MOTOROLA
MC 6809

Adressierungsarten

Otto-von-Guericke-Universität Magdeburg

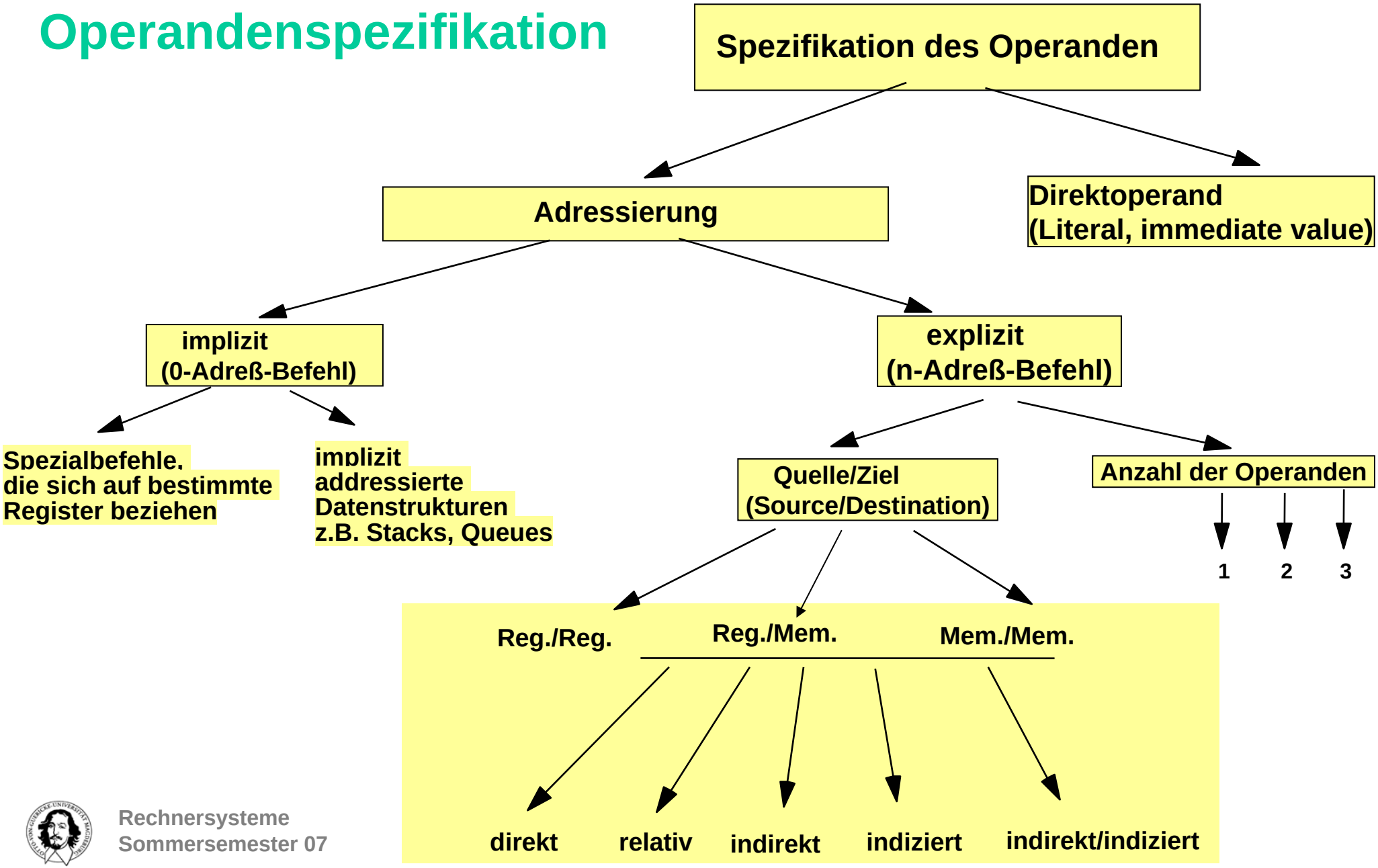
Ziele beim Entwurf eines Adressierungsmechanismus

- großer Adreßraum
- kleines Adreßfeld in der Instruktion
- Komfort für den Programmierer
- positionsunabhängiger (verschieblicher) Code

Lösung des Zielkonflikts zwischen Größe des AR und Größe des AF: Berücksichtigung von Lokalität der Berechnungen

Vereinfachung der Adressierung einfacher Datenstrukturen (Listen, Arrays)

Möglichkeiten der Operandenspezifikation



6809 Register Adressierung

Befehle: TFR, EXG,

Speicherplatz

m	OPCODE
m+1	Reg. Spec.
m+2	
m+3	
m+4	

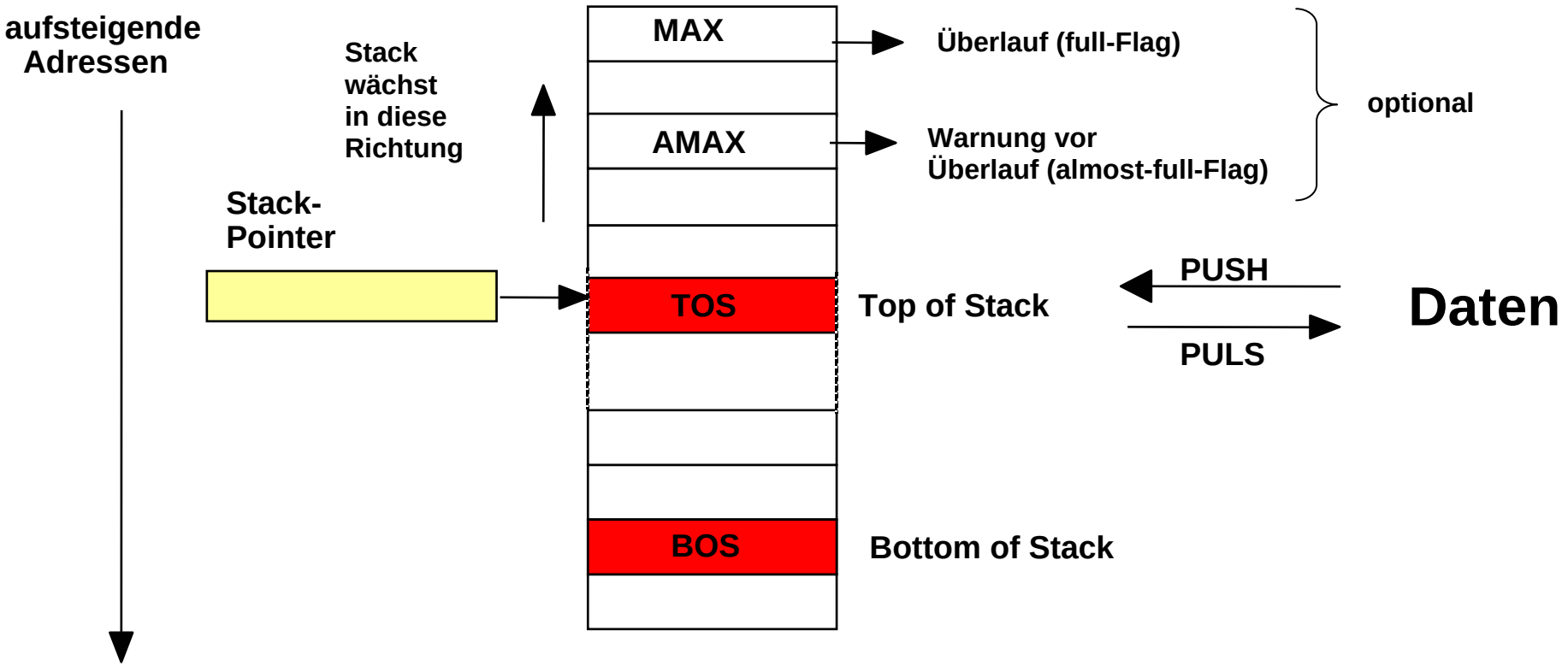
Postbyte Definition bei TFR und EXG:

7	4	3	0
Reg. 1 (S)		Reg. 2 (D)	

S: Source
D: Destination

Register	Code (Bin)	Hex
D	0000	0
X	0001	1
Y	0010	2
U	0011	3
S	0100	4
PC	0101	5
A	1000	8
B	1001	9
CCR	1010	A
DP	1011	B

Stack (Stapel, Keller, LIFO-Queue)



PUSH:

SP=MAX: Exception ‘full‘
SP← SP-1
Mem[SP]← Register
SP >AMAX: Signal ‘almost full‘

PULS:

SP=BOS: Exception ‘empty‘
Register← Mem[SP]
SP← SP+1

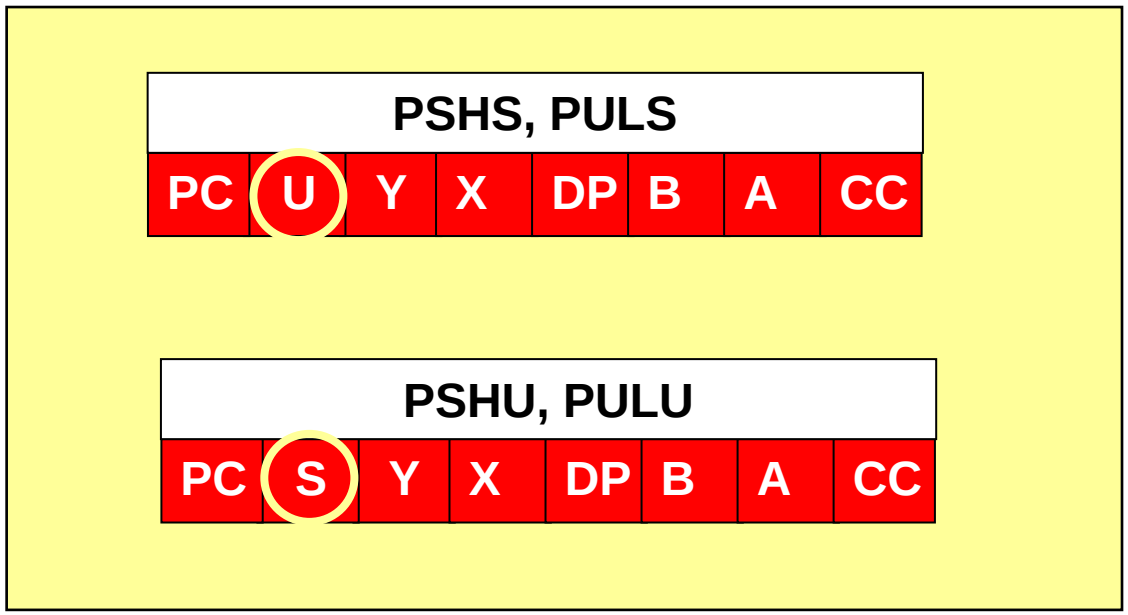
6809 Stack Adressierung

Befehle: PSHU, PSHS, PULU, PULS

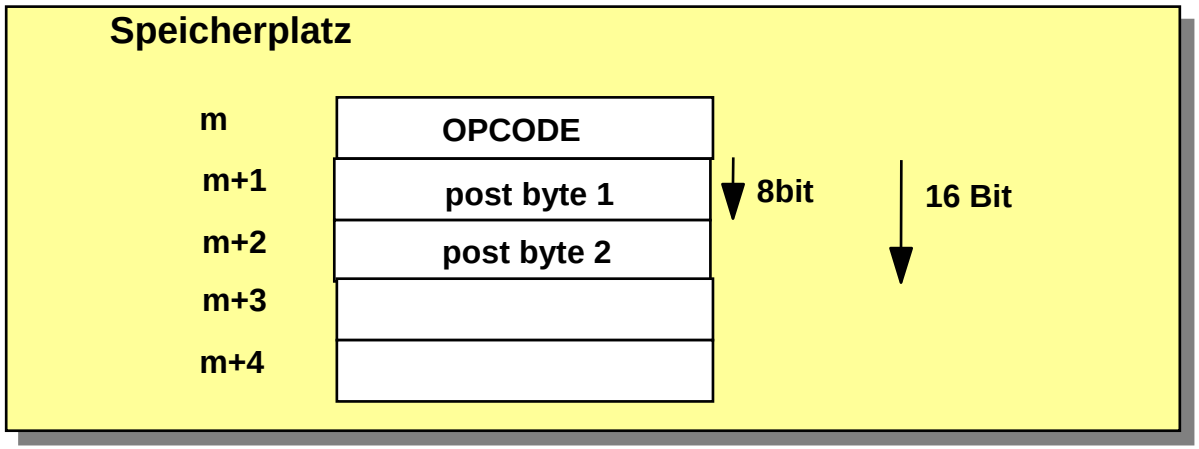
Speicherplatz

m	OPCODE
m+1	Register List.
m+2	
m+3	
m+4	

Post-Byte Definition bei PSHx und PULx



6809 Direktoperanden (unmittelbare Adressierung, engl.: immediate)



Beisp. : ADDA #$95

m	8B
m+1	95

PC = PC+2

ADDD #$1295

m	C3
m+1	12
m+2	95

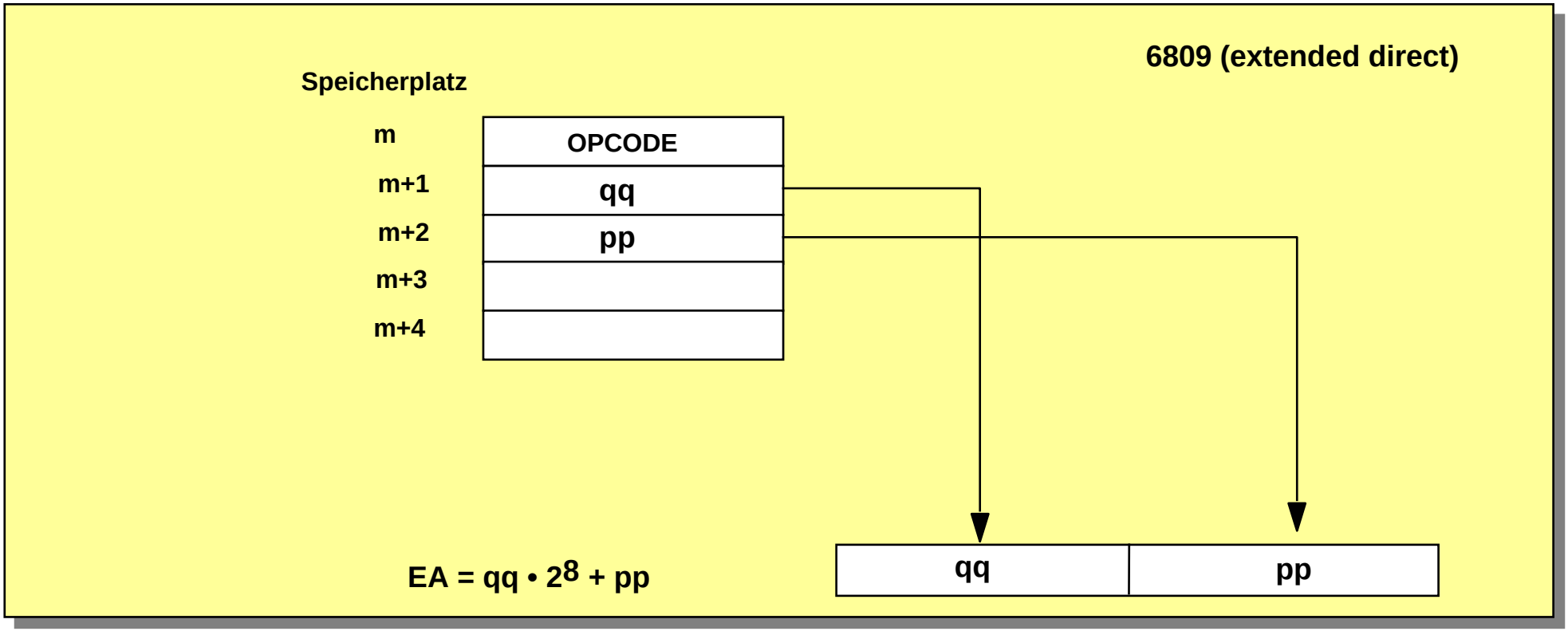
PC = PC+3

LDS #$ABCD

m	10
m+1	CE
m+2	AB
m+3	CD

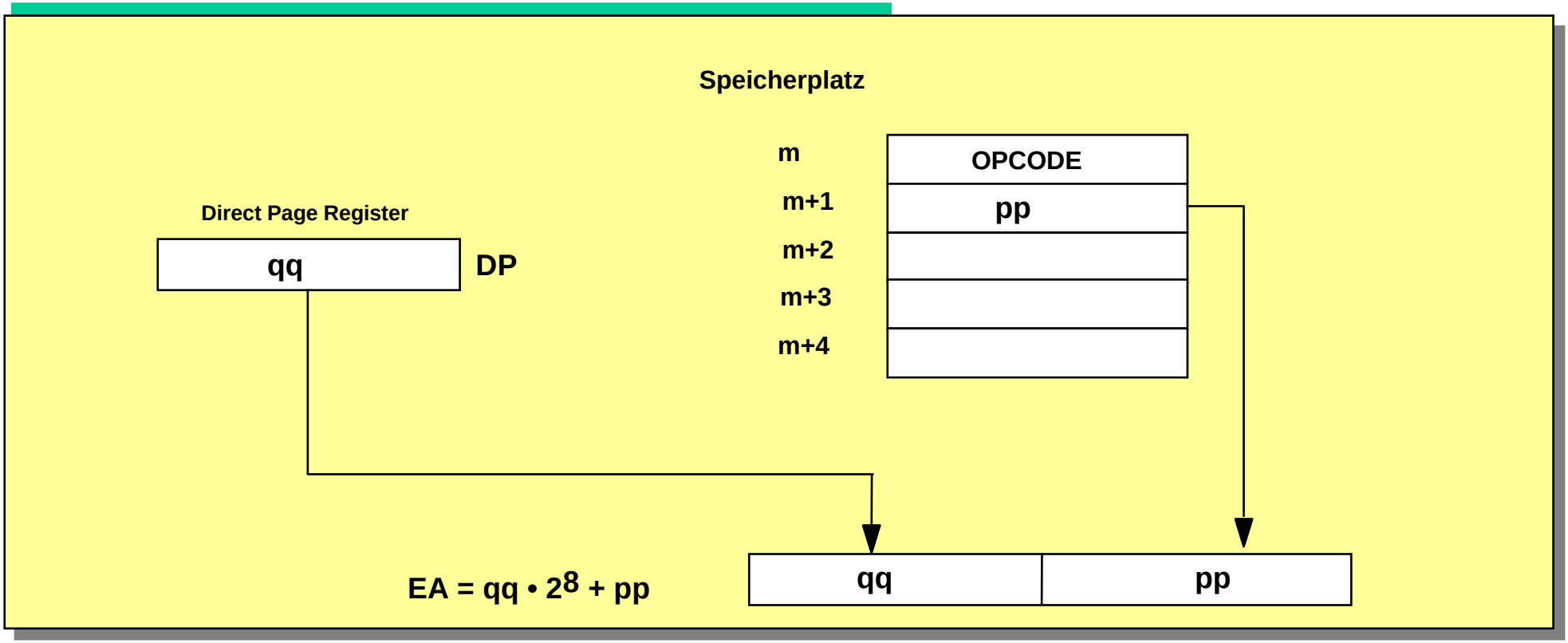
PC = PC+4

Direkte oder absolute Adressierung



Die **effektive Adresse EA** wird durch Konkatenation der beiden Post Bytes nach dem OPCODE gebildet. Das erste Post Byte (qq) gibt das höherwertige Byte, das zweite Postbyte (pp) das niederwertige Byte der EA an.

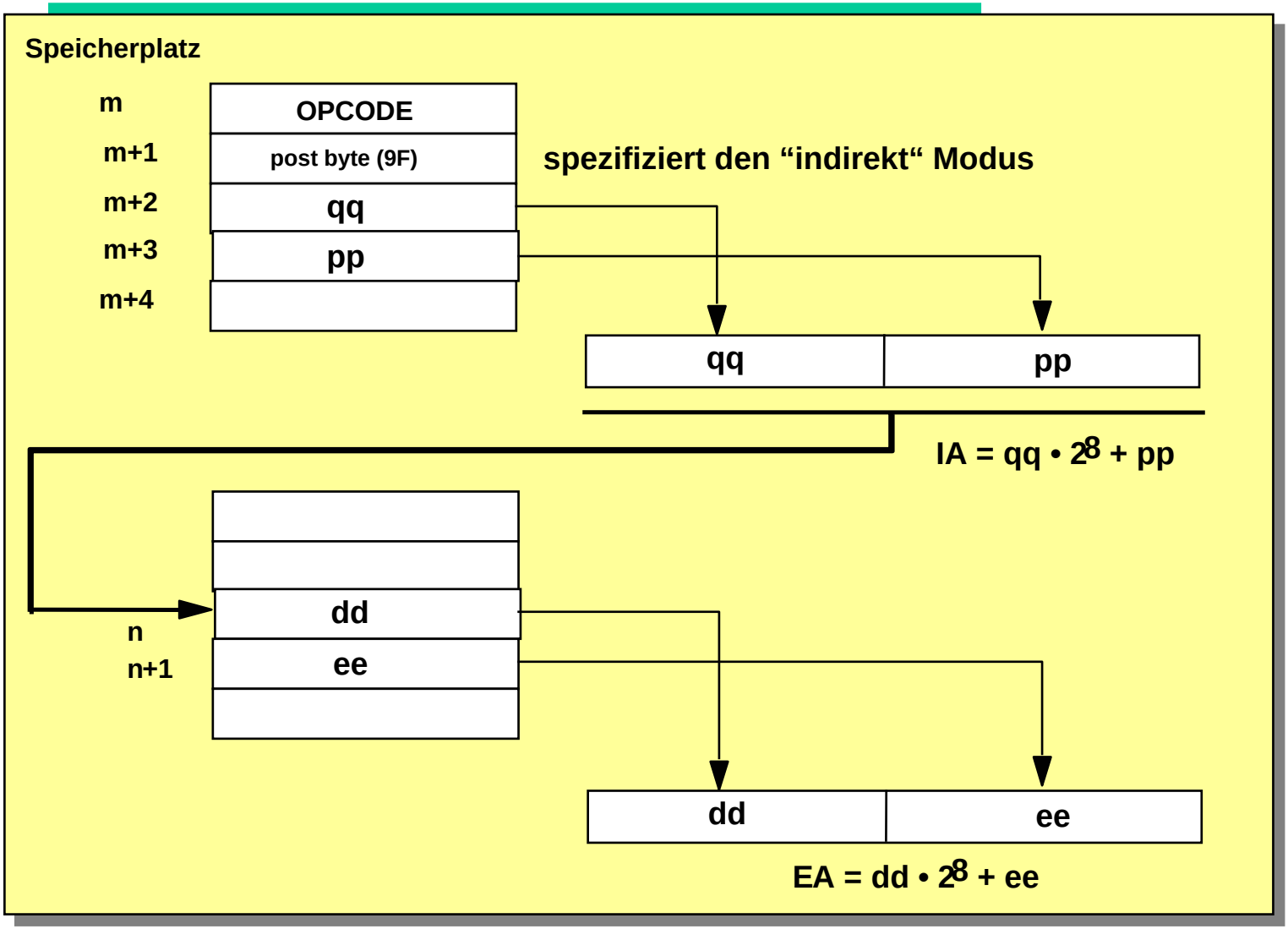
6809 (base page direct) Adressierung über Basisregister

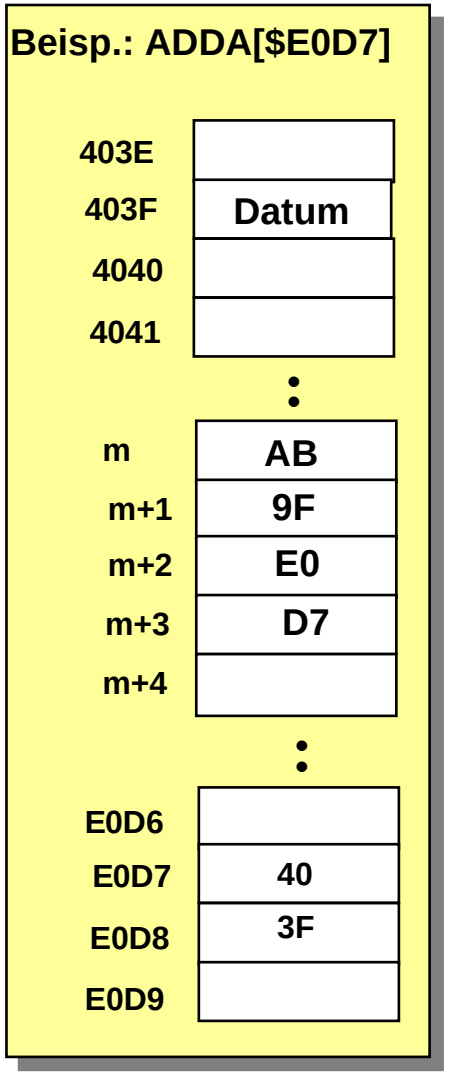


Die effektive Adresse EA wird durch Konkatenation einer kurzen Adresse nach dem OPCODE mit dem Inhalt des Direct Page Registers (DP) gebildet. Die kurze Adresse gibt das niederwertige Byte, der Inhalt des DP das höherwertige Byte der EA an.

Das DP kann dynamisch durch einen Transferbefehle z.B. TFR A, DP geladen werden oder durch eine Assemblerdirective SETDP gesetzt werden.

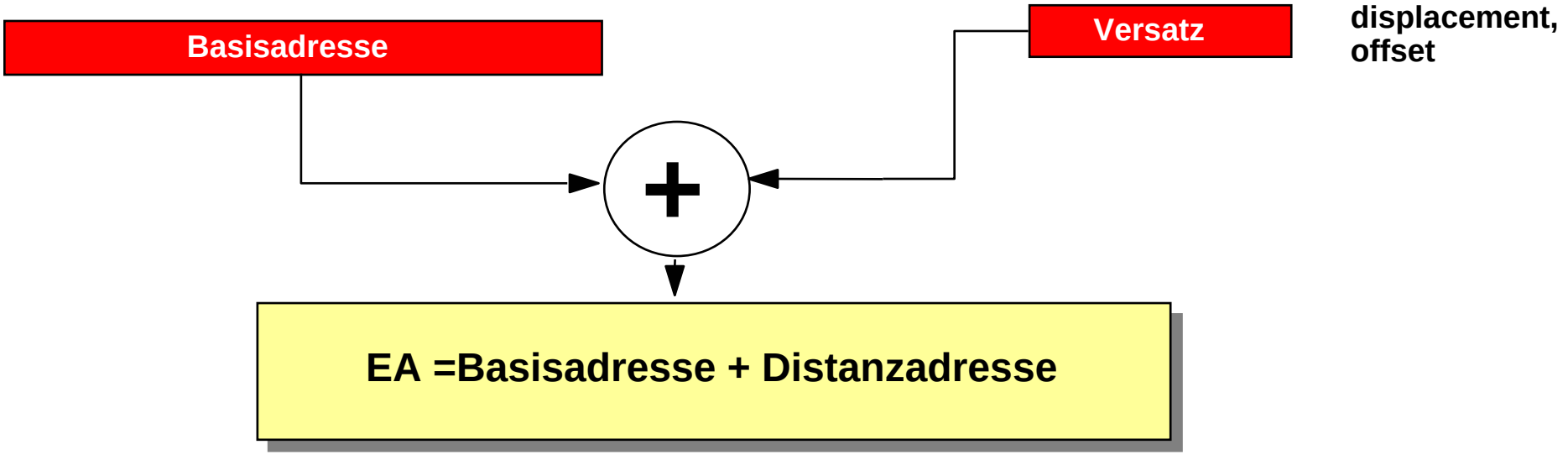
6809 (extended indirect) Indirekte Adressierung





Die dem OPCODE folgende Adresse gibt nicht an, wo sich die Daten für den Befehl befinden, sondern die Adresse, die auf die Daten zeigt (Unterstützung von Zeigern).

Indizierte Adressierung (Indexed Adressing)



Wo steht die Basisadresse ?

- **Register: Indexregister**
- **Speicher: indirekte Adressierung**

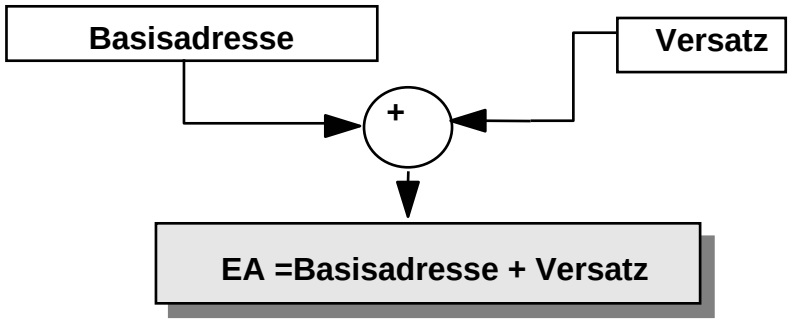
Wo steht der Versatz ?

- **Speicher (Literal)**
- **Register**
- **Autoincrement /Autodecrement**

Indizierte Adressierung beim MC 6809:

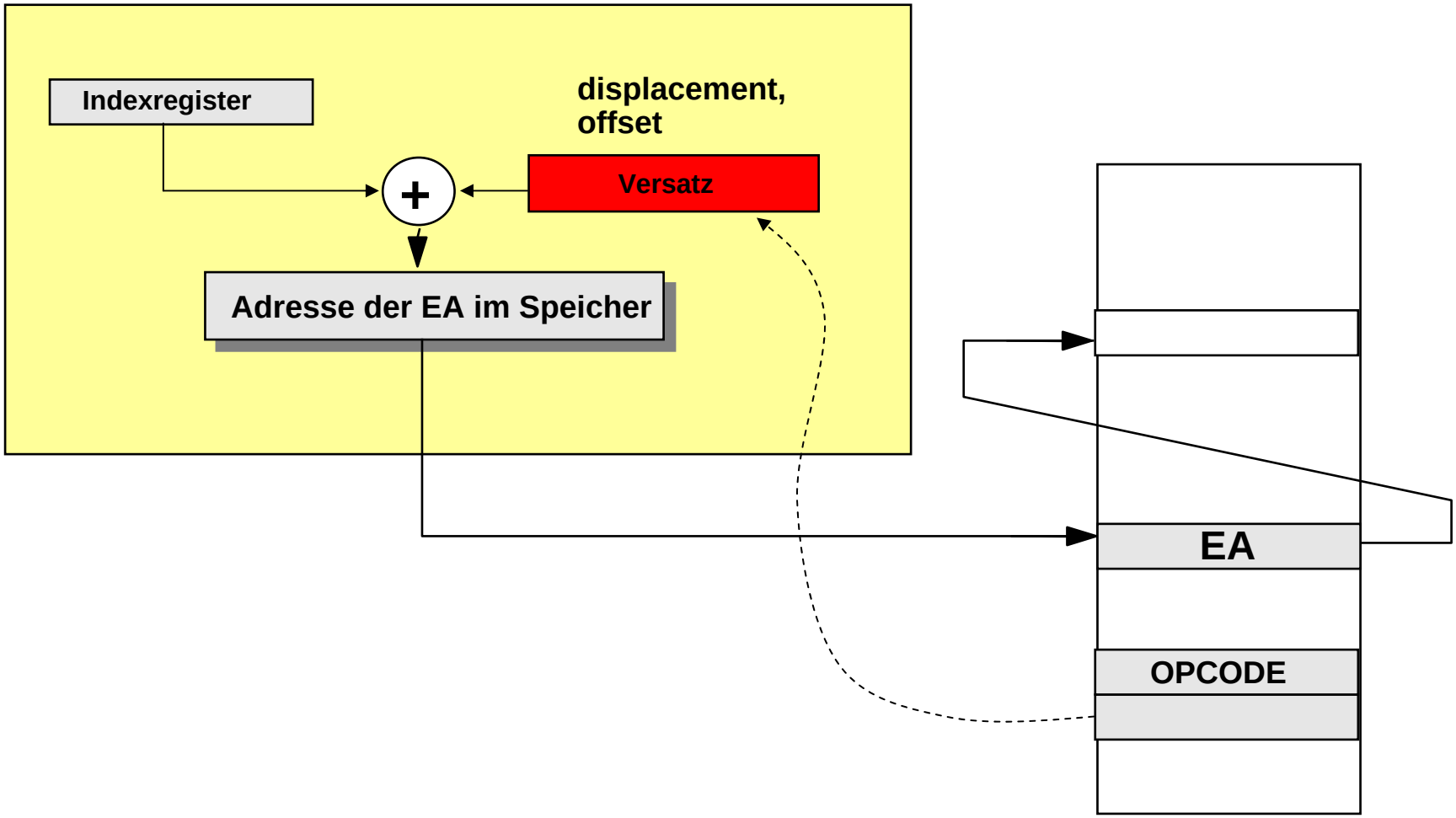
- Zero offset
- 5-Bit offset
- Long (16 Bit) constant offset
- Accumulator offset (Versatz kann verändert werden)
- Autoincrement
- Program Counter Relativ
- Relative Branches
- Indexed Indirect

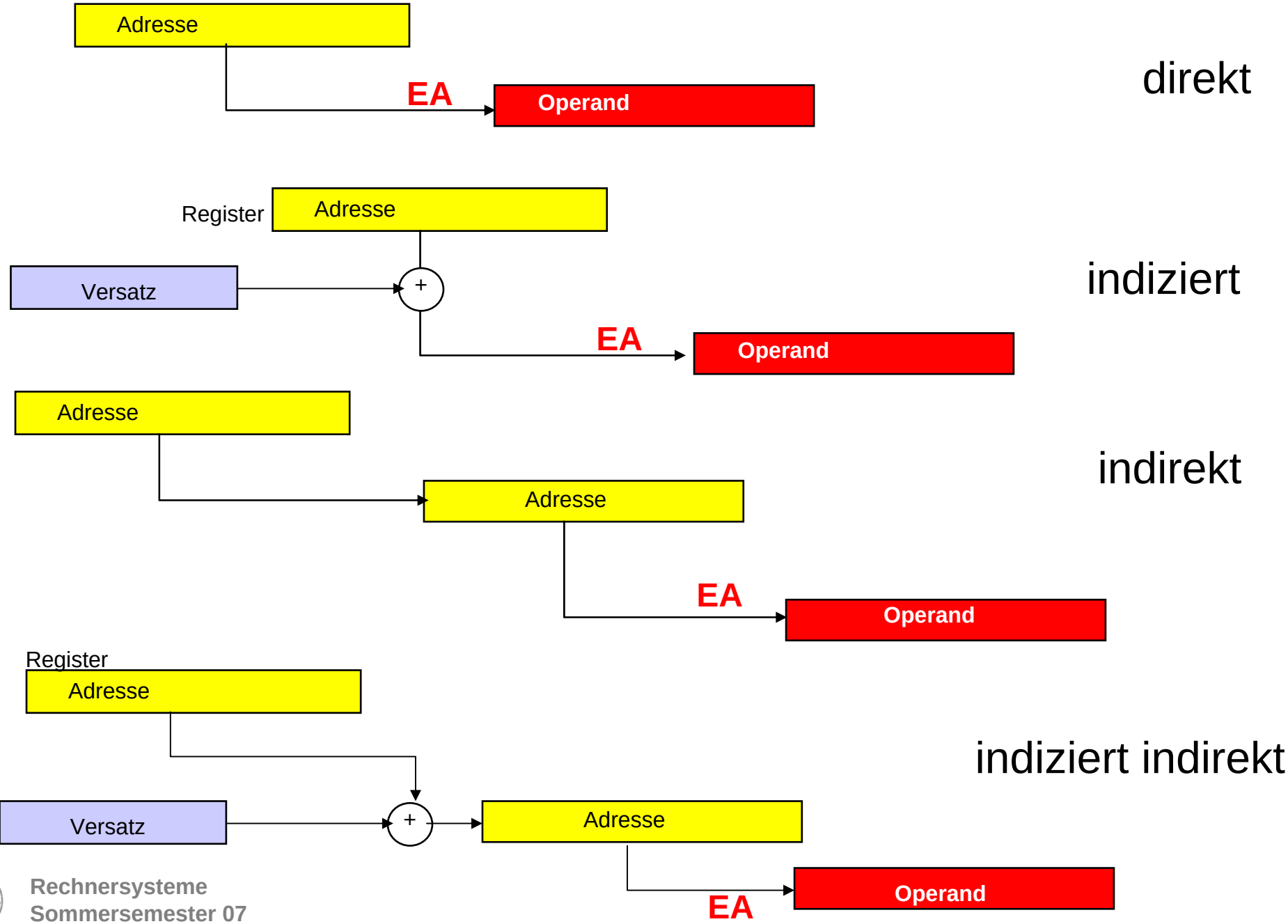
Grundformen der indizierten Adressierung



Basisadresse	Versatz	Bezeichnung
(Index-) Register X, Y, U, S, PC	Literal (kurz 5 Bit) Speicher (lang 8/16 Bit)	Adressierung mit konstanter Distanzadresse vom Indexregister “Constant Offset Mode “
(Index-) Register X, Y, U, S	Register A, B, D	Adressierung mit Distanzadresse aus dem Accumulator A, B oder D “Accumulator Offset“
(Index-) Register X, Y, U, S	fest ±1, ±2	Autoincrement / Autodecrement POST increment / PRE decrement

Grundform der indiziert indirekten Adressierung

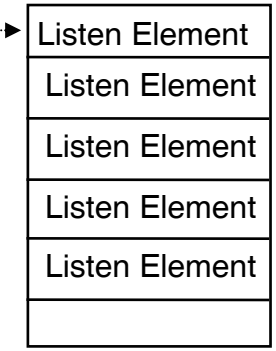



Adresse
EA
Operand
direkt
Register
Adresse
Versatz
+
EA
Operand
indiziert
Adresse
Adresse
EA
Operand
indirekt
Register
Adresse
Versatz
+
Adresse
EA
Operand
indiziert indirekt

```
        LDX     HEAD_OF_LIST
COMP    CMPX    End_lst
        BHI     Fin
        LDA     ,X+
        CMPA    #77
        BNE     COMP
        •
        •
        •
End_lst FDB     $ABCD
```

Index Reg.

Indiziert mit (Post-) Autoincrement

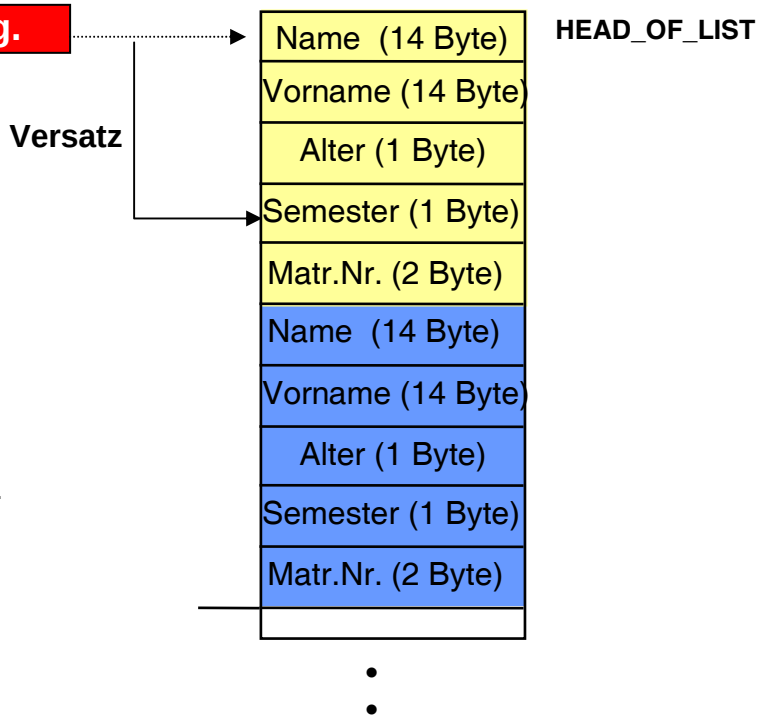



```
        LDX     HEAD_OF_LIST
        LDA     Sem_O
COMP    LDB     A ,X
        CMPB    S_zahl
        BHI     OUTPUT
        LEAX    RL, X
        CMPX    End_lst
        BHI     Fin
        BRA     COMP
        •
        •
        •
RL      FCB     32
Sem_O   FCB     29
S_zahl  FCB     12
End_lst FDB     $ABCD
```

Index Reg.

Indiziert mit Register A Offset
Falls Semesterzahl >12:
Ausgabe: Name, Vorname
Addieren der Record-Länge*
Ende der Liste erreicht?

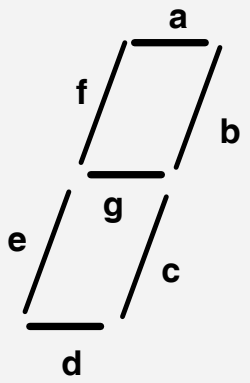
* da es keinen Additionsbefehl für das Indexregister X gibt, wird LEAX verwendet.



```
SP. Adr  OPC OP          SP.M  MNE.   OP          Kommentar
A000     3F              CLRB                     setzt den Fehlercode : löschen des Displays
A001     96  A0          LDA   $A0                Wert in den Acc. A holen
A003     81  09          CMPA  #9                 ist der Wert eine Ziffer 0…9 ?
A005     22  05          BHI   DONE               wenn A > 9 bleibt der Fehlercode gesetzt
A007     8E  AF00        LDX   #SSG               Lade X  mit dem Beginn der Conversionsliste “SSG“
A00A     E6  86          LDB   A, X               Lade 7-Segment Muster in Acc A
A00C     D7  …… DONE     STB   MMSEG              Abspeichern auf ein Ausgabegerät
A010     39              RTS                      Return from Subroutine

AF00             SSG     FCB   $3F, $06, $5B, $4F, $66
AF05                     FCB   $6D, $7D, $07, $7F, $6F

……               MMSEG …….   ……               Adresse des Ausgabegeräts
```

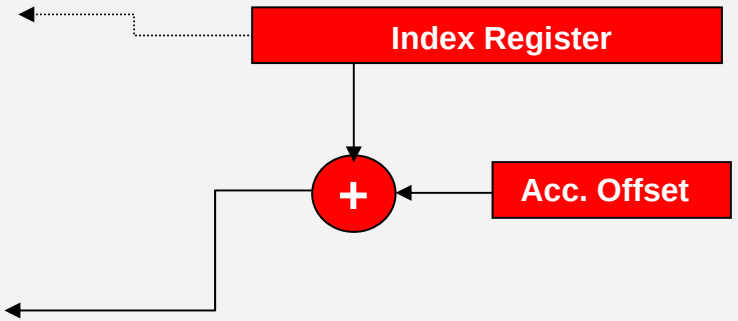


7	6	5	4	3	2	1	0
0	g	f	e	d	c	b	a

Zuordnung von Segmenten zu Bitpositionen im Ausgabewort

7-Segment Code:

0	3F
1	06
2	5B
3	4F
4	66
5	6D
6	7D
7	07
8	7F
9	6F



6809 Indizierte Adressierung (Indexed Adressing)

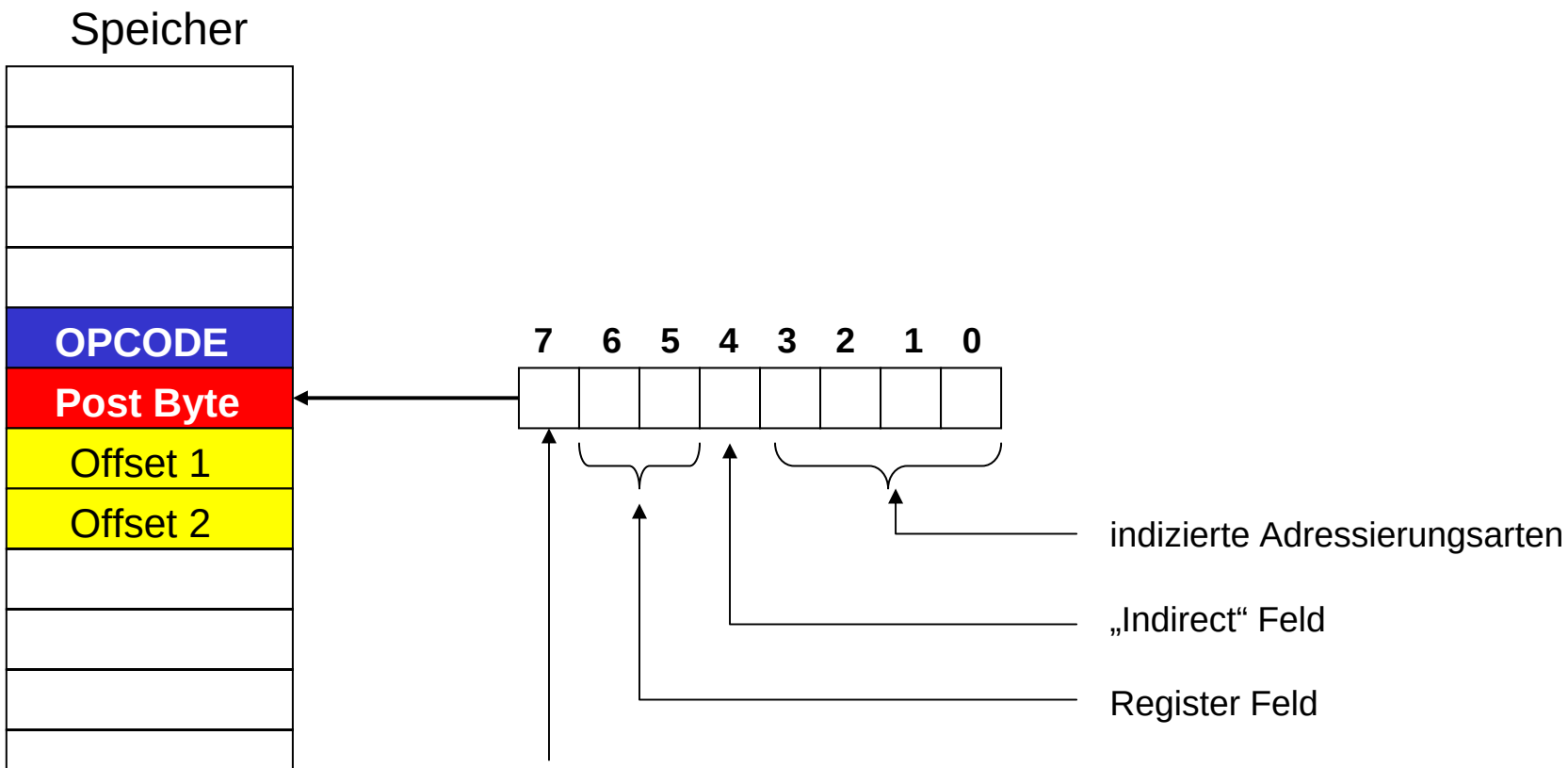
Modi für indizierte Adressierung:

Konstanter Versatz	**Langform (zusätzliches Byte)**	**lange Form(16 Bit)**
		kurze Form (8 Bit)
	Kurzform (Spez. im Post Byte)	**Versatz = 0**
		Versatz ist 5-Bit lang
	Programmzähler Offset	**8/16 Bit**
Accumulator Offset	**A, B und D**	
Auto Increment	**um 1 oder 2**	
Auto Decrement	**um 1 oder 2**	
Extended Indirect		

Assembler Notation:

Notation	Bedeutung
,R	das Register "R" (X, Y, U, S, PC) ist als Indexregister spezifiziert
offset ,R	offset wird zum Indexregister "R" addiert
label ,PCR	Programmzähler-relative Adressierung – den Versatz zwischen PC und Label wird in der Assemblierungsphase berechnet und als konstanter Versatz im Modus: "constant PC-Offset" eingesetzt
,R+(+)	Autoincrement um 1 (bzw. 2) des Registers "R" (X, Y, U, S, PC) (Postincrement)
, -(-)R	Autodecrement um 1 (bzw 2) des Registers "R" (X, Y, U, S, PC) (Predecrement)
[. . . . . .]	indizierte Adresse ist eine indirekte Adresse

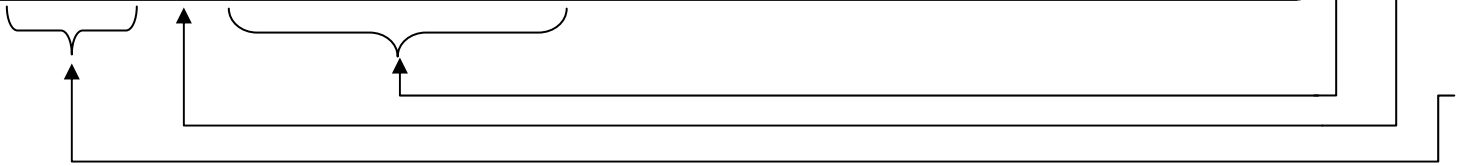
Codierung der Indizierten Adressierungsarten



„Short Offset“ 5-Bit Offset direkt im Post Byte 0-4 enthalten

Belegung des Post Bytes

Bit Number								Addressing Mode
7	6	5	4	3	2	1	0	
0	R	R	X	X	X	X	X	5-Bit Offset
1	R	R	0	0	0	0	0	Autoincrement by 1
1	R	R	i	0	0	0	1	Autoincrement by 2
1	R	R	0	0	0	1	0	Autodecrement by 1
1	R	R	i	0	0	1	1	Autoincrement by 2
1	R	R	i	0	1	0	0	Zero Offset
1	R	R	i	0	1	0	1	Acc. B Offset
1	R	R	i	0	1	1	0	Acc. A Offset
1	R	R	i	1	0	0	0	8-Bit Offset
1	R	R	i	1	0	0	1	16-Bit Offset
1	R	R	i	1	0	1	1	Acc. D Offset
1	R	R	i	1	1	0	0	PC 8-Bit Offset
1	R	R	i	1	1	0	1	PC 16-Bit Offset
1	R	R	1	1	1	1	1	Extended Indirect

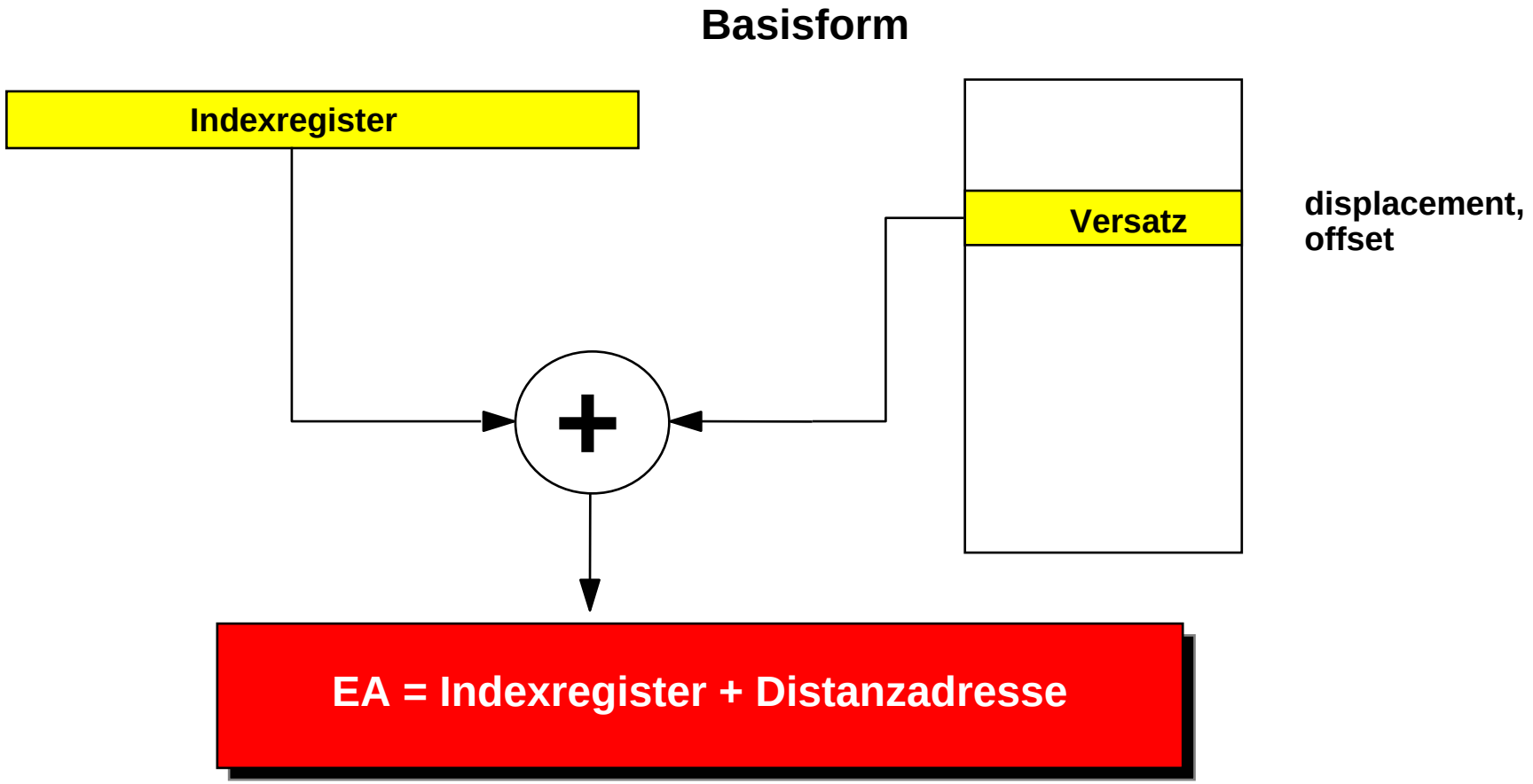


Addressing Mode

Indirect Field:
i=1 for indirect
i=0 for direct
sign bit when bit 7 =0

Register Field:
00 R = X
01 R = Y
10 R = U
11 R = S

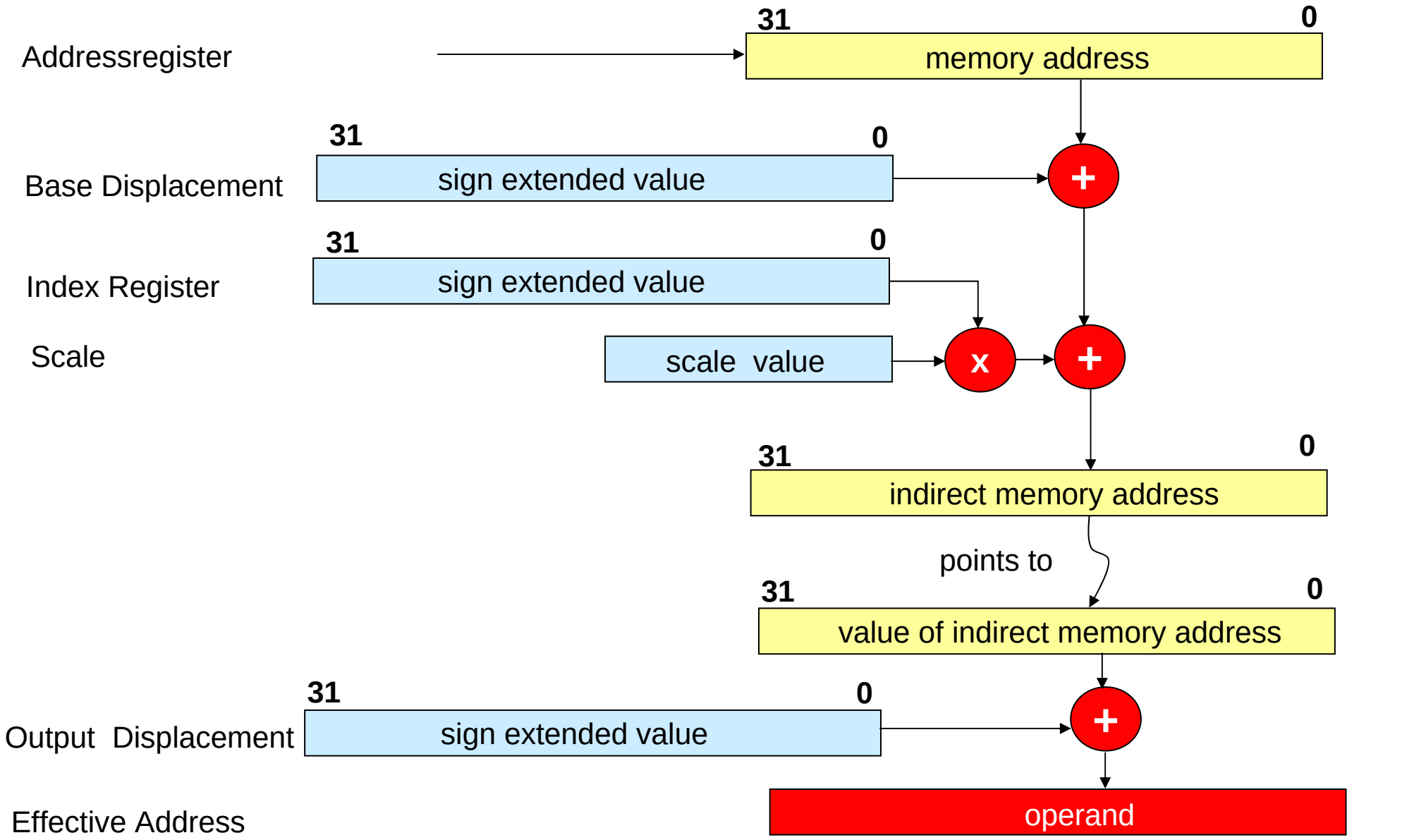
Erweiterungen für die Indizierte Adressierung

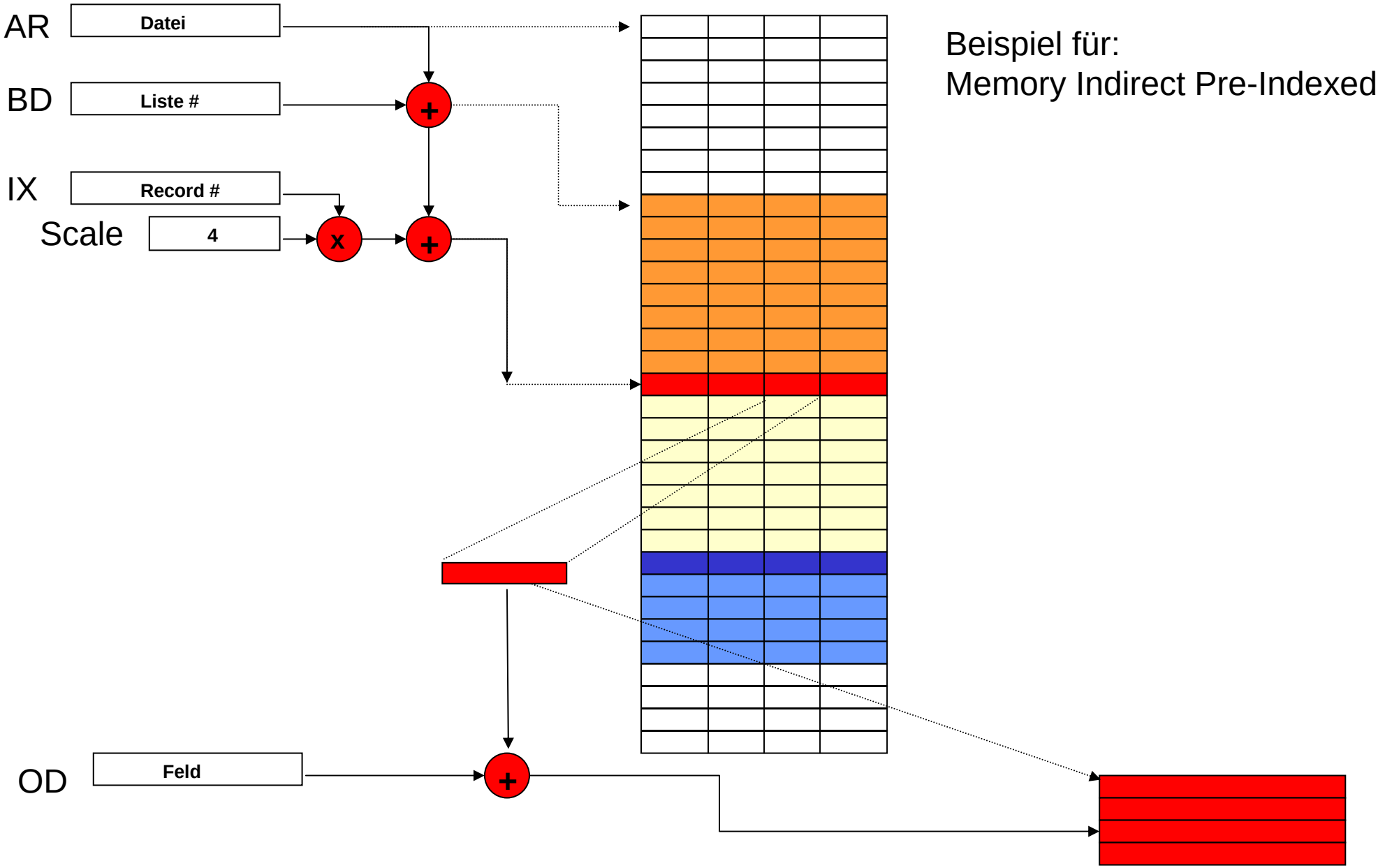


Erweiterungen:

- **skalierter Index**
- **zusätzliche feste Distanzadressen**
- **Indirektion**

Indiziert, indizierter, am ... Memory indirect pre-indexed Adressierungsart bei MC 68020





Beispiel für:
Memory Indirect Pre-Indexed

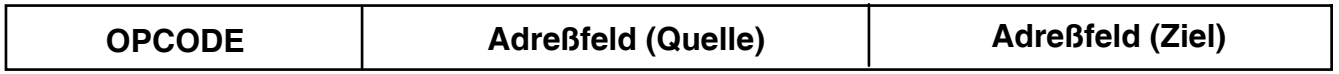
Befehls-formate:

Ein-Adreß-Befehl

Beisp.:
Motorola 6809

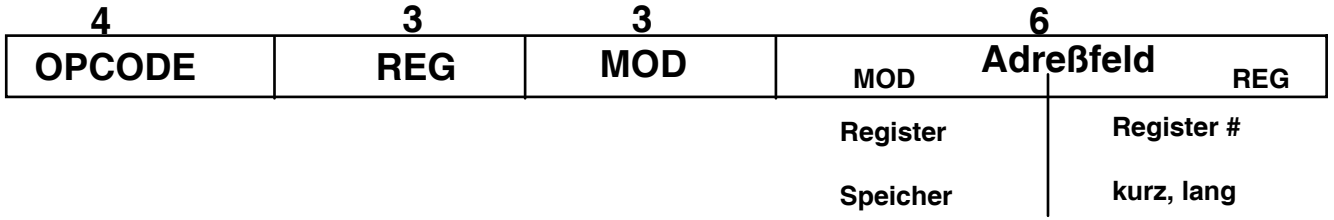
Zwei-Adreß-Befehl

680x0,
Intel 386,486



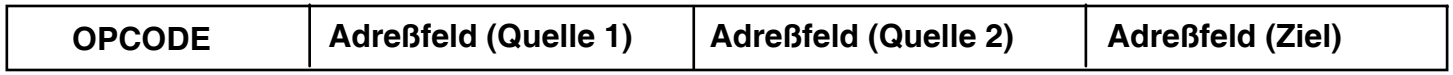
2.ter Quelloperand steht an Zieladresse und wird durch Operation überschrieben

Beisp.:
Motorola
680x0

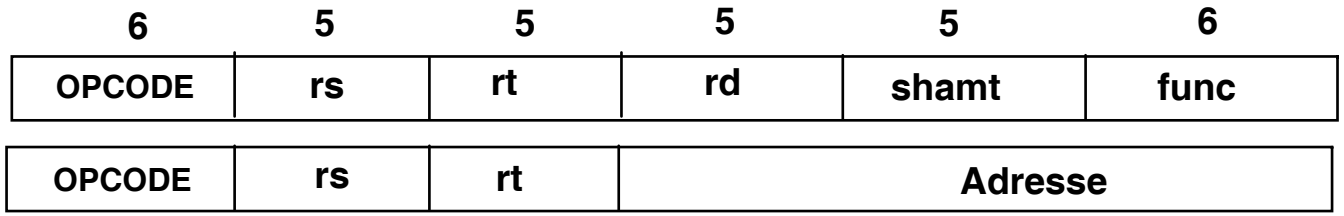


Drei-Adreß-Befehl

MIPS,
Alpha,
SPARC



Beisp.
MIPS



rs: erstes Quellreg.
rt: zweites Quellreg.
rd: Zielreg.
shamt : shift amount
func: funktionale Erw. des OPCODE