CT/IT Abi

Zusammenfassung

1. Hardware

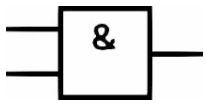
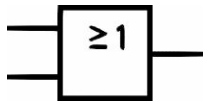
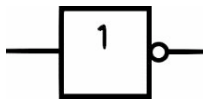
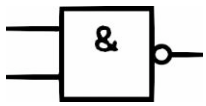
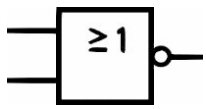
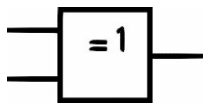
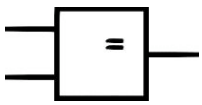
Inhalt:

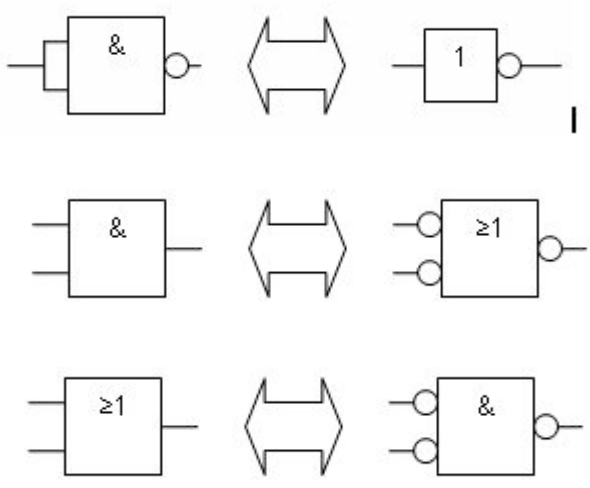
- Digitaltechnik
  - Logische Verknüpfungsschaltungen
  - Darstellung digitaler Schaltfunktionen
  - Multiplexer
  - Speicher
  - [CT] RS-FF
  - [CT] D-FF
  - [CT] [Zweispeicher-Flipflop (Master-Slave)
  - [CT] Master-Slave D-FF (flankengesteuert)
  - [CT] JK-FF
  - [CT] T-FF
  - Schaltnetz / Schaltwerk (Automat)
  - Beschreibung u. Darstellung synchroner Schaltwerke
  - Bsp. Moore-Automat
  - Bsp. Mealy-Automat
  - Asynchronzähler
  - Synchronzähler

- [CT] Microcontroller
  - Einfaches Assembler-Programm
  - Einfaches C-Programm
  - Interrupt
  - Assembler-Interrupt-Programm
  - C-Interrupt-Programm
  - Timer / Counter Funktionen
  - Assembler Timer Interrupt
  - C Timer Interrupt
  - C-Zeitschleife
  - Assembler Zeitschleife
  - Wahrheitstabelle mit Assembler
  - BCD-Code
  - Chipselect

1.1. Digitaltechnik

Logische Verknüpfungsschaltungen:

Bezeichnung	Symbol	Wahrheitstabelle	Gleichung
AND		a b x 0 0 0 0 1 0 1 0 0 1 1 1	$x = b \wedge a$
OR		a b x 0 0 0 0 1 1 1 0 1 1 1 1	$x = b\bar{a} \vee \bar{b}a \vee ba$
NOT		a x 0 1 1 0	$x = \bar{a}$
NAND		a b x 0 0 1 0 1 1 1 0 1 1 1 0	$x = \bar{b}\bar{a} \vee b\bar{a} \vee \bar{b}a$
NOR		a b x 0 0 1 0 1 0 1 0 0 1 1 0	$x = \bar{b}\bar{a}$
EXOR Antivalenz		a b x 0 0 0 0 1 1 1 0 1 1 1 0	$x = b\bar{a} \vee \bar{b}a$
Äquivalenz		a b x 0 0 1 0 1 0 1 0 0 1 1 1	$x = \bar{b}\bar{a} \vee ba$



Darstellung digitaler Schaltfunktionen:

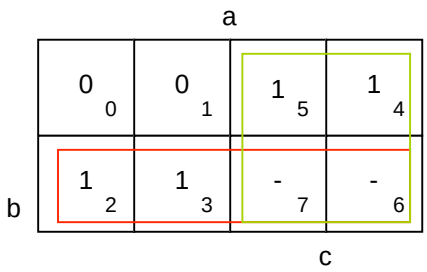
Wahrheitstabelle:

	Inputs			Outputs	
	c	b	a	y	
0	0	0	0	0	
1	0	0	1	0	
2	0	1	0	1	
3	0	1	1	1	
4	1	0	0	1	
5	1	0	1	1	
6	1	1	0	-	<- don't care
7	1	1	1	-	<- don't care

Disjunktive Normalform (DNF) in Funktionsdarstellung:

$y = \bar{c}b\bar{a} \vee \bar{c}ba \vee c\bar{b}\bar{a} \vee cb\bar{a}$

Grafische Darstellung und Minimalisierung mittels KV-Diagramm:

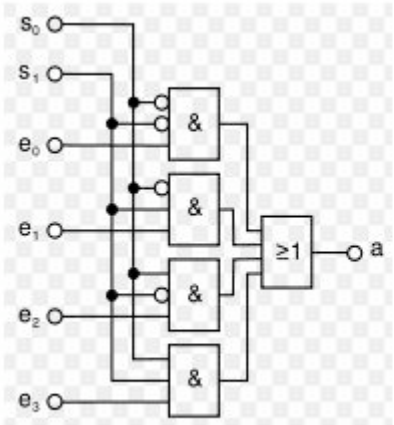


⇨ y = b ∨ c

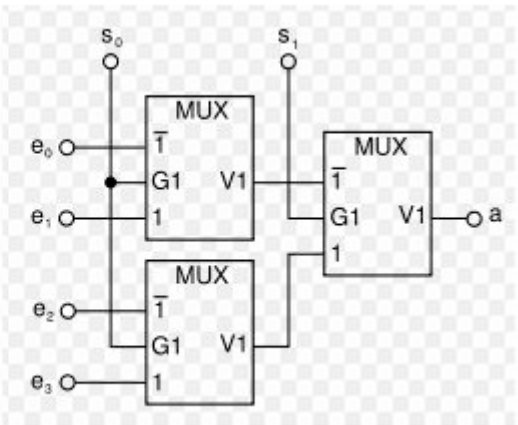
Multiplexer:

Ein Multiplexer (kurz: MUX) ist ein Selektionsschaltnetz in der Digitaltechnik, mit dem aus einer Anzahl von Eingangssignalen eines ausgewählt werden kann. Das Gegenstück zum Multiplexer ist der Demultiplexer, mit dem die zusammengefassten Datenkanäle wieder aufgetrennt werden.

Aufbau eines 2-MUX:

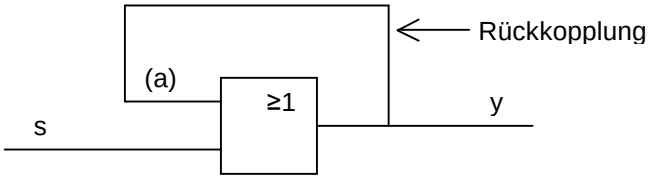


Aufbau eines 2-MUX aus drei 1-MUX:



Speicher:

(a)	s	y
0	0	0
0	1	1
1	0	1
1	1	1

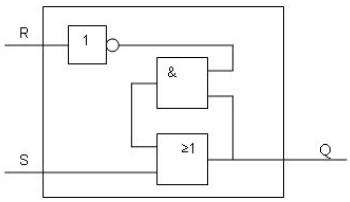


Wenn y = 1 dann bleibt y auf 1 => Kann nicht gelöscht werden.

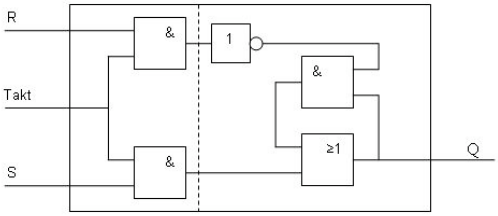
[CT] RS-FF:

Das RS-FF ist ein Speicher mit einem Reset.

RS-FF ohne Takt:

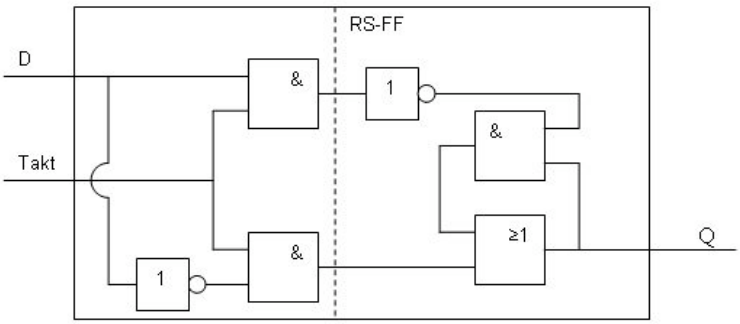


RS-FF mit Takt:

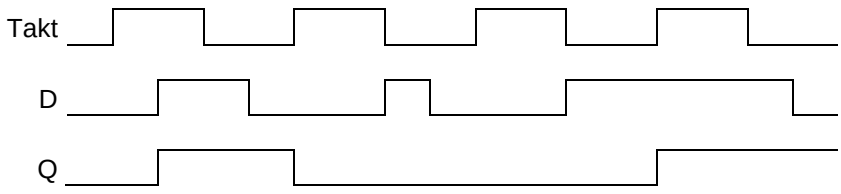


[CT] D-FF (pegelgesteuert):

Ein RS-Flipflop mit nur einem Eingang, der gleichzeitig Setz-Eingang und negierter Rücksetz-Eingang ist, wird *D-Flipflop* (D für *Delay*) genannt oder Latch. Bei jedem Takt (C für *Clock*) nimmt das Flipflop den Zustand an, der am Eingang anliegt. Der Vorteil des D-Flipflops ist, dass der illegale Zustand R = S = 1 vermieden wird.



Beispiel Impulsdiagramm:



[CT] Zweispeicher-Flipflop (Master-Slave):

Eine Kippstufe muss häufig eine neue Information aufnehmen und „gleichzeitig“ die alte gespeicherte Information weitergeben können.

⇨ Es ist eine Zwischenspeicherung in einem weiteren FF notwendig.

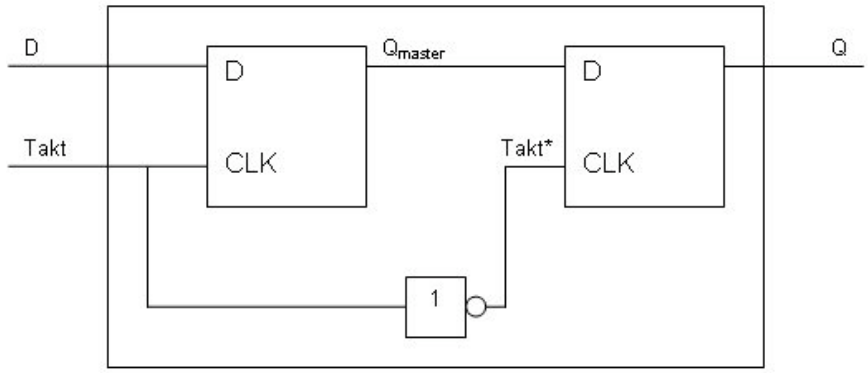
Master-Slave-Prinzip:

Die Kippstufe besteht aus

- Eingangslogik
- Zwischenspeicher (=Master-FF)
- Ausgangsspeicher (= Slave-FF)
- Steuerlogik

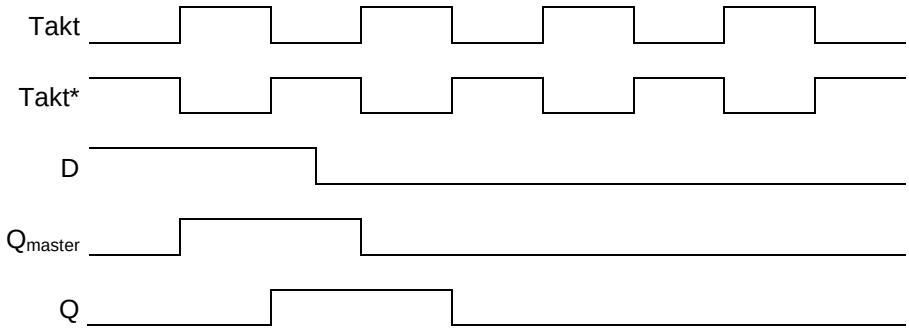
- Master-FF und Slave-FF können aus verschiedenen Grund-FF-Schaltungen aufgebaut werden. Es gilt die Wahrheitstabelle des Grundflipflops.
- Master-Slave-FF’s sind als integrierte Schaltungen erhältlich.

[CT] Master-Slave D-FF (flankengesteuert):



Durch dieses MS-D-FF erreicht man negative Flankensteuerung, wie im folgenden Beispiel deutlich wird.

Beispiel Impulsdiagramm:

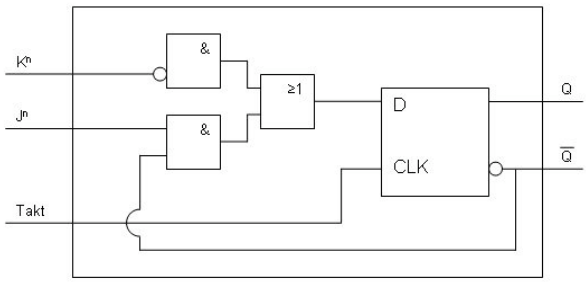


[CT] JK-FF:

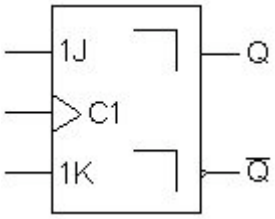
J^n	K^n	Q^n	Q^{n+1}	
0	0	0	0	Verhält sich wie ein RS-FF
0	0	1	1	
0	1	0	0	
0	1	1	0	
1	0	0	1	
1	0	1	1	
1	1	0	1	Toggle
1	1	1	0	

JK-FF sind Universal-FF mit stets definierten Ausgangszuständen. Pegelsteuerung ist mit JK-FFs kaum möglich (FF kippt dauernd, instabil).

JK-FF aus einem D-FF:



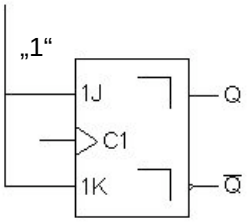
Schaltzeichen eines JK-MS-FFs:



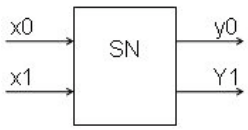
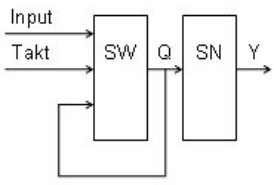
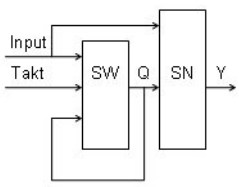
[CT] T-FF:

Wenn die beiden Eingänge des JK-FF mit logisch "1" verbunden werden, ergibt sich ein *T-Flipflop* (T für *en.: toggle* - Hin- und herschalten).

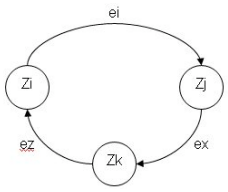
Aufbau eines T-FF aus einem JK-FF:



Schaltnetz / Schaltwerk (Automat):

Schaltnetz	Schaltwerk (Automat)		
	Synchron (mit Takt)		Asynchron (ohne Takt)
	Moore	Mealy	
			Bsp.: RS-FF’s ohne Takt Keine FF’s

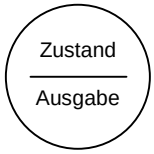
Beschreibung u. Darstellung synchroner Schaltwerke:

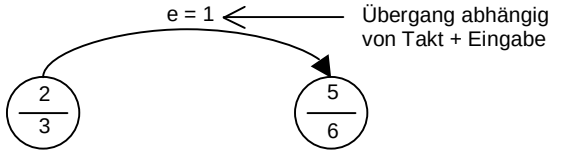
Automatengraf	Automatentabelle Zustandstabelle Ablauftabelle	Ansteuertabelle
	Momentane Zustände | Folgezustände	Momentane Zustände | Ansteuerfunktion

1. Wahrheitstabelle zeitunabhängig:
   Automatentabelle
   Ablauftabelle
2. Ansteuertabelle
3. Automatengraf (Struktugramm, PAP)
4. Gleichungen (Ansteuerung, Ausgabe)

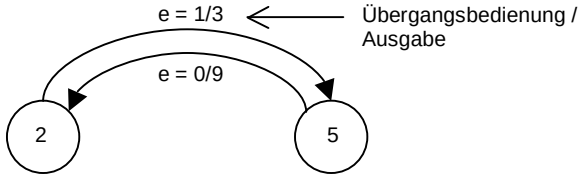
Automatengraf Aufbau:

Moor:

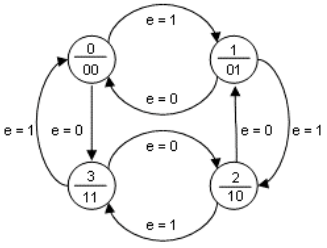




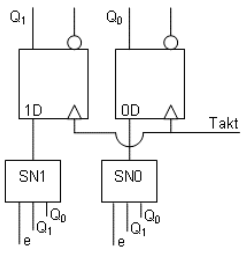
Mealy:



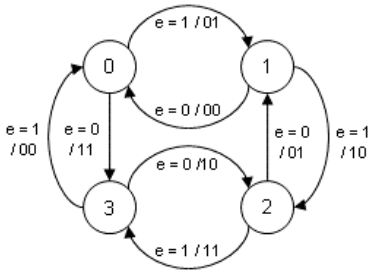
Bsp. Moore-Automat:



e	Q_1	Q_0	Q_1^{n+1}	Q_0^{n+1}	$1D^n$	$0D^n$
0	0	0	1	1	1	1
0	0	1	0	0	0	0
0	1	0	0	1	0	1
0	1	1	1	0	1	0
1	0	0	0	1	0	1
1	0	1	1	0	1	0
1	1	0	1	1	1	1
1	1	1	0	0	0	0

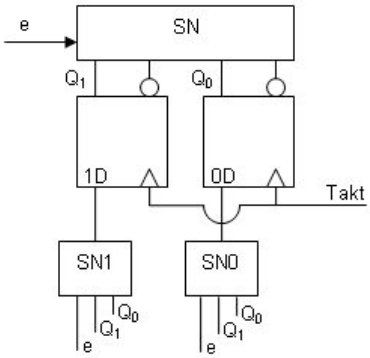


Bsp. Mealy-Automat:



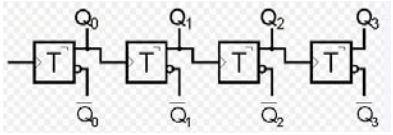
e	Q_1	Q_0	Q_1^{n+1}	Q_0^{n+1}
0	0	0	1	1
0	0	1	0	0
0	1	0	0	1
0	1	1	1	0
1	0	0	0	0
1	0	1	0	1
1	1	0	1	0
1	1	1	1	1

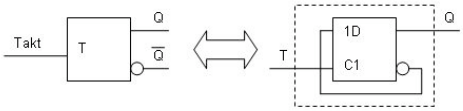
Diese Tabelle zeigt das Ausgabeschaltnetz

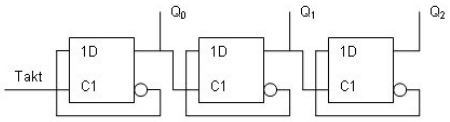


Asynchronzähler:

Ein Asynchronzähler ist ein Bauteil aus der Digitaltechnik, das eine Folge von natürlichen Zahlen erzeugt. Die Darstellung der Zahlen erfolgt im Dualsystem. Man spricht daher auch von *n-bit-Asynchronzählern*.







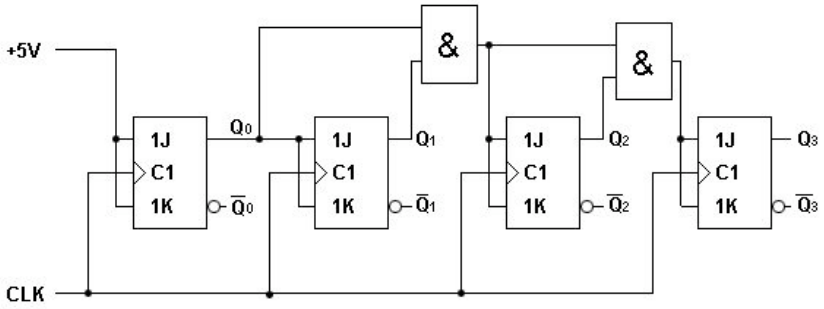
Wahrheitstabelle für einen 3-Bit-Asynchronzähler aus D-FF

	Q_2	Q_1	Q_0
0	1	1	1
1	1	1	0
2	1	0	1
3	1	0	0
4	0	1	1
5	0	1	0
6	0	0	1
7	0	0	0

Im Unterschied zum Synchronzähler wird beim Asynchronzähler nicht jedes Flipflop mit einem externen Takt C versorgt. Der Eingangstakt, der am ersten Flipflop anliegt, wird vielmehr durchgereicht. Durch die Schaltzeiten der internen Bauelemente kommt es daher beim Asynchronzähler zu Verzögerungen, die sich Bauteil für Bauteil aufsummieren. Bei einem n-Bit-Asynchronzähler mit großem n ergibt sich also eine deutliche Verschiebung zwischen dem Takt am ersten Flipflop und dem am letzten.

Synchronzähler:

Ein Synchronzähler ist ein Bauteil aus der Digitaltechnik, das eine Folge von natürlichen Zahlen erzeugt. Die Darstellung der Zahlen erfolgt im Dualsystem. Die Menge der darstellbaren Zahlen und ihre Reihenfolge ist bauteilabhängig. Man spricht daher auch von *n-bit-Synchronzählern*.



1.2. [CT] Microcontroller

Einfaches Assembler-Programm:

Folgendes Assembler-Programm gibt alles was an Port1 anliegt an Port2 wieder aus.

```
        include reg_51.pdf          ;Includedatei reg_51.pdf einbinden
        in equ P1                   ;“in“ als Alias für Port1

        code at 0                   ;Mit folgenden Code an Speicherstelle 0 beginnen

start:  mov a, in                   ;Was an Port1 anliegt in Akku schreiben
        mov P2, a                   ;Akku an Port2 ausgeben
        sjmp start                  ;Zu start springen
        end                         ;End muss am Ende jedes Assemblerprogramms stehen
```

Einfaches C-Programm:

Folgendes C-Programm gibt alles was an Port1 anliegt an Port2 wieder aus.

```
#include <stdio.h>                  // Header stdio.h einbinden
#include <at89c51ed2.h>             // Header at89c51ed2.h einbinden

sfr at P1 in;                       //“in“ als Alias für Port1

void main(void)                     //Beginn des Hauptprogramms
{
  while(1)                          //Endlosschleife Anfang
  {
    P2 = in;                        //Was an Port1 anliegt an Port2 ausgeben
  }                                 //Endlosschleife Ende
}
```

Interrupt:

In der Informatik versteht man unter Interrupt die kurzfristige Unterbrechung eines Programms durch eine von der CPU abzuarbeitende Befehlssequenz, die Interrupt Service Routine (=ISR) Anschließend wird die Ausführung des Programms an der Unterbrechungsstelle fortgesetzt.

Assembler-Interrupt-Programm:

```
        include reg_51.pdf
        code at 0
        ljmp init
        org 0003h           ;Folgenden Code an Speicherstelle 0003h da dort nach der ISR
                            ;gesucht wird
        ljmp isr            ;Springe zu isr
        org 0100d
init:   setb EA             ;Interrupts scharf machen(siehe Formelsammlung)
        setb EX0            ;Externer Interrupt0 scharf machen (siehe Formelsammlung)
        setb IT0            ;Interrupt auslösen bei abfallender Flanke
                            ;(siehe Formelsammlung)
        mov r0,#00h
go:     mov P0, r0          ;Inhalt von R0 an Port0 ausgeben
        sjmp go
isr:    inc r0              ;R0 um eins erhhöhen
        reti                ;Zurück ins Hauptprogramm springen
        end
```

C-Interrupt-Programm:

```
#include <stdio.h>                  // Header stdio.h einbinden
#include <at89c51ed2.h>             // Header at89c51ed2.h einbinden

void extInt0(void);                 //Prototyp für ISR

void main(void)
{
  unsigned char x;
  EA = 1;                           //Interrupts scharf machen (siehe Formelsammlung)
  EX0 = 1;                          //Externer Interrupt0 scharf machen (siehe Fosa)
  IT0 = 1;                          //Interrupt auslösen bei abfallender Flanke (siehe Fosa)
  x = 0;

  while(1)
  {
    P0=x;
  }
}

void extInt0(void) interrupt 0      //ISR
{
  x = x + 1;
}
```

Timer / Counter Funktionen:

Special Function Register IE (Interrupt Enable):

EA	0: alle Interrupts gesperrt	1: alle Interrupts freigegeben
ET0	0: Interrupt von Timer0 gesperrt	1: Interrupt von Timer0 freigegeben
ET1	0: Interrupt von Timer1 gesperrt	1: Interrupt von Timer1 freigegeben

TMOD: Timermodus-Kontrollregister für Timer1 und Timer0

Kontrolle Timer1				Kontrolle Timer0			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Gate	C/T	M1	M0	Gate	C/T	M1	M0
		0	0	Modus 0			
		0	1	Modus 1: 16-Bit-Timer ohne Nachladen			
		1	0	Modus 2: 8-Bit-Timer mit Auto-Reload			
		0	0	Modus 3: 2 Stück 8-Bit-Timer			
	0	Timer-Betrieb					
	1	Zähler-Betrieb					
0	Timer nur durch TR-Bit ein- und ausschalten						
1	Timer mit TR-Bit und Portpin ein- und ausschalten						

TCON: Timer-Kontrollregister für Timer1 u. 0, ext. Interrupt 1 u. 0

Kontrolle Timer1 u. 0				ext. Interrupt-Kontrolle			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
			0	Timer0 stopp			
			1	Timer0 läuft			
		0, 1	Wird beim Timer0-Überlauf gesetzt				
	0	Timer1 stopp					
	1	Timer1 läuft					
0, 1	Wird beim Timer1-Überlauf gesetzt						

Bei einem 16-Bit-Timer wird der Inhalt von TH0 und TL0 folgendermaßen berechnet:

2^{16} - Zeit nach der ein Interrupt ausgelöst werden soll (in µs) = TH0 | TL0

65536d – 10000d = 55536d (D8F0h) => TH0 = D8h, TL0 = F0h

Assembler Timer Interrupt:

Beispielcode für einen Zähler der alle 10 ms ein Interrupt auslöst:

```
                include reg_51.pdf
                LEDs    equ     p2

                code at 0
                ljmp    go

                org     000Bh
ISRtimer:       ljmp    ISRoutine

                org     100d
go:             mov     TMOD,#00000001b
                mov     TL0,#0F0h
                mov     TH0,#0D8h
                mov     TCON,#00010000b
                mov     ip,#0
                setb    ET0
                setb    EA
                mov     LEDs,#0

haupt:          sjmp    haupt

ISRoutine:      inc     LEDs
                mov     TL0,#0F0h
                mov     TH0,#0D8h
                reti

                end
```

C Timer Interrupt:

Beispielcode für einen Zähler der alle 10 ms ein Interrupt auslöst:

```
#include <at8951ed2.h>
sfr at P2 LEDs;
void timer0int(void);

void main(void)
{
  LEDs = 0;
  TMOD = 0x01;
  TL0 = 0x0F0;
  TH0 = 0x0D8;
  TCON = 0x10;
  IP = 0x00;
  ET0 = 1;
  EA = 1;

  while(1)
  {
  }
}

void timer0int(void) interrupt 1
{
  TL0 = 0x0F0;
  TH0 = 0x0D8;
  LEDs++;
}
```

C Zeitschleife:

```
void zeitschleife(unsigned char ms)
{
  unsigned char z1, z2;

  for(z1 = ms; z1 != 0; z1--)
  {
    for(z2 = 249; z2 != 0; z2--);
    for(z2 = 249; z2 != 0; z2--);
  }
}
```

Assembler Zeitschleife:

schleife:	*mov*	r0,#00Ah	1					
loop1:	*mov*	r1,#0FFh	1		1		1	
loop:	*djnz*	r1,loop	256 x 2		256 x 2		256 x 2	
	djnz	r0,loop1	2		2		2	
	ret						2	
					512 x 8			
			516	+	4120	+	517	= 5,153 ms

Wahrheitstabelle mit Assembler:

c	b	a	y1	y0
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

```
        include reg_51.pdf

        code at 0

start:  mov a, P1
        mov dptr, #datab
        movc a, @a+dptr
        mov P2, a
        sjmp start

datab:  DB 00h,
        DB 01h,
        DB 01h,
        DB 02h,
        DB 01h,
        DB 02h,
        DB 02h,
        DB 03h,

        end
```

BCD-Code:

BCD oder BCD-Code (von engl. Binary Coded Decimal = de. dualkodierte Dezimalziffer), bezeichnet in der Informatik den 8-4-2-1-Code. Dabei handelt es sich um einen numerischen Code, der jede Ziffer einer Dezimalzahl einzeln dualkodiert. Die Ziffernfolge 8-4-2-1 steht dabei für die Werte der Stellen in einer dualkodierten Dezimalziffer. Im Einzelfall wird die Bezeichnung BCD auch synonym zu Zifferncode verwendet, womit die allgemeine Binärkodierung einzelner Dezimalziffern gemeint ist.

Umwandlung in BCD-Code mit Assembler:

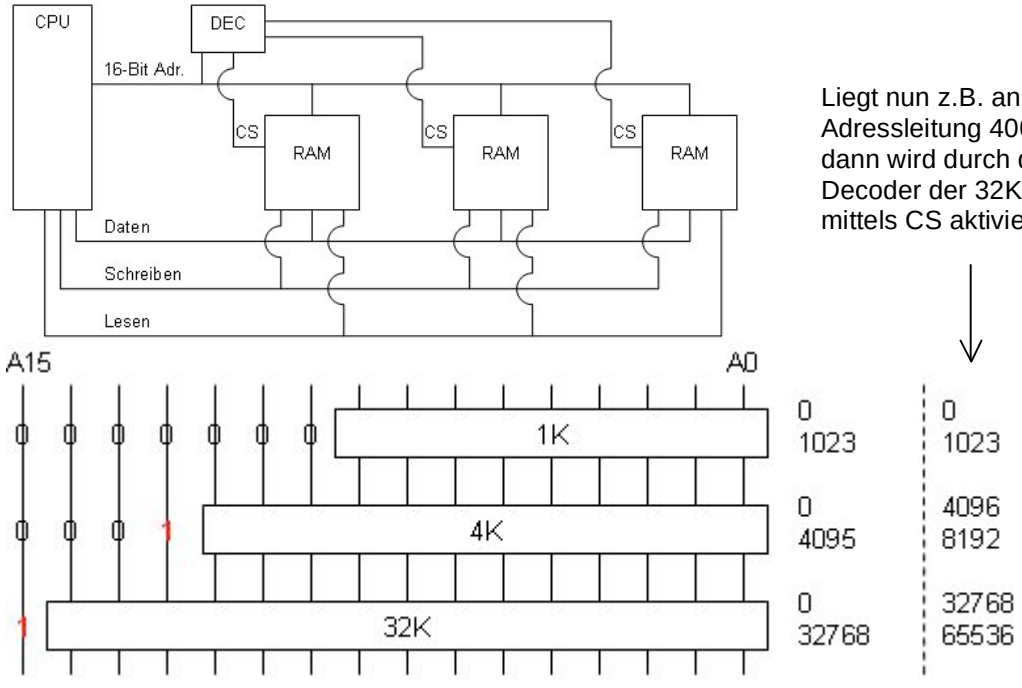
//angenommen: r1 = 54

```
Umwandlung:   mov B, #10        //10 in B laden, weil man durch 10 teilen muss um Zehner
                                //und Einer zu trennen
              mov A, r1         //Inhalt von r1 in A laden
              div ab            //A durch B teilen
                                //Danach steht in A 00000101 und in B 00000100
              swap a            //High- und Lowbit vertauschen
                                //Danach steht in A 01010000
              orl a, b          //A und B mit OR verknüpfen
                                //Danach steht in A 01010100
              mov P2, a         //A an P2 ausgeben
              ret
```

Chipselect:

Als Chipselect wird in der Digitaltechnik ein binäres Signal an einem integrierten Schaltkreis bezeichnet, mit dem man die Funktion eines solchen Schaltkreises ganz oder teilweise abschalten kann. Dies wird zum Beispiel gebraucht, um durch mehrere Speicherbausteine einen größeren Gesamtspeicher mit fortlaufenden Adressen zu konstruieren. Hierbei werden Busleitungen parallel an mehrere Speicherbausteine gelegt und mit dem Chipselect-Signal nur ein bestimmter Baustein ausgewählt, mit dem gearbeitet werden soll.
Häufig sind Chipselect-Eingänge "Low-Aktiv", das heißt der Baustein wird nicht mit einer logischen "1" aktiviert, sondern mit logisch "0".

Bsp. mit einem 1K, 4K und 32K RAM:



Liegt nun z.B. an der Adressleitung 40000 an, dann wird durch den Decoder der 32K RAM mittels CS aktiviert.

2. Software

Inhalt:

- [CT] OOA/OOD
  - Objektorientierter Ansatz
  - Objekte
  - Kapselung
  - Botschaften (Nachrichten)
  - Klassen
  - Konstruktor
  - Destruktor
  - Vererbung
  - Sicherheit von Attributen und Operationen
  - Abstrakte Klassen
  - Späte Bindung
  - Polymorphie

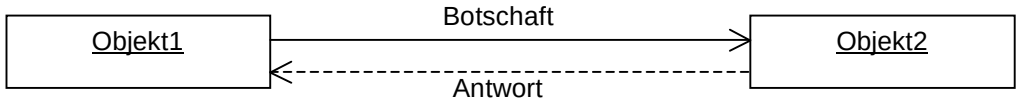
Hinweis:

Auf UML gehe ich hier nicht ein, da alles in der Formelsammlung steht.

2.1. [CT] OOA/OOD

Objektorientierter Ansatz:

Beim der objektorientierten Programmierung wird die Software in Objekte aufgeteilt. Diese Objekte bestehen aus selbstständig funktionsfähigen Teilen der Software. Untereinander kommunizieren die Objekte indem sie sich Botschaften schicken.



Objekte

Objekte besitzen Attribute (Datenspeicher) und Operationen (Methoden). Durch Operationen werden Attribute verändert (der innere Zustand des Objekts wird verändert).

Ein Objekt:

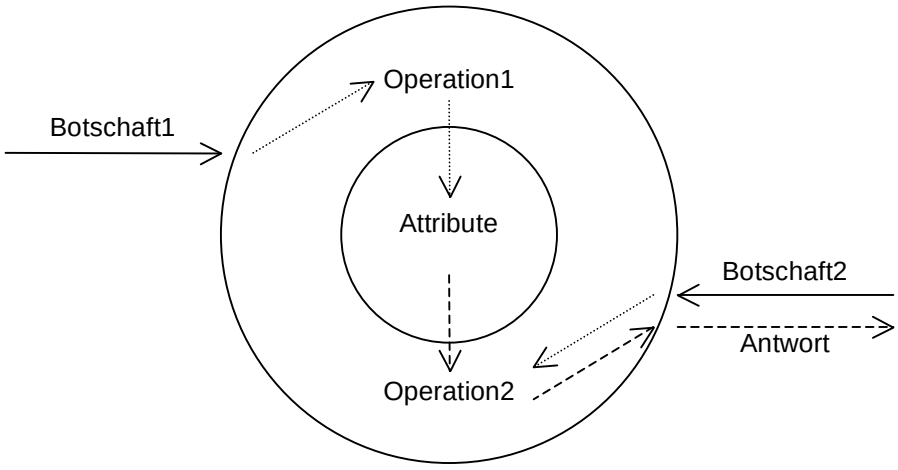
- Reagiert mit einem definierten Verhalten auf seine Umwelt (d.h. auf Botschaften). Das Verhalten ist vom Zustand des Objekts abhängig.
- Besitzt einen Zustand, der das Verhalten des Objekts beeinflusst
    - Der Zustand wird durch die Werte der Attribute beschrieben.
    - Das Verhalten eines Objekts wird durch eine Menge von Operationen (Methoden) beschrieben.
    - Die Operationen werden durch Botschaften aktiviert.

Eigenschaften von Objekten:

- Alle Objekte verfügen über die in der Klassendefinition festgelegten Attribute und Operationen.
- Jedes Objekt hat jedoch seine eigenen Attributwerte.
- Unabhängig von den Attributwerten hat jedes Objekt seine eigene Identität. Auch wenn zwei Objekte identische Attributwerte besitzen, sind sie doch unterscheidbar.

Kapselung:

Die Attribute eines Objekts können nur über Operationen verändert und gelesen werden. Ein direkter Zugriff auf die Attribute (Daten) eines Objekts ist nicht möglich.



Botschaften (Nachrichten):

Botschaften richten sich immer an ein Objekt und übermitteln diesem den Namen der auszuführenden Operation (Methode). Für Botschaften wird deshalb der selbe Bezeichner, wie für die Operation die aktiviert werden soll, verwendet

Klassen:

Klassen definieren die Operationen und Attribute für Objekte. Alle Objekte einer Klasse entsprechen dieser Definition. Jede Klasse besitzt eine Operation zum Erzeugen von Objekten (Instanzen): den Konstruktor.

Konstruktor:

Aufgaben:

- Erzeugen von Instanzen (Objekten).
  Wenn bei einer Klasse der Konstruktor nicht ausdrücklich deklariert ist, so wird vom Compiler ein „Default“ - Konstruktor (ohne Parameter) erstellt und benutzt.
- Zuweisen von Initialisierungswerten (an die Attribute).
  Hierzu kann der Konstruktor auch Parameter besitzen.
- Verallgemeinerung.
  Der Konstruktor enthält alle Anweißungen, die beim Erzeugen einer Instanz ausgeführt werden sollen.

Destruktor:

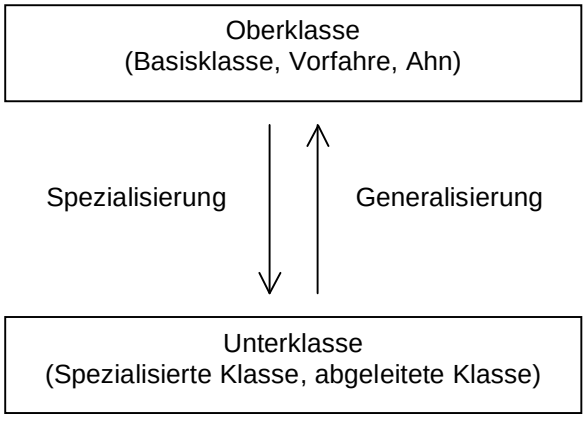
Der Destruktor enthält alle Anweisungen, die vor dem Zerstören (vernichten) einer Instanz ausgeführt werden müssen. Bei C++ wird der Destruktor beim Zerstören einer Instanz automatisch ausgerufen. Destruktoren haben keine Parameter. Eine statisch erzeugte Instanz wird beim Verlassen des Blocks, in dem sie definiert wurde (wie auch lokale Variablen) automatisch aus dem Speicher entfernt (zerstört).

Vererbung:

Durch Vererbung werden alle Operationen und Attribute der Klasse in eine Unterklasse übernommen, in die zusätzlich neue Attribute und Operationen eingefügt werden können.

Die Vorteile sind:

- Neue Klassen müssen nicht vollständig neu erstellt werden.
  Es müssen lediglich neue Operationen und Attribute erstellt werden.
- Durch vererben entsteht eine Vererbungsstruktur, eine so genannte Klassenhierarchie.



Sicherheit von Attributen und Operationen:

Private (-) : Nur innerhalb der Klasse sichtbar.
Protected (#): Innerhalb der Klasse sichtbar und in allen Unterklassen.
-> kein Zugriff auf andere Objekte.
Public (+): Für alle anderen Klassen (Objekte) sichtbar. Widerspricht jedoch der Entwurfsregel (Kapselung), wenn es auf Attribute angewendet wird.

Abstrakte Klassen:

Von einer abstrakten Klasse können keine Instanzen erzeugt werde. In der Regel wird in abstrakten Klassen ein Teil der Operationen nicht implementiert sondern deklariert. Die Operationen müssen in Abgeleiteten Klassen überschrieben werden. Im Klassendiagramm wird der Klassenbezeichner kursiv geschrieben und hinter den Klassenbezeichner wird {abstract} gesetzt.

Späte Bindung:

Späte Bindung liegt vor, wenn erst zur Laufzeit eines Programms festgestellt wird, welche Operation durch eine bestimmte Botschaft aktiviert werden soll. Damit späte Bindung möglich wird, muss die Operation in der Basisklasse mit „virtual“ deklariert werden.

Bsp.:
Kontoverwaltung -> buchen(). Der Compiler kann beim kompilieren nicht wissen, ob das Konto zu dem Objekt Girokonto oder Sparkonto gehört. Daher muss dies zur Laufzeit festgestellt werden.

Polymorphie:

Durch späte Bindung wird Polymorphie (Vielgestaltigkeit) ermöglicht.

Polymorphie liegt vor, wenn gleich lautende Botschaften an kompatible Objekte unterschiedlicher Klassen ein unterschiedliches Verhalten bewirken. Beispiel: Konstruktoren mit verschiedenen Parametern.

Statische Polymorphie:

In einer Klasse können mehrere Operationen denselben Namen besitzen, wenn sich ihre Signatur unterscheidet -> Operation wird überladen

Die Signatur ist nur dann unterschiedlich, wenn
- die Anzahl der Parameter unterschiedlich ist.
- der Datentyp mindestens eines Parameters unterschiedlich ist.

Ein unterschiedlicher Datentyp des Funktionswertes reicht nicht aus. Typisches Anwendungsbeispiel: mehrere Konstruktoren mit unterschiedlichen Parametersätzen.

Überladen -> innerhalb einer Klasse.
Überschreiben -> von einer Unterklasse überschrieben.

3. Systeme

Inhalt:

- [CT] Netzwerksysteme
  - Internet Protokoll (Schicht 3)
  - IP-Adresse
  - Subnetzmaske
  - Broadcast- und Netzadresse
  - Routing-Tabelle
  - Router
  - Routing
  - Protokollnummern
  - ICMP (Schicht 3)
  - TCP (Schicht 4)
  - UDP (Schicht 4)
  - NAT
  - Masquerading / PAT
  - Netzklassen
  - OSI Schichtenmodell
  - Beispiel Analyse von ICMP (Ping)
  - Hub & Co
  - Wichtige IP's
  - HDLC (Schicht 2)
  - CRC (cyclic redundancy check, zyklische Redundanzprüfung)
  - CSMA/CD
  - ARP/RARP
  - E-Mail Adresse
  - E-Mail Header
  - E-Mail Body
  - SMTP (Simple Mail Transfer Protocol)
    POP3 (Post Office Protocol Version 3)
  - IMAP (Internet Message Access Protocol)
  - SMAP (Simple Mail Access Protocol)
  - Webmail
  - PGP (Pretty Good Privacy)
  - S/MIME
  - SSL (Secure Sockets Layer) / Transport Layer Security (TLS)
  - Token Ring
  - Topologien
  - Verbundarten
  - Ethernet Arten
  - Koaxialkabel
  - Twisted Pair
  - Multimode-Stufenfaser
  - Multimode-Gradientenfaser
  - Monomodefaser
  - Vergleich der Glasfasertypen
  - Serielle Schnittstelle
  - RS-232
  - Paritätsbit

- Betriebssysteme
  - Vorgänge beim Laden eines Programms
  - Komponenten eines OS
  - Definition Betriebssystem
  - API
  - Betriebssystemarten

- Dateiverwaltung
  - Aufgaben der Dateiverwaltung
  - Dateiverwaltung
  - Unterschiedliche Ebenen der Dateiverwaltung
  - Kontinuierliche Allokation
  - Gestreute Allokation
  - [CT] FAT (Gestreute Allokation)
  - Berechnungen
  - Zugriff auf die einzelnen Cluster einer Datei mittels Index-Knoten (i-node)

- [CT only] Zugriffsrechte
  - Zugriffsrechte Linux

- Speicherverwaltung
  - Speicherverwaltung
  - Direkte Adressierung des Arbeitsspeichers
  - Virtueller Speicher
  - Segmentadressierung
  - Seitenadressierung

- Prozessverwaltung
  - Ziele der Prozessverwaltung
  - Prozesszustände
  - Prozesswechselmethoden
  - Arbeitsumgebung der Prozesse
  - Prozesse / Thread
  - Scheduling-Algorithmen
  - Verklemmung (Deadlock)
  - Begriffserklärung (Wikipedia)

- Datensicherung
  - Datensicherung
  - Großvater-Vater-Sohn Datensicherung

- [CT] DBMS
  - Begriffe / Definitionen
  - Hierarchisches Datenbankmodell
  - Netzwerkartige Datenbanken
  - Relationales Datenbankmodell
  - Objektorientirtes bzw. objektrelationales Datenbankmodell
  - Schlüsselkanidaten
  - Primärschlüssel
  - Fremdschlüssel
  - Kardinalität
  - Relationsschreibweise
  - ER-Diagramm, Relationsschreibweise und Implementierung
  - Normalisierung
  - Anomalien
  - Grundgerüst einer SQL-Abfrage
  - LIKE
  - Aggregatfunktionen
  - Verbund (Join)
  - UNION

3.1. [CT] Netzwerksysteme

Internet Protokoll (Schicht 3):

Aufgabe:
- Daten vom Sender zum Ziel leiten
- Adressierung
- Fragmentierung

IP-Adresse:

- Eine IPv4 Adressen besteht aus 32 Bits.
- Alle 8 Bits kommt ein Punkt.

Bsp.:

Dez	192	. 168	.1	. 1
Bin	1100 0000	. 1010 1000	. 0000 0001	. 0000 0001

Subnetzmaske:

Eine Subnetzmaske wird benötigt um den Netz- und den Hostanteil einer IP-Adresse fest zu legen. Rechner die in einem anderen Subnetz liegen können nicht direkt miteinander kommunizieren.

Bsp.:

Dez	255	. 255	. 255	. 192
Bin	1111 1111	. 1111 1111	. 1111 1111	. 1100 0000

Broadcast- und Netzadresse:

- Verknüpft man die Subnetzmaske und die IP-Adresse mit einem AND erhält man die Netzadresse.
- Wird ein Paket an die Broadcastadresse adressiert, erhalten alle Rechner im selben Subnetz dieses Paket.
- Die Broadcastadresse erhält man indem man den Host Anteil der IP-Adresse mit Einsen auffüllt.
- Die Netadresse und die Broadcastadress dürfen keinem Rechner zugewiesen werden, daher gilt:
  max. Anzahl der Rechner in einem Subnetz = $2^{\text{Anzahl der Host Bits}} - 2$

Bsp.:
Subnetzmaske:

Dez	255	. 255	. 255	. 192
Bin	1111 1111	. 1111 1111	. 1111 1111	. 1100 0000

IP-Adresse:

Dez	192	. 168	.1	. 206
Bin	1100 0000	. 1010 1000	. 0000 0001	. 11001110

Netadresse:

Dez	192	. 168	.1	. 192
Bin	1100 0000	. 1010 1000	. 0000 0001	. 1100 0000

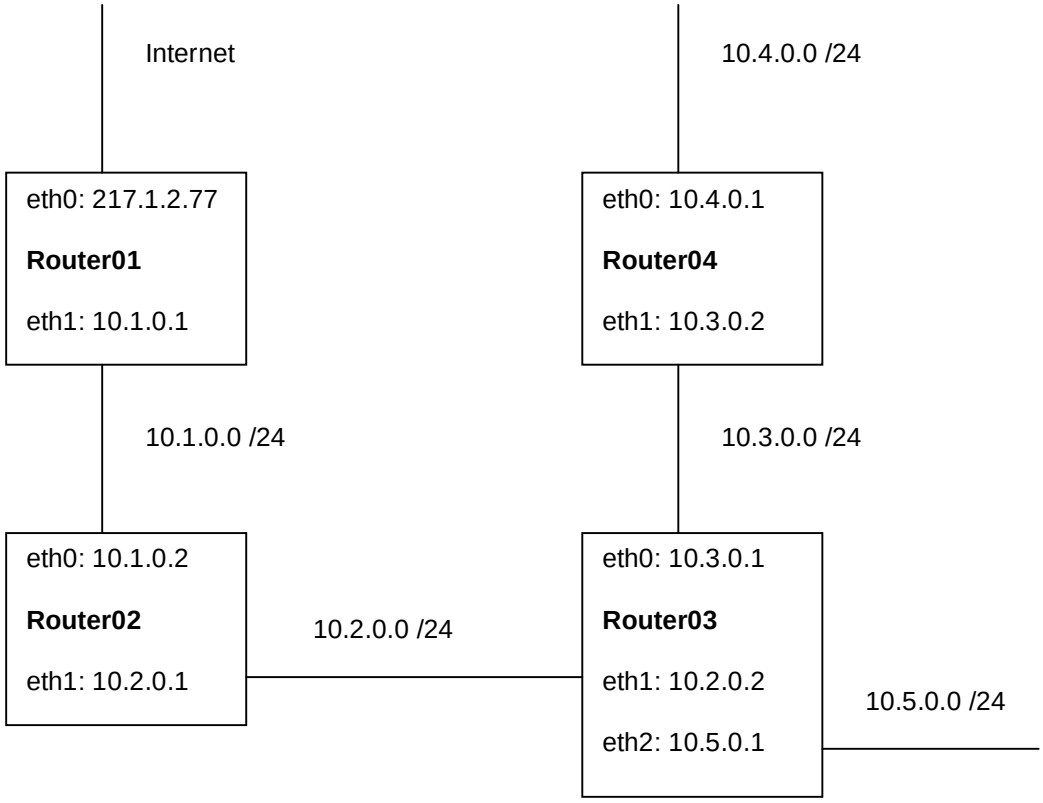
Broadcastadresse:

Dez	192	. 168	.1	. 255
Bin	1100 0000	. 1010 1000	. 0000 0001	. 1111 1111

Routing-Tabelle:

Bsp. für Router03:

Ziel/Netz	Router/Gateway	Mask	Interface
10.1.0.0	10.2.0.1	255.255.255.0	eth1
10.2.0.0	*	255.255.255.0	eth1
10.3.0.0	*	255.255.255.0	eth0
10.4.0.0	10.3.0.2	255.255.255.0	eth0
10.5.0.0	*	255.255.255.0	eth2
default	10.2.0.1	0.0.0.0	eth1



Router:

Ein Router ist ein Vermittlungsrechner der mehrer (Sub-)Netze koppelt, d.h. anhand von Schicht-3 Informationen leitet er ein Paket in das Zielnetz oder zu einem, am Zielnetz näher liegenden Router, weiter.

Routing:

Statisch:
Jeder Knoten unterhält eine Tabelle mit einer Zeile für jeden möglichen Zielknoten. Eine Zeile enthält Einträge, welche die beste, zweitbeste usw. Übertragungsleitung für dieses Ziel ist zusammen mit einer Gewichtung. Vor Weiterleitung eines Paketes wird der entsprechende Eintrag aus der Tabelle gewählt und auf eine der möglichen Leitungen gegeben.

Dynamisch:
Beim dynamischen Routing gibt es keine feste Routing-Tabelle, d.h. die Routingentscheidungen hängen vom Zustand des Netzes ab.

Protokollnummern:

Die Protokollnummern stehen in der IP-Header.

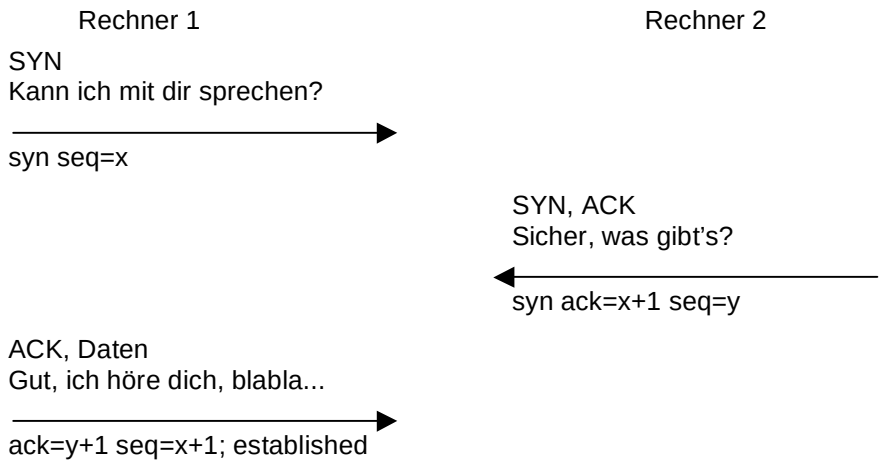
Protokollnummer	Protokoll
1	ICMP
2	IGMP
4	IP encapsulation
6	TCP
8	EGP
17	UDP
89	OSPF

ICMP (Schicht 3):

Das *Internet Control Message Protocol* (**ICMP**) benutzt wie TCP und UDP das Internet Protocol (IP), ist also ein Teil der Internetprotokollfamilie. Es dient in Netzwerken zum Austausch von Fehler- und Informationsmeldungen(Erkennt einige Fehlerzustände). Zum Beispiel ist der Ping und Traceroute eine Anwendung des ICMP.

TCP (Schicht 4):

TCP (Transmission Control Protocol) gewährleistet die zuverlässige Nachrichtenübermittlung zwischen Sender und Empfänger. TCP ist weit verbreitet und an keinen Hersteller gebunden. Erfolgt auf ein gesendetes Paket keine Antwort, wird erneut gesendet. Jede Anwendung bekommt einen Socket über die gleichzeitig gesendet werden kann (Multiplexing).
Three-Way-Handshake:



syn: synchronize
seq: Sequenznummer
ack: acknowledgment (Bestätigung)

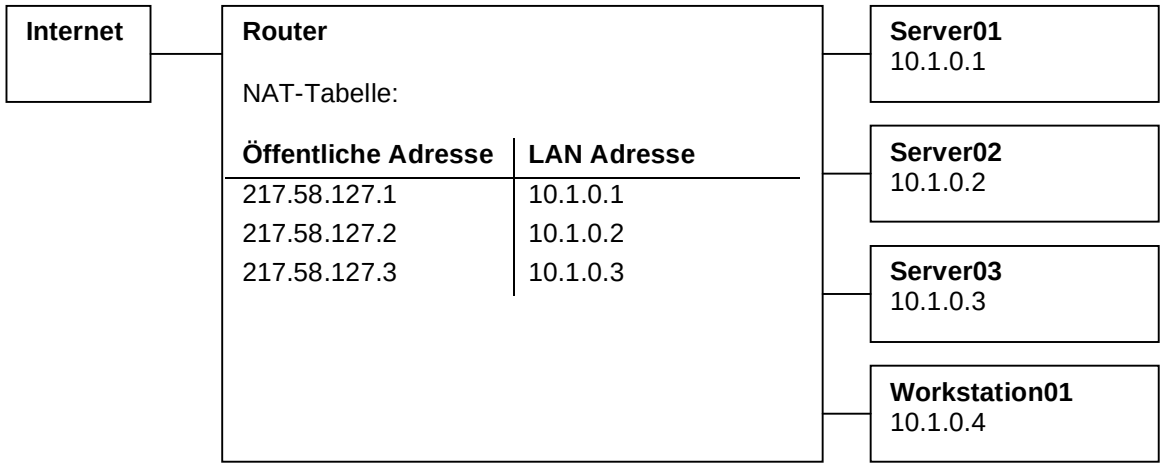
UDP (Schicht 4):

UDP (User Datagram Protocol) ist eine vereinfachte Form von TCP. Bei UDP gibt es keine Empfangsbestätigungen für Pakete, noch gibt es eine Rückmeldung über die Frequenz gesendeter Nachrichten. Daher ist bei UDP nur eine unzuverlässige und verbindungslos Datenübertragung möglich. Jedoch erreicht UDP die dreifache Geschwindigkeit von TCP.

NAT:

Durch NAT (Network Address Translatation) werden Pakete die an eine bestimmte öffentliche IP-Adresse gesendet werden an eine LAN-Adresse weitergeleitet. Damit ist es möglich, dass ein Server in einem LAN vom Internet aus angesprochen werden kann. Normalerweise müsste man um so etwas zu erreichen, jedem LAN-Rechner eine öffentliche IP-Adresse zuweisen, aber es gibt nicht genügend öffentliche IP-Adressen. Jedoch muss man um das NAT nutzen zu können mehrere statische, öffentlich IP-Adressen besitzen.

Bsp.:



Masquerading / PAT:

Beim PAT (Port Address Translation), wird nicht wie bei NAT, eine öffentliche IP-Adresse auf eine lokale IP-Adresse abgebildet. Sonder es wird zwischen den verschiedenen Ports differenziert, d.h. man benötigt nur eine öffentliche IP-Adresse und wird an diese eine Anfrage (z.B. http Port 80) gesendet wird diese Anfrage an einen im LAN befindlichen Server weitergeleitet.

Bsp.:

Port	LAN-Rechner
22	10.1.0.1
80	10.1.0.2
21	10.1.0.3

Netzklassen:

Früher gab es fest vorgeschriebene Einteilungen für Netzwerkklassen mit einer festen Länge. Diese Einteilung erwies sich allerdings als zu unflexibel, so dass man seit 1993 dazu übergegangen ist, sie bitvariabel im Classless Inter-Domain Routing-Verfahren durchzuführen. Heutzutage kann man genaugenommen nicht mehr von Klasse-A/B/C-Netzwerken sprechen, allerdings ist es im allgemeinen Sprachgebrauch erhalten geblieben und viele netzwerkfähige Betriebssysteme bestimmen die Standardnetzmaske anhand der alten Klassifikation.

Class A: Netze	0.0.0.0/8	bis 127.255.255.255/8
Class B: Netze	128.0.0.0/16	bis 191.255.255.255/16
Class C: Netze	192.0.0.0/24	bis 223.255.255.255/24
Class D: Multicast-Gruppen	224.0.0.0/4	bis 239.255.255.255/4
Class E: Reserviert	240.0.0.0/5	bis 255.255.255.255/5

OSI Schichtenmodell:

Schicht	Protokollbeispiel	Koppelungselement
7 Anwendung	HTTP FTP HTTPS SSH	Gateway
6 Darstellung		
5 Sitzung		
4 Transport	TCP, UDP	
3 Vermittlung	IP, ICMP	Router
2 Verbindung	Ethernet	Switch, Bridge
1 Bitübertragung		Hub, Repeater

Beispiel Analyse von ICMP (Ping) :

MAC Anteil
0000 00 0f b5 80 07 86 00 11 d8 23 dd c5 08 00 45 00
Ziel
Quelle
Typ (hier: IP)

IP Anteil:
0000 00 0f b5 80 07 86 00 11 d8 23 dd c5 08 00 45 00
0010 00 3c 9a 22 00 00 80 01 8c 91 0a 01 00 0b 0a 01
0020 00 01 …

Version
IP Header Length (IHL) IHL*4 = Länge des Headers in Byte
Type Of Service (TOS) (Priorität,…)
Paketlänge
Identifikation (Integer Zahl, um den Datenstrom korrekt zusammen zu setzen)
Flags und Fragmentbestand (Zeigt an ob das Paket geteilt wurde / ob weitere Pakete kommen)
Time To Live (TTL)
Protokoll (hier: ICMP siehe Tabelle auf Seite 3)
Kopf-Prüfsumme
IP-Sendeadresse
IP-Emfängeradresse

ICMP Anteil:
0020 00 01 08 00 43 5c 03 00 07 00 61 62 63 64 65 66
0030 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76
0040 77 61 62 63 64 65 66 67 68 69
ICMP-Typ (hier: Ping)
ICMP-Code
Prüfsumme
Daten

Hub & Co :

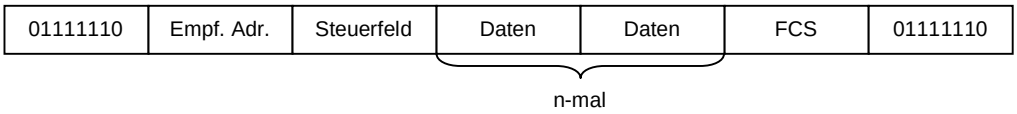
Schicht 1	Repeater	Ein Repeater bereitet empfangene Signale neu auf und leitet sie weiter. Ein Repeater mit mehreren Eingängen wird auch Hub genannt.
	Hub	Ein Hub wird in der Stern-Topologie eingesetzt um das Signal aufzubereiten und an alle angeschlossenen Rechner weiterzuleiten. Kommen zwei Packete gleichzeitig an geht eines verloren, weil der Hub nicht in der Lage ist zwei gleichzeitig zu verarbeiten.
Schicht 2	Switch / Bridge	Switches verbinden mehrere Rechner oder Netzsegmente. Sie analysieren die einzelnen Packete und leiten diese nur an die Ziel-MAC-Adresse weiter.Datenkollisionen kommen nicht vor. Die Netze werden mit phyiskalischen Adressen (MAC) voneinander getrennt. Broadcasts werden weitergeleitet.
Schicht 3	Router	Ein Router verbindet verschiedene Netze mithilfe logischer Adressen (IP-Adressen). Ein Router kann unterschiedliche Schicht 2 Protokolle (Ethernet &WLAN) miteinander verbinden. Fehlerhafte Datenpakete werden nicht weitergeleitet.
Alle Schichten	Gateway	Ein Gateway verbindet Netze völlig unterschiedlicher Protokolle miteinander.

Wichtige IP's:

127.0.0.0	Localnet
127.0.0.1	Localhost (z.B. http://127.0.0.1/ oder http://localhost/)
255.255.255.255	Broadcast

HDLC (Schicht 2):

High-Level Data Link Control ist ein Netzwerkprotokoll.



CRC (cyclic redundancy check, zyklische Redundanzprüfung):

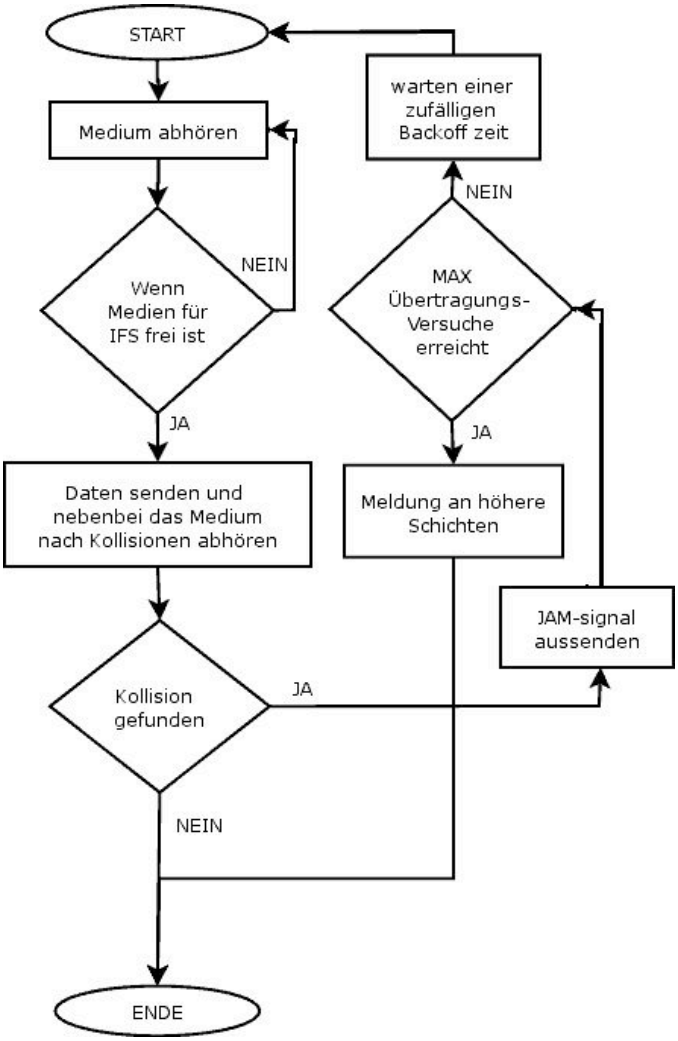
Bsp.: f=11100101, g=11011
An f müssen noch Nullen angehängt werden (g– 1 = Anzahl Nullen)

```
 1 1 1 0 0 1 0 1 0 0 0 0 : 1 1 0 1 1 = 1 0 1 0 1 1 0 0
 1 1 0 1 1
 0 0 1 1 1 1 0
     1 1 0 1 1
     0 0 1 0 1 1 0
         1 1 0 1 1
         0 1 1 0 1 0
           1 1 0 1 1
           0 0 0 0 1 0 0 → CRC-Summe
 ↑   ↑   ↑ ↑
 Immer mit der ersten gemeinsamen 1 anfangen
```

CSMA/CD:

Das CSMA/CD ist ein Medienzugriffsverfahren, das Kollisionen vermeiden soll oder nach einer Kollision eine weitere verhindern soll.

Funktionsweise:

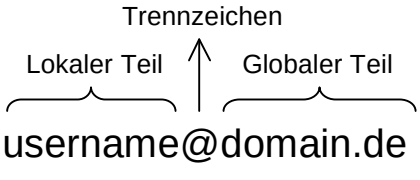


ARP/RARP (Schicht 2):

Das Address Resolution Protocol (ARP) ist ein Netzwerkprotokoll, das die Zuordnung von Netzwerkadressen zu Hardwareadressen möglich macht.
Das Reverse Address Resolution Protocol (RARP) funktioniert umgekehrt zu ARP. Es kann also MAC-Adressen zu IP-Adressen auflösen.

E-Mail Adresse:

Mit Hilfe der E-Mail Adresse kann der Empfänger und Absender einer E-Mail genau festgelegte werden. Die Adresse besteht genau aus zwei Teilen, dem lokalen Teil und dem globalen Teil. Die beiden Teile werden durch das „@“ getrennt.



E-Mail Header:

Im Header einer E-Mail findet man in der Regel Informationen über den Weg, den die E-Mail genommen hat, über den Inhalt der Nachricht, über deren Format und Angaben zum Empfänger.

Headerbestanteile:

- DATE
  Hier steht der Zeitpunkt des Absendens der E-Mail.
- FROM
  Eine oder mehrere durch Komma getrennte E-Mail-Adressen, die den oder die Absender einer E-Mail bezeichnen.
- SENDER
  Hier steht der technische Absender der E-Mail.
- SUBJECT
  Zeigt eine Kurzinformation über den Inhalt der E-Mail auf.
- REPLY-TO
  Eine oder mehrere durch Komma getrennte E-Mail Adressen, an die jeweils eine Antwort auf die E-Mail geschickt wird.
- TO
  Eine oder mehrere durch Komma getrennte E-Mail Adressen, an die die E-Mail primär gesendet wird.
- CC (Carbon Copy)
  Eine oder mehrere durch Komma getrennte E-Mail Adressen, an die eine Kopie der E-Mail geschickt wird.
- BCC (Blind Carbon Copy)
  Eine oder mehrere durch Komma getrennt E-Mail Adressen, an die eine Blindkopie geschickt wird. Mit dieser Blindkopie wird die Liste der Empfängeradressen nicht mitgeschickt.
- RECIEVED
  Liste der Mailserver, durch die die E-Mail durchlaufen ist.

E-Mail Body:

Der Body besteht aus Informationen die in einem oder mehreren Teilen übertragen werden. Die Information kann aus Text und anhängenden Dateien bestehen, welche als so genannten Anhang mitgeschickt werden. Dieser Anhang muss zuerst kodiert und anschließend als Text in den Body mit eingefügt werden. Diese Kodierung des Anhangs wird für jede mitgeschickte Datei einzeln durchgeführt und durch die Kodierung „MIME“ (Multipurpose Internet Mail Extension) geregelt.

SMTP (Simple Mail Transfer Protocol):

Das SMTP-Protokoll dient zum Austausch von E-Mails zwischen den einzelnen Computernetzen. Das Protokoll wird hierbei primär zum Einspeisen und Weiterleiten von E-Mails verwendet. Die Abwicklung durch dieses Protokoll wird meist unsichtbar für den Anwender durchgeführt, vom so genannten Mail User Agent (MUA). Das Client-Programm verbindet sich dann mit dem SMTP-Server, dem so genannten Mail Submission Agent (MSA). Dieser schickt die E-Mail nun zum Ziel, so kann diese gegeben falls über mehrere SMTP-Server laufen, die so genannten Mail Transfer Agent (MTA), bis diese ihr Ziel erreicht.

POP3 (Post Office Protocol Version 3):

Das POP3-Protokoll Ist ein Übertragungsprotokoll und dient dazu, um mit einem E-Mail Client, E-Mails von einem POP-Server abzuholen. Beim POP3-Protokoll ist keine ständige Verbindung zum POP-Server nötig. Das Protokoll ist jedoch in den Funktionen eingeschränkt und erlaubt nur das Abholen und Löschen von E-Mails.

IMAP (Internet Message Access Protocol):

Das IMAP-Protokoll erlaubt den Zugriff auf E-Mails, sowie die Verwaltung der E-Mails auf dem IMAP-Server. Die E-Mails verbleiben hierbei auf dem Mailserver und werden nur nach Bedarf auf den Client-Rechner übertragen. Dabei verhält sich das Protokoll als wären die empfangenen E-Mails auf dem lokalen Rechner. Für Zugriff auf die E-Mails ist eine Internetverbindung nötig.

SMAP (Simple Mail Access Protocol):

Das SMAP-Protokoll ist eine Weiterentwicklung vom IMAP-Protokoll. Dieses Protokoll bietet folgende Vorteile, welche mit IMAP nicht möglich sind:

- 25% weniger Bandbreitenbedarf bei MIME-kodierten Dateianhängen
- Erlaubt das gleichzeitige Senden und Ablegen einer E-Mail (bei IMAP sind zwei Schritte nötig)
- Verzeichnisnamen nutzen die Unicode Text-Kodierung => keine Sorgen mehr über zugrundeliegender Dateistruktur
- Beim öffnen eines Verzeichnisses muss kein kompletter Index vom Server wieder heruntergeladen werden => Index wird einfach aktualisiert

Webmail:

Unter Webmail versteht man die Verwaltung von E-Mails mit Hilfe eines Webbrowsers. Der Vorteil hier liegt bei der geringen Anforderung die beim Webmail benötigt wird. Ein Nachteil ist, dass beim Webmail oft weniger Funktionen zur Verfügung stehen als beim einem E-Mail Programm (MUA).

PGP (Pretty Good Privacy):

PGP ist ein von Phil Zimmermann entwickeltes Programm zur Verschlüsselung von Daten. Es benutzt ein sog. Public-Key-Verfahren, das heißt, es gibt ein eindeutig zugeordnetes Schlüsselpaar: Einen öffentlichen, mit dem jeder die Daten für den Empfänger verschlüsseln kann, und einen geheimen privaten Schlüssel, den nur der Empfänger besitzt und der durch ein Kennwort geschützt ist. Nachrichten an einen Empfänger werden mit seinem öffentlichen Schlüssel verschlüsselt und können dann nur durch den privaten Schlüssel des Empfängers entschlüsselt werden. Diese Verfahren werden auch asymmetrische Verfahren genannt, da Sender und Empfänger zwei unterschiedliche Schlüssel verwenden.

S/MIME:

Bei S/MIME handelt es sich um eine Erweiterung der MIME-Kodierung. Es wurde um die Funktionen Verschlüsselung und Signierung erweitert und arbeitet im Gegensatz zum PGP Verfahren nur mit Zertifikaten. Daher muss jeder Teilnehmer ein Zertifikat auf seinem Client-Rechner installiert haben. Zertifikate sind öffentliche Schlüssel, welche von einer CA (Certificate Authority) unterschrieben wurden. Hierbei handelt es sich um eine Client zu Client Verschlüsselung, was heißt das wir zum Verschlüsseln einmal das Zertifikat vom Empfänger und das von uns selbst brauchen.

SSL (Secure Sockets Layer) / Transport Layer Security (TLS):

TLS oder SSL ist ein Verschlüsselungsprotokoll für Datenübertragungen im Internet.
Im OSI-Modell ist SSL oberhalb der Transportschicht (z. B. TCP) und unter Anwendungsprotokolle wie HTTP oder SMTP angesiedelt. SSL arbeitet transparent, so dass es leicht eingesetzt werden kann, um Protokolle, die nicht über eigene Sicherheitsmechanismen verfügen, abgesicherte Verbindungen zur Verfügung zu stellen.

Token Ring:

Token Ring ist eine Vernetzungstechnologie für Computernetzwerke, festgelegt in der Spezifikation IEEE 802.5. Sie definiert Kabeltypen und Signalisierung für die Bitübertragungsschicht, Paketformate und Protokolle für die Medienzugriffskontrolle (Media Access Control, MAC)/Sicherungsschicht des OSI-Modells. Sie ist eine der beiden Realisierungsformen des Token-Passing-Verfahrens.
Die logische Topologie von Token Ring ist ein Ring.
Ein Token kreist bei Token-Ring-Netzen über den Ring: Das Token wird stets von einem Knoten an den nächsten weitergereicht. Selbst im Leerlauf geben die Stationen das Paket fortwährend weiter. Möchte nun ein Computer Daten versenden, wartet er, bis das Token ihn erreicht hat, dann hängt er seine Nutzdaten daran an. Zugleich ergänzt er das Token um Steuersignale und setzt außerdem das *Token-Bit* von 0 (für „freies Token“) auf 1, aus dem Frei-Token wird also ein Datenrahmen. Nach dem Vorgang setzt der Computer den Datenrahmen wieder auf den Ring, wo dieser genau wie das Frei-Token zuvor von den einzelnen Knoten weitergereicht wird. Jeder Rechner prüft, ob das Paket an ihn adressiert ist und setzt es anderenfalls zurück auf den Ring. Erhält der vorgesehene Empfänger den an ihn adressierten Datenrahmen, kopiert er die Nutzdaten und quittiert den Datenempfang. Der Sender erhält die Quittung und sendet den Token mit den nächsten Nutzdaten, oder setzt ein Frei-Token auf den Ring. Dabei darf ein Sender das Token nur eine bestimmte Zeit für sich in Anspruch nehmen, bevor er es wieder freigeben muss. Dadurch wird jedem Knoten in einem Ring garantiert, dass er nach Ablauf dieser festgelegten Zeit * die Anzahl der Knoten in einem Ring senden darf.

Topologien:

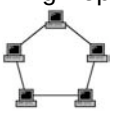
Es gibt in Netzwerken folgende Topologien:

- Bus-Topologie:

  Die Rechner sind in Reihe über ein Kabel, mit Abschlusswiderstand, verbunden.
  + Geringer Verkabelungsaufwand.
  + Einfache Erweiterbarkeit.
  - Bei Unterbrechung gesamtes Netz unterbrochen.

- Ring-Topologie:

  Jeder Rechner hat genau einen linken und einen rechten Nachbar. Die Pakete werden in definierter Weise weitergeleitet (z.B. Token Ring).
  + geringer Kabelaufwand.
  + Einfache Erweiterbarkeit
  - Bei Ausfall einer Station Ring unterbrochen.

- Stern-Topologie:

  Jeder Rechner ist mit einem zentralen Verteiler verbunden.
  + Hohe Ausfallsicherheit.
  - Hoher Kabelbedarf.
  Schwachpunkt: Verteiler.

Verbundarten:

Es gibt folgende Verbundarten:

- Datenverbund:
  Zugriff auf räumlich verteilte Daten
- Lastenverbund:
  Rechenlastverteilung -> Optimale Leistungsausnutzung.
- Funktionsverbund:
  Erweiterte lokale Funktionalität durch Zugriff auf netzweite Ressourcen (z.B. Netzdrucker).
- Leistungsverbund:
  Zerlegung eines Rechenproblems und Verteilung auf mehrere Knoten (z.B. Cluster).
- Verfügbarkeitsverbund:
  Stellt Mindestleistung an Ressourcen bei Ausfällen zur Verfügung (z.B. Hochverfügbarkeitsnetz).

Ethernet Arten:

10 MBit/s Ethernet:

- 10Base2:
  Koaxialkabel, max. 185m lang, max. 30 Teilnehmer.
- 10Base5:
  Koaxialkabel, max. 500m lang, max. 100 Teilnehmer.
- 10BaseT:
  Twisted Pair Kabel, max. 100m lang.
- 10BaseF:
  Glasfaserkabel.

100 MBit/s Ethernet:

- 100Base-T:
  Twisted Pair Kabel, max, 100m lang.
- 100Base-FX:
  Glasfaserkabel, max. 400m lang (mit Reapeater max. 2000m).

Gigabit Ethernet:

- 1000Base-T:
  Twisted Pair Kabel, max. 100m lang.
- 1000Base-LX/LH/ZX/CX/SX:
  Glasfaserkabel, 200m – 70000m möglich.

10 Gigabit Ethernet:

- 10GBase-T:
  Twisted Pair Kabel.
- 10GBase-LX4/SR/LR/ER:
  Glasfaserkabel, 26m – 40000m möglich.

Koaxialkabel:

Koaxialkabel sind zweiadrige Kabel mit konzentrischem Aufbau. Sie bestehen aus einem *Innenleiter* (auch *Seele* genannt), der von einem in konstantem Abstand um den Innenleiter angebrachten *Außenleiter* umgeben ist. Im Zwischenraum befindet sich ein Isolator bzw. Dielektrikum, dieses kann teilweise oder ganz aus Luft bestehen.

Vorteile:

- GHz-Frequenzbereich möglich.
- Preiswert.

Nachteile:

- Komplizierte Verlegung.
- Komplizierte Handhabung.

Twisted Pair:

Als Twisted-Pair-Kabel bzw. verseiltes Kabel werden in der Computertechnik Kabeltypen bezeichnet, bei denen die beiden Adern eines Adernpaares miteinander verseilt sind. Heute werden entsprechende Kabel meistens in der Netztechnik eingesetzt, zum Beispiel bei Ethernet-Kabeln, strukturierten Verkabelungen oder in der Feldbustechnik. Durch die jeweilige Verseilung des Hinleiters mit dem Rückleiter einer Stromschleife (das Adernpaar) ist die Datenübertragung weniger störanfällig.

Vorteile:

- Flexibel und einfach zu verlegen.
- Kostengünstig.

Es gibt folgende Arten von Twisted Pair Kabeln:

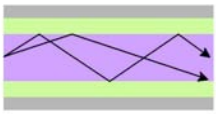
- STP:
  Paarweise geschirmte Kabel mit Gesamtschirmung aus Metallgeflecht.
- FTP:
  Wie STP allerdings Gesamtschirmung aus Metallfolie.
- S/UTP:
  Geschirmt wie STP, Schirm kann aber auch aus Metallfolie bestehen oder beidem.
- UTP:
  Kabel mit ungeschirmten Paaren und ohne Gesamtschirm.

Es gibt folgende Kategorien:

- Kategorie 1 (Cat. 1):
  Max. 100 kHz, werden als Telefonkabel verwendet.
- Kategorie 2 (Cat. 2):
  Max. 1,5 MHz, werden z.B. als ISDN-Kabel verwendet.
- Kategorie 3 (Cat. 3):
  Max. 16 MHz, können für 10Mbit/s Ethernet (10BaseT) verwendet werden.
- Kategorie 4 (Cat. 4):
  Max. 20 MBits, werden selten verwendet.
- Kategorie 5 (Cat. 5):
  Max. 100 MHz, Standartkabel für 100MBit/s Ethernet (100BaseT) kann auch für Gigabit Ethernet (1000BaseT) verwendet werden. Cat. 5e ist eine genauere Spezifikation.
- Kategorie 6 (Cat. 6):
  Max. 250 MHz, Cat. 6a 500 MHz, Cat. 6e 600 MHz.
- Kategorie 7 (Cat. 7):
  Max. 600 GHz, verwendbar für 10-Gigabit Ethernet.

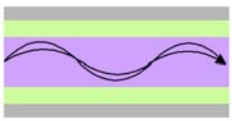
Multimode-Stufenfaser:

Der Glasfaserkern besitzt einen konstanten, aber höheren Brechungsindex als der Mantel, so dass es an der Grenzfläche zu einer Totalreflexion des Lichtes kommt, die diesem Typ ihren charakteristischen „Zich-Zack-Lauf“ des Lichtes verleiht.



Multimode-Gradientenfaser:

Dieser Kabeltyp besitzt einen von der Kernmitte zur Grenzfläche parabelförmig verlaufenden Brechungsindex, so dass sich das Licht sinusförmig im Kern bewegt.



Monomodefaser:

Bei einem Kerndurchmesser von 9 um kann sich das Licht, das eine Wellenlänge von 1,3 um besitzt, nur noch annähernd gradlinig ausbreiten. In diesem Fasertyp sind die qualitätsbeeinträchtigenden Wirkungen des Multimodekonkurrenten bis auf einen ausgeschaltet.

Vergleich der Glasfasertypen:

	Multimode-Stufenfaser	Multimode-Gradientenfaser	Monomodefaser
Bandbreite	200 Mbit/s	400 – 800 Mbit/s	800 Mbit/s
Reichweite	1-2 km	2 – 4 km	> 20 km
Dämpfung	3 – 5 db/km	0,8 – 2,7 db/km	0,5 – 1 db/km

Diese Informationen stammen aus dem Buch Rechnernetze, dass wir in der Schule verwenden. Jedoch stimmen diese angaben nicht da mir Multimode-Gradientenfaser und Monomodefaser eine Bandbreite von 10 GBit/s erreicht werden kann.

Serielle Schnittstelle:

Die Serielle Schnittstelle bezeichnet einen digitalen Eingang und Ausgang eines Computers oder eines Peripheriegerätes. Bei der seriellen Datenübertragung werden die Bits nacheinander über eine einzige Leitung übertragen. Wenn ohne nähere Kennzeichnung von einer „seriellen Schnittstelle“ gesprochen wird, ist damit fast immer die RS-232-Schnittstelle gemeint.

Vorteil:
- Nur wenige Leitungsadern sind notwendig, dadurch geringe Kosten. Relativ große Kabellänge möglich.

Nachteil:
- Ursprünglich niedrige Übertragungsrate. Protokollkenndaten (z. B. Rate, Bytelänge) müssen ggf. vorher vereinbart werden.

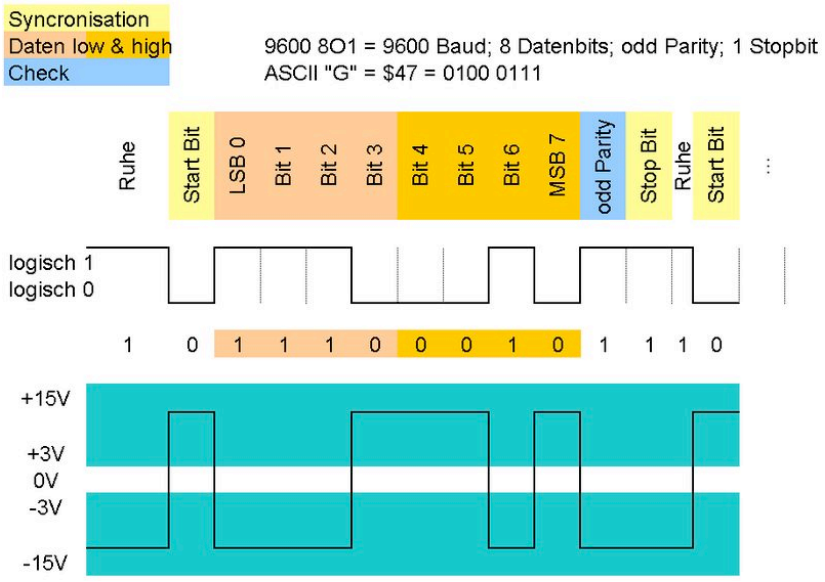
Bsp. für serielle Schnittstellen:
- RS-232
- FireWire
- USB
- S-ATA
- PCI-Express (löst analogen AGP-Port ab)
- DVI
- HDMI

RS-232:

Der Begriff **EIA-232**, ursprünglich **RS-232**, bezeichnet einen Standard für eine serielle Schnittstelle, die in den frühen 1960ern von einem US-amerikanischen Standardisierungskommitee (heute EIA - Electronic Industries Alliance) eingeführt wurde.

Timing:

Das Timingdiagramm zeigt ein Beispiel, wie ein Zeichen übertragen wird. Zunächst liegt der Ruhepegel an. Der Ruhezustand der Übertragungsleitung, der auch mit Mark bezeichnet wird, entspricht dem Pegel einer logischen 1. Als Erstes wird das Startbit logisch 0 (Space) gesendet, um den Empfänger mit dem Sender synchronisieren zu lassen. Die (zeitliche) "Länge" der jeweiligen Bits hängt von der Baudrate ab. Darauf folgen 5…8 Datenbits (Nutzdaten). Angefangen wird mit den LSB (least significant bit) und beendet mit dem MSB (most significant bit). In diesem Beispiel werden 8 Datenbits gesendet. Nun folgt optional ein Parity-Bit, welches zur Erkennung von Übertragungsfehlern dient. Das Paritätsbit bewirkt, dass bei gerader ("EVEN") Parität immer eine gerade bzw. bei ungerader ("ODD") Parität eine ungerade Anzahl von "1"-Bits übertragen wird. Es gibt also die Möglichkeiten E wie even parity oder O wie odd parity oder kein Parity-Bit entsprechend N wie non parity. Abgeschlossen wird die Übertragung mit ein oder zwei Stoppbits logisch "1". Die folgende Ruhezeit darf beliebig lang sein (hier im Beispiel ist sie ein halbes Bit lang).



Paritätsbit:

Die Paritätskontrolle dient der Erkennung fehlerhaft übertragener Informationsworte. Als *Informationswort* wird hier eine Folge von Bits bezeichnet. Die Paritätskontrollcodierung hängt dem *Informationswort* ein *Paritätskontrollbit* (N=1 d.h. die Anzahl der Kontrollbits ist 1), auch Paritybit, genannt an. Das Ergebnis, welches um ein Bit länger ist als das Informationswort, wird hier *Codewort* genannt. Die Methode der Fehlererkennung mittels Paritätsbits heißt *Paritätsprüfung*. Da nicht bekannt ist, wo innerhalb des Codewortes der Fehler aufgetreten ist, ist keine Fehlerkorrektur möglich. Außerdem ist bei einem Paritätsbit N=1 nur eine ungerade Anzahl von Bitfehlern in einem Codewort feststellbar. Eine gerade Anzahl von Bitfehlern wird nicht festgestellt.

Beim Sender werden alle Bits eines Datenblocks (hier Informationswort) modulo N addiert. Entsprechend lassen sich bis zu N Bitfehler erkennen. Für N=1 wird die Summe der Einsen (*Paritätssumme*) im Informationswort berechnet. Ist diese Summe gerade wird bei Even-Parity das Paritätsbit zu Null. Entsprechend ergibt eine ungerade Summe des Informationswortes das Paritätsbit Eins (Odd-Parity).

3.2. Betriebsysteme

Vorgänge beim Laden eines Programms:

Laden des Programmcodes vom externen Speicher

- Datei lokalisieren **Dateiverwaltung**
- EA-Geräte ansprechen und Verwalten **Treiber-Routine**
- In den Arbeitsspeicher **Speicherverwaltung**
- Rechenzeit zuteilen **Prozesswerwaltung**

Komponenten eines OS

Verwaltung EA-Geräte und Routinen (Treiber) 1
Speicherverwaltung
Prozessverwaltung
Dateiverwaltung
Benutzerverwaltung
Verbindungen zu anderen Rechnern (Protokolle)
Auftragsverwaltung
Kommandointerpreter / GUI 3
OS-Kern 1
Dienstprogramme 2

Definition Betriebssystem:

Ein BS ist ein Programm, das elementare Dienste zur Verfügung stellt und die scheinbar gleichzeitige Abarbeitung von mehreren Anwendung organisiert.

API:

Die API (applicant programmers interface) ist eine Schnittstelle für Anwendungsprogramme über die diese die Dinste des BS nutzen können. Die API ist BS spezifisch.

Betriebssystemarten:

- Anzahl der Nutzer
  Multiuser Singeluser
- Anzahl der Anwendungen die gleichzeitig ausgeführt werden können
  Multitasking Singeltasking
- Anzahl der vom BS verwalteten Prozessoren
  Mehrprozessoranlage Einprozessoranlage
- Nach der Antwortzeit auf externe Ereignisse
  Echtzeit-BS Dialog-BS
- Benutzerschnittstelle
  Textorientiert Grafisch
- Art der Programmausführung
  Dialogbetrieb Stapelbetrieb
- Anzahl der gekoppelten Rechner
  Server OS Endbenutzer

3.2.1. Dateiverwaltung

Aufgaben der Dateiverwaltung:

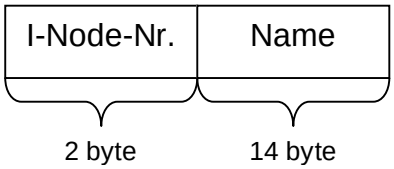
- Dateilisten verwalten (-> Verzeichnisse)
- Lokalisieren der Daten (einer Datei) auf Massenspeichern (-> lesen)
- Freien Speicher verwalten
- Dateien anlegen und löschen
- Zugriffsrechte Kontrollieren u. verwalten

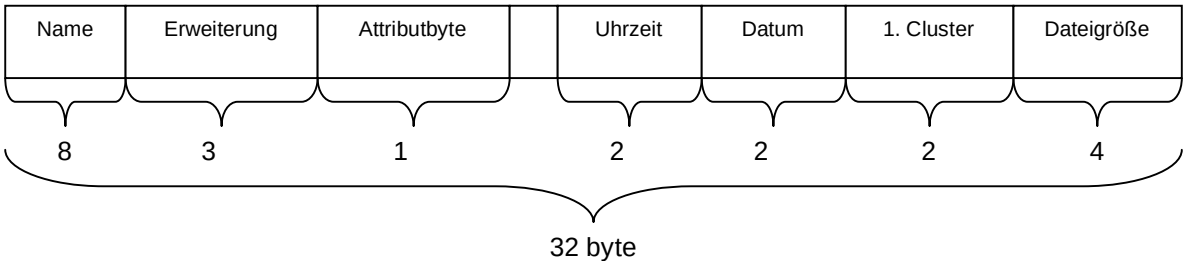
Dateiverwaltung

- zu jeder Datei gibt es einen Deskripter
- die Datei – Deskriptionen werden in Verzeichnissen verwaltet

Verzeichnisse u. Unterverzeichnisse

1. Bsp. Unix (ältere Version)
2. Bsp. MS – DOS





Unterschiedliche Ebenen der Dateiverwaltung

Anwendersicht
- Dateien (unterschiedlicher Typen: Text-, Datendateien und ausführbare Dateien
- Verzeichnisse

Sicht der Anwendungsprogrammierer
- Datensätze in typisierten Dateien : alle Datensätze sind gleich lang (direkter Zugriff auf die Datensätze ist möglich)
- Datensätze in Textdateien: Datensätze unterschiedlicher Länge (nur sequentieller Zugriff möglich)

Logische Sicht der Datenverwaltung
- Datei wird in Datenblöcken gleicher Länge zerlegt
  (die Datensatzlänge != Datenblocklänge)

Physikalische Sicht (an der Schnittstelle zur Festplatte)
- Die Datenblöcke werden in Clustern abgelegt (Blocklänge = Clusterlänge)
- Cluster : festgelegte Anzahl von Sektoren, die nur als Einheit verwaltet werden

Kontinuierliche Allokation

Vorteile:
- hohe Zugriffsgeschwindigkeit auf Datenblöcke einer Datei
- sehr einfache Verwaltung

Nachteile:
- Max. Größe einer Datei muss schon beim Anlegen der Datei festgelegt werden
- u.U. wird Speicherplatz reserviert der nie benötigt wird
- starke Fragmentierung der FP (vor allem nach dem Löschen u. Neuanlegen von Dateien)

Gestreute Allokation

Allokation mittels verketteter Liste. Jeder Datenblock enthält die Clusternummer des Folgeblocks oder die EOF Kennung (End of File).

Vorteile:
- Vollständige Nutzung des Speichers (keine reservierten Bereiche)
- Länge der Datei muss beim anlegen nicht festgelegt werden

Nachteile:
- Speicherplatz für die Verkettungsinformation wird benötigt
- Verlängerung der Ladezeit eine Datei gegenüber kont. Allokation
- Die Datenblöcke einer Datei können nur sequentiell geladen werden. (Direkter Zugriff auf einen Datenblock ist nicht möglich)

[CT] FAT (Gestreute Allokation)

Prinzip:
- Verkettungsinformation aller Dateien wird in einer Tabelle zusammengefasst
- Jedem Cluster ist ein Eintrag in der Tabelle zugeordnet
- Feie (und defekte) Cluster werden durch 0 (FFF7) -> FAT16 gekennzeichnet
- FAT : File Allocation Table

Nachteil:
- Speicherbedarf für die Allocation Tabelle

Vorteile:
- Zugriff auf einen beliebigen Datenblock einer Datei ist möglich (Verwaltungsinformationen aus der FAT auswerten -> schneller zugriff
- Verwaltung der freien und defekten Cluster kann mit Hilfe der FAT erfolgen -> keine zusätzliche Tabellenotwendig

Berechnungen:

KB 2^{10} MB 2^{20} GB 2^{30} TB 2^{40}

Berechnung der Größe einer FAT

FAT 16 pro Eintrag: 2 Byte max. 2^{16} Cluster
$2^{16} * 2^{1} = 131072 / 2^{10} = 128$ KB

FAT 32 pro Eintraf 4 Byte max. 2^{32} Cluster
$2^{32} * 2^{2} = 6{,}8*10^{10} / 2^{30} = 16$ GB

FP mit 20 GB, FAT 32, Cluster: 4KB
Anzahl der Cluster = 20GB / 4KB = $20 * 2^{30}$ B / $4 * 2^{10}$ B = $5 * 2^{20}$
Tabellengröße = $5* 2^{20} * 4$ B = 20MB

Zugriff auf die einzelnen Cluster einer Datei mittels Index-Knoten (i-node)

Prinzip:

- Jede Datei besitzt einen I-node der die Clusternummer der Cluster dieser Datei enthält.
- Bei größeren Dateien verweist ein (reservierter) Eintrag auf einen weiteren I-node, der die restlichen Clusternummern dieser Datei enthält (-> einfach indizierter Block)
- Bei großen und sehr großen Dateien können zwei und dreifach indizierte Blöcke auftreten.
- Im I-node sind die Dateiattribute gespeichert.

Vorteile:

- Beliebige Cluster einer Datei kann über die Datenblocknummer (-> index in der Tabelle des I-node) festgestellt werden. Im Gegensatz zu FAT: hier muss die verkettete Liste durchgegangen -> Schneller Zugriff aud die einzelnen Datenblöcke
- Flexibel

Nachteile:

- I-Nodes Zahl ist gegrenzt
- Wenn alle I-Nodes belegt sind können freie Cluster nicht mehr verwendet werden

[CT only] Zugriffsrechte

Zugriffsrechte Linux;

r	read	für Dateien
w	write	
x	execute	
r	Leserecht für Dateiliste	für Verzeichnisse
w	Verzeichniseinträge ändern (umbenennen, hinzufügen, löschen von Verzeichniseinträgen)	
r	Zutritt zum Verzeichnis erteilen bzw. entziehen (Das Recht den Verzeichnisnamen im Pfad nennen zu dürfen)	

Benutzerkategorien
- Eigentümer (user)
- Gruppe (group)
- alle Benutzer (others)

Zugriffsrechte ändern
chmod Rechte Dateiname(n)

```
|-> absolut z.B.  u=rw,go=r      entspricht rw-r--r--
|                 u=rx           entspricht r-x??????
|                 o=             entspricht ??????---
|-> relativ z.B.  u+rw,go+r      entspricht rw?r??r??
|                 o-r            others wird r entzogen
|        (Nur die genannten Rechte werden geändert)
|-> oktal z.B.    644            entspricht rw-r--r--
|                                110 100 100
```

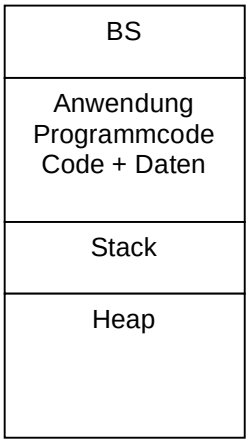
3.2.2. Speicherverwaltung

Speicherverwaltung:

Aufgaben:
- freien Speicher verwalten
- zuteilen von Speicher
- Zugriffsrechte verwalten (Speicherschutz)
- auslagern von Speichersegmenten auf den externen Speicher (Swapping)

Direkte Adressierung des Arbeitsspeichers:

- Gesamter Speicher wird als ein Block verwaltet
- Programmgröße ist durch die Speichergröße (physikalischer Speicher) begrenzt
- Beim Laden des Programms müssen die absoluten Adressen (Sprünge, UP-Aufrufe) neu berechnet werden (->Linkloader)



Virtueller Speicher:

virtueller Speicher (log. Adressen) -> physikalischer Speicher (Arbeitsspeicher phy. Adresse) + externer Speicher (Festplatte)

Der virtuelle Speicherraum wird auf dem Arbeitspeicher und dem externen Speicher abgebildet.
- Programmgröße ist nicht auf die Größe des realen Speichers begrenzt
- Nur Speicherzellen die in physikalischen Speicher präsent sind können angesprochen werden (von der CPU)
- Die Prozessorhardware und das BS organisieren die Adressrechnung und das Ein – und Auslagern von Speicherbereichen

Segmentadressierung:

Prinzip: - Aufteilen des logischen Adressraums in Segmente variabler Länge.
- Jeder Prozess bekommt seine eigenen Programm-, Daten- und Stacksegmente.
- Jedes Segment hat einen Adressraum, der mit der Adresse 0 beginnt.
  -> Programmcode kann an eine beliebige Stelle im physikalischen Speicher geladen
  werden. (Das Programm arbeitet nur den Offset-Adressen innerhalb des Segments)
- Speicherschutz z.B. bei Intel-Prozessoren
  - ➔ Programmcode nur aus CS-Segmenten
  - ➔ Daten nur aus DS-Segmenten
  - ➔ Speicherschutz kann gesetzt werden und durch Privilegstufen können Speichersegmente des BS vor Zugriffen aus Anwendungsprogrammen geschützt werden.
- Segmente (auch sehr große) können nur als Ganzes ausgelagert werden.
  -> u.U. erheblicher Zeitbedarf

Jeder Tabelleneintrag besitzt zusätzlich ein present-Bit, das anzeigt, ob das Segment im Arbeitsspeicher präsent ist oder ausgelagert wurde.

Seitenadressierung:

Prinzip: Der durch die Adressregisterbreite aufgespannte Adressraum wird in Seiten gleicher Größe (z.B. 2KB (0-7FFh) oder 4KB (0-FFFh)) aufgeteilt.

Adressraum
- lineare Adresse
- virtueller Speicher

realer Speicher (Arbeitsspeicher)
- in gleich große Bereiche (2KB … 4KB) aufgeteilt: Seitenrahmen
- physikalische Adresse

- Nur die Seiten des virtuellen Speichers, auf die gerade zugegriffen wird, müssen im realen Speicher vorhanden sein.
- Die Zuordnung der logischen Adresse (lineare Adresse) zur physikalischen Adresse erfolgt über die Seitentabelle
- In der Seitentabelle steht auch, ob die Seite im Arbeitsspeicher vorhanden ist. Falls nicht vorhanden, muss eine geladene Seite ausgelagert und diese geladen werden.

Jeder Tabelleneintrag besitzt zusätzlich ein in/out-Bit, das anzeigt, ob die Seite im RAM liegt (in einen Seitenrahmen „eingehängt“ ist) oder auf den externen Speicher verschoben wurde.

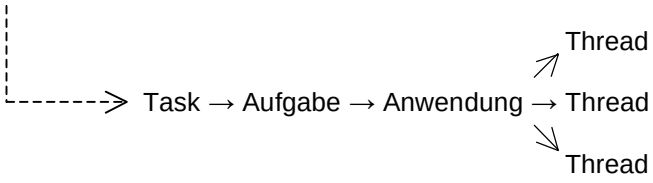
Vorteile gegenüber der Segmentadressierung
- Die Speicherbereiche, die ausgelagert bzw aus dem externen Speicher geladen werden, sind bei der Seitenadressierung alle gleich groß und relativ klein (2KB … 4KB)

Nachteile gegenüber der Segmentadressierung
- Der Speicherbereich einer Seite kann nur als Ganzes vergeben werden, auch wenn nur einige Byte benötigt werden
- Speicherschutz ist nicht so umfangreich wie bei der Segmentadressierung möglich.

3.2.3. Prozessverwaltung

Prozessverwaltung:

Multiuser-BS — Mehrere Anwendungen müssen *gleichzeitig* abgearbeitet werden.
Multitask-BS — *Gleichzeitig*: Für einen externen Beobachter (intern zyklisch sequentiell)



Bsp. Für Prozesse:

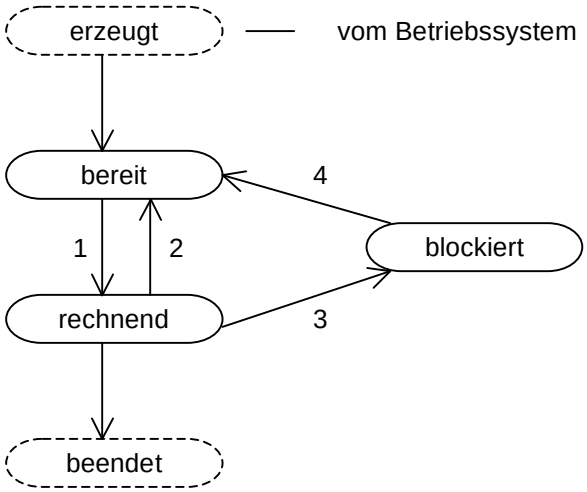
- Druckerspooler
- Word
- Explorer
- C++-Builder
  - Compiler
  - Linker
  - Testumgebung (Debugger)

} Kindprozesse

Ziele der Prozessverwaltung:

- Reaktionszeit für alle Benutzer möglichst kurz.
- Durchsatz (Effizienz) möglichst groß

Prozesszustände:



1. Dispatcher (Scheduler):
   Teilt dem Prozess den Prozessor (Rechenzeit) zu.
2. Der Prozess gibt den Prozessor wieder frei bzw. der Prozessor wird ihm entzogen.
3. Der Prozess gibt den Prozessor frei und geht in den Zustand „blockiert“ da er auf ein Ereignis wartet (Tastatureingabe, auf Festplatte…).
4. Ereignis ist eingetreten.

Prozesswechselmethoden:

1. Prozess behält den Prozessor bis er
   - terminiert
   - auf ein Ereignis wartet
   - durch einen Prozess mit höherer Priorität verdrängt wird

   } System für Stapelverwaltung

2. Prozess gibt den Prozessor „freiwillig“ und nach einer angemessenen Zeit frei
   => kooperatives Multitasking

3. Prozess bekommt den Prozessor nur für eine vorgegebene Zeit (Zeitscheibe, Quantum) zugeteilt, danach wird ihm der Prozessor entzogen
   => präemptives Multitasking

Vor –und Nachteile

Kooperatives Multitasking
- V: Relativ einfach zu implementieren.
  Unterbrechungen in kritischen Abschnitten (Zugriff auf gemeinsame Ressourcen) können vermieden werden.
- N: Nicht kooperative Prozesse bzw. Programme die in Schleifen „hängen bleiben“ (Programmierfehler) können das System lahm legen.

Präemptives Multitasking
- V: System kann durch nicht kooperative Prozesse und Programmierfehler in den Anwendungsprogrammen nicht blockiert werden.
- N: Implementierung aufwendiger.
  Prozess kann in einem kritischen Abschnitt unterbrochen werden.

Arbeitsumgebung für Prozesse:

Damit ein Prozess abgearbeitet werden kann, wird benötigt:

Programmcode im Arbeitsspeicher
Daten im Arbeitsspeicher
Stapelspeicher z.B. für Rücksprungadressen

Stapelzeiger (Prozessor-Register)
Datenregister
Programmzähler

} CPU-Register

} Wird für jeden Prozess exklusiv benötigt
=> wird im Prozessleitblock gespeichert

Registerinhalte werden bei einem Prozesswechsel im Prozessleitblock abgelegt und bei Zuteilung von Rechen zeit aus diesem wieder hergestellt.
Jeder Prozess besitzt einen eigenen Prozessleitblock dieser enthält zusätzlich:
- Den Namen des Prozesses
- Prozesszustand (falls blockiert: Ereignis auf das gewartet wird)
- Priorität des Prozesses

Die Prozessleitblöcke werden in Listen/Warteschlangen verwaltet. Es gibt mindestens eine BEREIT-Liste und BLOCKIERT-Liste.

Prozess/Thread:

Thread: Faden, Ablaufpfad

Prozesse bestehen aus:

- Programmcode im Speicher
- Daten im Speicher

} Kann von allen Threads einer Anwendung gemeinsam genutzt werden

- Stapelspeicher
- Prozessleitblock

} Für jeden Thread exklusiv notwendig. (Prozessleitblock → Threadleitblock)

Vorteile von Threads gegenüber Prozessen:

- Eine Anwendung kann (problemlos) auf mehrere Prozessoren verteilt werden.
- Nebenläufige „Prozesse“ (Programmteile die weitestgehend unabhängig voneinander sind, aber zyklisch bearbeitet werden müssen) innerhalb von einer Anwendung können einfacher implementiert werden.

Scheduling-Algorithmen:

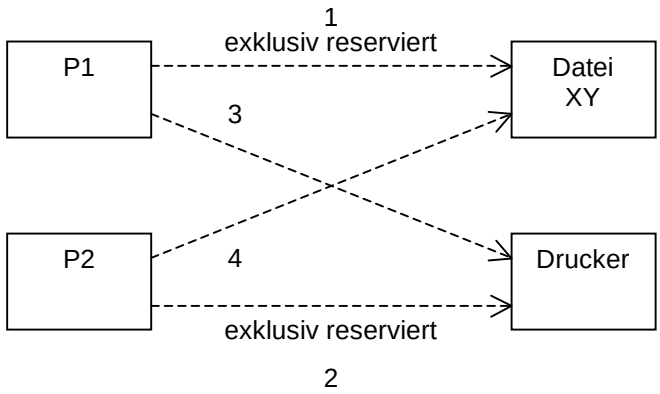
- First-Come-First-Serve *km*
  Wer zuerst kommt, wird zuerst bedient.
  Gute Systemauslastung, aber schlechtes Antwortzeitverhalten (lang laufende Prozesse behindern Kurzläufer).
- Shortest-Job-First *km*
  Kürzester Prozess zuerst.
  Langer Prozess läuft nie, wenn von der CPU-Zuteilung ein kürzerer Prozess kommt.
- Round Robin *km, pm*
  Jeder Prozess erhält eine feste Zeitspanne. Nach Ablauf dieser Zeitspanne wird er verdrängt und der nächste Prozess erhält die CPU => zyklische Zuteilung.
  Prozesse haben immer die gleiche Priorität.
  Antwortzeit abhängig durch Einteilung der Zeitscheibe.
- Prioryty Scheduling (Prioritätsssteuerung) *km, pm*
  Jedem Prozess wird eine Priorität zugeordnet. Vergabe an die CPU in absteigender Priorität.
  Niedriger Prioritätsprozess erhält die CPU erst dann, wenn alle höheren Prozesse abgearbeitet sind. Ein bereit werdender Prozess höherer Priorität verdrängt einen aktiven Prozess niedriger Priorität.
  Bei Prozessen mit gleicher Priorität:
  - Reine Priorität: Abarbeitung nach der Eingangsreihenfolge (z.B. in Echtzeit-BS).
  - Priorität mit unterlagertem Zeitscheibenverfahren: Prozesse werden nach dem Zeitscheibenverfahren abgearbeitet.
  - Dynamische Prioritätsvergabe: Die Priorität auf die CPU wartender Prozesse wird allmählich erhöht.
  - Mehrstufiges Herabsetzen (multilevel feedback): Jede Prioritätsstufe hat festgelegte max. Rechenzeit. Hat ein Prozess die max. Rechenzeit seiner Priorität verbraucht, bekommt er die nächst niedrigere Priorität, bis er die niedrigste Stufe erreicht hat.

Verfahren um das „Verhungern“ von Prozessen mit niedriger Priorität zu verhindern.

km: Kooperatives Multitasking
pm: präemptives Multitasking

Verklemmung (Deadlock):

Bsp. für eine Verklemmung:



3: P1 will den Drucker reservieren, da belegt: P1 wartet bis der Drucker freigegeben wird.

4: P2 will Datei XY öffnen, da belegt: P2 wartet bis die Datei freigegeben ist.

=> Verklemmung

Ein Deadlock tritt auf wenn zwei oder mehrer Prozesse auf Betriebsmittel (Ressourcen) warten die von einem anderen Prozess exklusiv reserviert sind und dieser selbst auf Ressourcen des wartenden Prozess wartet.

Begriffserklärung (Wikipedia):

Als Prozess wird in der Informatik der Ablauf eines Programms bezeichnet.

Ein Thread (auch Aktivitätsträger), in der deutschen Literatur vereinzelt auch als Faden bezeichnet, ist in der Informatik ein Ausführungsstrang beziehungsweise eine Ausführungsreihenfolge der Abarbeitung der Software.

Im Rahmen der Prozessverwaltung eines Betriebssystems dient der Dispatcher dazu, bei einem Prozesswechsel dem derzeit aktiven Prozess die CPU zu entziehen und anschließend dem nächsten Prozess die CPU zuzuteilen. Die Entscheidung, welcher Prozess der *nächste* ist, wird vom Scheduler im Rahmen der Warteschlangenorganisation getroffen.

Ein Prozess-Scheduler regelt die zeitliche Ausführung mehrerer Prozesse in Betriebssystemen.

3.3. Datensicherung

Datensicherung:

Sicherungsverfahren

1. Vollsicherung (Normal)
   Alle Daten – unabhängig vom Archivbit – auf dem Backup-Medium gesichert.
   Nach der Durchführung der Sicherung wird das Archivbit zurückgesetzt. Die Daten werden dadurch als gesichert gekennzeichnet.
   Wiederherstellung: Die letzte Vollsicherung wird benötigt.
2. Differentielles Backup (Differenz)
   Gespeichert werden die Daten, die seit der letzten Vollsicherung erstellt oder verändert wurden. Nach der Durchführung der Sicherung bleibt das Archivbit gesetzt. Die Daten werden bei der nächsten Sicherung wieder gespeichert.
   Wiederherstellung: Die letzte Vollsicherung und das letzte differentielle Backup werden benötigt.
3. Inkrementelles Backup (Hinzufügen, auch Zuwachssicherung)
   Die Daten, die seit dem letzten Backup erstellt oder geändert wurden, werden gespeichert. Das Archivbit der gesicherten Daten wird zurückgesetzt.
   Wiederherstellung: Die letzte Vollsicherung und alle Zuwachssicherungen werden benötigt.
4. Kopieren:
   Alle ausgewählten Dateien werden kopiert. Die Dateien werden aber im Gegensatz zur Vollsicherung nicht als gesichert gekennzeichnet (Archivbit wird nicht zurückgesetzt).
   Diese Sicherungsart bietet sich an, wenn zusätzlich zum regulären Sicherungszyklus eine zusätzliche Kopie hergestellt werden soll (z.B. ein monatliches Archivband).

Großvater-Vater-Sohn Datensicherung:

Eine *Großvater-Vater-Sohn Datensicherung* (auch: Generationenprinzip) ist ein altbekanntes Verfahren zur Datensicherung. Dabei wird von dem Datenbestand ständig ein dreifaches Backup verschiedenen Alters (Großvater, Vater, Sohn) von einem Datenträger gemacht. Veränderungen und Verluste der Daten können somit rekonstruiert werden. Sind die „Sohn“-Daten beschädigt, werden sie aus den „Vater“-Daten wieder erzeugt und die „Vater“-Daten gegebenenfalls aus den „Großvater“-Daten.

Bsp.:

1. Söhne:
   Am Abend der Wochentage Mo, Di, Mi und Do wird jeweils ein differentielles Backup durchgeführt.
   Die Beschriftung der Bänder lautet: Montag, Dienstag, Mittwoch, Donnerstag.
   Diese Bänder sind die 3 „Söhne“.

2. Väter:
   Am 1. 2. und 3. Freitag eines Monats wird jeweils ein Vollbackup durchgeführt.
   Beschriftung: 1. Woche, 2. Woche, 3. Woche
   Diese Bänder sind die 3 „Väter“.

3. Großväter:
   Immer am letzten Freitag eines Monats wird ein Vollbackup durchgeführt.
   Beschriftung: Januar, Februar, …Dezember.
   Diese Bänder sind die 12. „Großväter“.

Mit insgesamt 19 Bänder lässt sich über den Zeitraum eines Jahres folgende Wiederherstellungsgenauigkeit erreichen:

Aktuelle Woche:	tagesgenau
Letzter Monat:	Stand der jeweiligen Freitage
Letztes Jahr:	Stand des letzten Freitags im Monat

3.4. [CT] DBMS

Begriffe/Definitionen:

Datenbank: Sammlung von Daten.

Datenbanksystem: Daten + Software zum verwalten der Daten.

Redundanz: Dasselbe „Datenelement“ ist mehrfach gespeichert.

Logische Integrität: Keine Widersprüche in den abgespeicherten Datensätzen.

Referientielle Integrität: Integrität auf Beziehungsebene / Eindeutigkeit der Schlüssel.

Konsistenzprüfung: Z.B. bei der Eingabe von Daten wird geprüft ob diese die Intigrität der DB verletzen.

Transaktion: Anweisung die entweder ganz oder gar nicht ausgeführt werden darf.
➔ Schon ausgeführte Änderungen müssen rückgänig gemacht werden.

Entität: Individuelles, dentitizierbares Exemplar von Dingen, Personen oder Begriffen der realen oder der Vorstellungswelt (siehe Objekt bei OOA/OOD). Die einzelnen Entitäten werden durch ihre Attributwerte beschrieben bzw. unterscheinden. Entitäten entsprechen im relationalen Dantenbankmodell den Zeilen einer Tabelle.

Entitätsmenge (Entitätstyp): Enthält Entitäten mit den gleichen Eigenschaften (Attributen), die unter einem gemeinsamen Oberbegriff zusammgefasst werden können. Entitätstypen werden bei der Implementierung eines relationalen Datenbankmodells in Tabellen umgesetzt.

Entitätsmenge, Entitätstyp -> Menge
Entiät -> Ein Element dieser Menge

Bsp.:

Entitätstypen	Entitäten
Sudent	(27385, Hans Maier, 15.10.2004,…)
Fakultät	(27, Informatik, Bauer,…)

Attribut: Attribute entsprechen im relationalen Datenbankmodell den Spalten der Tabelle (Spalten enthalten Attributwerte).

Es gibt:
- Beschreibende Attribute (z.B. Name: Maier)
- Indentifizierende Attribute (z.B. MartikerlNr)

Mehrere Daten können auch in einem Attribut zusammengefasst werden (z.B. Attributwert: „75175 Pforzheim Graf-Leutrum-Str. 3“). Aber Attribute sind für das DBMS stets atomar d.h. Daten können in dem oben genannten Beispiel nicht nach dem Ort sortiert werden.

Hierarchisches Datenbankmodell:

Das hierarchische Datenbankmodell ist wie die Verzeichnisstruktur in einem Computer aufgebaut.

Vorteile:
- Sehr kurze Zugriffszeit.

Nachteile:
- Nicht in der hierarchischen Struktur vorgesehene Auswertungen können lange dauern.
- Komplexe Programmierung, da die hierarchische Struktur beachtet werden muss.
- Evtl. Redundanz zur Effizienzsteigerung möglich.

Netzwerkartige Datenbanken:

Vorteile:
- Kurze Zugriffszeit.
- Geringe Redundanz.

Nachteile:
- Komplexe Programmierung.
- Änderungen an der Struktur kaum möglich.

Relationales Datenbankmodell:

Vorteile:
- Leichter Änderbarkeit der logischen Struktur.
- Programmierung von Abfragen einfacher als bei netzwerkartigen Datenbanksystemen und standardisiert (SQL).

Nachteile:
- Viele Ein-/Ausgabeoperation (-> externe Datenträger) notwendig.
- Höhere Rechenleistung zu Verknüpfung von Datensätzen notwendig.

Objektorientirtes bzw. objekrelationales Datenbankmodell:

Vorteile:
- Objektorientierter Aufbau (Vererbung, Kapselung).

Nachteile:
- Noch mehr Rechenleistung als bei relationalen Datenbanksystemen notewendig.

Schlüsselkanidaten:

Attribute, bei dem jede Entität einen anderen Attributwert besitzt. (Es können auch mehrere Attribute zusammengefasst werden)

Primärschlüssel:

Aus diesen Schlüsselkanidaten wird einer als Primärschlüsseln ausgewählt.

Fremdschlüssel:

Fremdschlüssel dienen dazu um Beziehungen zwischen zwei Entitäten herzustellen. Wenn der Primärschlüssel (FNr) eines Enititätstypen (z.B. aus Fakultät) als Attribut in einen anderen Entitätstypen (z.B. Studen) eingefügt wird, dann ist dieser (FNr in Student) ein Fremdschlüssel.

Kardinalität:

Kennzeichnung	Bedeutung
1	Genau 1 Entität zugeordnet
C	0 oder 1 Entität zugeordnet
N (M)	Viele Entitäten zugeordnet
NC (MC)	0, 1 oder viele Entitäten zugeordnet

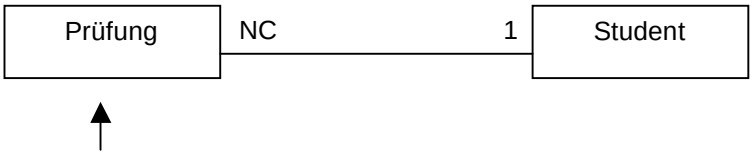
Relationsschreibweise:

Die Relationsschreibweise bildet, die Struktur eines DBS, in kompakter Schreibweise ab.

- Für jeden Entitätstyp wird eine Relation erstellt.
  Notation: Entitätstypbezeichner(Attribut1, Attribut2, …, AtributN)
- Primärschlüssel warden durch unterstreichen gekennzeichnet.
- Fremdschlüssel werden gestrichelt unterstrichen.
  Voraussetzung: N:M–Beziehungen müssen aufgelöst sein!

Bsp.: Uni Verwaltung
Fakultät(FNr, FName, Dekan)
Student(MatNr, Name, Studienbeginn, FNr)
Professor(PNr, PName, Fachgebiet, FNr)
Prüfung(PruefNr, Fach, Datum, Note, PNr, MatNr)

Der Fremdschlüssel darf nur bei einer der beiden Entitätstypen, die in Beziehung zueinander gesetzt werden, eingesetzt werden.
Bei einer N(C):1-Beziehung muss der Fremdschlüssel bei der Relation (dem Entitätstypen) eingesetzt werden, bei dem die Kardinalität 1 am entfernten Ende (bei dem gegenüberliegenden Entitätstyp der Beziehungslinie) liegt.



Fremdschlüssel (=Primärschlüssel von Student) hier einfügen.
=> Prüfung(PruefNr, Fach, Datum, Note, PNr, MatNr)

ER-Diagramm, Relationsschreibweise und Implementierung:

„Element“	ER-Diagramm	Relationsschreibweise	Implementierung
Entitätstyp	Rechteck mit Bezeichner des Typen (Substaniv, Singular)	Bezeichnung der Relation	Tabelle
Entität	-	-	Zeile (Datensatz)
Attribute	Ellipse an den Entitätstypen	Liste der Attribute in Klammer nach dem Relationsbezeichner	Spalte in der Tabelle
Beziehungstyp	Linien zwischen Entitätstypen mit Kardinalität evtl. Beschreibung (Verb in einer Raute)	Durch die Kombination Primärschlüssel Fremdschlüssel abbilden	Beziehung zwischen Attributen

Normalisierung:

1. Normalform (1. NF):

   - Attribute elementar (atomar)
     Zum Beispiel muss das Attribut Adresse in Straße, Ort, … unterteilt werden.
   - Keine Wiederholungsgruppen
     Bsp. für eine Wiederholungsgruppe: Telefon1, Telefon2, Telefon3
   - Primärschlüssel

2. Normalform (2. NF):
   - 1. NF
   - Volle funktionale Abhängigkeit der Nicht-Schlüsselattribute vom Primärschlüssel.

3. Noramlform (3. NF):
   - 2. NF
   - Alle Nicht-Schlüsselattribute müssen direkt vom Primärschlüssel abhängig sein (nicht nur indirekt bzw. transitiv)
     Bsp.: In einer Tabelle wird zu einer CD das Gründungsjahr der Band gespeichert, dies kann aber zu Redundanz führen, da es sein kann, dass mehrere CDs einer Band vorhanden sind, somit wird das Gründungsdatum mehrmals gespeichert (Redundanz). Die Lösung ist eine extra Tabelle zu erstellen in der, der Name der Band der PKey ist und das Gründungsjahr ein Attribut ist.

Das wichtigste:
- Keine N:M Beziehungen
- Keine Wiederholungsgruppen
- Redundanz minimieren

Anomalien:

Eine Anomalie tritt auf wenn die Integrität einer DB verletzt wied.

Arten von Anomalien:
- Lösch-Anomalie:
  Eine Lösch-Anomalie tritt auf, wenn beim löschen eines Datensatzes andere davon abhängige Datensätze nicht mit gelöscht werden.
- Einfüge-Anomalie:
  Einfügen falscher Datensätze. Bsp.: Ein Datensatz wird doppelt eingefügt.
- Änderungs-Anomalie:
  Wenn beim Ändern redundanter Datensätze, nicht alle Datensätze geändert werden.

Grundgerüst einer SQL-Abfrage:

```
SELECT      [DISTINCT] ausdruck
FROM        from_element
WHERE       bedinung
GROUP BY    ausdruck
HAVING      bedinung
{UNION}
ORDER BY    ausdruck [DESC];
```

Grundgerüst einer SQL-Abfrage 2:

SELECT:
Der Ausdruck hinter SELECT bestimmt, welche Spalten der Quelle auszugeben sind (* für alle) und ob Aggregatfunktionen anzuwenden sind. Wenn DISTINCT angegeben ist, werden doppelte Zeilen aus der Ergebnismenge entfernt.

FROM:
Die FROM-Klausel gibt eine oder mehrere Queltabellen an.

WHERE:
Jede Zeile die die Bedinung nicht erfüllt wird nicht ausgegeben.

GROUP BY:
GROUP BY fasst alle Zeilen, die den gleichen Wert für die Gruppierungsausdrücke haben, zu jeweils einer Zeile zusammen.

HAVING:
HAVING entfernt Gruppenzeilen, die die Bedinung nicht erfüllen.

ORDER BY:
Die ORDER BY-Klausel sortiert die Ergebniszeilen anhand der angegebenen Ausdrücke. Wird DESC angegeben werden die Ergebniszeilen in abfallender Reihenfolge ausgegeben.

LIKE:

LIKE ist ein weiteres Schlüsselwort, das in Bedinungen (WHERE, HAVING) Verwendung findet. Im Wesentlichen ermöglicht LIKE eine Suche auf der Grundlage eines Musters an Stelle einer genauen Angabe.

Bsp.:
```
SELECT Nachname, Vorname
FROM  Personal
WHERE Nachname LIKE „D*“ OR Vorname LIKE „An*“;
```

Aggregatfunktionen:

Es gibt folgende Aggregatfunktionen:

COUNT ([DISTINCT]expr)	Ermittelt die Anzahl der Datensätze
AVG (spalte)	Durchschnitt
MAX (expr)	Maximalwert
MIN (expr)	Minimalwert
SUM (expr)	Summe

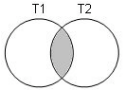
Werden die Datensätze mit GROUP BY gruppiert und soll darauf dann eine Aggregatfunktion angewendet werden, dann müssen die Spalten die keine Aggregationsfunktion enthalten im Ausdruck der GROUP BY-Klausel stehen. Außerdem kann die WHERE-Klausel nicht verwendet werden um Gruppen einzuschränken. Dies ist nur mir der HAVING-Klausel möglich.

Bsp.:
```
SELECT kundennummer, name, AVG(umsatz) AS [Durchschnittlicher Umsatz]
FROM Kunden
GROUP BY kundennummer, name
HAVING AVG(umsatz) > 10000;
```

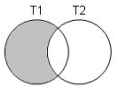
Verbund (Join):

Ein Verbund verknüpft Tabellen miteinander.
Es gibt folgende Joins:

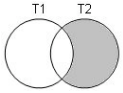
- INNER JOIN



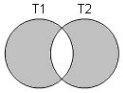
- LEFT JOIN



- RIGHT JOIN



- OUTER JOIN



Bsp.: SELECT DISTINCT L.Firma, L.Land, L.PLZ, L.Ort
FROM (Lieferanten L INNER JOIN Artikel A ON L.[Lieferanten-Nr] = A.[Lieferanten-Nr])
INNER JOIN Kategorie K ON A.[Kategorie-Nr] = K.[Kategorie-Nr]
WHERE K.Kategoriename = „Gewürze“
ORDER BY 1;

UNION:

Mit UNION können mehrere Tabellen oder Abfragen kombiniert werden.
Bedinungen, Besonderheiten:

- Die Spalten der einzelnen Hauptteile müssen miteinander verträglich sein (z.B. nicht Zeichenkette versus Zahlenwert).
- Mehrfache Datensätze werden automatisch entfernt (DISTINCT als Standardeinstellung). Dies kann umgangen werden mit „…UNION ALL…“. Dann werden gleiche Datensätze (Tupel) nicht entfernt.

SELECT Name, Anrede
FROM Kunden
UNION
SELECT Firmenname AS Name, Firmenbez. AS Anrede
FROM Lieferanten;

4. [CT] Kryptologie

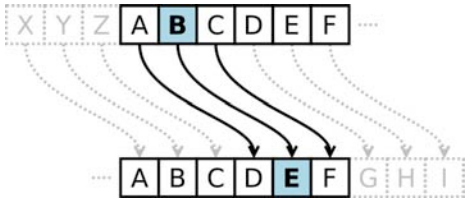
Inhalt:

- Monoalphabetische Verschlüsselung
- Polyalphabetische Verschlüsselung
- Kryptoanalyse
- Symmetrische Kryptosysteme
- Asymmetrische Kryptosysteme
- Hybride Verfahren

Monoalphabetische Verschlüsselung:

Die monoalphabetische oder einfache Substitution ist ein Verschlüsselungsverfahren, bei dem jeder Buchstabe oder jedes Zeichen durch ein anderes Zeichen desselben Alphabets ersetzt wird.

Bsp. Caesar:
Hierbei gibt es ein Klartext-Alphabet und ein Ciphertext-Alphabet (Gehiemtext-Alphabet). Das Ciphertext-Alphabet wird zum Klartext-Alphabet um eine bestimmte Verschiebezahl verschoben, diese Verschiebezahl stellt den Schlüssel dar.



Die einfache monoalphabetische Verschlüsselung ist wesentlich sicherer als die Cäsarchiffre, da es nicht nur 25 Möglichkeiten, sondern 26! unterschiedliche Alphabete gibt, das sind ungefähr 4·10^26 Möglichkeiten.

Bsp. Atbash (nur ein Schlüssel):
```
Klartextalphabet:   a b c d e f g h i j k l m n o p q r s t u v w x y z
Geheimtextalphabet: Z Y X W V U T S R Q P O N M L K J I H G F E D C B A
```

Polyalphabetische Verschlüsselung:

Polyalphabetische Ersetzungschiffren bezeichnen in der Kryptographie Formen der Textverschlüsselung, bei der einem Buchstaben/Zeichen jeweils ein anderer Buchstabe/Zeichen zugeordnet wird. Im Gegensatz zur monoalphabetischen Substitution werden für die Zeichen des Klartextes mehrere Geheimtextalphabete verwendet.

Bsp. Vigenère-Tafel:
```
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
B C D E F G H I J K L M N O P Q R S T U V W X Y Z A
C D E F G H I J K L M N O P Q R S T U V W X Y Z A B
D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
E F G H I J K L M N O P Q R S T U V W X Y Z A B C D
F G H I J K L M N O P Q R S T U V W X Y Z A B C D E
G H I J K L M N O P Q R S T U V W X Y Z A B C D E F
H I J K L M N O P Q R S T U V W X Y Z A B C D E F G
I J K L M N O P Q R S T U V W X Y Z A B C D E F G H
J K L M N O P Q R S T U V W X Y Z A B C D E F G H I
K L M N O P Q R S T U V W X Y Z A B C D E F G H I J
L M N O P Q R S T U V W X Y Z A B C D E F G H I J K
M N O P Q R S T U V W X Y Z A B C D E F G H I J K L
N O P Q R S T U V W X Y Z A B C D E F G H I J K L M
O P Q R S T U V W X Y Z A B C D E F G H I J K L M N
P Q R S T U V W X Y Z A B C D E F G H I J K L M N O
Q R S T U V W X Y Z A B C D E F G H I J K L M N O P
R S T U V W X Y Z A B C D E F G H I J K L M N O P Q
S T U V W X Y Z A B C D E F G H I J K L M N O P Q R
T U V W X Y Z A B C D E F G H I J K L M N O P Q R S
U V W X Y Z A B C D E F G H I J K L M N O P Q R S T
V W X Y Z A B C D E F G H I J K L M N O P Q R S T U
W X Y Z A B C D E F G H I J K L M N O P Q R S T U V
X Y Z A B C D E F G H I J K L M N O P Q R S T U V W
Y Z A B C D E F G H I J K L M N O P Q R S T U V W X
Z A B C D E F G H I J K L M N O P Q R S T U V W X Y
```

```
Schlüssel (TEST):  TESTTESTTESTT
Klar-Text:         DIESISTGEHEIM
Geheim-Text:       WMWLBWLZXLWBF
```

Kryptoanalyse:

Kryptoanalyse bei monoalphabetischen Verschlüsselungen:

- Häufigkeitsanalyse:
  Hierbei wird analysiert wie häufig bestimmte Buchstaben im Geheimtext vorkommen, das Ergebnis wird dann mit der Häufigkeitstabelle einer Sprache verglichen.
- Klartextangriff:
  Sind einem Teile des Klartextes bekannt (einzelne Begriffe), so kann man nach diesem Muster im Geheimtext suchen, indem man beispielsweise nach Doppelbuchstaben Ausschau hält. Im Klar- sowie im Geheimtext sollten bei einer monoalphabetischen Substitution an denselben Stellen doppelte Zeichen vorkommen. In gleicher Weise kann man auch nach Mustern im Geheimtext suchen, die dem Muster des vermuteten Wortes entsprechen.

Kryptoanalyse bei polyalphabetischen Verschlüsselungen:
Bei der Kryptoanalyse von polyalphabetischen Verschlüsselungen muss man zunächst die Schlüssellänge bestimmen dies kann man mit Kasiski Test bewerkstelligen.

Kasiski Test:
Treten zwei Folgen gleicher Buchstaben (z.B. 'ein') im Klartaxt auf, so werden sie im Allgemeinen im Geheimtext verschieden ausfallen. Werden die Anfangsbuchstaben der beiden Folgen gleich verschlüsselt, sind sie auch im Geheimtext gleich, ebenso die nachfolgenden Buchstaben. Dies ist genau dann der Fall, wenn der Abstand zwischen den Buchstaben genau der Schlüssellänge entspricht.
Findet man nun gleiche Folgen im Geheimtext, geht man davon aus , daß deren Abstand wahrscheinlich (je länger die Folgen desto sicherer) ein Vielfaches der Schlüssellänge ist.

Hat man die Schlüssellänge bestimmt teilt man den Text in Gruppen auf, jeder Geheimtextbuchstabe der mit dem gleichen Schlüsselposition verschlüsselt wurde kommt in eine Gruppe. Da die Buchstaben in einer Gruppe mit dem gleichen Alphabet verschlüsselt wurden kann man auf die Gruppen die Häufigkeitsanalyse anwenden.

Symmetrische Kryptosysteme:

Ein symmetrisches Kryptosystem ist ein Kryptosystem, welches im Gegensatz zu einem asymmetrischen Kryptosystem den gleichen Schlüssel zur Ver- und Entschlüsselung verwendet. Bei manchen symmetrischen Verfahren (z. B. IDEA) ist es dafür zunächst notwendig, den Verschlüsselungs-Schlüssel in einen Entschlüsselungs-Schlüssel zu transformieren.

Beispiele für symmetrische Kryptosysteme:

- AES (Advanced Encryption Standard), Nachfolger des DES
- DES (Data Encryption Standard)
- Triple-DES
- IDEA
- Blowfish
- Twofish

Asymmetrische Kryptosysteme:

Ein asymmetrisches Kryptosystem ist ein Kryptosystem, bei dem jeder der kommunizierenden Parteien ein Schlüsselpaar besitzt, das aus einem geheimen Teil (privater Schlüssel) und einem nicht geheimen Teil (öffentlicher Schlüssel) besteht. Der private Schlüssel ermöglicht es seinem Inhaber z.B., Daten zu entschlüsseln, digitale Signaturen zu erzeugen oder sich zu authentifizieren. Der öffentliche Schlüssel ermöglicht es jedermann, Daten für den Schlüsselinhaber zu verschlüsseln, dessen digitale Signaturen zu prüfen oder ihn zu authentifizieren. Im Gegensatz zu einem symmetrischen Kryptosystem müssen die kommunizierenden Parteien keinen gemeinsamen geheimen Schlüssel kennen.

Beispiele für symmetrische Kryptosysteme:

- RSA
- Elgamal

Erzeugung des öffentlichen und privaten Schlüssels:

Der öffentliche Schlüssel (public key) ist ein Zahlenpaar (*e*,*N*) und der private Schlüssel (private key) ein Zahlenpaar (*d*,*N*), wobei *N* bei beiden Schlüsseln gleich ist. Man nennt *N* den *RSA-Modul*, *e* den *Verschlüsselungsexponenten* und *d* den *Entschlüsselungsexponenten*. Diese Zahlen werden durch das folgende Verfahren erzeugt:

Wähle zufällig und stochastisch unabhängig zwei Primzahlen $p \neq q$. In der Praxis rät man dazu jeweils eine Zahl und führt mit dieser anschließend einen Primzahltest durch.

1. Berechne den RSA-Modul $N = p \cdot q$.
2. Berechne die Eulersche φ-Funktion von *N* $\varphi(N) = (p-1) \cdot (q-1)$
3. Wähle eine zu $\varphi(N)$ teilerfremde Zahl *e*, für die gilt $1 < e < \varphi(N)$.
4. Berechne den Entschlüsselungsexponent *d* als Multiplikativ Inverses von *e* bezüglich des Modulus $\varphi(N)$. Es soll also die folgende Kongruenz gelten
   $e \cdot d \equiv 1 \pmod{\varphi(N)}$

Die Zahlen *p*, *q* und $\varphi(N)$werden nicht mehr benötigt und sollten nach der Schlüsselerstellung auf sichere Weise gelöscht werden.

Bsp.:

1. Wir wählen *p* = 11 und *q* = 13 für die beiden Primzahlen.
2. Der RSA-Modul ist $N = p \cdot q = 143$.
3. Die eulersche φ-Funktion nimmt damit den Wert
   $\varphi(N) = \varphi(143) = (p-1)(q-1) = 120$ an.
4. Die Zahl *e* muss zu 120 teilerfremd sein. Wir wählen *e* = 23. Damit bilden *e* = 23 und *N* = 143 den öffentlichen Schlüssel.
5. Berechnung der Inversen zu *e*: Es gilt $ed + k\varphi(N) = 1$. Mit dem erweiterten euklidischen Algorithmus berechnet man den privaten Schlüssel *d* = 47 und die im Weiteren nicht mehr benötigte Zahl *k*.

Hybride Verfahren:

Hybride Verfahren sind eine Kombination aus symmetrischen und asymmetrischen Verfahren. Hybride Verfahren werden eingesetzt, da man bei symmetrischen Verfahren einen sicheren Weg braucht um den Schlüssel zu übertragen und da asymmetrische Verfahren langsam arbeiten. Daher nutzt z.B. SSH ein asymmetrisches Verfahren um den Schlüssel eines symmetrischen Kryptosystems zu übertragen.