# Zahlensysteme, Codierungen

2. binäre Zahlenwerte in Digitalrechnern

- *Wie werden binäre Zahlenwerte in Digitalrechnern dargestellt?*
- *Was versteht man unter Störabstand?*

- als niedrige (0) oder hohe Spannung (1)
- Störabstand = Spannungsbereich zwischen High&Low, in dem der Wert nicht definiert ist

3. ganze Zahlen

- *Erklären Sie, wie ganze Zahlen typischerweise im Rechner dargestellt werden!*
- *Wie wird mit vorzeichenbehafteten Zahlen umgegangen?*

- Codiert in Binär (0&1)
- Zweikomplement ( 0 am Anfang +, 1 am Anfang - )

4. Dez->Bin

- *Zeigen Sie die Darstellung der dezimalen Zahl 250 in der binären und in der hexadezimalen Schreibweise (8 Bit)!*
- *Welchem Wert entspricht die Zahl, wenn die Bits als Zweierkomplement-Codierung aufgefasst werden?*

- 250 -128(2^7)=122 | 1
- 122 -64 (2^6)=58 | 11
- 58 -32 (2^5)=26 | 111
- 26 -16 (2^4)=10 | 1111
- 10 - 8 (2^3)=2 | 11111
- 2 < 4(2^2) --> 0| 111110
- 2 - 2 (2^1)=0 | 1111101
- 0 <1 (2^0) | 11111010

- 4 binäre Stellen = 1 hexadezimale Stelle
- 1111 | 1010 -> F | A (15=F | 10=A)

- 11111010 als Zweierkomplement -> erste Stelle = - ; Rest invertieren und +1
- 1 -> -
- 1111010 -> 0000101 + 1 -> 0000110
- -6

5. Erklären Sie die binäre Struktur von 3 unterschiedlichen Datentypen Ihrer Wahl!

- Boolean ( 1Bit = 0-Falsch ; 1-Wahr)
- Char ( 8Bit = jede Zahl entspricht einem Zeichen nach ASCII)
- Integer ( 16Bit =Zweierkomplement [-65536 bis 65535])

# Boolsche Algebra und Digitale Schaltungen

6. Beschreiben Sie den Zusammenhang zwischen Logikwert, -pegel und Störabstand!

- Logikwert = 0 - niedrige Spanng; 1 - hohe Spannung
- Logikpegel = zugewiesener Bereich für niedrige (1V-5V) und hohe Spannung (17V-24V)

- Störabstand = Bereich zwischen beiden Pegeln, in welchen nicht genau unterschieden werden kann, welche Zahl gilt, dieser Bereich wird beim Umschalten durchschritten

7. natürlches Signal -> digitaler Zahlenwert

- *Was ist bei der Abbildung einer kontinuierlichen physikalischen Größe (natürliches Signal) auf digitale Zahlenwerte zu beachten?*
- *Erklären Sie den Zusammenhang zwischen Bitzahl und Genauigkeit!*

- das natürliche Signal wird nur in einem bestimmten Takt aufgenommen
- die physikalische Größe wird bei der Umwandlung quanitisiert
- d.h. abhängig von der Auflösung(Bitzahl) wird die phys. Größe unterschiedlich genau gespeichert/wahrgenommen

8. Wie erfolgt die Abbildung einer kontinuierlichen physikalischen Größe auf digitale Werte?

- maxP - max. phys. Größe | maxD - maximale digitale Größe (in Dezimal)
- P - physikalischer Wert. | D - digitaler Wert (in Dezimal)
- maxP/maxD = P/D
- der digitale Wert muss dabei gerundet werden und in den Speichertyp umgewandelt werden

9. Was ist bei realen kombinatorischen Schaltungen hinsichtlich des Zeitverhaltens zu beachten?

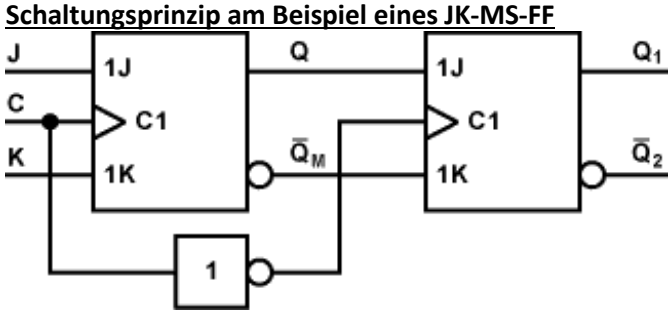
- das natürliche Signal wird nur in einem bestimmten Takt aufgenommen

# Digitale Funktionen

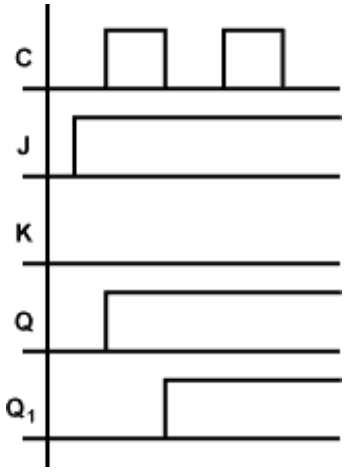
10. Master-Slave-Flipflop

- *Erläutern Sie die Wirkungsweise eines Master-Slave-Flipflops anhand eines Schaltbilds und des Takt-Zeitverlaufs zur Übernahme der Zustände in Master bzw. Slave!*
- *Für welche Anwendungen ist es besonders geeignet?*

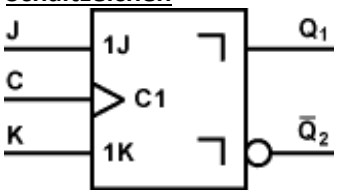
- Alle zweiflankengesteuerten Flip-Flops sind Master-Slave-Flip-Flops. Sie reagieren auf die positive, wie auch auf die negative Taktflanke.
- Bei der positiven Taktflanke werden die am Eingang anstehenden Daten eingelesen. Bei der negativen Taktflanke werden die Daten verzögert ausgegeben.
- **Schaltungsprinzip am Beispiel eines JK-MS-FF**




- Das JK-MS-Flip-Flop besteht aus zwei einzelnen JK-Flip-Flops, die direkt miteinander verbunden sind. Die Ausgänge des ersten, dem Master-Flip-Flop sind auf die Eingänge des zweiten, dem Slave-Flip-Flop geschaltet.
- Das erste Flip-Flop reagiert auf die steigende Taktflanke. Das zweite Flip-Flop auf die fallende Taktflanke.
- Damit das Slave-Flip-Flop auf die fallende Flanke reagiert wird der Takteingang mit einer NICHT-Verknüpfung negiert.
- **Impulsdiagramm**




- Im Impulsdiagramm kennzeichnet C das Taktsignal. J und K sind die beiden Eingänge und Q der Ausgang.
- Mit der positiven Taktflanke wird der Flip-Flop-Zustand eingelesen. Mit der negativen Taktflanke wird der Zustand an den Ausgang weitergegeben.
- **Schaltzeichen**

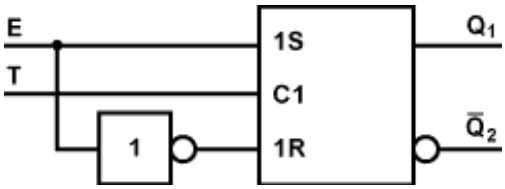


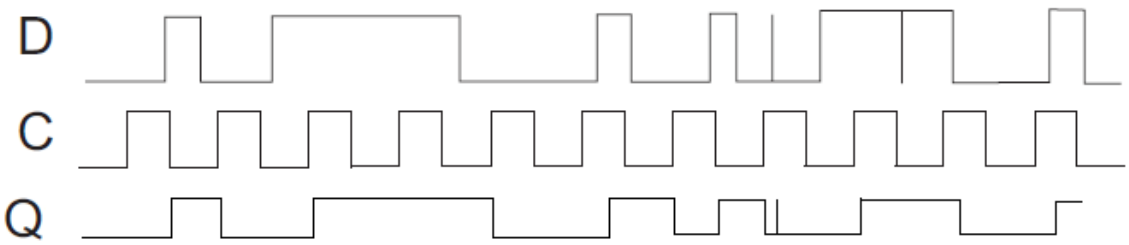

- Anwendung: erhöhte Störsicherheit (u.a. Messen)

*(http://elektroniktutor.de/digitaltechnik/ms_ff.html)*

11. D-Flipflop

- *Erläutern Sie Funktion und Anwendung eines D-Flipflops!*
- *Wie erfolgt die Übernahme der Daten bei gegebenen Verlauf von Takt C und Dateneingang D, wenn der Ausgang Q am Anfang der Logikwert 0 hat (Taktzustandgesteuertes D-Flipflop)?*
- *Zeichnen Sie den Verlauf von Q in folgendes Taktdiagramm ein:*

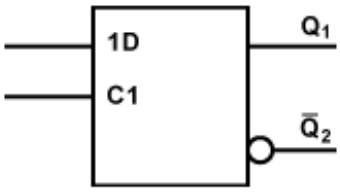



- Das D-Flip-Flop besteht aus einem RS-Flip-Flop, bei dem der Rücksetzeingang zum Setzeingang negiert ist. Dadurch wird verhindert, dass der unbestimmte Zustand eintritt.
- Das D-Flip-Flop gibt es als taktzustandsgesteuertes (siehe Schaltzeichen) und auch als taktflankengesteuertes Flip-Flop. Doch wenn ein D-Flip-Flop RS-Eingänge hat, so lässt es sich über diese Eingänge auch taktunabhängig steuern.
- Das D-Flip-Flop stellt das Grundelement für statische Schreib-Lese-Speicher dar.

- Der einzige Eingang wird als Daten-Eingang bezeichnet. Die Speicherung wird nur mit dem Takteingang gesteuert. Immer wenn der Taktimpuls anliegt, wird der Wert am Eingang an den Ausgang übernommen.
- Funktion/Anwundung: Speicherung o. Verzögerung der Daten

**Schaltzeichen**




**Wahrheitstabelle**

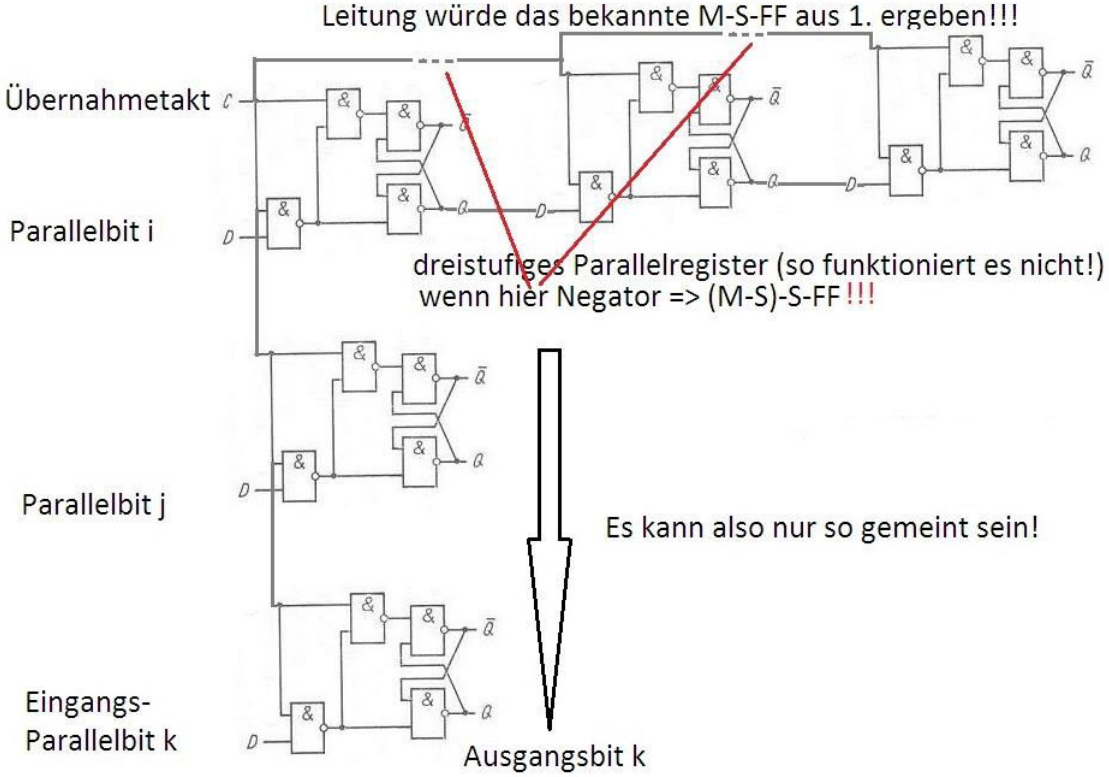
| E/1D | T/C1 | $Q_1$ | Funktion |
|---|---|---|---|
| 0 | 0 | n | Speichern |
| 0 | 1 | 0 | Rücksetzen |
| 1 | 0 | n | Speichern |
| 1 | 1 | 1 | Setzen |

- Immer, wenn am Takteingang eine Null anliegt, wird egal welchen Pegel der Dateneingang hat, der vorhergehende Pegel am Ausgang gespeichert. Liegt am Takteingang ein High-Pegel und ein Low-Pegel am Dateneingang, so wird das Flip-Flop zurückgesetzt. Liegt am Takteingang ein High-Pegel und ein High-Pegel am Dateneingang, so wird das Flip-Flop gesetzt.

12.Geben Sie die Schaltung für ein dreistufiges Parallelregister bestehend aus D-Flipflops an

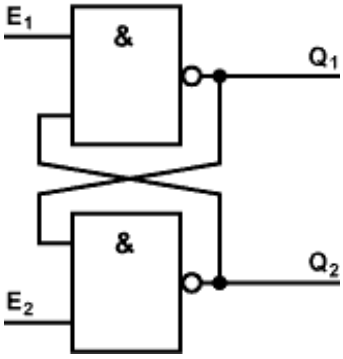
7)

Eine Verbindung in dieser Art bringt ein nur durch die Gatterlaufzeit verzögertes Signal Q bezogen auf D (rasender Informationsdurchlauf!), der Einbau eines Negators in der Leitung würde das bekannte M-S-FF aus 1. ergeben!!!



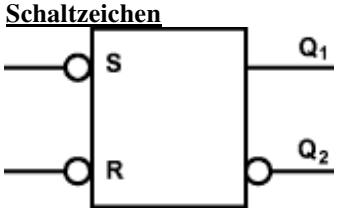

13. RS-Flipflop

- *Zeigen Sie, wie ein RS-Flipflop aus NAND-Gattern aufgebaut werden kann, und geben Sie die Schaltbelegungstabelle an!*




- Dieses wird durch L-Pegel am S-Eingang gesetzt und am R-Eingang rückgesetzt. Der Speicherzustand wird durch H-Pegel an beiden Eingängen hergestellt.
- In der Regel sind die beiden Ausgänge (Q1 und Q2) zueinander negiert. Doch weil die Ausgänge gleichzeitig einen L-Pegel ausgeben können, müssen sie immer getrennt betrachtet werden.

- **Schaltzeichen**




- Ein RS-Flip-Flop mit NAND-Verknüpfungen erkennt man an den negierten Eingängen.
- Im Schaltzeichen werden die Eingänge mit S (setzen) und R (rücksetzen) bezeichnet. Q2 ist zu Q1 negiert.

- Bei diesem Schaltzeichen handelt es sich allerdings nicht um ein richtiges RS-Flip-Flop. Es handelt sich eher um das Schaltzeichen eines NAND-Flip-Flops. Erst mit jeweils einer NICHT-Verknüpfung vor den Eingänge wird es zu einem richtigen RS-Flip-Flop. Das bedeutet, erst mit zusätzlicher Beschaltung, von zwei NICHT-Verknüpfungsgliedern wird ein NAND-Flip-Flop zum RS-Flip-Flop.

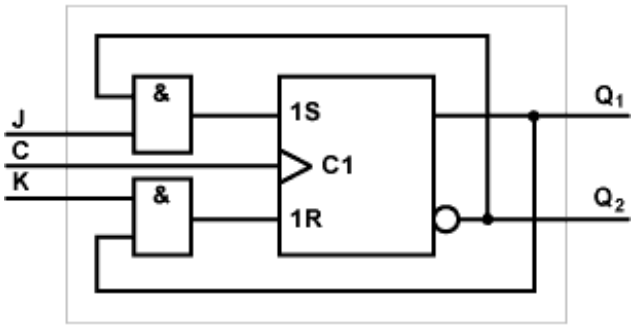
**Wahrheitstabelle**

| $E_1$ (S) | $E_2$ (R) | $Q_1$ | $Q_2$ | Zustand |
|---|---|---|---|---|
| 0 | 1 | 1 | 0 | Setzen (set) |
| 1 | 1 | X | X | Speichern |
| 1 | 0 | 0 | 1 | Rücksetzen (reset) |
| 0 | 0 | 1 | 1 | nicht speicherbar |

14. JK-Flipflops

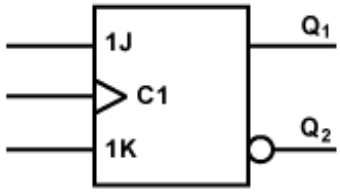
- *Erläutern Sie die Wirkungsweise eines JK-Flipflops anhand des Schaltzeichens und der Logiktabelle!*
- *Für welche Anwendungen ist es geeignet?*

- Ein JK-Flip-Flop wechselt beim Anlegen eines Taktimpulses seinen Ausgangszustand, wenn an beiden Eingängen (J und K) H-Pegel anliegen. Dieses Verhalten wird als Toggeln (kippen) bezeichnet.
- Wenn ein JK-Flip-Flop RS-Eingänge hat, so lässt es sich taktunabhängig steuern. Bei diesem Flip-Flop ist der unbestimmte Zustand ausgeschlossen.




- Der Eingang C des JK-Flip-Flops ist der Takteingang, abgeleitet vom englischen clock (Takt). Hier sollte ein Rechtecksignal anliegen. Die beiden Eingänge J und K sind Steuereingänge. Das JK-Flip-Flop hat eine Steuerung auf der Taktflanke, also dem Übergang an C von 0 nach 1 oder umgekehrt von 1 nach 0. Die Ausgängen Q1 und Q2 werden in Abhängigkeit der Ansteuerung der Eingänge J und K gesteuert. Das JK-Flip-Flop gibt es als taktflankengesteuertes und taktzustandsgesteuertes Flip-Flop.

## Schaltzeichen (taktflankengesteuertes JK-Flip-Flop)




## Wahrheitstabelle (taktflankengesteuertes JK-Flip-Flop)

| C | K | J | $Q_1$ | $Q_2$ | Funktion |
|---|---|---|---|---|---|
| 0 > 1 | 0 | 0 | n | n | Speichern |
| 0 > 1 | 0 | 1 | 1 | 0 | Setzen |
| 0 > 1 | 1 | 0 | 0 | 1 | Rücksetzen |
| 0 > 1 | 1 | 1 | X | X | Wechseln (Toggeln) |

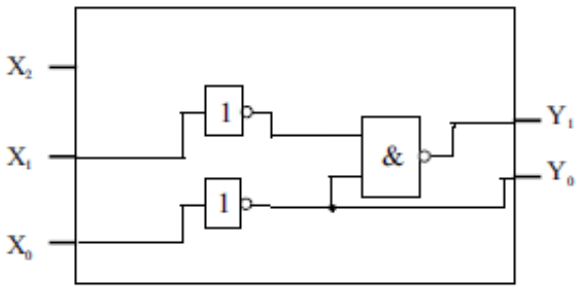
n = Pegel abhängig von J und K (0 oder 1)
X = Pegel abhängig vom vorherigen Zustand ( 0 -> 1 und 1 -> 0 )

- Liegt kein High-Pegel am Takteingang, so wird der an den Ausgängen anstehende Pegel gespeichert. Liegt am Setzeingang (J) und am Takteingang (C) ein High-Pegel, so wird das Flip-Flop gesetzt. Liegt am Rücksetzeingang (K) und am Takteingang ein High-Pegel, so wird das Flip-Flop zurückgesetzt. Liegt an beiden Steuereingängen ein High-Pegel, so wird der gespeicherte Wert gewechselt, d. h. aus High wird Low, aus Low wird High.
- Anwendung: diskrete digitale Schaltungen

15. Codewandler

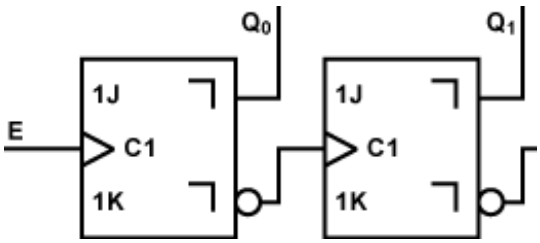
- *Es soll ein Codewandler entworfen werden, der die Eingangssignale $X_i$ (0 i 2) in die Ausgangssignale $Y_1$, $Y_0$ gemäß folgender Schaltbelegungstabelle:*
  $X_2$ $X_1$ $X_0$ $Y_1$ $Y_0$
  0 0 1 1 0
  0 1 0 1 1
  1 1 1 1 0
  1 0 0 0 1
  *umsetzt.*
- *Der Codewandler ist mit den vorgegebenen Gattern zu realisieren.*
- *Verdrahten Sie die vorgegebenen Gatter des Codewandlers (siehe Bild).*
- *Gehen Sie von den Logikgleichungen für $Y_1$, $Y_0$ aus.*

- $Y_0$ = not($X_0$) [direkt sehen oder über Formeln $X_2$ & $X_1$ rausfallen lassen]
- $Y_1$ = not($X_0$) nand not($X_1$) [4. Zeile als Grundformel nutzen, fertig]




16. asynchroner dualer Rückwärtszähler

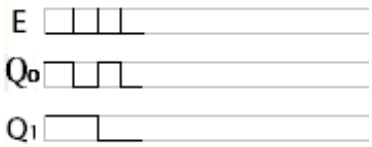
- *Zeichnen Sie Schaltbild und Impulsdiagramm für einen aus 2 Zähl-Flipflop bestehenden asynchronen dualen Rückwärtszähler!*




- JK-Flipflops
- JK-Eingänge immerauf 1 -> Eingehende Taktflanke (0>1) wechselt Ausgänge

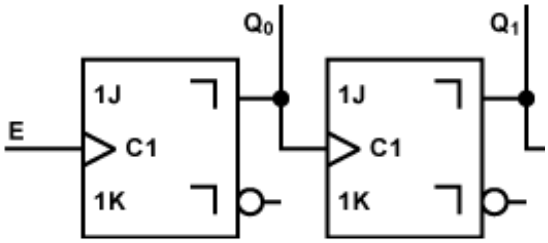
**Wahrheitstabelle**

| Takt (E) | $Q_1$ | $Q_0$ |
|---|---|---|
| 3 | 1 | 1 |
| 2 | 1 | 0 |
| 1 | 0 | 1 |
| 0 | 0 | 0 |




17. asynchroner dualer Vorwärtszähler

- *Zeichnen Sie Schaltbild und Impulsdiagramm für einen aus 2 Flipflops bestehenden asynchronen dualen Vorwärtszähler!*
- *Welche Art von Flipflops ist geeignet? Wie werden Zähler in Timer-Bausteinen (z. B. in aktuellen Mikrocontrollern ) verwendet?*



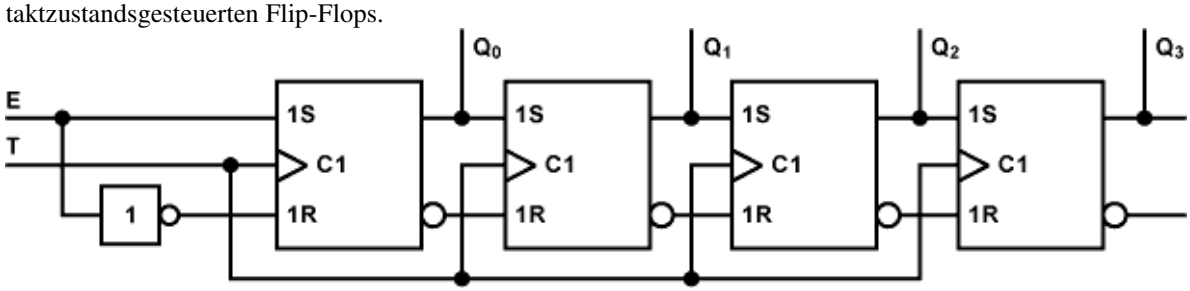

**Wahrheitstabelle**

| Takt (E) | $Q_1$ | $Q_0$ |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 2 | 1 | 0 |
| 3 | 1 | 1 |

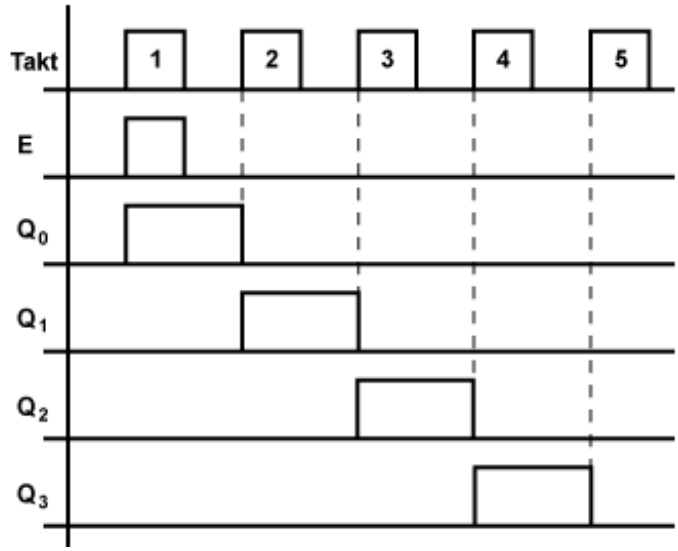
- Impulsdiagram ähnlich 16.
- T-Flip-Flops, JK-Flip-Flops, JK-Master-Slave-Flip-Flops oder RS-Flip-Flops
- Summieren der Taktzeiten auf gesetzte Zeit -> Taktzeit*Summe = Timer

18. Parallel- und Serienregister

*a) Welche Arten von Schieberegistern unterscheidet man?*
*Nennen Sie entsprechende Anwendungsbereiche.*
*b) Geben Sie die Schaltung für ein zweistufiges Schieberegister mit seriellem Ein- und Ausgang an, bei dem die Schieberichtung umschaltbar ist!*

- Schieberegister sind Schaltungen, die mehrstellige binäre Signale taktgesteuert aufnehmen, speichern und wieder abgeben können. Schieberegister arbeiten entweder mit einer seriellen oder einer parallelen Ein- und Ausgabe. Der Unterschied liegt in der Anzahl der Ein- und Ausgänge.
- Für den diskreten Aufbau eines Schieberegisters eignen sich taktflankengesteuerte D-Flip-Flops, SR-Flip-Flops und JK-Flip-Flops. Häufig verwendete Schieberegister stehen schon fertig als integrierte Schaltungen zur Verfügung. Da ein Flip-Flop nur ein Bit speichern kann, werden mehrere Flip-Flops zu einem Schieberegister zusammengeschaltet. Das hier dargestellte Schieberegister besteht aus 4 taktzustandsgesteuerten Flip-Flops.
- 


**Impulsdiagramm**

- 

- Parallel- & Serienregister
- Anwendungen:

Speicherung von Daten, Adressen & Programmcode
CPU-Register
Ports (parallel/seriel)
stat. RAM
Serialisierung und Parallelisierung von Bitmustern (z.B. Seriel->Parallel)
Bitverschiebung in Maschinensprachen
Multiplikation von Binärzahlen
Verwendung als Puffer
Erzeugung von Pseudozufallszahlen
Zyklische Redundanzprüfung

6a)

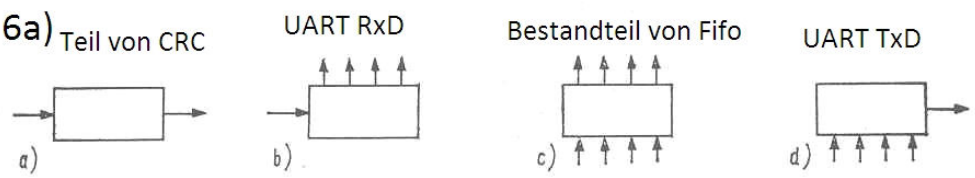



*Bild 9.7 Vier Möglichkeiten der Informationsein- und -ausgabe bei Schieberegistern*
a) seriell ein, seriell aus; b) seriell ein, parallel aus; c) parallel ein, parallel aus; d) parallel ein, seriell aus

6b)

264 9 Halbleiterspeicher

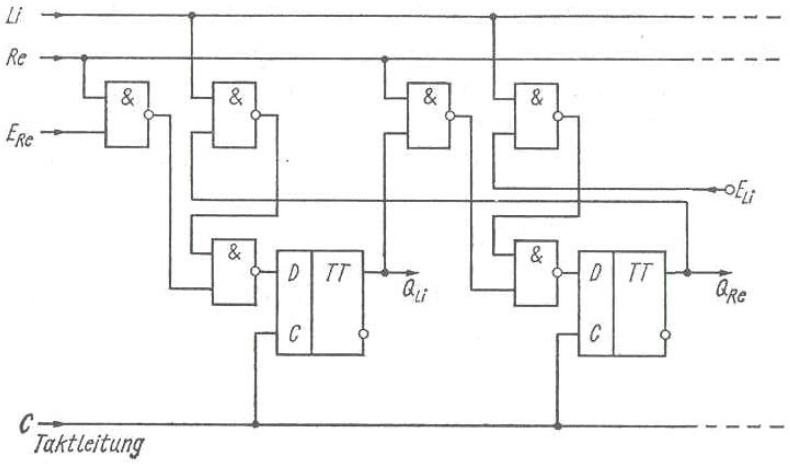
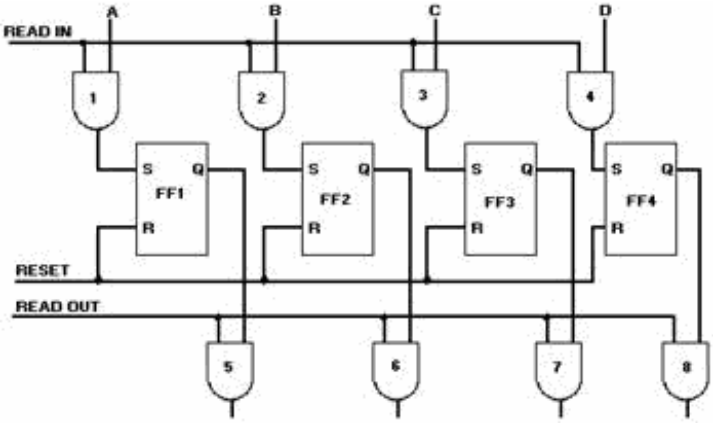



*Bild 9.9 Schieberegister mit umschaltbarer Schieberichtung*
$E_{Re}$ ($E_{Li}$): Eingang für Rechts(Links)schieben; $Q_R$ ($Q_L$): Serieller Ausgang für Rechts(Links)schieben; $Q_L$: Ausgang des linken Flipflops. Steuerleitungen: Re = 1, Li = 0: Rechtsschieben, Re = 0, Li = 1: Linksschieben

19. Register

- *Wie können Register hinsichtlich der Eingabe bzw. des Auslesens der Daten unterschieden werden?*
- *Welche Flipflops eignen sich für die Realisierung?*
- *Zeigen Sie als Beispiel die Realisierung für ein 3-Bit-Parallel-Register und nennen Sie Anwendungsmöglichkeiten!*

- Register: siehe 18
- D-Flip-Flops, SR-Flip-Flops und JK-Flip-Flops (sihe 18)



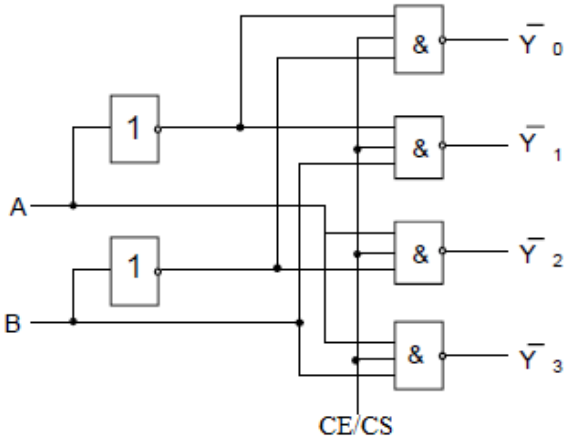

- (1-8 AND-Gatter)
- (http://rfcafe.com/references/electrical/neets-modules/NEETS-Module-13-3-31-3-40.htm)
- Speicherung von Zahlen, Datenübertragung zur richtigen Zeit, (...siehe 18)

21. 1-aus-4-Decoder

- *Zeigen Sie, wie ein 1-aus-4-Decoder durch NAND-Gatter realisiert werden kann!*
- *Gehen Sie dabei von einer kanonischen Grundform, der Schaltbelegungstabelle und den Logikgleichungen aus!*

Adressierung von Speichern: oft Ausgänge negiert

| A | B | $y_0$ | $y_1$ | $y_2$ | $y_3$ |
|---|---|---|---|---|---|
| 0 | 0 | 1 | | | |
| 0 | 1 | | 1 | | |
| 1 | 0 | | | 1 | |
| 1 | 1 | | | | 1 |

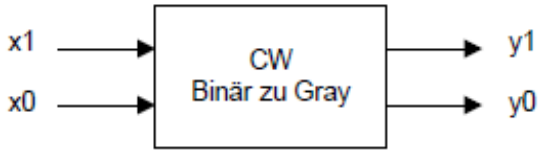



*(aus Vorlesungfolie "geklaut", ohne Y-Negation jeweils $Y_0$&$Y_3$ sowie $Y_1$&$Y_2$ vertauschen)*

- $Y_0 = \overline{A \wedge B}$
- $Y_1 = \overline{A \wedge \overline{B}}$
- $Y_2 = \overline{\overline{A} \wedge B}$
- $Y_3 = \overline{\overline{A} \wedge \overline{B}}$

22. 2-Bit-Binär-zu-Gray-Code-Wandler

- *Entwerfen Sie einen 2-Bit-Binär-zu-Gray-Code-Wandler (siehe Abbildung) aus NAND-Gattern!*

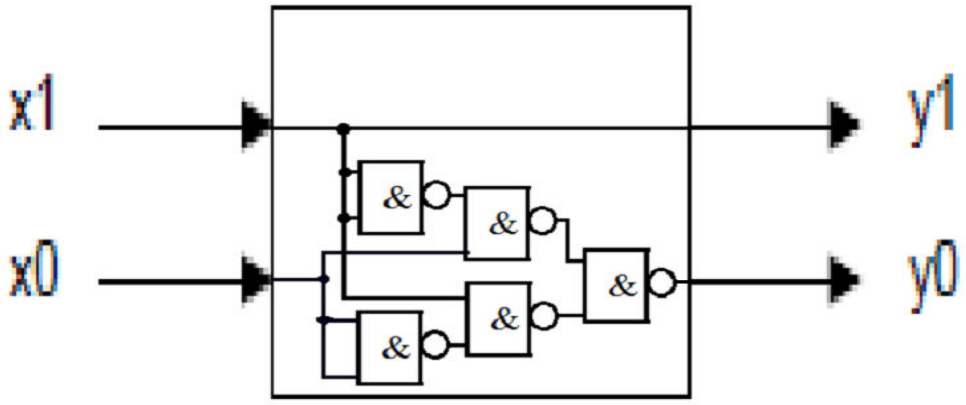



- *Die Funktion wird durch folgende Schaltbelegungstabelle definiert:*

| x1 | x0 | y1 | y0 |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 |

- *Gehen Sie beim Entwurf der Schaltung von den in der Tabelle vorgegebenen logischen Beziehungen aus!*

- $Y_1 = X_1$
- $Y_0 = X_1 \text{ XOR } X_0 = \overline{\overline{(\overline{X_1} \wedge X_0)} \wedge \overline{(X_1 \wedge \overline{X_0})}}$



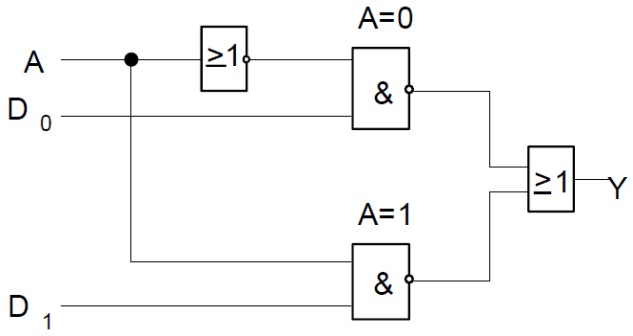

23. Umschalter (Multiplexer)

- *Entwerfen Sie als Sonderfall des Multiplexers einen einfachen Umschalter gemäß folgender Schaltbelegungstabelle:*

| $A$ | $Y$ | $\overline{Y}$ |
|---|---|---|
| 0 | $D_0$ | $\overline{D_0}$ |
| 1 | $D_1$ | $\overline{D_1}$ |

*a) Stellen Sie die Gleichung Y = f(A;D0;D1) auf!*
*b) Geben Sie eine Schaltung an!*

- Y= ($\bar{A}$^D0) OR (A^D1)

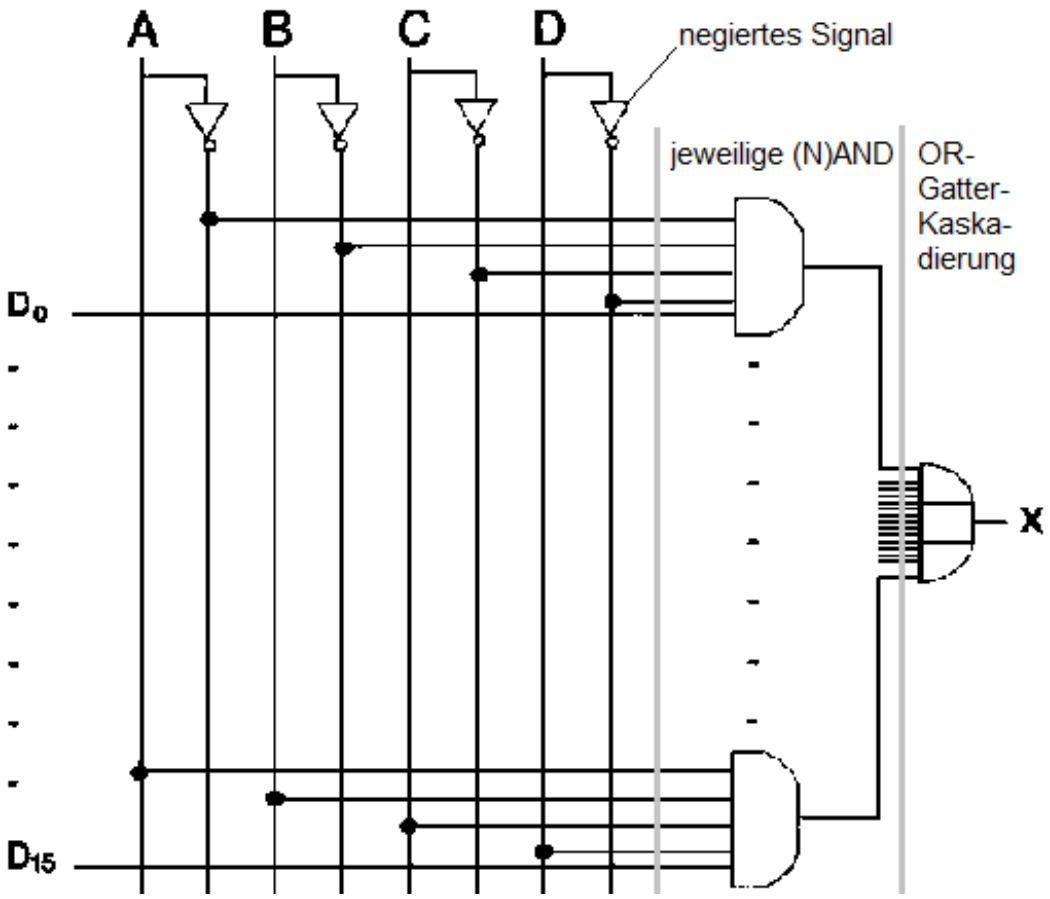



24. 16-auf-1-Multiplexers

- *Erläutern Sie die Funktion eines 16-auf-1-Multiplexers anhand des Blockschaltbilds und der Schaltbelegungstabelle!*
- *Welche Ein- und Ausgangssignale werden verwendet?*

| A | B | C | D | Y |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | D0 |
| 0 | 0 | 0 | 1 | D1 |
| 0 | 0 | 1 | 0 | D2 |
| 0 | 0 | 1 | 1 | D3 |
| 0 | 1 | 0 | 0 | D4 |
| 0 | 1 | 0 | 1 | D5 |
| 0 | 1 | 1 | 0 | D6 |
| 0 | 1 | 1 | 1 | D7 |

| A | B | C | D | Y |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | D8 |
| 1 | 0 | 0 | 1 | D9 |
| 1 | 0 | 1 | 0 | D10 |
| 1 | 0 | 1 | 1 | D11 |
| 1 | 1 | 0 | 0 | D12 |
| 1 | 1 | 0 | 1 | D13 |
| 1 | 1 | 1 | 0 | D14 |
| 1 | 1 | 1 | 1 | D15 |

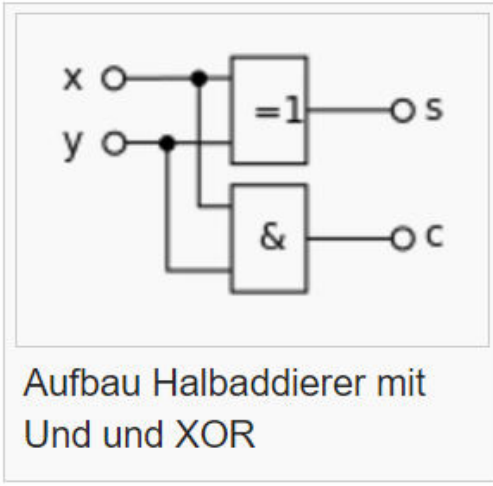



*(Blockschaltbild = D's & A,B,C,D gehen in einem Block aus dem X o. Y rausgeht)*

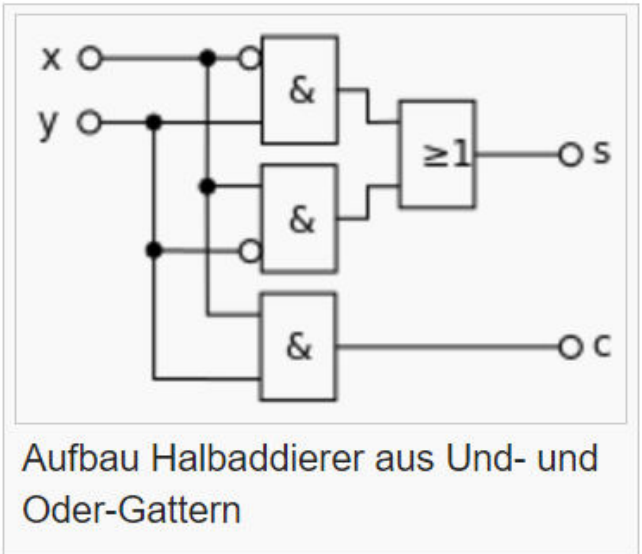
- Eingang: $D_0$ bis $D_{15}$
- Kontrolleingänge: A,B,C,D
- Ausgang: X/Y

25. Halbaddierer

- *Geben Sie die Schaltbelegungstabelle, die Funktionsgleichung sowie eine logische Schaltung mit Grundgattern für einen Halbaddierer an!*




Aufbau Halbaddierer mit Und und XOR




Aufbau Halbaddierer aus Und- und Oder-Gattern

| x | y | Übertrag c | Summe s |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

Das entspricht den Gleichungen

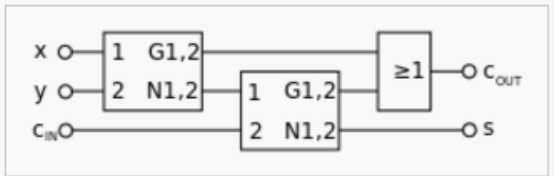
$$c = x \wedge y$$

und

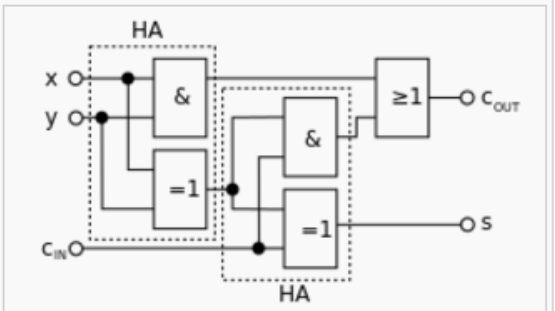
$$s = x \veebar y = x \oplus y = (x \wedge \neg y) \vee (\neg x \wedge y)$$

$\veebar$ $\oplus$ XOR $\neg$ NOT

26. Zeigen Sie die Kaskadierung von 3 Volladdierern!

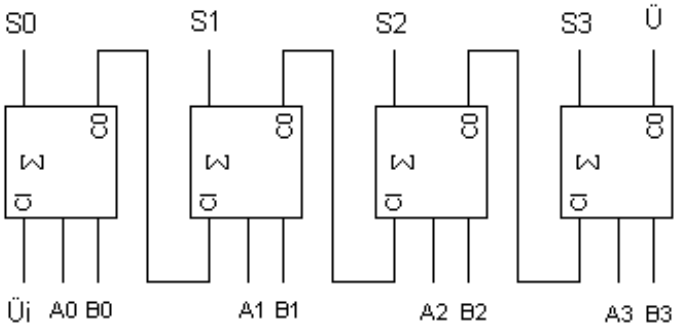



Aufbau eines Volladdierers mittels zweier Halbaddierer und eines Oder-Gatters nach DIN 40900




Aufbau eines Volladdierers mit zwei UND-, zwei XOR- und einem ODER-Gatter nach DIN 40900

*(C - Carry (in - von vorigen Übertrag ; out - jetziger Übertrag) , S wieder "richtiger" Wert)*




*(hier 4 Volladdierer | A/B - X/Y | Stellenweises Addieren einer 4Bit-Zahl)*

27. asynchrone Zähler

- *Wie funktionieren asynchrone Zähler und was ist bzgl. des zeitlichen Verhaltens zu beachten?*

- siehe 16&17
- die Kombination aus Verbindung des Ausgangs mit dem nächsten Takteingang & der Flankensteuerung ermöglichen das "binäre" Hoch- bzw. Runterzählen (mit Überlauf)
- Verzögerung (auf letzte Stelle) = einzelnes Zeitverhalten * Gesamtanzahl

# Rechner

28. Wie können Rechnersysteme bezüglich der Verarbeitung von Befehlen und Daten klassifiziert werden (Flynn)?

Die von Flynn angegebene Klassifizierung für Rechnerarchitekturen orientiert sich

an der Effektivität verschiedener Organisationsformen. Rechner werden hierin als Operatoren auf zwei verschiedenen Informationsströmen, dem Befehlstrom und dem Datenstrom, angesehen. Dementsprechend ergibt sich eine zweidimensionale Klassifizierung nach den Kriterien

• Ein Rechner bearbeitet zu einem Zeitpunkt einen oder mehrere Instruktionen.
• Ein Rechner bearbeitet zu einem Zeitpunkt einen oder mehrere Datenwerte.

Die entspricht der Einteilung in die 4 Klassen SISD, SIMD, MISD und MIMD, wobei der Klasse MISD nur eine Bedeutung innerhalb von Teilbereichen der CPU, etwa im Pipeliningaufbau, zukommt. SISD bezeichnet die klassischen VonNeumann-Rechner, SIMD die Vektorprozessoren, MIMD die Parallelrechner.
Die Flynn'sche Klassifizierung enthält zwei wesentliche Schwachpunkte:
1. Das sehr hohe Abstraktionsniveau der einzelnen Klassen führt dazu, daß sehr unterschiedliche Rechnerarchitekturen letztendlich in der gleichen Klasse geführt werden, obwohl sie unterschieden werden müssten.
2. Die Klasse MISD ist in der Systematik nur aus Vollständigkeitsgründen enthalten; gegenwärtige Rechnertypen in dieser Klasse existieren nicht.
Insbesondere der erste Schwachpunkt führt dazu, dass dieses Klassifizierungsschema in der Praxis zwar gerne für grobe Einteilungen (schlagwortartig) genutzt wird, real aber keine Relevanz besitzt und daher im Rahmen der weiterführenden Vorlesung nicht weiter angewendet wird. (TU-Clausthal)

~~29. Auswahl von Ein-/Ausgabebaugruppen~~

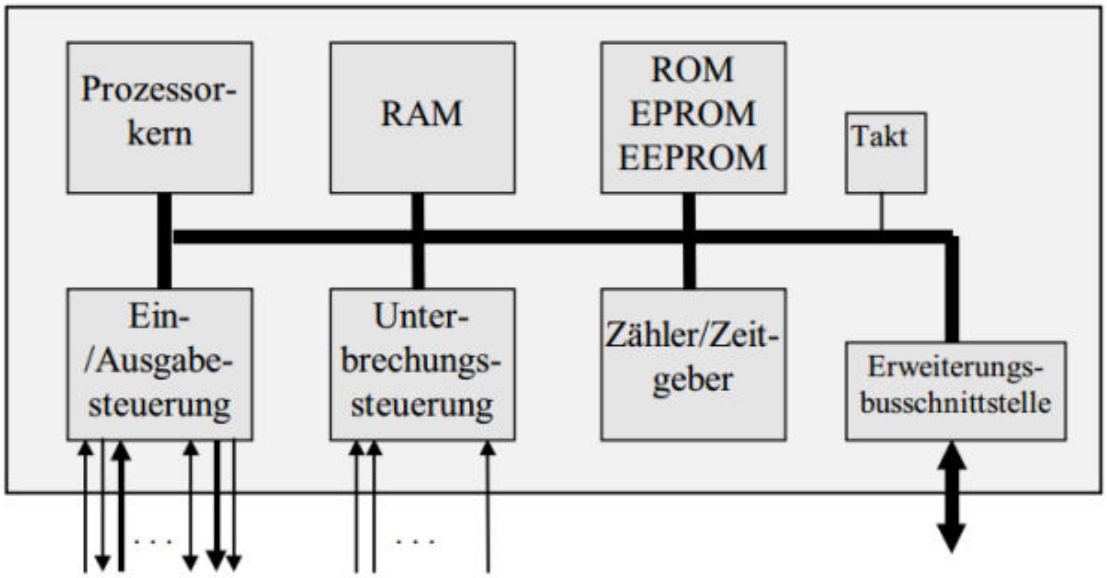
- ~~*Erläutern Sie das Prinzip der Auswahl von Ein-/Ausgabebaugruppen mittels eines Adressdecoders und die Adressierung!*~~
- ~~*Mit welchen Befehlen wird auf die Ports zugegriffen und welche Steuersignale werden dabei aktiviert?*~~

~~*30. Einchip-Mikrocontroller zum Messen*~~

- ~~*Ein Einchip-Mikrocontroller soll für eine Messaufgabe eingesetzt werden. Dazu werden ein 8-Bit-ADU zur Analogwertüberwachung sowie zwei Leuchtdioden zur Anzeige von Über- oder Unterschreitungen angeschlossen.*~~
- ~~*Zur Datenübernahme am ADU sind Handshake-Bits vorzusehen.*~~
  - ~~*a) Entwerfen Sie eine Schaltung!*~~
  - ~~*b) Wie funktioniert der Handshake-Betrieb?*~~

31. Microcontroller Struktur

- *Beschreiben Sie Mikrocontrollern zugrunde liegende Strukturen.*
- *Gehen Sie auf strukturelle Unterschiede ein!*
- *Welche Peripherieeinheiten sind typischerweise in aktuellen Mikrocontrollern integriert?*
- *Welche besonderen Eigenschaften privilegieren sie für den Einsatz in Smart Devices?*




Ein Mikrocontroller besteht vor allem aus:

- einem Prozessor, der die notwendigen Berechnung und andere Operationen durchführt
- meist zwei Speicherarten (flüchtiger Schreib-/Lesespeicher und nichtflüchtiger Festwertspeicher, zweiter ist nicht zwingend notwendig)
- einer Ein-/Ausgabesteuerung (E/A)

Zusätzlich wird im Normalfall

- eine Unterbrechungs-(Interrupt-)steuerung benötigt
- Zähler und/oder Zeitgeber
- diverse Erweiterungsschnittstellen (meist aus Kostengründen knapp gehalten )

Ein Mikrocontroller besitzt ein Prozessor und zusätzlich diverse Elemente auf einem Chip, er kann mit seinen drei Kernelementen (Prozessor, Speicher und Ein-/Ausgabe) bereits als kompletter Mikrorechner eingesetzt werden. Bei ihm geht es meist weniger um tatsächliche Rechenleistung als darum, seine Aufgaben kostengünstig und mit möglichst wenig externen Bauteilen zu erfüllen.

Einsatz Smart-Device:

- in Leistung und Ausstattung auf die jeweilige Anwendung angepasst.
- dadurch deutliche Vorteile beiKosten und Leistungsaufnahme.
  (Kleine Mikrocontroller sind in höheren Stückzahlen für deutlich unter 1 € verfügbar)

Prozessortypen:

- CISC („Complex Instruction Set Computer“) - viele Befehlssätze
- RISC(„Reduced Instruction Set Computer“) - weniger Befehle, mehr Leistung & günstiger (da einfach herzustellen)
- Datenbreite (Einteilung nach 4,8,16,32 Bit)

## 32. Einchipcontroller

- *Skizzieren Sie den prinzipiellen Aufbau eines Einchipcontrollers und erläutern Sie in Stichpunkten seine Wirkungsweise der Bestandteile.*
- *Nennen Sie typische Anwendungen!*

- siehe 31

## 33. RAM

- *Wie erfolgen Auswahl und Zugriff auf eine Speicherzelle in einem Speicher mit wahlfreiem Zugriff (RAM) prinzipiell?*

- Der Speicherzugriff wird durch ein Programm veranlasst, das die Adresse eines bestimmten Speicherinhaltes kennt. Um den Zugriff auf die betreffende Speicherzelle zu veranlassen, wird zunächst die Adresse in das Speicheradressregister des Prozessors geladen. Dies bewirkt, dass die Adresse auch auf dem Adressbus angelegt wird. Wenn der Prozessor nun seinen lesenden Zugriff auf den Arbeitsspeicher ausführt, wird durch die auf dem Adressbus vorgegebene Adresse die richtige Speicherzelle ausgelesen und ihr Inhalt über den Datenbus in das Speicherinhaltsregister des Prozessors übertragen. Damit steht der angeforderte Inhalt der Speicherzelle für weitere Operationen zur Verfügung, und der Speicherzugriff ist beendet.

34. Befehlspipelining (Fließbandverarbeitung)

- *Erläutern Sie die Grundidee und die prinzipielle Funktionsweise des Befehlspipelinings in Prozessoren!*

- Befehle üerlappend & parallel durch Bearbeitungsstufen schleusen
- während ein Befehl bearbeitet wird, wird der nächste bereits geladen
- mit jeden Takt:
    1. der letzte Befehl nach dem Zurückschreibe der Ergebnisse aus der Pipeline herausgenommen
    2. jeder andere Befehl in die nachfolgende Bearbeitungsstufe gerüct
    3. in die erste Stufe der Pipeline ein neuer Befehl aufgenommen

| Takt | Befehl lesen | Dekodieren | Operanden lesen | Ausführen | Ergebnisse zurückschr. |
|---|---|---|---|---|---|
| 1 | Befehl 1 | | | | |
| 2 | Befehl 2 | Befehl 1 | | | |
| 3 | Befehl 3 | Befehl 2 | Befehl 1 | | |
| 4 | Befehl 4 | Befehl 3 | Befehl 2 | Befehl 1 | |
| 5 | Befehl 5 | Befehl 4 | Befehl 3 | Befehl 2 | Befehl 1 |
| 6 | Befehl 6 | Befehl 5 | Befehl 4 | Befehl 3 | Befehl 2 |
| 7 | Befehl 7 | Befehl 6 | Befehl 5 | Befehl 4 | Befehl 3 |
| 8 | Befehl 8 | Befehl 7 | Befehl 6 | Befehl 5 | Befehl 4 |

Latenzzeit

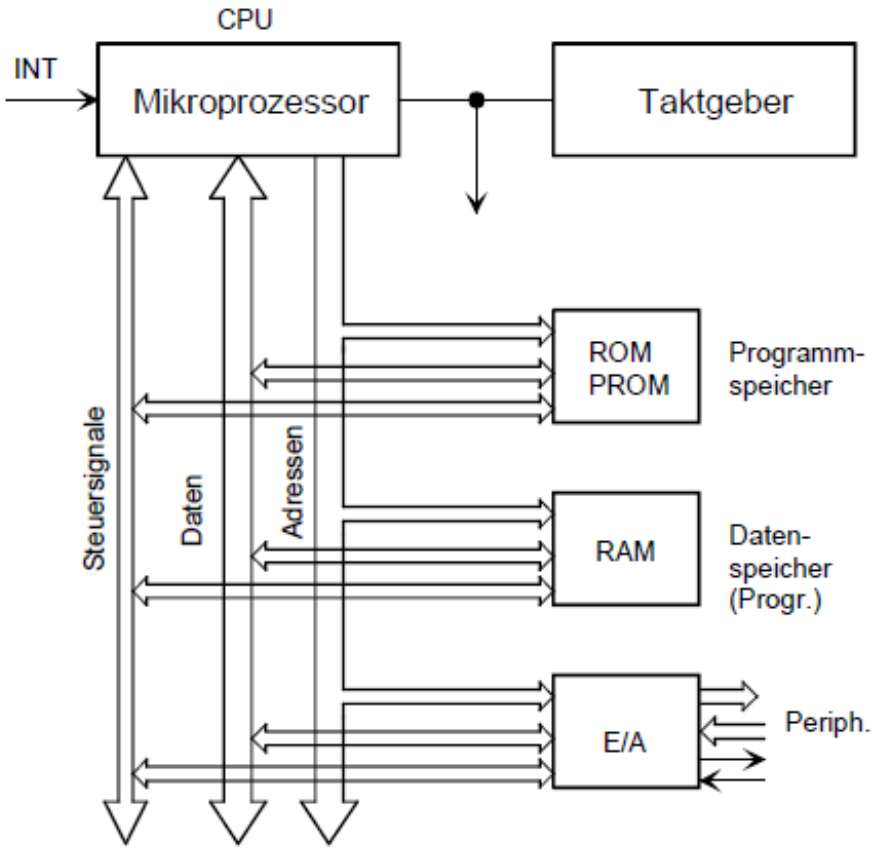
ein Befehl pro Takt

Abbildung 11.2: Eine ideale 5-stufige Pipeline. Nachdem die Pipeline gefüllt ist (Latenzzeit) wird ein Befehl pro Takt beendet.

([https://books.google.de/books?id=cTq6c9goVe8C](https://books.google.de/books?id=cTq6c9goVe8C))

35. Aufbau einfacher Rechner

- *Beschreiben Sie den Aufbau einfacher Rechner und die Funktion ihrer Elemente hinsichtlich ihres Zusammenwirkens* auf *Busebene!*




- 

36. Befehlsabarbeitung in einfachen Rechnern

- *Erklären Sie, wie die Befehlsabarbeitung in einfachen Rechnern erfolgt!*

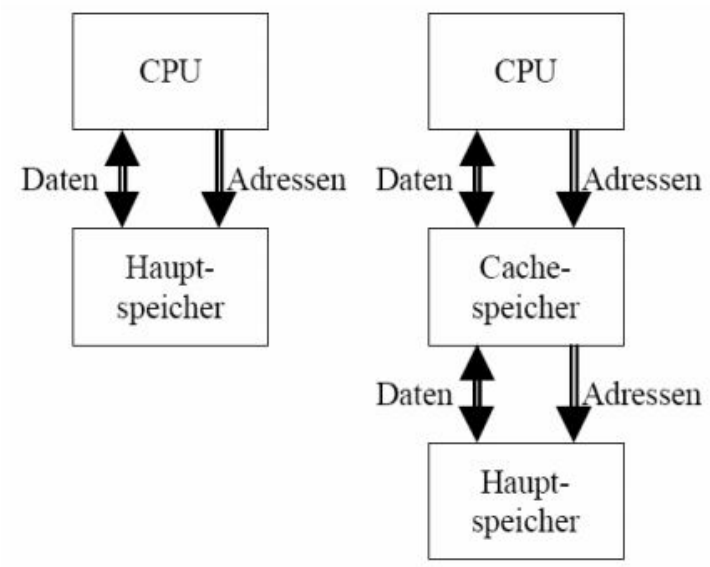
- in einem einfachen CISC-Prozessor / Von-Neuman:
  1.Takt: Befehlslesezyklus (Opcode Fetch) / Befehlholphase (BH)
  2.Takt: Dekordieren (Decode) /Dekodierphase (DE)
  3.Takt: Operanden einlesen (Operand Fetch) /Operandenholphase (OH)
  4.Takt: Ausführen (Execute) /Ausführungsphsae (AP)
  5.Takt: Zurückschreiben (Write Back) /Rückschreibphase (RS)
  (6.Takt:) /Adressierungsphsae (AD)

  (http://me-lrt.de/befehlsabarbeitung-phase-neumann)

37. Cache-Speicher

- *Erläutern Sie die Aufgabe und die Wirkungsweise von Cache-Speichern an einem Blockschaltbild!*
- *Wodurch und unter welchen Bedingungen wird der Datenzugriff gegenüber einem Zugriff auf den Hauptspeicher beschleunigt?*

- Unter dem Begriff "Cache" versteht man einen schnellen Puffer-Speicher, der sich zwischen dem Speicher und der CPU befindet und dabei hilft, Daten zwischenzuspeichern. (kürzere Zugriffszeiten)

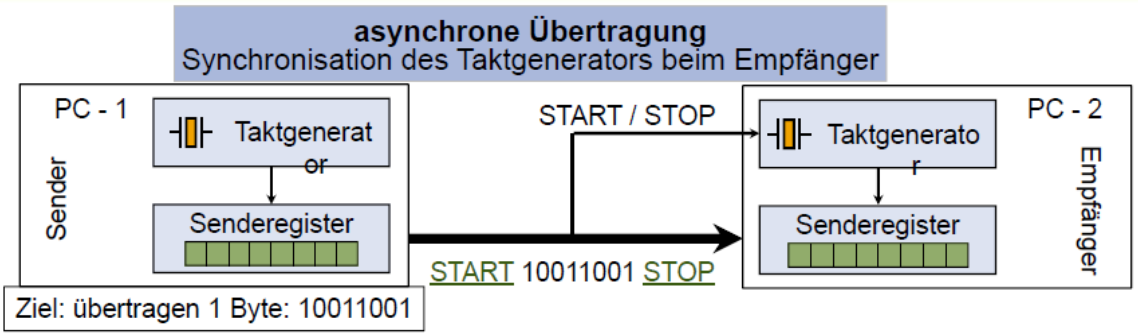



- bei (skalaren) Mikroprozessoren
- langsames Hintergrundmedium
- niedrigen Budge (Datenübertragungsrate für Rest kann niedriger sein, bei guten Cache, da nur 1x in den Chache geladen werden muss)

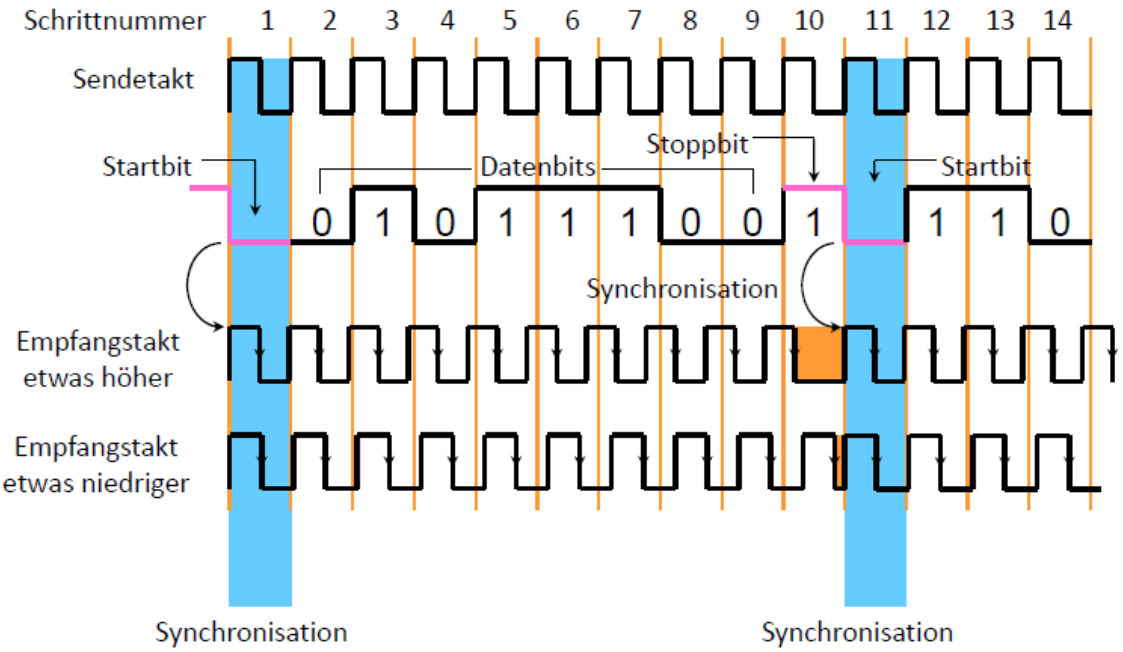
# Peripherieanbindung

38. asynchrone serielle Datenübertragung

- *Erläutern Sie das Prinzip einer asynchronen seriellen Datenübertragung unter Berücksichtigung von Hard- und Software von Sender und Empfänger!*
- *Erläutern Sie an einem Zeitdiagramm die Aufgabe von Start- und Stopbit.*

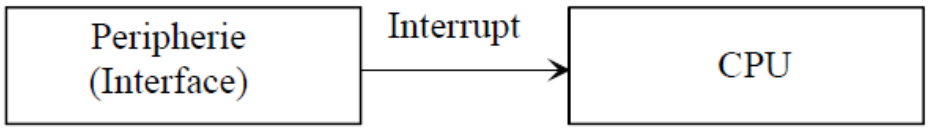



- Sender und Empfänger besitzen voneinander unabhängige (lokale) Taktgeber
- "Freie Leitung" entspricht einem kontinuierlich gesendeten 1-Bit
- Das Start-Bit setzt die Leitung auf 0 und startet den Taktgeber des Empfängers
- Ein Rahmen mit 5 bis 8 Bits (= ein Zeichen) wird übertragen
- Das Stopp-Bit setzt die Leitung wieder auf 1.
  Dieses Signal muss 1, 1,5 oder 2 Bit-Intervalle andauern.

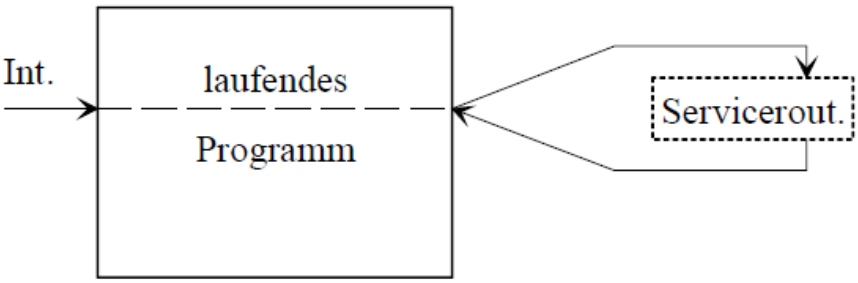


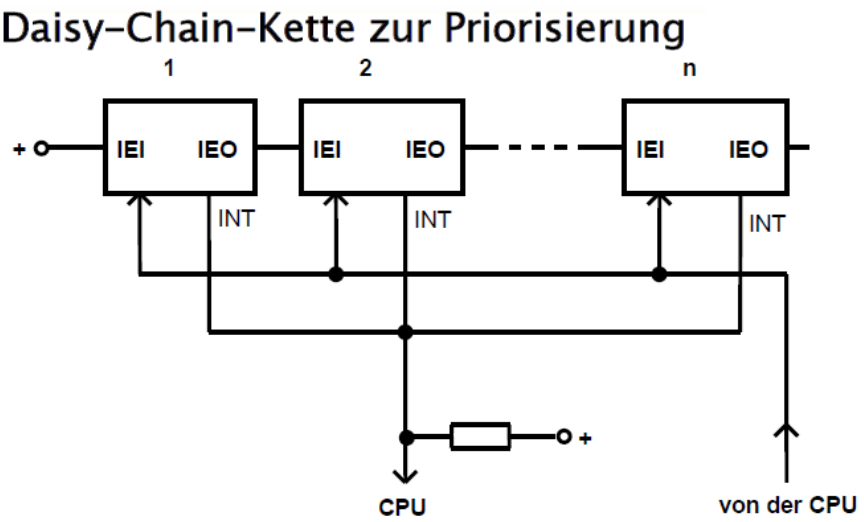

39. Interrupt

- *Wie erfolgt die Steuerung einer Eingabeoperation durch Interrupt?*
- *Erläutern Sie dabei die hardwareseitigen Voraussetzungen anhand der Verschaltung der beteiligten Bausteine!*
- *Was muss bei der Programmierung eines Interrupts beachtet werden und wie wirkt die Unterbrechung auf den Programmablauf der CPU ein?*




Grundgedanke:
Interrupt unterbricht laufendes Programm, das nach Abarbeitung der Serviceroutine fortgestzt wird!






- Was passiert beim Auftreten eines Interrupts?
  1. Sperren weiterer Unterbrechungen mit gleicher oder geringerer Priotität
     Unterbrechungen mit höherer Wichtigkeit dürfen normalerweise solche mit geringerer Wichtigkeit wieder unterbrechen
  2. Rettung wichtiger Register-Informationen(Prozessorstatus)
     alle Prozessor-Register retten, die durch die Interruptbehandlung überschrieben würden (heute gibt es dafür spezielle Maschinenbefehle)
  3. Bestimmen der Interruptquelle (durch Hardware realisiert)
  4. Laden des zugehörigen Interruptvektors
     d.h. das Herstellen des Anfangszustandes für gewählte Interruptroutine
  5. Abarbeitung der Interruptroutine
     *Retten weiterer Zustandsinformationen, sofern nicht durch Hardware realisiert meistens Übernahme weiterer Parameter von definierten Stellen (bei
     *Systemaufruf Ruf-Nr. und weitere Parameter oder bei Geräte-Interrupt Gerätestatusbits wie E/A Fortschritt, Fehler etc.)
     *eigentliche Behandlung des Interrupts, z.B. Setzen eines Flags (z.B. bei Gleitkommaüberlauf oder Aufruf zum Rückpositionieren und erneutem Lesen bei Lesefehler bei Magnetbandkassette (komplizierterer Fall)
  6. Rückkehr zur unterbrochenen Aufgabe entweder
     *Rückspeichern der geretteten Registerinformationen, d.h. Wiederherstellen des Prozessorzustandes
     *oder Bearbeitung einer neuen Aufgabe, z.B. bei Uhrinterrupt nach Ablauf einer Zeitscheibe
     *oder Zustand "HALT" nach schwerem Fehler, z.B. Spannungsausfall (abort)

## 40. Interrupt vs. Polling-Betrieb

- *Welchen Vorteil besitzt Interruptverarbeitung gegenüber dem Polling-Betrieb?*
- *Wie erfolgt die Priorisierung von Interrupts bei mehreren Interrupt-Quellen?*

- http://www.electronics-base.com/useful-info/software-related/90-polling-vs-interrupt
- Interrupt ermöglicht sofortiges reagieren auf Signale (z.B. wichtig bei Fehlern)
- bei Polling tritt im schlimmsten Fall ein Fehler direkt nach der Abfrage auf, welcher erst beim nächsten Polling erkannt wird (Zeitverzug)

- siehe 40 (Daisy-Chain-Kette)

## 41. Verarbeitung mehrerer Interrupts

- *Erläutern Sie die Verarbeitung mehrerer Interrupts verschiedener Interruptquellen!*
- *Was sind typische Interrupts und wie erfolgt der Sprung zur Interruptserviceroutine?*

- Interrupts werden nach ihrer Priorität abgearbeitet
  (z.B. 2 unterbricht 3, 1 unterbricht danach 2, nach 1 fertig wieder 2 dann 3)

- typische Interrupts:
  System-Timer
  Festplatten I/O
  Aus-Signale
  Fallen (ähnlich Interrupt)
  Datentrnsfer (z.B. Ethernet)
  I/O (z.B. Maus, Tastatur,....)

## 42. Timer

- *Erläutern Sie die prinzipielle Arbeitsweise eines Zähler-/Zeitgeberbausteins (Timer).*
- *Wie interagiert der Timer mit der CPU einerseits und der Peripherie andererseits?*

- eingehende Impulse senken oder erhöhen Zähler
- Zählerbetrieb (Counter)
  Impulse von außen werden gezählt
- Zeitgeberbetrieb (Timer)
  Impulse durch Herunterteilen des internen Iszillatortaktes

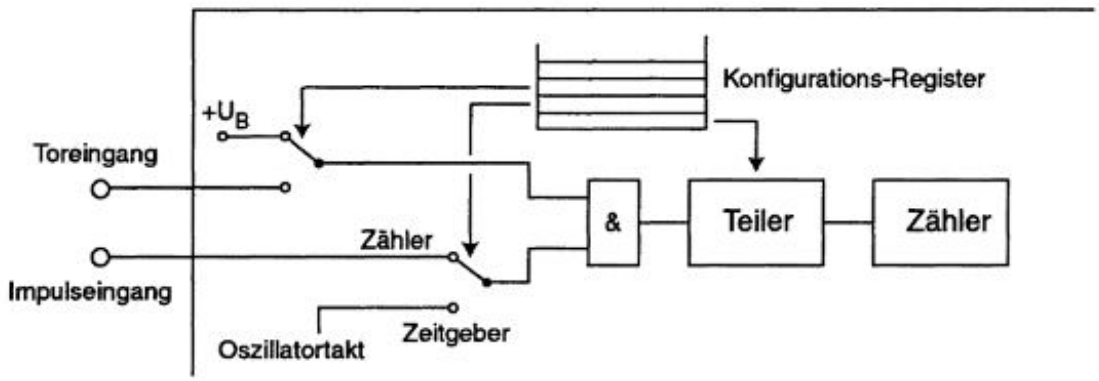



Abbildung 12.1: Eine Zähler/Zeitgeber-Einheit. Die Aktivierung des Toreinganges, die Umschaltung zwischen Zähler- und Zeitgeberfunktion und der Teiler werden jeweils durch Konfigurations-Register gesteuert.

- über Torsteuerung(Gated Timer) Erkennung von Impulsen über bestimmte Zeit
- Interaktion mit CPU - bei Überlauf auslösen eines Interrupts
- Interaktion mit Peripherie - Impulse messen, Daten rausgeben (Zeit, Anzahl...)
  *(https://books.google.de/books?id=laqlBgAAQBAJ)*

## 43. Bildansteuerung

- *Wie erfolgt die Ansteuerung des Bildschirms in einem Standard-PC?*

- Ansteuerung der einzelenen Bildpunkte (Pixel) mit jeweiligen Farbwert
  (True-Color: 24bit pro Pixel)
  (ähnlich zur folgenden 7-Segment-Anzeige)

## 44. BCD-Code

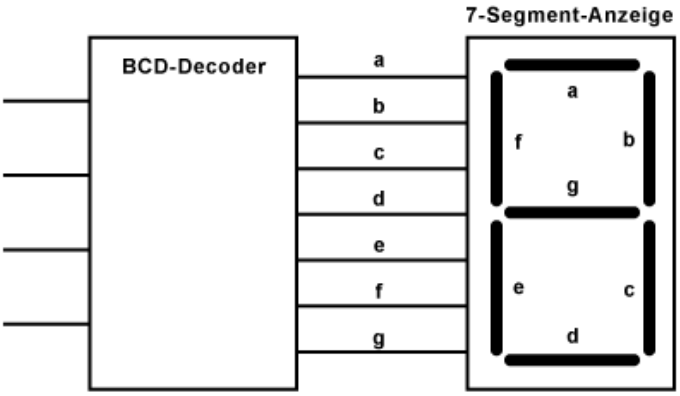
- *Was ist ein BCD-Code?*
- *Zeigen Sie, wie mit einem 7-Segment-Decoder eine entsprechende Anzeige angesteuert werden kann!*

- *Geben Sie eine Schaltbelegungstabelle (mindestens auszugweise) an!*

- BCD (Binary Coded Decimals - binär codierte Dezimalziffer)
- jede Dezimalziffer durch 4-Bit, also 4 binäre Stellen dargestellt

| Dezimal | $2^3$ | $2^2$ | $2^1$ | $2^0$ | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | Tetraden |
| 1 | 0 | 0 | 0 | 1 | |
| 2 | 0 | 0 | 1 | 0 | |
| 3 | 0 | 0 | 1 | 1 | |
| 4 | 0 | 1 | 0 | 0 | |
| 5 | 0 | 1 | 0 | 1 | |
| 6 | 0 | 1 | 1 | 0 | |
| 7 | 0 | 1 | 1 | 1 | |
| 8 | 1 | 0 | 0 | 0 | |
| 9 | 1 | 0 | 0 | 1 | |
| | 1 | 0 | 1 | 0 | Pseudotetraden |
| | 1 | 0 | 1 | 1 | |
| | 1 | 1 | 0 | 0 | |
| | 1 | 1 | 0 | 1 | |
| | 1 | 1 | 1 | 0 | |
| | 1 | 1 | 1 | 1 | |

**7-Segment-Anzeige**

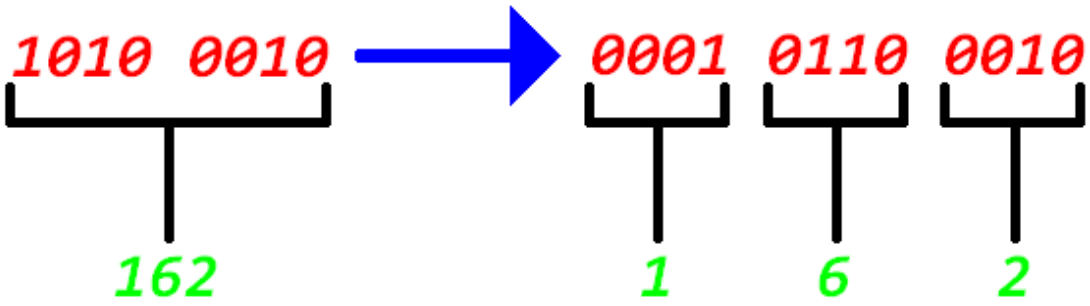



| d\y | a | b | c | d | e | f | g |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 2 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 3 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 4 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| 5 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 6 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 7 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 9 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |

45. 8-Bit -> BCD

- *Erklären Sie, wie ein dual codierter 8-Bit-Wert x in eine entsprechende Anzahl von BCD-Stellen umgewandelt werden kann!*
- *Mit Hilfe von 7-Segment-Decodern soll derselbe Wert x dezimal angezeigt werden.*
- *Wieviel Dezimalstellen werden benötigt?*
- *Wie erfolgt die Umwandlung der einzelnen Stellen?*

# Binary to BCD Conversion Algorithm




## Algorithm:

1. If any column (100's, 10's, 1's, etc.) is 5 or greater, add 3 to that column.
2. Shift all #'s to the left 1 position.
3. If 8 shifts have been performed, it's done! Evaluate each column for the BCD values.
4. Go to step 1.

## Algorithm In Action:

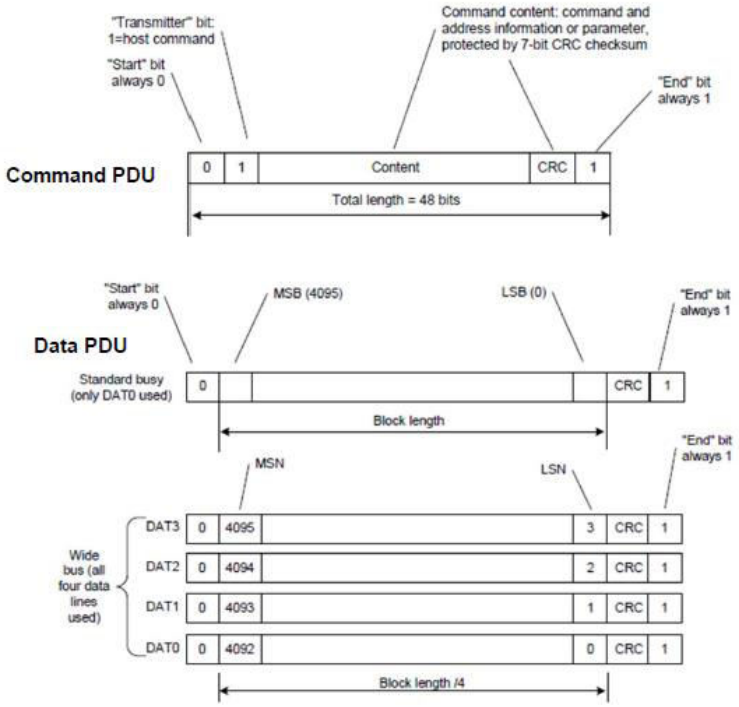
| 100's | 10's | 1's | Binary | Operation |
|---|---|---|---|---|
| | | | 1010 0010 | 162 |
| | | 1 | 010 0010 | << #1 |
| | | 10 | 10 0010 | << #2 |
| | | 101 | 0 0010 | << #3 |
| | | 1000 | | add 3 |
| | 1 | 0000 | 0010 | << #4 |
| | 10 | 0000 | 010 | << #5 |
| | 100 | 0000 | 10 | << #6 |
| | 1000 | 0001 | 0 | << #7 |
| | 1011 | | | add 3 |
| 1 | 0110 | 0010 | | << #8 |

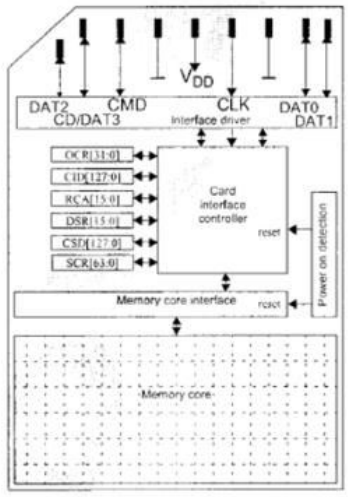
1 6 2

(http://www.eng.utah.edu/~nmcdonal/Tutorials/BCDTutorial/BCDConversion.html)

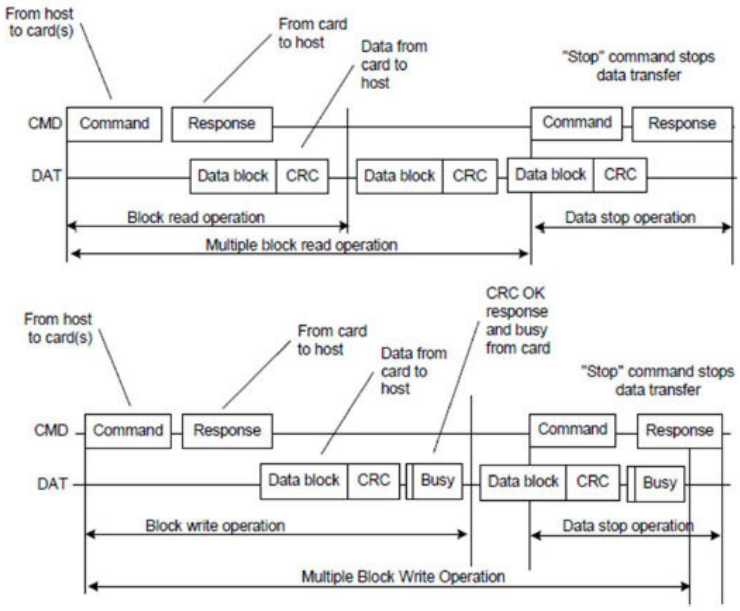
46. SD-Karte

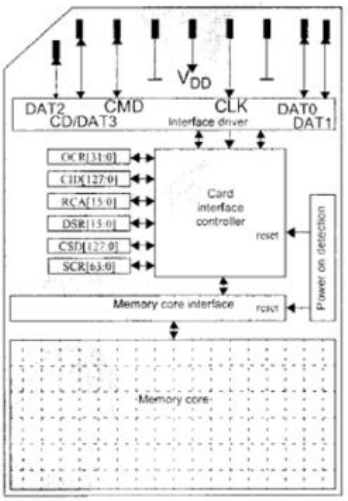
- *Erläutern Sie die prinzipielle Funktionsweise einer SD-Karte!*

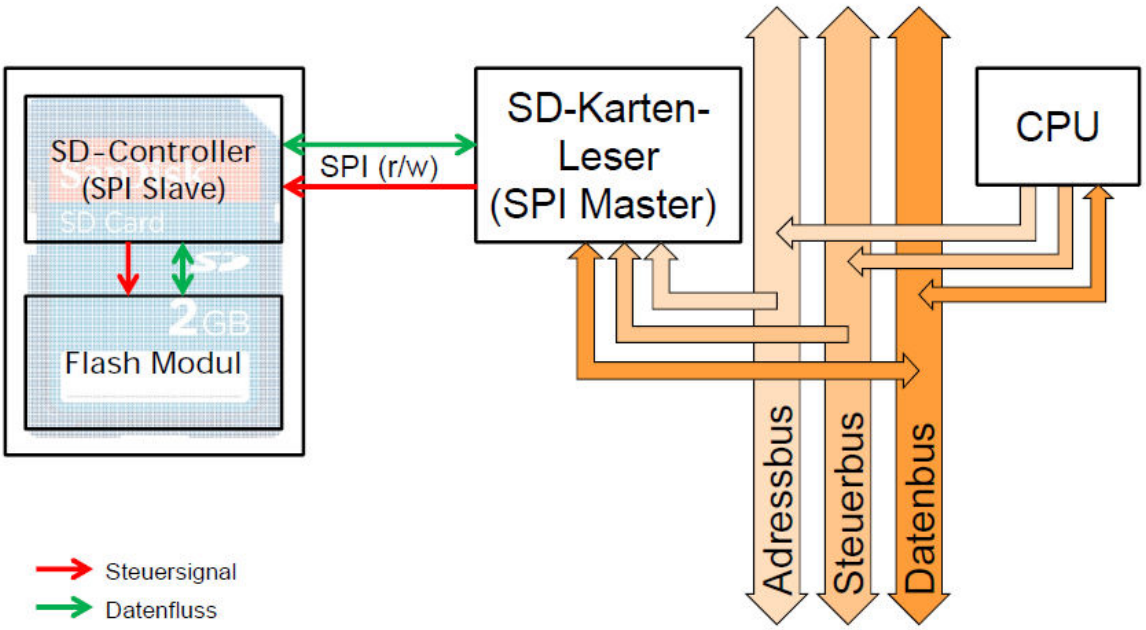



PDU – Protocol data unit






- Pnzipielle Funktionsweise:
  CPU sendet Befehl an SD-Karte (Adresse + Befehl)
  SD-PDU liest und führt den Befehl aus (falls erlaubt)
  Daten von Datenbus auf SD (Flash-Speicher) schreiben
  oder Daten von SD auf Datenbus legen

47. optisches Speichermedium

- *Wie funktioniert die Speicherung von Daten auf einem optischen Speichermedium?*
- *Was ist beim Auslesen der Daten zu beachten?*

- z.B. CD - auf Oberflächen Vertiefungen (0) und Höhen (1)
- Niederfrequente Signalanteile stören den Spurfolgemechanismus
- Laser kann aufeinanderfolgende Einsen nicht erkennen
- Codierung von 8 Bit auf 14 Bit
- 2-10 Nullen zwischen zwei Einsen
- Trennbits zwischen Codewörtern:
  000, 001, 010 oder 100
- Gleichmäßige Verteilung von Einsen

48. Anschluss peripherer Geräte über serielle Bussysteme

- *Moderne Computer ermöglichen den Anschluss peripherer Geräte über serielle Bussysteme.*
- *Weit verbreitet sind USB und Firewire.*
- *Charakterisieren Sie diese kurz und erläutern Sie Vor- und Nachteile gegenüber einer Datenein-/-ausgabe über eine Schnittstelle, die direkt den Systembus des Rechners nutzt!*

- USB&Firewire besitzen jeweils einen eigenen Controller, welcher die Kommunikation steuert
- USB 4 Leitungen (1xStrom,1xMasse, 1x Daten (normal), 1x Daten (invertiert)
- Firewire 4+ Leitungen ( evtl. (Strom&Masse), 4x Daten (wie USB))
- interner Controller hat DMA (DirectMemoryAcess) auf PC (per eigenes HUB)
  durch diesen werden u.a. Treiberinstallation ohne Nutzereinwirkung ermöglicht

- Datenübertragung serial - Bit für Bit (anstatt parallel wie bei SD)
- Geschwindigkeit direkt abhängig von USB-Controller (& evtl. CPU oder Festplatte)

- Vorteile (Schnittstelle gegen direkten Systembusanschluss):
  *universal (jedes Gerät an jedem System mit Schnittstelle nutzbar)
  *Auffächerung möglich (mehrere Geräte gleichzeitig an selber Schnittstelle -> mehr Geräte anschließbar)
  *eigenständige Peripheriegeräte ermöglichen durch Spezialisierung Beschleunigung spez. Vorgänge

- Nachteile:
  *unsicher (kaum Controlle der Vorgänge (von den angeschlossenen Geräte) vom System aus - Peripheriegeräte relativ eigenständig)

## 49. Anzeige

- *Erläutern Sie, wie auf einer Anzeige (Display, Typ Ihrer Wahl) der Farbwert eines Pixels erzeugt werden kann!*

- LED-Anzeige: ein Pixel besteht aus 3 dimmbaren LED's (Rot,Grün,Blau), welche einzeln angesteuert werden und zusammen den Farbwert eines Pixels ergeben

## 50. Peripheriebusse

- *Erläutern Sie grundlegende Eigenschaften eines Peripheriebusses Ihrer Wahl!*

- Datenbus:
  * überträgt bidirektional Daten zwischen den Peripheriegeräten
  * Datenbusbreite meist = Datenbreite der CPU
- Adressbus:
  * übertragt Speicheraddressen unidirektional(einseitig) vom Busmaster (meist CPU) an Peripheriegeräte
  * Busbreite = gleichzeitig ansprechbarer Speicher
- Steuerbus (Kontrolbus)
  * unidirektional
  * bewerktstelligt Bussystemsteuerung
  * u.a. Leitungen für die Lese-/Schreib-Steuerung (Richtung auf dem Datenbus)
    Interrupt-Steuerung
    Buszugriffssteuerung
    Taktung (falls ein Bustakt erforderlich ist)
    Reset- und Statusleitungen

## 51. Polling

- *Was verstehen Sie unter den Begriff „Polling“ im Zusammenhang mit der Einbindung von Peripheriebaugruppen?*
- *Welche Vor- und Nachteile beinhaltet das Verfahren?*

- Polling ist das zyklische Abfragen von einen oder mehreren E/A-Devices zur Feststellung der Kommunikationsbereitschaft bzw. zum Einholen von Kommunikationswünschen.

- Vorteile des Pollings:
  Einfach zu Implementieren
  Kommunikationsanforderungen erfolgen synchron zum Programmablauf

Je mehr Geräte am Bus hängen, um so mehr steigt Reaktionszeit

- Nachteile des Pollings
  Hoher Programm-Overhead
  Die meisten Anfragen an die Geräte sind unnötig
  . Priorisierung bei zeitgleichen Anfragen erfordert zusätzlichen Zeitaufwand

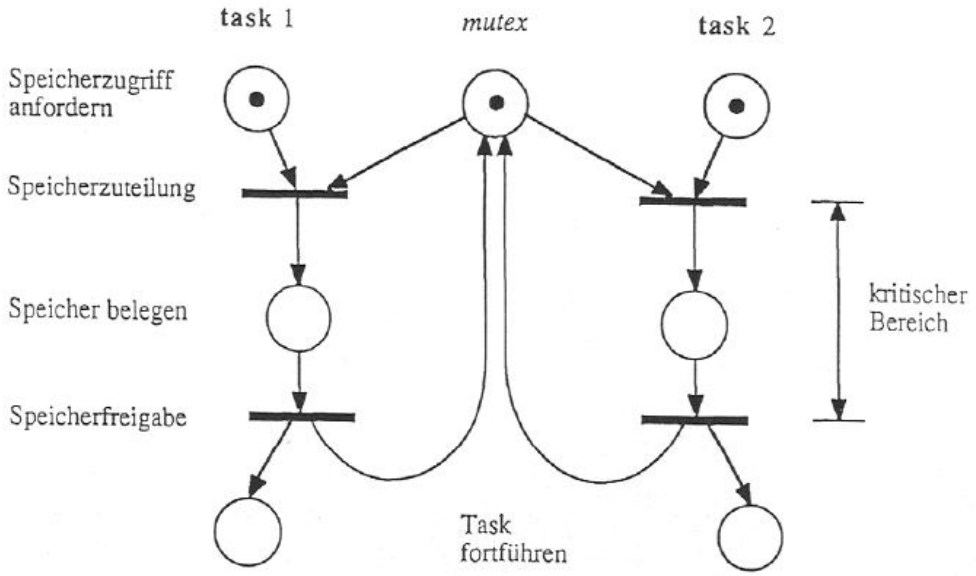
- Aufgrund der vielen Nachteile sollte besser eine asynchrone Kommunikation mit den Geräten durch die Hardware unterstützt werden (Interrupts).

# Echtzeit

52. wechselseitigen Ausschluss von Tasks

- *Erläutern Sie den wechselseitigen Ausschluss von Tasks in einem Echtzeit-System!*
- *Wie ist der kritische Bereich definiert?*

## Wechselseitiger Ausschluß




Petri-Netz-Darstellung des wechselseitigen Ausschlusses mittels des Semaphors *mutex*

Der Begriff wechselseitiger Ausschluss bzw. Mutex (Abk. für engl. mutual exclusion) bezeichnet eine Gruppe von Verfahren, mit denen das Problem des kritischen Abschnitts gelöst wird. Mutex-Verfahren verhindern, dass nebenläufige Prozesse bzw. Threads gleichzeitig oder zeitlich verschränkt gemeinsam genutzte Datenstrukturen unkoordiniert verändern, wodurch die Datenstrukturen in einen inkonsistenten Zustand geraten können, auch wenn die Aktionen jedes einzelnen Prozesses oder Threads für sich betrachtet konsistenzerhaltend sind. Mutex-Verfahren koordinieren den zeitlichen Ablauf nebenläufiger Prozesse/Threads derart, dass andere Prozesse/Threads von der Ausführung kritischer Abschnitte ausgeschlossen sind, wenn sich bereits ein Prozess/Thread im kritischen Abschnitt befindet (die Datenstruktur verändert).

Solange ein Task bestimmte Resourcen nutzt, kann ein anderer, welche die Resourcen benötigt (je nach der verfügbaren Zeit) nicht starten.

**Kritischer Abschnitt**

- Kennzeichnung einer Ansammlung von Programmanweisungen zum Zwecke der Ablaufsteuerung
- zu selben Zeit nur ein einziger Prozess/Thread (im krit. Bereich)

- Kritische Abschnitte bestehen aus mehreren Einzelanweisungen, deren Zwischenergebnisse inkonsistente Zustände darstellen, auf die die anderen Threads keinen Zugriff erhalten dürfen. Das Ergebnis eines kritischen Abschnitts darf nur als eine unteilbare Einheit nach außen sichtbar werden.

53. Echtzeitsysteme

- *Wie funktionieren Echtzeitsysteme?*

  Als Echtzeitsysteme (englisch real-time systems) werden „Systeme zur unmittelbaren Steuerung und Abwicklung von Prozessen“[1] bezeichnet, die dafür an sie gestellte quantitative Echtzeitanforderungen erfüllen müssen. -Wikipedia

- liefern in gegebener Zeit garantiert ein korrektes Ergebnis

**Arten:**

- harte Echtzeitsystem:
  Ergebnis MUSS im Zeitintervall vorliegen
- weiche Echtzeitsystem:
  Ergebnis KANN im Zeitintervall vorliegen
  Überschreitung möglich
  solange Durchschnitt OK Aufgabe erfüllt
- feste Echtzeitsystem:
  Ergebnis SOLLTE im Zeitintervall vorliegen
  bei Überschreitung der Zeit wird das Ergebis verworfen,allerdings ohne Schaden

**Gestaltungsparadigmen:**

- ereignisgesteuert
  *schnellstmögliche Reaktion auf von außen kommendes Ereignis
  *meist mittels Interrupt
  *mögliche Überlastung bei zu vielen auf einmal auftretenden Ereignissen
  daher meist Reserve geplannt
  Abhandlung per Priorität, wobei trotzdem alles erfüllt werden muss

- zeitgesteuert
  *Prozesse aufgrund eines festgelegten Zeitplans gestartet
  *jeder Prozess hat eine genaue Ablaufzeit (z.B. 10ms von 100ms)
  *Überlastungen kaum möglich, jedoch müssen genaue Zeiten bekannt sein
  & alles Bugfrei (keine Überschreitung des PLans)
  *100% Auslastung möglich

**Umsetzung**

- feste periodische Triggerung
  *geforderte Reaktionszeit einer spez. Anwendung erfüllen
  *eigene Funktionseinheit die Aufgabe mit bestimmter Frequenz erfüllt
  (f = 1/Reaktionszeit | z.B. Musikwiedergabe mit 44,1 kHz (entspricht Reaktionszeit $\leq$ 22,7 Mikrosekunden)
  *erfüllt zuveerlässig hartes EZ-Kriteritum (da speziel für Aufgabe entwickelt)

- synchrone Ansätze
- prozeessbasierte Ansätze

54. Task-Betrieb

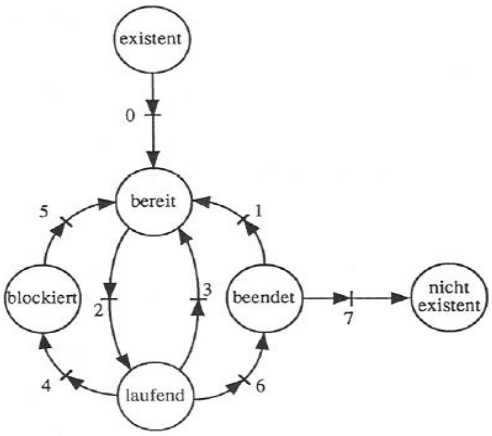
- *Erläutern Sie den Task-Betrieb in einem Echtzeit-System!*

- *Gehen Sie dazu auf Taskzustandsautomaten und den Task Control Block ein!*

Task-Zustände:

- Bereit
- Laufend
- Wartend/Blockiert/Suspendiert
- Leerlauf
- Beendet

**Task-Zustandsautomat**




0 Task wird erzeugt
1 Erneuter Taskstart
2 Prozessor zugeteilt
3 Task wird verdrängt
4 Betriebsmittel fehlen
5 Betriebsmittel vorhanden
6 Task abgearbeitet
7 Task eliminiert

b)

Mögliche Taskzustände (a) und Bedingungen für den Eintritt der Zustandsübergänge (b)

- 4 - Betriebsmittel fehlen - siehe 52.

**Task-Control-Block**

**Struktur TCB**

- Taskverwaltung
  - Befehlszähler
  - Befehlsregister
  - Statusregister
  - Register
  - Taskidentifikation
  - Taskpriorität
  - Taskzustand
  - Elterntask
  - Kindtask

- Speicherverwaltung
  - Anfangsadresse des Taskcodes
  - Anfangsadresse des Rettbereiches für die Taskdatei
  - Stackzeiger
  - Identifikation von Taskgruppen belegter Arbeitsspeicher

- E/A-Verwaltung
  - Erteilte E/A-Aufträge
  - Zugriffsrechte auf Geräte
  - Angeforderte Geräte
  - Belegte Geräte
  - Zeiger auf Pufferbereich

# Quellen


- http://www.kreissl.info/ra - Interrupts
- http://www.elektronik-kompendium.de/ - Flip-Flops
- http://www.wikipedia.com - diverses, u.a. Echtzeitsysteme
- http://www6.in.tum.de/pub/Main/TeachingWs2014ProseminarMicrocontrollerEmbedded/Was_ist_ein_Microcontroller.pdf - Microcontroller
- http://rfcafe.com/references/electrical/neets-modules/NEETS-Module-13-3-31-3-40.htm
- http://www2.htw-dresden.de/~robge/ezs/vl/ezs-01-einfuehrung.pdf - Echtzeitsysteme
- http://user.it.uu.se/~yi/courses/rts/dvp-rts-08/notes/RTOS.pdf - Tasks
- Vorlesung(sfolien) & Übung an OvGU^^