA-Steckkarte für den AT-Bus. Er war jedoch eher ein sogenannter Host-Adapter, der nur die notwendigen Systembussignale mit Pufferung zur Festplattenelektronik weiterleitete. Das **40-adrige IDE-Flachbandkabel** stellte praktisch die **Verlängerung des Systembusses** dar. Später wurde der IDE-Controller fest auf dem Motherboard integriert.

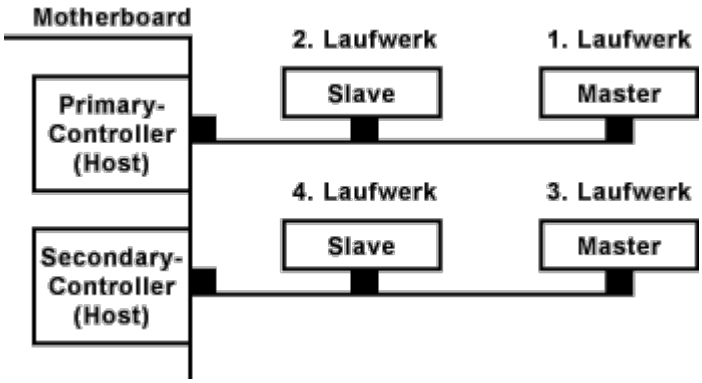


Das **IDE-Flachbandkabel hat drei Steckerleisten**. Die eine ist für den **Hostanschluss** auf dem IDE-Controller. Die anderen beiden Steckerleisten sind für das **Master- und Slave-Endgerät**. **Pro IDE-Controller** lassen sich **zwei Festplatten** betreiben. Weil die eigentliche Steuerung auf den Festplatten sitzt, muss die eine Festplatte, am besten die schnellste, als Master und die andere als Slave konfiguriert werden. Dazu müssen Jumper oder Dip-Schalter gesetzt werden. Die Master-Slave-Konfiguration sorgt dafür, dass beim Systemstart der Master die höhere Priorität hat. Bevor er Funktionsbereitschaft an das Bios meldet, wartet er auf die Bestätigung der Slave-Festplatte. Beide **Festplatten (Laufwerke) arbeiten unabhängig voneinander**. Sie belegen aber die gleichen Adressen im Computersystem.

P-ATA / Ultra-ATA / EIDE

EIDE bzw. ATA sind **alte Schnittstellen für den Anschluss von Festplatten und Wechselspeicher-Laufwerken**, wie CD-ROM, DVD oder Streamer in einem Computer. **EIDE wurde von SATA abgelöst**.

Die **EIDE-Schnittstelle (Enhandced Integrated Drive Electronics)** ist eine **Weiterentwicklung des IDE-Standards**. Die EIDE-Schnittstelle bezeichnet man auch als **ATA-Schnittstelle**. ATA steht für **Advanced Technology (AT) Attachment**. Die Bezeichnung EIDE wird nur noch selten verwendet. Mit dem Aufkommen von **Serial ATA (S-ATA)** wurde die Bezeichnung **P-ATA** immer gebräuchlicher. Wobei das **P für parallel** steht.

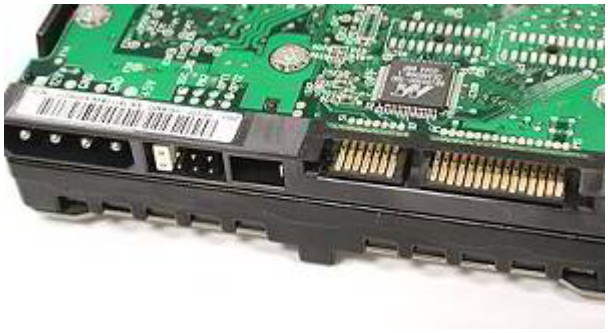


S-ATA / Serial-ATA

Serial-ATA, kurz SATA oder S-ATA, ist eine **Schnittstelle zum Anschluss von Massenspeichern, wie Festplatten und Wechselspeicher-Laufwerken**. Schnittstellen für Massenspeicher waren **ursprünglich immer Bussysteme mit parallel geführten Signalleitungen** in Leiterbahnen und Anschlusskabeln. Mit **zunehmender Übertragungsgeschwindigkeit** ergaben sich **technische Schwierigkeiten**, die für die Übertragungsrate eine obere Grenze setzten. So blieb auch die ATA (EIDE)-Schnittstelle nicht davon verschont, dass sie auf eine **seriellen Betriebsart umgestellt** wurde.

Im **Jahr 2000** setzten sich mehrere Firmen aus dem IT-Sektor zusammen, um eine **Spezifikation über Serial-ATA (Serielles ATA) zu erstellen**. Im Jahr 2001 wurde die erste Version von Serial-ATA vorgestellt. Anfang 2003 waren bereits die ersten Controller und Festplatten erhältlich. Bis zur vollständigen Marktdurchdringung hat es noch bis zum Jahr 2004 gedauert. Mit **150 MByte/s hat**

SATA direkt an die parallele EIDE-Schnittstelle (P-ATA) mit 133 MByte/s angeknüpft. Die **SATA-Schnittstelle unterstützt 1,5 GBit/s** bei einer **Nettodatenrate von ca. 150 MByte/s**. **Festplatten mit 10.000 Umdrehungen** in der Minute (U/m) liefern rund **75 MByte/s** an Daten. Mit **SATA 6G** erreichen herkömmliche Festplatten **fast 500 MByte/s (Schreibgeschwindigkeit)**.



Zwar wurde SATA mit SATA-II und SATA 6G noch zwei mal auf maximal 600 MByte/s beschleunigt. Für **Datenspeicher mit Flash-Memory (SSD, Solid State Drive)** ist das aber **nicht schnell genug**. Allerdings gibt es SSDs, die Daten mit weit über 1.000 MByte/s lesen und schreiben können. Dafür bedarf es auch einer Schnittstelle, die diese Datenmenge bewältigen kann. SATA kann das nicht leisten. Deshalb wird **SATA durch SATA Express (SATAe) oder PCI Express (PCIe) als Massenspeicher-Schnittstelle abgelöst**.

Schnittstelle	Bezeichnung	Transferrate		Reichweite	Geräteanzahl	Einführung
Serial-ATA	SATA	1,5 GBit/s	150 MByte/s	1 m	4	2003
Serial-ATA-2	SATA-II	3 GBit/s	300 MByte/s	1 m	16	2005
Serial-ATA-3	SATA 6G / SATA-600	6 GBit/s	600 MByte/s	1 m	16	2007
Serial Attached SCSI (SAS)	SAS	3 GBit/s	300 MByte/s	1 m	16.384	2004
Serial Attached SCSI 2 (SAS 2)	SAS 6G	6 GBit/s	600 MByte/s	1 m	16.384	2007
Serial Attached SCSI 3 (SAS 3)	SAS 12G	12 GBit/s	1.200 MByte/s	1 m	16.384	2010

SATAe / Serial-ATA Express

SATA Express ist der **offizielle Nachfolger von SATA 6G** und nutzt einen zu SATA 6G abwärtskompatiblen Steckverbinder. Dieser Steckverbinder bündelt **zwei SATA-Ports** die **wahlweise für die SATA-6G- oder PCIe-Übertragung** genutzt werden können. Denn **SATA Express beherrscht sowohl SATA-6G, als auch PCIe**. Die SATAe-SSD hat dabei die Form einer herkömmlichen Festplatte. Mit PCIe 2.0 bringt SATA Express aber nur 1 GByte/s und mit PCIe 3.0 wären es auch „nur“ 2 GByte/s. Das ist für viele Anwendungen viel zu wenig. **SATA Express** scheint **schon vor seiner breiten Nutzung veraltet**.

Es ist davon auszugehen, dass es sich bei SATAe nur um Übergangslösungen handelt. Mittelfristig wird sich für **SSDs der PCI Express als Standardschnittstelle** durchsetzen.

Externe Schnittstellen

Externe Schnittstellen werden aus dem Computer-Gehäuse herausgeführt. Sie **verbinden Systeme oder Peripherie-Geräte** mit dem Computer. Die Verbindung wird mit einer **Kombination aus Stecker und Buchse** realisiert.

Serielle Schnittstelle

An jeder seriellen Schnittstelle kann nur ein weiteres Gerät angeschlossen werden. Die Bits werden nacheinander (seriell) über eine einzige Leitung übertragen, deshalb entsteht hier nur ein sehr geringer Kostenaufwand, aber auch eine niedrige Übertragungsrate. Klassische Endgeräte einer seriellen Schnittstelle sind Maus und Modem aber auch zahlreiche technische Einrichtungen, wie zB. Messgeräte, haben diese Schnittstelle, um mit Computern verbunden werden zu können.

Parallele Schnittstelle

Diese Schnittstelle wurde ursprünglich vom Drucker-Hersteller Centronics (daher oft Centronics-Schnittstelle genannt) entwickelt und war eine der ersten Schnittstellen für Drucker. Hier werden bereits 8 Bit gleichzeitig über jeweils eine eigene Leitung übertragen was im Vergleich zur seriellen Schnittstelle zu einer höheren Übertragungsrate führt. Je nach der Qualität des Kabels kann problemlos eine Leitungslänge von 5m für eine fehlerlose Datenübertragung genutzt werden.

USB (Universal Serial Bus)

Der USB ist eine universelle, externe Schnittstelle für alle Peripheriegeräte, die an einem Computer angeschlossen werden. Egal ob Tastatur, Maus, Modem, Drucker, Mikrofon, Lautsprecher, Kamera oder Scanner. Mit dem USB sind die Anwender unabhängig von der Anzahl der verfügbaren Schnittstellen und Steckplätze für Erweiterungskarten. Der USB lässt sich durch Steckkarten oder USB-Hubs fast beliebig erweitern.
Die Identifikation der Geräte wird vom USB-Hostadapter im Computer durchgeführt, der auch das Laden der Treiber und die Grundkonfiguration vornimmt. Zusätzlich verbessert sich durch Hot-Plugging, das Hinzufügen und Entfernen von Peripherie-Geräten im laufenden Betrieb, die Bedienerfreundlichkeit.

Der USB erfüllt folgende Anforderungen:

- eine einheitliche Schnittstelle für alle Peripherie-Geräte
- eine mechanisch stabile und einfache Steckverbindung
- kleine platzsparende Stecker und Buchsen

Zusätzlich weisen alle USB-Spezifikationen folgende Eigenschaften auf:

- billig
- abwärtskompatibel

USB-Version	USB 1.0/1.1		USB 2.0	USB 3.0	USB 3.1	USB 3.2
	Low Speed	Full Speed	High Speed	Super-Speed	Super-Speed-Plus	
Symbolrate	1,875 MBi/ts	15 MBit/s	600 MBit/s	5 GBit/s	10 GBit/s	20 GBit/s
Datenrate (brutto)	1,5 MBit/s	12 MBit/s	480 MBit/s	4 GBit/s	-	-
Datenrate (theoretisch)	188 kByte/s	1,5 MByte/s	60 MByte/s	600 MByte/s	1.200 MByte/s	
Datenrate (netto)	ca. 150 kByte/s	ca. 1 MByte/s	ca. 36 bis 44 MByte/s	ca. 480 MByte/s	ca. 800 MByte/s	ca. 1.600 MByte/s
Interface	UHCI/OHCI	UHCI/OHCI	EHCI	xHCI	-	-
Leitungslänge	5 m	5m	5m	3 m	1 m	1 m
Anwendungen	Maus Tastatur	Audio	Video, Speichermedien			

FireWire

Firewire wurde ursprünglich von Apple entwickelt und wurde dann aus urheberrechtlichen Gründen von Sony in i.LINK umbenannt. Eine weitere Bezeichnung für diese Schnittstelle ist der vom Standardisierungsgremium IEEE festgelegte Name IEEE 1394. Die besondere Fähigkeit von FireWire ist der schnelle Datenaustausch zwischen externem Gerät und Computer, mit einer Übertragungsrate von bis zu 800 MBit/s. Durch diese hohe Geschwindigkeit eignet sich diese Schnittstelle besonders gut zur Übertragung von Bildern und Videos und zum Anschluss von Videokameras, Festplatten und DVD-Brennern. Im Gegensatz zu USB 2.0 wird die theoretische Datenübertragungsrate nahezu erreicht.

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:4:4_04

Last update: **2018/10/23 15:32**

Präsentationstechniken

Unter **Präsentationstechnik** versteht man die Grundsätze, die eine Präsentation erfolgreich machen. Die Präsentation ist eine zweckbestimmte und empfangsorientierte Informationsbeschreibung, welche versucht, den Kommunikationsfluss zu verbessern und Expertenwissen anderen zugänglich zu machen.

Vorbereitung einer Präsentation

- Wem genau stelle ich meine Arbeit vor?
- Welche Einblicke soll das Publikum in meine Arbeit erhalten?
- Welche Kernbotschaften sollen sich schließlich im Gedächtnis festsetzen?
- Wodurch kann sich meine Präsentation von den anderen abgrenzen
- Welche Fragen könnte das Publikum stellen?

Zusätzlich ist zu beachten, dass pro Minute der Präsentation etwa 5 bis 30 Minuten an Vorbereitung einzuplanen sind. Je kürzer die Präsentation, desto kritischer ist die Vorbereitung.

Gliederung

Einstiegsphase

- kurze Begrüßung und Selbstvorstellung
- knappe Erläuterung des persönlichen Zugangs zum Thema
- Vorstellung der Fragestellung und der Präsentationsziele

Je interessanter und überraschender die Präsentation beginnt, desto einfacher gelingt die Kontaktaufnahme mit der Prüfungskommission und desto angenehmer ist die Gesprächsatmosphäre. Die Möglichkeiten sind vielfältig: Gemeinsamkeit mit Zuhörern herstellen, sachlich und ernst beginnen, persönliche Erfahrungen einbringen, Vergleiche herstellen, Anschauungsmaterial einsetzen, historische Rückschau halten, rhetorische Frage stellen, auf aktuelles Geschehen hinweisen, ein Zitat voranstellen, …

Hauptteil

- 3 - 5 Module
- Vorstellung der Vorgangsweise/Methodik
- Vermittlung der Kernbotschaften

Ausstiegsphase

- Zusammenfassung der Kerninhalte
- Ausblick

- Dank an das Publikum und Überleitung zur Diskussion

Der Schluss bleibt am längsten in Erinnerung und ist daher mindestens ebenso wichtig wie die Einstiegsphase.

Dimensionen der Verständlichkeit

- Einfachheit – Vermeide Kompliziertheit!
- Gliederung, Ordnung – Vermeide Unübersichtlichkeit!
- Kürze, Prägnanz – Vermeide Weitschweifigkeit!
- Zusätzliche Stimulanz – Vermeide einkehrende Langeweile und wecke Emotionen!

„Nichts ist schwerer, als bedeutende Gedanken so auszudrücken, dass jeder sie verstehen muss.“ (Schopenhauer)

- kurze Sätze verwenden
- Einschiebungen vermeiden
- Fremdwörter sparsam dosieren
- Füllwörter ausmerzen
- Vergleiche/ Beispiele bringen
- Standardsprache verwenden

PowerPointPräsentation – praktische Tipps

Zielsetzung

- Sichtbarmachen einer Gliederung
- Schaffung von Orientierung
- Visualisierung (komplexer) Inhalte durch Grafiken, …

Layout

- angemessenes Foliendesign
- dezente Übergänge
- sparsame Effekte
- einheitliches Muster (in Bezug auf Aufzählungszeichen, Einrückung, Farbe, Schriftgröße, …)
- ansprechende Gestaltung der Titelfolie (evtl. grafischer Eyecatcher)
- bewusster Verzicht auf „Danke für die Aufmerksamkeit“-Folien

Reduktion und Übersichtlichkeit

- „Weniger ist mehr“
- überschaubare Anzahl von Folien
- wenig Text auf den einzelnen Folien (Stichwörter!)
- „Ein Bild sagt mehr als tausend Worte“

Korrektheit

- inhaltlich einwandfrei
- stilistisch ansprechend (Ausdruck)
- sprachlich richtig (Rechtschreibung, Grammatik)

Wirkung durch Körpersprache

- authentisch
- Blickkontakt mit dem Publikum (→ Präsenz)
- Mimik: Ein Lächeln wirkt Wunder.
- Gestik: Was du sagst, „liegt auf der Hand“.
- Körperhaltung: Der Körper ist der Übersetzer der Seele ins Sichtbare.
- äußere Erscheinung: You never get a second chance to make a first impression!

Stimme macht Stimmung

Haltung

Eine gute Grundhaltung sorgt für eine funktionierende Atmung und somit für eine natürliche, klangvolle Stimme.

Artikulation

Um Nuscheln vorzubeugen und den Mundraum zur Verstärkung des Stimmklangs zu nutzen, bleibt der Unterkiefer locker und der Mund beim Sprechen geöffnet.

Stimmdynamik

Dynamisch ist eine Sprechweise, die sehr lebendig ist und auch Sprechpausen beinhaltet. Wörter werden sinnvoll betont (laut/leise), die Sätze werden melodiös gesprochen (hohe/tiefe Stimme), Satzteile werden manchmal kurz, manchmal gestreckt gesprochen (dynamisches Sprechtempo).

Notizen

- PowerPointPräsentation, Flipchart, … als Stichwortgeber
- allenfalls Verwendung von Karteikarten (DIN A6, Karton)
- ein Leitgedanke und drei bis fünf weiterführende Stichworte pro Kärtchen
- Beschreibung der Kärtchen auf der Vorderseite
- Durchnummerierung der Kärtchen
- freie Rede anhand der Leitgedanken
- Vergewisserung anhand der Stichworte

- Tipps für eine gute Präsentation

Powerpoint

- Übungen zu Powerpoint

Prezi

- http://prezi.com/
- Prezi-ähnliche Systeme

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:5

Last update: **2018/10/15 08:22**

Bildbearbeitung

From:
http://elearn.bgamstetten.ac.at/wiki/ - **Wiki**

Permanent link:
http://elearn.bgamstetten.ac.at/wiki/doku.php?id=inf:inf5bi_201819:6

Last update: **2018/10/23 15:50**