Rechnerhardware

Literatur:

Bringschulte, Ungerer, „Mikrocontroller und Mikroprozessoren“, Springer
Bähring, Mikrorechnertechnik, Springer
aktuelle Informationen über Prozessoren, Chip-Sätze etc. im Internet

Rechnerhardware - Einführung

Rechner in der Physik heute meist PCs,
d.h. Mikrocomputersysteme bestehend aus:

- Mikroprozessor(en)
- (flüchtigem) Halbleiter-Speicher
- (permanentem Magnet-) Festplattenspeicher
- Ein-Ausgabeschnittstellen
- Tastatur, Bildschirm, Maus, Wechsellaufwerke, Drucker, Netzwerk, ...

Verfügbar als

- Einzelplatz-Desktop-PC
- vernetzter Cluster aus Desktop-PCs
- PC-Farm mit Fileservern
- vernetzte PC-Farmen („GRID“)

Rechnerhardware

Grundlage der Rechnertechnik ist die Digitalelektronik:

- Zustände 1 und 0 repräsentiert durch elektrische Signale

Boolesche Algebra mit Hilfe von elektronischen Bauteilen wie

- Gattern (AND, OR, XOR)
- Bistabilen Stufen („FlipFlops“) als Speicherelemente

daraus aufgebaut:

- Register, Addierer

und weiter

- Rechenwerk, Steuerwerk, Speicher eines Mikroprozessors

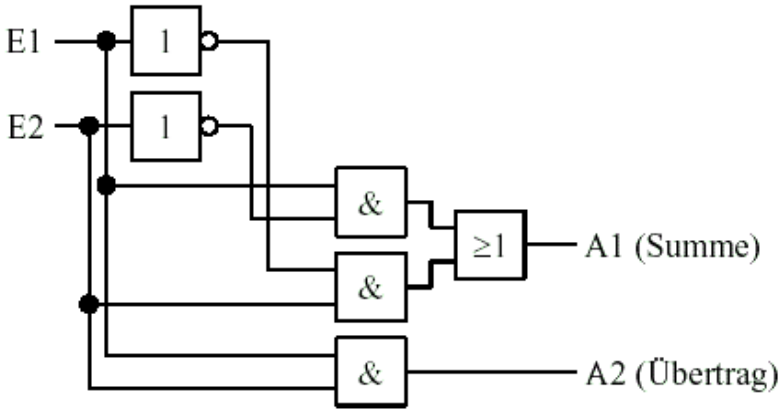
Zahlendarstellung im Binärsystem, d.h. z.B. 10 = 1010;
meist 4 Binärstellen als eine „Hex“-Ziffer angegeben im Sedezimalsystem,
also 1010=$A (oder 0xA)

Gesamte Rechnerhardware auf wenigen Chips integriert

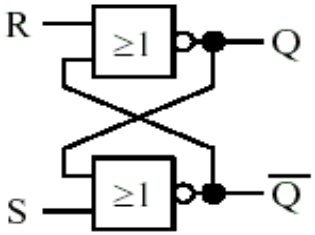
Stand der Technik 2012:

22 nm Sturkturgröße, ~800 Millionen Transistoren auf ~1 cm^2 Chip-Fläche

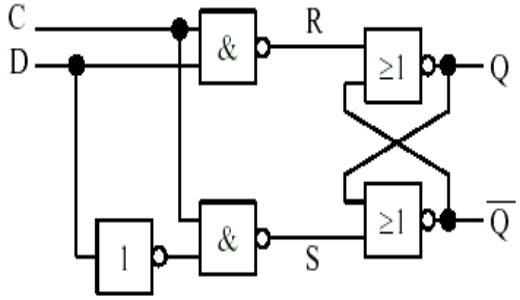
Rechnerhardware – einige logische Baugruppen



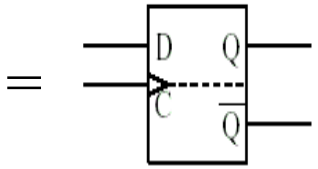
1Bit Halbaddierer

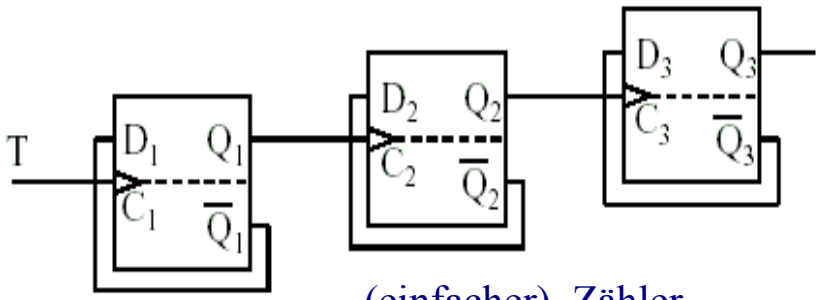


„Flip-Flop“, Speicher

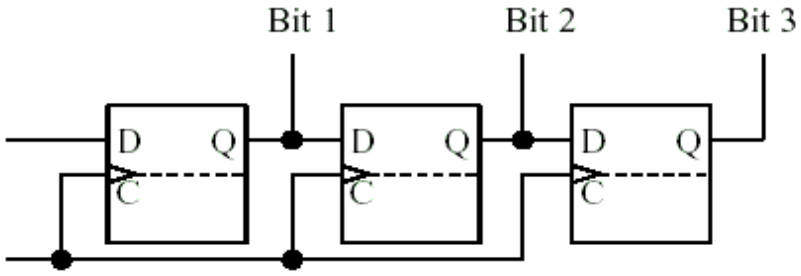


getakteter Speicher



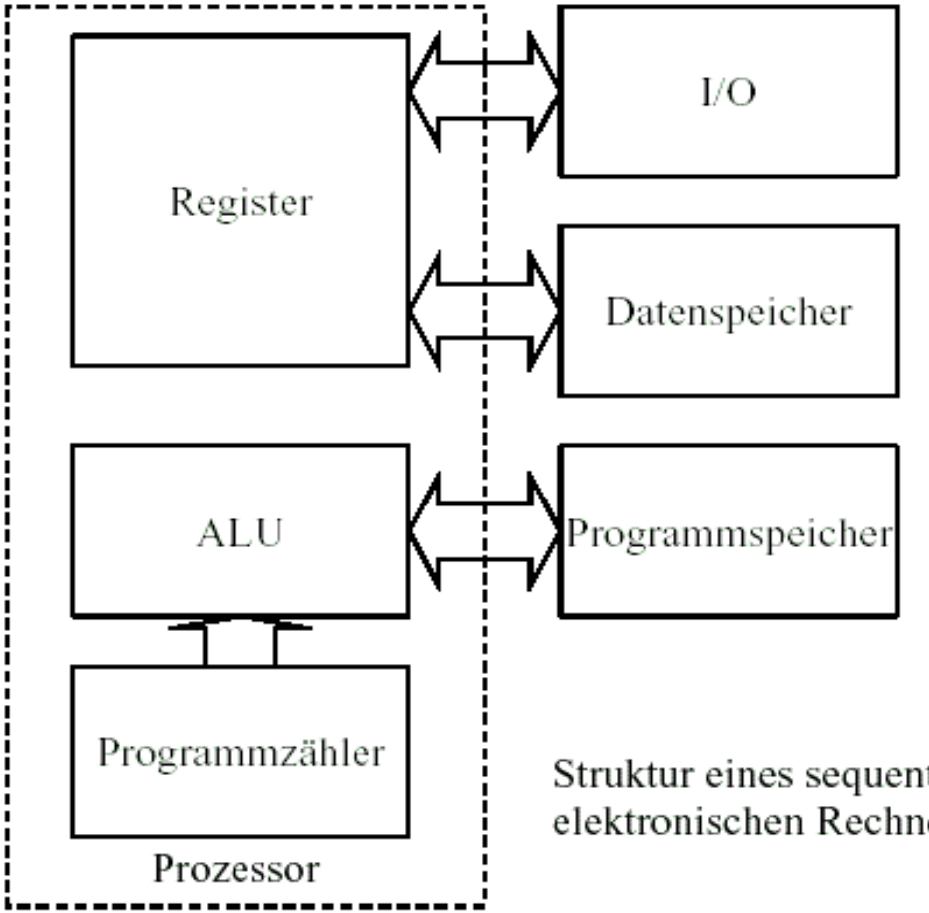


(einfacher) Zähler



Schieberegister, mult. mit 2

Hardware – logische Baugruppen

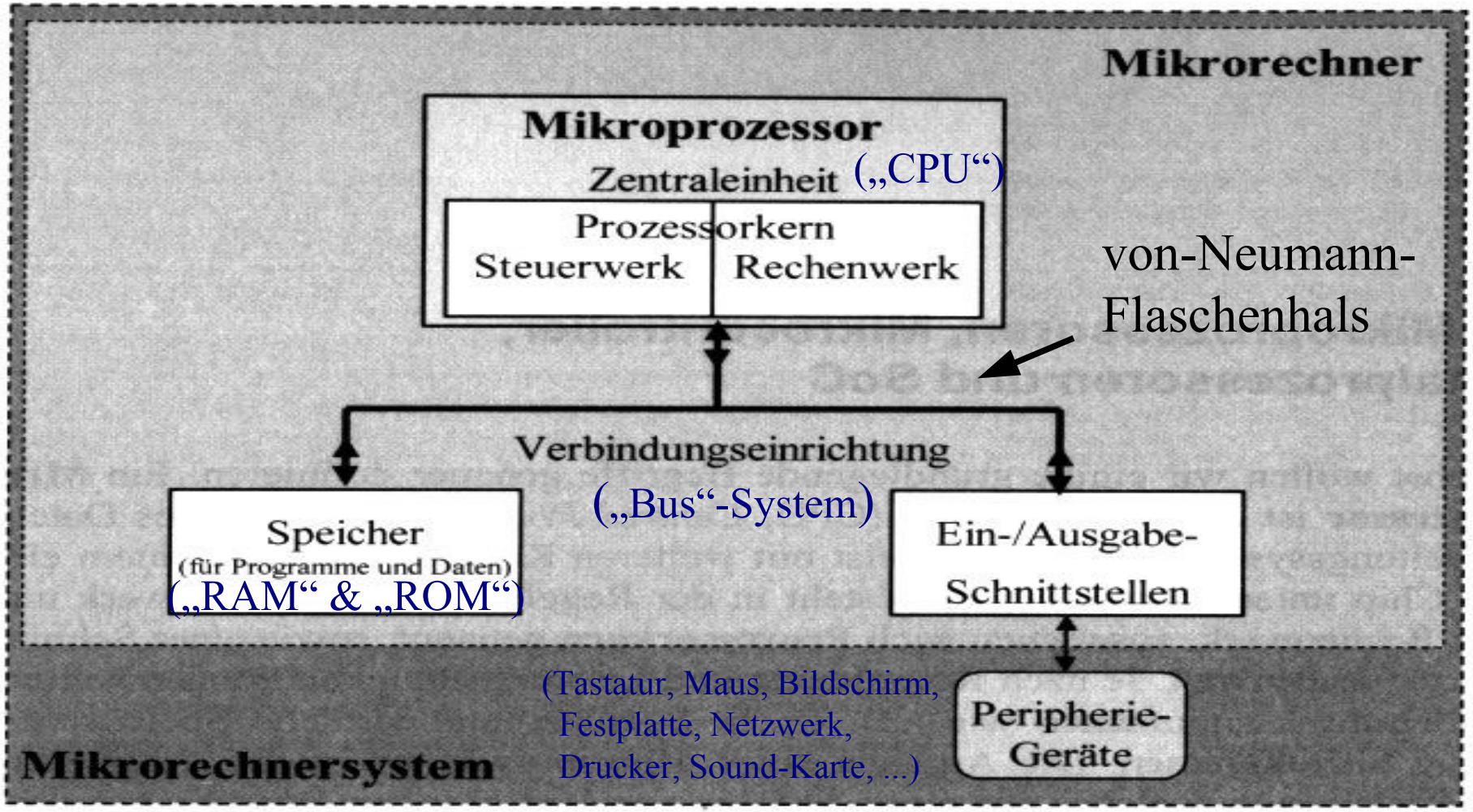


Struktur eines sequentiellen elektronischen Rechners

Eine typische Abfolge von Operationen in einem sequentiellen Rechner lautet:

Programmzähler	Befehl
k	Lade Datum A in Register 1
k+1	Lade Datum B in Register 2
k+2	Addiere Register 1 und Register 2 in Register 3
k+3	Speichere Register 3 in Datum C

Rechnerhardware – von-Neumann-Architektur



Getrennter Speicher für Programme und Daten: Harvard-Architektur

Hardware - Komponenten

Ein PC besteht aus

- Mikroprozessor (mehrere 1000 MIPS (Millionen Instruktionen/sec))
- (schnellem) Cache-Speicher mit Systemgeschwindigkeit
- einer „North-Bridge“ als Systembus zum Hauptspeicher und zur Grafikkarte
  Übertragungsrate ca. 8 Gbyte/sec
- einer Brücke zur Peripherie (Festplatte, CD-Rom, USB ...)
  Übertragungsrate von
  PCI-Express x1: 500 Mbyte/sec 2011: PCIe 3.0 x1: 985 MB/sec
  SATA: 150 Mbyte/sec 2011: SATA 3 ~600 MB/sec
  USB: 60 Mbyte/sec 2011: USB3.0 ~500 MB/sec
- Controllern für Datentransfers
  (z.B. „DMA“, **D**irect **M**emory **A**ccess, SCSI, IDE, USB, ...)

Rechnerhardware - Komponenten

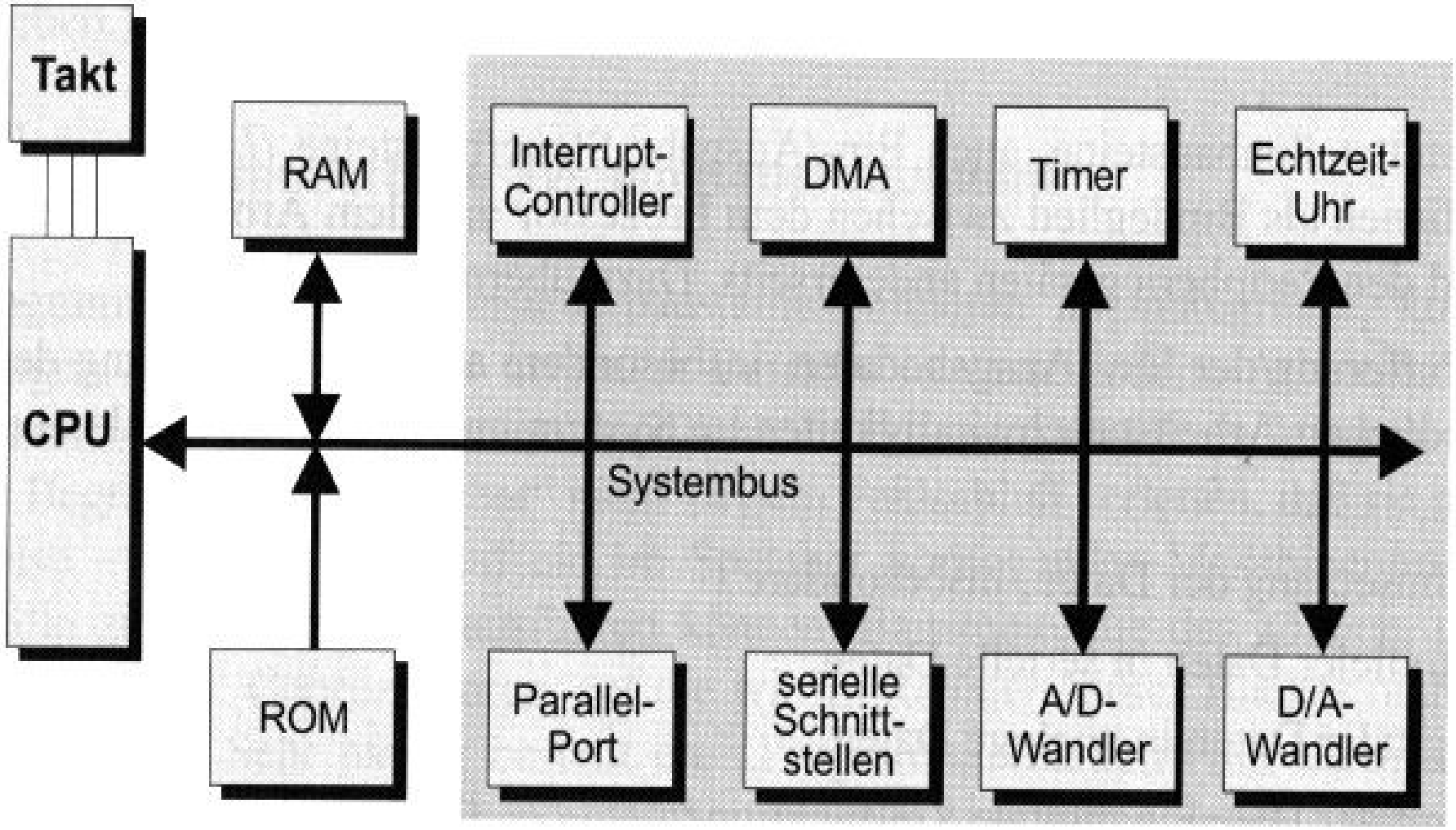


Bild 3.1-1: Die wichtigsten Systembausteine in einem Mikrorechner

Hardware - Mainboard

Typischerweise alle Komponenten eines PC (bis auf Netzteil,Laufwerke) auf einer Platine untergebracht

=> **„Mainboard“**

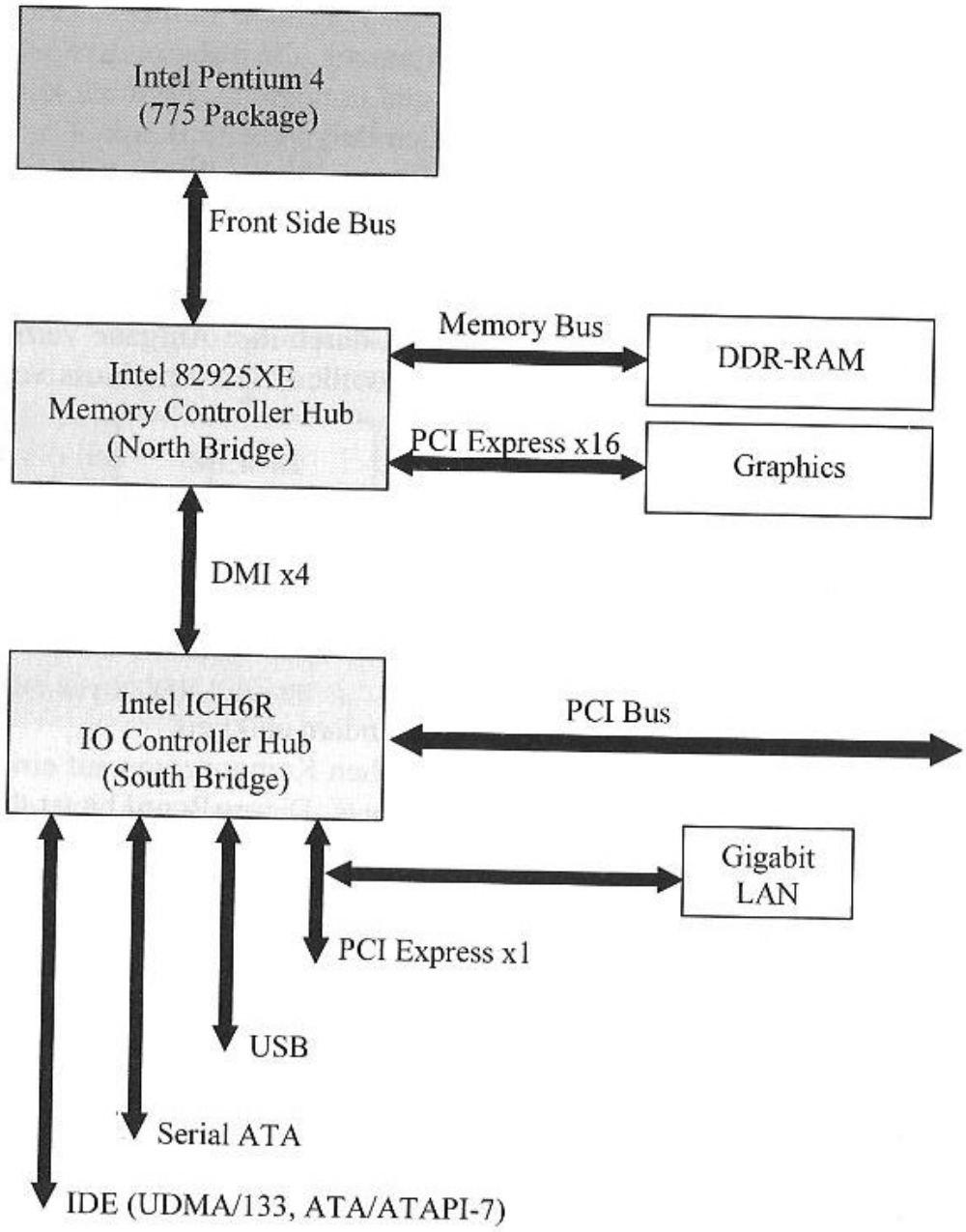


Abb. 1.2. Aufbau des Mainboards P5AD2-E von Asus

Hardware – Aufbau des Mainboards P5AD2-E von Asus

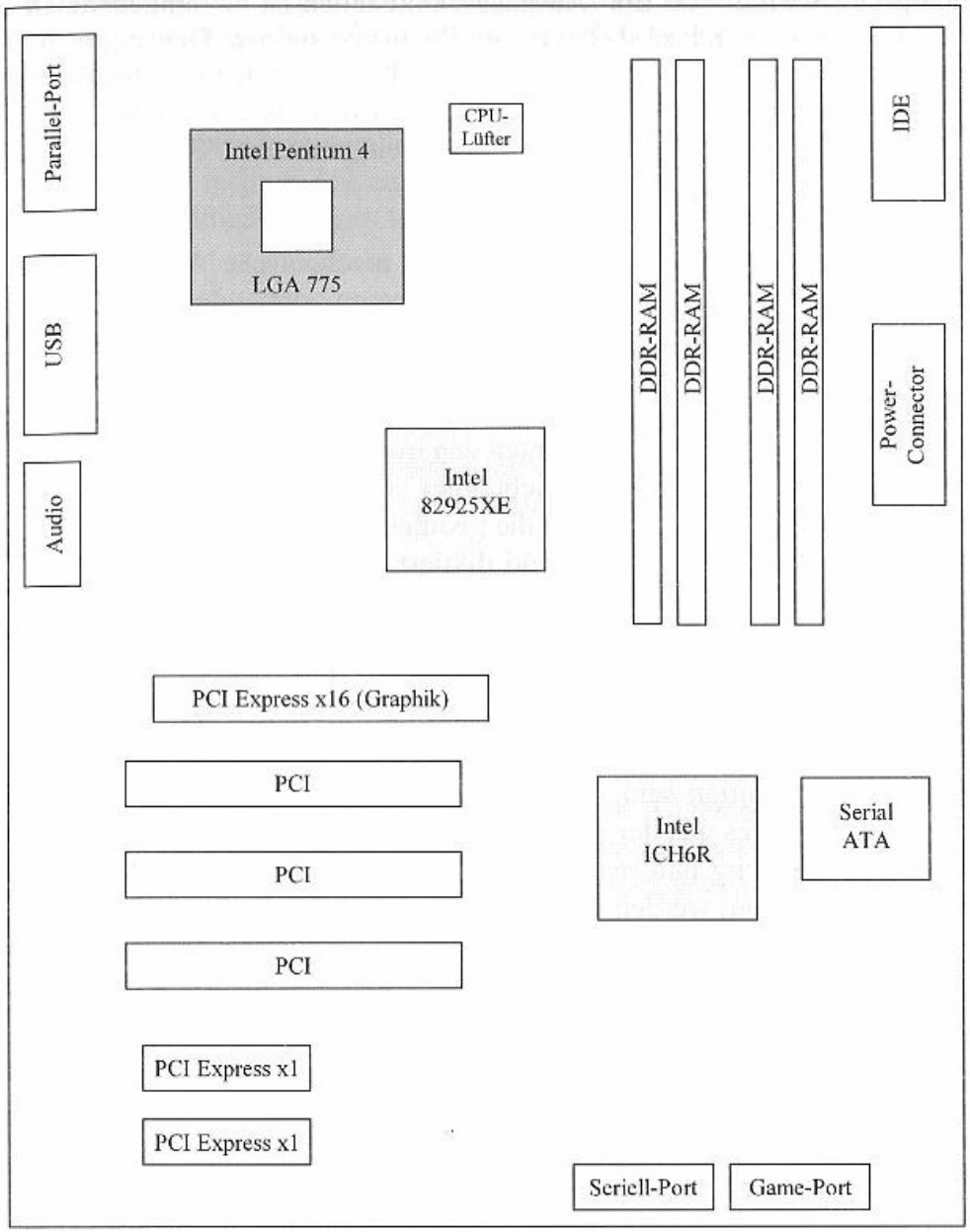
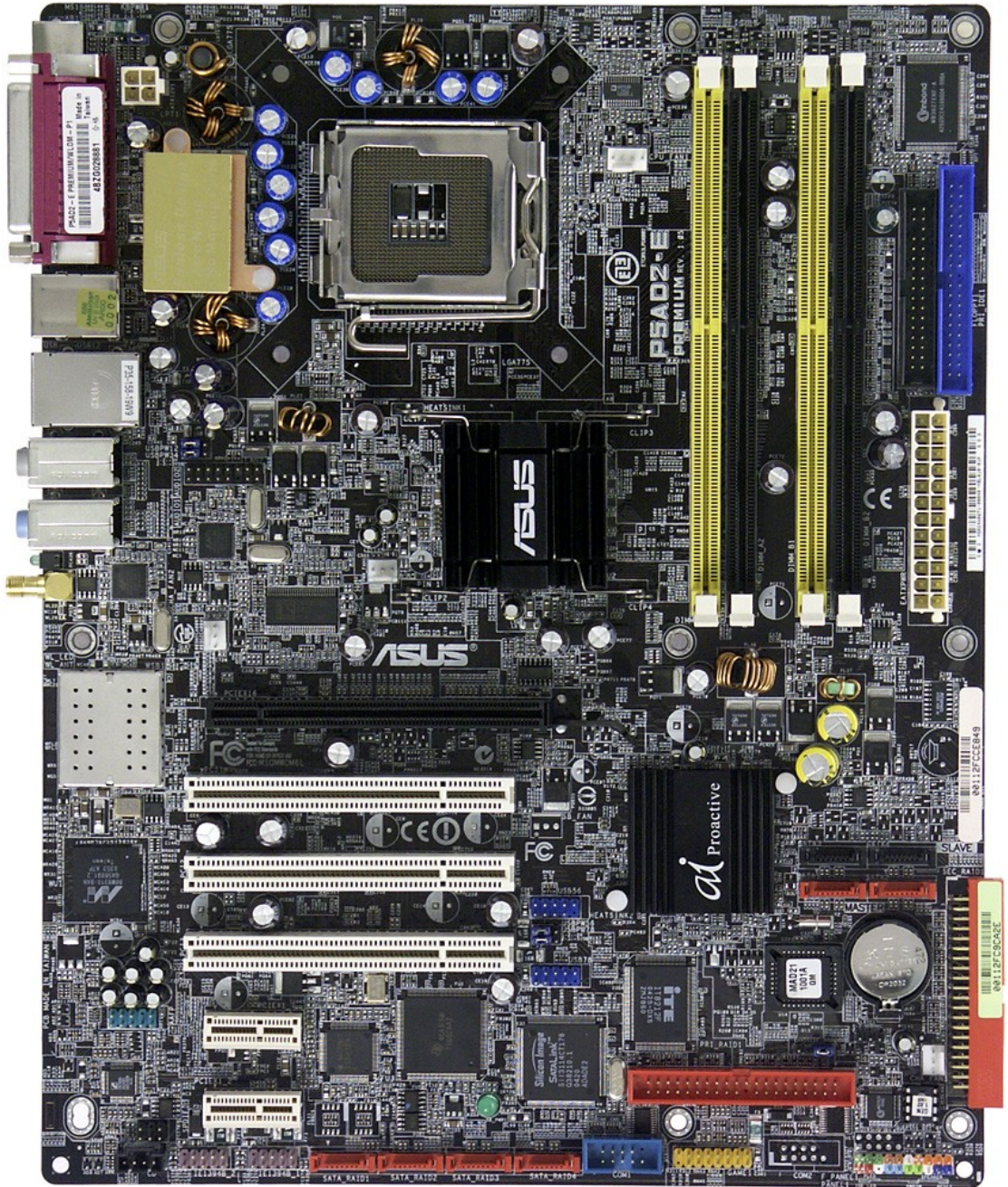


Abb. 1.3. Layout-Skizze des Motherboards P5AD2-E von Asus

Hardware – Foto des Mainboards P5AD2-E von Asus



Hardware - Mikroprozessor

Mikroprozessor ist die „CPU“ (**C**entral **P**rocessing **U**nit), die alle Systemkomponenten steuert und arithmethisch-logische Operationen ausführt.

- Befehlscodes für verschiedene Operationen im Speicher abgelegt („Programm“)
- Register als schnelle interne Datenspeicher
- Datentransfers Speicher <=> Register
- logische oder Arithmetische Operationen auf Registerinhalt oder externem Speicher

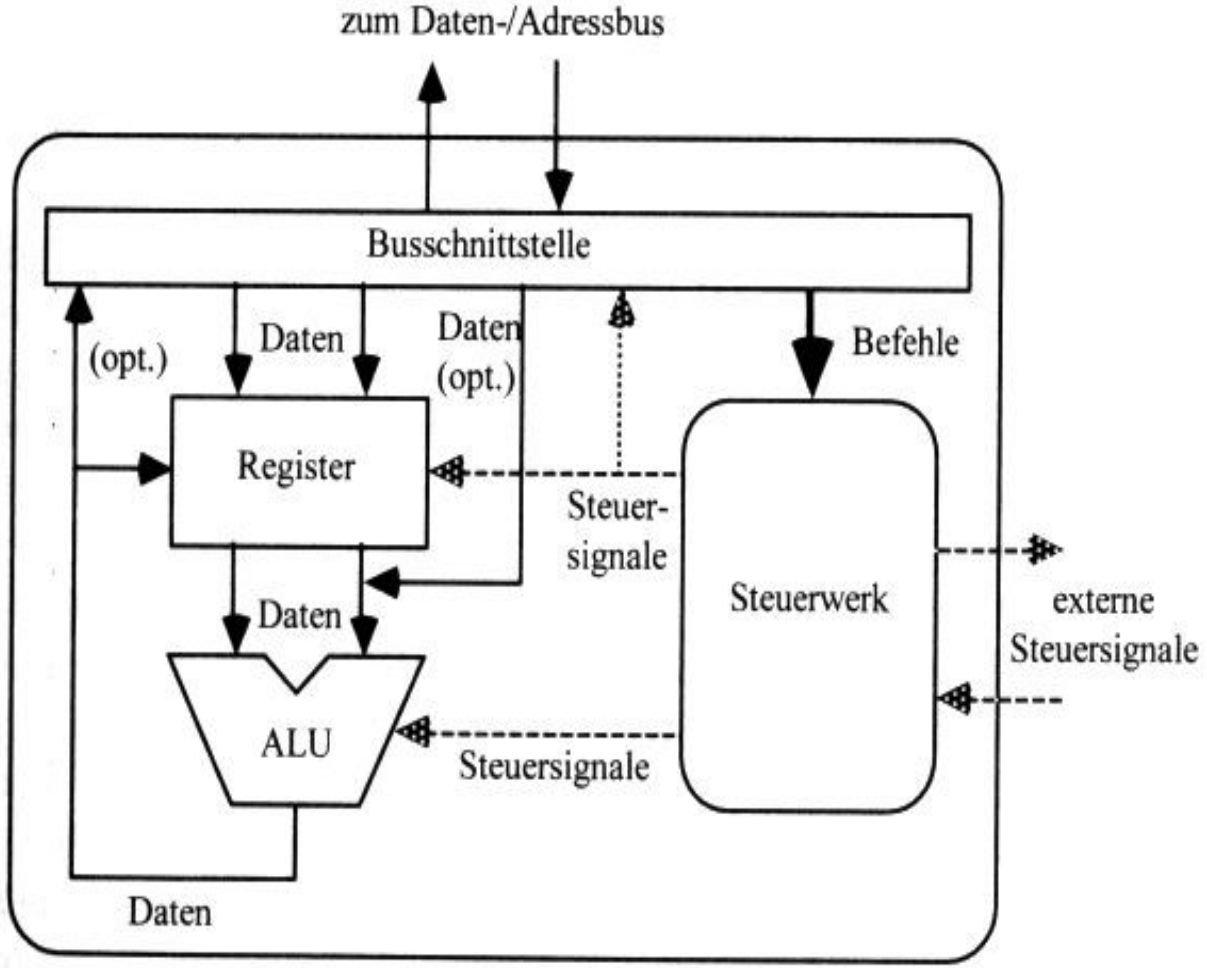


Abb. 2.17. Struktur eines einfachen Mikroprozessors

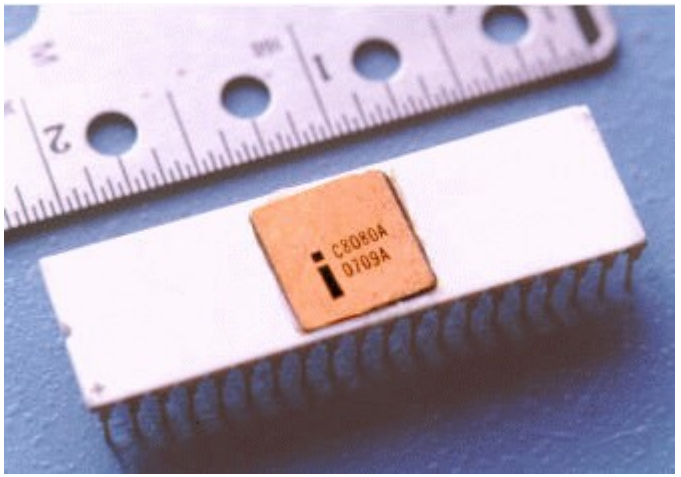
Hardware – Geschichtliches zu Mikroprozessoren (1)

1971
Der erste Ein-Chip Mikroprozessor
0.74 MHz, 4 bit
2300 Transistoren

Intel 4004

1974
Die klassische Intel-CPU
2-3 Mhz, 8 bit
4500 Transistoren

auch moderne CPUs beherrschen immer noch 8080 Code



Intel 8080

Hardware – Geschichtliches zu Mikroprozessoren (2)

1975
CPU vieler Heimcomputer
(Apple, Commodore ....)

MOS Technology 6502

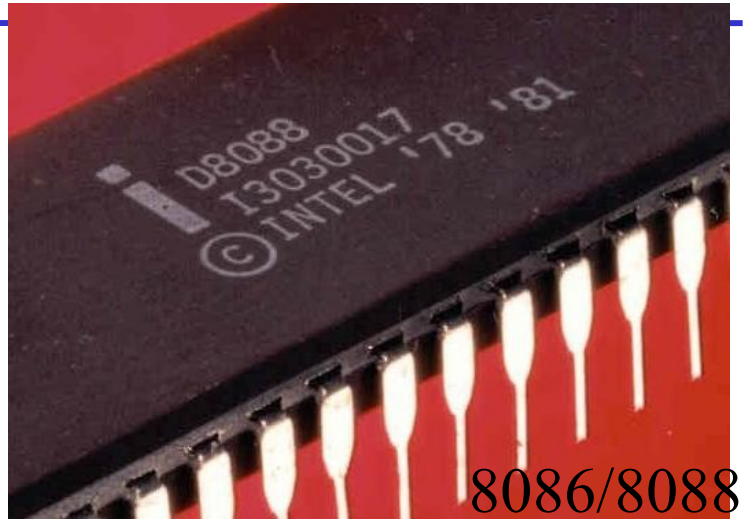
1976
Alle anderen Computer außer Apple und Commodore nutzten den Z80; wird heute noch produziert und verwendet.

führte block-move und block-copy ein
(frühe Variante von 3DNOW etc.)

Zilog Z80

Hardware – Geschichtliches zu Mikroprozessoren (3)

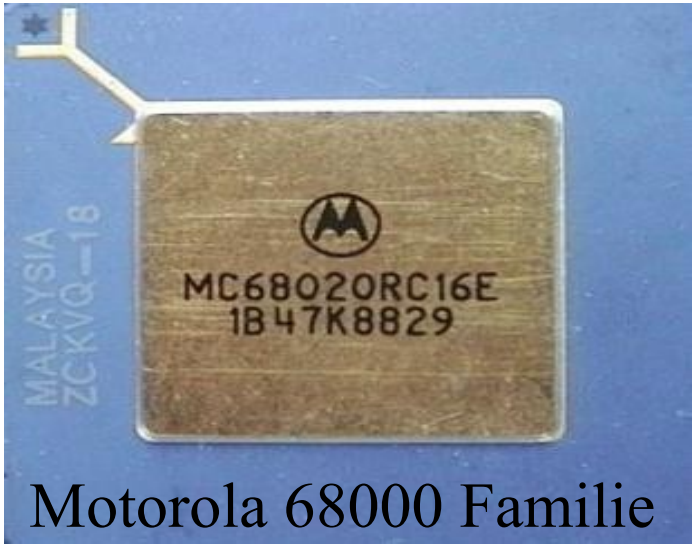
1978
CPU des ersten IBM PCs und Tausender
weiterer Modelle („Nachbauten“).
Dieses x86-Design ist Grundlage aller Nachfolger
(186, 286, 386, 486, Pentium)
ursprünglich 8 MHz, 16 Bit,
29.000 Transistoren



8086/8088

1979
Die „beste“ 16-bit-CPU, eingebaut in Apple
Lisa und MAC, Amiga, Atari ST ...)
8-25 MHz, 16-bit (seit 68020 32-bit),
68.000 Transistoren

eleganter und effizienter Befehlssatz

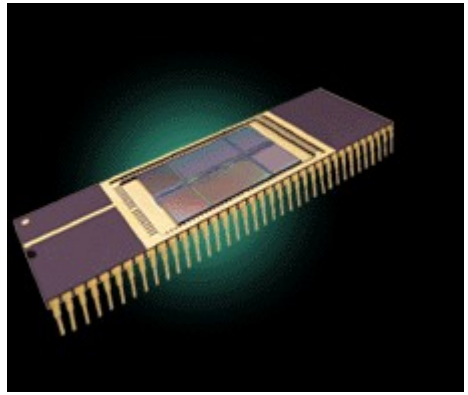


Motorola 68000 Familie

68000-Prozessoren werden noch gebaut
(man trifft sie in „embedded Systems“)

Hardware – Geschichtliches zu Mikroprozessoren (5)

1982
Weit verbreiteter Nachfolger des 8086
(mit 186 als Zwischenschritt), 16 bit, 12 u. 16. MHZ,
134.000 Transistoren
Konnte mehr als 1 Mbyte Speicher adressieren
(in einem speziellen Betriebsmodus, dem „protected mode“)



286

Erfolgreich als CPU aller IBM-kompatiblen PCs
der frühen neunziger Jahre, wurde 10 Jahre lang in Massenproduktion vermarktet

1985
Basis aller aktuellen PC-Prozessoren, i386-Architektur
Voraussetzung aller aktuellen Betriebssysteme,
32 bit, >2000 Mhz., 275.000 Transistoren

386

1989
486 integrierte FPU und Cache im Chip,
erweiterte Parallelität, Speichermanagement
1,2 Mio Transistoren

Hardware – Geschichtliches zu Mikroprozessoren (6)

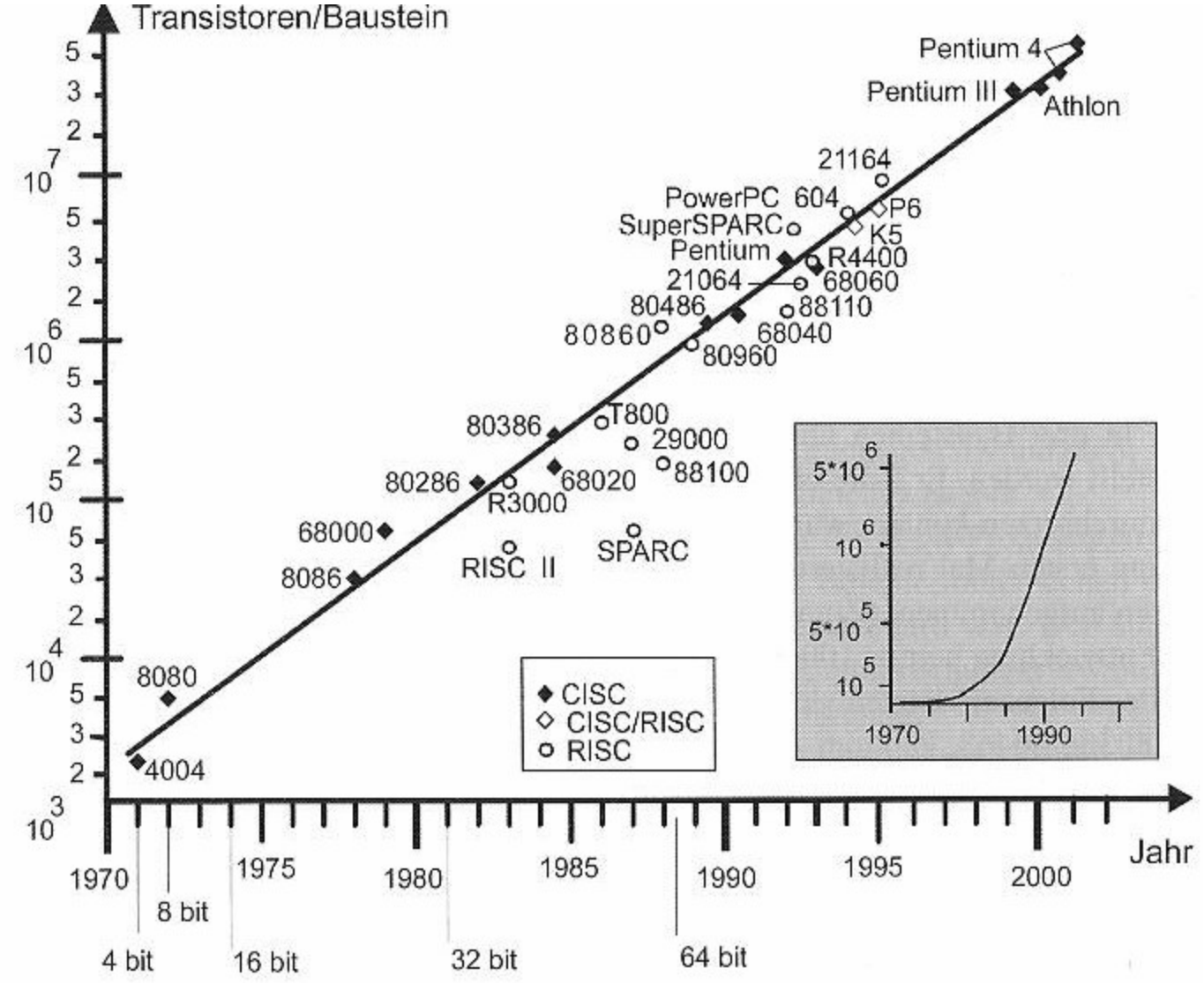
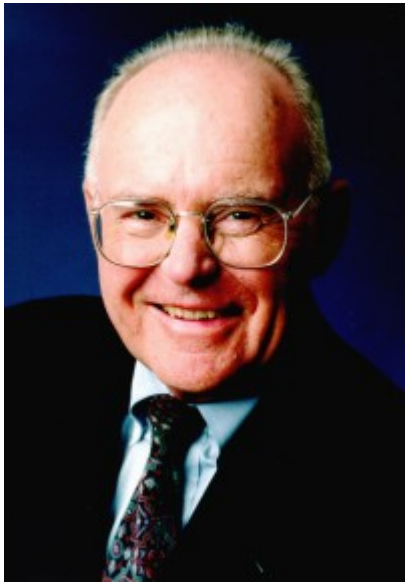


Bild 1.2-3: Die Entwicklung der Großintegration bei Mikroprozessoren



Moore'sches Gesetz: Verdopplung der Anzahl Transistoren alle 1½ Jahre

Hardware – Befehlssatz von Mikroprozessoren

Mikroprozessoren führen lediglich recht einfache Operationen aus;
die wichtigsten sind:

- Datentransfers Quelle => Ziel
- Bit-Manipulationen: Setzen, Löschen von Bits, Schiebe-Operationen
- logische Operationen (bitweise): AND, OR, XOR
- arithmetische Operationen: +,-,*,/ für Integer und Float-Typen
- Sprünge im Programm-Code
- Vergleich und ggf. Programm-Verzweigung

Aufbau eines Befehls:

OpCode [Adresse1 [Adresse2 [Adresse3]]]

Hardware - x86 Befehlssatz

Instruction	Meaning
AAA	ASCII adjust AL after addition
AAD	ASCII adjust AX before division
AAM	ASCII adjust AX after multiplication
AAS	ASCII adjust AL after subtraction
ADC	Add with carry
ADD	Add
AND	Logical AND
CALL	Call procedure
CBW	Convert byte to word
CLC	Clear carry flag
CLD	Clear direction flag
CLI	Clear interrupt flag
CMC	Complement carry flag
CMP	Compare operands
CMPSB	Compare bytes in memory
CMPSW	Compare words
CWD	Convert word to doubleword
DAA	Decimal adjust AL after addition
DAS	Decimal adjust AL after subtraction
DEC	Decrement by 1
DIV	Unsigned divide
ESC	Used with floating-point unit
HLT	Enter halt state
IDIV	Signed divide
IMUL	Signed multiply
IN	Input from port
INC	Increment by 1
INTO	Call to interupt
INTO	Call to interupt if overflow

Instruction	Meaning
IRET	Return from interrupt
Jxx	Jump if condition
JMP	Jump
LAHF	Load flags into AH register
LDS	Load pointer using DS
LEA	Load Effective Address
LES	Load ES with pointer
LOCK	Assert BUS LOCK# signal
LODSB	Load byte
LODSW	Load word
LOOP/LOOPx	Loop control
MOV	Move
MOVSB	Move byte from string to string
MOVSW	Move word from string to string
MUL	Unsigned multiply
NEG	Two's complement negation
NOP	No operation
NOT	Negate the operand, logical NOT
OR	Logical OR
OUT	Output to port
POP	Pop data from stack
POPF	Pop data into flags register
PUSH	Push data onto stack
PUSHF	Push flags onto stack
RCL	Rotate left (with carry)
RCR	Rotate right (with carry)
REPxx	Repeat CMPS/MOVS/SCAS/STOS
RET	Return from procedure
RETN	Return from near procedure

Instruction	Meaning
RETF	Return from far procedure
ROL	Rotate left
ROR	Rotate right
SAHF	Store AH into flags
SAL	Shift Arithmetically left (multiply)
SAR	Shift Arithmetically right (signed divide)
SBB	Subtraction with borrow
SCASB	Compare byte string
SCASW	Compare word string
SHL	Shift left (multiply)
SHR	Shift right (unsigned divide)
STC	Set carry flag
STD	Set direction flag
STI	Set interrupt flag
STOSB	Store byte in string
STOSW	Store word in string
SUB	Subtraction
TEST	Logical compare (AND)
WAIT	Wait until not busy
XCHG	Exchange data
XLAT	Table look-up translation
XOR	Exclusive OR

Hardware - Mikroprozessor

Ausführliches Beispiel:
der MOV-Befehl
zum Verschieben von
Daten

Adressierungsarten:
- Register
- unmittelbar
- absolut / direkt
- indirekt
- registerindirekt
  (mit Autoin-/de-krement)
- registerindirekt
  mit Verschiebung
- ...

MOV — Move Data — MOV

Syntax MOV op1, op2

Operation (op1) ← (op2)

Data Types WORD

Description Moves the contents of the source operand specified by op2 to the location specified by the destination operand op1. The contents of the moved data is examined, and the condition codes are updated accordingly.

Condition Flags

E	Z	V	C	N
*	*	-	-	*

E Set if the value of op2 represents the lowest possible negative number. Cleared otherwise. Used to signal the end of a table.

Z Set if the value of the source operand op2 equals zero. Cleared otherwise.

V Not affected.

C Not affected.

N Set if the most significant bit of the source operand op2 is set. Cleared otherwise.

Addressing Modes

Mnemonic		Format	Bytes
MOV	Rw_n, Rw_m	F0 nm	2
MOV	Rw_n, #data4	E0 #n	2
MOV	reg, #data16	E6 RR ## ##	4
MOV	Rw_n, [Rw_m]	A8 nm	2
MOV	Rw_n, [Rw_m+]	98 nm	2
MOV	[Rw_m], Rw_n	B8 nm	2
MOV	[-Rw_m], Rw_n	88 nm	2
MOV	[Rw_n], [Rw_m]	C8 nm	2
MOV	[Rw_n+], [Rw_m]	D8 nm	2
MOV	[Rw_n], [Rw_m+]	E8 nm	2
MOV	Rw_n, [Rw_m+#data16]	D4 nm ## ##	4
MOV	[Rw_m+#data16], Rw_n	C4 nm ## ##	4
MOV	[Rw_n], mem	84 0n MM MM	4
MOV	mem, [Rw_n]	94 0n MM MM	4
MOV	reg, mem	F2 RR MM MM	4
MOV	mem, reg	F6 RR MM MM	4

Hardware – Was ein Mikroprozessor tut ...

Mikroprozessoren verarbeiten intern Folgen von Bits

Bitfolge kann als
- Befehl
- Datum
- Adresse

interpretiert werden (kontextabhängig!)

Hochsprache

```
for(ii=0; ii<10; ii++) {
  a = ii; b = ii+1;
  if(a < 5) a += b;
  }
```

Assembler

```
                for (ii=0; ii<10; ii++) {
00000406 E00D   MOV     R13,#0x00
                a = ii;
00000408 F06D   MOV     R6,R13
                b = ii+1;
0000040A F04D   MOV     R4,R13
0000040C 0841   ADD     R4,#1
                if (a < 5) a += b;
0000040E 4865   CMP     R6,#5
00000410 9D01   JMPR    CC_NC,0x000414
00000412 0064   ADD     R6,R4
                }
00000414 809D   CMPI1   R13,#0x09
00000416 8DF8   JMPR    CC_NC,0x000408
```

Compiler

Bitfolge auf Maschinenebene

```
1110000000001101 1111000001101101
1111000001001101 0000100001000001
0100100001100101 1001110100000001
0000000001100100 1000000010011101
1000110111111000
```

Hardware – Superskalare Mikroprozessoren

Skalar: Ausführung eines Befehls pro Taktzyklus

Beschleunigung der Programmausführung durch:

- Pipelining
  (Problem: Daten-/ Ressourcen-/ Steuerflusskonflikte)
- Caches
- Sprungvorhersage
- Mehrere Prozessorkerne

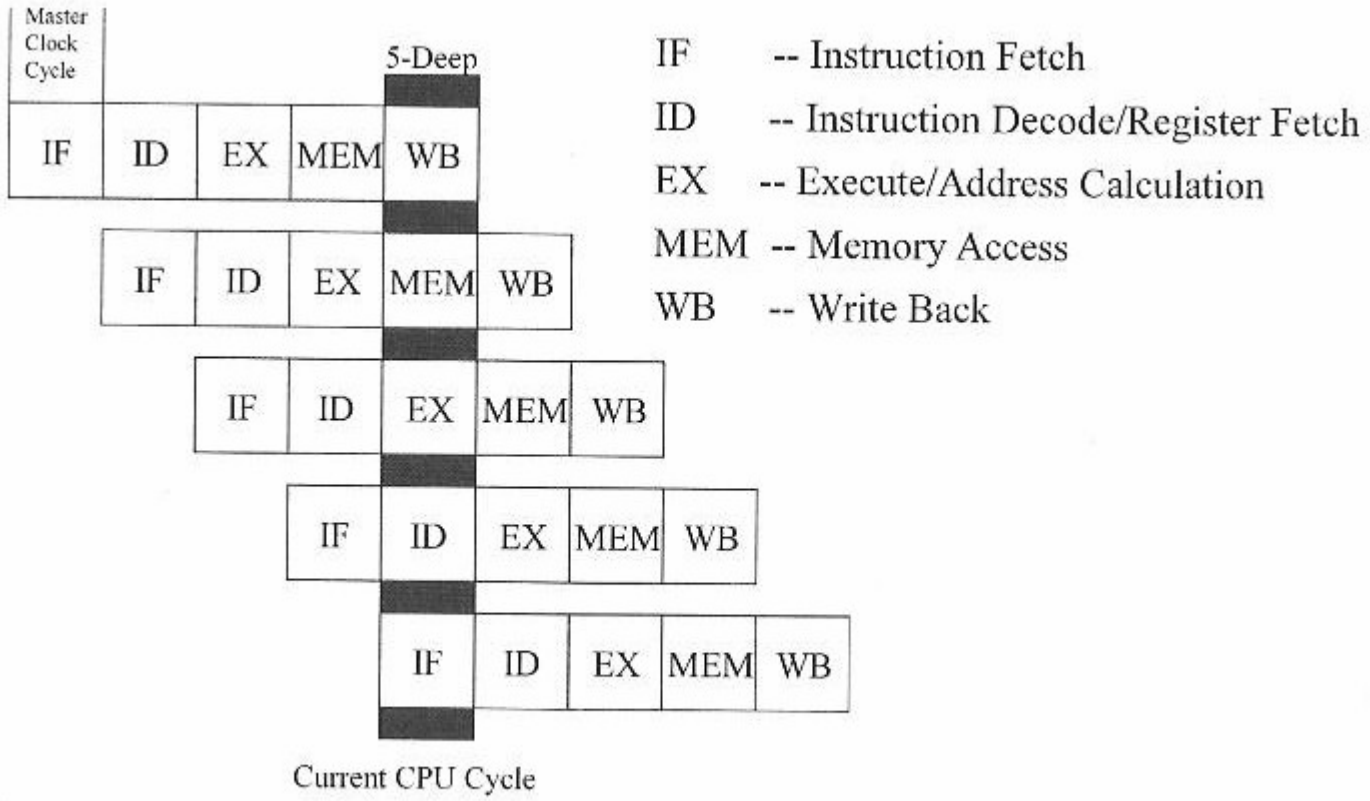


Abb. 2.20. Grundlegendes Pipelining

Hardware – Interrupts

Reaktion auf (interne oder externe) Ereignisse über Interrupts:

- Komponente meldet Wunsch zur Datenübertragung über spezielle Interruptleitung
- Ausnahmesituation bei Programmausführung (z.B. Division durch 0)
- Softwareinterrupt

Jeder Art von Interrupt ist eine Nummer zugeordnet

-> legt die Priorität fest

-> bestimmt die Adresse der Behandlungsroutine über die Interrupt-Vektortabelle

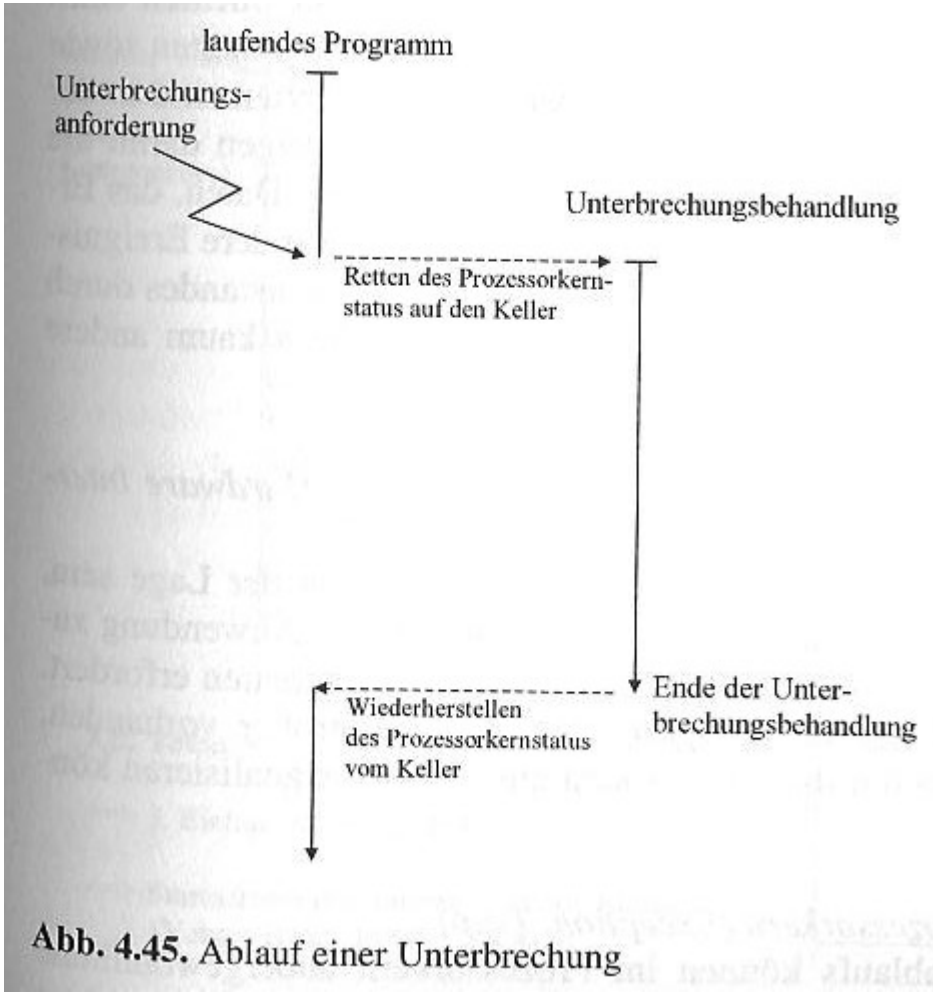


Abb. 4.45. Ablauf einer Unterbrechung

Hardware – Interrupt-Vektor

Tabelle 4.1. Beispiel einer festen Zuordnung eines Vektors zu einer Unterbrechungsquelle

Unterbrechungsquelle	Vektor	Priorität	Typ
Parallele Ein-/Ausgabe 1	0	nieder	interner Hardware-Interrupt
Parallele Ein-/Ausgabe 2	1		"
Serielle Ein-/Ausgabe	2		"
Analog/Digitalwandler 1	3		"
Analog/Digitalwandler 2	4		"
Analog/Digitalwandler 3	5		"
Zeitgeber 1	6		"
Zeitgeber 2	7		"
Capture & Compare	8		"
Externer Interrupt-Eingang 1	9		externer Hardware-Interrupt
Externer Interrupt-Eingang 2	10		"
Externer Interrupt-Eingang 3	11		"
...			
Break	200		Software-Interrupt
...			
Unbekannter Befehlscode	253		Exception
Division durch 0	254		"
Reset	255	hoch	"

Hardware – Bus-System

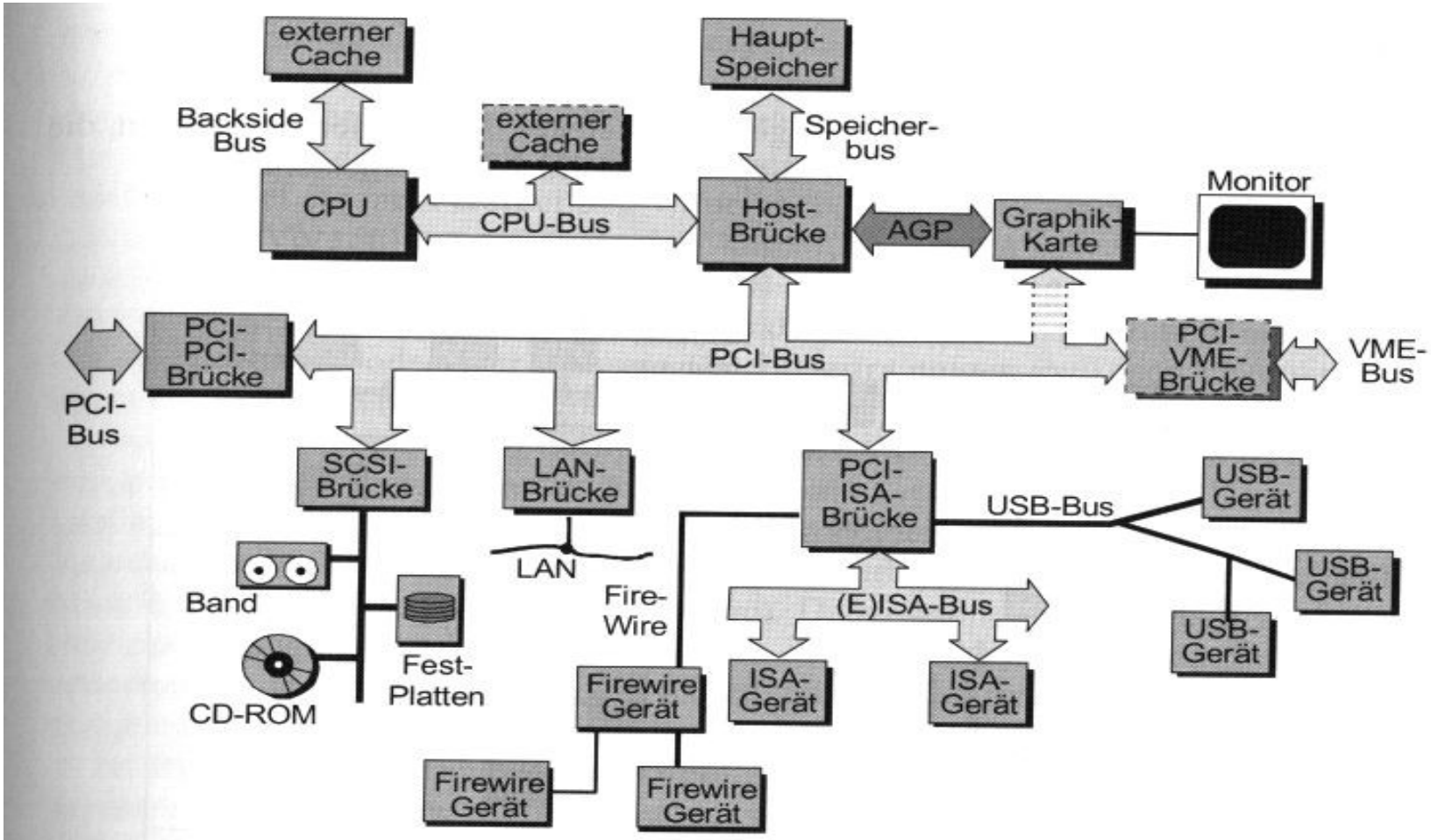
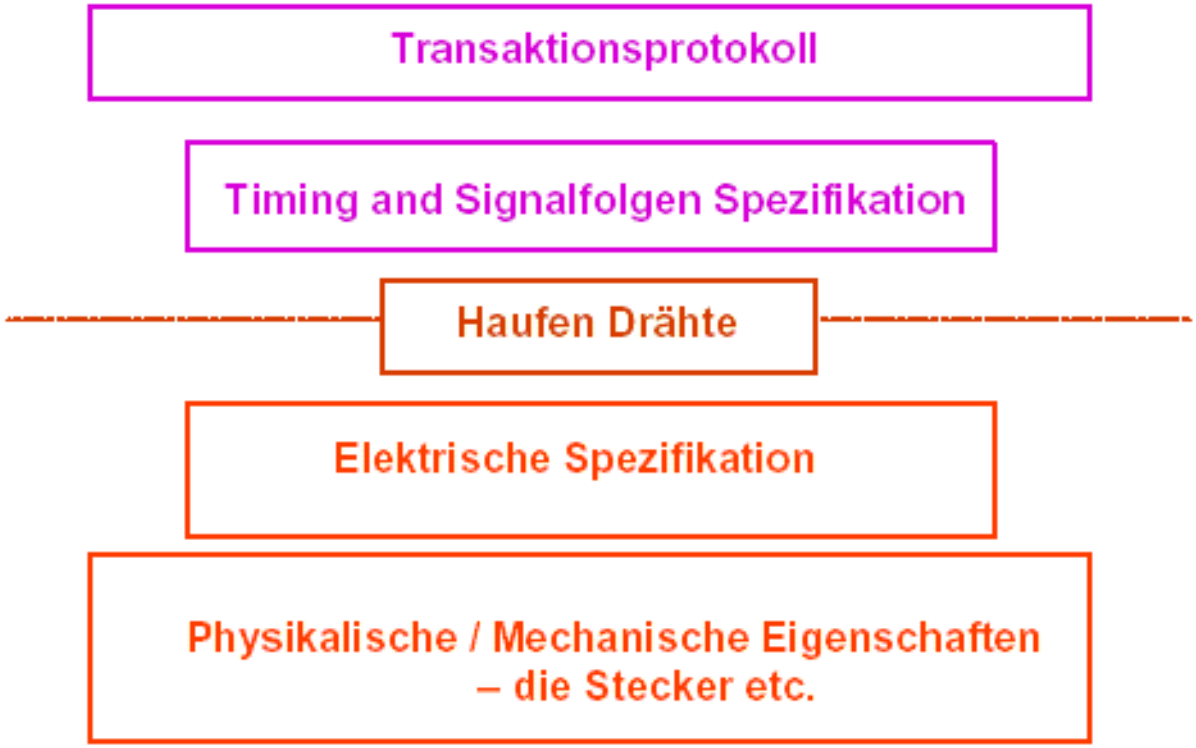


Bild 1.2-2: Typisches hierarchisches Bussystem

Busteilnehmer: Master/Slave; Arbiter

Hardware - Bussysteme

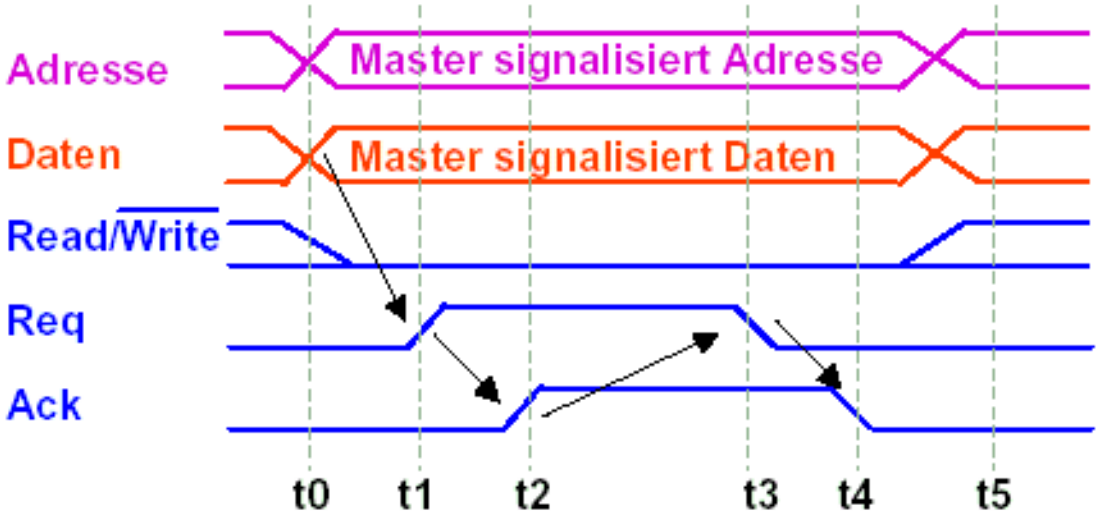
Die wichtigsten Bestanddteile eines Busses



Daten- und Signaltypen auf Bussen:

(Multiplexing: mehrere Typen auf einem Bus)

Einfachstes Protokoll für asynchrone Datenübertragung (Schreibzyklus):



- t0 : Master hat Buskontrolle und signalisiert Adresse, Richtung, Daten, wartet eine spezifizierte Zeit bis Daten am Bus stabil getrieben
- t1: Master signalisiert Anforderung Signal (Req)
- t2: Slave signalisiert Antwort (Ack) später, wodurch der Datenempfang bestätigt wird
- t3: Master löscht Anforderungssignal (Req)
- t4: Slave löscht Antwortsignal (Ack)

Busankopplung über Tristate-Gatter; Zustände:
L (low)
H (high)
Z (hochohmig)

Echte Busse sind viel komplizierter !

Hardware - Speicher

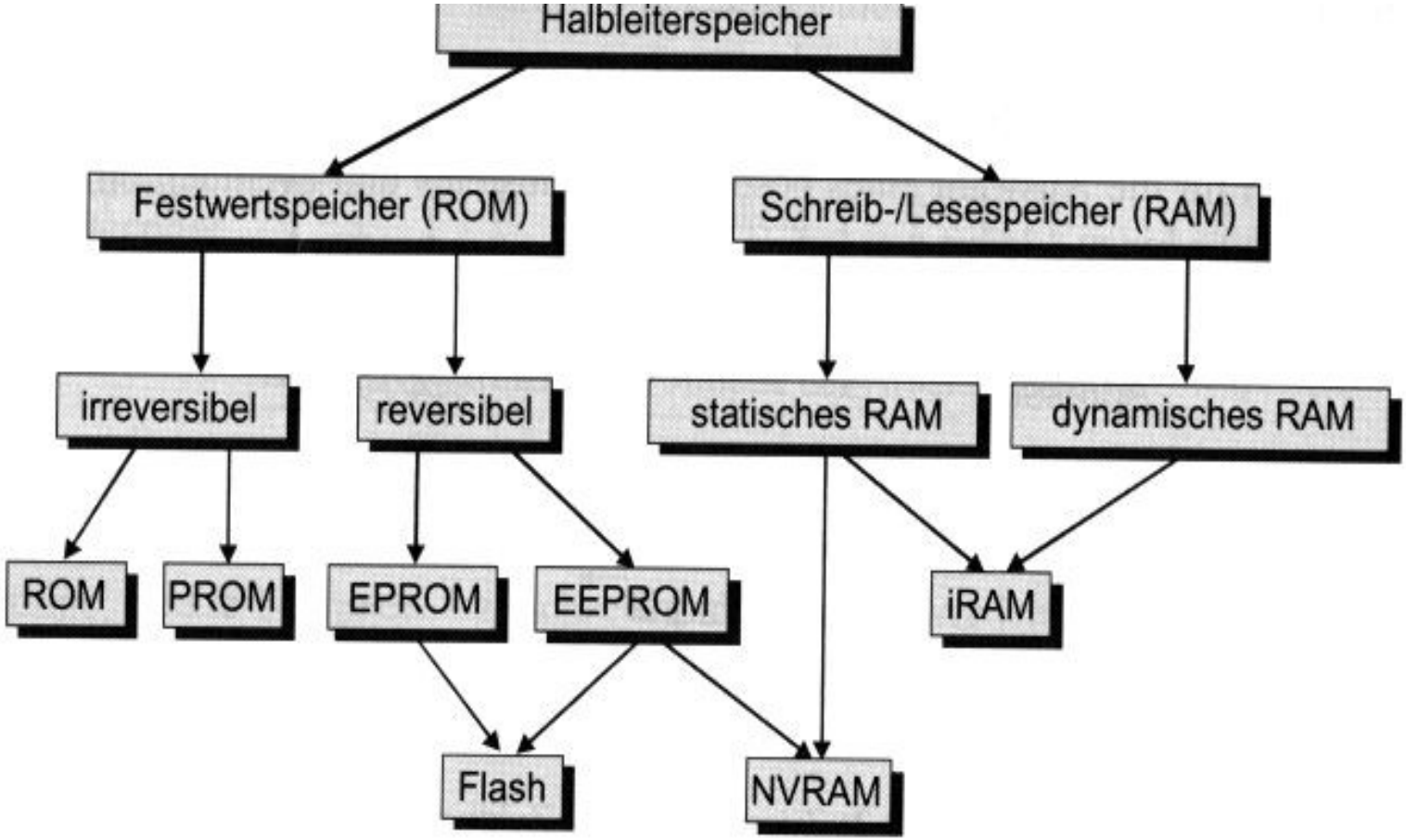
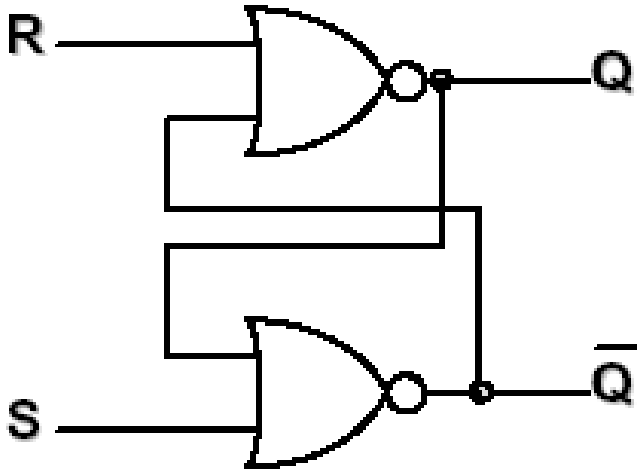
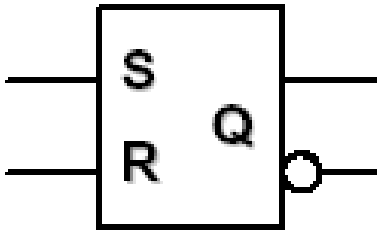


Bild 2.1-3: Die wichtigsten Typen der Halbleiterspeicher

Hardware – Statischer Speicher (SRAM)

Besteht aus einzelnen Flip-Flops (wie früher schon gezeigt)

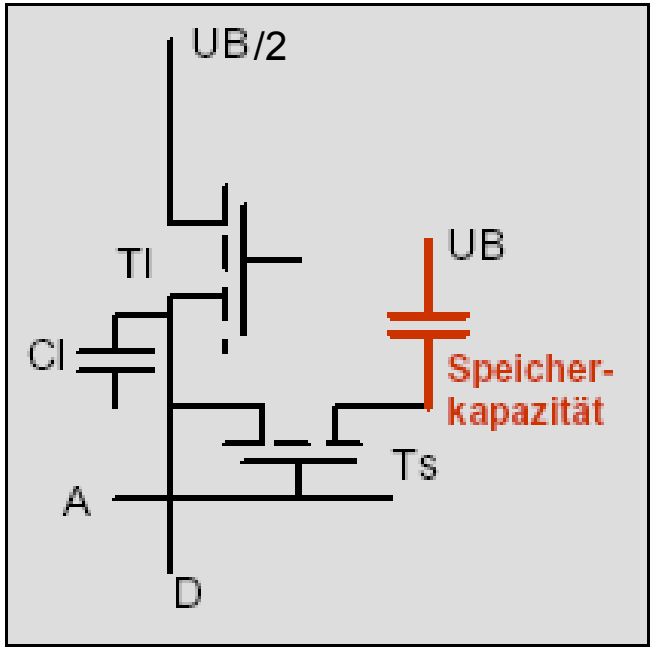




- solche Speicher sind schnell
- in CMOS-Technologie aufgebaute Zellen brauchen recht wenig Strom
- man braucht 8 Transistoren pro Bit, sehr aufwändig
  (Chipfläche und Struktur)

Hardware – Dynamischer Speicher (DRAM)

Datenspeicherung als Ladung auf Kondensator



Speicherkapazität 0,05-0.5 pF;
(~100000 Elektronen)

Lesen

- Datenleitung über Transistor TI vorladen („precharge“)
- Ladungsaustausch mit Speicherkapazität über Transistor TS
- Leseverstärker detektiert, ob Ladung gepeichert war
- Zersörendes Lesen! Zelle muss mit ausgelesenem Datum neu beschreiben werden

Schreiben

- Ts aktivieren
  - Massepotential auf Datenleitung => Kondensator lädt sich auf
  - positives Potential auf Datenleitung => Kondensator entlädt sich

Refresh

- Leckströme entladen Kondensatoren langsam => Information auf Speicherzellen muss periodisch erneuert werden (10-100 ms)
- Refresch im Speicherchip oder durch Speichercontroller

Hardware – Schnittstellen

Datenaustausch zwischen
- zwischen Rechner und Peripheriegeräten bzw.
- zwischen mehreren Rechnern
erfolgt über Schnittstellen.

Schnittstellen müssen standardisiert sein:

- elektrische Signale
- Steckermechanik und Belegung
- Timing
- Übertragungsprotokoll

Hardware – die wichtigsten Schnittstellen

Klassisch:

- die serielle Schnittstelle (RS232 oder V.24 Standard)

  (früher) gebräuchlich für Maus, Tastatur, Modem, div. Meßgeräte

- die parallele Schnittstelle

- USB („universal serial bus“)

  die aktuelle Schnittstelle (Maus, Tastatur,PC-Erweiterungen, Meßgeräte)
  es existieren Adaper seriell <=>USB oder Parallel<=>USB

- FireWire

  besonders im Multi-Media-Bereich (DigiCams, aber auch Festplatten etc.)

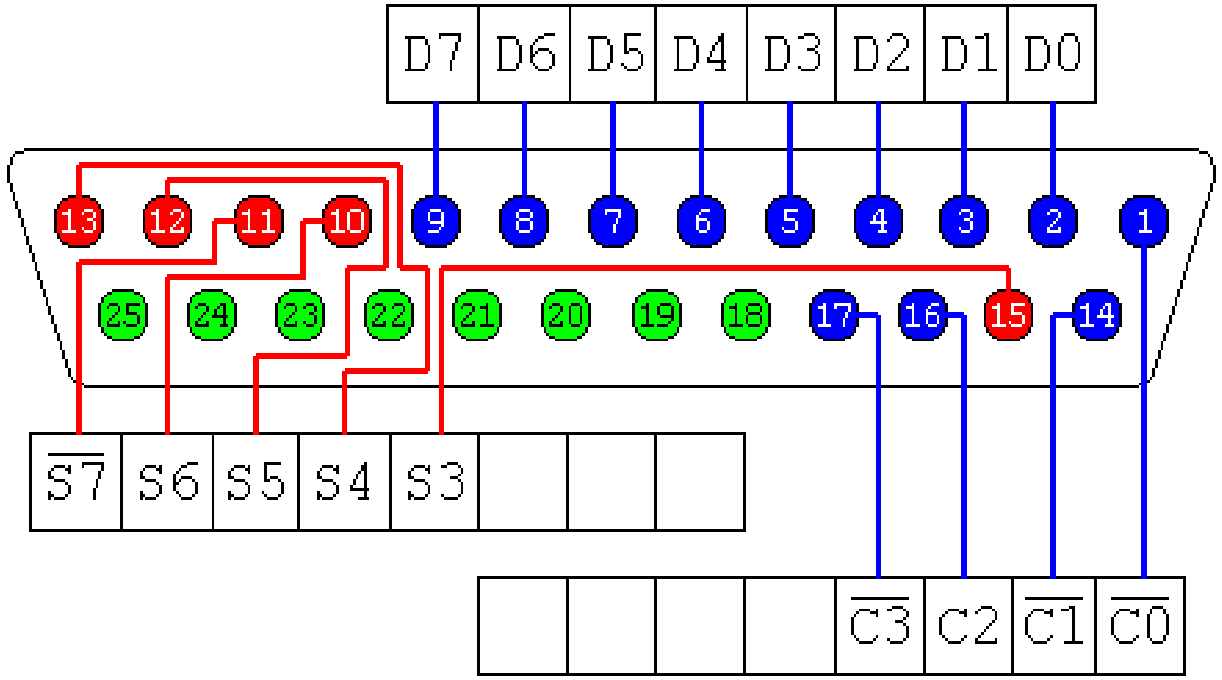
- Zugriff auf PC-Bus mittels Steckkarten (ISA (veraltet) oder PCI)

  selbst PCI-Karten sind heutzutage nur wenig schneller als USB 2

- Zugang über Netzwerk (Ethernet-Schnittstelle, 10/100/1000 Mbit/sec)

Hardware – der Parallel-Port

Mehrere Leitungen gleichzeitig zur Datenübertragung genutzt.



8 Datenleitungen (output)

5 input Eingänge (Status)

4 outputs (Control)

8 Pins geerdet

Anwendung: Drucker, manche Meßgeräte
Mittleweile weitgehend durch USB ersetzt

Nachteil der parallelen Übertragung:

alle Datenleitungen müssen völlig synchron übertragen; bei hohen Datenraten wegen unterschiedlicher Kabellaufzeiten und Störeinflüssen extrem schwierig.

-> Begrenzung der Kabellänge

Außerdem höhere Kosten von Steckern und Kabeln

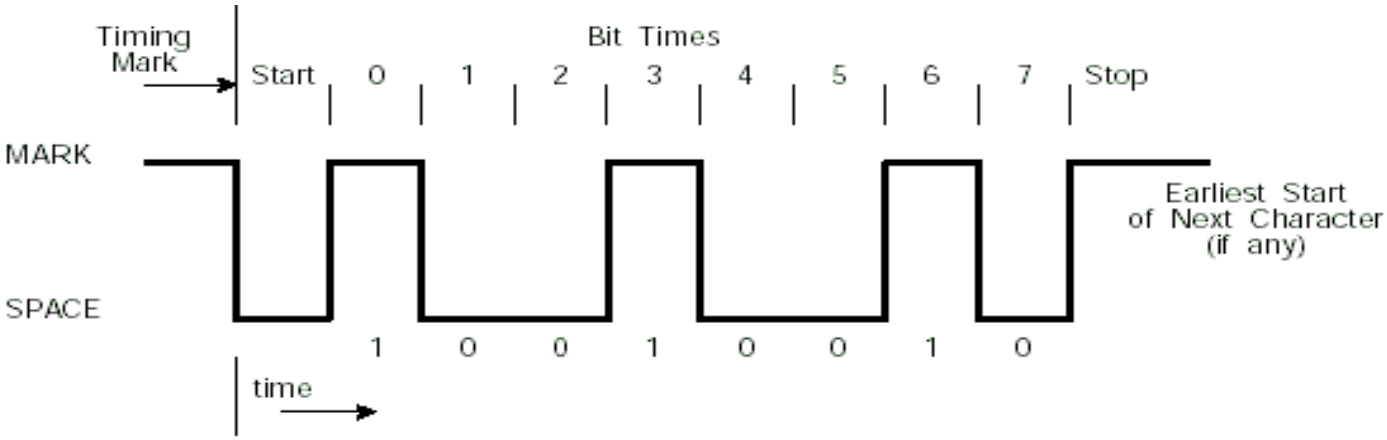
Hardware – die serielle Datenübertragung

Datenübertragung auf einer Leitung je Richtung.
Evtl. weitere Leitungen für „handshake“ zwischen Sender und Empfänger

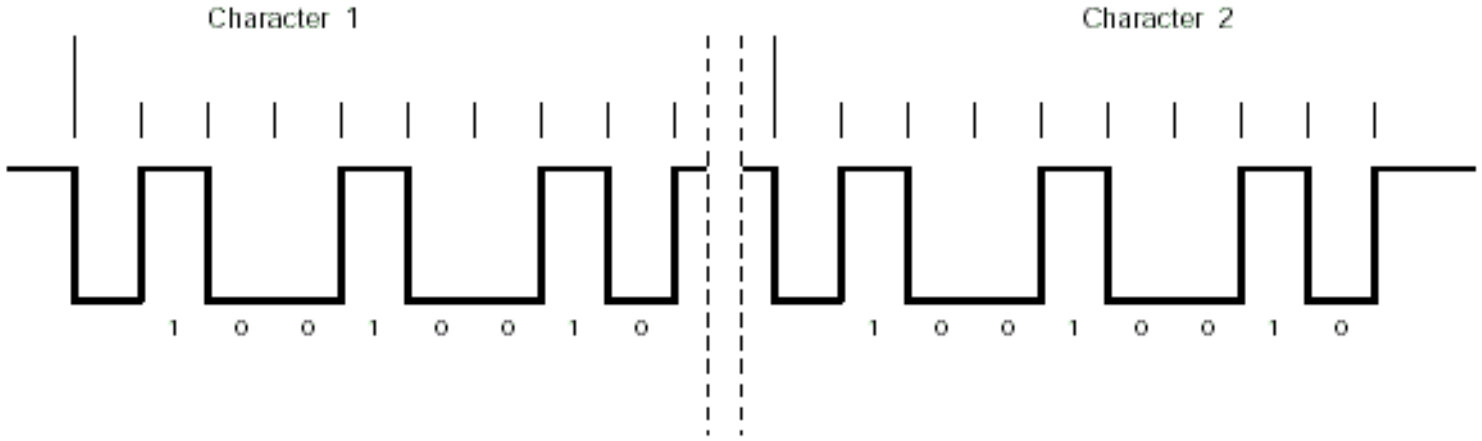
Einzelne Bits werden nacheinander übertragen.
- Erkennung von Übertragungsfehlern durch Paritätsbits (gerade/ungerade Anzahl Einsen) bzw. Error Detection Codes (EDC)
- Evtl. sogar Korrektur von Übertragungsfehlern durch Error Correction Codes (ECC);
  Nachteil: geringerer Nutzdatenanteil

Übertragungsarten:
- asynchron
- synchron mit gemeinsamer Clock
- synchron mit „embedded clock“

Hardware – asynchrone serielle Übertragung

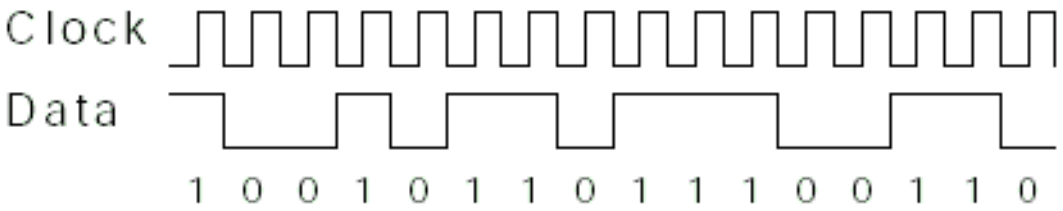


An Asynchronous Character Transmission

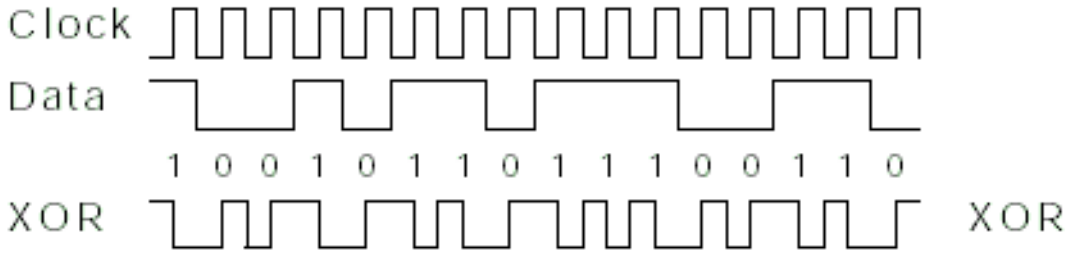


Successive Characters on an Asynchronous Line

Hardware – synchrone serielle Übertragung



Separate Clock Synchronous Data



Embedded Clock Data Transmission

Clock-Signal wird vom Sender mit übertragen, entweder auf einer extra Leitung, oder durch Exclusiv-Oder der Daten mit der Clock erfordert (bei diesem Verfahren) doppelte Taktrate !

Hardware – RS232 serielle Schnittstelle

Elektrische Spezifikation des RS232-Standards:

Outputs: - open-circuit voltage in the range ±25V
- output voltage 5 to 15V for SPACE, -5 to -15V for MARK for a resistive load in the range 3000-7000 Ω.
- shall survive a short-circuit between any two pins (including two outputs) and shall pass a current of <0.5A under those conditions
- shall have a rate of change of output of less than 30V/µsec

Inputs: - shall have an impedance in the range 3000-7000 Ω
- shall respond to voltages >3V as a SPACE and <-3V as a MARK.
- shall survive input voltages in range ±25V

Geschwindigkeiten: 1200, 2400, 4800,9600, 14400,28800,57600,115200 bit/sec

Hardware – RS232-Schnittstelle

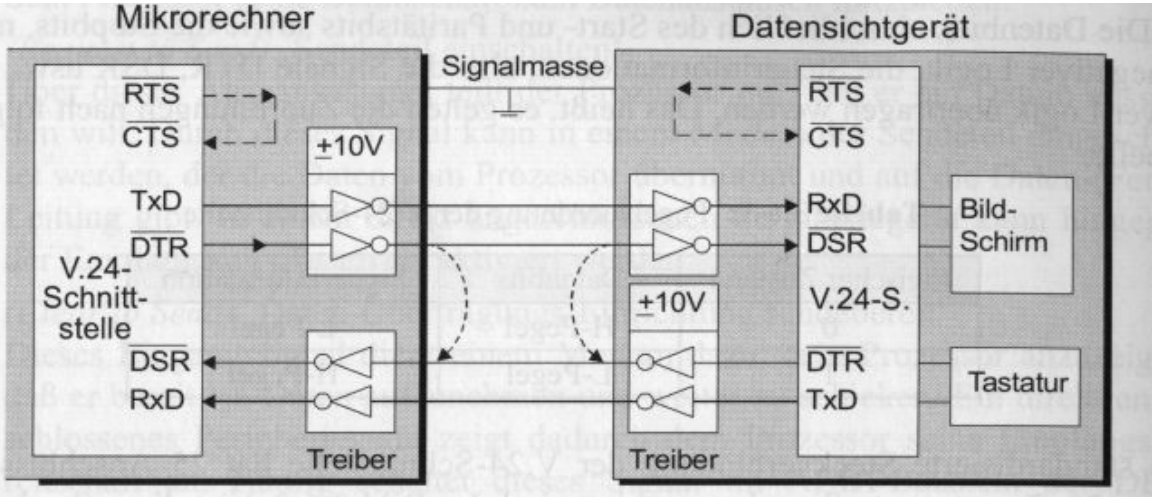


Bild 3.6-4: Anschluß eines Datensichtgeräts über eine V.24-Schnittstelle

Je eine Datenleitung pro Richtung:

- TxD – Transmit Data
- RxD – Receive Data

Steuerleitungen:

- DTR – Data Terminal Ready
- DSR – Data Set Ready
- RTS – Ready to Send
- CTS – Clear to Send

Hardware-Synchronisation über RTS/CTS-Protokoll

Software-Synchronisation über XON/XOFF-Protokoll (Ctrl-S / Ctrl-W)

Hardware – Serielle Schnittstelle

Hardware-Protokolle in speziellen Chips implementiert (Klassiker: UART, ACIA)
mittlerweile im Chip-Satz der MainBoards integriert.

Chips haben Register, die programmiertechisch
wie Speicherstellen ansprechbar sind (sog. „Ports“);

Setzen und Abfragen der Statusregister
und Lesen/Schreiben der Datenregister
per Software mit standardisierten Schnittstellen gesteuert (die sog. „Treiber“).

Aufruf der Treiber durch Interruptanforderung an die CPU
=> „Hardwareunterbrechung“ des laufenden Programms und
Ausführen des entsprechenden Treibers, falls Daten vorliegen.

Betriebssystem (BIOS) oder höhere Programmiersprachen liefern
Funktionen zum Lesen und Schreiben der Standard-Schnittstellen eines PC

Der Universal Serial BUS (USB)

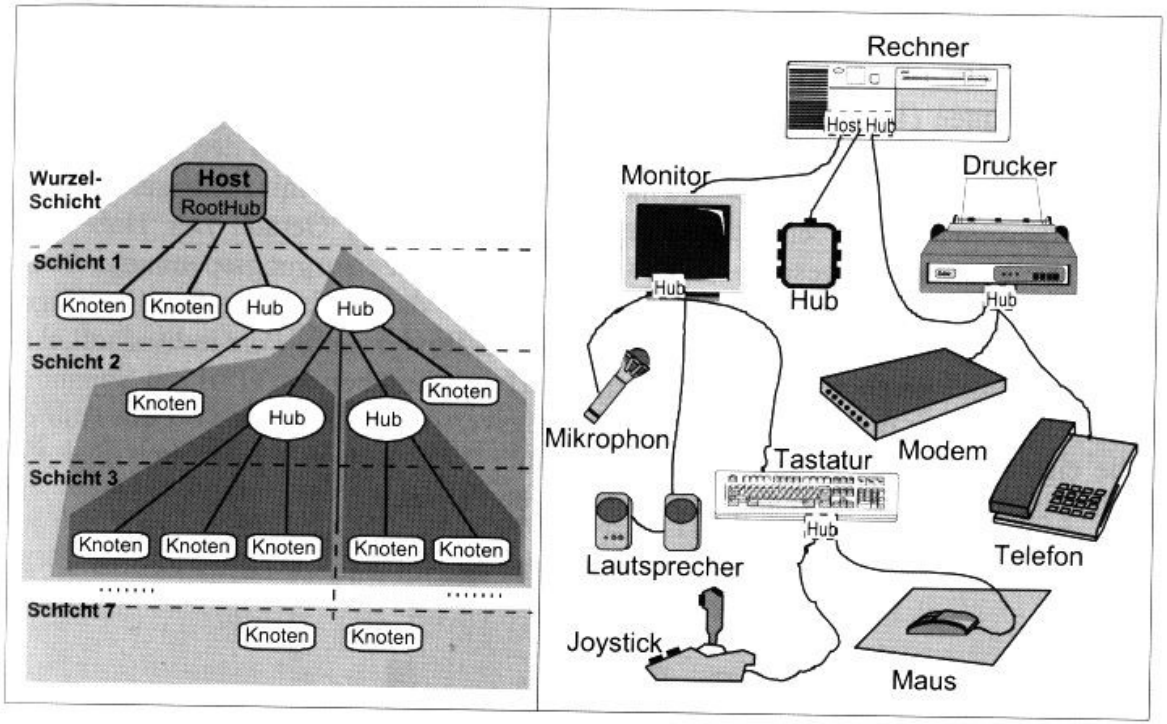


Bild 1.7-2: USB-Topologie und Systembeispiel

Preiswerte Stecker und Kabel,
flexibel konfigurierbar,
„plug-and-play“

Geschwindigkeiten:

USB 1: 1,5 und 12 Mbit/sec

USB 2: 480 Mbit/sec,
kompatibel zu USB1

USB 3: 4000 Mbit/sec

Hardware – FireWire (IEEE 1394)

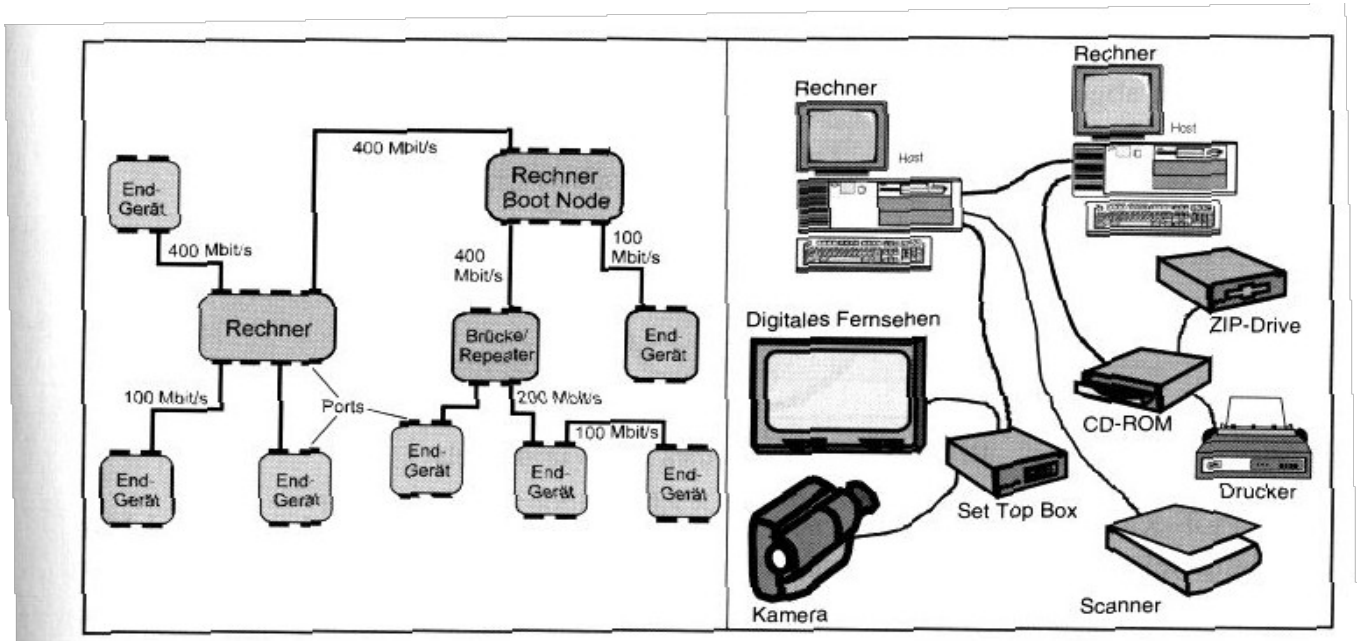


Bild 1.8-1: Struktur eines FireWire-Systems

Konkurrenz zu USB, hauptsächlich verwendet im Video-Bereich, aber z.B. auch bei Massenspeichern.

Geschwindigkeiten bis 400 Mbit/sec,
Nachfolger IEEE 1394-2008 bis 3200 Mbit/sec

Hardware – SCSI (Small Computer System Interface)

Hauptsächlich zur Anbindung von Massenspeichern (intern und extern!)
- findet Verwendung für hochwertige Platten, Bandspeicher, ...
- SCSI-Protokoll Grundlage für z.B. Brenner, USB-Speicher etc.;
  viele Linux-Treiber verwenden ein (virtuelles) SCSI-Laufwerk

SCSI schon lange im Gebrauch, wird ständig weiter entwickelt.

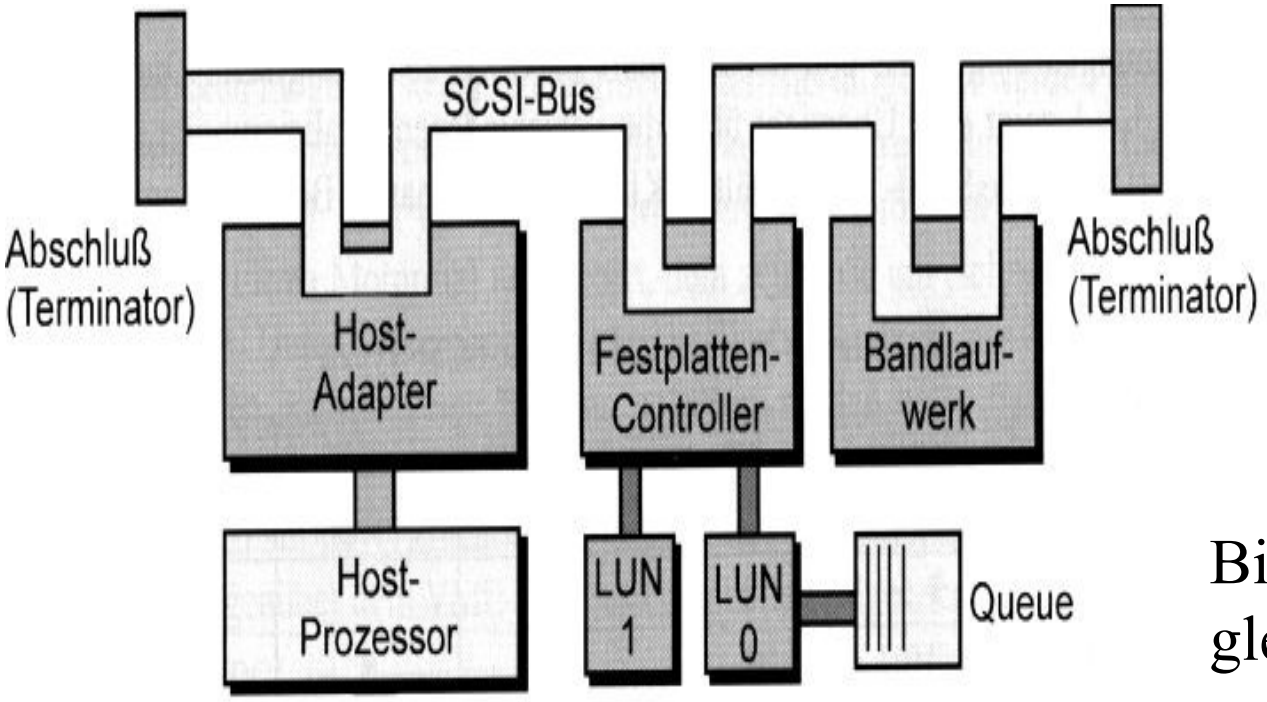


Bild 1.6-1: Aufbau eines SCSI-Systems

Bis zu 7 Geräte am gleichen Bus möglich

Hardware - CAN – Bus (Controller Area Network)

Robustes serielles Bussystem mit differentieller, recht langsamer Datenübertragung zur Verbindung von unabhängigen Controllern (analog zu LAN – Local Area Network).

Anwendung in Steuerung u. Regelung, „Slow Control“ von Experimenten, kommt aus dem Automobilbau.

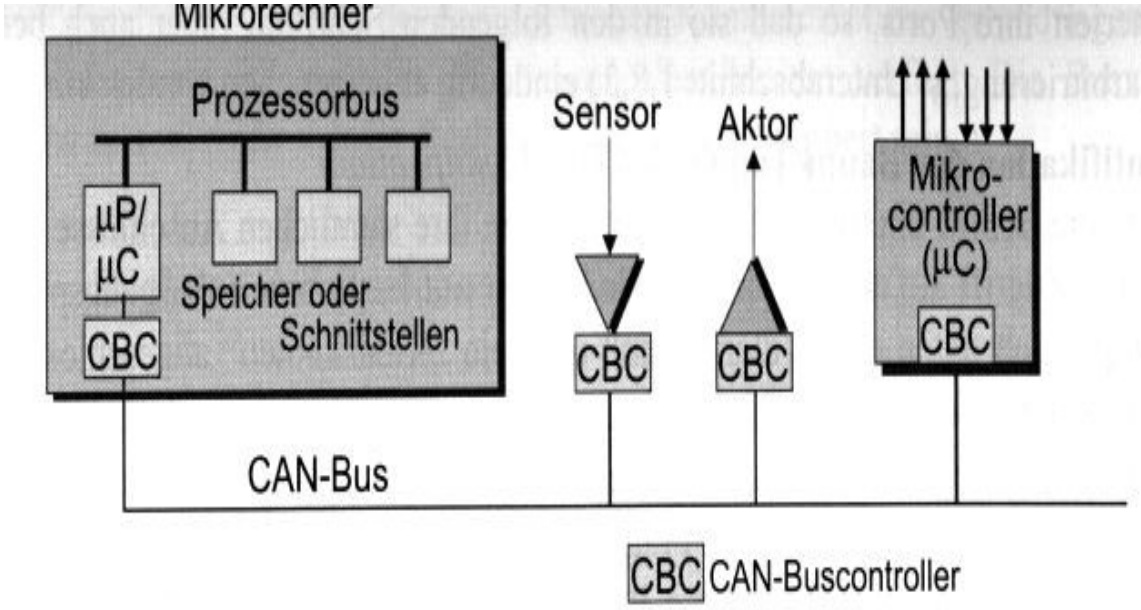
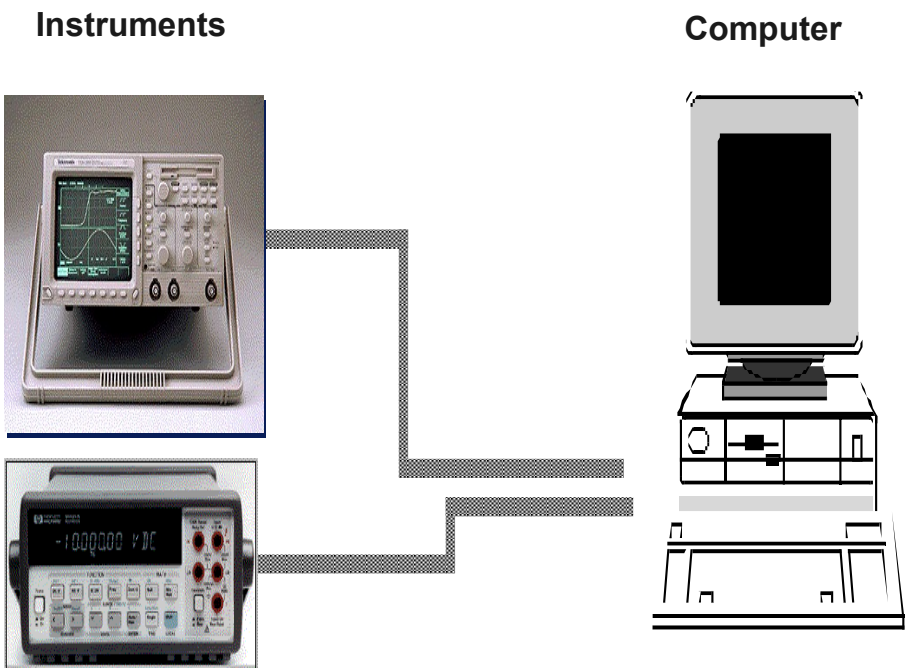


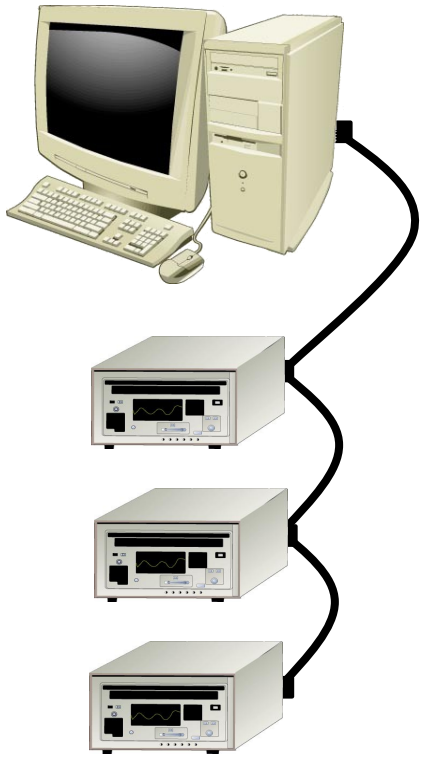
Bild 1.9-1: Struktur eines CAN-Systems

Hardware – GPIB General Purpose Interface Bus (IEEE 488)

8-Bit paralleler Bus mit ≥1Mbyte/sec Übertragungsrate;
ein Systemcontroller (meist PCI-Interface zu Computer)
mit bis zu 14 Instrumenten



Stern-Konfiguration



Lineare Konfiguration

Hardware – Ethernet (IEEE 802.3)

Ethernet ist **DER** Netzwerk-Standard im LAN (=Local Area Network)

Serielle Übertragung mit „embedded clock“

Protokoll: **c**arrier **s**ense **m**ultiple **a**ccess **c**ollision **d**etect (CSMA/CD)

- 10 Mbps (10Base-T Ethernet) (Koaxial-Kabel, „twisted pair“)
- 100 Mbps (Fast Ethernet) (twisted pair)
- 1000 Mbps (Gigabit Ethernet) (twisted pair, optisch)

10-Gigabit Ethernet (optisch)

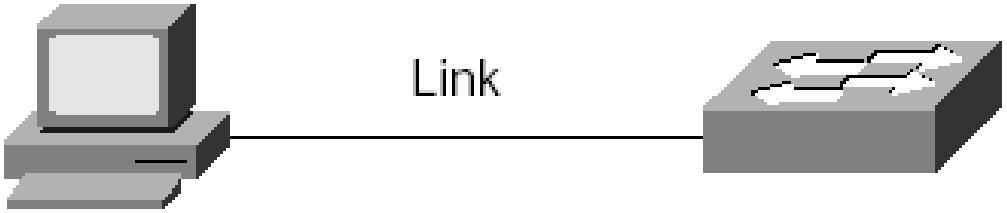
Hardware-Schnittstelle am PCI-Bus (Einsteckkarte oder auf MB integriert)

Verbindung von Rechnern, und – zunehmend - auch (intelligenter) Peripherie über Hubs, Switches, Router (verbinden Sub-Netze)

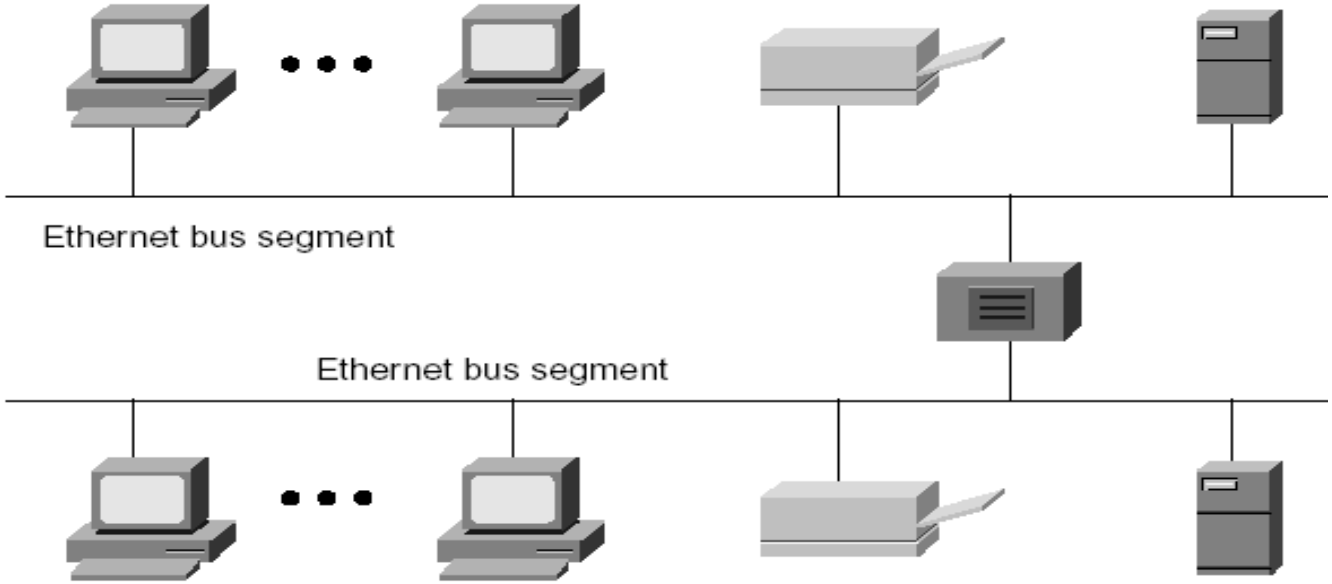
Direkte Verbindung von zwei Computern im voll-Duplex-Modus (twisted pair) durch „cross-over“-Kabel

Preiswerte Kabel, vielseitige Netz-Topologien

Hardware - Ethernet



Punkt-zu-Punkt

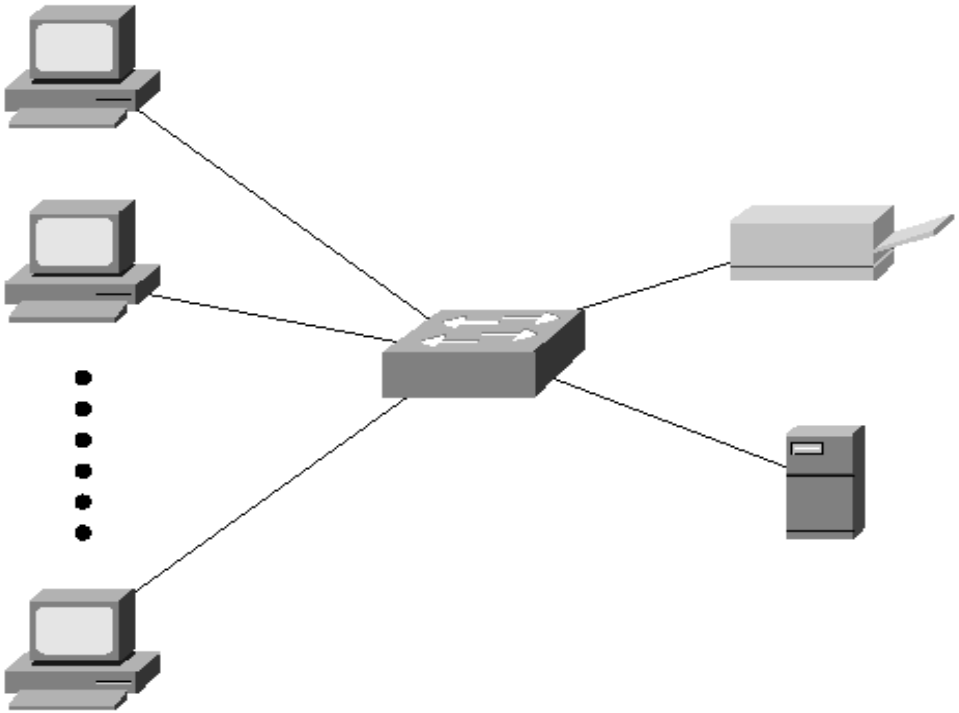


„Thin-wire Ethernet“

Kabel-Segmente, verbunden durch „Repeater“; bis zu 500 m Länge, ~100 Stationen

Achtung, historisch !

Hardware - Ethernet



Stern-Topologie
über Hub (simpler Verstärker)
oder Switch (mit „Intelligenz“)

Einzelne Kabel bis max. 100m

Jedes Gerät hat eine eindeutige Hardware-Adresse: die MAC-Adresse;

Auf „Protokoll-Ebene“ (IP-Protokoll) gibt es eine logische Adresse
momentan 4 Bytes, in Zukunft 16 („Ipv6“), z.B. 192.168.0.1
fast immer auch mit einem Namen verknüpft, z.B. ekplx1.physik.uni-karlsruhe.de

Hardware - Bussysteme

Es gibt noch eine Vielzahl anderer Bussysteme

- CAMAC (alter Industrie-Standard)
- VME **(Industriestandard, basierend auf Motrola 68000 Busprotokoll)**
- FASTBUS (10 Mbit, 32-bit parallel, HEP Eigenentwicklung)
- Fibre-Channel (z.Zt. 2 Gbit seriell, optisch, für Storage Area Networks)
- iSCSI (SCSI über Ehternet)
- InfiniBand
- ...

Heute ist es meist nicht mehr nötig, eigene Bus-Interfaces zu entwickeln – es gibt
- fertige Chips mit
- entsprechenden Treibern für die gängigen Betriebssysteme

Hardware – Steuerung von Datentransfers

Datenübertragung zwischen Peripherie und Speicher muss gesteuert werden:

1. „Polling“: vollständig unter CPU-Kontrolle; CPU fragt Status ab und transferiert Daten über die internen CPU-Register in den Speicher

   hohe CPU-Belastung während Datentransfer, während der Abfrage ist die CPU blockiert

2. Interrupt-Gesteuert: Schnittstelle unterbricht die CPU nach dem aktuellen Befehl über spezielle Interrupt-Leitungen; Datentransfer über CPU;

   Rücksprung zu vorheriger Tätigkeit

   keine Blockierung der CPU durch Status-Abfragen, CPU-Belastung durch Datentransfers

3. DMA (**D**irect **M**emory **A**ccess): Interrupt-gesteuert, CPU initiiert Datentransfer, Controller überträgt Daten und signalisiert Ende der Übertragung per Interrupt. Lohnt sich bei der Übertragung großer Datenmengen

   geringst-mögliche CPU-Belastung

Virtuelle Hardware

Software-Emulation von Hardware auf Wirts-System

- seit langem bei Großrechnern
- Alte Rechner auf moderner Hardware/OS, z.B. Atari, V24, Apple
- Windows- oder Linux-Betriebssystem auf Linux und/oder Windows (bisher kommerziell)
- Neue Entwicklung: - Open Source-Virtualisierung XEN oder KVM unter Linux (schon enthalten in modernen Distributionen)
  - Freier Client für VM-Ware virtuelle Maschinen DOS, Windows oder Linux auf Linux oder Windows siehe http://www.vmware.com
  - VirtualBox, Freie Virtualisierungslösung Windows, Linux u. Mac, siehe http://www.virtualbox.org

Virtuelle Maschine mit (freier) Praktikumsosftware unter Ubuntu-Linux
http://www-ekp.physik.uni-karlsruhe.de/~quast/VMroot/