

## 9 Ein- und Ausgabe und Ausnahmebehandlung

### 9.1 Busse

### 9.2 Erfordernisse der Ein- und Ausgabe

### 9.3 Programmierte Ein- und Ausgabe

### 9.4 Beispiel: PDP-11

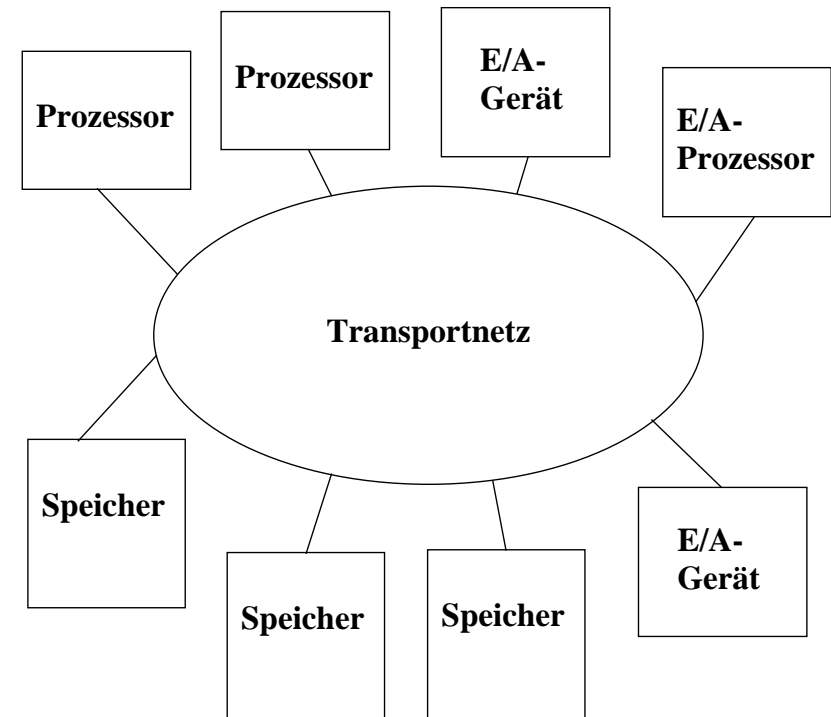
### 9.5 Unterbrechungen

### 9.6 Beispiel: IBM-PC

### 9.7 Ausnahmebehandlung in Sparc-Systemen

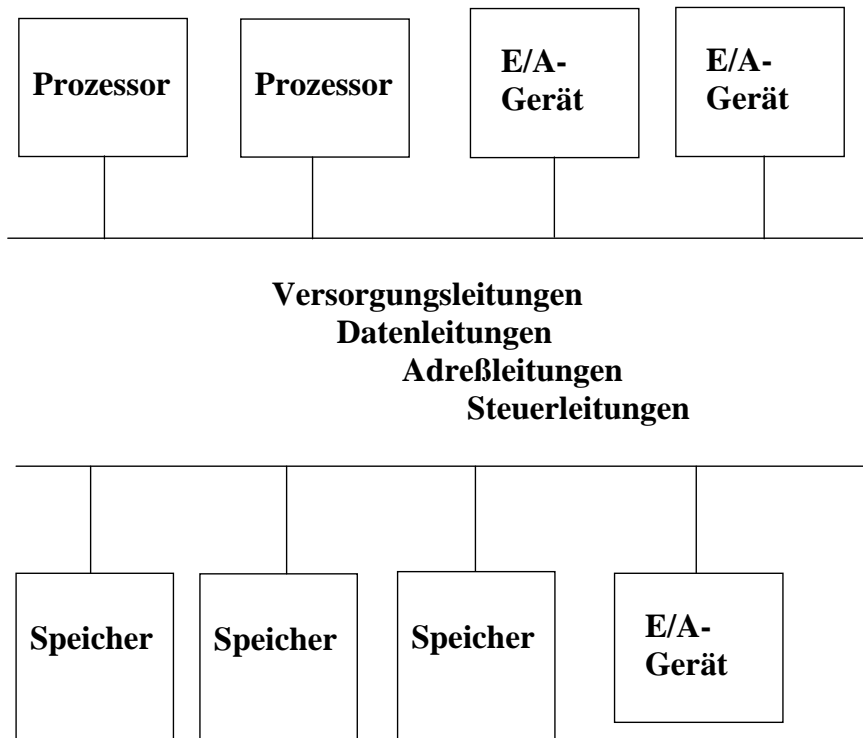
### 9.8 "Direct Memory Access"

Rechenanlage als Ansammlung von Einzelgeräten:



**Bemerkung:** Der Datenaustausch zwischen zwei Geräten ist durch Protokolle zu regeln.

Ein einfaches Transportsystem ist der Bus.



**Bemerkung:** Das Wort Bus ist eine Verkürzung des lateinischen Wortes omnibus. Es existieren viele Busstandards, zwei bekannte sind der PCI-Bus und der SCSI-Bus.

Überblick über SCSI-Transferraten:

SCSI = Small Computer System Interface, Weiterentwicklung von SASI = Shugart Associates Systems Interface; SASI wurde in 1979 entwickelt und bot eine Transferrate von 1,5 MB/s. Im Juni 1986 wurde der SCSI-1 Standard vom ANSI Komitee X3T9.2 verabschiedet.

Name	Bus-Breite in Bit	Bus-Takt in MHz	Transferrate in MB/s
SCSI-1	8	5	5
Fast SCSI	8	10	10
Fast Wide SCSI	16	10	20
Ultra SCSI	8	20	20
Ultra Wide SCSI	16	20	40
Ultra2 SCSI	8	40	40
Ultra2 Wide SCSI	16	40	80
Ultra3 SCSI	16	40	160
Ultra-320 SCSI	16	80	320
Ultra-640 SCSI	16	160	640

SAS = Serial Attached SCSI Transferrate = 3 Gbit/s

**Kenngößen eines Busses:**

**Verbindungsvielfalt:**

- Verbindung zwischen zwei Geräten,
- Verbindung zwischen mehreren ( $\geq 3$ ) Geräten.

**Nutzungsart:**

- Mehrfachnutzung von Leitungen,
- dedizierte Leitungen.

**Bestimmung des temporären Busherrschers:**

- zentralisiertes Verfahren,
- verteiltes Verfahren.

**Zeitsteuerung:**

- Taktsteuerung,
- kein Zentraltakt ("asynchron").

**Busbreite (Zahl der Leitungen für Teilbusse):**

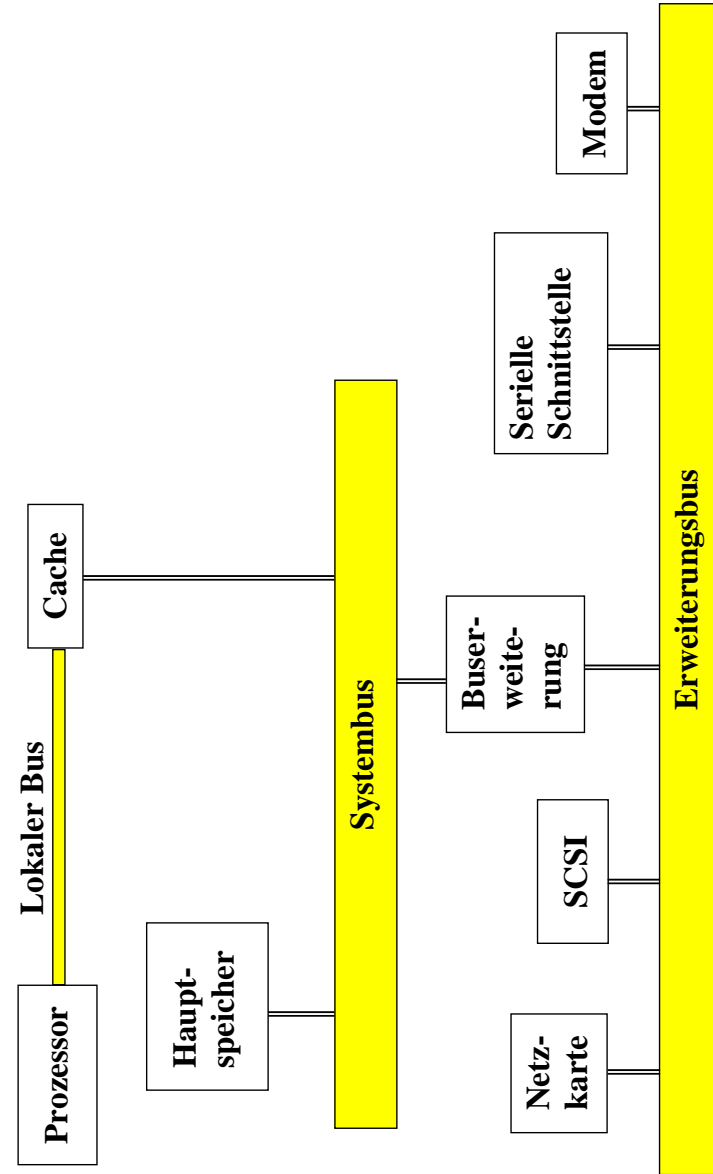
- Daten,
- Adresse.

**Daten-Transfer-Art:**

- "read",
- "write",
- "read-modify-write",
- "read-after-write",
- "block-transfer".

**Bemerkung:** Ein Bus, der viele Einheiten verbindet, stellt einen Systemengpaß dar. Daher finden sich in einer Rechananlage häufig mehrere Bussysteme.

**Beispiel einer traditionellen Mehrbus-Architektur:**



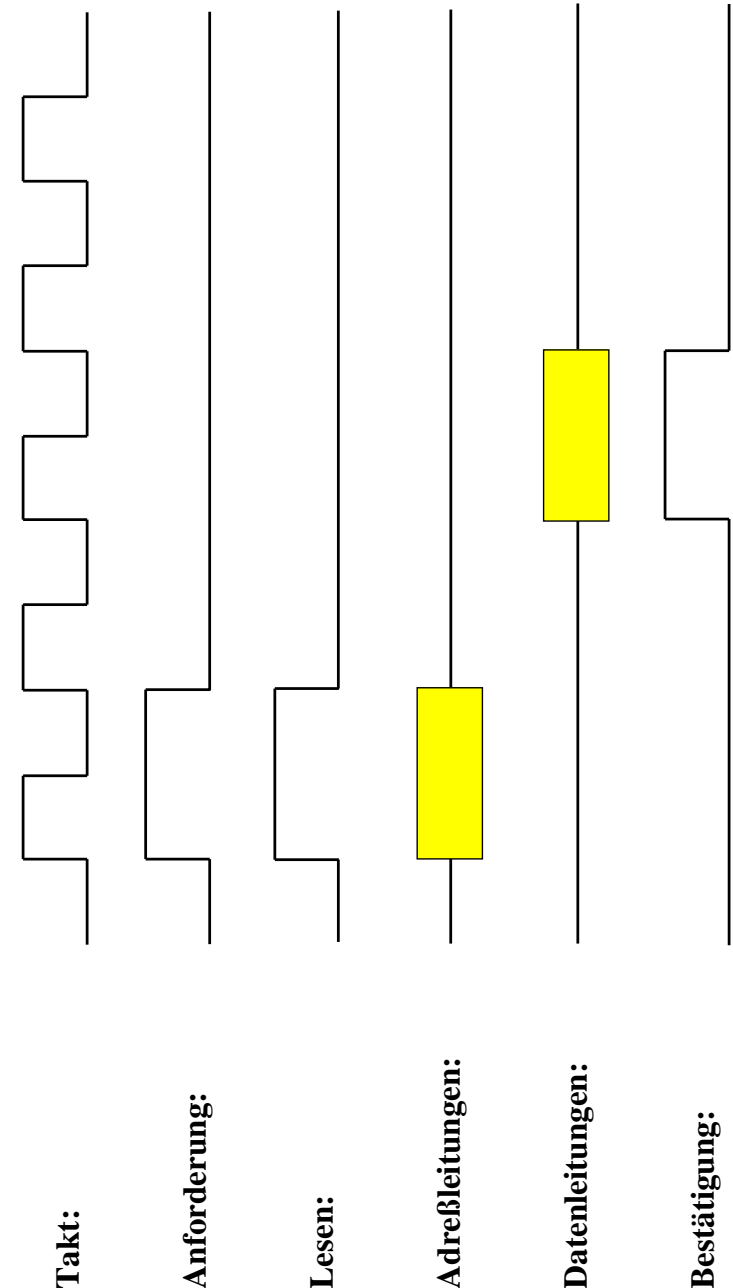
Die beiden nachfolgenden Zeitdiagramme zeigen die Abläufe beim Transport eines Datums aus dem Hauptspeicher zum Prozessor. Es sei angenommen, daß der Prozessor den Lesevorgang steuert.

Die Schritte im einzelnen für einen getakteten Bus:

1. Der Prozessor legt für die Dauer eines Zeittakts die Adresse des Datums auf die Adreßleitungen, er aktiviert die Leseleitung und die Anforderungsleitung.
2. Im gleichen Zeittakt liest der Speicher die Adreßinformation und die Operation.
3. Im nächsten Zeittakt stellt der Speicher das gewünschte Datum bereit.
4. Im übernächsten Zeittakt legt der Speicher das Datum auf die Datenleitungen. Daß die Datenleitungen gültige Information tragen, wird durch die Aktivierung der Bestätigungsleitung angezeigt.
5. Im gleichen Zeittakt übernimmt der Prozessor das Datum.

**Bemerkung:** Der Busentwerfer strebt an, daß jede Busoperation genau einen Zeittakt dauert. Dies einzuhalten ist bei der Verschiedenheit der an den Bus anzuschließenden Einheiten nahezu unmöglich.

Ablauf eines Lesevorgangs auf einem getakteten Bus:

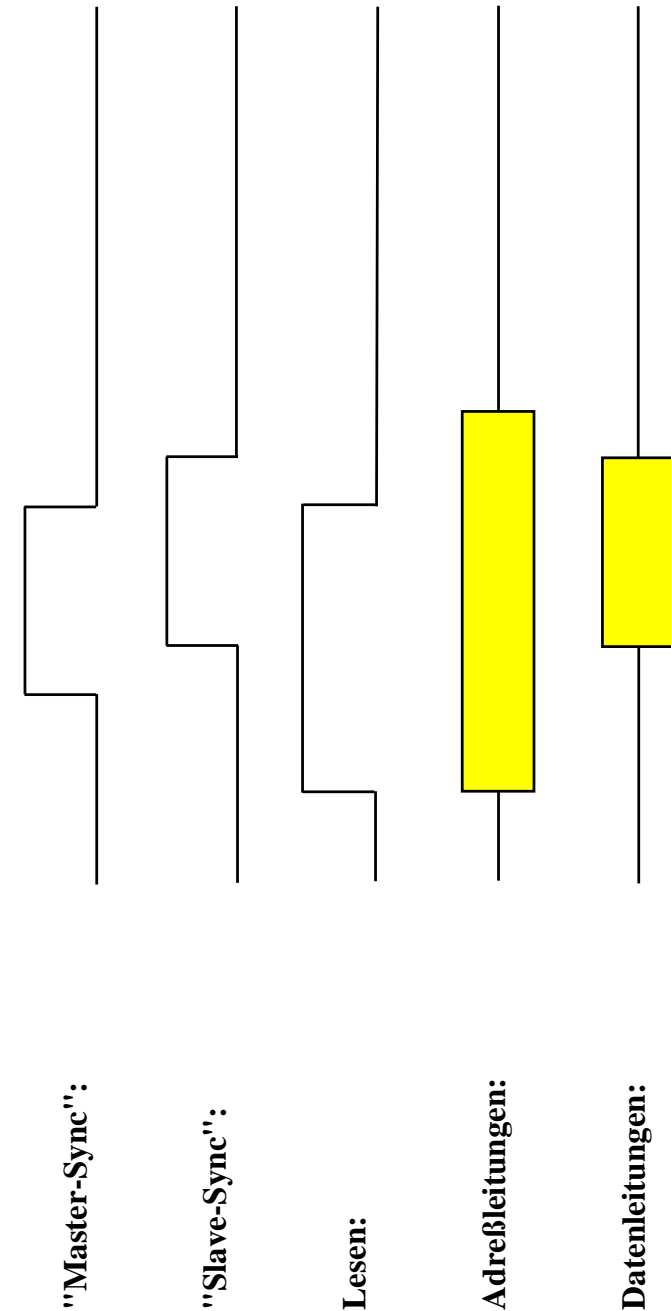


### Die Schritte im einzelnen für einen asynchronen Bus:

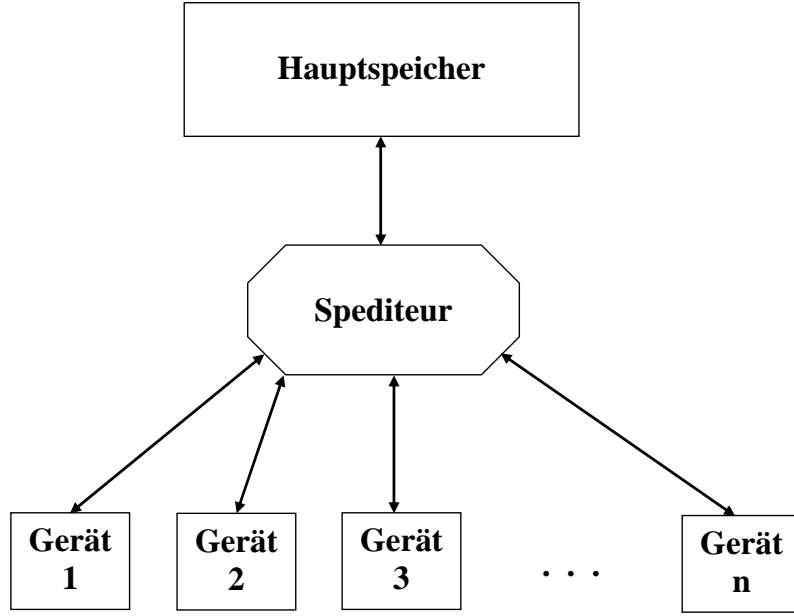
1. Der Prozessor legt die Adresse des gewünschten Datums auf die Adreßleitungen und aktiviert die Leseleitung.
2. Nachdem die Signale auf den Datenleitungen und der Leseleitung sich stabilisiert haben, wird über die "Master-Sync-Leitung" angezeigt, daß sich gültige Information auf einem Teil des Busses befindet.
3. Der Speicher reagiert auf den Businhalt und stellt das gewünschte Datum bereit.
4. Der Speicher plaziert das Datum auf den Datenleitungen und bestätigt durch Aktivierung des Signals "Slave-Sync", daß sie gültige Information tragen.
5. Der Prozessor übernimmt das Datum vom Bus und invalidiert die Adreßinformation durch Deaktivierung der Leseleitung und der "Master-Sync-Leitung".

**Bemerkung:** Im Prinzip kann ein synchroner Bus ein höheres Transportvolumen bewältigen als ein asynchroner Bus, denn der Synchronisierungsaufwand entfällt. Dagegen ist ein asynchroner Bus flexibler.

### Ablauf eines Lesevorgangs auf einem asynchronen Bus:

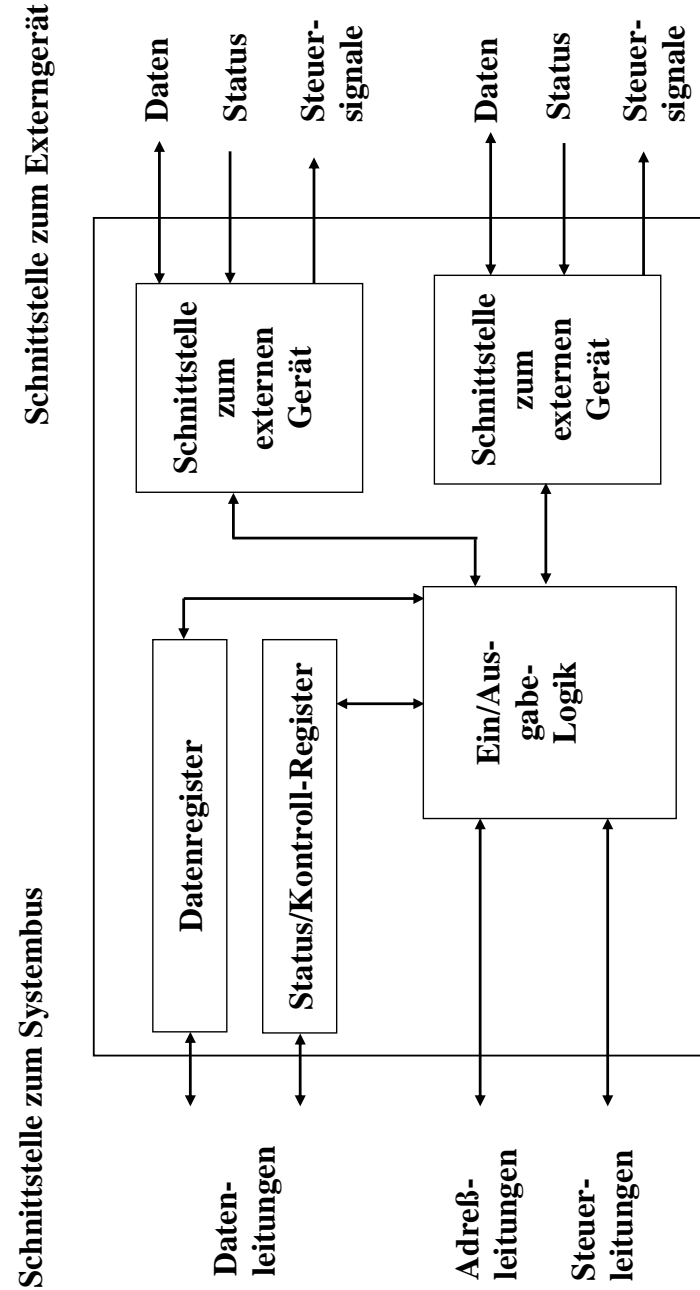


Im Rechnerbau versteht man unter Ein- und Ausgabe den Transport von Daten zwischen Hauptspeicher und Externgeräten.



Als Spediteur zwischen dem Hauptspeicher und den Externgeräten kann der Zentralprozessor fungieren. Da die Aufgabe des Datentransports eine eingeschränkte ist, setzt man für sie spezielle Ein/Ausgabeprozessoren ein. Das Spektrum der E/A-Prozessoren reicht vom Allzweckprozessor über den selbständigen Kanal mit eigenem Befehlsatz bis zur DMA-Einheit. Einen generischen E/A-Modul zeigt die nächste Folie.

**Generischer E/A-Modul:**

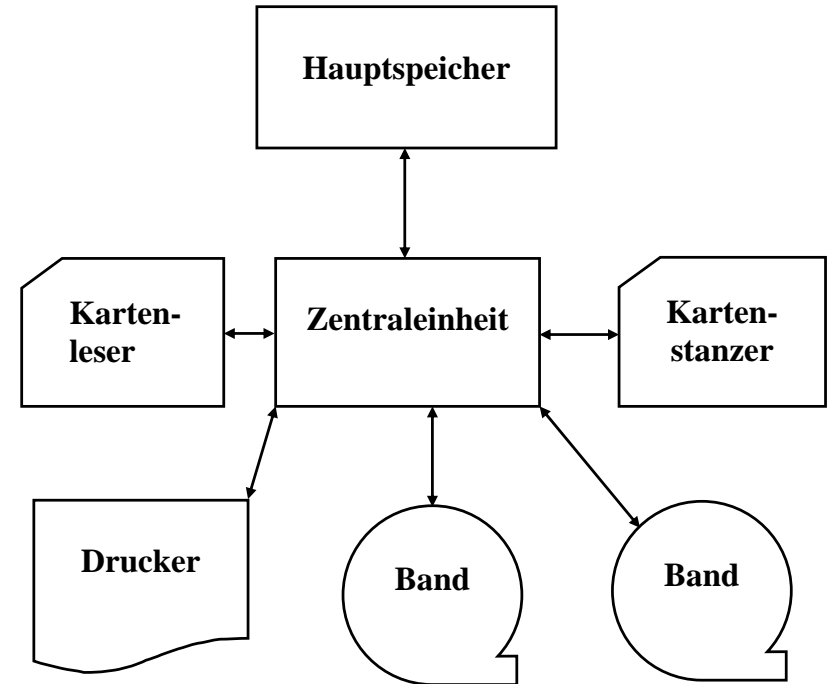


## Einige typische Datentransferraten:

<b>Internet2</b>	$10^{11}$ Bit/s
<b>Gigabit Ethernet</b>	$10^9$ Bit/s
<b>Bildschirm</b>	$10^9$ Bit/s
640 x 480, 8 Bit Farbtiefe, 70 fps:	$1,7 * 10^8$ Bit/s
1024 x 768, 24 Bit Farbtiefe, 100 fps:	$1,9 * 10^9$ Bit/s
1600 x 1200, 24 Bit Farbtiefe, 100 fps:	$4,6 * 10^9$ Bit/s
<b>Ultra DMA 133</b>	$10^9$ Bit/s
<b>Fast Ethernet</b>	$10^8$ Bit/s
<b>Festplatte</b>	$10^8$ Bit/s
<b>Ethernet</b>	$10^7$ Bit/s
<b>Laser Drucker</b>	$10^6$ Bit/s
<b>Modem</b>	$5 * 10^4$ Bit/s
<b>Maus</b>	$10^2$ Bit/s
<b>Tastatur</b>	$10^2$ Bit/s

fps = frames per second

## Ein frühes Rechnersystem:



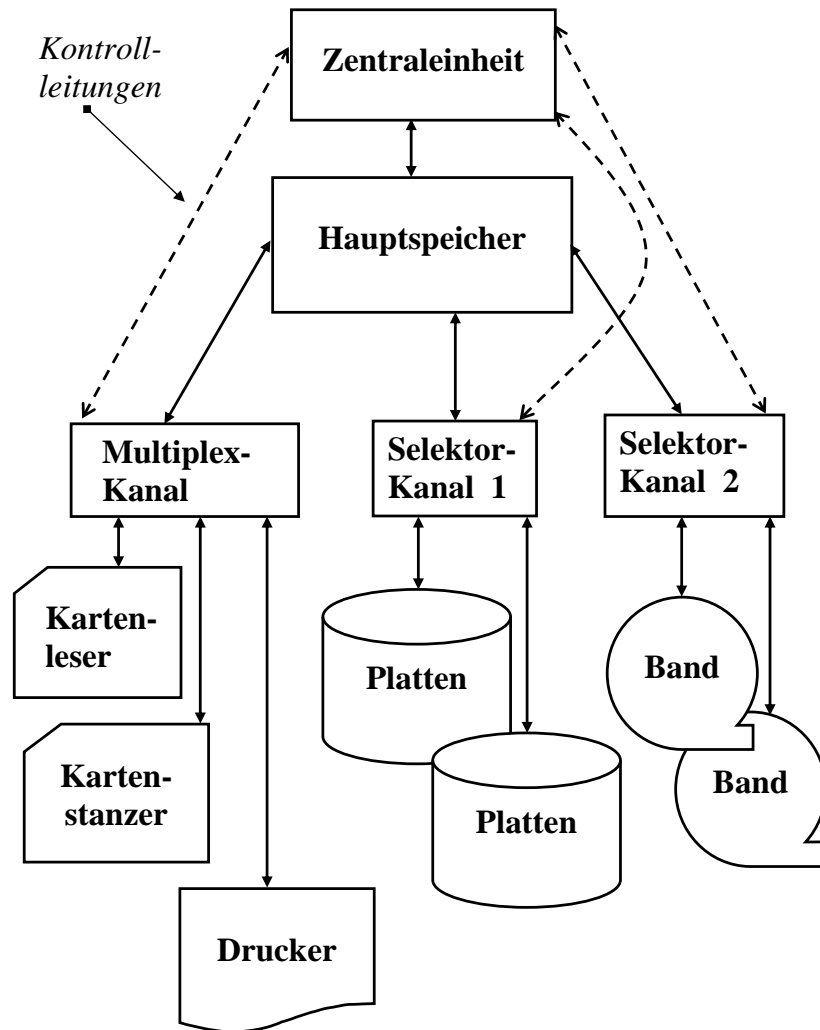
In frühen Rechnersystemen wurde die Ein- und Ausgabe über den Rechnerkern abgewickelt. Neben Befehlen wie ADD und SUBTRACT existierten spezielle Ein/Ausgabebefehle wie READ CARD, PUNCH CARD und PRINT LINE.

Zum Vergleich zwei typische Zeiten:

ADD etwa 1 Millisekunde,

READ CARD etwa 500 Millisekunden  
(120 Karten pro Minute).

## Struktur eines Rechensystems Anfang der siebziger Jahre:



**Bemerkung:** Kanäle sind Spezialprozessoren mit eigenem Befehlssatz.

## Probleme der Ein- und Ausgabe:

Die Vielfalt der Datenendgeräte spiegelt sich in den

Datenübertragungseinheiten (1 Bit bis  $2^{16}$  Bit),  
Datenformaten ("unbegrenzt"),  
Übertragungsgeschwindigkeiten (10 Bit/s bis  $10^{10}$  Bit/s),  
Vorkehrungen zur Fehlerkontrolle.

Die Transferraten zwischen Endgeräten und Hauptspeicher sind bedeutend geringer als die zwischen Hauptspeicher und Zentralprozessor.

Man kennt drei grundsätzliche Arten der Ein- und Ausgabe.

Die programmierte Ein- und Ausgabe, der Zentralprozessor führt die Ein- und Ausgabe durch.

Die unterbrechungsgesteuerte Ein- und Ausgabe, der Zentralprozessor überwacht die Ein- und Ausgabe.

Ein Spezialprozessor führt selbständig den Datentransport zwischen Externgerät und Hauptspeicher durch (z. B. DMA = "Direct Memory Access").

## Programmierte Ein- und Ausgabe:

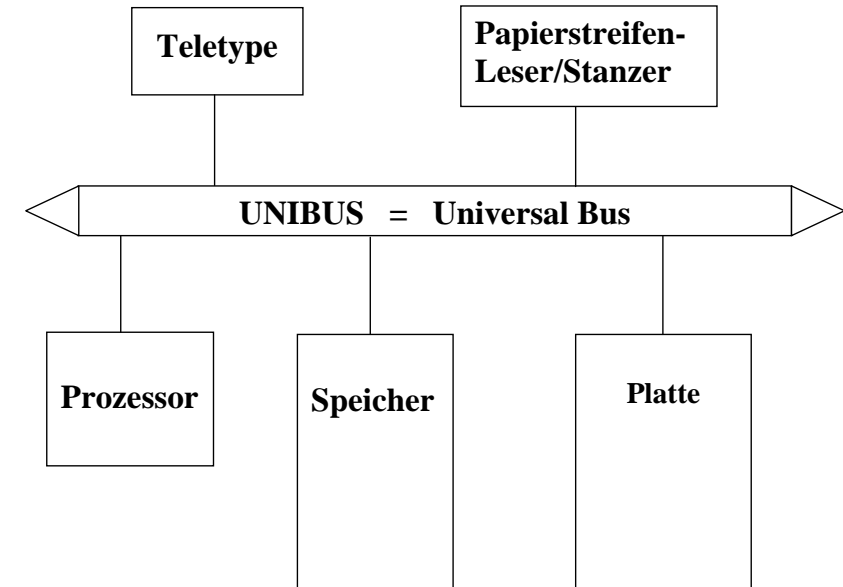
### Schritte bei einer fehlerfreien Dateneingabe vom Gerät G:

1. Der Prozessor bestimmt, welche E/A-Einheit EE das Gerät G verwaltet.
2. Der Prozessor prüft, ob die E/A-Einheit und das Gerät arbeitsbereit sind.
3. Der Prozessor fordert Daten vom Gerät G der Einheit EE.
4. Die E/A-Einheit liest die Daten vom Gerät.
5. Der Prozessor wartet, bis die Daten in der E/A-Einheit vorliegen.
6. Der Prozessor prüft, ob die Daten korrekt an die E/A-Einheit übermittelt wurden.
7. Der Prozessor übernimmt Daten von der E/A-Einheit.

### Bemerkungen:

- (i) Die Datenausgabe gestaltet sich ähnlich.
- (ii) Indem der Prozessor in Schritt 5 ständig Register der E/A-Einheit abfragt, vergeudet er bei langsamen Peripheriegeräten wertvolle Rechenzeit.
- (iii) Die programmierte Ein- und Ausgabe ist einfach. Auf Ausnahmesituation kann sofort reagiert werden.

## Aufbau eines PDP-11-Rechners:



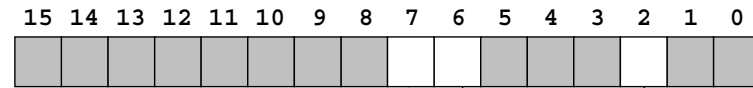
### Kennzeichen:

1. Einheitlicher Adreßraum für Speicher und Geräte, daher Nutzung des gesamten Befehls-repertoires für die Ein- und Ausgabe.
2. Jedes Gerät wird durch einen kleinen Satz von Registern dargestellt.



**Programmbeispiel, um Zeichen vom Lochstreifenleser auf den Drucker des Teletype zu kopieren:**

**Statusregister für Schreiben (TPS) an Adresse 0777564:**



Ready \_\_\_\_\_  
 Interrupt enable \_\_\_\_\_  
 Maintenance \_\_\_\_\_

**TKS = 0777560 ; Adresse des Statusregisters des  
 ; Lochstreifenlesers**  
**TKB = 0777562 ; Adresse des Pufferregisters des  
 ; Lochstreifenlesers**  
**TPS = 0777564 ; Adresse des Statusregisters des  
 ; Teletype-Druckers**  
**TPB = 0777566 ; Adresse des Pufferregisters des  
 ; Teletype-Druckers**

```

COPY:      INC      TKS      ; Bit 0 des Statusregisters
           ; wird auf 1 gesetzt, der
           ; Leser beginnt zu arbeiten.
LOOP1:     TSTB     TKS      ; Fertig?
           BPL      LOOP1   ; Bei nein Warten.
LOOP2:     TSTB     TPS      ; Bereit?
           BPL      LOOP2   ; Bei nein Warten.
           MOVB     TKB, TPB ; Übertragung eines
           ; Zeichens zum Drucker
           BR       COPY    ; nächstes Zeichen,
           ; endlos?
  
```

**Unterbrechungen:**

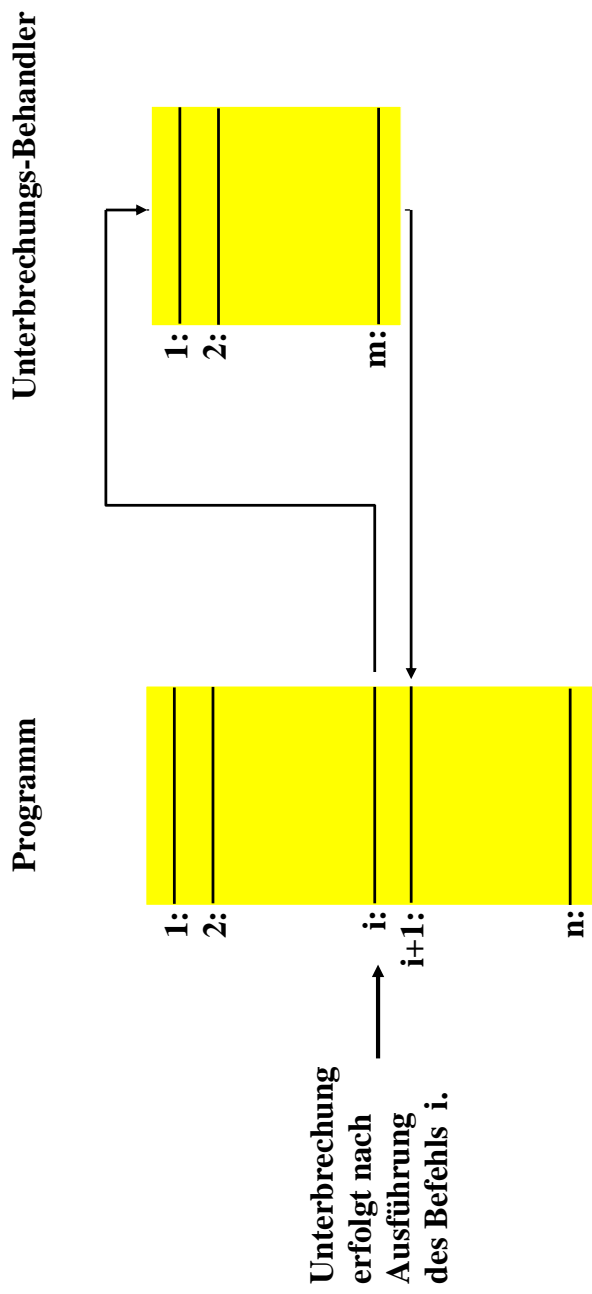
**Durch eine Unterbrechung wird der Prozessor dem laufenden Programm entzogen und einem anderen Programm zugeteilt. Auslöser für einen derartigen Programmwechsel ist im allgemeinen ein Hardware-Signal.**

**Grober Ablauf einer Unterbrechung:**

1. Ein Unterbrechungssignal S tritt auf und wird erkannt.
2. Es wird geprüft, ob das Signal S das aktive Programm AP unterbrechen darf.
3. Wenn nein, wird die Behandlung des Signals S zurückgestellt.
4. Wenn ja, wird der Programmkontext des aktiven Programms an vorbestimmten Orten gerettet.
5. Das dem Signal S zugeordnete Programm SP wird gestartet.
6. Hat das Programm SP seine Aufgabe erledigt, dann wird der Programmkontext von AP restauriert und das Programm AP setzt seine Arbeit fort. Im Idealfall hat eine Unterbrechung keine Auswirkungen auf das unterbrochene Programm.

**Bemerkung:** Im allgemeinen wird eine Unterbrechung nur nach Ausführung eines Maschinenbefehls behandelt.

### Übergabe der Kontrolle mittels Unterbrechungen:



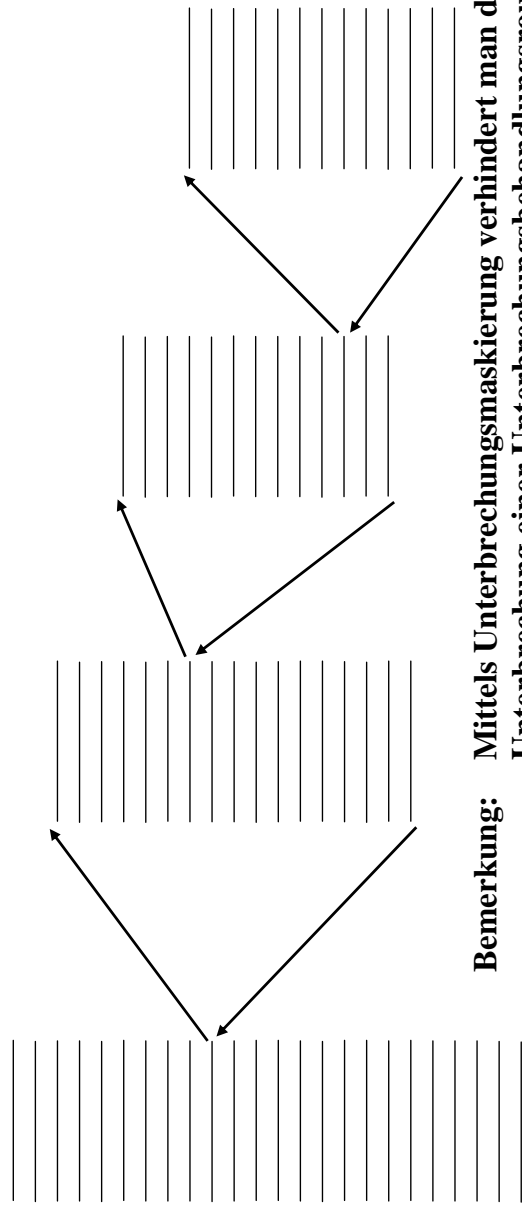
Eine Unterbrechungsbehandlungsroutine kann auch wieder unterbrochen werden.

**Aktives Programm**

**Unterbrechung durch Drucker**

**Unterbrechung durch Netzkarte**

**Unterbrechung durch Uhr**



### Bemerkungen zum Unterbrechungsbegriff:

- (i) Die Unterbrechungsbehandlung wird integriert in eine allgemeine Behandlung von Ausnahmesituationen. Zu den Ausnahmen zählt man u. a. Division durch 0, Schutzverletzungen, Ausrichtungsverletzungen und Seitenalarme.
- (ii) Damit der Kontextwechsel schnell durchgeführt werden kann, wird er weitgehend in Hardware realisiert.
- (iii) Den einzelnen Ausnahme-Ursachen ordnet man verschiedene Prioritäten zu. Im Konfliktfall wird die Unterbrechung höherer Priorität behandelt.
- (iv) Unterbrechungsanforderungen können durch Überschreiben verloren gehen.
- (v) Die Unterbrechungen numeriert man gerne durch. Da die Startadresse einer Unterbrechungsbehandlungsroutine ohne weitere Umstände aus der Unterbrechungsnummer herleitbar sein sollte, nutzt man oft die Wörter mit niederen Hauptspeicheradressen als Unterbrechungsvektoren.

### Beispiel zur unterbrechungsgesteuerten Ein- und Ausgabe:

Lesen von 10 Zeichen von der Tastatur (PDP-11 Text):

```
START:  MOV  #10, R0      ; Setzen Zähler.
        MOV  #2000, R1   ; Puffer bereitstellen
        MOV  #101, TKS   ; Gerät und Unter-
                               ; brechung initialisieren.
WARTE:  BR    WARTE     ; Hier nur Demonstration,
                               ; Hauptprogramm wartet.

        :

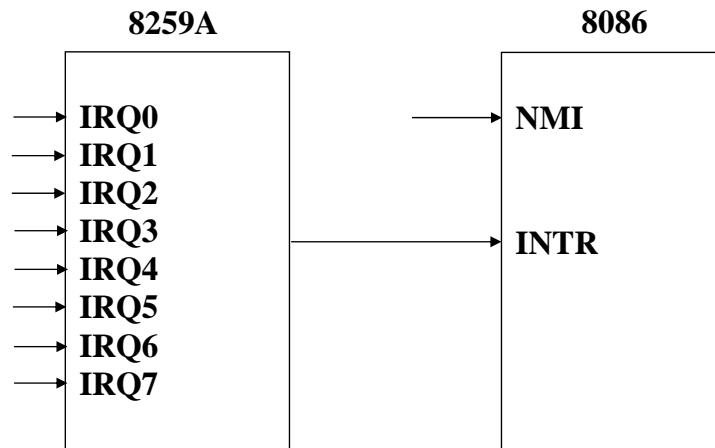
; Die Adressen 60 - 63 sind der Tastatur zugeordnet.
60:     RDINT           ; Anfangsadresse der
                               ; Unterbrechungsroutine
62:     000200         ; Priorität 4

        :

RDINT:  MOVB  TKB, (R1)+ ; Zeichen in Puffer.
        DEC  R0         ; Zähler justieren.
        BEQ  E1         ; Letztes Zeichen?
        INC  TKS        ; Leser initialisieren.
        RTI           ; Return from Interrupt
E1:     TST (SP)+, (SP)+ ; Keller bereinigen.
        CLR  TKS        ; Nullzustand
                               ; herstellen.
        BR   WARTE + 2 ; Rückkehr zum
                               ; Hauptprogramm.
```

**Bemerkung:** Das Beispiel nutzt nicht die Vorteile der Auftragsdelegation.

## 8086-Systeme:



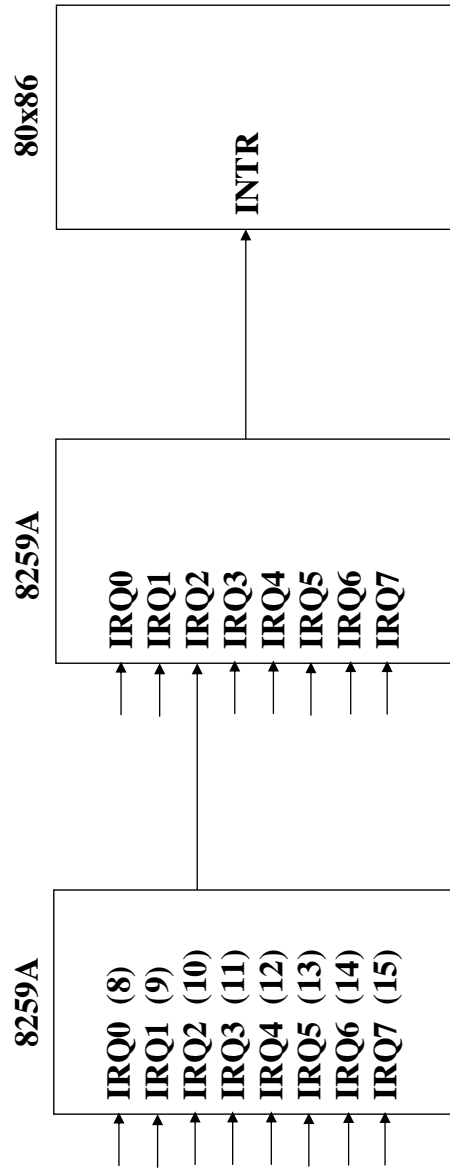
## Bemerkungen:

- (i) 80x86 Prozessoren besitzen zwei Unterbrechungseingänge, einen normalen und einen NMI-Eingang (NMI = "Non-Maskable Interrupt").
- (ii) Ein 8086-System besitzt acht Unterbrechungsleitungen, die von einem 8259A-Baustein verwaltet werden. Der Unterbrechungs-Controller 8259A ist über die Ports 20h und 21h programmierbar.
- (iii) Die nicht maskierbare Unterbrechung kann nicht über den CPU-Befehl CLI unterdrückt werden, wohl aber über Port 70h. Der "NMI" dient dazu, schwerwiegende Hardware-Fehler, wie z. B. Paritätsfehler und Stromausfall, zu erkennen.

## Struktur einer Unterbrechungs-Behandlungs-Routine bei einem Einzel 8259A PIC (PIC = Programmable Interrupt Controller):

1. Rette alle Register.
2. Führe nicht unterbrechbaren Code aus.
3. Ermögliche Unterbrechungen.
4. Führe unterbrechbaren Code aus.
5. Unterbinde Unterbrechungen.
6. Sende nichtspezifischen End-Of-Interrupt an PIC.
7. Restauriere Register.
8. Rückkehr aus Unterbrechungsbehandler.

### Kaskadierung von Unterbrechungen:



Die normale Prioritätsanordnung der Unterbrechungen ist:  
IRQ0, IRQ1, IRQ8 – IRQ15, IRQ3 – IRQ7

### Struktur einer Unterbrechungsbehandlungsroutine bei Kaskadierung von 8259A Bausteinen:

B = "Slave" und A = "Master":

1. Rette alle Register.
2. Führe nicht unterbrechbaren Code aus.
3. Ermögliche Unterbrechungen.
4. Führe unterbrechbaren Code aus.
5. Unterbinde Unterbrechungen.
6. Sende nichtspezifischen End-Of-Interrupt an PIC B.
7. Lese ISR des PIC B.
8. Ist niedriger priorisierte Unterbrechung von B in Bearbeitung?

Nein: Sende nichtspezifischen End-Of-Interrupt an A.

Restauriere Register.

Rückkehr von Interrupt.

Ja: Restauriere Register.

Rückkehr von Interrupt.

**Unterbrechungen im PC, Zuordnung von Unterbrechungsleitungen und Unterbrechungsvektoren:**

1. Der 80x86 erlaubt bis zu 256 Unterbrechungen. Die Adressen der Unterbrechungsroutinen sind 0:0 bis 0:03fch. Unterbrechung ist hier ein Oberbegriff für Ausnahme, Software-Interrupt und Hardware-Interrupt.
2. Beispiele für Ausnahmen:
  - INT 0: Divisionsfehler
  - INT 1: Einzelschrittmodus
  - INT 5: Grenzverletzungen bei Array-Zugriff
  - INT 6: Falscher Befehlscode
3. Geräte-Unterbrechungen im AT:

Leitung	U-Vektor	Gerät
IRQ 0	8	Zeitunterbrechung
IRQ 1	9	Tastatur
IRQ 2	0ah	Kaskade für Controller 2
IRQ 3	0bh	Serielle Schnittstelle 2
IRQ 4	0ch	Serielle Schnittstelle 1
IRQ 5	0dh	Parallele Schnittstelle 2
IRQ 6	0eh	Diskettenlaufwerk
IRQ 7	0fh	Parallele Schnittstelle 1
IRQ 8	70h	Echtzeituhr
IRQ 9	71h	umgeleitet auf IRQ 2
IRQ 10	72h	reserviert
IRQ 11	73h	reserviert
IRQ 12	74h	reserviert
IRQ 13	75h	FPU Unterbrechung
IRQ 14	76h	Festplattenlaufwerk
IRQ 15	77h	reserviert

**Unterbrechungsbehandlung für PC-Bus im Überblick:**

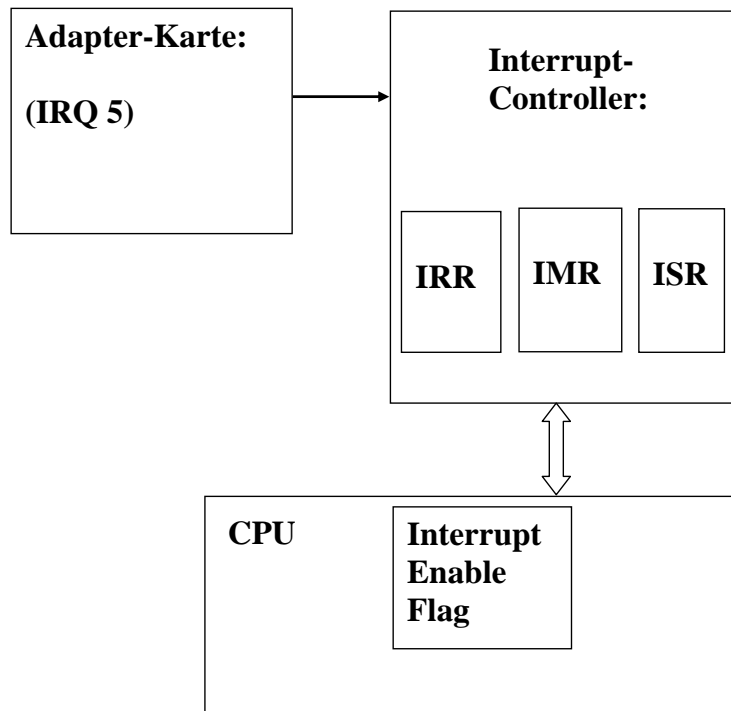
**Schritte der Unterbrechungsbehandlung im System:**

8259A akzeptiert Unterbrechung,  
 8259A bestimmt Priorität,  
 8259A signalisiert Unterbrechung,  
 Prozessor bestätigt Unterbrechung,  
 8259A legt Unterbrechungsvektor auf Datenbus, Prozessor bearbeitet Unterbrechung.

**Schritte der Unterbrechungsbehandlung durch Prozessor:**

Prozessor erhält Unterbrechungsaufforderung, Prozessor beendet gegenwärtigen Befehl, Prozessor bestätigt Unterbrechungssignal, Prozessor rettet Statuswort und Befehlszähler, Prozessor lädt neues Statuswort und neue Befehlsadresse, der volle Maschinenzustand wird von Unterbrechungsroutine gerettet, Unterbrechung wird bearbeitet, Maschinenzustand wird restauriert, altes Statuswort und alter Befehlszähler werden restauriert.

### Ein Beispiel zum Ablauf einer Unterbrechung im PC:



### Ablauf einer Unterbrechung, hier als Beispiel mit Unterbrechung 5:

1. Ein Gerät meldet eine Unterbrechung über Eingang IR 5 des Interrupt-Controllers an.
2. Im IRR des Controllers wird das Bit 5 gesetzt.
3. Es wird geprüft, ob Unterbrechung 5 zulässig ist, bei Zulässigkeit wird zu Schritt 4 übergegangen.
4. Es wird überprüft, ob eine höher priorisierte Unterbrechung ansteht.
5. Falls nein, wird die CPU über Leitung INT benachrichtigt.
6. Falls Unterbrechungen zulässig sind, sendet die CPU in kurzem Abstand zweimal ein INTA-Signal.
7. Bei Empfang des ersten INTA-Signals wird das Bit 5 im ISR gesetzt und Bit 5 im IRR gelöscht. Bei Empfang des zweiten INTA-Signals sendet der Controller die Adresse der Unterbrechungs-Behandlungs-Routine an die CPU.
8. Die CPU rettet den Zustand des gegenwärtigen Prozesses auf den Prozeßkeller und beginnt die Ausführung der Unterbrechungs-Routine.
9. Die Unterbrechungsbehandlung wird ausgeführt, in ihrem Verlauf wird das Unterbrechungssignal zurückgenommen.
10. Die Unterbrechungs-Routine sendet End-of-Interrupt an den Controller, Datenwert 20h an Port 20h.
11. Der End-of-Interrupt Befehl löscht das Bit 5 im ISR, nun kann wieder eine neue Unterbrechung über IR 5 behandelt werden.
12. Mit dem Befehl IRET ("Return from Interrupt") wird die Unterbrechungsbehandlung beendet und der unterbrochene Prozeß wieder aufgenommen.

### Prozessor-Status-Wort:

31..28	27..24	23..20	19..14	13	12	11..8	7	6	5	4..0
Impl	Ver	ICC	reserved	EC	EF	PIL	S	PS	ET	CWP

### Abkürzungen:

**Impl** = Implementation  
**Ver** = Version  
**ICC** = Integer Condition Codes = (N, Z, V, C)  
**EC** = Enable Coprocessor  
**EF** = Enable Floating-Point Processor  
**PIL** = Processor Interrupt Level  
**S** = Supervisor Mode  
**PS** = Previous Supervisor Mode  
**ET** = Enable Trap  
**CWP** = Current Window Pointer

### Trap Base Register

31..12	11..4	3..0
Trap Base Address	Trap Type	zero

Die Verteiltable für Traps enthält 256 Einträge. Jeder Eintrag umfaßt 16 Byte. Diese stellen die ersten vier Befehle des den betreffenden Trap behandelnden Codes dar.

### Beispiel für Trap Nr. 2:

```
rd    %psr, %l0
sethi %hi(nwindows), %l3
jmp   %l3 + 0x280
rd    %wim, %l3
:
```

## Kennzeichen von Benutzer- und Betriebssystemmodus:

### Benutzermodus:

**Benutzercode kann nur auf eigene Text- und Datensegmente zugreifen.**

**Nur unkritische Befehle dürfen ausgeführt werden.**

**Es ist nur ein beschränkter Zugriff auf die Maschinenregister möglich.**

### Betriebssystemmodus:

**Der Zugriff auf den Speicher ist unbeschränkt.**

**Der gesamte Befehlssatz kann genutzt werden.**

**Auf alle Maschinenressourcen kann zugegriffen werden.**

**Bemerkung:** Nur über Traps gelangt man in den Betriebssystemmodus.

## Auszug aus Liste von Unterbrechungen und Ausnahmen:

Name	Priorität	Nummer
reset	1	
instruction access error	3	0x21
data access error	12	0x29
register access error	4	0x20
instruction access exception	5	0x1
data access exception	13	0x9
mem-address not aligned	10	0x7
instruction access MMU miss	2	0x3c
data access MMU miss	12	0x2c
window overflow	9	0x05
window underflow	9	0x06
privileged instruction	6	0x03
illegal instruction	7	0x02
fp disabled	8	0x04
fp exception	11	0x08
division by zero	15	0x2a
tag overflow	14	0x0a
trap instruction	16	0x80 - 0xff
interrupt level 15	17	0x1f
⋮		
interrupt level 1	31	0x11

## Ablauf eines Trap:

(i) Sei Enable Trap gesetzt.

1. `ET := 0` /\* Verhindern eines weiteren Trap \*/
2. `PS := S` /\* Retten des Prozessorzustands \*/
3. `S := 1` /\* Übergang in Betriebssystemmodus \*/
4. `CWP := (CWP-1) mod NWINDOWS`  
/\* Übergang zum nächsten Register-  
fenster ohne Prüfung auf Überlauf \*/
5. `%l1 := PC, %l2 := nPC`
6. `TBR.tt := Trap-Nummer`
7. `PC := TBR; nPC := TBR+4`  
/\* Ausführung der Trap-Routine \*/  
/\* Falls die lokalen Register zur Ausnahmebe-  
handlung nicht ausreichen, muß das Register-  
fenster explizit gerettet werden. \*/
8. Rückkehr zum unterbrochenen Programm mittels  
`rett.` (`rett = return from trap` \*)

(ii) Sei Enable Trap nicht gesetzt.

Unterbrechungen werden ignoriert.  
Genuiner Trap führt zu einem Maschinen-Reset.

## Zwei Arten des Rücksprungs:

(i) Wiederholen des Trap-Verursachers:

```
jmpl    %l1, %g0    ! alter Befehlszähler
rett    %l2
```

Ablauf: Ausführen von `jmpl`,  
Ausführen von `rett`,  
erneutes Ausführen des Trap-Verursachers.

(ii) Rücksprung nach "letzter" Instruktion:

```
jmpl    %l2, %g0    ! alter nPC
rett    %l2 + 4
```

Ablauf: Ausführen von `jmpl`,  
Ausführen von `rett`,  
Ausführen von Befehl bei "old nPC"

## Bemerkungen:

1. Damit die obigen Rückkehreschemata sinnvoll sind, darf eine Trap-Instruktion nicht verzögert ausgeführt werden.
2. Es kann sinnvoll sein, eine Instruktion ein zweites Mal auszuführen, z.B. `save`.

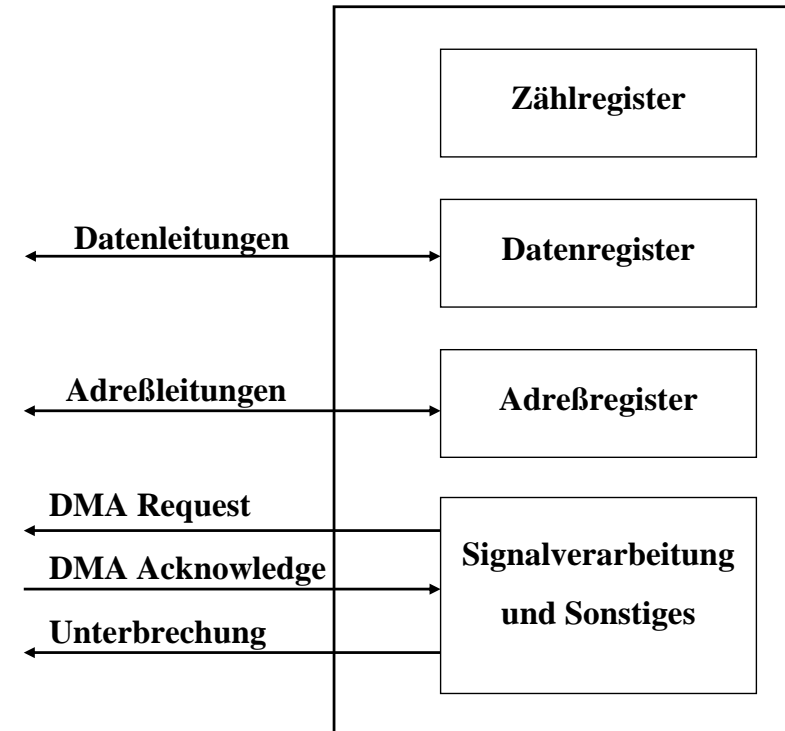
**Der Befehl Ticc der Sparc V8:**

Kürzel	Bezeichnung	Bedingung
TA	Trap Always	true
TN	Trap Never	false
TNZ	Trap on Not Zero	not Z
TZ	Trap on Zero	Z
TG	Trap on Greater	not (Z or (N xor V))
TLE	Trap on Less or Equal	Z or (N xor V)
TGE	Trap on Greater or Equal	not (N xor V)
TL	Trap on Less	N xor V
TGU	Trap on Greater Unsigned	not (C or Z)
TLEU	Trap on Less or Equal Unsigned	C or Z
TCC	Trap on Carry Clear	not C
TLU	Trap on Less, Unsigned	C
TPOS	Trap on Positive	not N
TNEG	Trap on Negative	N
TVC	Trap on Overflow Clear	not V
TVS	Trap on Overflow Set	V

**Bemerkung:** Die Trap-Nummer errechnet sich aus der Summe von 128 und den niederwertigsten sieben Bit des Operanden.

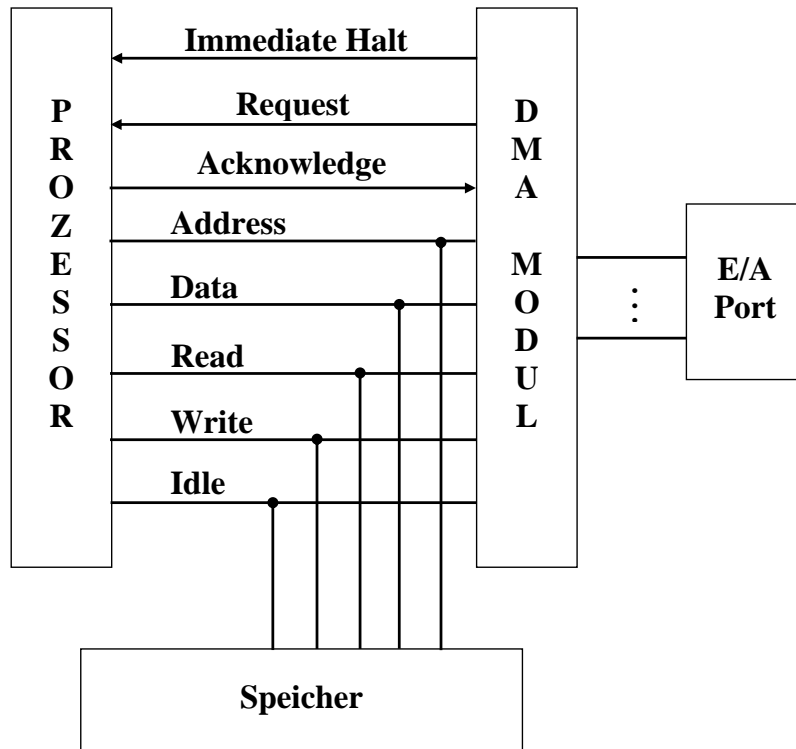
**DMA ("Direct Memory Access"):**

**Ein DMA-Modul ist ein sehr einfacher E/A-Modul:**



**Bemerkung:** Die Anbindung des DMA-Moduls an die Ein- und Ausgabegeräte wird nicht gezeigt.

### Zur Kommunikation Prozessor – DMA-Modul:



**Bemerkung:** Im Bild sind drei Möglichkeiten der exklusiven Nutzung des Systembusses durch den DMA-Modul vorgesehenen:

- (i) über Immediate Halt,
- (ii) über Abfragen der Idle-Leitung,
- (iii) über ein Request – Acknowledge – Protokoll.

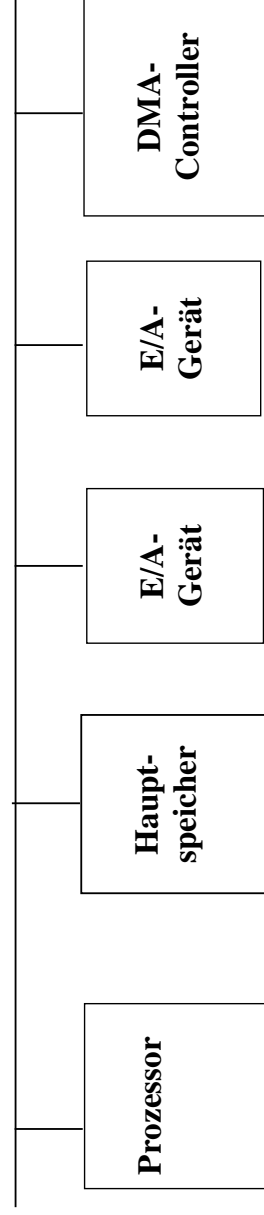
### Grober Ablauf eines DMA-Transfers:

1. Der Prozessor lädt das Zählregister und das Adreßregister des DMA-Moduls mit ihren Anfangswerten. Im Adreßregister steht die Speicheradresse, im Zählregister die Wortzahl.
2. Ist der DMA-Modul bereit, einen Datentransfer durchzuführen, aktiviert er das DMA Request Signal. Der Prozessor honoriert die Busanforderung mit dem Signal DMA Acknowledge.
3. Ein Datenwort zwischen Speicher und DMA-Modul wird transferiert. Zähler und Adresse werden aktualisiert.
4. Der DMA-Modul gibt den Bus frei durch Deaktivierung des DMA Request Signals. Der Prozessor reagiert mit der Deaktivierung des Signals DMA Acknowledge.
5. Falls der Zähler ungleich 0 ist, erfolgt eine Wiederholung des Zyklus ab Schritt 2. Falls der Zähler gleich 0 ist, wird der Prozessor durch ein Unterbrechungssignal benachrichtigt.

**Bemerkung:** Die zeitweilige, kurzfristige Benutzung des Systembusses zur Datenübertragung durch den DMA-Modul ist unter dem Namen "cycle stealing" bekannt.

## DMA-Konfiguration (1)

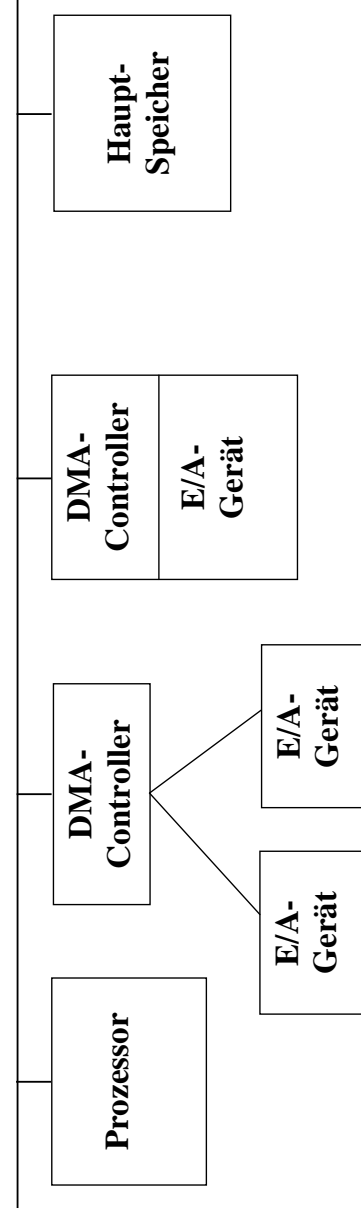
### Ein-Bus-System:



**Bemerkung:** In dieser Konfiguration ist es denkbar, daß jeder Datentransfer den Bus zweimal beansprucht, ein Datum wird zwischen E/A-Gerät und DMA-Controller und zwischen DMA-Controller und Hauptspeicher übertragen; der Prozessor wird zweimal gestört.

## DMA-Konfiguration (2)

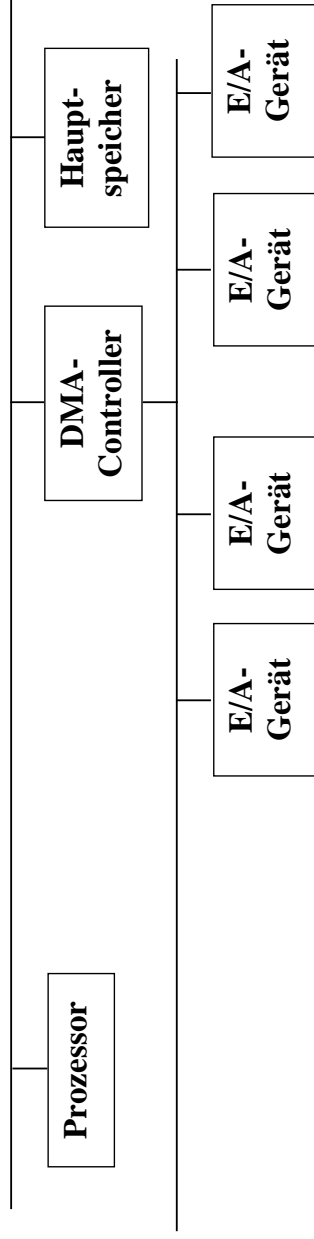
### Ein-Bus-System mit integrierten DMA-Controllern:



**Bemerkung:** Jeder Datentransfer beansprucht den Bus nur einmal.

### DMA-Konfiguration (3)

Separater E/A-Bus:



**Bemerkung:** Jeder Datentransfer beansprucht den Bus nur einmal, daher wird der Prozessor nur einmal pro Datentransfer gestört.