

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Formelsammlung

1.5.2 TG Informationstechnik

Inkl. Lokale Formelsammlung HW: Arduino-IDE

Version: V 4.30

Gültig ab Abitur 2024

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Inhaltsverzeichnis:

1	Beschreibung von Systemzuständen mit UML-Zustandsdiagrammen.....	5
1.1	UML-Zustandsdiagramme (allgemein)	5
1.2	Begriffserklärung für UML-Zustandsdiagramme	6
1.3	Ergänzungen für Mikrocontroller.....	6
	Zustandsdefinition in C/CPP	6
	Zustandsvariable C/CPP	6
	Der Start-Pseudozustand	7
	Verhalten	7
	Zustandsübergang mit Wächterbedingung	7
	Zustandsübergang mit Ereignis und Wächterbedingung	8
	Selbsttransition und internes Ereignis	8
	Varianten von Transitionen	9
2	Hardware - Digitaltechnik.....	10
2.1	Logikgatter	10
2.2	Schaltnetze	11
2.3	Schaltwerke	12
	Taktgenerator	12
	Flip-Flops	12
	RAM.....	12
	ROM	12
	Schieberegister	13
	Zähler (Blockschaltbild).....	13
	Zähler (4-Bit).....	13
2.4	Sensoren.....	13
2.5	Aktoren.....	14
3	Hardware - Mikrokontrollertechnik.....	14
3.1	Blockschaltbild „Prüfungscontroller“	14
3.2	Prozessorarchitektur	15
	Programmiermodell.....	15
	Prozessorkern CPU	15
	Blockschaltbild Mikrocontroller	16
	Befehlspipeline einer RISC-CPU.....	16
	Speicherarchitektur	16
3.3	Onchip Peripherie	17
	Externer Interrupt.....	17
	Timer	17
	Puls-Weiten-Modulation (PWM)	18
	Analog – Digital – Wandlung	18
	Digital – Analog – Wandlung	18
3.4	Externe Kommunikationsmöglichkeiten	19
	Serial Peripheral Interface (SPI).....	19
	Universal Asynchronous Receiver Transmitter (UART).....	19
	Inter-Integrated Circuit (I ² C) SCL (Serial Clock): Taktleitung SDA (Serial Data): Datenleitung	20
3.5	Glossar.....	21
4	Hardware => Lokaler Teil: Arduino-IDE	23
4.1	Hochsprache C/CPP	23
	Datentypen	23
	Zeiger und Referenzen	23
	Operatoren.....	23
4.2	Schleifen	25

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

4.3	Programmverzweigungen	26
	Einfache Verzweigung mit if	26
4.4	Operationen (Unterprogramme, Funktionen)	28
4.5	Beispiel eines C/CPP-Programms.....	29
4.6	On Chip Peripherie.....	30
	Portpin: Eingabe und Ausgabe.....	30
4.7	Externer Interrupt	30
4.8	Timer.....	31
4.9	Puls-Weiten-Modulation (PWM)	32
4.10	Analog – Digital – Wandlung	33
	Digital – Analog – Wandlung	33
4.11	Externe Kommunikationsmöglichkeiten	34
	Universal Asynchronous Receiver Transmitter (UART).....	34
	Serial Peripheral Interface (SPI).....	35
	Inter-Integrated Circuit (I ² C) SCL (Serial Clock): Taktleitung SDA (Serial Data): Datenleitung	36
5	Programmentwicklung und Objektorientierter Entwurf	38
5.1	Vergleichsoperatoren für Bedingungen (Pseudocode)	38
5.2	Kontrollstrukturen (Pseudocode)	38
5.3	Datentypen.....	39
	Elementare Datentypen	39
	Komplexe Datentypen.....	39
5.4	Klassen	40
	Attribute	40
	Operationen.....	41
	Assoziationen, Rollennamen und Multiplizitäten	41
	Beispiel einer Operation mit einer Kollektion in Pseudocode.....	42
5.5	Vererbung	42
5.6	Abstrakte Klassen und Schnittstellen	42
5.7	Objektdiagramme.....	43
5.8	Sequenzdiagramme	44
5.9	Zustandsdiagramme.....	46
6	Datenstrukturen.....	47
6.1	Verkettete Liste	47
6.2	Stapel.....	47
6.3	Warteschlange	48
6.4	Binärbaum.....	48
	Beispiel für einen Binärbaum der Tiefe 3	48
	Datenstruktur	49
	Operation ausgebenDatenInorder() der Klasse Knoten in Pseudocode.....	49
7	Künstliche Intelligenz.....	50
7.1	Klassifikation	50
8	Datenbanken	51
8.1	Datenbankmanagementsystem	51
8.2	Entity-Relationship-Diagramm (ER-Diagramm)	51
8.3	Relationenmodell	52
8.4	Abfrageformulierung mit SQL.....	52
	Projektion und Formatierung.....	52
	Selektion.....	53
	Verbund von Tabellen.....	54
	Aggregatfunktion.....	55
	Aggregatfunktion mit Gruppierung	56
	Selektion von Gruppen	56
	Komplette SQL-Anweisung	56

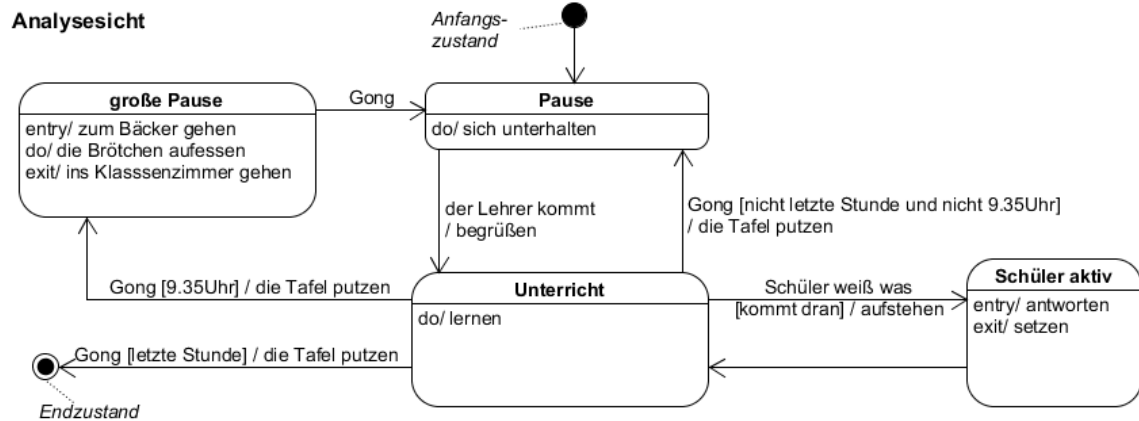
Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

9 Vernetzte Systeme.....	57
9.1 Netzwerktechnik.....	57
Netzwerksymbole.....	57
Routing-Tabelle (IPv4)	57
Aufbau IPv4-Adresse	58
Aufbau IPv6-Adresse	58
9.2 Schichtenmodelle.....	59
ISO-OSI-7-Schichtenmodell	59
TCP-IP-Schichtenmodell.....	59
9.3 Header	59
Ethernet II	59
IPv4-Header.....	59
IPv6-Header.....	60
TCP –Header.....	60
UDP –Header	60
9.4 Internet der Dinge (IoT)	61
MQTT-Protokoll (Message Queuing Telemetry Transport)	61
HTTP-Protokoll (Hypertext Transfer Protocol)	62

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

1 Beschreibung von Systemzuständen mit UML-Zustandsdiagrammen

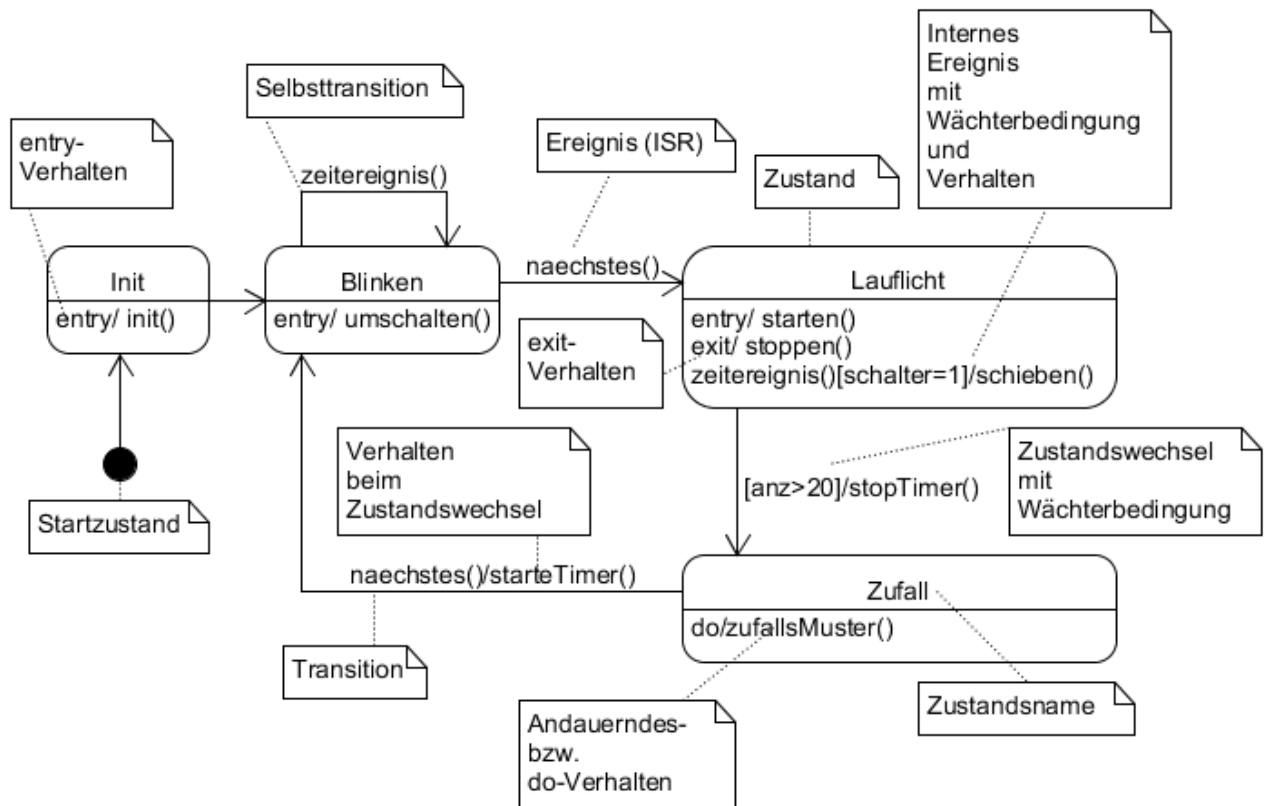
1.1 UML-Zustandsdiagramme (allgemein)



Notation	
Zustand	<div> Zustandsname entry/ Eintrittsverhalten do/ Andauernverhalten exit/ Austrittsverhalten internesEreignis/ Verhalten </div>
Transitionen	<div> Ereignis [Wächterbedingung] / Verhalten Ereignis Ereignis [Wächterbedingung] Ereignis / Verhalten [Wächterbedingung] [Wächterbedingung] / Verhalten / Verhalten </div>
Ereignisse	Signal, Botschaft: Z.B. click(taste), einFensterAuf(), Gong
Hinweis	Ereignisse können nur das andauernde Verhalten (do-Verhalten) unterbrechen bzw. beenden, nicht aber das Eintritts- und das Austrittsverhalten.

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

1.2 Begriffserklärung für UML-Zustandsdiagramme



1.3 Ergänzungen für Mikrocontroller

Hinweis: Die folgenden Codebeispiele sind nicht verbindlich

Zustandsdefinition in C/CPP

Zustände sollten aus Gründen der Übersichtlichkeit Namen gegeben werden. Dadurch wird der Zusammenhang von Zustandsdiagramm und Programm verdeutlicht.

Allgemein	Beispiel
#define Zustandsname Zustandsnummer	#define Init 0 #define Blinken 1
oder	
enum zustandstyp {Zustandsname=Zustandsnummer, ... }	enum zustandstyp {Init=0, Blinken=1, ...};

Zustandsvariable C/CPP

Ein Zustand kann durch eine Zustandsvariable gekennzeichnet werden:

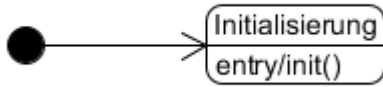
Beispiele	Erklärung
int zustand;	Zustandsvariable vom Typ int
PortOut zustand(PortC,0xFF);	Eine Portkonfiguration repräsentiert den Zustand
zustandstyp zustand;	Zustandsvariable als enum (siehe oben)

Hinweis: Eine Zustandsvariable kann auch ein Ausgangsport des Mikrocontrollers sein (2. Beispiel). In diesem Fall bewirkt ein Zustandswechsel gleichzeitig, dass die Ausgänge entsprechend dem neuen Zustand angepasst werden.

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Der Start-Pseudozustand

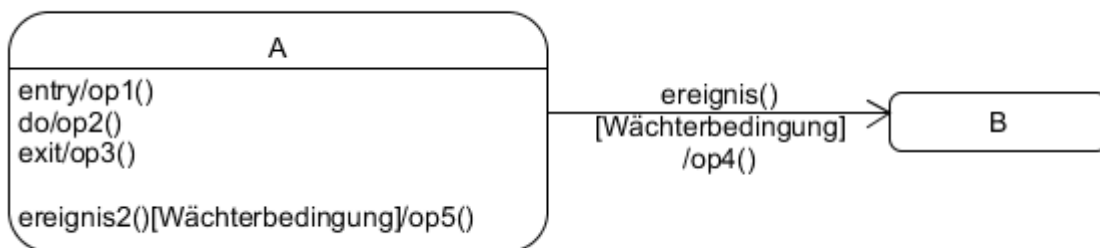
Die meisten Zustandsdiagramme beginnen mit einem Start-Pseudozustand:



Der ausgefüllte Kreis symbolisiert den Startpunkt des Zustandsdiagramms. Oft ist er mit dem Start des Mikrocontrollerprogramms gleich zu setzen. Die Transition vom Startpunkt zum ersten Zustand ist immer unbeschriftet.

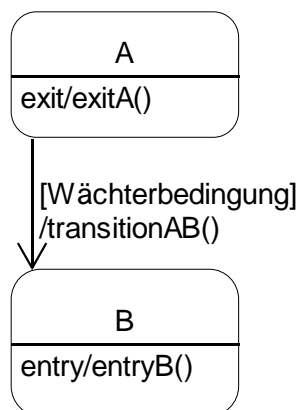
Verhalten

Verhalten sind Operationen oder Anweisungen, die an bestimmten Stellen des Zustandsdiagramms ausgeführt werden



Verhalten	Ausführung	Beispiel
Entry-Verhalten	bei Eintritt in einen Zustand	op1()
Do-Verhalten	andauernd, solange der Zustand anhält	op2()
Exit-Verhalten	bei Verlassen des Zustands	op3()
Verhalten an der Transition	beim Zustandswechsel	op4()
Verhalten am internen Ereignis	Wenn das interne Ereignis eintritt und gegebenenfalls eine Wächterbedingung erfüllt ist	op5()

Zustandsübergang mit Wächterbedingung



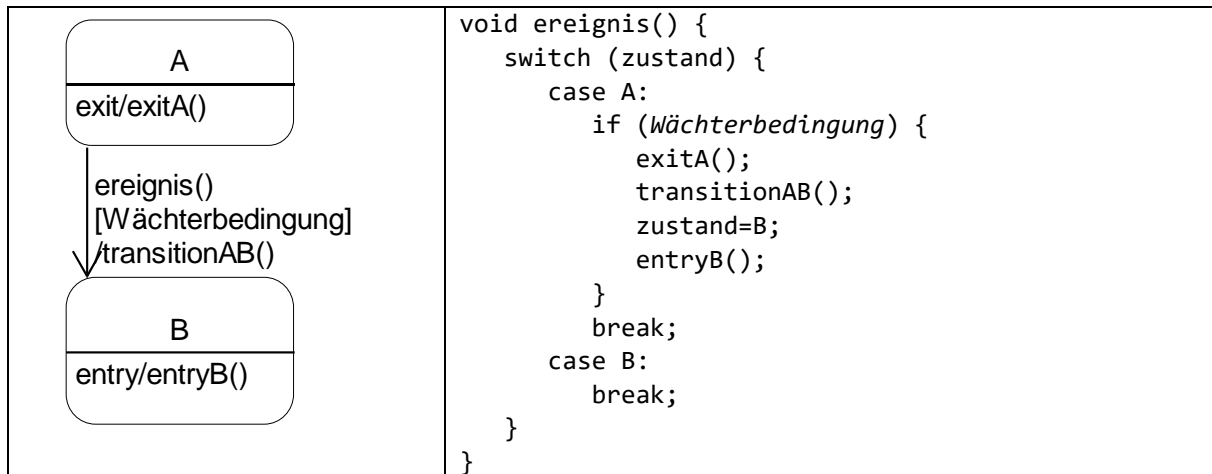
```
int main() {
    while(true) {
        switch (zustand) {
            case A:
                if (Wächterbedingung) {
                    exitA();
                    transitionAB();
                    zustand=B;
                    entryB();
                }
                break;
            case B:
                break;
        }
    }
}
```

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

In der Endlosschleife wird zuerst der Zustand geprüft. Falls sich der Mikrocontroller im Zustand A befindet, wird die Wächterbedingung an der Transition überprüft. Falls die Wächterbedingung erfüllt ist, erfolgt der Zustandswechsel. Es werden dann in folgender Reihenfolge die Verhalten ausgeführt:

1. Exit-Verhalten von Zustand A: `exitA()`
2. Verhalten an der Transition: `transitionAB()`
3. Zustandswechsel: `zustand=B`
4. Entry-Verhalten von Zustand B: `entryB()`

Zustandsübergang mit Ereignis und Wächterbedingung



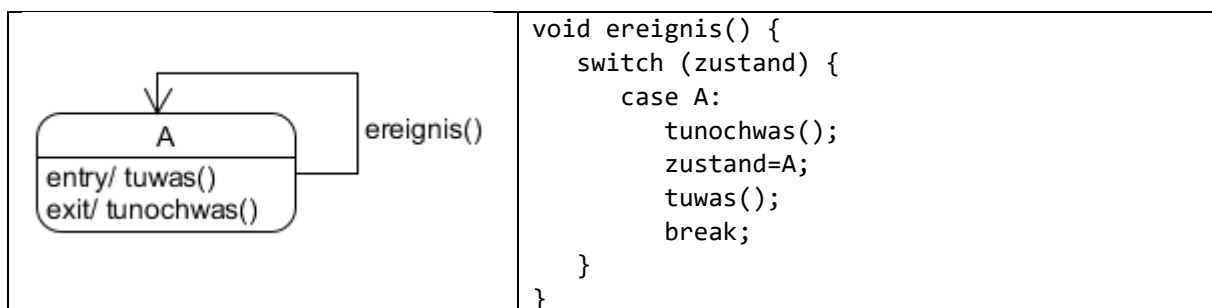
Es gibt **Aufruf-** und **Signal-Ereignisse**. Bei Signal-Ereignissen handelt es sich um Interrupts. Als Ereignisbezeichnung wird der Name der **Interrupt Service Routine (ISR)** verwendet.

In der ISR wird zuerst der Zustand geprüft. Falls sich der Mikrocontroller im Zustand A befindet, wird die Wächterbedingung an der Transition überprüft. Falls die Wächterbedingung erfüllt ist, erfolgt der Zustandswechsel. Es wird dann in folgender Reihenfolge das Verhalten ausgeführt:

1. Exit-Verhalten von Zustand A: `exitA()`
2. Verhalten an der Transition: `transitionAB()`
3. Zustandswechsel: `zustand=B`
4. Entry-Verhalten von Zustand B: `entryB()`

Selbsttransition und internes Ereignis

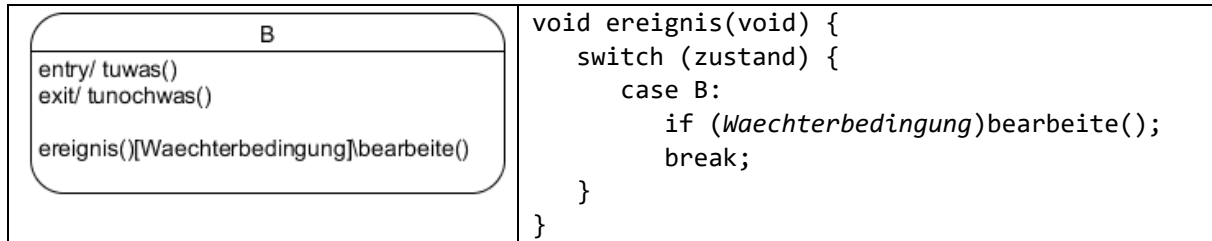
Selbsttransition



In der ISR `ereignis()` wird zuerst der Zustand geprüft. Falls sich der Mikrocontroller im Zustand A befindet, wird die Wächterbedingung, falls vorhanden, an der Transition überprüft. Falls die Wächterbedingung erfüllt ist, erfolgt der Zustandswechsel wieder nach A. Es werden nacheinander die `exit`-, `Transitions`- und `entry`-Verhalten ausgeführt.

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Internes Ereignis



In der ISR ereignis() wird zuerst der Zustand geprüft. Falls sich der Mikrocontroller im Zustand B befindet, wird die Wächterbedingung, falls vorhanden, an der Transition überprüft. Falls die Wächterbedingung erfüllt ist, wird der Code, der zu diesem Ereignis in diesem Zustand gehört ausgeführt.

Varianten von Transitionen








Transitionen bezeichnen Zustandsübergänge und werden als Pfeil mit offener Spitze vom Ausgangszustand zum Zielzustand gezeichnet.

- | | |
|--|--|
| → | a) Transition ohne Beschriftung: Sie bewirkt einen unmittelbaren Zustandswechsel, nachdem das entry-Verhalten beendet wurde. |
| → [Wächterbedingung] | b) Transition mit Wächterbedingung: Sie bewirkt einen Zustandswechsel, sobald die Wächterbedingung erfüllt ist. |
| → [Wächterbedingung]/verhalten() | c) Transition mit Wächterbedingung und Verhalten: Sie bewirkt einen Zustandswechsel, sobald die Wächterbedingung erfüllt ist. Beim Zustandswechsel wird das Verhalten ausgeführt. |
| → ereignis() | d) Transition mit Ereignis: Sie bewirkt einen Zustandswechsel beim Eintreten des Ereignisses ereignis(). |
| → ereignis()/verhalten() | e) Transition mit Ereignis und Verhalten: Verhalten wie bei d). Zusätzlich wird beim Zustandswechsel noch das Verhalten verhalten() ausgeführt. |
| → ereignis()[Wächterbedingung] | f) Transition mit Ereignis und Wächterbedingung: Verhalten wie bei d), falls zusätzlich die Wächterbedingung erfüllt ist. |
| → ereignis()[Wächterbedingung]/verhalten() | g) Transition mit Ereignis, Wächterbedingung und Verhalten: Verhalten wie bei f). Zusätzlich wird beim Zustandswechsel noch das Verhalten verhalten() ausgeführt. |

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

2 Hardware - Digitaltechnik

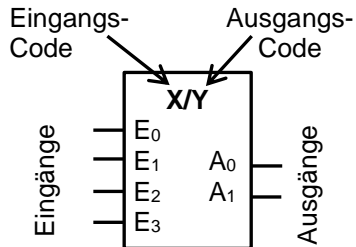
2.1 Logikgatter

<div><div><div>NOT (Negation)</div><div><div><div>A</div><div></div><div>out</div></div></div><div><div><div>A</div><div><div>1</div></div><div><div>Y</div></div></div></div><div><div>$Y = \bar{A}$</div><div>$Y = !A$</div><div>$Y = \neg A$</div></div></div><div><table><tr><th>A</th><th>Y</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table></div></div>	A	Y	0	1	1	0																									
A	Y																														
0	1																														
1	0																														
<div><div><div>AND (Konjunktion)</div><div><div><div>A</div><div>B</div><div></div><div>out</div></div></div><div><div><div>A</div><div>B</div><div><div>&</div></div><div>Y</div></div></div><div><div>$Y = A \wedge B$</div><div>$Y = A \& B$</div></div></div><div><table><tr><th>B</th><th>A</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></div></div>	B	A	Y	0	0	0	0	1	0	1	0	0	1	1	1	<div><div><div>NAND</div><div><div><div>A</div><div>B</div><div></div><div>out</div></div></div><div><div><div>A</div><div>B</div><div><div>&</div></div><div><div>Y</div></div></div></div><div><div>$Y = \overline{A \wedge B}$</div><div>$Y = !(A \& B)$</div></div></div><div><table><tr><th>B</th><th>A</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table></div></div>	B	A	Y	0	0	1	0	1	1	1	0	1	1	1	0
B	A	Y																													
0	0	0																													
0	1	0																													
1	0	0																													
1	1	1																													
B	A	Y																													
0	0	1																													
0	1	1																													
1	0	1																													
1	1	0																													
<div><div><div>OR (Disjunktion)</div><div><div><div>A</div><div>B</div><div></div><div>out</div></div></div><div><div><div>A</div><div>B</div><div><div>≥1</div></div><div>Y</div></div></div><div><div>$Y = A \vee B$</div><div>$Y = A \# B$</div></div></div><div><table><tr><th>B</th><th>A</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></div></div>	B	A	Y	0	0	0	0	1	1	1	0	1	1	1	1	<div><div><div>NOR</div><div><div><div>A</div><div>B</div><div></div><div>out</div></div></div><div><div><div>A</div><div>B</div><div><div>≥1</div></div><div><div>Y</div></div></div></div><div><div>$Y = \overline{A \vee B}$</div><div>$Y = !(A \# B)$</div></div></div><div><table><tr><th>B</th><th>A</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table></div></div>	B	A	Y	0	0	1	0	1	0	1	0	0	1	1	0
B	A	Y																													
0	0	0																													
0	1	1																													
1	0	1																													
1	1	1																													
B	A	Y																													
0	0	1																													
0	1	0																													
1	0	0																													
1	1	0																													
<div><div><div>XOR (Antivalenz)</div><div><div><div>A</div><div>B</div><div></div><div>out</div></div></div><div><div><div>A</div><div>B</div><div><div>=1</div></div><div>Y</div></div></div><div><div>$Y = (\bar{A} \wedge B) \vee (A \wedge \bar{B})$</div><div>$Y = (!A \& B) \# (A \& !B)$</div></div></div><div><table><tr><th>B</th><th>A</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table></div></div>	B	A	Y	0	0	0	0	1	1	1	0	1	1	1	0	<div><div><div>XNOR (Äquivalenz)</div><div><div><div>A</div><div>B</div><div></div><div>out</div></div></div><div><div><div>A</div><div>B</div><div><div>=</div></div><div>Y</div></div></div><div><div>$Y = (\bar{A} \wedge \bar{B}) \vee (A \wedge B)$</div><div>$Y = (!A \& !B) \# (A \& B)$</div></div></div><div><table><tr><th>B</th><th>A</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></div></div>	B	A	Y	0	0	1	0	1	0	1	0	0	1	1	1
B	A	Y																													
0	0	0																													
0	1	1																													
1	0	1																													
1	1	0																													
B	A	Y																													
0	0	1																													
0	1	0																													
1	0	0																													
1	1	1																													

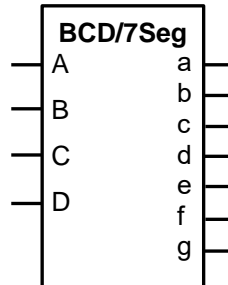
Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

2.2 Schaltnetze

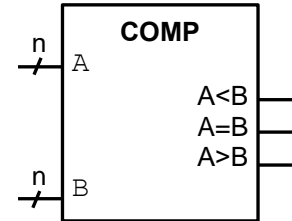
Codeumsetzer (Umcodierer)



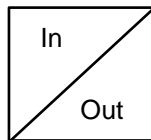
BCD zu 7 Seg



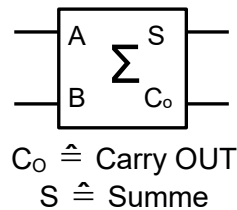
Vergleicher



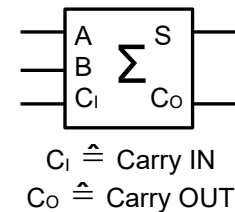
BSB Codewandler



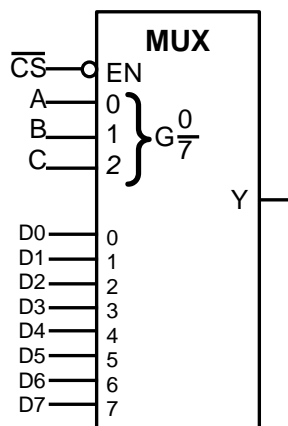
Halbaddierer



Volladdierer

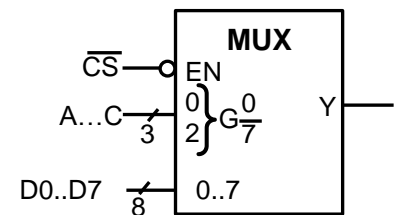


MUX (8 zu 1)



C	B	A	\overline{CS}	Y
x	x	x	1	0
0	0	0	0	D0
0	0	1	0	D1
0	1	0	0	D2
0	1	1	0	D3
1	0	0	0	D4
1	0	1	0	D5
1	1	0	0	D6
1	1	1	0	D7

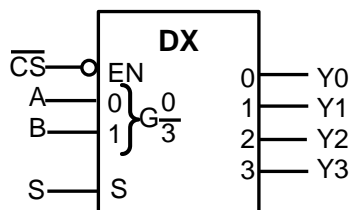
x $\hat{=}$ don't care



Adress- und Datenleitungen können auch zusammen-gefasst werden

CS = chip select (low active)

DEMUX (1 zu 4) Decodierer

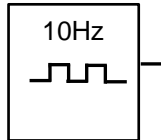


B	A	\overline{CS}	Y3	Y2	Y1	Y0
X	X	1	0	0	0	0
0	0	0	0	0	0	S
0	1	0	0	0	S	0
1	0	0	0	S	0	0
1	1	0	S	0	0	0

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

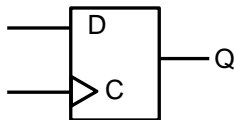
2.3 Schaltwerke

Taktgenerator



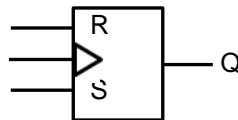
Flip-Flops

D-Flip-Flop



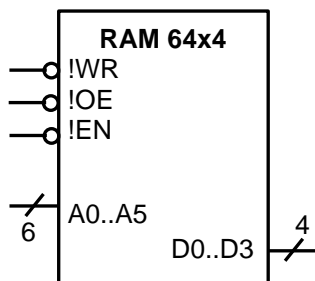
Takt	D	Q^{n+1}
↑	0	0
↑	1	1
sonst	X	Q^n

RS-Flip-Flop



Takt	R	S	Q^{n+1}
↑	0	0	Q^n
↑	1	0	0
↑	0	1	1
↑	1	1	Undefiniert
sonst	x	x	Q^n

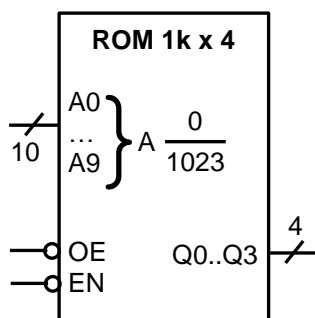
RAM



Schreib-Lese-Speicher mit 64 mal 4 Bit

- 4-Bit Registerbreite
- 64 Register gesamt
- **A0-A5**: Adresseingänge
- **D0-D3**: Ein-/Ausgabe des Speicherinhalts
- **WR=0**: lesen (von **D0-D3** in den Speicher)
- **WR=1**: schreiben (vom Speicher an **D0-D3**)
- **OE=1**: Tri-State
- **OE=0**: Speicherinhalt lesen
- **EN=0**: aktiviert den Baustein

ROM

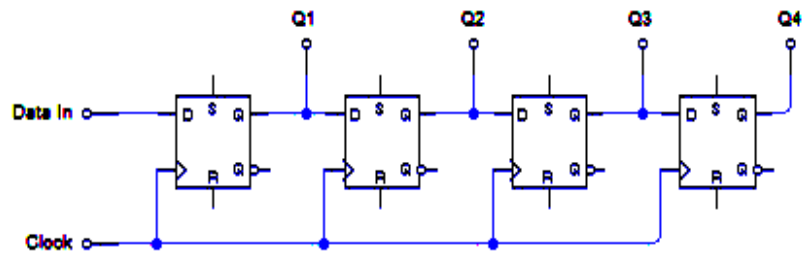
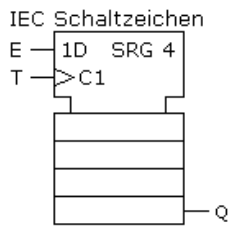


Festwertspeicher mit 1024 (1KiBi) mal 4 Bit

- **A0-A9**: Adresseingänge
- **OE=1**: Tri-State
- **OE=0**: Speicherinhalt lesen
- **EN=0**: aktiviert den Baustein
- **Q0-Q3**: Wert der Speicherzelle an Adresse A

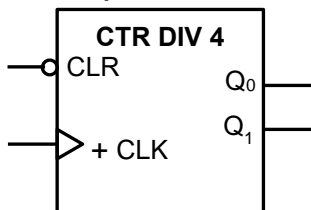
Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Schieberegister



Beispiel: Seriell In => Parallel Out

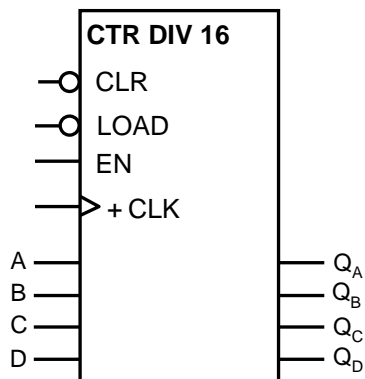
Zähler (Blockschaltbild)



Mit jeder steigenden Flanke an **CLK** wird der Zählerwert um 1 erhöht. Nach dem maximalen Wert wird der Zählwert wieder auf 0 gesetzt.

- **CTR**: Zähler (counter)
- **DIV 4**: 4 verschiedene binäre Zustände
- **CLR = 0** setzt den Counter auf den Wert **0** zurück
- **Q_n** gibt den Zählerzustand aus

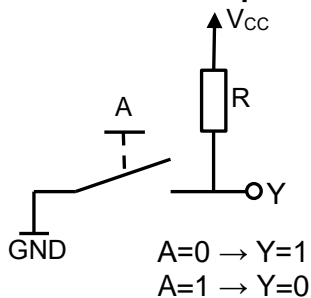
Zähler (4-Bit)



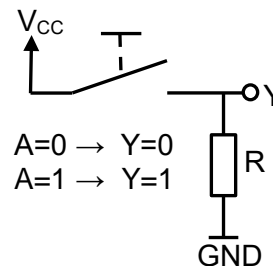
- **CTR**: Zähler (counter)
- **DIV 16**: 16 verschiedene binäre Zustände
- Vorwärtszähler (+)
- EN = 1 und die positive Taktflanke führen zum nächsten Zählzustand
- Mit **LOAD = 0** kann ein Anfangszustand geladen werden
- **CLR = 0** setzt den Counter auf den Wert 0 zurück

2.4 Sensoren

Taster mit Pull-Up-Widerstand



Taster mit Pull-Down Widerstand

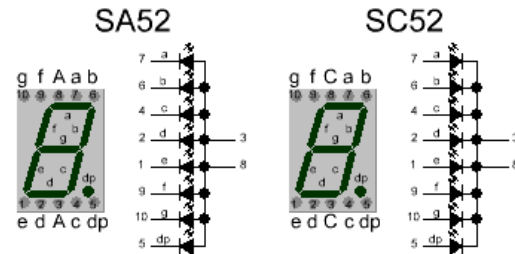


Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

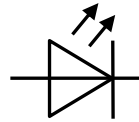
2.5 Aktoren

7-Segmentanzeige

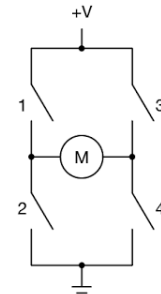
Common Anode ⇔ Common Cathode



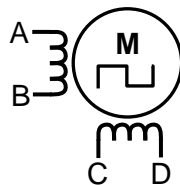
LED



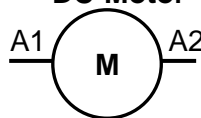
H-Brücke



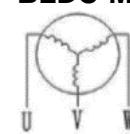
Schrittmotor



DC-Motor

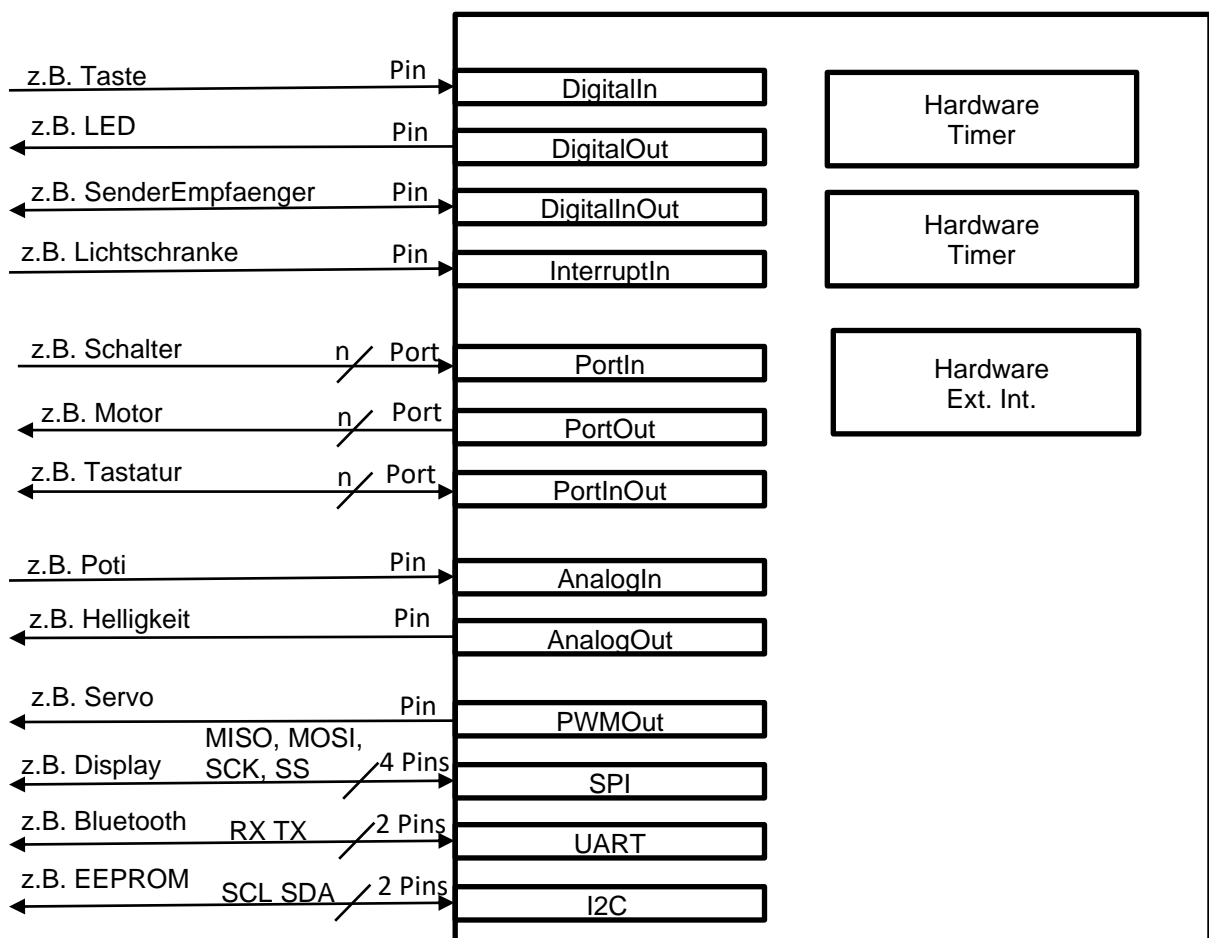


BLDC-Motor



3 Hardware - Mikrocontrollertechnik

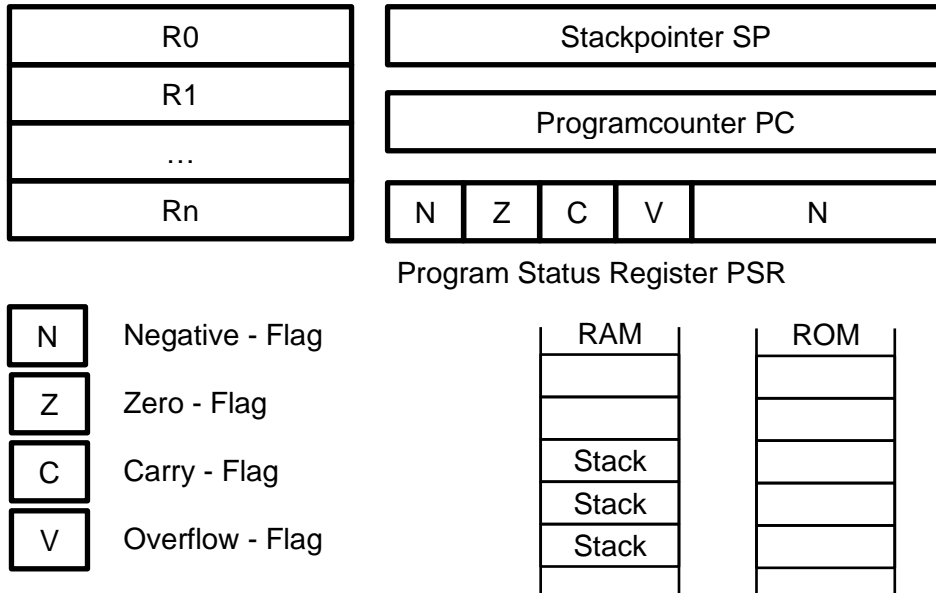
3.1 Blockschaltbild „Prüfungscontroller“



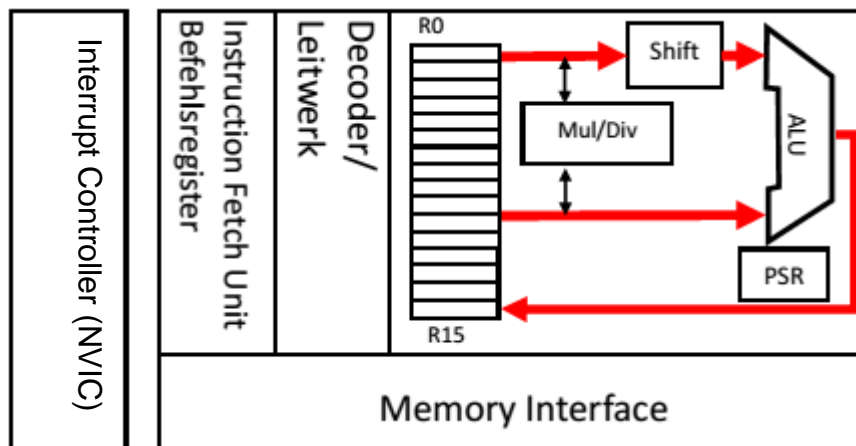
Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

3.2 Prozessorarchitektur

Programmiermodell

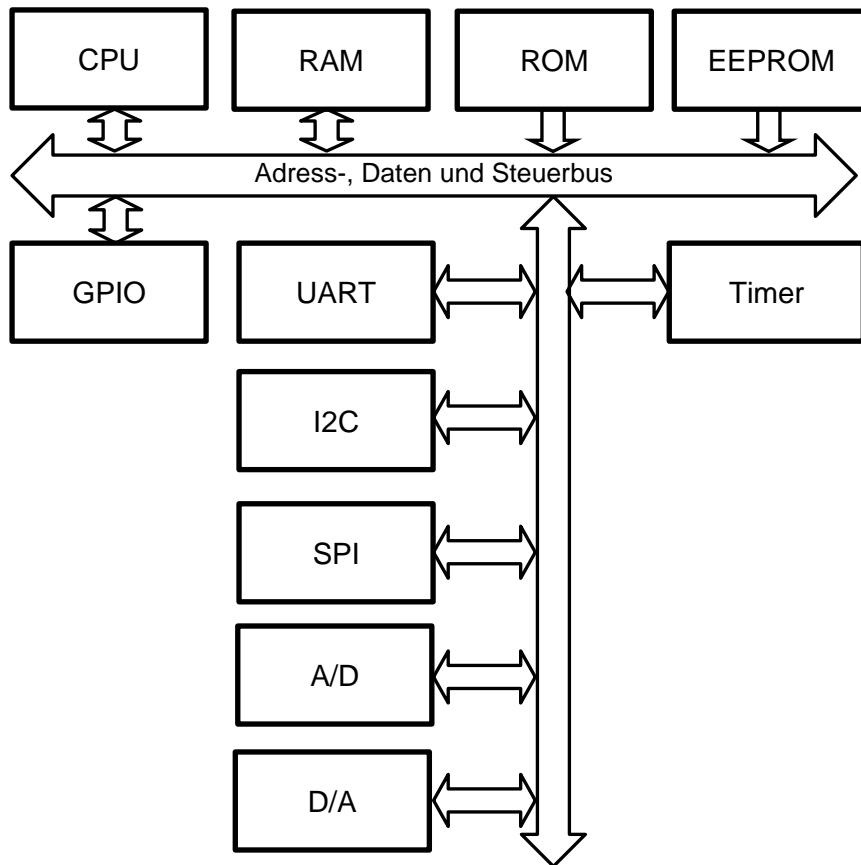


Prozessorkern CPU

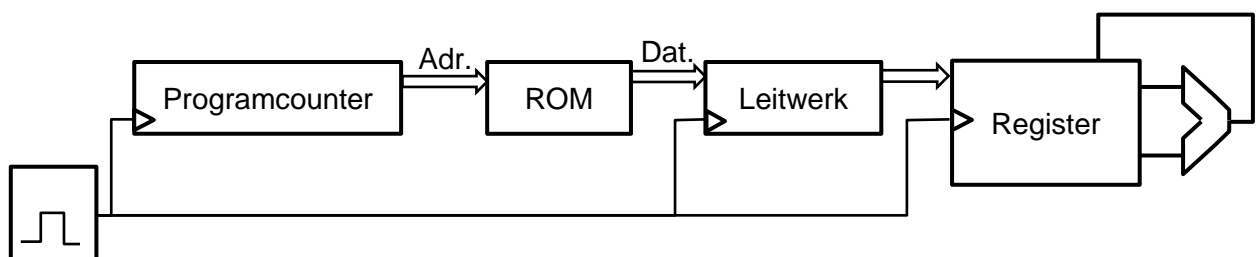


Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Blockschaltbild Mikrocontroller

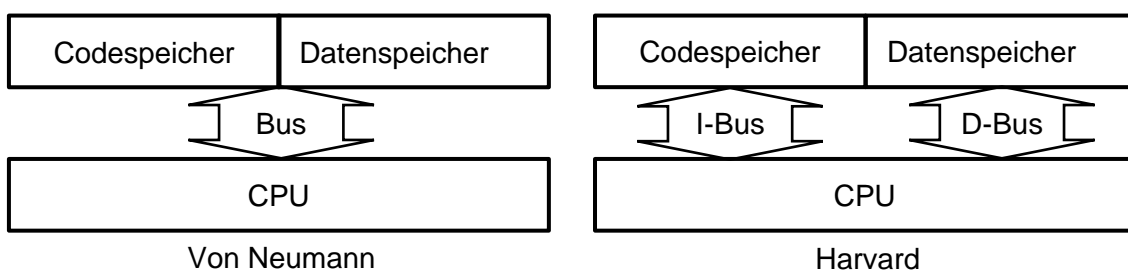


Befehlspipeline einer RISC-CPU



Takt	1	2	3	4	5	6
	Fetch 1	Decode 1	Execute 1	Fetch 4	Decode 4	Execute 4
		Fetch 2	Decode 2	Execute 2	Fetch 5	Decode 5
			Fetch 3	Decode 3	Execute 3	Fetch 6

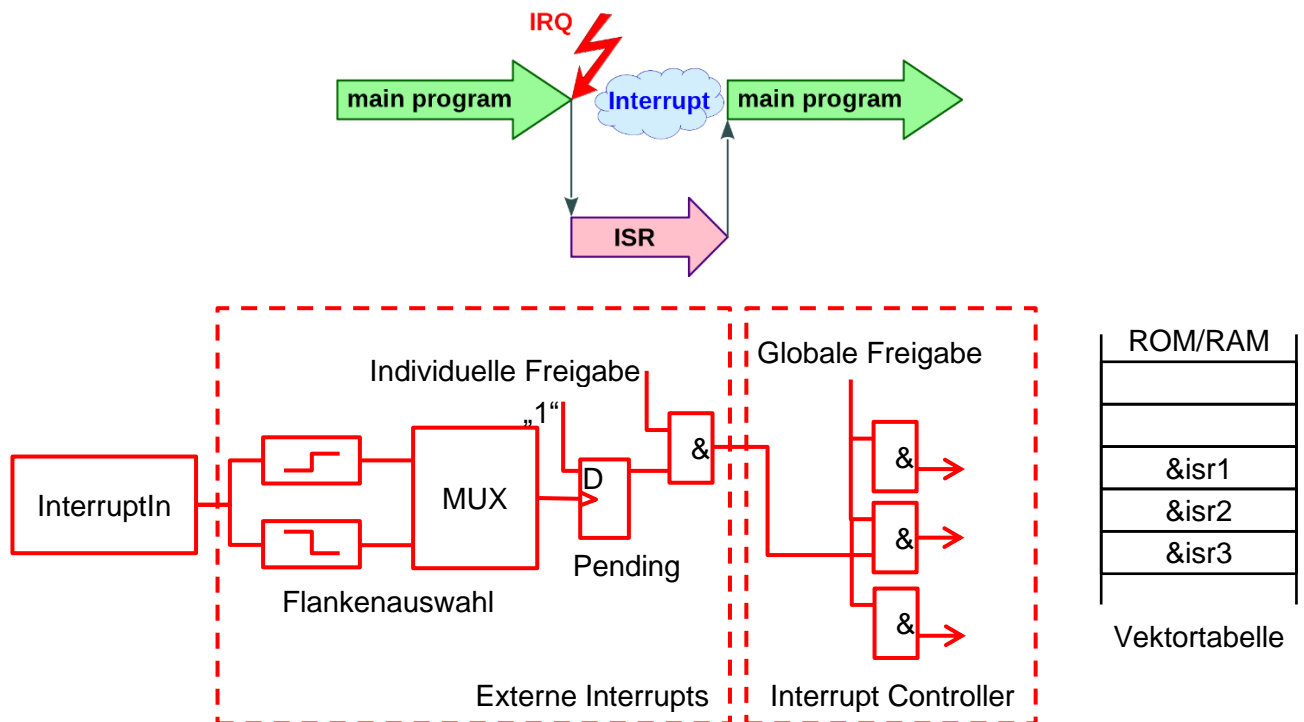
Speicherarchitektur



Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

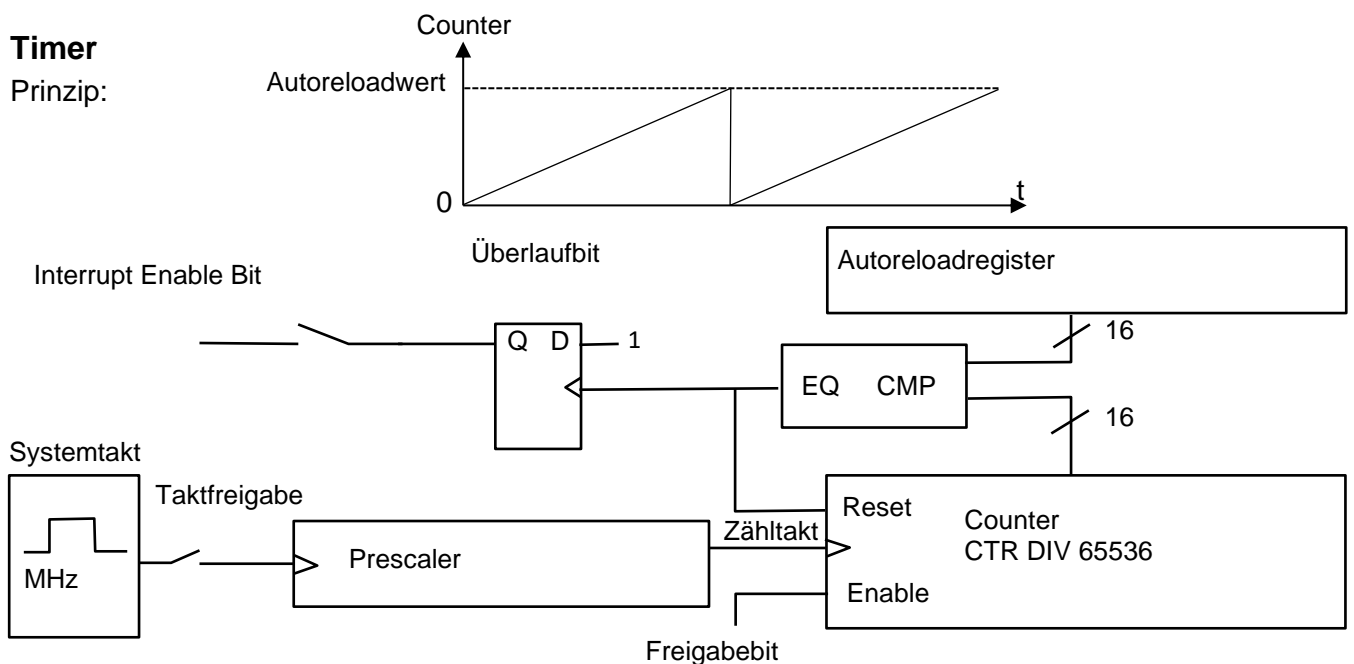
3.3 Onchip Peripherie

Externer Interrupt



Timer

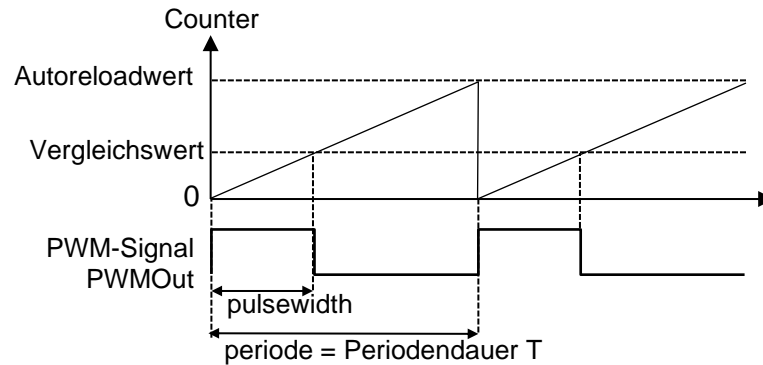
Prinzip:



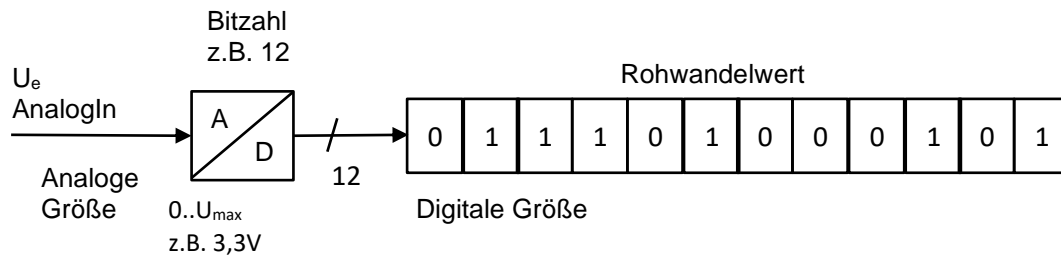
Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Puls-Weiten-Modulation (PWM)

Prinzip:



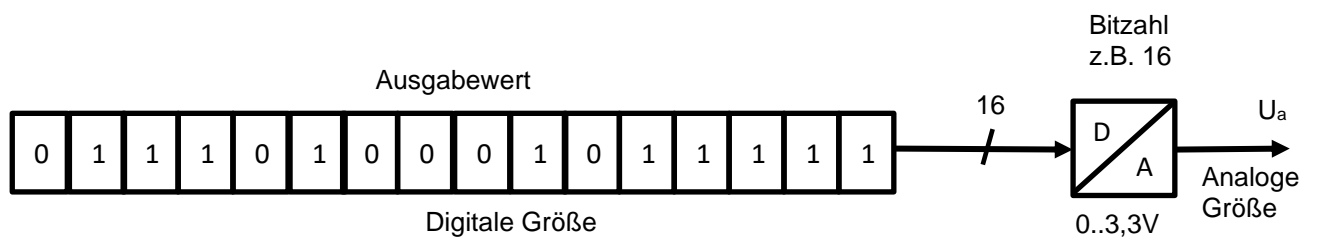
Analog – Digital – Wandlung



Berechnungsformeln

$$\begin{aligned} \text{Rohwandelwert} &= \frac{U_e}{U_{\max}} \cdot (2^{\text{Bitzahl}} - 1) && \text{z.B. } \frac{U_e}{3,3V} \cdot 4095 \\ \text{Wandelwert als Kommazahl:} & x = (U_e / U_{\max}) && \text{z.B. } x = (U_e / 3.3) \\ \text{Stufung (analoge Auflösung):} & U_{\max} / 4095 && \text{z.B. } 3,3V / 4095 \\ \text{Wandelwert als Ganzzahl linksbündig:} & \text{unsigned short } x = (U_e / U_{\max}) * 65535 \end{aligned}$$

Digital – Analog – Wandlung



Berechnungsformeln

$$\begin{aligned} \text{float } x: & U_a = x \cdot 3,3V \\ \text{unsigned short } x: & U_a = \frac{x \cdot 3,3V}{65535} \end{aligned}$$

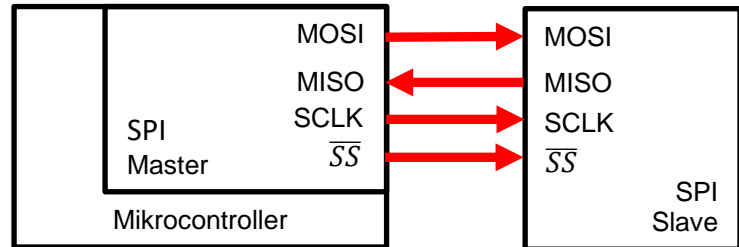
Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

3.4 Externe Kommunikationsmöglichkeiten

Serial Peripheral Interface (SPI)

Das **Serial Peripheral Interface (SPI)** dient der Kommunikation des Mikrocontrollers mit **Modulen** auf der Platine. Module sind

- Anzeigen,
- Speicher,
- LAN-Bausteine
- ...



Signale

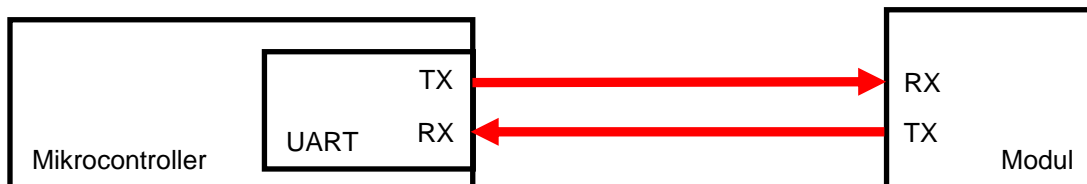
MOSI (Master Out Slave In): Sendeleitung

MISO (Master In Slave Out): Empfangsleitung

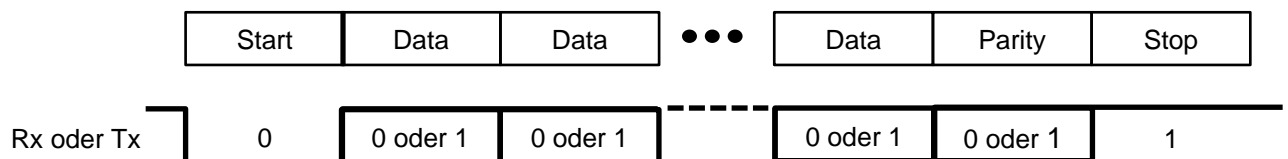
SCLK (Serial Clock): Taktleitung

SS (Slave Select): Auswahl des Slaves (Lowaktiv)

Universal Asynchronous Receiver Transmitter (UART)



Frame



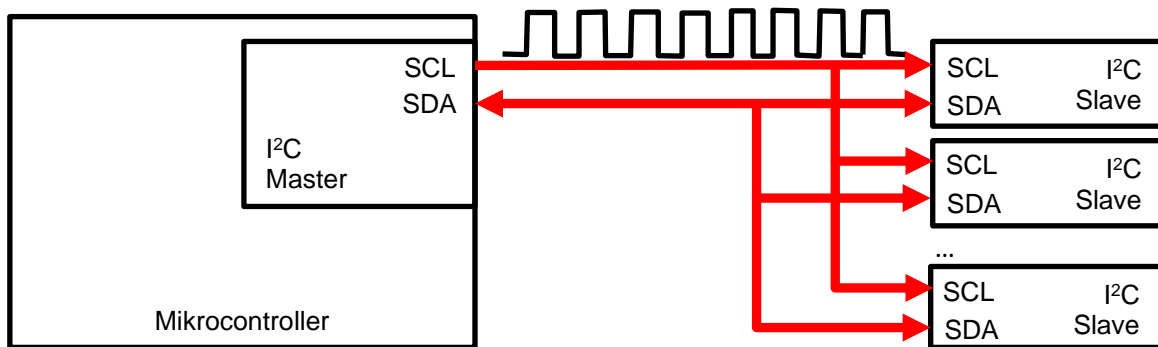
Eine UART-Übertragung beginnt immer mit einem Startbit (Low). Darauf folgen

- 5-8 **Datenbits** (Standard = 8)
- 0 oder 1 **Paritybit** (Standard = 0 none)
- 1 oder 2 **Stopbits** (Standard = 1)

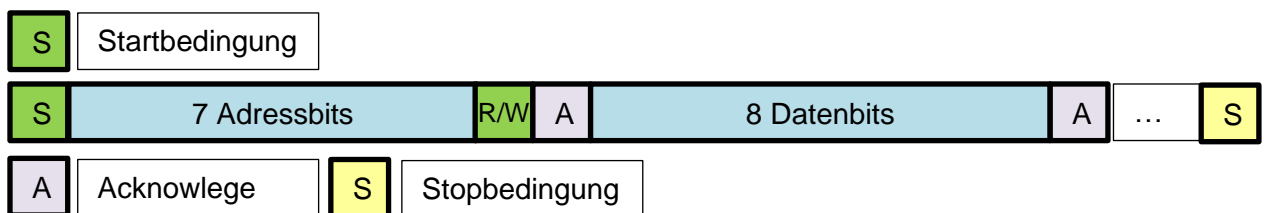
Falls ein Paritybit programmiert wurde, kann es gerade Parity (even) oder ungerade Parity (odd) anzeigen.

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Inter-Integrated Circuit (I²C) SCL (Serial Clock): Taktleitung SDA (Serial Data): Datenleitung



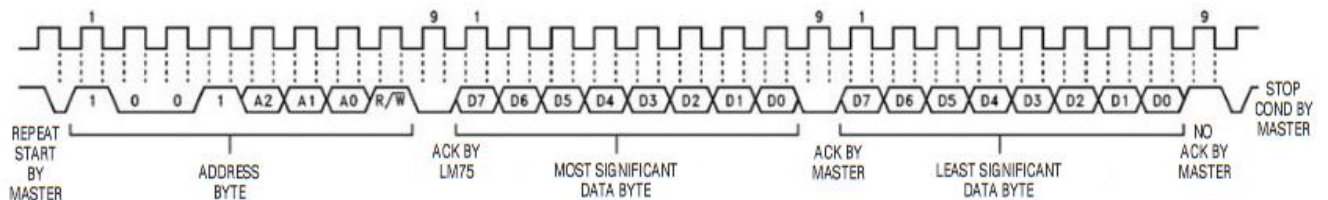
Frame:



Beispielhaft aufgeführte I²C-Bausteine bzw. Auszug Datenblätter Quelle: www.alldatasheet.com

LM 75:

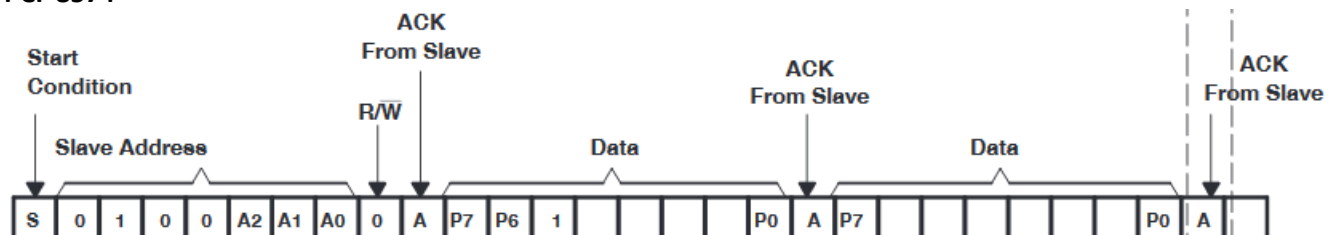
BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
1	0	0	1	A2	A1	A0	RD/W



UPPER BYTE								LOWER BYTE							
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Sign bit 1= Negative 0 = Positive	MSB 64°C	32°C	16°C	8°C	4°C	2°C	1°C	LSB 0.5°C	X	X	X	X	X	X	X

X = Don't care.

PCF 8574



Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

3.5 Glossar

Acknowledge	Quittierung
AD-Wandler	Analog-Digital-Wandler
ALU	Arithmetisch-Logische Einheit
AnalogIn	Analogeingang Pin
AnalogOut	,Analogausgang Pin
BCD	Binär Codiert Dezimal
BLDC-Motor	Bürstenloser Gleichstrommotor, Brushless DC-Motor
Bluetooth	Funkstandard zur Datenübertragung
Carry	Übertrag
CISC	Complex Instructionset Computer
CLK	Clock, Takt
CPU	Central Processing Unit
CS	Steuerleitung für Chip Select
CTR	Counter
DA-Wandler	Digital-Analog-Wandler
DEMUX	Demultiplexer
DigitalIn	Digitaleingang Pin
DigitalInOut	Digital Input Output Pin bidirektional
DigitalOut	Digitalausgang Pin
DIV16	Modulo 16
EN	Enable, Freigabe
EPROM	Erasable Programmable Read Only Memory
EVA	Eingabe Verarbeitung Ausgabe
Even	gerade
Frame	Rahmen
GPIO	General Purpose Input Output
Hardware Timer	16-Bit Timer
I2C	Inter-Integrated Circuit
InterruptIn	Interrupteingang Pin
LED	Light Emitting Diode Leuchtdiode
LOAD	laden
MISO	MasterIn – SlaveOut
MOSI	MasterOut – SlaveIn
MUX	Multiplexer
NVIC	Nested Vector Interrupt Controller
Odd	Ungerade
OE	Steuerleitung für Output Enable
Overflow	Überlauf
Parity	Geradzahligkeit

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Periode	Periodendauer
PortIn	Digitaleingang Port
PortInOut	Digital Input Output Port bidirektional
PortOut	Digitalausgang Port
Poti	Potentiometer Einstellwiderstand für analoge Eingabe
Pulsewidth	Pulsbreite
PWM	Puls-Weiten-Modulation
R0 usw.	Prozessorregister
RAM	Random Access Memory
RD	Steuerleitung für lesen
RISC	Reduced Instructionset Computer
ROM	Read Only Memory
Rx	Receive
SCL(K)	Serial Clock
SDA	Serial Data
SPI	Serial Peripheral Interface
SRG	Schieberegister
SS	Slave Select
Stack	Stapel
Stackpointer	Stapelzeiger
Tx	Transmit
UART	Universal Synchronous Asynchronous Receiver Transmitter
WR	Steuerleitung für schreiben

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

4 Hardware => Lokaler Teil: Arduino-IDE

4.1 Hochsprache C/CPP

Datentypen

Datentyp	Bits	Vorzeichen	Wertebereich
unsigned char	8	+	0 .. 255
(signed) char	8	- +	-128 .. 127
uint_32t/uint16_t	32/16	+	0 .. 4294967295 bzw. 0 .. 65535
int_32t/int16_t	32/16	- +	-2147483648 .. 2147483647 bzw. 32768 .. 32767
float	32	- +	-3,4E38 .. 3,4E38

Zeiger und Referenzen

```
int x=127; //Wert
int *y; //Zeiger
```

```
*y=x; //der Zeiger weist auf eine Variable mit dem Wert von x
y=&x; //der Zeiger bekommt die Adresse der Variable x im Speicher
Beispiel:
```

	Adresse	RAM
x	0x20000000	127
y	0x20000004	0x20000000

```
printf("%d %x %d\r\n",x,(int)y,*y);
liefert folgende Ausgabe: 127 0x20000000 127
```

Operatoren

Mathematische Operatoren		Priorität	Vergleichs- und logische Operatoren	
++	Inkrement	Höchste	!	NOT
--	Dekrement		>	Größer
-	Vorzeichen		>=	Größer gleich
*	Multiplikation		<	Kleiner
/	Division		<=	Kleiner gleich
%	Modulo, Rest der Division	Niedrigste	==	Gleich
+	Plus		!=	Ungleich
-	Minus		&&	AND
				OR

Da ein Gleichheitszeichen in C ein Zuweisungsoperator ist, weist man z.B. mit `x = 10;` der Variablen x den Wert 10 zu.

Bitweise Operatoren	
&	UND
	ODER
^	EXOR
~	Einerkomplement
<<	Nach links schieben
>>	Nach rechts schieben

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Kurzschreibweisen	
+=	x += 3; wie x = x + 3
-=	x -= 3; wie x = x - 3
*=	x *= 5; wie x = x * 5
/=	x /= 7; wie x = x / 7

Beispiele	Ergebnisse
x = 10;	
y = ++x;	y = 11
y = x++;	y = 10
y = 0x11;	
y = y<<1;	y = 0x22
(Bitweise um 1 nach links schieben)	

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

4.2 Schleifen

FOR-Schleife (zählergesteuerte Schleife)

Schleife, mit einer genau berechenbaren Anzahl an Wiederholungen.

```
for (<zaehlvariable=startwert>;<bedingung>;<schrittweite>) {
    ...
}
```

- **startwert** Anfangswert der Zählvariablen
- **bedingung** Schleife wird so lange durchlaufen, wie die Bedingung wahr ist
- **schrittweite** Anweisung zum Erhöhen oder Erniedrigen der Zählvariablen

Beispiel:

```
// DigitalOut ausgang(PC_0) 10x invertieren
for (int i=0; i<10; i++) {
    ausgang = !ausgang;
}
```

WHILE-Schleife (kopfgesteuerte Schleife)

Schleife, die wiederholt wird, so lange die Bedingung am Schleifenanfang erfüllt ist.

```
while (<bedingung>) {
    ...
}
```

Solange die am Anfang stehende **Bedingung erfüllt ist**, wird die Schleife wiederholt. Die Prüfbedingung steht **vor den Anweisungen**, sie heißt deshalb **kopfgesteuerte Schleife**.

Wenn die am Schleifenanfang stehende **Bedingung nicht erfüllt ist**, dann wird die gesamte Schleife übersprungen.

Beispiel:

```
// Solange der Taster DigitalIn taster(PA_1) gedrückt ist, wird der
// Ausgang DigitalOut ausgang(PC_0) invertiert
while (taster==true) {
    ausgang = !ausgang;
}
```

Do-WHILE-Schleife (fußgesteuerte Schleife)

Schleife, die mindestens einmal durchlaufen wird, da erst am Ende der Schleife mit der Überprüfung der Bedingung entschieden wird, ob die Schleife wiederholt werden muss.

```
do {
    ...
} while (<bedingung>;
```

Beispiel:

```
// Die Schleife wird maximal 100 mal und minimal 1 mal durchlaufen. Sie wird
// früh-
// zeitig abgebrochen, wenn der Taster DigitalIn taster(PA_1) gedrückt (=1) wird.
x = 100;
do {
    x--;
} while ((x>0) && taster==0);
```

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

4.3 Programmverzweigungen

Einfache Verzweigung mit if

Bei der if-Anweisung werden die Anweisungen innerhalb des if-Blocks nur dann ausgeführt, falls die Bedingung wahr ist.

```
if (<bedingung>) {
    <anweisung1>;
    <anweisung2>;
    ...
}
```

Beispiel:

```
// Wenn taster1 gedrückt ist, soll ausgang1 eins und ausgang2 null werden.
// Drückt man dagegen taster2, wird nur ausgang2 zu eins.
if (taster1==1) {
    ausgang1 = 1;        // Block mit mehreren Anweisungen wird ausgeführt,
    ausgang2 = 0;        // wenn die Bedingung hinter if wahr ist
}
if (taster2==1)
    ausgang2 = 1;        // Nur eine Anweisung, keine {} notwendig
```

Zweiseitige Verzweigung mit if

Bei der if/else-Anweisung kann zwischen **zwei Alternativen** entschieden werden. Ist die Bedingung wahr, so wird die erste Alternative (if-Block), ansonsten die zweite Alternative (else-Block) an Anweisungen ausgeführt.

```
if (<bedingung>) {
    <anweisung1>;
    <anweisung2>;
    ...
} else {
    <anweisung3>;
    <anweisung4>;
    ...
}
```

Beispiel:

```
// Wenn taster1 gedrückt ist, soll ausgang1 eins und ausgang2 null werden,
// andernfalls soll ausgang1 null und ausgang2 eins werden.
if (taster1==1) {
    ausgang1 = 1;        // Block mit mehreren Anweisungen wird ausgeführt,
    ausgang2 = 0;        // wenn die Bedingung hinter if wahr ist
} else {
    ausgang1 = 0;        // Block mit mehreren Anweisungen wird ausgeführt,
    ausgang2 = 1;        // wenn die Bedingung hinter if nicht wahr ist
}
```

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Mehrere Verzweigungen mit if

```

if (<bedingung1>) {
    <anweisung1>;
    ...
} else if (<bedingung2>) {
    <anweisung2>;
    ...
} else {
    <anweisung3>;
    ...
}

```

Fallunterscheidung mit switch

Mit der switch-Anweisung kann aus einer **Reihe von Alternativen** ausgewählt werden. Es ist zulässig, dass mehrere Möglichkeiten gültig sind und dieselbe Wirkung haben. Sie werden nacheinander aufgelistet. Passt keine der Möglichkeiten, dann wird die **default**-Einstellung ausgeführt.

Achtung! Auf keinen Fall **break** vergessen!!!

```

switch (<vergleichswert>) {
    case <wert1>:
        <anweisung1>;
        <anweisung2>;
        ...
        break;
    case <wert2>:
        <anweisung3>;
        <anweisung4>;
        ...
        break;
    ...
    default:
        <anweisung5>;
        ...
}

```

Beispiel:

```

// In der Variablen ergebnis ist ein Messergebnis oder eine Zahl gespeichert.
// Abhängig vom genauen Wert sollen nun bestimmte Reaktionen erfolgen.
switch (ergebnis) {
    case 0x00:
    case 0x10:
    case 0x20:
        ausgang1 = 1;
        break;
    case 0x30:
        ausgang1 = 0;
        break;
    case 0x40:
        ausgang1 = ~ausgang1;
        break;
    default:
        ausgang2 = 1;
        break;
}

```

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Hinweis: **Switch-Variablen** müssen einen **einfachen Datentyp** verwenden. Hinter **case** müssen **Konstanten** stehen. Dies können mit #define am Anfang des Programms deklariert werden.

Beispiel:

```
# define rechts  0x10      // ohne Semikolon!!
# define links   0x20
# define rechtskurve 0b0100
# define linkskurve 0b1000

unsigned char richtung;
...

switch (richtung) {
    case rechts:
        motor = rechtskurve;
        break;
    case links:
        motor = linkskurve;
        break;
    default:
        motor = vorwaerts;
        break;
}
```

4.4 Operationen (Unterprogramme, Funktionen)

Deklaration von Operationen

Beispiele:

```
void addieren(void);           // ohne Rückgabewert, ohne Parameter
void zeitms(int msec);        // ohne Rückgabewert, mit Parameter
float berechneQuadrat(float pQ); // mit Rückgabewert, mit Parameter
```

Definition von Operationen

Beispiel:

```
int a, result;                // globale Variablen
void addieren(void) {          // Operationsname
    result = a + a;             // Anweisung(en)
}
```

Operationen mit Übergabewert

Beispiel:

```
void zeitms(int msec) {        // Übergabewert msec
    int t1;                    // lokale Variable
    for (t1=msec;t1!=0;t1--)
        wait_us(1000);         // Zeitschleife;
}
```

Operationen mit Rückgabewert

Beispiel:

```
float berechneQuadrat(float pQ=10) { // Parameter mit Standardwert
    return pQ*pQ;                     // Rückgabewert
}
```

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

4.5 Beispiel eines C/CPP-Programms

```
volatile bool Anforderung_Fussgaenger = false; //globale Variablen
int z;
int S1 = PA4;
int LED_rt = D1, LED_ge = D2, LED_gn = D3;
char phasen[4]= { //array
    0b001, //0 rot
    0b101, //1 rotgelb
    0b010, //2 grün
    0b100}; //3 gelb

void setup()
{
    pinMode(LED_rt, OUTPUT);
    pinMode(LED_ge, OUTPUT);
    pinMode(LED_gn, OUTPUT);
    pinMode(S1, INPUT_PULLUP);
    z=0;
}

void loop()
{
    if (digitalRead(S1) == LOW)
    {
        Anforderung_Fussgaenger = true;
    }
    digitalWrite(LED_rt,HIGH);
    digitalWrite(LED_gn,LOW);
    digitalWrite(LED_ge,LOW);
    delay(500);
    digitalWrite(LED_ge,HIGH);
    delay(500);
    digitalWrite(LED_gn, HIGH);
    digitalWrite(LED_rt, LOW);
    digitalWrite(LED_ge, LOW);
    delay(500);
    digitalWrite(LED_gn, LOW);
    digitalWrite(LED_ge, HIGH);
    delay(500);
    if (Anforderung_Fussgaenger) //Fußgängeranforderung auswerten
    {
        digitalWrite(LED_rt, HIGH);
        delay(500);
        Anforderung_Fussgaenger = false;
        digitalWrite(LED_rt, LOW);
    }
    z++;
    if (z>8)
    {
        z=0;
    }
}
```

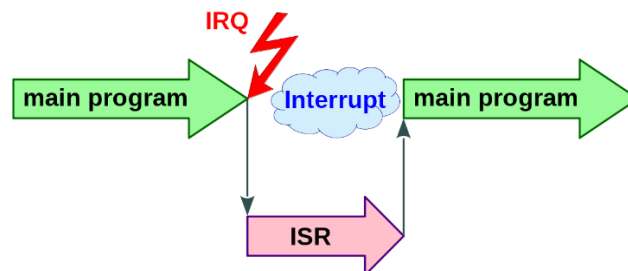
Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

4.6 On Chip Peripherie

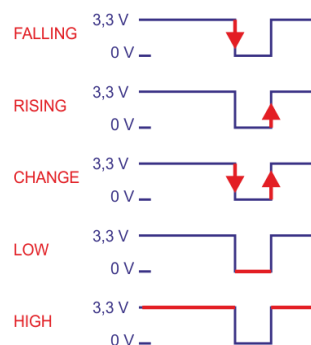
Portpin: Eingabe und Ausgabe

	Befehl	Beispiel
Portausgabe: Bsp.: <code>GPIOA->ODR = 0xFF00; // Highbyte auf HIGH gesetzt...</code>		
Einzelbitausgabe		
Deklaration	<code>#define Name Pin</code> Name = Pin-Bezeichnung	<code>#define LED_D12 D12</code>
Konfiguration	<code>pinMode(LED_D12, OUTPUT);</code>	
Verwendung	<code>digitalWrite(name, WERT)</code> WERT=HIGH, LOW	<code>digitalWrite(LED_rt,HIGH);</code>
Einzelbiteingabe		
Deklaration	<code>#define Name Pin</code> Mögliche Werte für Pin: PA0..PA15, PB0..PB15, PC0..PC15 oder D1, A0...	<code>#define S2 PA4</code>
Konfiguration	<code>pinMode(name, konfig);</code> Mögliche Werte für konfig: = INPUT_PULLUP, INPUT_PULLDOWN, INPUT	<code>pinMode (S2, INPUT);</code>
Verwendung	Var = Pin Zustand lesen	<code>buttonState = digitalRead(S2);</code>

4.7 Externer Interrupt



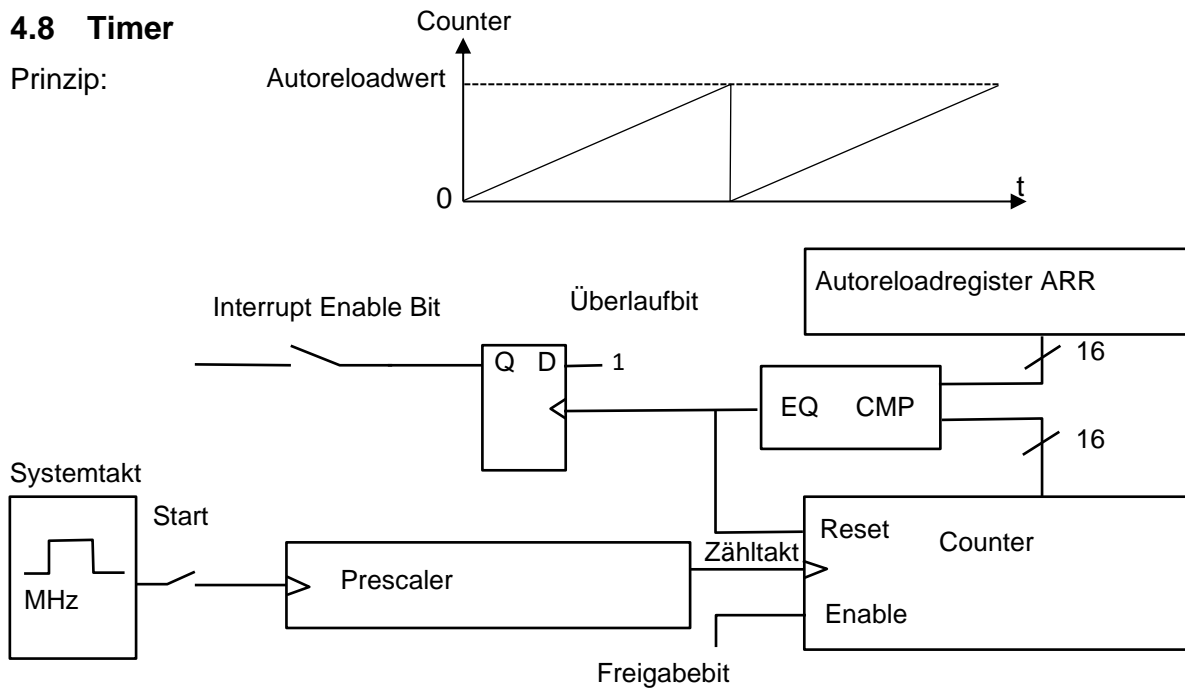
	Befehl	
Externer Interrupt		
Deklaration	<code>#define Name Pin</code> Mögliche Werte für Portpin: PA0..PA15, PB0..PB15, PC0..PC15, oder D1, A0...	<code>#define S2 PA4</code>
Konfiguration	<code>attachInterrupt (digitalPinToInterrupt (PIN), ISR_Name, Aktion);</code>	
Bsp.:	<code>attachInterrupt (digitalPinToInterrupt (S2), ISR_EXT_IR, FALLING);</code>	
	Aktion: FALLING, RISING, CHANGE, HIGH, LOW	
Hinweis:	Variable(n) in der ISR sollten als <code>volatile</code> deklariert werden	



Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

4.8 Timer

Prinzip:



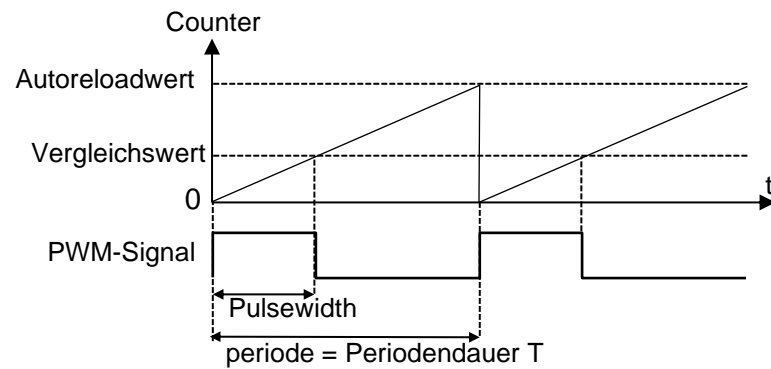
Beispiel STM32L152

Maßnahme	Syntax
Timerauswahl:	<code>static HardwareTimer mytimer = HardwareTimer(TIM3);</code>
TimerÜberlauf nach Zeit konfigurieren	<code>mytimer.setOverflow(2000, MICROSEC_FORMAT);</code> [≤ 16 Bit] \Rightarrow Wenn Counter-Reg und ARR gleich sind
TimerÜberlauf nach Frequenz konfigurieren	<code>mytimer.setOverflow(2000, HERTZ_FORMAT);</code> [≤ 16 Bit]
zusätzlich Vorteiler nutzen:	<code>mytimer.setPrescaleFactor(100);</code> [≤ 16 Bit]
TimerInterrupt aktivieren und ISR aufrufen	<code>mytimer.attachInterrupt(ISR_Timer);</code>
TimerInterrupt deaktivieren	<code>mytimer.detachInterrupt();</code>
Timer starten	<code>mytimer.resume();</code>
Timer stoppen	<code>mytimer.pause();</code>
Hinweis:	Variable(n) in der ISR sollten als <code>volatile</code> deklariert werden

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

4.9 Puls-Weiten-Modulation (PWM)

Prinzip:

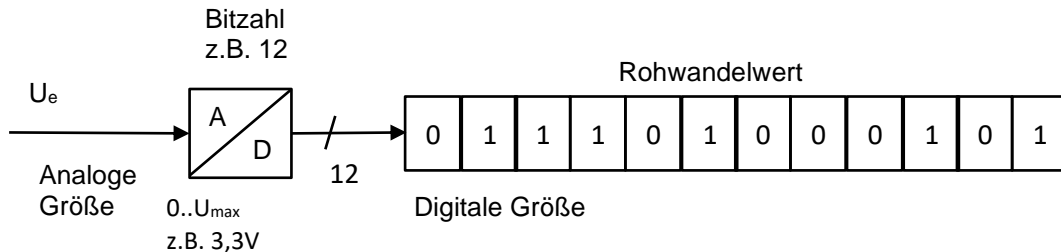


PWM	Befehl	Beispiel
Deklaration	<code>#define name (Portpin)</code>	<code>#define RGB_b1 D10</code>
Konfiguration f-Frequenz in Hz => 1/Periodendauer	<code>analogWriteFrequency(var);</code>	<code>analogWriteFrequency(2000);</code> //entspricht 2KHz
Bitbreite Pulsweite	<code>analogWriteResolution(8-16);</code>	<code>analogWriteResolution (16);</code>
Verwendung	<code>analogWrite(Pinname, Pulsweite);</code>	<code>analogWrite(RGB_r, 200);</code>

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

4.10 Analog – Digital – Wandlung

AD-Wandler	Befehl	Beispiel
Deklaration	Datentyp Name Pin	#define A0_Pin = A0;
	Mögliche Werte für Portpin: A0-A5	
Verwendung	Var = analogRead(Pin)	sensorValue = analogRead(A0_Pin);

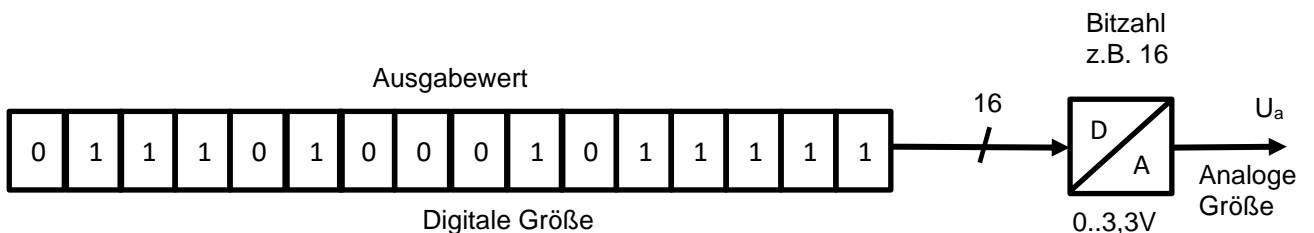


Berechnungsformeln

$$\begin{aligned} \text{Rohwandelwert} &= \frac{U_e}{U_{\max}} \cdot (2^{\text{Bitzahl}} - 1) && \text{z.B. } \frac{U_e}{3,3V} \cdot 4095 \\ \text{Wandelwert als Kommazahl: } x &= (U_e / U_{\max}) && \text{z.B. } x = (U_e / 3.3) \\ \text{Stufung (analoge Auflösung): } &U_{\max} / 4095 && \text{z.B. } 3,3V / 4095 \\ \text{Wandelwert als Ganzzahl linksbündig: } &\text{unsigned short } x = (U_e / U_{\max}) * 65535 \end{aligned}$$

Digital – Analog – Wandlung

	Befehl	Beispiel
Digital-Analog Wandler		
Deklaration	AnalogOut meinAnalogOut(Portpin);	AnalogOut ausgang(ledPin);
	Mögliche Werte für Portpin: PA5	
Verwendung	analogWrite(Pin, Wert)	analogWrite(ledPin, outputValue);



Berechnungsformeln

$$\begin{aligned} \text{float } x: & U_a = x \cdot 3,3V \\ \text{uint16_t: } & U_a = \frac{x \cdot 3,3V}{65535} \end{aligned}$$

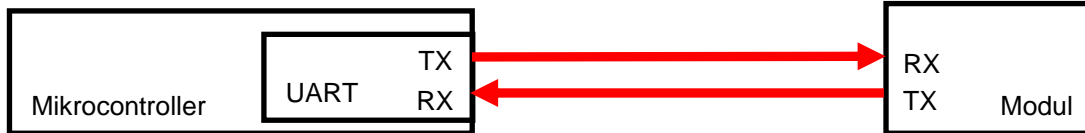
Werteanpassung / Skalierung mit map():

Werteanpassung = map(Wert, 0, 1023, 0, 255); // Werte von 0-1023 werden auf 0-255 angepasst

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

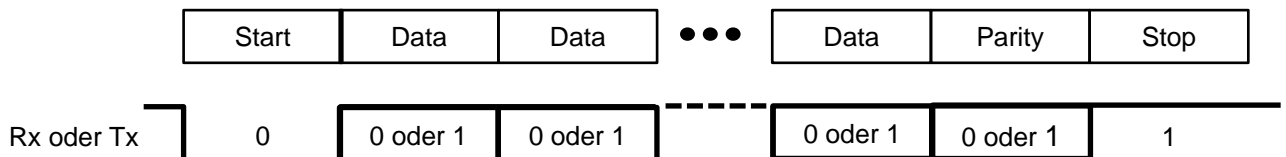
4.11 Externe Kommunikationsmöglichkeiten

Universal Asynchronous Receiver Transmitter (UART)



	Befehl	Beispiel
Universal Asynchronous Receiver Transmitter (UART)		
#include <SoftwareSerial.h> => Software Serial Modus		
Deklaration	<pre>#define softserial #define RX Pin? #define TX Pin?</pre>	
	SoftwareSerial SerialBsp(RX, TX);	
Verwendung	SerialBsp.begin(9600); // Baudrate	
Daten empfangen	Fragen nach daten im Serial-Buffer Wenn ja, dann in Variable lesen...	<pre>if (SerialBsp.available()) { msg = SerialBsp.readString(); }</pre>
Daten senden	Mit print(ln) String schreiben Mit print Var-wert schreiben	<pre>SerialBsp.print("LED an"); SerialBsp.print(LED);</pre>

Frame



Eine UART-Übertragung beginnt immer mit einem Startbit (Low). Darauf folgen

- 5-8 **Datenbits** (Standard = 8)
- 0 oder 1 **Paritybit** (Standard = 0 none)
- 1 oder 2 **Stopbits** (Standard = 1)

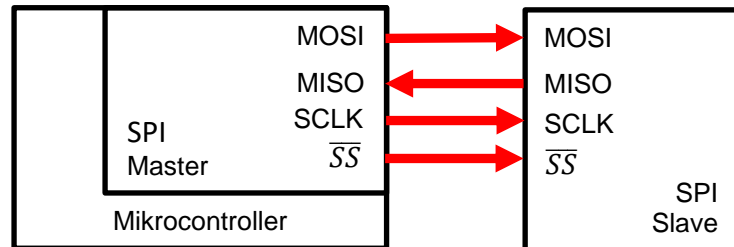
Falls ein Paritybit programmiert wurde, kann es gerade Parity (even) oder ungerade Parity (odd) anzeigen.

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Serial Peripheral Interface (SPI)

Das **Serial Peripheral Interface (SPI)** dient der Kommunikation des Mikrocontrollers mit **Modulen** auf der Platine. Module sind

- Anzeigen,
- Speicher,
- LAN-Bausteine
- ...



Signale

Master/Slave (OLD)	Controller/Peripheral (NEW)
Master In Slave Out (MISO)	Controller In, Peripheral Out (CIPO)
Master Out Slave In (MOSI)	Controller Out Peripheral In (COPI)
Slave Select pin (SS)	Chip Select Pin (CS)

Sendeleitung

Empfangsleitung

Auswahl Slaves/Chip (Lowaktiv)

SCLK (Serial Clock):

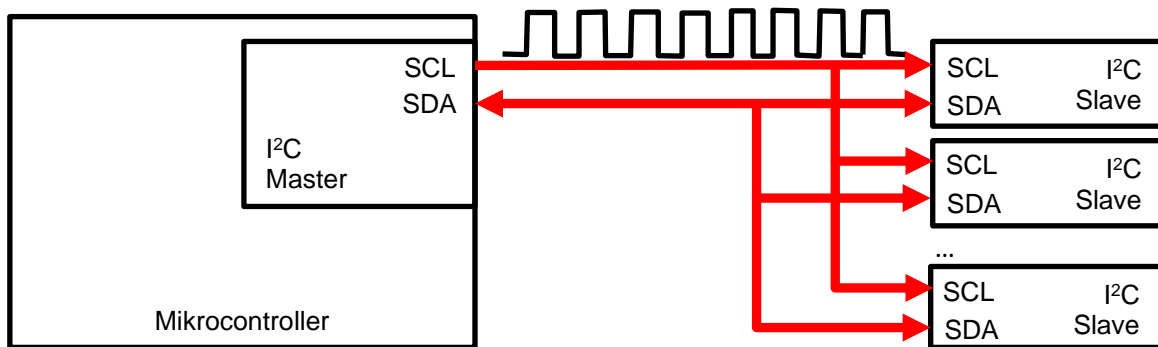
Taktleitung

Mode	Clock Polarity (CPOL)	Clock Phase (CPHA)	Output Edge	Data Capture
SPI_MODE0	0	0	Falling	Rising
SPI_MODE1	0	1	Rising	Falling
SPI_MODE2	1	0	Rising	Falling
SPI_MODE3	1	1	Falling	Rising

	Befehl	Beispiel
	#include <SPI.h> //Bibliothek einbinden	
Initialisiere SPI	SPI.begin()	
	SPISettings(14000000, MSBFIRST, SPI_MODE0) f, dataOrder, Modus Oder: SPI.setBitOrder(MSBFIRST); SPI.setClockDivider(SPI_CLOCK_DIV32); SPI.setDataMode(SPI_MODE3); SPI.end();	
Konfiguration CS-Pin	#define chipSelectPin (CS) = D5	
	pinMode(chipSelectPin, OUTPUT);	
Verwendung	SPI.begin(); digitalWrite(CS, LOW); SPI.transfer(address_w); SPI.transfer(0x09); SPI.transfer(Bitmuster); digitalWrite(CS, HIGH); SPI.end();	

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Inter-Integrated Circuit (I²C) SCL (Serial Clock): Taktleitung SDA (Serial Data): Datenleitung



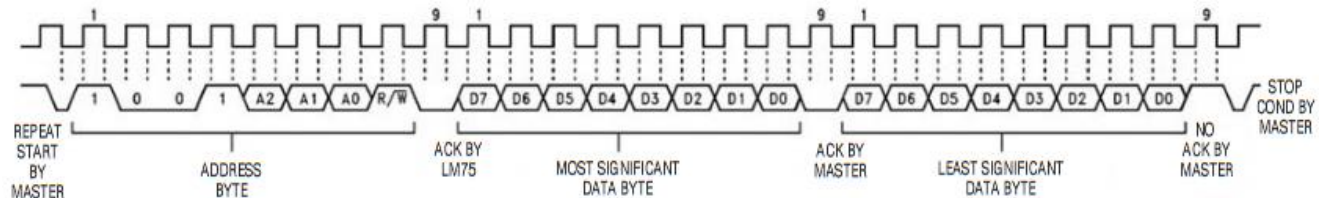
I ² C	Befehl	Beispiel
	#include <Wire.h> => I2C-Library	JE NACH Datenblatt!
Deklaration	#define adress => 7 Bit!	address 0b0100000 //0x40
Initialisierung	Bibliothek starten I2C-Frequenz einstellen	Wire.begin(); Wire.setClock(10000);
Verwendung	I2C-Übertragung starten inkl. Adresse Baustein	Wire.beginTransmission(address);
Daten senden	Daten auf Bus schreiben Übertragung abschließen	Wire.write(0x09); Wire.endTransmission();
Daten empfangen	I2C-Übertragung starten inkl. Adresse Baustein Daten auf Bus schreiben	Wire.beginTransmission(address); Wire.write(0x00); Wire.endTransmission();
	Nach Empfangenen Daten fragen... Var = daten lesen Übertragung beenden	Wire.requestFrom(address, 2); Temp_H = Wire.read(); Temp_L = Wire.read(); Wire.endTransmission();

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Beispielhaft aufgeführte I²C-Bausteine bzw. Auszug Datenblätter Quelle: www.alldatasheet.com

LM 75:

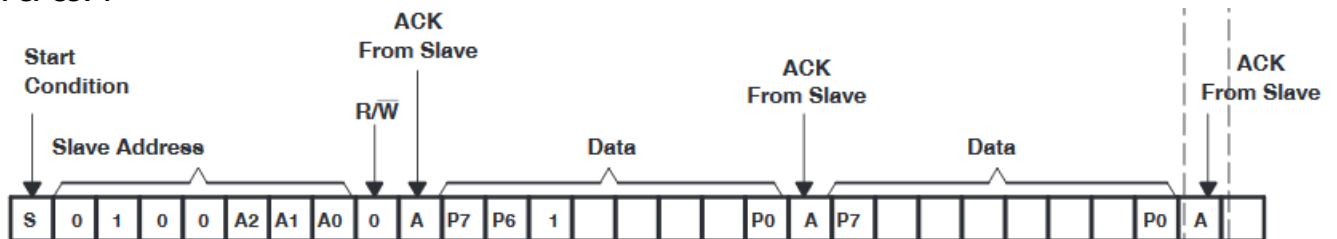
BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
1	0	0	1	A2	A1	A0	RD/W



UPPER BYTE								LOWER BYTE							
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Sign bit 1= Negative 0 = Positive	MSB 64°C	32°C	16°C	8°C	4°C	2°C	1°C	LSB 0.5°C	X	X	X	X	X	X	X

X = Don't care.

PCF 8574



Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

5 Programmentwicklung und Objektorientierter Entwurf

5.1 Vergleichsoperatoren für Bedingungen (Pseudocode)

<, <=, >, >=, == oder =, ≠ oder !=

Anmerkung: Die Operatoren für Vergleiche und Wertzuweisungen müssen unterschieden werden können.

5.2 Kontrollstrukturen (Pseudocode)

Zuweisung

```
dieVariable ← derAusdruck
dieVariable := derAusdruck
dieVariable = derAusdruck
```

Sequenz

```
anweisung1
anweisung2
anweisung3
```

Auswahl

Einseitige Auswahl

```
WENN bedingung
    anweisung1
...
ENDE WENN
```

Zweiseitige Auswahl

```
WENN bedingung
    anweisungA1
...
SONST
    anweisungB1
...
ENDE WENN
```

Mehrfachauswahl

```
FALLS variable GLEICH
    bedingung1: anweisungA1
    ...
    bedingung2: anweisungB1
    ...
    bedingung3: anweisungC1
    ...
    SONST: anweisungD1
    ...
ENDE FALLS
```

Schleife (Iteration)

Schleife mit Eintrittsbedingung

```
SOLANGE bedingung
    anweisung1
...
ENDE SOLANGE
```

Schleife mit Austrittsbedingung

```
WIEDERHOLE
    anweisung1
...
SOLANGE bedingung
```

Zählschleife

```
FÜR i ← 0 BIS n SCHRITT s
    anweisung1
...
ENDE FÜR
```

Schleife über Kollektion

```
FÜR element IN kollektion
    anweisung1
...
ENDE FÜR
```

Schleife mit Abbruchbedingung

```
FÜR element IN kollektion
    anweisungA1
    ...
    WENN bedingung
        ABBRUCH
    ENDE WENN
    anweisungB1
    ...
ENDE FÜR
```

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

5.3 Datentypen

Elementare Datentypen

Datentyp	Abkürzungen	Werte
Boolscher Datentyp	Boolean, boolean, bool, ...	wahr, falsch, true, false
Ganzzahliger Datentyp	GZ, Integer, int, ...	-24, 0, 123, ...
Fließkomma-Datentyp	FKZ, Real, double, ...	-3.567, 0.0, 3.141, ...
Zeichen-Datentyp	Zeichen, char, ...	'Z', 'a', '&', ...
Text-Datentyp	Text, String, string, ...	"Hello world!!!", ...

Für den Datentyp Text ist als Vergleichsoperator nur == bzw. = definiert. Außerdem kann der Operator + für die Verbindung von zwei Texten verwendet werden. Auch bei Texten muss der Vergleich und die Zuweisung eindeutig unterschieden werden können (vgl. 4.1).

Komplexe Datentypen

Zeit
...
+Zeit() +Zeit(pStunde:GZ,pMinute:GZ,pSekunde:GZ) +gibStunde():GZ +gibMinute():GZ +gibSekunde():GZ +istVor(pZeit:Zeit):Boolean +istNach(pZeit:Zeit):Boolean +zeitMinusSekunden(pSekunden:GZ):Zeit +zeitPlusSekunden(pSekunden:GZ):Zeit +gibText():Text

Datum
...
+Datum() +Datum(pTag:GZ,pMonat:GZ,pJahr) +gibTag():GZ +gibMonat():GZ +gibJahr():GZ +istVor(pDatum:Datum):Boolean +istNach(pDatum:Datum):Boolean +anzahlTageBis(pDatum:Datum):GZ +anzahlTageSeit(pDatum:Datum):GZ +gibText():Text

Liste<Typ>
...
+Liste<Typ>() +gibLaenge():GZ +gib(pIndex:GZ):Typ +ersetzen(pIndex:GZ,pElement:Typ) +einfuegen(pIndex:GZ,pElement:Typ) +anhaengen(pElement:Typ) +verketteten(pListe>Liste<Typ>) +entfernen(pIndex:GZ):Typ +entfernen(pElement:Typ) +enthaelt(pElement:Typ):Boolean +kopieren():Liste<Typ> ...

Listen beinhalten Daten vom gleichen Typ. Dabei kann es sich um elementare oder komplexe Datentypen (Klassen) handeln, z.B. Liste<GZ> oder Liste<Person>.

Die Operationen ersetzen und einfuegen unterscheiden sich dadurch, dass beim Ersetzen das Element am Index pIndex ersetzt wird und die Liste somit ihre Länge behält, während beim Einfügen die Liste verlängert wird, da das Element pElement die nachfolgenden Elemente um eine Position nach hinten verschiebt.

Die Operation entfernen ist überladen. Wird sie mit einer ganzzahligen Löschesposition als Argument aufgerufen, gibt die Operation das gelöschte Objekt vom Datentyp Typ zurück. Wird entfernen mit einem Argument vom Datentyp Typ aufgerufen, wird dieses Objekt in der Liste von vorne gesucht und das erste gefundene Objekt, falls vorhanden, gelöscht.

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Alternative Notationen für Listen

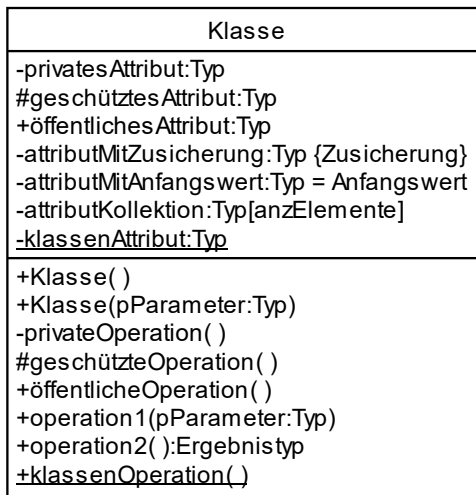
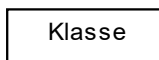
Liste highscore vom Datentyp Liste<GZ>

Standardnotation	Alternative Notation	Bedeutung
highscore ← NEU Liste<GZ>()	highscore ← []	Leere Liste anlegen.
highscore ← NEU Liste<GZ>() FÜR i←0 BIS 2 SCHRITT 1 highscore.anhaengen(0)	highscore ← [0, 0, 0]	Liste mit drei Elementen anlegen.
h ← highscore.gib(0)	h ← highscore[0]	Element einer Liste lesen.
highscore.ersetzen(3,5)	highscore[3] ← 5	Element einer Liste schreiben.

Notationen für Felder

Standardnotation	Bedeutung
highscore ← NEU GZ[10]	Feld für 10 Highscores anlegen.
highscore[0] ← 15	Ersten Highscore auf 15 setzen.

5.4 Klassen



Attribute

Die Bezeichner von Attributen beginnen mit einem Kleinbuchstaben (vgl. UML-Standard). Attribute haben im Klassendiagramm folgenden Aufbau:

Sichtbarkeit bezeichner:Typ<[Multiplizität]><=Anfangswert><{Zusicherung}>

Die in spitzen Klammern notierten Inhalte, z.B. <[Multiplizität]>, sind optionale Bestandteile der Attribute.

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Sichtbarkeit	Zeichen
privat	-
geschützt	#
öffentlich	+

Typ
Elementarer Datentyp
Komplexer Datentyp (Klasse)

Anfangswert
Wert den das Attribut bei der Erzeugung des Objekts annimmt.

Zusicherung
Vorschriften für Attribute {wert>0}, {read only}.

Operationen

Prozeduren bzw. Funktionen von Programmiersprachen nennt man im Kontext der Objektorientierung Operationen. Ihre Bezeichner starten, wenn möglich, mit einem Verb. Wie bei Attributen ist der erste Buchstabe ein Kleinbuchstabe. Operationen haben im Klassendiagramm folgenden Aufbau:

Sichtbarkeit operationsbezeichner(<Parameterliste><:Rückgabetyp>

Eine Parameterliste kann leer sein oder einen oder mehrere Parameter enthalten. Die Parameter werden nach folgendem Schema definiert:

pName:Typ, ...

Die in spitzen Klammern notierten Inhalte, z.B. <Parameterliste>, sind optionale Bestandteile der Operationsdeklaration.

Assoziationen, Rollennamen und Multiplizitäten



Gerichtete Assoziation

Bidirektionale Assoziation

Multiplizität	Bedeutung
1	genau 1
0..1	0 oder 1
3..6	3, 4, 5 oder 6
*	0 bis viele
2..*	2 bis viele

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Beispiel einer Operation mit einer Kollektion in Pseudocode

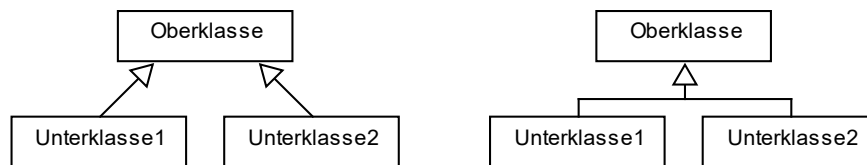
OPERATION anlegenPerson(pName:Text,personen:Liste<Person>):Boolean

Lokale Variablen: gefunden:Boolean, neuePerson:Person, person:Person

```

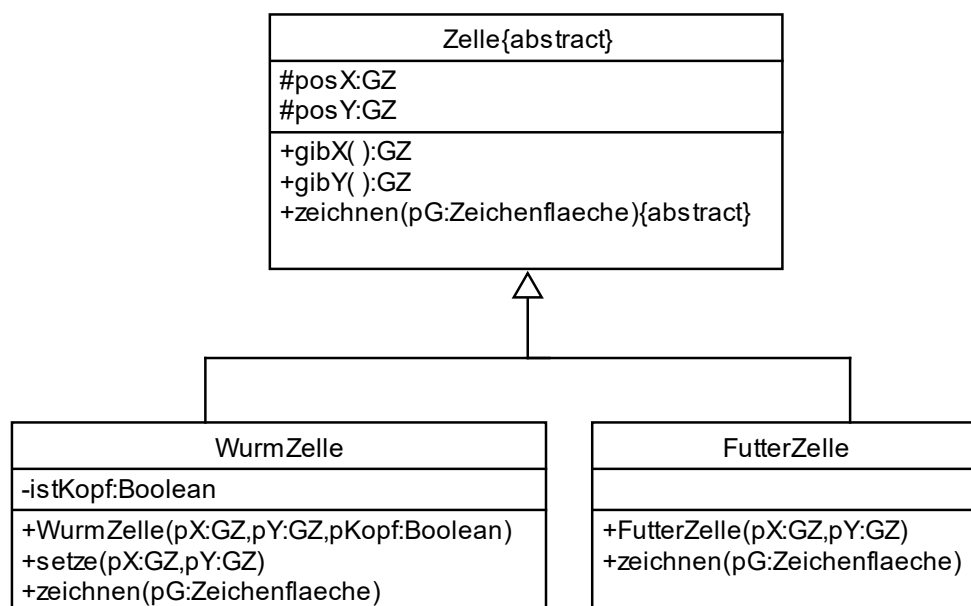
gefunden ← falsch
FÜR person IN personen
    WENN person.gibName() = pName
        gefunden ← wahr
        ABBRUCH
    ENDE WENN
ENDE FÜR
WENN gefunden = falsch
    neuePerson ← NEU Person(pName)
    personen.anhaengen(neuePerson)
ENDE WENN
RÜCKGABE gefunden
    
```

5.5 Vererbung

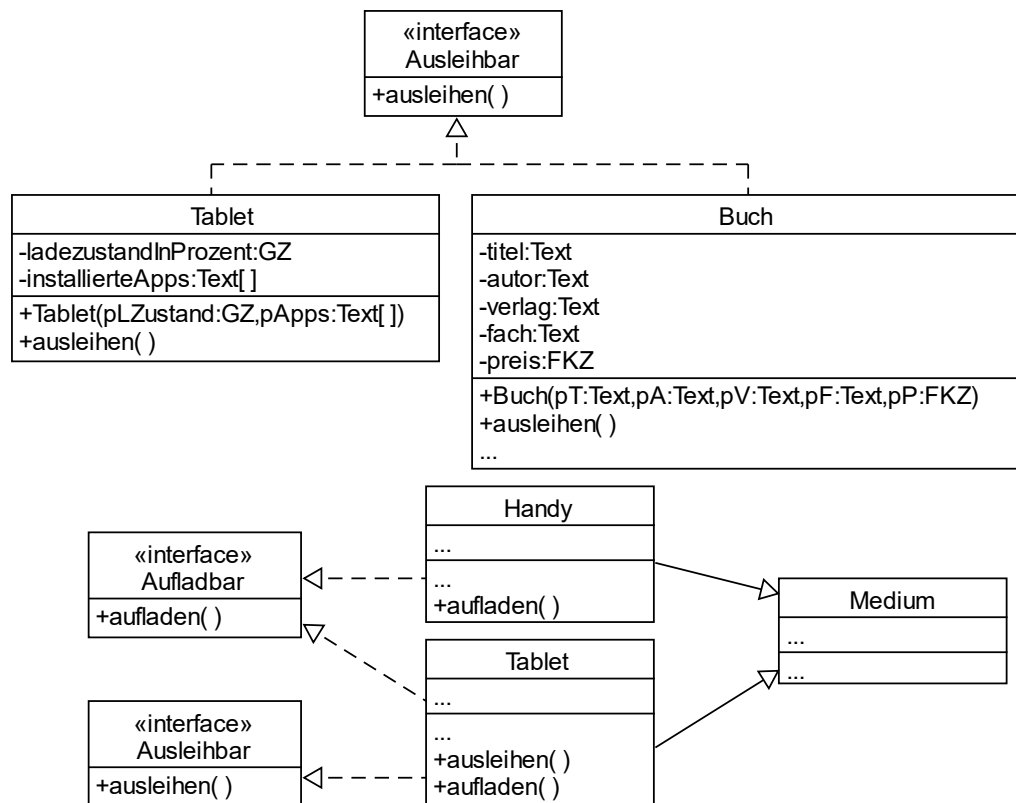


Oberklassen sind Generalisierungen und Unterklassen Spezialisierungen.

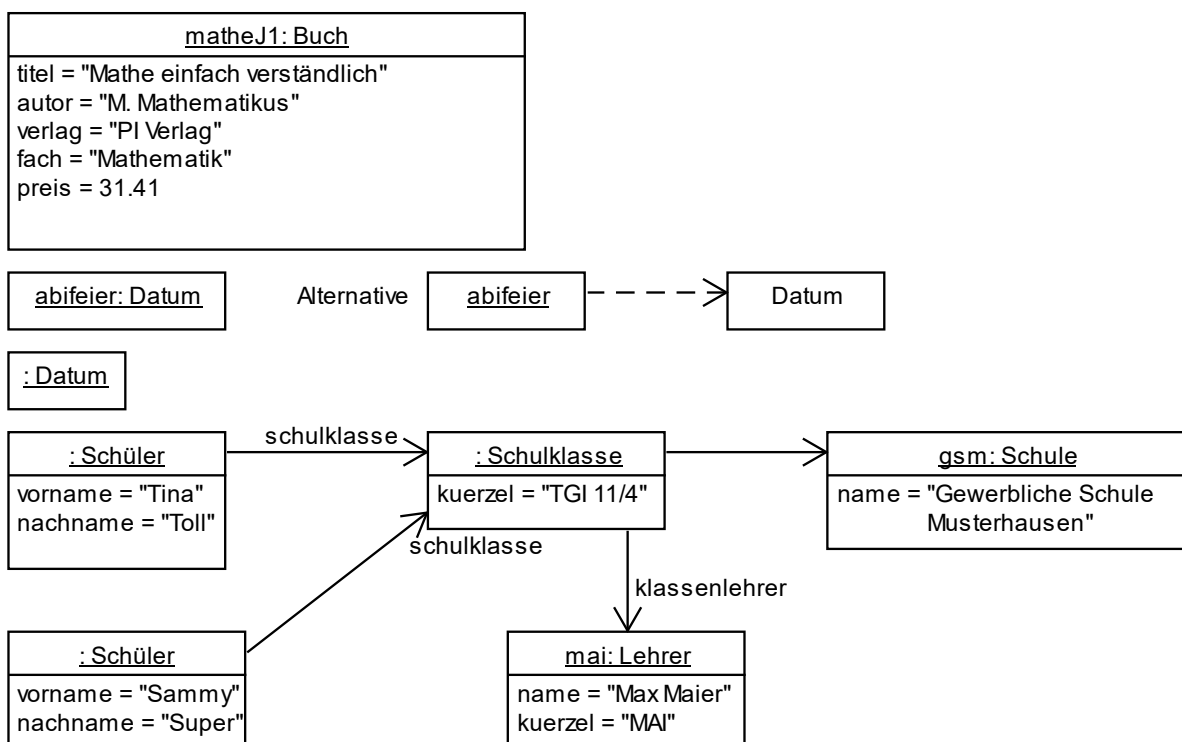
5.6 Abstrakte Klassen und Schnittstellen



Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik



5.7 Objektdiagramme



Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

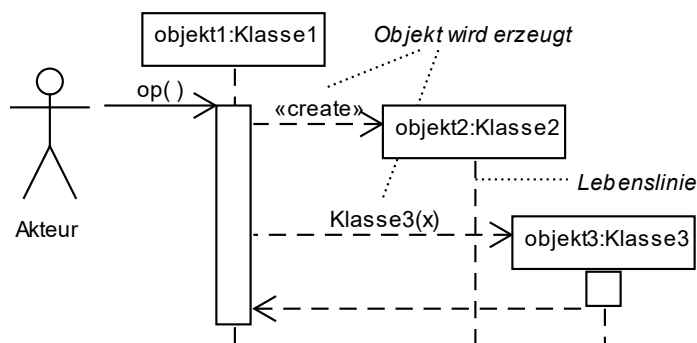
5.8 Sequenzdiagramme

Allgemeines:

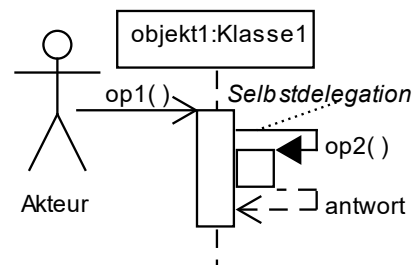
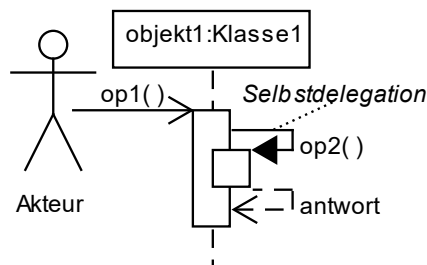
- Es wird nicht zwischen unterstrichenen und nicht-unterstrichenen Objekten im Sequenzdiagramm unterschieden.

Erzeugung von Objekten

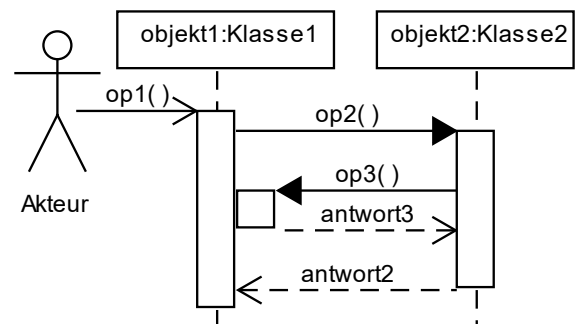
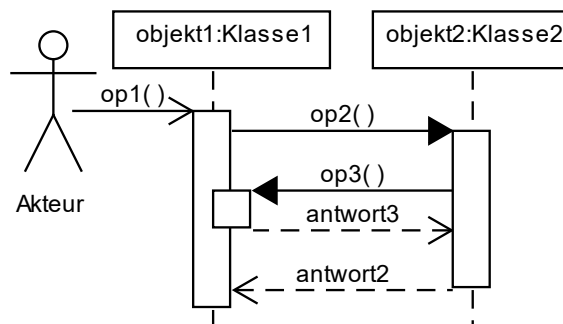
Ein Objekt kann im Sequenzdiagramm immer mit einem spezifischen Konstruktor erzeugt werden. Ist die Auswahl des Konstruktors nicht bedeutsam, so wird die Objekterzeugung durch `<<create>>` dargestellt.



Selbstdelegation (alternative Darstellungen)

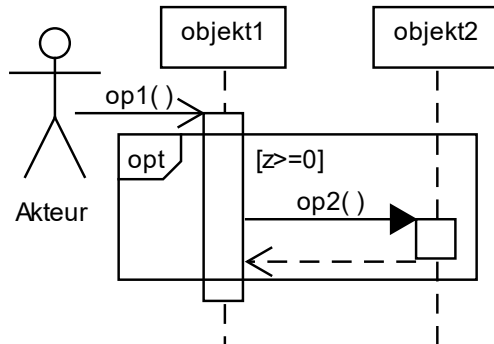


Wechselseitige Botschaften (alternative Darstellungen)

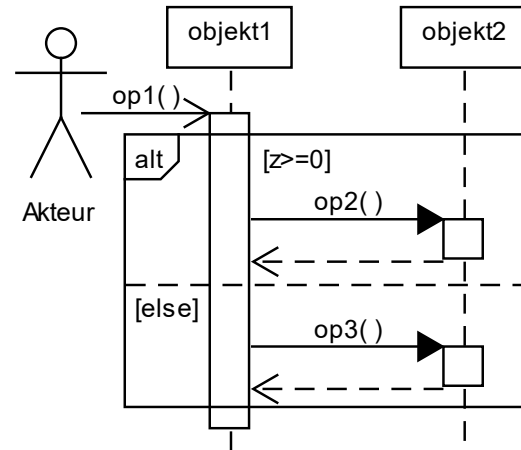


Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

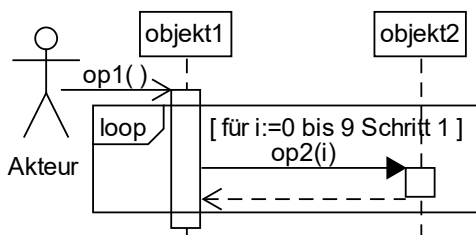
Option – einseitige Verzweigung



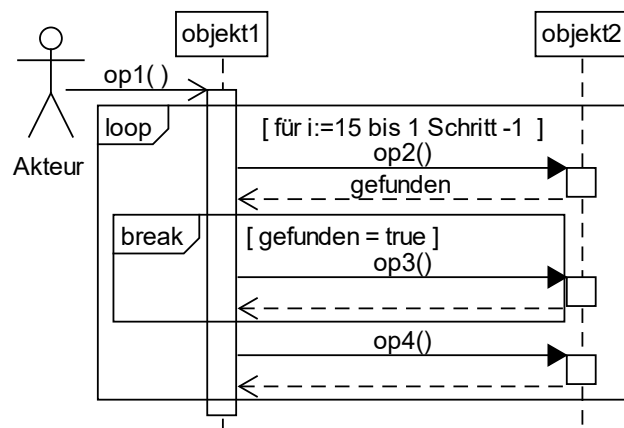
Alternative – mehrseitige Verzweigung



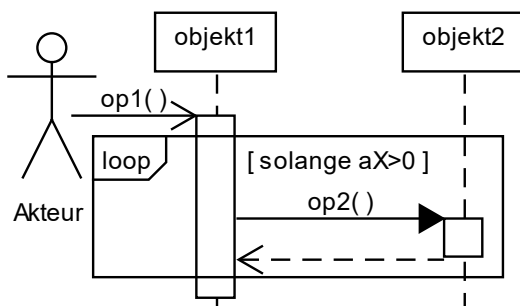
Zählschleife



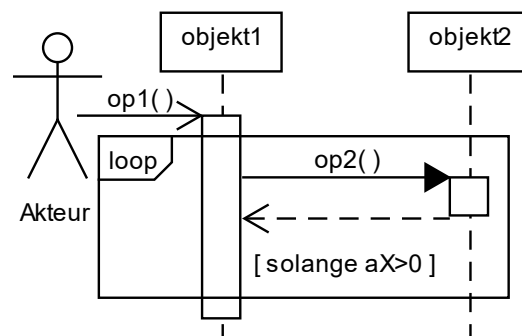
Schleife mit Abbruch



Kopfgesteuerte Schleife

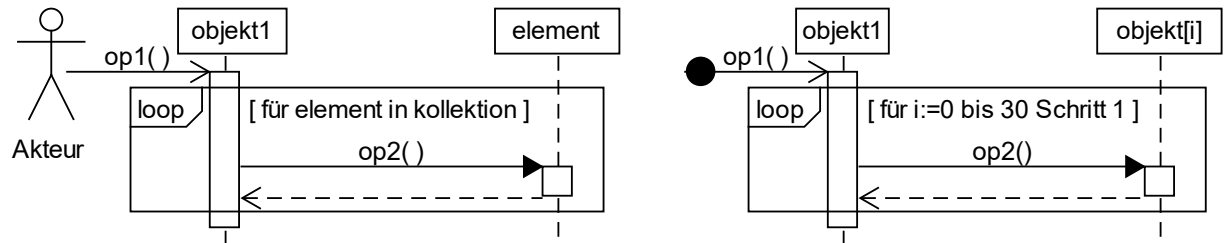


Fußgesteuerte Schleife



Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Schleife über Kollektion



● op1() → Nachricht, bei welcher der Sender nicht spezifiziert ist.

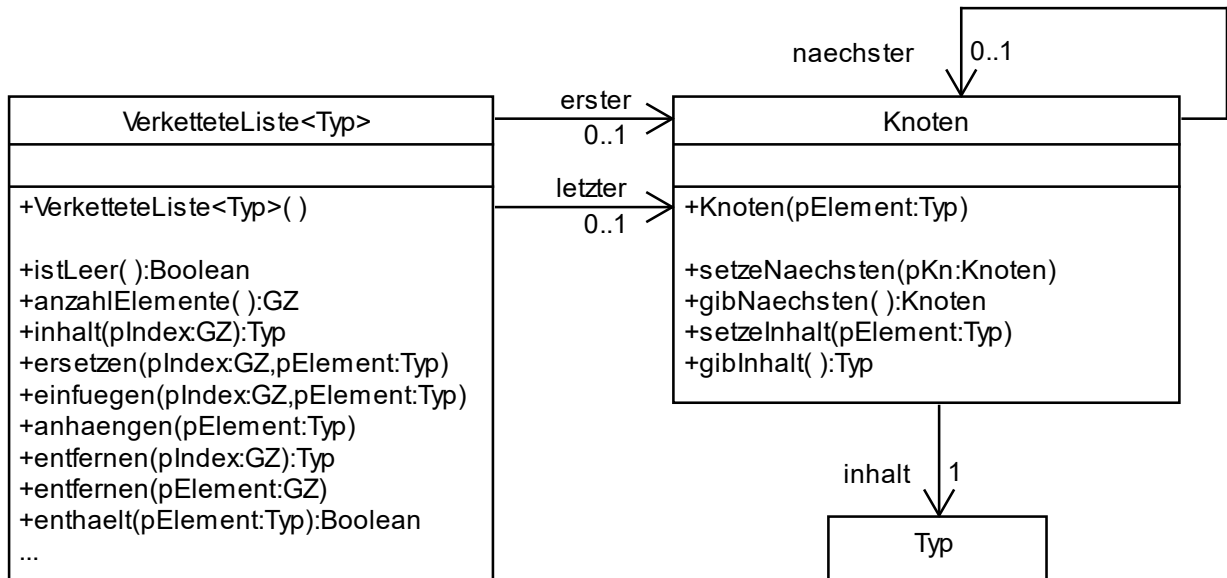
5.9 Zustandsdiagramme

Zustandsdiagramme siehe Kapitel 1

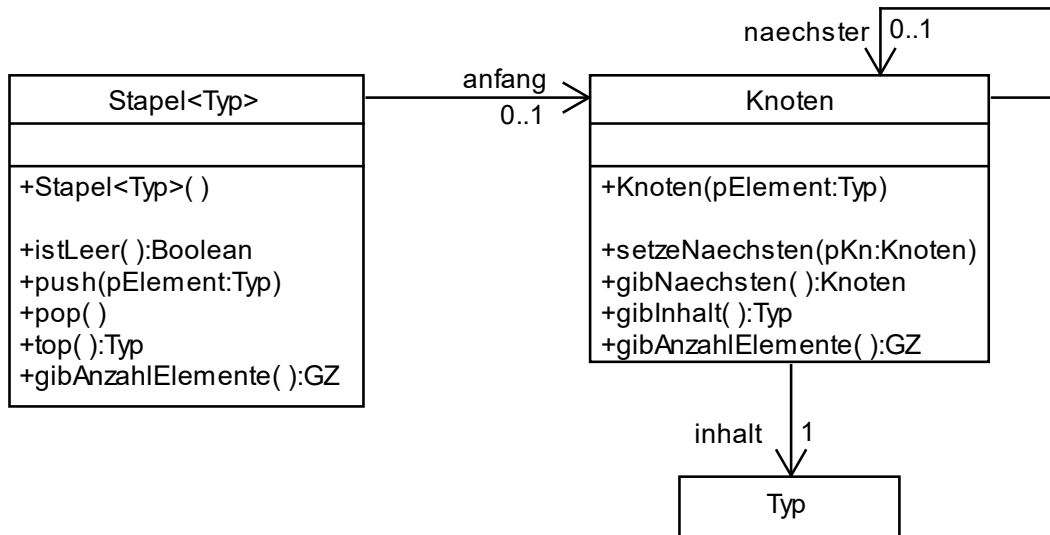
Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

6 Datenstrukturen

6.1 Verkettete Liste

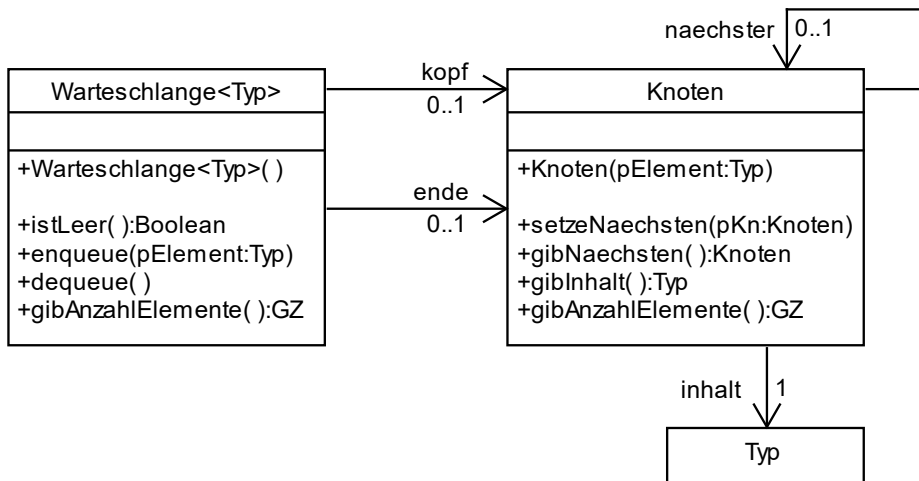


6.2 Stapel



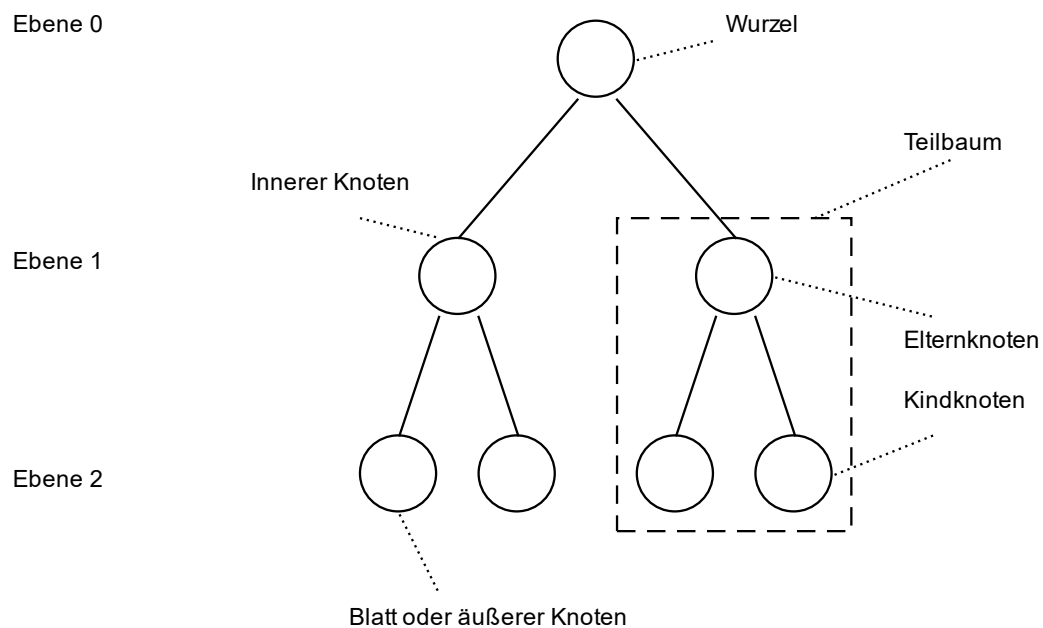
Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

6.3 Warteschlange



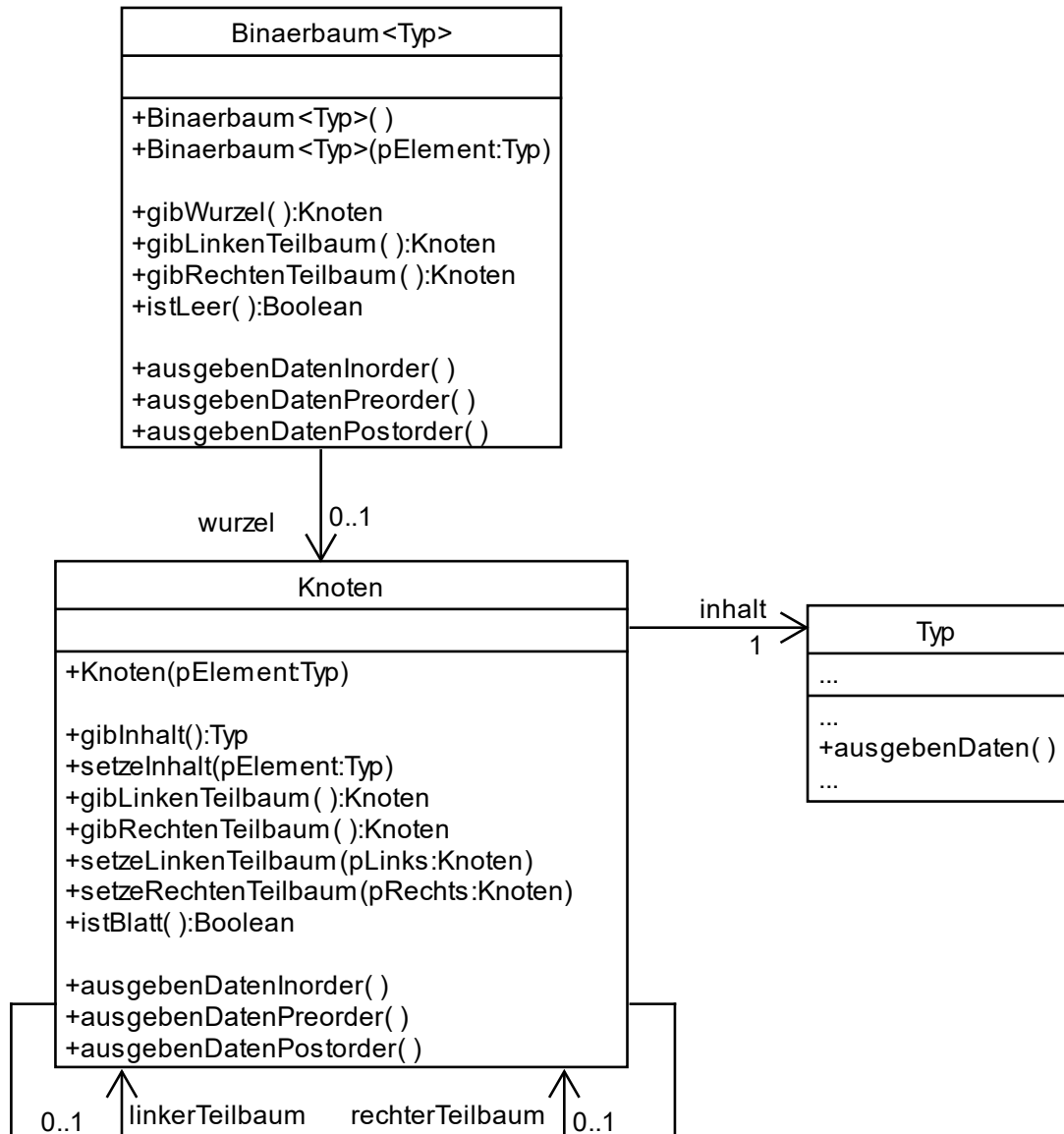
6.4 Binärbaum

Beispiel für einen Binärbaum der Tiefe 3



Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Datenstruktur



Operation **ausgebenDatenInorder()** der Klasse **Knoten** in Pseudocode

OPERATION **ausgebenDatenInorder()** der Klasse **Knoten**

```

WENN linkerTeilbaum != NICHTS
    linkerTeilbaum.ausgebenDatenInorder()
ENDE WENN
inhalt.ausgebenDaten()
WENN rechterTeilbaum != NICHTS
    rechterTeilbaum.ausgebenDatenInorder()
ENDE WENN
    
```

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

7 Künstliche Intelligenz

7.1 Klassifikation

Distanzfunktionen für $x = (x_1, \dots, x_n)$ und $y = (y_1, \dots, y_n)$

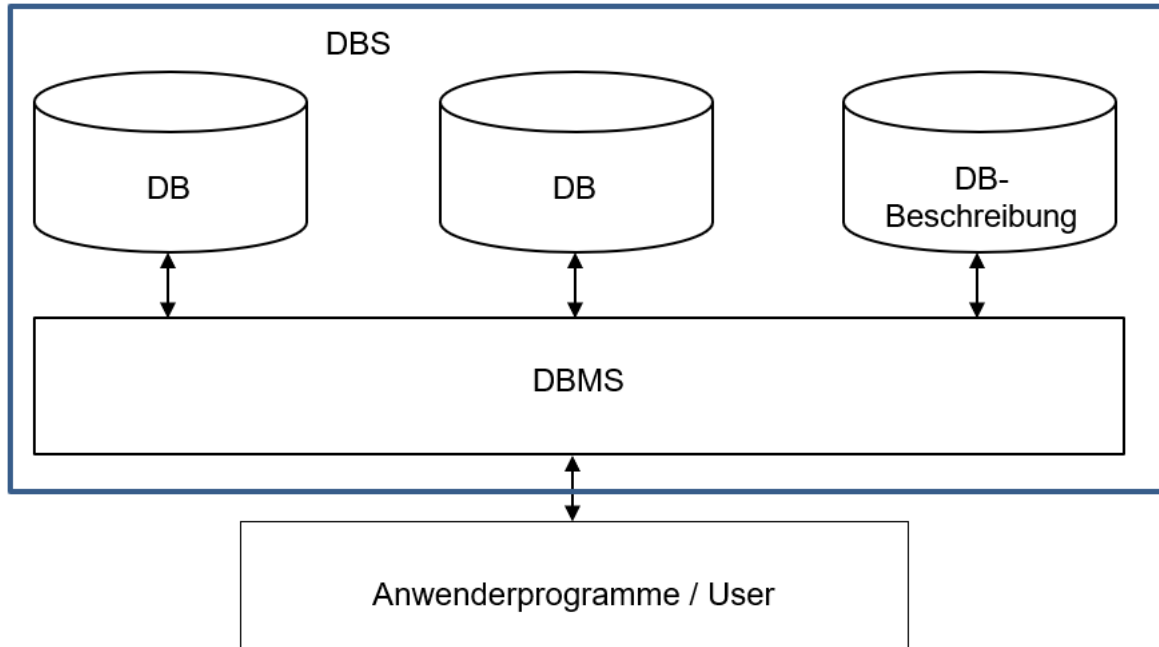
- Euklidische Distanz $d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$
- Manhattan-Distanz $d(x, y) = \sum_{i=1}^n |x_i - y_i|$
- Maximum-Distanz $d(x, y) = \max(|x_i - y_i|)$

Anmerkung: Mit der Erweiterung des KI-Themenumfangs in zukünftigen Abiturprüfungen durch Anforderungserlässe wird auch in den nächsten Jahren die Formelsammlung im Bereich Künstliche Intelligenz erweitert.

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

8 Datenbanken

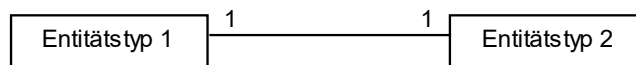
8.1 Datenbankmanagementsystem



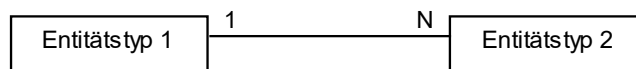
DBS = Datenbanksystem DBMS = Datenbankmanagementsystem DB = Datenbank

8.2 Entity-Relationship-Diagramm (ER-Diagramm)

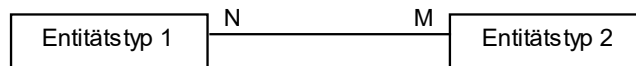
1:1 Beziehung



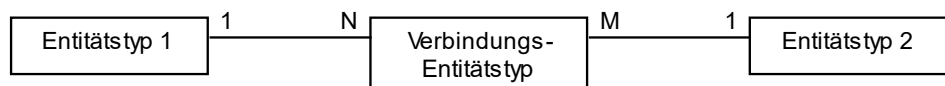
1:N Beziehung



N:M Beziehung



N:M Beziehung
aufgelöst



Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

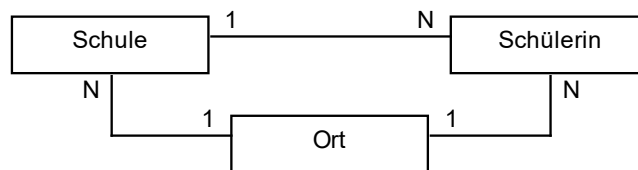
8.3 Relationenmodell

Alle Entitätstypen des Entity-Relationship-Diagramms mit Primär- und Fremdschlüsseln und allen Attributen der Entitätstypen in folgender Form:

Entitätstyp(Primärschlüssel, Attribut1, Attribut2, ..., Fremdschlüssel1, ...)

Beispiel: Schülerinnen, die ein Mädchengymnasium besuchen.

Entity-Relationship-Diagramm



Relationenmodell

Ort(OrtsNr, PLZ, Name)

Schule(SchulNr, Schulname, Straße, OrtsNr)

Schülerin(SchuelerinNr, Vorname, Name, Straße, OrtsNr, SchulNr)

8.4 Abfrageformulierung mit SQL

Projektion und Formatierung

Auswahl aller Spalten einer Tabelle

Syntax: SELECT *
 FROM <Tabelle>;

Auswahl mehrerer Spalten einer Tabelle

Syntax: SELECT <Spalte1>,<Spalte2>,<Spalte3>
 FROM <Tabelle>;

Auswahl ohne mehrfaches Auftreten derselben Zeile

Syntax: SELECT DISTINCT <Spalte>
 FROM <Tabelle>;

Umbenennen von Spalten bei der Ausgabe

Syntax: SELECT <Spalte> AS <neuer Spaltenname>
 FROM <Tabelle>;

Sortierung aufsteigend (ASC (optional)) oder absteigend (DESC)

Syntax: SELECT <Spalte>
 FROM <Tabelle>
 ORDER BY <Spalte> [ASC];

 SELECT <Spalte>
 FROM <Tabelle>
 ORDER BY <Spalte> DESC;

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Beispiel Relationenmodell Schüler (SID, Vorname, Name, Klasse)

```
SELECT *
FROM Schüler
ORDER BY Name, Vorname;

SELECT Vorname, Name
FROM Schüler
ORDER BY Name ASC;

SELECT DISTINCT Klasse
FROM Schüler
ORDER BY Klasse DESC;

SELECT Name AS "Nachname", Vorname
FROM Schüler;
```

Selektion

Auswahl von Zeilen

```
Syntax:      SELECT    <Spalte>
              FROM      <Tabelle>
              WHERE      <Bedingung>;
```

Vergleichsoperatoren

- =, <>, >, <, >=, <= (<> ungleich)
- BETWEEN wert1 AND wert2
- LIKE '_...%' oder "...%"
 - (_ ein Zeichen)
 - (% beliebig viele Zeichen)
- IN ('Wert1','Wert2') oder IN ("Wert1","Wert2")
- NOT IN ('Wert1','Wert2','Wert3')
- IS NULL
- IS NOT NULL

Logische Operatoren AND, OR, NOT

Beispiel Relationenmodell Schüler (SID, Vorname, Name, Klasse)

```
SELECT *
FROM Schüler

    Alle Schüler der TGI-J2
    WHERE Klasse = "TGI-J2";

    Alle Schüler der TG-Klassen
    WHERE Klasse LIKE 'TG%';

    Alle Schüler der TGI-Klassen
    WHERE Klasse IN ('TGI-E', 'TGI-J1', 'TGI-J2');

    Alle Schüler, die noch keiner Klasse zugeordnet sind
    WHERE Klasse IS NULL;
```

Beispiel Relationenmodell Laborübung(LID, Thema, Dauer)

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

```
SELECT *
FROM Laborübung

    Alle Laborübungen die mindestens 60 und höchstens 90 Minuten ( $60 \leq \text{Dauer} \leq 90$ ) gedauert
    haben
    WHERE Dauer BETWEEN 60 AND 90;

    Alle Laborübungen, deren Themen nichts mit Radioaktivität oder Atmosphärenchemie zu tun
    haben
    WHERE Thema NOT IN ("Radioaktivität", "Atmosphärenchemie");

    Alle Laborübungen zur Organik, die kürzer als 60 Minuten waren
    WHERE Thema = "Organik"
        AND Dauer < 60;
```

Verbund von Tabellen

Equi-Join

Syntax: SELECT <Spalte1>,<Spalte2>
 FROM <Tabelle1>,<Tabelle2>
 WHERE <Join-Bedingung>;

In der Join-Bedingung wird festgelegt, dass der Inhalt bestimmter Spalten identisch sein muss.

Beispiel Relationenmodell Schüler (SID, Vorname, Name, Klasse)
 Teilnahme(TID, SID, LID, Datum, Punkte)
 Laborübung(LID, Thema, Dauer)

```
SELECT Vorname, Name, Datum, Punkte
FROM   Schüler, Teilnahme
WHERE  Schüler.SID = Teilnahme.SID;
```

Anmerkung: Tabellennamen können in FROM durch Aliase abgekürzt werden.

```
SELECT Vorname, Name, Datum, Punkte
FROM   Schüler S, Teilnahme T
WHERE  S.SID = T.SID;
```

```
SELECT Vorname, Name, Datum, Thema, Dauer
FROM   Schüler S, Teilnahme T, Laborübung L
WHERE  S.SID = T.SID
      AND L.LID = T.LID;
```

Inner Join mit zwei Tabellen

Syntax: SELECT A.<Spalte1>,B.<Spalte2>
 FROM <Tabelle1> A INNER JOIN <Tabelle2> B
 ON A.<Spalte1> = B.<Spalte2>

Beispiel Relationenmodell Schüler (SID, Vorname, Name, Klasse)
 Teilnahme(TID, SID, Datum, Punkte)

```
SELECT Vorname, Name, Datum, Punkte
FROM   Schüler S INNER JOIN Teilnahme T ON S.SID = T.SID;
```

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Aggregatfunktion

Aggregatfunktionen können auf einer ganzen Tabelle bzw. Zwischentabelle ausgeführt werden. Ihre Ergebnistabelle besteht dann aus einer Zelle.

Syntax: SELECT Aggregatfunktion(<Spalte>)
 FROM <Tabelle>;

SUM	Summierung der numerischen Werte in der Spalte
MIN	Minimum der Spalte
MAX	Maximum der Spalte
AVG	Durchschnitt der numerischen Werte in der Spalte
COUNT	Anzahl der Zeilen des Zwischenergebnisses

Hinweis: NULL-Werte werden vor der Auswertung einer Aggregatfunktion eliminiert.

Beispiel Relationenmodell Schüler (SID, Vorname, Name, Klasse)
 Teilnahme(TID, SID, Datum, Punkte)

Summe der von den Schülern der Klasse TGI-E am 24.07.2021 erreichten Punkte
SELECT SUM(Punkte) AS "Gesamtpunktzahl der Klasse TGI-E am 24.07.21"
FROM Schüler S, Teilnahme T
WHERE S.SID = T.SID
 AND Klasse = "TGI-E"
 AND Datum = #24/07/2021#;

Maximal erreichte Punktezahl
SELECT MAX(Punkte) AS "Max. Punkte"
FROM Teilnahme;

Datum der ersten Teilnahme, d.h. des ersten Termins der Veranstaltung
SELECT MIN(Datum) AS "Startdatum"
FROM Teilnahme;

Punktedurchschnitt der Klasse TGI-E
SELECT AVG(Punkte) AS "Klassendurchschnitt TGI-E"
FROM Schüler S, Teilnahme T
WHERE S.SID = T.SID
 AND Klasse = "TGI-E";

Anzahl der Schüler in der Klasse TGI-E
SELECT COUNT(*) AS "Anzahl Schüler TGI-E"
FROM Schüler
WHERE Klasse = "TGI-E";

Spezialfall: COUNT(DISTINCT ...)

Beispiel Relationenmodell Schüler (SID, Vorname, Name, Klasse)
 Teilnahme(TID, SID, Datum, Punkte)

Anzahl Klassen
SELECT COUNT(DISTINCT Klasse) AS "Anzahl Klassen"
FROM Schüler S, Teilnahme T
WHERE S.SID = T.SID;

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Aggregatfunktion mit Gruppierung

Mit GROUP BY werden Abfrageergebnisse nach bestimmten Kriterien in Gruppen zusammengefasst. Auf jeder Gruppe wird einzeln die Aggregatfunktion ausgewertet und ein eigener Wert berechnet. Somit besteht die Ergebnistabelle aus den Aggregatwerten der einzelnen Gruppen.

Syntax: SELECT <Spalte1>, Aggregatfunktion(<Spalte2>) AS <Name>
 FROM <Tabelle>
 GROUP BY <Spalte1>;

Beispiel Relationenmodell Schüler (SID, Vorname, Name, Klasse)
 Teilnahme(TID, SID, Datum, Punkte)

Punktedurchschnitte pro Klasse
SELECT Klasse, AVG(Punkte) AS "Gesamtpunktzahl pro Klasse"
FROM Schüler S, Teilnahme T
WHERE S.SID = T.SID
GROUP BY Klasse;

Beste Leistung pro Tag
SELECT Datum, MAX(Punkte) AS "Bestes Tagesergebnis"
FROM Teilnahme T
GROUP BY Datum;

Selektion von Gruppen

Im Unterschied zur einfachen Selektion mit SELECT können mit HAVING Abfrageergebnisse von Aggregatfunktionen auf Gruppen selektiert werden.

Syntax: SELECT <Spalte1>, Aggregatfunktion(<Spalte2>) AS <Name>
 FROM <Tabelle>
 WHERE <Bedingung>
 GROUP BY <Spalte1>
 HAVING <Bedingung für Aggregatfunktion>;

Beispiel Relationenmodell Schüler (SID, Vorname, Name, Klasse)
 Teilnahme(TID, SID, Datum, Punkte)

Punktedurchschnitte pro Klasse, aber nur wenn der Durchschnitt größer als 20 Punkte ist.
SELECT Klasse, AVG(Punkte) AS "Gesamtpunktzahl pro Klasse"
FROM Schüler S, Teilnahme T
WHERE S.SID = T.SID
GROUP BY Klasse
HAVING AVG(Punkte)>20;

Komplette SQL-Anweisung

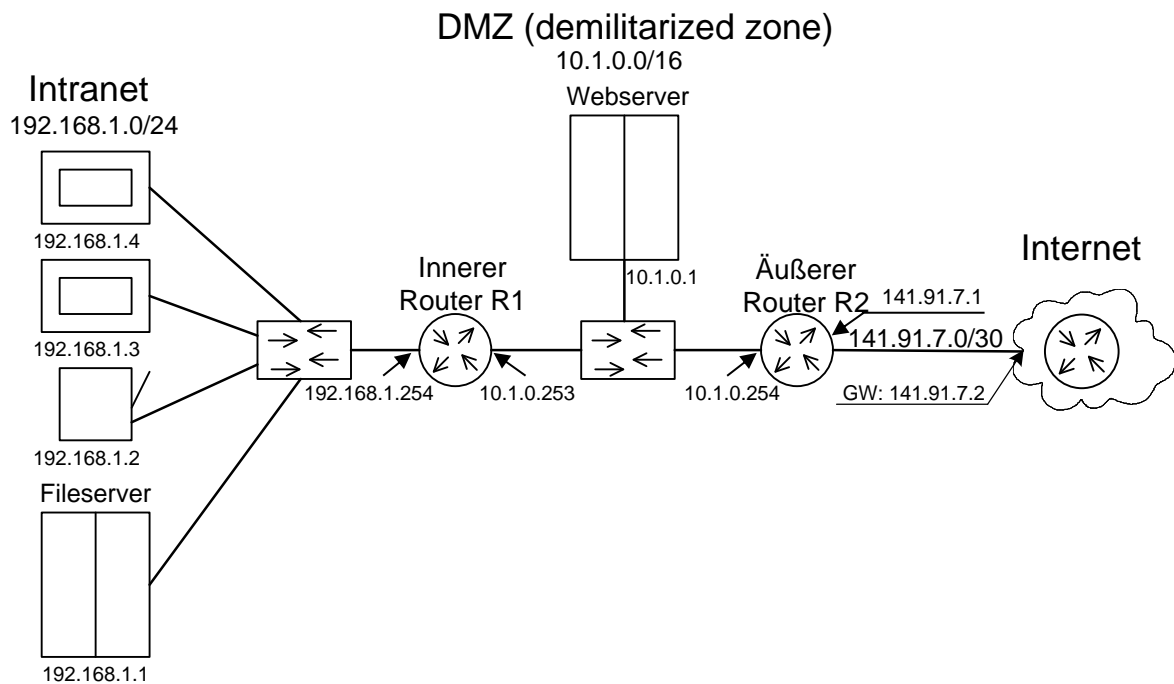
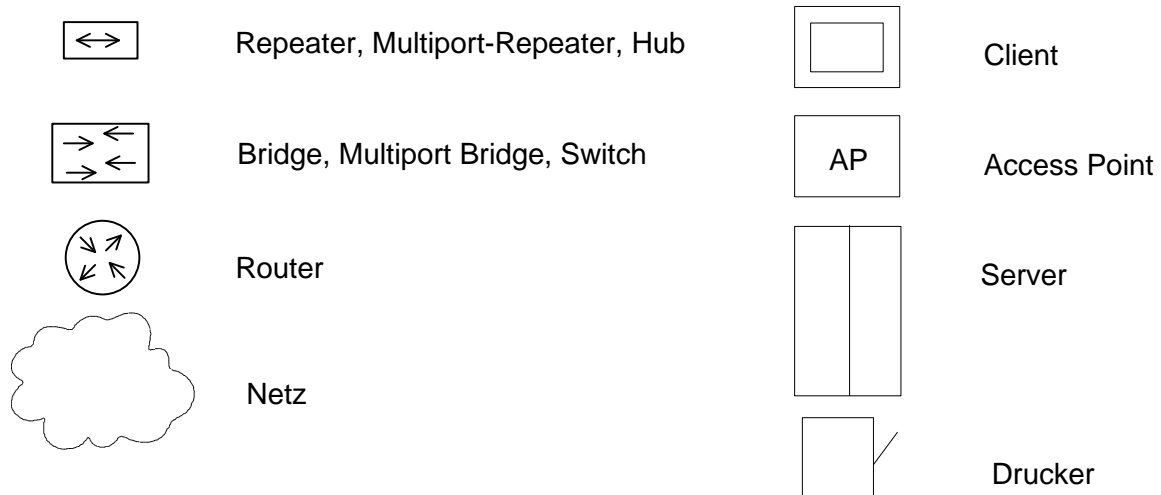
Syntax: SELECT ...
 FROM ...
 WHERE ...
 GROUP BY ...
 HAVING ...
 ORDER BY ... ;

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

9 Vernetzte Systeme

9.1 Netzwerktechnik

Netzwerksymbole



Routing-Tabelle (IPv4)

Die Routingtabelle des Router R2 sieht folgendermaßen aus:

Netzadresse	Subnetzmaske	Gateway
141.91.7.0	/30	*
10.1.0.0	/16	*
192.168.1.0	/24	10.1.0.253
0.0.0.0	0.0.0.0	141.91.7.2

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Aufbau IPv4-Adresse

IP-Adresse (dotted-decimal-format): z.B. 177 . 17 . 223 . 1
 IP-Adresse (binär): 10110001.00010001.11011111.00000001

8 Bit = 1 Oktett

32 Bit = 4 Bytes

IP-Adresse z.B.	192.168. 1 . 1	→	11000000.10101000.00000001.00000001
Netzmaske z.B. /24 =	255.255.255. 0	→	11111111.11111111.11111111.00000000
Netz-ID	192.168. 1 . 0	←	11000000.10101000.00000001.00000000
Host-ID	0 . 0 . 0 . 1	←	00000000.00000000.00000000.00000001

Alle Host-ID-Bits = 0: Netz-Adresse, hier 192.168.1.0

Alle Host-ID-Bits = 1: Broadcast-Adresse, hier 192.168.1.255

Aufbau IPv6-Adresse

IP-Adresse (hexadezimal): z.B. 2001:07C0:8280:0253:0000:0000:0000:0020

16 Bit

8 Blöcke (16 Bit) = 128 Bit

Weitere IPv6-Schreibweise:

Führende Nullen können ausgelassen werden → 2001:7C0:8280:253:0:0:0:20

Aufeinanderfolgende Null-Blöcke können durch zwei Doppelpunkte einmal ersetzt werden → 2001:7C0:8280:253::20

IPv4-Adressen können in IPv6-Adressen eingebettet werden, z.B. 192.168.1.1 → 0:0:0:0:0:0:192.168.1.1
 → ::192.168.1.1 → ::C0A80101

Adressformat:

64 Bits	64 Bits
Netzwerk Präfix	Interface Identifier (IID)
48 Bits	16 Bits
Global Routing Präfix	Subnetz ID

Netzwerk-Präfix:

2001:07C0:8280:0253 → Global Routing Präfix

2001:07C0:8280:**0253** → Subnetz Identifier

Adressbereich-Zuweisung:

2001:07C0:8280:0253::**64**

2001:07C0:8280:0200::**56**

2001:07C0:8280::**48**

Interface Identifier:

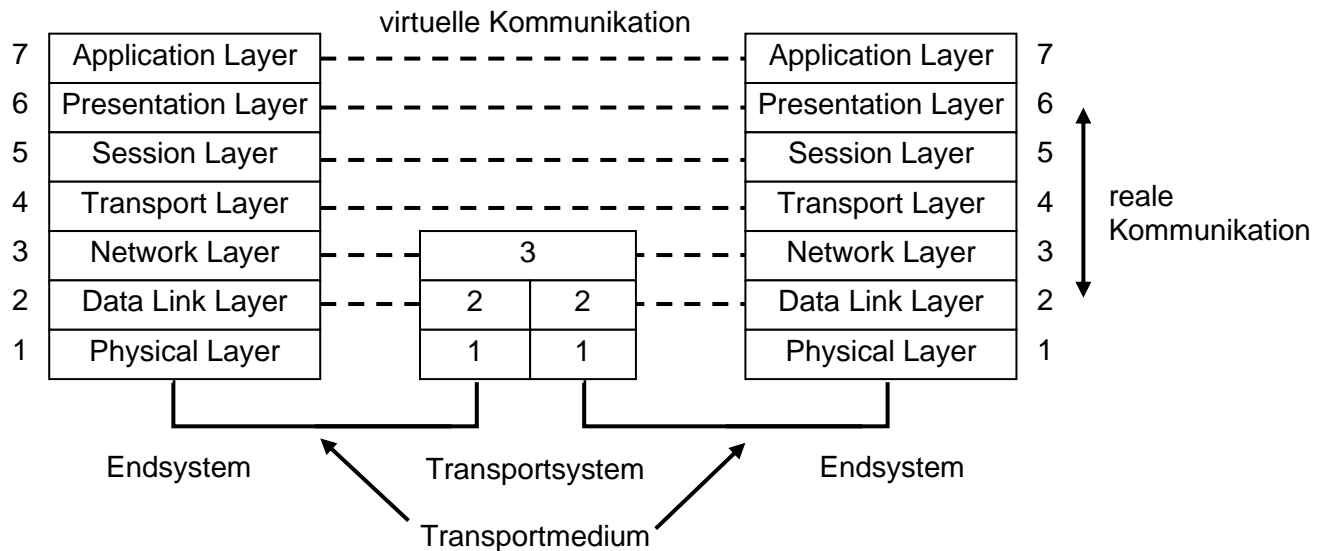
xxxx:xxxx:xxxx:xxxx:**0000:0000:0000:0020**

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

2001:07C0::/32

9.2 Schichtenmodelle

ISO-OSI-7-Schichtenmodell



TCP-IP-Schichtenmodell

OSI-Schicht	TCP/IP-Schicht	Protokoll-Beispiele
7	Anwendungen	HTTP, FTP, SMTP, Telnet, DHCP, MQTT, ...
6		
5		
4	Transport	TLS
3	Internet	TCP, UDP
2	Netzzugang	IP (IPv4, IPv6), ICMP
1		Ethernet

9.3 Header

Ethernet II

Präambel	Zieladresse	Absenderadresse	Typ	Daten	Link Trailer
8	6	6	2	46...1500	4 Byte

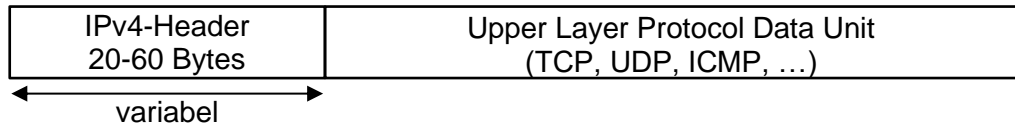
IPv4-Header

Byte	Inhalt
0	Version IHL
1	TOS
2-3	Paketlänge
4-5	Identifikation
6	Flags Fragmentabstand
7	Fragmentabstand
8	Time To Live (TTL)
9	Protokoll
10-11	Kopf-Prüfsumme
12-15	IP-Sendeadresse
16-19	IP-Empfängeradresse

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

20 ...	Optionen (mit evtl. Füllzeichen)
--------	----------------------------------

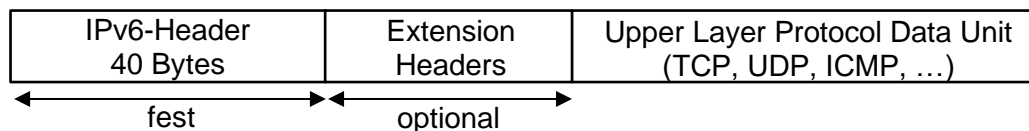
IPv4-Paketstruktur:



IPv6-Header

Byte	Inhalt
0-3	Version Traffic Class Flow Label
4-7	Payload Length Next Header Hop Limit
8-23	Source Address
24-39	Destination Address

IPv6-Paketstruktur:



TCP –Header

Byte	Inhalt
0-1	Source Port
2-3	Destination Port
4-7	Sequenznummer
8-11	Quittungsfeld (Piggyback, Acknowledgement Number)
12	Header-Länge reserviert
13	reserviert URG ACK PSH RST SYN FIN
14-15	Fenster Größe
16-17	Prüfsumme
18-19	Urgent Zeiger
20 ...	Optionen (evtl. mit Füllzeichen)

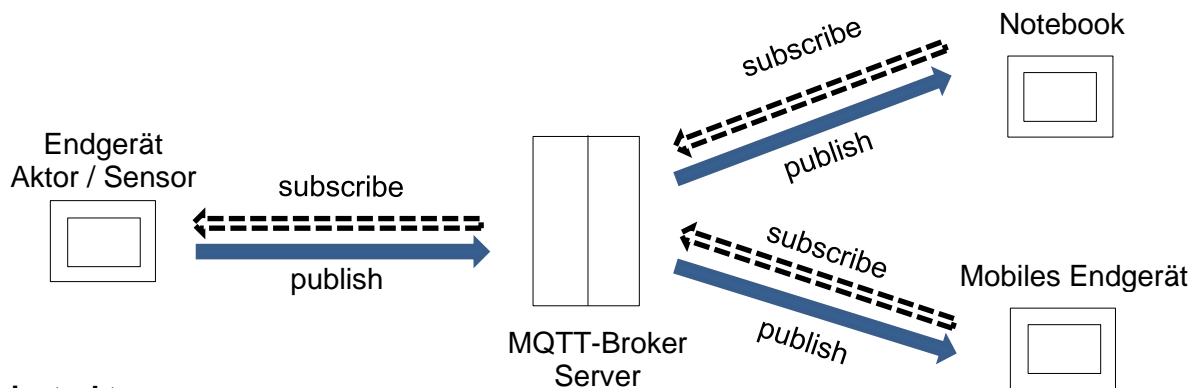
UDP –Header

Byte	Inhalt
0-1	Source Port
2-3	Destination Port
4-5	Länge des Datagramms
6-7	Check-Summe

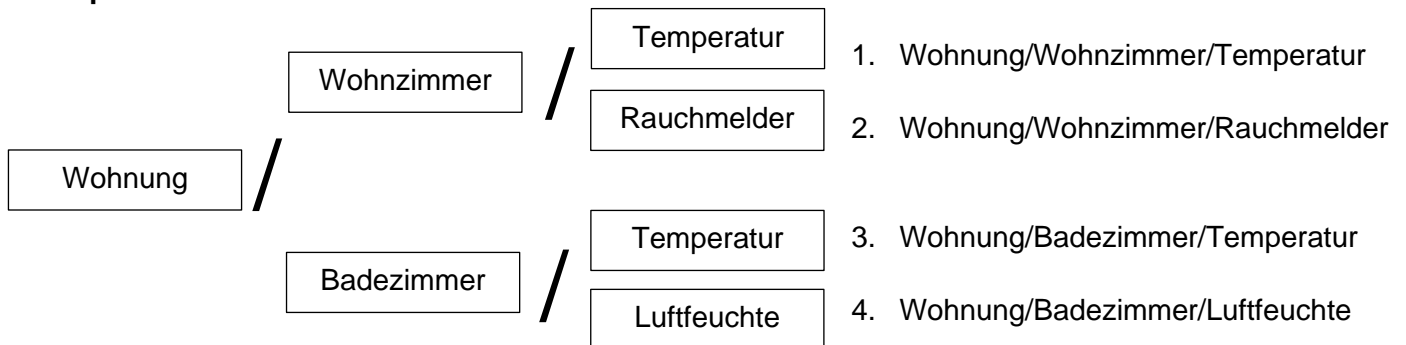
Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

9.4 Internet der Dinge (IoT)

MQTT-Protokoll (Message Queuing Telemetry Transport)



Topicstruktur:



Multi-Level-Wildcard:

Wohnung/Wohnzimmer/#

Single-Level-Wildcard: +

Wohnung+/Temperatur

Qualitätsstandards:

QoS 0

publish

QoS 1

publish

puback

QoS 2

publish

pubrec

pubrel

pubcomp

Ports:

1883 : MQTT, unverschlüsselt

8883 : MQTT, verschlüsselt

8884 : MQTT, verschlüsselt, Client Zertifikat notwendig

8080 : MQTT über WebSockets, unverschlüsselt

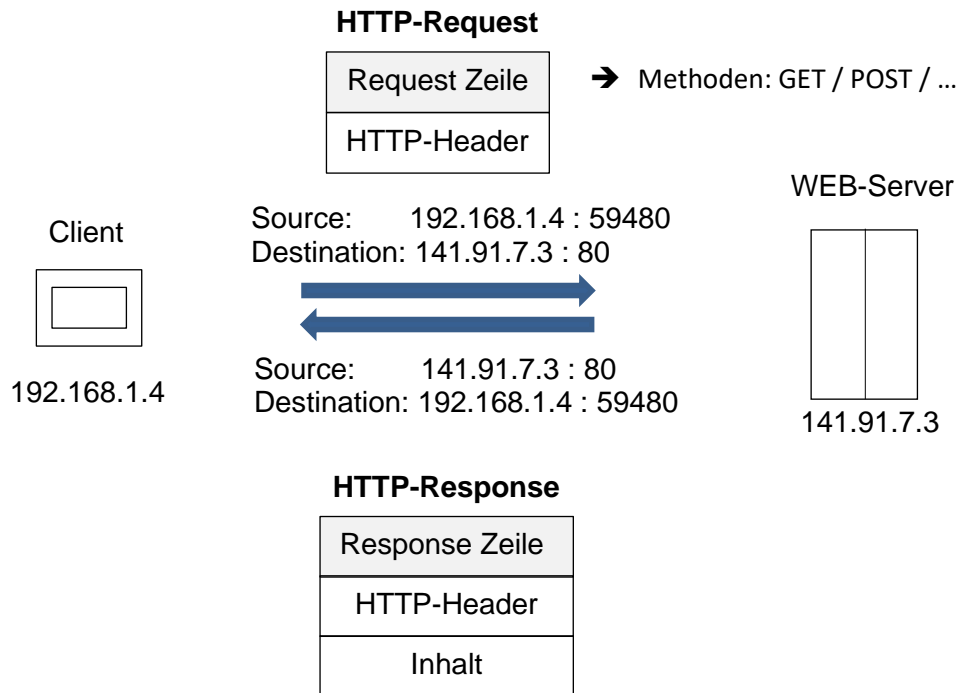
MQTT -Header: (Beispiel - Publish Message)

Byte	Inhalt
0	Nachrichtentyp (4 Bit) Dub-Flag Quality of Service Retain-Flag
1	Länge des restlichen MQTT-Pakets
...	MQTT-Topic → Topic-Länge / Topic / Payload

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

HTTP-Protokoll (Hypertext Transfer Protocol)

Kommunikationsprinzip:



URL (Uniform Resource Locator):

Protokoll	Domain	Pfad
https://	gsoe.de	/bildungsangebote/technisches-gymnasium/

Ports:

80 : HTTP, unverschlüsselt

443 : HTTPS, verschlüsselt

Abiturprüfung ab 2024	Berufliches Gymnasium (TG)
Formelsammlung	1.5.2 Informationstechnik

Request HTTP 1/1

Method	Pfad	Protokoll
GET	/wp/content/uploads/2020/11/pixels-fauxels.jpg	HTTP/1.1\r\n

HTTP-Header - Name: Wert (Beispiele)	
Host:	→ Domain-Name des Servers
User-Agent:	→ User-Agent des Clients
Accept:	→ Welche Inhaltstypen der Client verarbeiten kann
z.B.	<ul style="list-style-type: none"> Accept-Charset: → Welche Zeichensätze der Client anzeigen kann. Accept-Encoding: → Welche komprimierten Formate der Client unterstützt. Accept-Language: → Gewünschte Sprachversion
Date:	→ Datum und Zeit des Requests
Connection:	→ Bevorzugte Art der Verbindung
Referrer:	→ URL der Ressource, von der aus verlinkt wurde.
Content-Length:	→ Länge des Request-Bodys
Content-Type:	→ MIME-Typ des Bodys (bei POST- und PUT-Requests)

Response HTTP 1/1

Protokoll	Status-Code
HTTP/1.1	200 OK\r\n

HTTP-Header - Name: Wert (Beispiele)	
Date:	→ Zeitpunkt der Response
Server:	→ Kennung des Servers
Accept-Ranges:	→ Welche Einheiten der Server akzeptiert
Allow:	→ Erlaubte Request-Typen (Methoden)
Connection:	→ Bevorzugte Art der Verbindung

Status-Codes	(Beispiele)
100 ... 199:	Information
200 ... 299:	Client-Anfrage erfolgreich z.B. 200 – OK
300 ... 399:	Client-Anfrage umgeleitet z.B. 301 – Moved Permanently 302 – Moved Temporarily
400 ... 499:	Fehlen des Dokuments z.B. 403 – Forbidden 404 – Not Found
500 ... 599:	Serverfehler