

## 5. Entwurf mit Bausteinen

Bei komplexen Systemen wird nicht mehr auf der Ebene logischer Variablen entworfen:

- Anzahl der Variablen würde zu groß
- Aufwand für Minimierung und Zustandszuweisung zu hoch
- Übersicht über den Entwurf geht verloren

Entwurf mit Bausteinen z.B. Register, Decoder, Multiplexer, ALU

- Diese Bausteine werden zusammenschaltet und mit einer übergeordneten Steuerung versehen.

Standard-Bausteine vom Typ Schaltnetz:

- Decoder
- Multiplexer
- Lesespeicher (ROM)
- Programmierbare Logische Anordnung (PLA)

Argumente für den Einsatz von Bausteinen:

- „Klassisch“ optimierte Funktionen (wenige Variable u. Gatter) stellen nicht unbedingt die optimale Lösung dar, wegen des Entwurfsaufwandes und der geringen Übersicht
- Gewünscht ist eine Minimierung von Anschlüssen und externen Verbindungen
- Nutzungsgrad der Bausteine ist nicht von großer Bedeutung

Vorgehen beim Entwurf:

- Prüfen, ob Baustein gewünschter Funktion auf dem Markt erhältlich ist
- Wenn nicht, prüfen, ob Gesamtfunktion auf Standardbausteine aufgeteilt werden kann
- Wenn nicht, Realisierung der Funktion auf Bausteinen mit Standardtechniken

### 5.1 Decoder

Ein Decoder ist ein Schaltnetz, das Binärzeichen von  $n$  Eingängen auf max.  $2^n$  Ausgänge einsetzt. Damit können unterschiedliche Darstellungen von Zeichen ineinander überführt werden (Umcodierung). Unvollständige Decoder haben weniger als  $2^n$  Ausgänge.

### 5.2 Decoder Schaltnetz

Decoder erzeugen Minterme bzw. Maxterme der Eingangsvariablen. Beliebige Schaltfunktionen lassen sich mit Mintermen oder Maxtermen realisieren. Beliebige Schaltfunktionen lassen sich mit Decodern realisieren, indem man die benötigten Min- bzw. Maxterme ausgangseitig verknüpft.

### 5.3 Demultiplexer

Decoder mit ENABLE-Eingängen können als Demultiplexer eingesetzt werden. Die Decodereingänge werden zu Auswahleingängen (Select) des Demultiplexers. Der ENABLE-Eingang wird zum Dateneingang.

### 5.4 Multiplexer

Erlaubt, aus einer Anzahl von Dateneingängen mit Hilfe von Selectvariablen einen auszuwählen. Jede Kombination der Selectvariablen wählt einen Eingang zur Weiterschaltung aus.

Arbeitsblatt 8 – 4x1 Multiplexer

$$a = C_0 \bar{S}_0 \bar{S}_1 + C_1 \bar{S}_0 S_1 + \bar{C}_2 S_0 S_1 + C_2 \bar{S}_0 \bar{S}_1 = S_0 S_1 + S_0 S_1 = S_0 = S_1$$

$$C_3 = C_0 = 1$$

$$C_1 = C_0 = 0$$

## 5.5 Schaltnetze mit Multiplexern

In Multiplexern sind vollständige Decoder bzgl. der Selectvariablen enthalten d.h. mit MUX sind Schaltnetze realisierbar. Dazu wird eine Eingangsbeschaltung benötigt um die gewünschten Minterme auszuwählen. Im einfachsten Fall werden die Datenleitungen ( $C_1 \dots C_n$ ) mit 0 bzw. 1 beschaltet.

Aus Aufwandsgründen werden bei der Realisierung von Schaltfunktionen Multiplexer verwendet, die weniger Selecteingänge haben als die Schaltfunktionen Variable.

$$a = f(e_3, e_2, e_1) = \bar{e}_1 \bar{e}_2 \bar{e}_3 + e_1 \bar{e}_2 \bar{e}_3 + \bar{e}_1 e_2 \bar{e}_3 + e_1 e_2 e_3$$

Beispiel mit 4 Variablen

$$a = f(e_3, e_2, e_1, e_0) = \bar{e}_3 e_2 \bar{e}_1 \bar{e}_0 + \bar{e}_3 e_2 \bar{e}_1 e_0 + \bar{e}_3 e_2 e_1 \bar{e}_0 + \bar{e}_3 e_2 e_1 e_0 + e_3 \bar{e}_2 \bar{e}_1 \bar{e}_0 + e_3 \bar{e}_2 \bar{e}_1 e_0 + e_3 e_2 \bar{e}_1 \bar{e}_0 + e_3 e_2 \bar{e}_1 e_0 + e_3 e_2 e_1 \bar{e}_0 + e_3 e_2 e_1 e_0$$

a:

		e <sub>0</sub>		e <sub>1</sub>		
0	0	0	0	0	0	
1	1	1	1	1	1	
0	1	1	1	1	1	e <sub>2</sub> —
1	0	0	0	0	0	e <sub>3</sub> —
C <sub>0</sub>	C <sub>1</sub>	C <sub>3</sub>	C <sub>2</sub>			

s<sub>1</sub> = e<sub>1</sub>  
s<sub>0</sub> = e<sub>0</sub>

$$C_0 = e_2 \bar{e}_3 + e_2 e_3 = e_2 \neq e_3$$

$$C_1 = C_2 = C_3 = e_2$$

b:

		e <sub>0</sub>		e <sub>1</sub>		
0	0	1	1	0	0	
0	0	1	1	0	0	
0	0	1	1	0	0	e <sub>2</sub> —
1	0	1	0	0	0	e <sub>3</sub> —
C <sub>0</sub>	C <sub>1</sub>	C <sub>3</sub>	C <sub>2</sub>			

s<sub>1</sub> = e<sub>1</sub>  
s<sub>0</sub> = e<sub>0</sub>

$$C_0 = \bar{e}_1 e_3; C_1 = 0$$

$$C_2 = e_1 + \bar{e}_3; C_3 = 1$$

Vergleich von Multiplexern und Decoder Schaltnetz Realisierung

- Decoder sind für Funktionsbündel geeignet, Multiplexer für Einzelfunktionen
- Beide für kleine bis mittlere Schaltnetzgrößen geeignet, da alle Minterme erzeugt werden

## 5.6 Lesespeicher (ROM read only memory)

Lesespeicher werden durch einen Adressdecoder adressiert Eine bestimmte Adresse erzeugt auf einer Adressleitung eine „1“. Die Adressleitung ist über ODER-Verknüpfungen mit allen Ausgängen verbunden, Diese Verbindung ist „programmierbar“.

- Verbindungen durchgeschaltet heisst: „1“ gespeichert.
- Verbindung unterbrochen heisst: „0“ gespeichert.

## 5.7 ROM-Schaltnetze

Ergeben sich dadurch, dass man die Minterme einer Schaltfunktion mit dem Adressdecoder erzeugt und auf einen der Ausgänge durchschaltet.

Beispiel :

$$a_1 = \bar{e}_1 + \bar{e}_2$$

$$a_2 = e_1 \neq e_2$$

### 5.8 Programmierbare logische Anordnung (PLA)

geeignet für Schaltungen mit einer größeren Anzahl redundanter Eingangskombinationen. Beliebige (auch minimierte) Darstellungen der Funktion sind möglich. Anzahl der Eingänge  $n$  und Anzahl der UND-Gatter  $k$  in gewissen Grenzen frei wählbar. Anzahl der ODER-Gatter hängt ab von der Anzahl der zu realisierenden Funktionen.

Typisches Beispiel:

16 Eingänge, 48 UND-Gatter, 8 Ausgänge

Anzahl der programmierbaren Verbindungen:

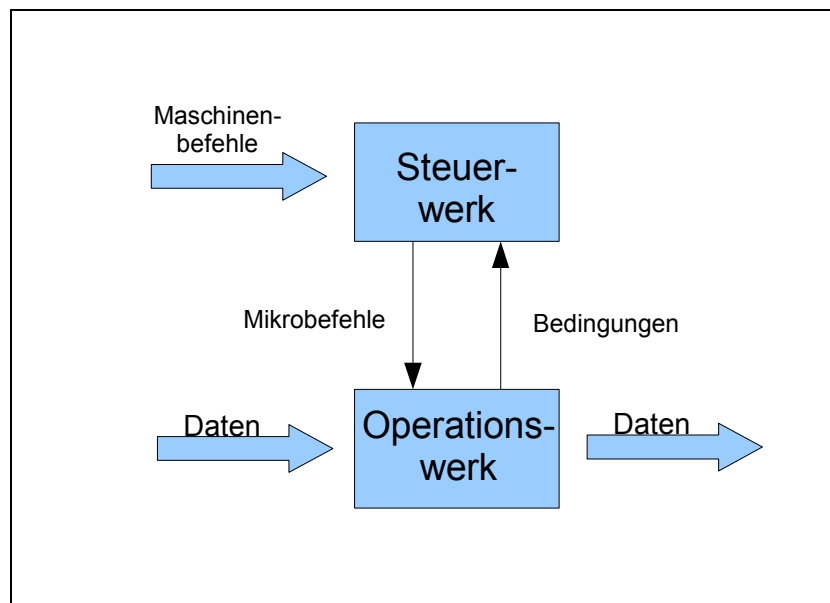
PLA  $2nk + km + 2m$ , ROM  $2^n \times m$

UND und ODER wird meist durch NAND und NOR realisiert

### 5.9 PLA-Programmierung

Es wird eine Programmier­tabelle aufgestellt (AB15). Die Ausgangs­verbindungen berücksichtigen, ob die Funktion negiert oder nicht negiert realisiert wird. Die Verwendung von negierten und nicht negierten Funktionen erfolgt mit dem Ziel, Koppel­terme (das sind Terme, die in mehreren Funktionen auftreten) auszunutzen, um so die Anzahl der benötigten UND-Terme zu minimieren. Verglichen mit den anderen Bausteinen (ROM, MUX) ist die PLA-Realisierung am flexibelsten und auch für minimierte Funktionen geeignet.

## 6. Steuerwerksentwurf



Mikrooperationen (-befehle) werden auf der Hardware direkt ausgeführt, in einem Schritt (eine Taktperiode).

Steuerwerk verarbeitet Befehle (Maschinen – oder Assemblerbefehle) erzeugt Mikrobefehle und steuert deren Abfolge durch die Verarbeitung von Bedingungen.

### 6.1 Entwurfsspezifikation

Zur Beschreibung der Funktion eines Steuerwerks. Verwendet werden Registertransfer- (RT) oder

Hardwarebeschreibungssprachen. Schaltfunktionen wären möglich, sind jedoch zu detailliert.

Binärinformation: als Codierung von Zahlen, Buchstaben ... die je nach Typ unterschiedlich verarbeitet werden kann.

Operationen: sind entweder Transfers (Registerverschiebungen) oder Verarbeitung.

Steuerfunktionen: legen fest, wann (zeitlich und logisch) Operationen auszuführen sind. Verarbeiten Zeit- und Bedingungs-signale. Zeitsignale entstehen durch die Taktung des Steuerwerks. Bedingungs-signale entstehen bei der Verarbeitung im Operationswerk.

## 6.2 Registertransfer

### Bustransfers

Bussysteme stellen eine ökonomische Alternative zu dedizierten Verbindungen dar. Es handelt sich um Verbindungen, die mit Multiplexern und Decodern geschaltet werden. Durch die Quellregisterauswahl Q1Q0 und die Zielregisterauswahl Z1Z0 werden Verbindungen temporär geschaltet.

### Speichertransfers

Befehle und Daten werden in einem Speicher mit wahlfreiem Zugriff gehalten. Der Zugriff erfolgt über ein Speicheradressregister (SAR) und ein Speicherdatenregister (SDR). Lesevorgang (L) auf einer Speichereinheit (SE): L:SDR SE

Bei einem Buszugang zum Speicher, ist es möglich, Speicherdatenregister und -adressregister starr zu koppeln, indem dieselbe Bitkombination für den Adressbus-MUX und für den Datenbus DEC verwendet wird.

### Verarbeitende Operationen

Beim Transfer zwischen Registern werden die Registerinhalte modifiziert, z.B.

arithmetische Operationen:

$A \leftarrow A + B$  (addition)

$A \leftarrow A - 1$  (decrement)

logische Operationen:

$A \leftarrow \neg A$  (negation)

$A \leftarrow A \times B$  (konjunktion)

### Bedingungsanweisungen

Nicht unbedingt erforderlich, wegen Analogie zu höheren Programmiersprachen üblich.

T: IF (Bedingung) THEN (Mikrooperation[en])

ELSE (Mikrooperation[en])

Beispiel: siehe Arbeitsblatt 26

## 7. Steuerlogik

Damit wird ein Schaltwerk bezeichnet, das Maschinenbefehle verarbeitet und daraus Mikrobefehle erzeugt. Normales Schaltwerk mit „vielen“ Zuständen und Variablen. Ziel ist eine regulär strukturierte Standardhardware.

### 7.1 Steuerformen

(1 aus n) Steuerung ordnet jedem Mikrobefehl ein Flipflop fest zu. Ohne externe Eingänge erhält man ein Schieberegister. Bei Verbindungen von Eingang und Ausgang einen Ringzähler.

- Register-Decoder-Steuerung  
Mikrobefehle im Zustandsregister kompakt gespeichert und für die Ausführung decodiert. Ohne externe Bedingungen ergibt sich ein Zähler.
- PLA-Steuerung  
Fast Steuerung und Decoder zusammen für die Programmierung in einem PLA
- Mikroprogrammsteuerung  
Mikroprogramm Speicher sind normalerweise Lesespeicher (ROM). Durch die Maschinenbefehle werden Mikrobefehlsfolgen adressiert und durch Auslesen ausgeführt

## 7.2 Entwurf eines Steuerwerks

- Schritte:
1. Spezifikation der Aufgabe
  2. Festlegung einer Hardwarestruktur
  3. Algorithmenentwurf
  4. Entwurf der Steuerung

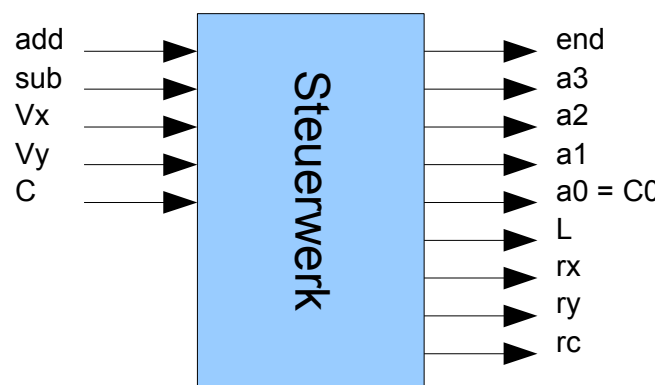
Beispiel: Addition vorzeichenbehafteter Festpunktzahlen

1. Addition, negative Zahlen bestehen aus Vorzeichen und Betrag, Zahlen haben gleiche Länge, Ergebnis sei vorzeichenbehaftete Festpunktzahl
2. Konfiguration soll umfassen: 2 Register X und Y, Z Flipflops für Vorzeichen  $V_x$ ,  $V_y$ , ein Überlauf-Bit C und eine ALU

Vorausgesetzt wird, dass X, Y,  $V_x$ ,  $V_y$  durch eine separate Operation vorab geladen werden. Befehle sollen mit add bzw. Sub bezeichnet werden. Eine Steuervariable zeigt an, dass ein Maschinenbefehl beendet ist und ein neuer begonnen werden kann.

3. Algorithmenentwurf – Subtraktion erfolgt durch Addition des 2-Komplements, C macht eine Aussage über positives oder negatives Ergebnis, das negative Ergebnis erscheint als 2-Komplement und ist umzuformen

Blockschaltbild Steuerlogik



4. Entwurf der Steuerung – Zustandsübergangsgraph: Mikrooperationen des Algorithmus werden Zustände zugeordnet. (Es gibt verschiedene richtige Lösungen)

## Realisierung

Versuch der Umsetzung des Steuerschaltwerks in eine standardisierte Lösung

- Gewählte Methode: 1 Flipflop pro Zustand, realisiert mit D-Flipflop, die mit dem zu speichernden Signal angesteuert werden
- Jedem Mikroschritt wird ein Flipflop fest zugeordnet, seine Ansteuerung kann direkt aus dem Graphen entnommen werden: z.B.

$$Z0_{t+1} = \text{add}(n)_t \times \text{sub}(n)_t \times z0_t + z3_t + c_t \times z5_t + z7_t$$

$$Z1_{t+1} = \text{sub}_t \times z0_t$$

$$Z2_{t+1} = \text{add}_t \times z0_t + z1_t$$

$$Z3_{t+1} = (V_{xt} \equiv V_{yt}) \times z2_t$$

$$Z4_{t+1} = (V_{xt} \neq V_{yt}) \times z2_t$$

$$Z5_{t+1} = z4_t$$

$$Z6_{t+1} = z5_t \times c(n)_t$$

$$Z7_{t+1} = z6_t$$

- Ausgangsfunktionen (keine Bedingungen – bereits verarbeitet)

$$\text{end}_t = z0_t$$

$$a3_t = z6_t$$

$$a2_t = z4_t + z6_t$$

$$a1_t = z3_t + z6_t$$

$$a0_t = z4_t + z7_t$$

$$L = z3_t + z4_t + z6_t + z7_t$$

$$r_x = z7_t$$

$$r_y = z1_t$$

$$r_c = z5_t$$

### 7.3 Entwurf einer Mikroprogrammsteuerung

Steuerinformation wird nicht durch ein Steuerschaltwerk erzeugt, sondern zuvor berechnet und in ein Mikroprogramm Speicher abgelegt.

Die Adressen entsprechen den Zuständen der Hardwaresteuerung. Die Steuerinformation bildet ein so genanntes Mikrosteuerwort, das für bestimmte Mikrooperationen spezifisch ist. Die Länge des Mikrosteuerwortes wird festgelegt durch die Steueranforderung. (Steuervariable), Statusbits, Adresslänge.

Der Mikrosteuerteil (Steuervariablen) kann aus der Hardwaresteuerung übernommen werden (die Steueranforderungen unverändert) Folgeadresse steht im Adressteil oder ergibt sich durch Inkrementierung.

### 7.4 Prozessor und Steuerwerksstrukturen

Speicher muss so dimensioniert werden, dass er alle möglichen Mikroprogramme aufnehmen kann. Steuervariablen müssen alle möglichen Steuerfolgen zu realisieren gestatten. Alle notwendigen Statusbits sind zur Verfügung zu stellen.

Prozessor arbeitet mit 7 allgemeinen Registern die als Quelle oder Senke verwendet werden können. Die Auswahl erfolgt durch Multiplexer (A,B,Z). Diese können zusätzlich eine direkte Datenein- und ausgabe steuern. Die Steueranforderungen für ALU und Shifter sind 7 Bit erzeugt werden 8 Statusbits. Insgesamt ergibt sich für den Prozessor eine Steuerwortbreite von 16 Bit. Die Mikrosteuereinheit kann 64 Wort adressieren (6 Bit Adressteil), 16 Bit für den Prozessor, 3 Bit Statusbitauswahl, 1 Bit Adressauswahl => Länge eines Mikrosteuerwortes von 26 Bit.

### 7.5 Erstellen eines Mikroprogrammes

Beispiel: Die Anzahl der 1-Werte im Register R1 soll gewählt werden, das Ergebnis soll im Register R2 abgelegt werden. Die Startadresse des Mikroprogramms sei 8.

R2 und C auf NULL gesetzt. Statusbits N wird abgefragt für  $R1 < 0$ ?

$R2 < 0$ : Realisiert durch einen Shifterbefehl Ausgang=0

$R1 = 0$ ?: R1 wird über ALU und Shifter nach R1 übertragen. Automatisch wird  $C=0$  erzeugt. Danach Abfrage von N.

### 7.6 Mikroprogramm-Adressfortschaltung (Mikro-Sequenz)

dient dazu, im Rahmen der Mikroprogrammierung Unterprogramme zu benutzen, um häufiger

benutzte Programmteile nur einmal speichern zu müssen.

a <sub>1</sub> a <sub>0</sub>	INC / S	DEC / L	Operationen	Benutzung
00	0	0	MAR <- MAR+1	
01	0	1	MAR <- TOS	TOS = Top od Stack
10	0	0	MAR <- INA	INA interne Adresse
11	0	0	MAR <- EXA	EXA externe Adresse
xx	1	0	TOS <- MAR +1	Adresse speichern

## 8. Systemaufbau

Rechnerorganisation bezeichnet die Anordnung von Rechnerkomponenten, ihre Verbindungen und die zugehörige Steuerung. (durch Hardware, Mikroprogramme, Betriebsprogramme)

### 8.1 Rechnerkomponenten

als Realisierung von Grundanforderungen an ein Rechnersystem (5 Komponenten bzw. von Neumann-Rechner)

1. Eingabe für Operanten und Befehle
2. Speicher für Operanten, Befehle und Ergebnisse (in beliebiger Reihenfolge)
3. Operationswerk (Rechnerwerk, Verarbeitungseinheit) für die Ausführung von Operationen mit Operanden aus dem Speicher
4. Ausgabe von Ergebnissen
5. Steuerung durch Abfrage und Setzen von Bedingungen

Ein-/Ausgabe Lesen und Schreiben von Benutzungsprogrammen und Daten und Datenpufferung, um Unterschiede in den Arbeitsgeschwindigkeiten auszugleichen und eine Datenwandlung, weil unterschiedliche Formate verwendet werden (ASCI-Zeichen für Tastatur, prüfbare Codes für Übertragungen, Wortorientierungen des Arbeitsspeichers)

Steuerung enthält Befehle aus dem Speicher und erzeugt daraus Steuersignale, um andere Komponenten zu Aktionen zu veranlassen. Statusinformation wird abgefragt und beeinflusst die Steuersignale.

Speicher dient als Sammelstelle für die Daten die im System bewegt werden. Ist nach Informationseinheit organisiert (Bit, Byte, Wort, ...) die mit einer numerischen Adresse angesprochen werden.

### Organisationsfragen

1. Adressierung
  1. Anzahl der Adressen
    1. Null-Adress-Maschinen (Stack-Computer)
    2. Ein-Adress-Maschinen (Akkumulator)
    3. Mehr-Adress-Maschinen (Multiregister-Prozessoren)
  2. Adressierungsformen
    1. absolute Adressierung
    2. relative Adressierung
    3. virtuelle Adressierung
2. Ablaufsteuerung
  1. fest verdrahtet
  2. mikroprogrammiert
3. Ein/Aus Steuerung

Weitere Organisationsprobleme entstehen bei Parallelarbeit zur Durchsatzsteigerung und durch den Benutzerkomfort:

- Forderung nach Entlastung des Benutzers von maschinenspezifischen Detailproblemen
- Effizienzsteigerung erfordert Adressverteilung und Speicherorganisation

Betriebssystem ist ein Programmsystem, welches eine virtuelle Maschine definiert, die dem Benutzer zur Verfügung steht und diese mit Hilfe der gegebenen Hardware realisiert.



organisieren ist.

Forderung: Information schreiben und lesen, Informationseinheiten eindeutig adressieren.  
Bits, Bytes, Worte, Blöcke, abhängig von der Zugriffszeit und dem Medium.

Kennzeichen: Zugriffszeit (ns ... min)  
Zykluszeit (ns ... min) Zeit zwischen zwei Zugriffen  
Zugriffsart (wahlfrei - Zugriffszeit für alle Zellen gleich)  
(sequentiell - lesen aller Inhalte zwischen Position der Schreib-  
Leseinrichtung und gewünschter Position)  
(zyklisch - Kombination von wahlfrei und sequentiell)

Mikroprogramm Speicher: Lesespeicher als Halbleiterspeicher kurzer Zugriffszeit

Pufferspeicher: Register im Prozessor, Cache-Speicher zwischen Prozessor und Arbeitsspeicher

Arbeitsspeicher: in Form von Halbleiterspeichern

Hintergrundspeicher: in Form von magnetomotorischen Schichtspeichern oder optoelektronische Speicher

Archivspeicher: Bandkassetten robotergesteuert, Jukeboxes

historisch: 1. *Einzelbenutzerbetrieb* Programme wurden mit den Adressen ausgeführt, mit denen sie im Speicher standen.  
2. *Mehrbenutzerbetrieb* verschiebliche Programme durch relative oder virtuelle Adressierung

### 8.3.1 Direkte und indirekte Adressierung

direkt: Programmadressen und Speicheradressen stimmen überein.

indirekt (relative): ergibt sich die Speicheradresse aus der Programmadresse und der Basisadresse  
Basisadressregister wird durch den Benutzer oder des Betriebssystems verwaltet. Programme sind verschiebbar im Speicher. Mehrprogrammbetrieb ist möglich.

Randbedingungen:

- vor der Ausführung müssen Programme und Daten vollständig im Arbeitsspeicher stehen.
- Adressierung bezieht sich nur auf den Arbeitsspeicher.
- Anwender muss Einlagerungen von Hintergrundspeichern selbst veranlassen.
- Belastung ist für das Betriebssystem gering

### 8.3.2 Virtuelle Adressierung

die Verteilung des Adressraumes hängt davon ab, wie die momentane Relevanz der Daten in diesem Teilbereich ist.

Virtuelle Adressen enthalten Pagedescriptor (Seitennummer), Segmentdescriptor optional, Adresse relativ zum Seitenanfang (Displacement)

Einlagerung von Hintergrundspeicher in den Arbeitsspeicher erfolgt auf Anforderung (Demand paging)

Ersetzungstrategien: dienen dazu, festzulegen, welche Teile des Arbeitsspeichers bzw. des Assoziativspeichers ausgelagert werden sollen.  
FIFO first in first ot, SC second chance  
LRM least recently used

### 8.4 Ein-Ausgabe-Organisation

betrifft die Kommunikation eines Rechners mit seiner Umgebung z.B. Mit EA-Geräten. Ein-Ausgabe ist mit der Behandlung von Unterbrechungen (interrupts) verbunden. Sie stellen eine Mitteilung an die Rechnersteuerung dar, den Unterbrechungswunsch zu behandeln.

Unterbrechungen können auch im Zusammenhang mit Fehlerbedingungen auftreten, z.B. Überlauf oder ungültige Speicheradressen. Diese bezeichnet man als interne Untersuchungen im Gegensatz

zu den externen.

Die externen Unterbrechungen betreffen vor allem die Kommunikation zwischen einem „schnellen“ Prozessor und Geräten die im Vergleich dazu sehr langsam sind, wie z.B. Plattenlaufwerke.

Die Kommunikation erfolgt asynchron.

#### 8.4.1 Unterbrechungssystem

Eingehende Unterbrechungen werden in einem Register UBR (Unterbrechungsregister) erfasst. 1 Bit pro möglicher Unterbrechung. Ein Maskenregister legt fest, welche Unterbrechungen in Abhängigkeit vom Systemstatus zulässig sind. Bei mehreren zulässigen Unterbrechungen entscheidet eine hardwaremäßige implementierte Prioritätslogik.

#### 8.4.2 EA-Organisationen

Schritte:

- Prüfen, ob ein bestimmtes Gerät verfügbar ist.
- Wenn ja, Anstoß des Geräts
- Datenübertragung
- Rückkehr in den Grundzustand

Prüfen durch ständige Abfrage oder Unterbrechungssignal des Geräts an den Prozessor.

Anstoß ist nur bei Geräten erforderlich, die nicht ständig im Übertragungsmodus sind.

Übertragung programmiert programmiert, gepuffert, mit diskreten Speicherzugriff (DMA, cycle stealing) heißt das die Übertragung vollständig unter der Kontrolle des Zentralprozessors abläuft, das Programm zur Unterbrechungsbehandlung steht im Arbeitsspeicher, d.h. Viele Speicherzugriffe, retten des vollständigen Prozessorstatus => langsam, aufwendig.

gepuffert durch zusätzliche Logik und Datenwege wird die Belastung durch die EA für den (Hardware-gesteuert) Zentraöprozessor reduziert. Unterbrechungsprozedur erfordert keine Befehlszugriffe

DMA (cycle stealing) stellt für die Puffer-Steuer-Folge einen eigenen Zugang zum Arbeitsspeicher bereit, der immer dann zugreift, wenn vom Zentralprozessor keine Zugriffe erfolgen.

#### 8.4.3 Kanaltypen

Kanal ist eine Steuereinheit, die speziell auf die Behandlung von EA-Abläufen zugeschnitten ist.

Typen:

- Selektorkanal  
für schnelle EA-Geräte, solange verbunden, wie Übertragung erfolgt
- Multiplex-Kanal  
bedient langsame EA-Einheiten im Zeitmultiplex