

# ***AWL-BEFEHLSBESCHREIBUNG***

Version **2.00** (April 1992)

**Herausgeber:** Bernecker und Rainer Industrie-Elektronik GmbH

Best. Nr.: **MAAWLKB-0**

Inhaltliche Änderungen dieser Kurzbeschreibung behalten wir uns ohne Ankündigung vor. Die Bernecker und Rainer Industrie-Elektronik GmbH haftet nicht für technische oder drucktechnische Fehler und Mängel in dieser Kurzbeschreibung. Außerdem übernimmt die Bernecker und Rainer Industrie-Elektronik GmbH keine Haftung für Schäden, die direkt oder indirekt auf Lieferung, Leistung und Nutzung dieses Materials zurückzuführen sind.

# INHALTSVERZEICHNIS

1. Erklärung der Syntax	9	4. Mathematik-Routinen	179
1.1. CPU-Typ	9	4.1. Allgemeines	179
1.2. Seiteneinteilung	9	4.2. Grundrechenarten	
1.3. Kurzinformation	10	MADD - Addition im Fließkommaformat	183
1.3.1. Prozessor	10	MSUB - Subtraktion im Fließkommaformat	183
1.3.2. Mnemonic (Befehlsabkürzung)	10	MMUL - Multiplikation im Fließkommaformat	184
1.3.3. Funktion	11	MDIV - Division im Fließkommaformat	184
1.3.4. Betriebsart	11	MSQR - Quadratwurzel im Fließkommaformat	185
1.3.5. Adressierungsarten / Opcode	12	4.3. Operandenmanipulation	
1.3.6. Adreßvorwahlen	13	MSGN - Vorzeichenwechsel von Operand 1	185
1.3.7. Statusregister	14	MCOP - Operand OP1 nach Operand OP2 kopieren	186
1.4. Genauere Beschreibung	15	MEXG - Operanden OP1 und OP2 miteinander vertauschen	186
2. Adressierungsarten	17	4.4. Operanden laden	
2.1. IMPL. - Implizite Adressierung	17	LAL1 - Lade Operand OP1 mit Zahl (Absolut-Lang)	187
2.2. DIR. - Direkte Adressierung	19	LAL2 - Lade Operand OP2 mit Zahl (Absolut-Lang)	187
2.3. EXT. - Absolute Adressierung	21	LAW1 - Lade Operand OP1 mit Zahl (Absolut-Kurz)	188
2.4. IMMED. - Unmittelbare Adressierung	23	LAW2 - Lade Operand OP2 mit Zahl (Absolut-Kurz)	188
2.5. IND. - Indizierte Adressierung	25	LIL1 - Lade Operand OP1 mit Zahl (Integer-Lang)	189
2.6. REL. - Relative Adressierung	29	LIL2 - Lade Operand OP2 mit Zahl (Integer-Lang)	189
2.7. Indirekte Adressierung	30	LIW1 - Lade Operand OP1 mit Zahl (Integer-Kurz)	190
2.7. Negation	32	LIW2 - Lade Operand OP2 mit Zahl (Integer-Kurz)	190
3. AWL-Befehle	33	LF1 - Lade Operand OP1 mit Zahl (IEEE-Format)	191
3.1. Ladebefehle	33	LF2 - Lade Operand OP2 mit Zahl (IEEE-Format)	191
3.2. Speicherbefehle	51	4.5. Operand abspeichern	
3.3. Datenaustausch zwischen Registern	59	SAL - Operand OP1 im Format Absolut-Lang abspeichern	192
3.4. Stapeloperationen	69	SAW - Operand OP1 im Format Absolut-Kurz abspeichern	192
3.5. Logische Verknüpfungen	81	SIL - Operand OP1 im Format Integer-Lang abspeichern	193
3.6. Arithmetische Operationen	91	SIW - Operand OP1 im Format Integer-Kurz abspeichern	193
3.7. Vergleichs- und Testbefehle	107	SFX - Operand OP1 im IEEE-Format abspeichern	194
3.8. Inkrement- und Dekrementbefehle	119		
3.9. Schiebe- und Rotierbefehle	131		
3.10. Sprungbefehle	147		
3.11. Sonstiges	163		

4.6. Operandenzwischenspeicher		4.10. Hilfsroutinen	
SFM1 - Operand OP1 im Zwischenspeicher 1 speichern	195	FCOP - Speicherbereich kopieren	226
SFM2 - Operand OP1 im Zwischenspeicher 2 speichern	195	FSMB - Speicherbereich mit 1 Byte-Wert initialisieren	227
SFM3 - Operand OP1 im Zwischenspeicher 3 speichern	195	FSMW - Speicherbereich mit 2 Byte-Wert initialisieren	228
RFM1 - Operand OP2 aus Zwischenspeicher 1 laden	196	FCLR - Speicherbereich löschen	229
RFM2 - Operand OP2 aus Zwischenspeicher 2 laden	196	MCMP - OP1 und OP2 vergleichen	230
RFM3 - Operand OP2 aus Zwischenspeicher 3 laden	196	MHIL - Begrenzen auf Obergrenze; Wenn OP1 > OP2, dann OP2 $\Rightarrow$ OP1	231
4.7. Schnelle Multiplikationen		MLOL - Begrenzen auf Untergrenze; Wenn OP1 < OP2, dann OP2 $\Rightarrow$ OP1	232
FM2B - 2 Byte X 2 Byte Multiplikation	197		
FM3B - 3 Byte X 2 Byte Multiplikation	197		
FM4B - 4 Byte X 2 Byte Multiplikation	198		
4.8. Zahlenumwandlungen		5. Sonderbefehle	233
CAF - ASCII-String in IEEE-Format umwandeln	199	5.1. Arithmetikprozessor (nur CP80)	233
CFA - OP1 in ASCII ohne Vornullen wandeln	200	5.1.1. Operanden	233
CFA0 - OP1 in ASCII mit Vornullen wandeln	202	5.1.2. Aufruf von APU-Befehlen	234
CFEA - OP1 in ASCII mit Exponentendarstellung wandeln	204	5.1.3. Zahlenformate	235
CIA - Binär in ASCII ohne Vornullen wandeln	206	5.1.4. Befehlssatz des 68881 Arithmetikprozessors	236
CIA0 - Binär in ASCII mit Vornullen wandeln	208	5.1.5. Konstanten	238
CBCD - Binär in BCD wandeln	210	5.2. Schnittstellenbefehle (nur CP80, PP60 und NTCP6#)	241
CBIN - BCD in Binär wandeln	211	5.2.1. SOB - Zeichen ausgeben	241
4.9. Umrechnungsroutinen		5.2.2. SIB - Zeichen einlesen	242
Allgemeines zu Umrechnung Binär $\Leftrightarrow$ Physikalisch	212	5.2.3. SC - Schnittstellenstatus anfordern	243
CBPP - Berechnung der Faktoren k und d aus zwei Geradenstützpunkten	214	5.2.4. SF - Schnittstellenfunktionen	244
CBPQ - Wandle Binär $\Rightarrow$ Physikalisch, schnell	216	5.2.5. Unterschiedliche Behandlung der Handshake-Leitungen	255
CPBQ - Wandle Physikalisch $\Rightarrow$ Binär, schnell	217		
CBP - Wandle Binär $\Rightarrow$ Physikalisch	218	6. Anhang	259
CPB - Wandle Physikalisch $\Rightarrow$ Binär	220	6.1. Alphabetische Übersicht der B&R Mnemonics	259
CIM - Wandle Zoll $\Rightarrow$ Millimeter	222	6.2. Alphabetische Übersicht der Motorola Mnemonics	260
CMI - Wandle Millimeter $\Rightarrow$ Zoll	224	6.3. Alphabetische Übersicht der Mathematik-Routinen	261
		6.4. Alphabetische Übersicht der B&R Kurz Mnemonics	261
		6.5. Stichwortverzeichnis	262

# Register

	Erklärung der Syntax	
	Adressierungsarten	
<b>AWL-Befehle</b>	Ladebefehle	
	Speicherbefehle	
	Datenaustausch zwischen Registern	
	Stapeloperationen	
	Logische Verknüpfungen	
	Arithmetische Operationen	
	Vergleichs- und Testbefehle	
	Inkrement- und Dekrementbefehle	
	Schiebe- und Rotierbefehle	
	Sprungbefehle	
	Sonstiges	
<b>Mathematik-Routinen</b>	Grundrechenarten	
	Operandenmanipulation	
	Operanden laden	
	Operand abspeichern	
	Operandenzwischenspeicher	
	Schnelle Multiplikationen	
	Zahlenumwandlungen	
	Binär <=> Physikalisch-Wandlung	
	Hilfsroutinen	
<b>Sonderbefehle</b>	Arithmetikprozessor (nur CP80)	
	Schnittstellenbefehle (nur CP80, PP60 und NTCp6#)	

# 1. ERKLÄRUNG DER SYNTAX

## 1.1. CPU-TYP

Je nach verwendetem Prozessor wird zwischen zwei CPU-Typen unterschieden.

Prozessor	CPU-TYP	Modul
<b>6303</b>	<b>A</b>	CP40, CP41, NTCP33, NTCP34, PSCP35, PP40, CP30, CP31, CP32
<b>6809</b>	<b>B</b>	CP60, CP80, NTCP63, NTCP64, PSCP65, PP60

## 1.2. SEITENEINTEILUNG

Diese Unterscheidung wird bei der Beschreibung der AWL-Befehle in der Weise berücksichtigt, daß ein Befehl je einmal für den CPU TYP A und einmal für den CPU TYP B beschrieben wird. Dazu wird eine Seite in zwei Teile gegliedert:

- Linke Seite: Beschreibung des 6303 Befehles (CPU Typ A)
- Rechte Seite: Beschreibung des 6809 Befehles (CPU Typ B)

## 1.3. KURZINFORMATION

Zu jedem Befehl gibt es in Tabellenform eine Kurzinformation, die wie folgt aussieht:

Motorola		Funktion																CPU Typ A / 6303									
B&R																											
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen														PG1000	
																										PG-PC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	

Motorola		Funktion														CPU Typ B / 6809										
B&R																										
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														CP 80						
																				PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C

### 1.3.1. Prozessor

Rechts oben in der Tabelle wird der CPU TYP angegeben, für den die Beschreibung gilt. Auf der linken Seite steht immer die Beschreibung für den **CPU TYP A** und auf der rechten für den **CPU TYP B**. Ist eine Seite **grau** hinterlegt, bezieht sich die folgende Beschreibung nur auf diesen CPU-Typ (in diesem Fall CPU Typ B).

### 1.3.2. Mnemonic (Befehlsabkürzung)

Für die Schreibweise eines Befehles werden drei mögliche Mnemonics angegeben, die je nach Einstellung des PROgrammierSYStemes verwendet werden können:

- 1) **Motorola:** MOTOROLA® Mnemonic
- 2) **B&R:** B&R Mnemonic
- 3) **Kurz:** Für einige B&R Mnemonics gibt es Abkürzungen, die zur schnelleren Eingabe eines AWL-Befehles verwendet werden können. Nach Eingabe einer solchen Abkürzung wird vom PROgrammierSYStem automatisch die entsprechende B&R-Mnemonic (im B&R Modus) bzw. die MOTOROLA®-Mnemonic (im MIXM Modus) eingesetzt.

## 1.3.3. Funktion

In diesem Feld wird der Befehl kurz mit symbolischen Zeichen beschrieben. Dazu werden folgende Abkürzungen und Symbole verwendet:

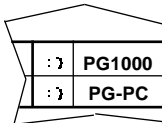
ERA	Ergebnisregister A (8 Bit)
ERB	Ergebnisregister B (8 Bit)
ERD	Ergebnisregister D (16 Bit; ERA = MSB, ERB = LSB)
SR	Statusregister (8 Bit)
DP	Register für direkte Adressierung (8 Bit) (Direct Page Register)
R	Indexregister X (16 Bit)
Y	Indexregister Y (16 Bit)
SP!	System-Stapelzeiger (16 Bit)
SPU	Anwender- oder User-Stapelzeiger (16 Bit)
PC	Programmzähler (Program counter)
M	Adresse einer Speicherstelle (Memory)
(M)	Inhalt der Speicherstelle mit der Adresse M (8 Bit)
(M:M+1)	Inhalt der beiden Speicherstellen mit den Adressen M und M+1 (16 Bit)

$r_8$	8-Bit Register (ERA, ERB, SR, DP)
$r_{16}$	16-Bit Register (ERD, R, Y SP!, SPU)
I	IRQ Mask
$\wedge$	Logische UND Verknüpfung
$\vee$	Logische ODER Verknüpfung
$\oplus$	Logische Exklusiv-Oder (EXOR) Verknüpfung
MSB	Höherwertiges Byte eines 2 Byte Wertes
LSB	Niederwertiges Byte eines 2 Byte Wertes
EA	Effektive Adresse
IMM	Immediate Wert (Konstante)
dn	Datenbit n eines Registers oder einer Speicherstelle

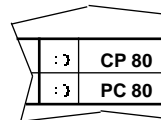
## 1.3.4. Betriebsart

Das Symbol “:J” zeigt an, daß ein Befehl in dieser Betriebsart des PROgrammierSYSTEMes verfügbar ist.

Beispiel:



oder



## 1.3.5. Adressierungsarten / Opcode

Bei Adressierungsarten, die für einen Befehl möglich sind, werden zwei Werte getrennt durch "/" angegeben.

- Beispiel:** 4/2
- Der erste Wert gibt die *Ausführungszeit* des Befehles in *Taktzyklen* an.
  - Der zweite Wert gibt die *Länge* des Befehles in *Byte* an.

Neben einem dieser Werte kann sich ein "+" befinden (z.B.: 4+/2+). In diesem Fall heißt dies: Die Ausführungszeit des Befehles beträgt **4 oder mehr** Taktzyklen und der Befehl ist **2 oder mehr** Byte lang.

Der Opcode eines Befehles wird unter den Angaben Befehlslänge und Taktzyklen angeführt. Besteht ein Opcode aus mehr als einem Byte, werden diese durch ein Leerzeichen getrennt.

Überblick über die möglichen Adressierungsarten:

<b>IMPL.</b>	<b>Implizit</b> Der Befehl braucht keine zusätzlichen Parameter. Der Opcode enthält bereits alle nötigen Adreßinformationen zur Ausführung des Befehles.	<b>IMMED.</b>	<b>Unmittelbar (Konstante)</b> Der Wert (1 oder 2 Byte), den der Befehl verarbeitet, wird unmittelbar mit dem Befehl eingegeben.
<b>DIR.<sup>1)</sup></b>	<b>Direkt</b> Zusätzlich zum Befehl wird das niederwertige Byte der Adresse angegeben. Das höherwertige Byte wird durch das DP-Register (Direct Page Register) definiert.	<b>IND.</b>	<b>Indiziert (Indirekt Indiziert<sup>2)</sup>)</b> Bei der indizierten Adressierung wird die effektive Adresse aus dem Offset, der zusätzlich zum Befehl eingegeben wird, und einem der Indexregister (I, Y <sup>1)</sup> , SP <sup>1)</sup> , SPU <sup>1)</sup> ) berechnet.
<b>EXT.</b>	<b>Absolut (Indirekt Absolut<sup>2)</sup>)</b> Zusätzlich zum Befehl wird die ganze Adresse (2 Byte) eingegeben.	<b>REL.</b>	<b>Relativ</b> Die relative Adressierung wird bei bedingten Sprüngen verwendet. Die Zieladresse berechnet sich aus dem aktuellen PC (Program counter) und dem angegebenen 1 oder 2 <sup>1)</sup> Byte Offset.

<sup>1)</sup> nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar.

<sup>2)</sup> **Indirekt:** Für die Adressierungsarten EXT. und IND. gibt es die zusätzliche Möglichkeit der *indirekten* Adressierung. Bei dieser Art der Adressierung befindet sich die effektive Adresse, auf die sich der Befehl bezieht, an der angegebenen Adresse (nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar).



## 1.3.6. Adreßvorwahlen

Es sind verschiedene Adreßvorwahlen möglich:

MOTOROLA	B&R		
I	E	Eingang	
O	A	Ausgang	
F	M	1-Bit Speicherstelle	
S	F	Freigabe einer Zeit	
T	Z	Signal einer abgelaufenen Zeit	
#	#	Konstante	
P	P	Peripherie-Adresse	
R	C	8-Bit Speicherstelle	
X	I	Indexregister R (X)	} indizierte Adressierung
Y	Y	Indexregister Y	
U	U	Anwender Stapelzeiger	
!	!	System Stapelzeiger	
D	D	Direkte Adressierung	
G	G	Globales RAM im PP60 (erweiterter Koppel-RAM Bereich) <sup>1)</sup>	
B	B	Blockspeicher im PP60 <sup>1)</sup>	

Weiters werden noch folgende Symbole verwendet:

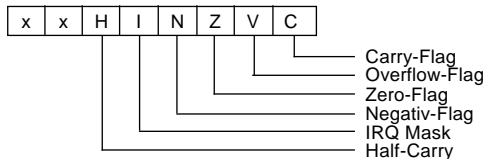
- : } verfügbar in den angegebenen Betriebsarten
- verfügbar ab PROgrammierSYstem Version 5.00 in Betriebsart PC 80

1) verfügbar ab PROgrammierSYstem Version 5.00 in Betriebsart PC 80

## 1.3.7. Statusregister

Die Bits des Statusregisters werden entsprechend dem Ergebnis eines Befehles verändert. Der Zustand des Statusregisters beeinflusst z.B. bedingte Sprünge (abhängig vom Statusregister wird ein Sprung ausgeführt oder der nächste Befehl wird abgearbeitet).

Das Statusregister sieht wie folgt aus:



**Carry-Flag:** Dieses Bit wird auf log. 1 gesetzt, wenn die vorhergehende Rechenoperation (Addition, Subtraktion) einen Übertrag ergibt, d.h.: das Ergebnis ist größer oder kleiner, als das Rechenregister fassen kann.

**Overflow-Flag:** Dieses Bit wird auf log. 1 gesetzt, wenn die vorhergehende Rechenoperation einen 2er-Komplement Überlauf verursacht, d.h. das Ergebnis ist außerhalb des Bereiches -128 bis +127 (bei 1 Byte Werten) bzw. außerhalb -32768 bis +32767 (bei 2 Byte Werten).

**Zero-Flag:** Dieses Bit wird auf log. 1 gesetzt, wenn das

Ergebnis des zuletzt ausgeführten Befehles definitiv NULL war, andernfalls erhält es den Wert 0

**Negativ-Flag:** Dieses Bit enthält den Wert des höherwertigsten Bits des Ergebnisses des zuletzt ausgeführten Befehles.

**IRQ Mask:** Wird dieses Bit auf log. 1 gesetzt, werden alle Interrupts des Prozessors gesperrt.

**Half-Carry:** Dieses Bit wird auf log. 1 gesetzt, wenn bei einer Addition (nur ADD, ++B, + und +B) ein Übertrag von Bit 3 auf Bit 4 erfolgt. Dieses Bit wird vom Befehl DK verwendet, um die Dezimalkorrektur durchzuführen.

**Bit 6 und 7:**

Im 6805 werden diese Bits nicht verwendet und sind immer auf log. 1 gesetzt.

Im 6809 sollten diese Bits **nicht** vom Anwender verändert werden!

Folgende Symbole werden verwendet:

- ⋄ Flag wird nicht beeinflusst.
- Flag wird entsprechend der Operation verändert.
- ▲ Flag wird auf log. 1 gesetzt.
- ▼ Flag wird auf log. 0 gesetzt.
- × Zustand des Flags ist nicht definiert.

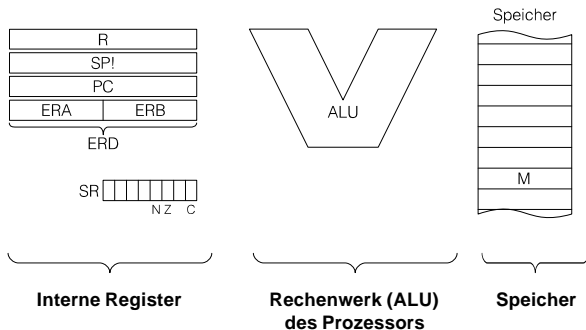
## 1.4. Genauere Beschreibung

Nach der Kurzinformation über jeden Befehl folgt:

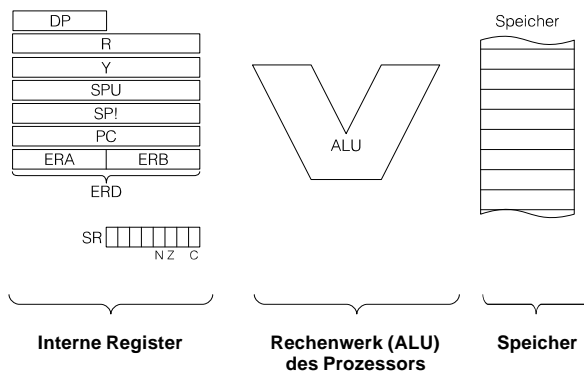
- 1) eine genauere Beschreibung des Befehles.
- 2) eine Skizze über den Datenfluß.

Die Skizze kann je nach CPU-Typ folgende Elemente enthalten:

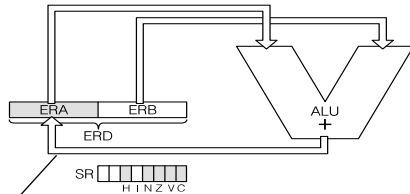
### CPU-Typ A



### CPU-Typ B

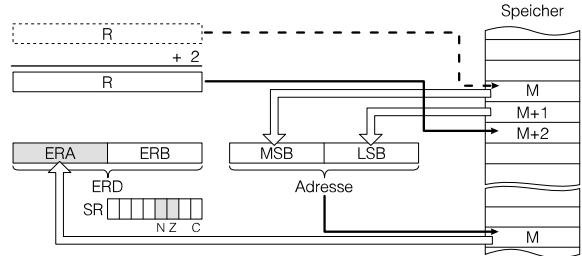


Register, Speicher oder Statusregisterbits, die bei der Ausführung des Befehles verändert werden könnten, werden **grau hinterlegt** gekennzeichnet.



**Breite Pfeile** beschreiben einen Datenfluß.

**Schmale Pfeile** stellen einen Zeiger dar (Der Inhalt eines Registers zeigt auf eine Speicherstelle).

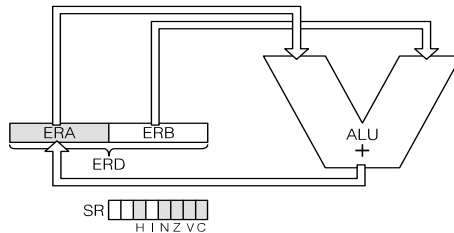


**Strichlierte** Register und Zeiger stellen den Zustand vor Ausführung des Befehles dar.

## 2. ADRESSIERUNGSARTEN

### 2.1. IMPL. - IMPLIZITE ADRESSIERUNG

Der Opcode des Befehles enthält bereits alle nötigen Adreßinformationen zur Ausführung des Befehles. Die Angabe eines Parameters zusätzlich zum Befehl ist nicht notwendig. Solche ein Befehl wäre z.B.: A+B (MOTOROLA: ABA). Der Prozessor verwendet nur die zwei Register ERA und ERB. Der Datenfluß würde wie folgt aussehen:



# Befehlsübersicht / Implizite Adressierung

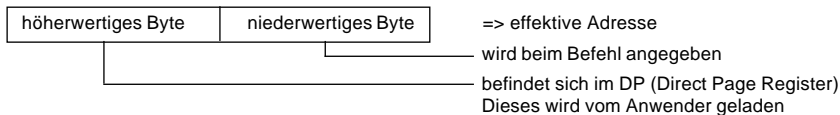
MOTOROLA	B&R	MOTOROLA	B&R	MOTOROLA	B&R
ABA	A+B	DEX	DR	PULX	RVS
ABX	B+R	EXG <sup>1)</sup>	EXG <sup>1)</sup>	ROLA	RLA
ASLA	SLA	INCA	IA	ROLB	RLB
ASLB	SLB	INCB	IB	RORA	RRA
ASLD	SLD	INS	IS	RORB	RRB
CBA	AVB	INX	IR	RTS	RET
CLC	CLC	LSRA	SRA	SBA	A-B
CLI	CLI	LSRB	SRB	SEC	SEC
CLRA <sup>2)</sup>	CLA <sup>2)</sup>	LSRD	SRD	SEI	SEI
CLRB <sup>2)</sup>	CLB <sup>2)</sup>	MUL	A*B	TAB	MAB
COMA <sup>2)</sup>	COA <sup>2)</sup>	NOP	NOP	TBA	MBA
COMB <sup>2)</sup>	COB <sup>2)</sup>	PSHA	ANS	TFR <sup>1)</sup>	TFR <sup>1)</sup>
DAA	DK	PSHB	BNS	TPA	MCA
DECA	DA	PSHX	RNS	TSX	MSR
DECB	DB	PULA	AVS	TXS	MRS
DES	DS	PULB	BVS	XGDX	DXR

<sup>1)</sup> nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar

<sup>2)</sup> nur ab PROgrammierSYStem Version 5.00 in den Betriebsarten PG-PC oder PC 80 verfügbar

## 2.2. DIR. - DIREKTE ADRESSIERUNG

Diese Adressierungsart ist vergleichbar mit der indizierten Adressierung. Die Adresse, auf die ein Befehl zugreift, setzt sich aus zwei Teilen zusammen:

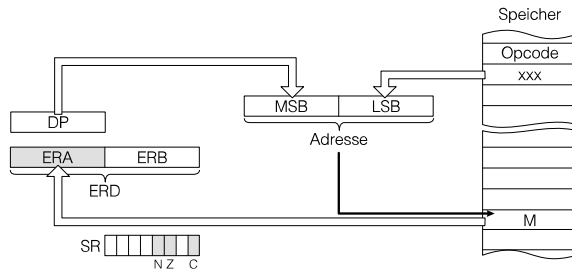


Der Speicherbereich wird so in "Pages" (Seiten) zu je 256 Byte unterteilt. Eine Page kann mit der Adreßvorwahl "D" direkt adressiert werden. Die Schreibweise würde z.B. wie folgt aussehen:

**LAD    D 000**  
**=       D 200**

Der Vorteil dieser Adressierung liegt in einer kürzeren Ausführungszeit des Befehles. Diese Adressierungsart wird dann angewendet, wenn häufige und schnelle Zugriffe auf einen bestimmten Speicherbereich erforderlich sind.

**Beispiel:    LAD    D xxx**



# Befehlsübersicht / Direkte Adressierung:

Diese Adressierungsart ist nur ab PROgrammierSYstem Version 5.00 in Betriebsart PC 80 verfügbar.

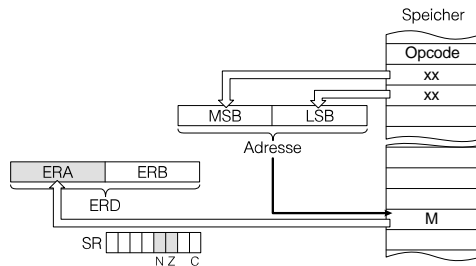
MOTOROLA	B&R	MOTOROLA	B&R	MOTOROLA	B&R
ADCA	ADD	CMPA	CMP	ORAB	OB
ADCB	++B	CMPB	VB	ROL	SLI
ADDA	+	COM	K	ROR	SRE
ADDB	+B	DEC	DEC	SBCA	SUB
ADDD	+D	EORA	EXO	SBCB	--B
ANDA	UND	EORB	EB	STAA	=
ANDB	UB	INC	INC	STAB	=B
ASL	SL	LDAA	LAD	STD	=D
BITA	B	LDAB	LB	SUBA	-
BITB	BB	LDD	LD	SUBB	-B
CLR	CLR	ORAA	OD	SUBD	-D



## 2.3. EXT. - ABSOLUTE ADRESSIERUNG

Bei der absoluten Adressierung folgen dem Opcode 2 Bytes, die die 16 Bit Adresse darstellen, welche vom Befehl verwendet wird. Bei der absoluten Adressierung sind die Adreßvorwahlen E, A, M, F, Z, P, C, B und G möglich.

**Beispiel:** LAD C 0000



# Befehlsübersicht / Absolute Adressierung

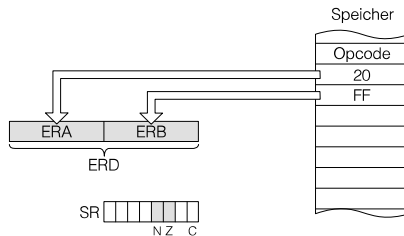
MOTOROLA	B&R	MOTOROLA	B&R	MOTOROLA	B&R
ADCA	ADD	DEC	DEC	ROL	SLI
ADCB	++B	EORA	EXO	ROR	SRE
ADDA	+	EORB	EB	RST	RST
ADDB	+B	INC	INC	SBCA	SUB
ADDD	+D	JMP	SPI	SBCB	--B
ANDA	UND	JSR	SPU	SET	SET
ANDB	UB	LDAA	LAD	STAA	=
ASL	SL	LDAB	LB	STAB	=B
BITA	B	LDD	LD	STD	=D
BITB	BB	LDS	LS	STS	=S
CLR	CLR	LDX	LR	STX	=R
CMPA	CMP	LDY <sup>1)</sup>	LY <sup>1)</sup>	STY <sup>1)</sup>	=Y <sup>1)</sup>
CMPB	VB	LSR	SR	SUBA	-
COM	K	ORAA	OD	SUBB	-B
CPX	VR	ORAB	OB	SUBD	-D
CPY <sup>1)</sup>	VY <sup>1)</sup>	PRS	PRS		

<sup>1)</sup> nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar

## 2.4. IMMED. - UNMITTELBARE ADRESSIERUNG

In diesem Fall wird der 1 bzw. 2 Byte Wert, der vom Befehl verarbeitet wird, nach dem Befehl angegeben. Dieser Wert wird in der auf den Opcode folgenden Speicherstelle gespeichert.

**Beispiel:** LD # \$20FF



# Befehlsübersicht / Unmittelbare Adressierung

MOTOROLA	B&R	MOTOROLA	B&R	MOTOROLA	B&R
ADCA	ADD	CPX#	VRK	LDY# <sup>1)</sup>	LYK <sup>1)</sup>
ADCB	++B	CPY# <sup>1)</sup>	VYK <sup>1)</sup>	LDYL <sup>1)</sup>	LYL <sup>1)</sup>
ADDA	+	EIM <sup>2)</sup>	EIM <sup>2)</sup>	OIM <sup>2)</sup>	OIM <sup>2)</sup>
ADDB	+B	EORA	EXO	ORAA	OD
ADDD	+D	EORB	EB	ORAB	OB
AIM <sup>2)</sup>	AIM <sup>2)</sup>	LDAA	LAD	SBCA	SUB
ANDA	UND	LDAB	LB	SBCB	--B
ANDB	UB	LDD	LD	SUBA	-
BITA	B	LDK <sup>3)</sup>	LDK <sup>3)</sup>	SUBB	-B
BITB	BB	LDX#	LRK	SUBD	-D
CMPA	CMP	LDXL <sup>3)</sup>	LDL <sup>3)</sup>	TIM <sup>2)</sup>	TIM <sup>2)</sup>
CMPB	VB				

<sup>1)</sup> nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar

<sup>2)</sup> nur ab PROgrammierSYStem Version 5.00 in Betriebsart PG-PC verfügbar (nur CPU-Typ A)

<sup>3)</sup> nur ab PROgrammierSYStem Version 5.00 in Betriebsart PG-PC oder PC 80 verfügbar

## 2.5. IND. - INDIZIERTE ADRESSIERUNG

Die indizierte Adressierung ist mit folgenden Adreßvorwahlen möglich: I, Y, U, !

Bei der indizierten Adressierung wird ein Befehl auf den Inhalt einer oder mehrerer Speicherstellen angewendet, auf die ein Indexregister (R, Y, SPU, SP!) zeigt. Es sind drei verschiedene Arten der indizierten Adressierung möglich:

### 2.5.1. INDIZIERT MIT KONSTANTEM OFFSET

Minimale und maximale Größe des Offsets hängt von der Betriebsart des PROgrammierSYStemes ab:

Betriebsart	PG1000	CP 80	PG-PC	PC 80
Offset	I 000 bis I 255	I -128 bis I 127	I 000 bis I 255	I -32768 bis I 32767 Y -32768 bis Y 32767 U -32768 bis U 32767 ! -32768 bis ! 32767

Im PC 80 Modus hängt die Länge des Befehles von der Größe des Offsets ab:

Offsetwert	-16 bis +15	-128 bis -17 +16 bis +127	-32768 bis -129 +128 bis +32767
Befehlslänge	Opcode + 0 Byte	Opcode + 1 Byte	Opcode + 2 Byte

Die Offsetwerte werden in der AWL-Eingabezeile unmittelbar nach dem Befehl und der Adreßvorwahl eingegeben.

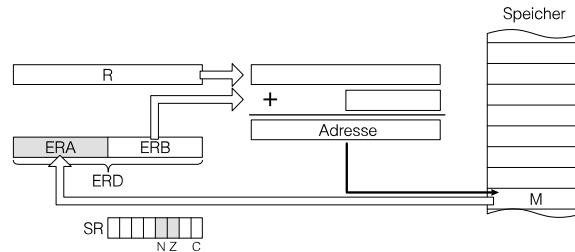
**Beispiel:** Berechnung der effektiven Befehlslänge in Byte für den Befehl LAD. In der Beschreibung dieses Befehles wird die Befehlslänge bei der indizierten Adressierung (IND.) mit 2+ angegeben.

Offsetwert	-16 bis +15	-128 bis -17 +16 bis +127	-32768 bis -129 +128 bis +32767
Befehlslänge	2 + 0 = 2 Byte	2 + 1 = 3 Byte	2 + 2 = 4 Byte

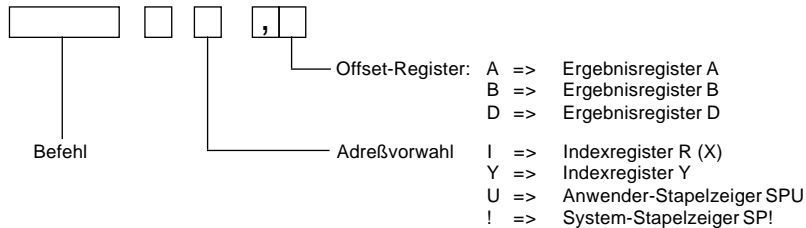
### 2.5.2. INDIZIERT MIT REGISTEROFFSET (VARIABLER OFFSET)

Der Inhalt des Offset-Registers (Ergebnisregister ERA, ERB oder ERD) wird als Zweierkomplementzahl zum Inhalt des Indexregisters (R, Y, SPU oder SP!) addiert. Diese Art der Adressierung ist nur in Betriebsart PC 80 möglich.

**Beispiel:** LAD I ,B

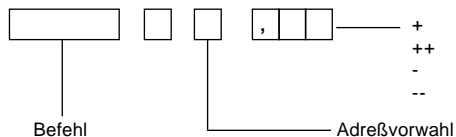


Für die indizierte Adressierung mit Registeroffset gibt es folgende Möglichkeiten:



### 2.5.3. INDIZIERT MIT POSTINKREMENT ODER PREDEKREMENT DES INDEXREGISTERS

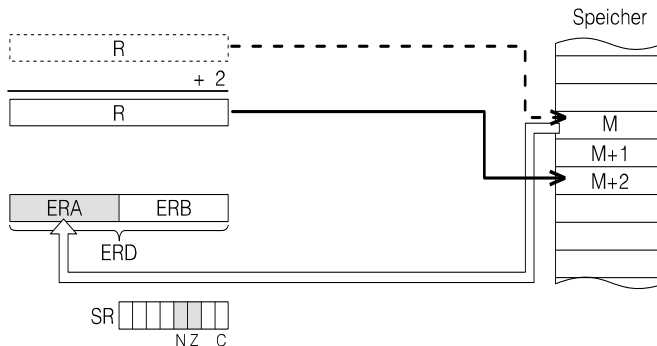
Indexregister können automatisch vor der Ausführung des Befehles dekrementiert (vermindert) oder nach der Ausführung inkrementiert (erhöht) werden. Diese Adressierungsart ist nur ab PROgrammierSYstem Version 5.00 in Betriebsart PC 80 verfügbar.



**Inkrement** des Indexregisters **nach** Ausführung des Befehles um 1  
**Inkrement** des Indexregisters **nach** Ausführung des Befehles um 2  
**Dekrement** des Indexregisters **vor** Ausführung des Befehles um 1  
**Dekrement** des Indexregisters **vor** Ausführung des Befehles um 2

I => Indexregister R (X)  
Y => Indexregister Y  
U => Anwender-Stapelzeiger SPU  
! => System-Stapelzeiger SP!

**Beispiel:** LAD I , ++



# Befehlsübersicht / Indizierte Adressierung

MOTOROLA	B&R	MOTOROLA	B&R	MOTOROLA	B&R
ADCA	ADD	DEC	DEC	LSR	SR
ADCB	++B	EORA	EXO	ORAA	OD
ADDA	+	EORB	EB	ORAB	OB
ADDB	+B	INC	INC	ROL	SLI
ADDD	+D	JMP	SPI	ROR	SRE
ANDA	UND	JSR	SPU	SBCA	SUB
ANDB	UB	LDAA	LAD	SBCB	--B
ASL	SL	LDAB	LB	STAA	=
BITA	B	LDD	LD	STAB	=B
BITB	BB	LDS	LS	STD	=D
CLR	CLR	LDX	LR	STS	=S
CMPA	CMP	LDY <sup>1)</sup>	LY <sup>1)</sup>	STX	=R
CMPB	VB	LEA <sup>1)</sup>	LE <sup>1)</sup>	STY <sup>1)</sup>	=Y <sup>1)</sup>
COM	K	LEAU <sup>1)</sup>	LEU <sup>1)</sup>	SUBA	-
CPX	VR	LEAX <sup>1)</sup>	LER <sup>1)</sup>	SUBB	-B
CPY <sup>1)</sup>	VY <sup>1)</sup>	LEAY <sup>1)</sup>	LEY <sup>1)</sup>	SUBD	-D

<sup>1)</sup> nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar



## 2.6. REL. - RELATIVE ADRESSIERUNG

Die relative Adressierung wird nur bei bedingten Sprungbefehlen verwendet. Ein Offset (Zweierkomplementzahl) wird zum Program counter addiert. Je nach Größe des Offsets wird zwischen **kurzen** (1 Byte Offset) und **langen** (2 Byte Offset) Sprüngen unterschieden.

### Befehlsübersicht / Relative Adressierung

KURZ		LANG <sup>1)</sup>	
MOTOROLA	B&R	MOTOROLA	B&R
BCC	JC0	BCCL	JC0L
BEQ	SP0	BEQL	SP0L
BHI	SP>	BHIL	SP>L
BCS	SP<	BCSL	SP<L
BLS	J<=	BLSL	J<=L
BMI	J-	BMIL	J-L
BNE	SN0	BNEL	SN0L
BPL	J+	BPLL	J+L

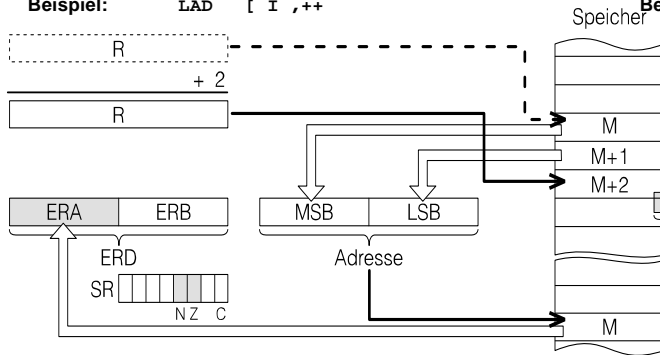
<sup>1)</sup> nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar

## 2.7. INDIREKTE ADRESSIERUNG

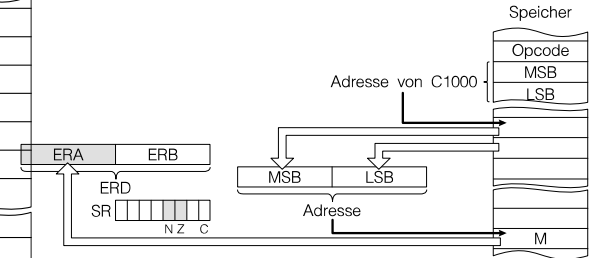
Zusätzlich zur absoluten und zur indizierten Adressierung besteht die Option, die resultierende Adresse als "Adresse der Adresse" zu verwenden, d.h.: wird die **indirekt absolute** oder **indirekt indizierte** Adressierung verwendet, befindet sich die effektive Adresse, auf die der Befehl angewendet wird, an der Speicherstelle, auf die die ursprüngliche Adresse zeigt. Diese Adressierungsart wird durch die eckige Klammer "[" gekennzeichnet.

Um die indirekte Adressierung auszuwählen muß der Cursor in der AWL-Eingabzeile auf dem Feld "Adreßvorwahl" stehen".

**Beispiel:** LAD [ I , ++



**Beispiel:** LAD [ C 1000



Die indirekte Adressierung ist nur bei Zwei-Byte Dekrement bzw. Inkrement möglich.

# Befehlsübersicht / Indirekte Adressierung

MOTOROLA	B&R	MOTOROLA	B&R	MOTOROLA	B&R
ADCA	ADD	DEC	DEC	LSR	SR
ADCB	++B	EORA	EXO	ORAA	OD
ADDA	+	EORB	EB	ORAB	OB
ADDB	+B	INC	INC	ROL	SLI
ADDD	+D	JMP	SPI	ROR	SRE
ANDA	UND	JSR	SPU	SBCA	SUB
ANDB	UB	LDAA	LAD	SBCB	--B
ASL	SL	LDAB	LB	STAA	=
BITA	B	LDD	LD	STAB	=B
BITB	BB	LDS	LS	STD	=D
CLR	CLR	LDX	LR	STS	=S
CMPA	CMP	LDY <sup>1)</sup>	LY <sup>1)</sup>	STX	=R
CMPB	VB	LEA! <sup>1) 2)</sup>	LE! <sup>1) 2)</sup>	STY <sup>1)</sup>	=Y <sup>1)</sup>
COM	K	LEAU <sup>1) 2)</sup>	LEU <sup>1) 2)</sup>	SUBA	-
CPX	VR	LEAX <sup>1) 2)</sup>	LER <sup>1) 2)</sup>	SUBB	-B
CPY <sup>1)</sup>	VY <sup>1)</sup>	LEAY <sup>1) 2)</sup>	LEY <sup>1) 2)</sup>	SUBD	-D

<sup>1)</sup> nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar

<sup>2)</sup> nicht für indirekt absolute Adressierung möglich.

## 2.7. NEGATION

Die Negation ist nur bei 1 Bit-Adressen möglich. Bei folgenden Adreßvorwahlen ist die Negation möglich:

E	Digitaler Eingang
A	Digitaler Ausgang
M	1-Bit-Speicherstelle
F	Freigabe einer Zeit (1 Bit Signal zum Starten einer Zeit)
Z	Abfrage einer Zeit (1 Bit Signal einer abgelaufenen Zeit)

Die Negation kann nur gewählt werden, wenn sich der Cursor in der AWL-Eingabezeile auf dem Feld "Adreßvorwahl" befindet.

### Befehlsübersicht / Negation

MOTOROLA	B&R	MOTOROLA	B&R	MOTOROLA	B&R
ADCA	ADD	CPX#	VRK	ORAA	OD
ADCB	++B	CPY# <sup>1)</sup>	VYK <sup>1)</sup>	ORAB	OB
ADDA	+	DEC	DEC	PRS	PRS
ADDB	+B	EORA	EXO	ROL	SLI
ADDD	+D	EORB	EB	ROR	SRE
ANDA	UND	INC	INC	RST	RST
ANDB	UB	LDAA	LAD	SBCA	SUB
ASL	SL	LDAB	LB	SBCB	--B
BITA	B	LDD	LD	SET	SET
BITB	BB	LDK	LDK	STAA	=
CLR	CLR	LDX#	LRK	STAB	=B
CMPA	CMP	LDY <sup>1)</sup>	LYK <sup>1)</sup>	STD	=D
CMPB	VB	LSR	SR	SUBA	-
COM	K			SUBB	-B

<sup>1)</sup> nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar

# 3. AWL-BEFEHLE

## 3.1. LADEBEFEHLE

In diesem Abschnitt werden alle Befehle beschrieben, die Daten (1 oder 2 Byte) von einer angegebenen Speicherstelle in ein Register laden.

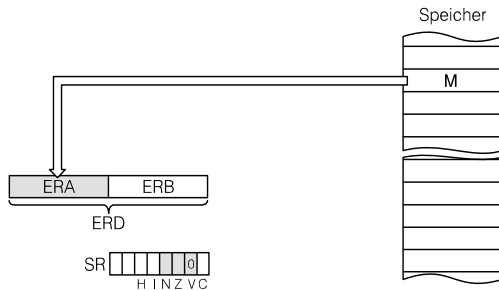
Motorola	B&R	Betriebsart			
		PG1000	PG-PC	CP 80	PC 80
LDAA	LAD	○	○	○	○
LDAB	LB	○	○	○	○
LDD	LD	○	○	○	○
LDK	LDK		○		○
LDL	LDL		○		○
LDX	LR	○	○	○	○
LDX#	L RK	○	○	○	○
LDXL	L RL	○	○	○	○
LDY	LY				○
LDY#	LYK				○
LDYL	LYL				○
LDS	LS	○	○	○	○
LEA!	LE!				○
LEAU	LEU				○
LEAX	LER			○	○
LEAY	LEY				○

Motorola	LDAA	Funktion														CPU Typ A / 6303										
B&R	LAD	(M) ⇒ ERA																								
Kurz	L																									
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	96	B6	86	A6		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	●	●	▼	●

Motorola	LDAA	Funktion														CPU Typ B / 6809											
B&R	LAD	(M) ⇒ ERA																									
Kurz	L																										
Adressierungsarten / Opcode													Adreßvorwahlen													<input type="radio"/> CP 80 <input type="radio"/> PC 80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
	96	B6	86	A6		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Das Ergebnisregister ERA wird mit dem Inhalt der Speicherstelle M geladen.

Beim Laden von 1 Bit Daten (bei den Adreßvorwahlen E, A, M, F, Z) enthält das Datenbit 0 von ERA die entsprechende Information. Die Datenbits 1 bis 7 erhalten den Wert NULL.

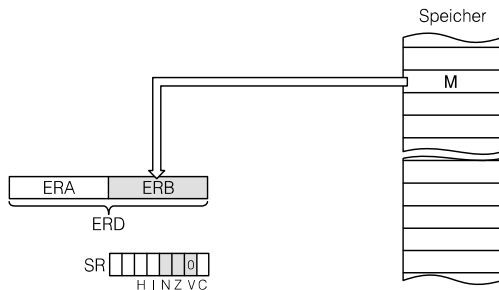


Motorola	LDAB	Funktion	CPU Typ A / 6303																							
B&R	LB	(M) ⇒ ERB																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D6	F6	C6	E6		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	LDAB	Funktion	CPU Typ B / 6809																							
B&R	LB	(M) ⇒ ERB																								
Kurz																										
Adressierungsarten / Opcode			<div><input type="radio"/> CP 80 <input type="radio"/> PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D6	F6	C6	E6		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Das Ergebnisregister ERB wird mit dem Inhalt der Speicherstelle M geladen.

Beim Laden von 1 Bit Daten (bei den Adreßvorwahlen E, A, M, F, Z) enthält das Datenbit 0 von ERB die entsprechende Information. Die Datenbits 1 bis 7 erhalten den Wert NULL.

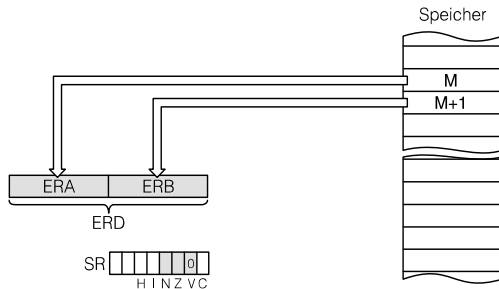


Motorola	LDD	Funktion	CPU Typ A / 6303																		
B&R	LD	(M:M+1) ⇨ ERD																			
Kurz																					
Adressierungsarten / Opcode			<div><input type="radio"/> PG1000</div> <div><input type="radio"/> PG-PC</div>																		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
	4/2	5/3	3/3	5/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H I N Z V C
	DC	FC	CC	EC		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0 ● ● ▼

Motorola	LDD	Funktion	CPU Typ B / 6809																							
B&R	LD	(M:M+1) ⇒ ERD																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div><div></div>CP 80</div> <div><div></div>PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	5/2	6/3	3/3	5+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	DC	FC	CC	EC		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Das Ergebnisregister ERD wird mit dem Inhalt der Speicherstellen M und M+1 geladen.

Beim Laden von 1 Bit Daten (bei den Adreßvorwahlen E, A, M, F, Z) enthält das Datenbit 0 von ERA den Inhalt der Speicherstelle M und das Bit 0 von ERB den Inhalt der Speicherstelle mit der nächsthöheren Adresse M+1. Die Datenbits 1 bis 7 erhalten jeweils den Wert NULL.





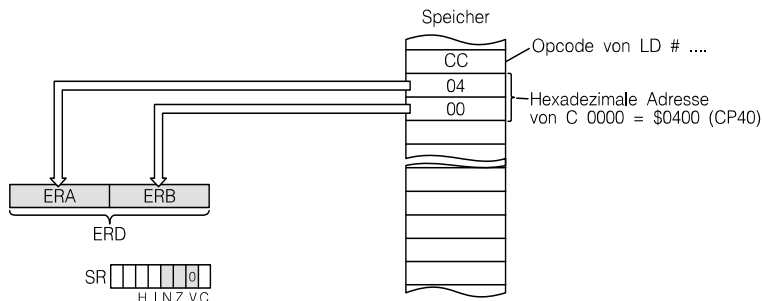
Motorola	LDK	Funktion	CPU Typ A / 6303																			
B&R	LDK	M ⇒ ERD																				
Kurz																						
Adressierungsarten / Opcode		Adreßvorwahlen	PG1000 PG-PC																			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister	
			3/3			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H I N Z V C	
			CC			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	LDK	Funktion	CPU Typ B / 6809																							
B&R	LDK																		M ⇨ ERD							
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen																CP 80 PC 80				
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
			3/3			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
			CC			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Das Ergebnisregister ERD wird mit der Adresse M geladen.

Der Befehl entspricht dem LD # xxxx Befehl. Die Adresse, die der Anwender in die AWL-Eingabezeile eingibt, ersetzt das PROgrammier-SYSTEM durch die effektive Adresse (hexadezimaler Wert), die die B&R Adresse (= Adreßvorwahl + Adreßteil) im Speicher der SPS hat.

**Beispiel:** LDK C 0000 (in einer CP40)



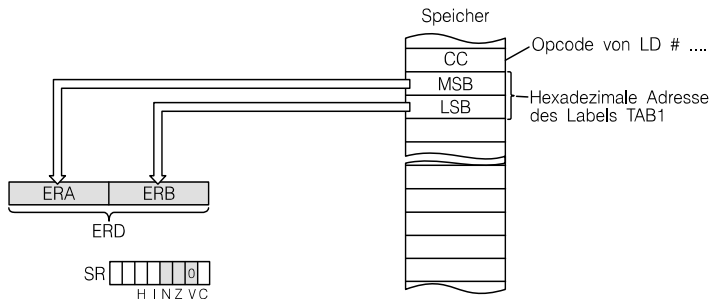
Motorola	LDL	Funktion														CPU Typ A / 6303										
B&R	LDL	M ⇒ ERD																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														PG1000 PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
			3/3			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
			CC																							

Motorola	LDL	Funktion														CPU Typ B / 6809										
B&R	LDL	M ⇨ ERD																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														CP 80 ○ PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
			3/3			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
			CC																							

Das Ergebnisregister ERD wird mit der Adresse M eines Labels geladen.

Der Befehl entspricht dem LD # xxxx Befehl. Der Anwender kann jedoch nur einen Label-Namen in die AWL-Eingabezeile eingeben. Das PROgrammierSYStem ersetzt diesen Namen durch die effektive Adresse (hexadezimaler Wert), die der Label im Speicher der SPS hat.

**Beispiel:** LDL TAB1

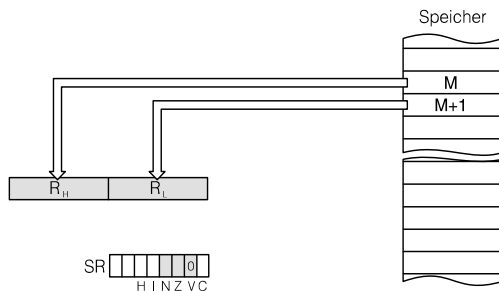


**Anmerkung:** Wird für den Label ein Tabellenname eingegeben, enthält das Ergebnisregister ERD nach der Ausführung des Befehles die Adresse des Tabellen-Headers. Das erste Datenbyte ist effektiv in der Speicherstelle mit der Adresse ERD + 15 gespeichert (siehe Bedienerhandbuch PROgrammierSYStem, Kapitel 7 Tabellen-Editor).

Motorola	LDX	Funktion														CPU Typ A / 6303										
B&R	LR	(M:M+1) ⇒ R																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	3/3	5/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	DE	FE	CE	EE																						

Motorola	LDX	Funktion	CPU Typ B / 6809																							
B&R	LR	(M:M+1) ⇒ R																								
Kurz																										
Adressierungsarten / Opcode			<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	5/2	6/3	3/3	5+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	9E	BE	8E	AE								◁	◁	◁	●	●	●	●	●	●	●	●	●	●	▼	◁

Das Indexregister R (X) wird mit dem Inhalt der Speicherstellen M und M+1 geladen.



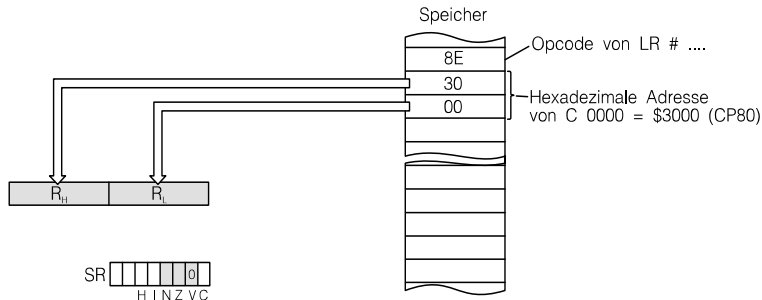
Motorola	LDX#	Funktion														CPU Typ A / 6303										
B&R	L R K	M ⇨ R																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000						
																				<input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
			3/3			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
			CE			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	LDX#	Funktion	CPU Typ B / 6809																							
B&R	L R K	M ⇒ R																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div><div></div>CP 80</div> <div><div></div>PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
			3/3			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
			8E			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Das Indexregister R (X) wird mit der Adresse M geladen.

Der Befehl entspricht dem LR # xxxx Befehl. Die Adresse, die der Anwender in die AWL-Eingabezeile eingibt, ersetzt das PROgrammier-SYSTEM durch die effektive Adresse (hexadezimaler Wert), die die B&R Adresse (= Adreßvorwahl + Adreßteil) im Speicher der SPS hat.

**Beispiel:** L RK C 0000 (in einer CP80  $\Rightarrow$  CPU Typ B / 6809)



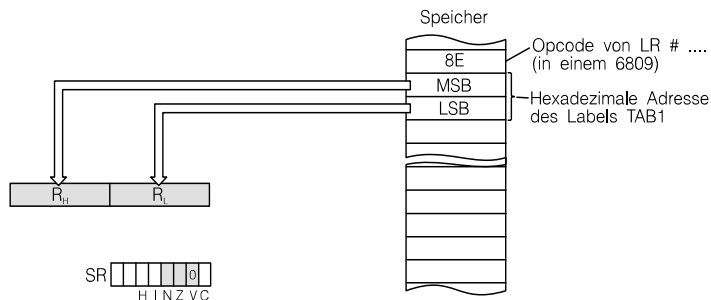
Motorola	LDXL	Funktion	CPU Typ A / 6303																							
B&R	LRL	M ⇨ R																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
			3/3			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
			CE																							

Motorola	LDXL	Funktion	CPU Typ B / 6809																							
B&R	LRL	M ⇨ R																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
			3/3			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
			8E																							
																					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Das Indexregister R (X) wird mit der Adresse M eines Labels geladen.

Der Befehl entspricht dem LR # xxxx Befehl. Der Anwender kann jedoch nur einen Label-Namen in die AWL-Eingabezeile eingeben. Das PROgrammierSYStem ersetzt diesen Namen durch die effektive Adresse (hexadezimaler Wert), die der Label im Speicher der SPS hat.

**Beispiel:** LRL TAB1

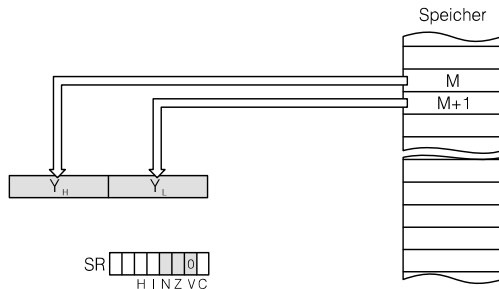


**Anmerkung:** Wird für den Label ein Tabellenname eingegeben, enthält das Indexregister R (X) nach der Ausführung des Befehles die Adresse des Tabellen-Headers. Das erste Datenbyte ist effektiv in der Speicherstelle mit der Adresse ERD + 15 gespeichert (siehe Bedienerhandbuch PROgrammierSYStem, Kapitel 7 Tabellen-Editor).

Motorola		Funktion										CPU Typ A / 6303									
B&R																					
Kurz																					
Adressierungsarten / Opcode						Adreßvorwahlen										PG1000					
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H

Motorola		Funktion										CPU Typ B / 6809									
B&R		LDY										(M:M+1) ⇨ Y									
Kurz		LY																			
Adressierungsarten / Opcode						Adreßvorwahlen										CP 80					
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H

Das Indexregister Y wird mit dem Inhalt der angegebenen Speicherstellen M und M+1 geladen.



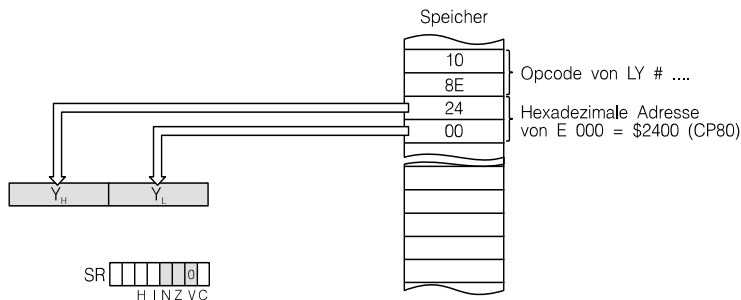
Motorola		Funktion														CPU Typ A / 6303											
B&R																											
Kurz																											
Adressierungsarten / Opcode						Adreßvorwahlen														PG1000							
																				PG-PC							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
							E	A	M	F	Z <td>#</td> <td>P</td> <td>C</td> <td>I</td> <td>Y</td> <td>D<td>U<th>I</th><th>B</th><th>G</th><th>H</th><th>I</th><th>N</th><th>Z</th><th>V</th><th>C</th></td></td>	#	P	C	I	Y	D <td>U<th>I</th><th>B</th><th>G</th><th>H</th><th>I</th><th>N</th><th>Z</th><th>V</th><th>C</th></td>	U <th>I</th> <th>B</th> <th>G</th> <th>H</th> <th>I</th> <th>N</th> <th>Z</th> <th>V</th> <th>C</th>	I	B	G	H	I	N	Z	V	C

Motorola	LDY#	Funktion														CPU Typ B / 6809														
B&R	LYK	M ⇒ Y																												
Kurz																														
Adressierungsarten / Opcode						Adreßvorwahlen														<div>CP 80</div> <div>PC 80</div>										
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister									
			4/4			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C				
			10 8E			●	●	●	●	●		●	●						●	●	○	●	●	▼	○					

Das Indexregister Y wird mit der Adresse M geladen.

Der Befehl entspricht dem LY # xxxx Befehl. Die Adresse, die der Anwender in die AWL-Eingabezeile eingibt, ersetzt das PROgrammier-SYStem durch die effektive Adresse (hexadezimaler Wert), die die B&R Adresse (= Adreßvorwahl + Adreßteil) im Speicher der SPS hat.

**Beispiel:**      **LYK**      **E 000**      (in einer CP80 = CPU Typ B / 6809)



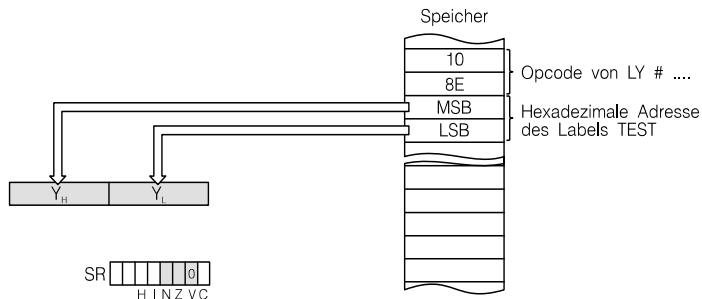
Motorola		Funktion																CPU Typ A / 6303								
B&R																										
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														PG1000						
																				PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C

Motorola	LDYL	Funktion														CPU Typ B / 6809					
B&R	LYL	M ⇨ Y																			
Kurz																					
Adressierungsarten / Opcode						Adreßvorwahlen														CP 80	
																				○ PC 80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
			4/4																		
			10 8E				E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G

Das Indexregister Y wird mit der Adresse M eines Labels geladen.

Der Befehl entspricht dem LY # xxxx Befehl. Der Anwender kann jedoch nur einen Label-Namen in die AWL-Eingabezeile eingeben. Das PROgrammierSYStem ersetzt diesen Namen durch die effektive Adresse (hexadezimaler Wert), die der Label im Speicher der SPS hat.

**Beispiel:**      LYL      TEST



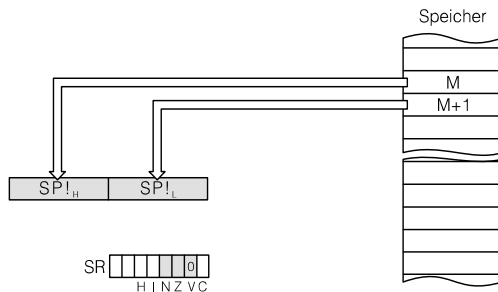
**Anmerkung:** Wird für den Label ein Tabellenname eingegeben, enthält das Indexregister Y nach der Ausführung des Befehles die Adresse des Tabellen-Headers. Das erste Datenbyte ist effektiv in der Speicherstelle mit der Adresse ERD + 15 gespeichert (siehe Bedienerhandbuch PROgrammierSYStem, Kapitel 7 Tabellen-Editor).



Motorola	LDS	Funktion														CPU Typ A / 6303										
B&R	LS	(M:M+1) ⇒ SP!																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	3/3	5/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	9E	BE	8E	AE																						

Motorola	LDS	Funktion														CPU Typ B / 6809										
B&R	LS	(M:M+1) ⇒ SP!																								
Kurz																										
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/>	CP 80	
																								<input type="radio"/>	PC 80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	6/3	7/4	4/4	6+/3+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	10 DE	10 FE	10 CE	10 EE																						

Der System-Stapelzeiger SP! wird mit dem Inhalt der Speicherstellen M und M+1 geladen.



Motorola		Funktion															CPU Typ A / 6303										
B&R																											
Kurz																											
Adressierungsarten / Opcode							Adreßvorwahlen															PG1000		Statusregister			
																							PG-PC				
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.		I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G						
							E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C

Motorola	LEA!	Funktion															CPU Typ B / 6809											
B&R	LE!	EA ⇒ SP!																										
Kurz																												
Adressierungsarten / Opcode							Adreßvorwahlen															CP 80						
																						○	PC 80					
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.		I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
			4+/2+				E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
			32													●	●	●	●			○	○	○	○	○	○	

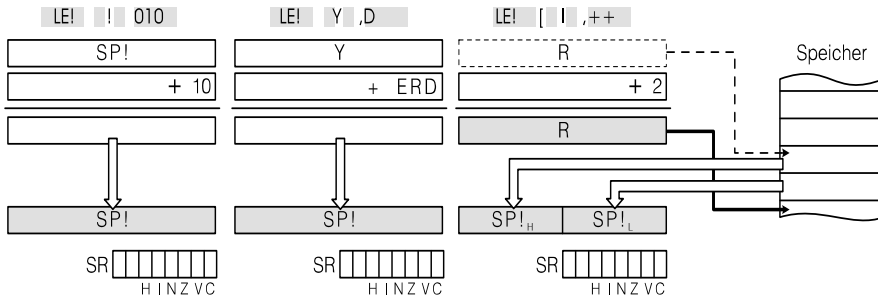
Die effektive Adresse EA wird aus der beim Befehl angegebenen indizierten Adressierung errechnet und im System-Stapelzeiger SP! gespeichert.

**Beispiele:**

**LE!    !    010**                       $SP! + 10 \Rightarrow SP!$

**LE!    Y ,D**                               $Y + ERD \Rightarrow SP!$

**LE!    [ I ,++**                             $(R:R+1) \Rightarrow SP!; R + 2 \Rightarrow R$



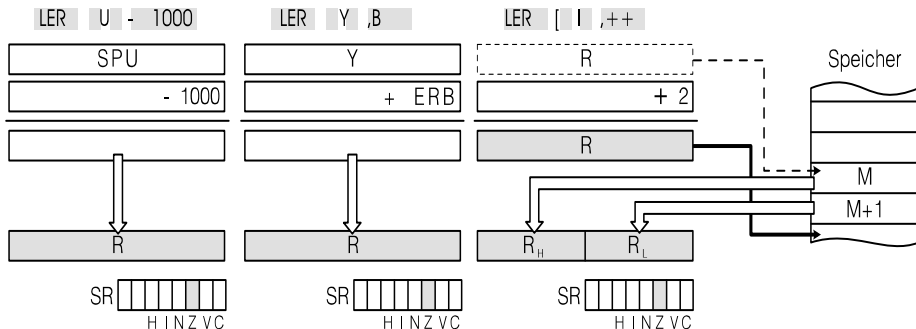


Motorola		Funktion															CPU Typ A / 6303									
B&R																										
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														PG1000 PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C

Motorola	LEAX	Funktion															CPU Typ B / 6809										
B&R	LER	EA ⇒ R																									
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen										<input type="radio"/> CP 80 <input type="radio"/> PC 80					
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
			4+/2+			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
			30													○	●	●	●			○	○	○	●	○	○

Die effektive Adresse EA wird aus der beim Befehl angegebenen indizierten Adressierung errechnet und im Indexregister R (X) gespeichert.

**Beispiele:** LER U - 1000 SPU - 1000  $\Rightarrow$  R  
LER Y ,B Y + B  $\Rightarrow$  R  
LER [ I ,++ (R:R+1)  $\Rightarrow$  R; R + 2 R



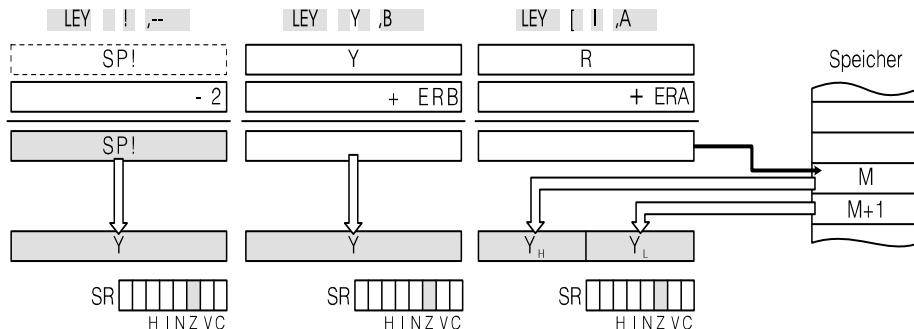
Motorola		Funktion														CPU Typ A / 6303											
B&R																											
Kurz																											
Adressierungsarten / Opcode						Adreßvorwahlen														PG1000							
																				PG-PC							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	

Motorola	LEAY	Funktion										CPU Typ B / 6809														
B&R	LEY	EA ⇒ Y																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen										CP 80		PC 80								
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
			4+/2+			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
			31											●	●		●	●			○	○	○	○	○	○

Die effektive Adresse EA wird aus der beim Befehl angegebenen indizierten Adressierung errechnet und im Indexregister Y gespeichert.

**Beispiele:**

- LEY I ,--** SP! -2  $\Rightarrow$  SP!; SP!  $\Rightarrow$  Y
- LEY Y ,B** Y + B  $\Rightarrow$  Y
- LEY [ I ,A** (R+ERA:R+ERA+1)  $\Rightarrow$  Y





## 3.2. SPEICHERBEFEHLE

In diesem Abschnitt werden alle Befehle beschrieben, die Daten (1 oder 2 Byte) aus einem Register in einer angegebenen Speicherstelle speichern.

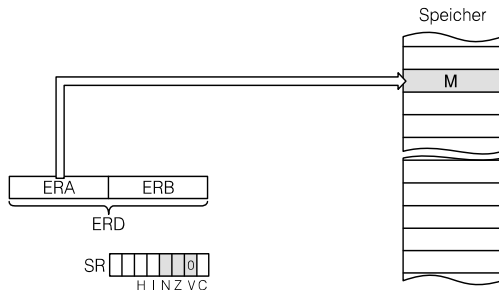
Motorola	B&R	Betriebsart			
		PG1000	PG-PC	CP 80	PC 80
STAA	=	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
STAB	=B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
STD	=D	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
STX	=R	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
STY	=Y	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
STS	=S	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Motorola	STAA	Funktion														CPU Typ A / 6303										
B&R	=	ERA ⇨ (M)																								
Kurz	I																									
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3		4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	97	B7		A7		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				<input type="radio"/>	<input type="radio"/>	<input type="radio"/>							<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Motorola	STAA	Funktion	CPU Typ B / 6809																							
B&R	=	ERA ⇨ (M)																								
Kurz	I																									
Adressierungsarten / Opcode		Adreßvorwahlen	<div><input type="radio"/> CP 80</div> <div><input type="radio"/> PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3		4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	97	B7		A7			0	0	0			0	0	0							0	0	0	0	0	0

Der Inhalt des Ergebnisregisters ERA wird in der Speicherstelle mit der Adresse M gespeichert. ERA wird dabei nicht verändert.

Ist die Zieladresse eine 1 Bit-Speicherstelle (A, M, F), wird nur das Datenbit 0 von ERA gespeichert.



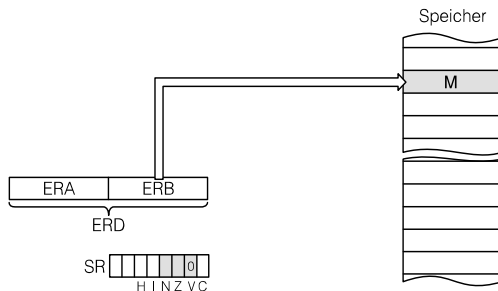


Motorola	STAB	Funktion														CPU Typ A / 6303											
B&R	=B	ERB ⇨ (M)																									
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> PG1000 <input type="radio"/> PG-PC			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
	3/2	4/3		4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
	D7	F7		E7		0	0	0				0	0	0						0	0	0	0	0	0	0	0

Motorola	STAB	Funktion														CPU Typ B / 6809											
B&R	=B	ERB ⇨ (M)																									
Kurz																											
Adressierungsarten / Opcode											Adreßvorwahlen											<input type="radio"/> CP 80 <input type="radio"/> PC 80					
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
	4/2	5/3		4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
	D7	F7		E7		0	0	0				0	0	0	●	●	●	●	●	●	0	●	0	●	●	▼	0

Der Inhalt des Ergebnisregisters ERB wird in der Speicherstelle mit der Adresse M gespeichert. ERB wird dabei nicht verändert.

Ist die Zieladresse eine 1 Bit-Speicherstelle (A, M, F), wird nur das Datenbit 0 von ERB gespeichert.

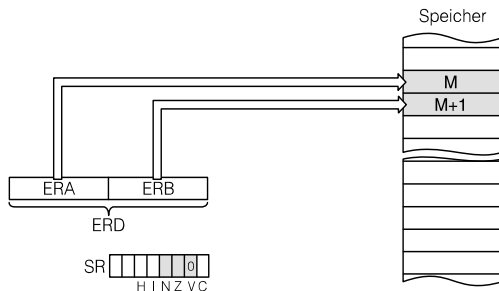


Motorola	STAD	Funktion	CPU Typ A / 6303																							
B&R	=D	ERD $\Rightarrow$ (M:M+1)																								
Kurz																										
Adressierungsarten / Opcode			<div><input type="radio"/> PG1000</div> <div><input type="radio"/> PG-PC</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3		5/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	DD	FD		ED		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Motorola	STAD	Funktion	CPU Typ B / 6809																							
B&R	=D	ERD $\Rightarrow$ (M:M+1)																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div><input type="radio"/> CP 80</div> <div><input type="radio"/> PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	5/2	6/3		5+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	DD	FD		ED		<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>	<div><input type="radio"/></div>

Der Inhalt des Ergebnisregisters ERD wird in den Speicherstellen mit den Adressen M und M+1 gespeichert. ERD wird dabei nicht verändert.

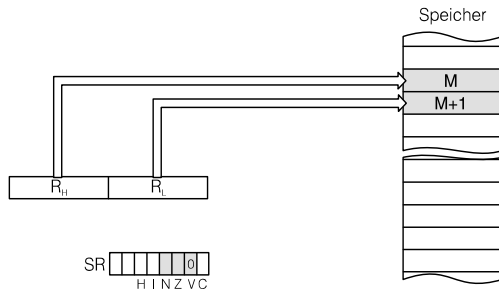
Ist die Zieladresse eine 1 Bit-Speicherstelle (A, M, F), wird das Datenbit 0 von ERA in der Speicherstelle M und das Datenbit 0 von ERB in der Speicherstelle M+1 gespeichert.



Motorola	STX	Funktion														CPU Typ A / 6303										
B&R	=R	R ⇒ (M:M+1)																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000	<input type="radio"/> PG-PC					
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
	4/2	5/3		5/2		E	A	M	F	Z	#	P	C	I	Y	D <td>U<td>!<td>B</td><td>G</td><td>H</td><td>I</td><td>N</td><td>Z</td><td>V</td><td>C</td></td></td>	U <td>!<td>B</td><td>G</td><td>H</td><td>I</td><td>N</td><td>Z</td><td>V</td><td>C</td></td>	! <td>B</td> <td>G</td> <td>H</td> <td>I</td> <td>N</td> <td>Z</td> <td>V</td> <td>C</td>	B	G	H	I	N	Z	V	C
	DF	FF		EF																						

Motorola	STX	Funktion														CPU Typ B / 6809										
B&R	=R	R ⇒ (M:M+1)																								
Kurz																										
Adressierungsarten / Opcode												Adreßvorwahlen										<input type="radio"/> CP 80 <input type="radio"/> PC 80				
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
	5/2	6/3		5+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
	9F	BF		AF																						

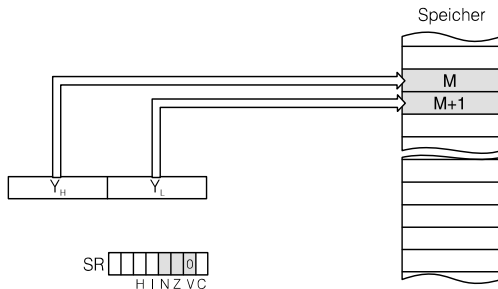
Der Inhalt des Indexregisters R (X) wird in den Speicherstellen mit den Adressen M und M+1 gespeichert. R (X) wird dabei nicht verändert.



Motorola		Funktion										CPU Typ A / 6303									
B&R																					
Kurz																					
Adressierungsarten / Opcode						Adreßvorwahlen						PG1000									
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H
																					I
																					N
																					Z
																					V
																					C

Motorola		Funktion										CPU Typ B / 6809									
B&R		=Y										Y ⇔ (M:M+1)									
Kurz																					
Adressierungsarten / Opcode						Adreßvorwahlen						CP 80									
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H
	6/3	7/4		6+/3+																	I
	10 9F	10 BF		10 AF																	N
																					Z
																					V
																					C

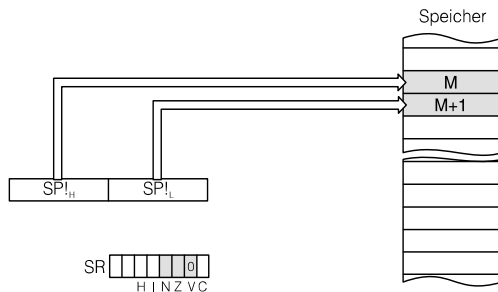
Der Inhalt des Indexregisters Y wird in den Speicherstellen mit den Adressen M und M+1 gespeichert. Y wird dabei nicht verändert.



Motorola	STS	Funktion	CPU Typ A / 6303																							
B&R	=S	SP! ⇨ (M:M+1)																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3		5/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	9F	BF		AF																						

Motorola	STS	Funktion	CPU Typ B / 6809																							
B&R	=S	SP! ⇨ (M:M+1)																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div><input type="radio"/> CP 80</div> <div><input type="radio"/> PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	6/3	7/4		6+/3+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	10 DF	10 FF		10 EF																						

Der Inhalt des System-Stapelzeigers SP! wird in den Speicherstellen mit den Adressen M und M+1 gespeichert. SP! wird dabei nicht verändert.





### 3.3. DATENAUSTAUSCH ZWISCHEN REGISTERN

In diesem Abschnitt werden alle Befehle beschrieben, die Daten (1 oder 2 Byte) aus einem Register in einem anderen speichern oder zwischen zwei Registern austauschen.

Motorola	B&R	Betriebsart			
		PG1000	PG-PC	CP 80	PC 80
TAB	MAB	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
TBA	MBA	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
TAP	MAC	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
TPA	MCA	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
TSX	MSR	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
TXS	MRS	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
XGDX	DXR	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
TFR	TFR				<input type="radio"/>
EXG	EXG				<input type="radio"/>

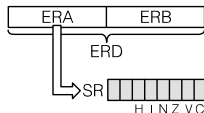






Motorola	TAP	Funktion																CPU Typ A / 6303								
B&R	MAC	ERA ⇨ SR																								
Kurz																										
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> PG1000		
																								<input type="radio"/> PG-PC		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
06																						•	•	•	•	•

Der Inhalt des Ergebnisregisters ERA wird in das Statusregister SR kopiert.



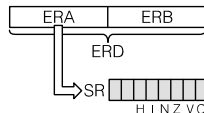
#### Hinweis:

Wird das Interrupt Inhibit Bit I (Bit 4 des Statusregisters) vom Anwender auf log. 1 gesetzt, werden alle Interrupts gesperrt. Diese Technik wird dann verwendet, wenn Programnteile des Anwenderprogrammes in keinem Fall durch einen Interrupt unterbrochen werden dürfen.

**Bevor der Befehl END abgearbeitet wird, muß das Bit I unbedingt wieder auf log. 0 gesetzt werden.**

Motorola	TAP	Funktion																CPU Typ B / 6809								
B&R	MAC	ERA ⇨ SR																								
Kurz																										
Adressierungsarten / Opcode										Adreßvorwahlen										<input type="radio"/> CP 80						
																				<input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
6/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
1F 8A																						●	●	●	●	●

Der Inhalt des Ergebnisregisters ERA wird in das Statusregister SR kopiert.



#### Hinweis:

Wird das Interrupt Inhibit Bit I (Bit 4 des Statusregisters) vom Anwender auf log. 1 gesetzt, werden alle Interrupts gesperrt. Diese Technik wird dann verwendet, wenn Programnteile des Anwenderprogrammes in keinem Fall durch einen Interrupt unterbrochen werden dürfen.

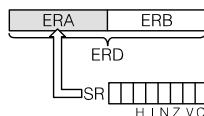
**Bevor der Befehl END abgearbeitet wird, muß das Bit I unbedingt wieder auf log. 0 gesetzt werden.**

**Bit 7 und 6 des Statusregisters dürfen vom Anwender nicht verändert werden.**

Motorola	TPA	Funktion														CPU Typ A / 6303										
B&R	MCA	SR ⇨ ERA																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
07																										

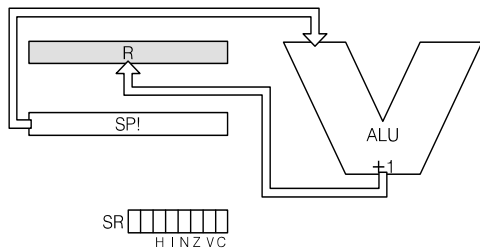
Motorola	TPA	Funktion														CPU Typ B / 6809										
B&R	MCA	SR ⇔ ERA																								
Kurz																										
Adressierungsarten / Opcode										Adreßvorwahlen										<input type="radio"/> CP 80						
																				<input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
6/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
1F A8																										

Der Inhalt des Statusregisters SR wird in das Ergebnisregister ERA kopiert.



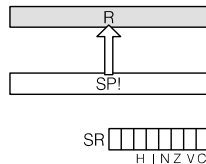
Motorola	TSX	Funktion														CPU Typ A / 6303											
B&R	MSR	SP! + 1 ⇨ R																									
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> PG1000 <input type="radio"/> PG-PC			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
30																											

Das Indexregister R (X) wird mit der Adresse der Speicherstelle geladen, die den letzten gültigen Wert des Stapels enthält. Da im Prozessor 6303 der Stapelzeiger auf den nächsten freien Platz im Stapel zeigt, wird der Wert 1 zum System-Stapelzeiger SP! addiert und das Ergebnis in R (X) gespeichert.



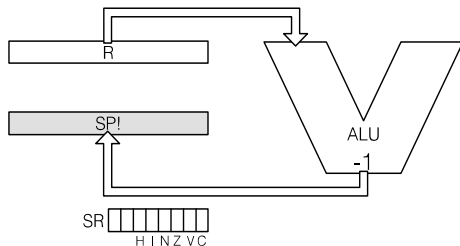
Motorola	TSX	Funktion	CPU Typ B / 6809																		
B&R	MSR	SP! ⇨ R																			
Kurz																					
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
6/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H I N Z V C
1F 41																					⌋ ⌋ ⌋ ⌋ ⌋ ⌋

Das Indexregister R (X) wird mit der Adresse der Speicherstelle geladen, die den letzten gültigen Wert des Stapels enthält. Da im Prozessor 6809 der Stapelzeiger auf den letzten gültigen Wert im Stapel zeigt, muß keine Korrektur vorgenommen werden, wie es beim 6303 notwendig ist, d.h. SP! wird nach R (X) kopiert.



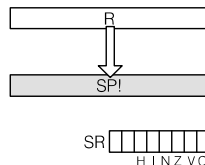
Motorola	TXS	Funktion														CPU Typ A / 6303											
B&R	MRS	R - 1 ⇒ SP!																									
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen														<input type="radio"/> PG1000	
																										<input type="radio"/> PG-PC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
35																											

Dieser Befehl ist die Umkehrung zum Befehl MSR (TSX). Der System-Stapelzeiger wird mit dem um 1 verminderten Inhalt des Indexregisters R (X) geladen.



Motorola	TXS	Funktion														CPU Typ B / 6809					
B&R	MRS	R ⇒ SP!																			
Kurz																					
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80	
																				<input type="radio"/> PC 80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
6/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H I N Z V C
1F 14																					⌋ ⌋ ⌋ ⌋

Dieser Befehl ist die Umkehrung zum Befehl MSR (TSX). Der System-Stapelzeiger wird mit dem Inhalt des Indexregisters R (X) geladen.

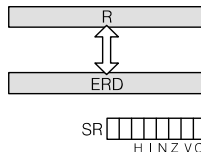


**Achtung:** Der System-Stapelzeiger SP! darf nur bei genauer Kenntnis der Abläufe von Stapeloperationen verändert werden. Wird SP! nicht vorschriftsmäßig verändert, kann es zu einem Fehler in der CPU kommen. Dies bewirkt einen HALT in der CPU; im Statustest wird die Fehlermeldung "STACKFEHLER" ausgegeben.

Motorola	XGDX	Funktion																CPU Typ A / 6303								
B&R	DXR	ERD ⇔ R																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000						
																				<input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
18																										

Motorola	XGDX	Funktion																CPU Typ B / 6809								
B&R	DXR	ERD ⇔ R																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80						
																				<input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
7/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
1E 10																										

Der Inhalt des Ergebnisregisters ERD wird mit dem Inhalt des Indexregisters R (X) vertauscht.



Motorola	Funktion															CPU Typ A / 6303
B&R																
Kurz																
Adressierungsarten / Opcode						Adreßvorwahlen										PG1000
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	PG-PC
						E	A	M	F	Z	#	P	C	I	Y	Statusregister

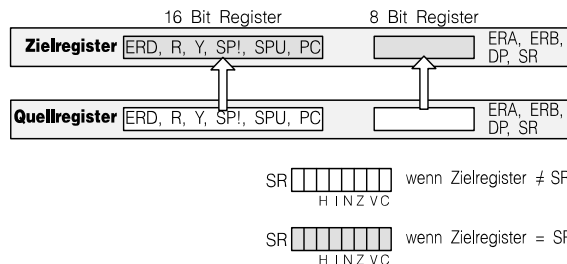
Motorola	TFR	Funktion															CPU Typ B / 6809				
B&R	TFR	$r_{\text{Quelle}} \Rightarrow r_{\text{Ziel}}$																			
Kurz																					
Adressierungsarten / Opcode						Adreßvorwahlen											<input type="radio"/> CP 80				
																	<input type="radio"/> PC 80				
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
6/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H I N Z V C
1F																					• • • • •

Der Inhalt des angegebenen Quellregisters wird in das Zielregister kopiert. Quell- und Zielregister müssen die selbe Größe haben, d.h. entweder 1 Byte Register (ERA, ERB, DP, SR) oder 2 Byte Register (ERD, R, Y, SPI, SPU, PC).

**Syntax:** **TFR** **r, r-** Zielregister  
 Quellregister

Der ganze Befehl besteht aus zwei Byte. Das erste Byte ist der Opcode (Befehl). Im zweiten Byte (Postbyte) wird angegeben, welches Register das Quell- bzw Zielregister ist.

Ziel-/Quellregister	r	Postbyte	
		Quelle	Ziel
ERD	D	0000	0000
R(X)	I(X)	0001	0001
Y	Y	0010	0010
SPU	U	0011	0011
SPI	!	0100	0100
PC	\$	0101	0101
ERA	A	1000	1000
ERB	B	1001	1001
DP	P	1011	1011
SR	C	1010	1010



**Achtung:** Der System-Stapelzeiger SPI darf nur bei genauer Kenntnis der Abläufe von Stapeloperationen verändert werden. Wird SPI nicht vorschriftsmäßig verändert, kann es zu einem Fehler in der CPU kommen. Dies bewirkt einen HALT in der CPU; im Statustest wird die Fehlermeldung "STACKFEHLER" ausgegeben. Die Veränderung des Programmzählers PC bewirkt einen unbedingten Sprung an die Adresse, mit der PC geladen wurde.

**Achtung:** Der System-Stapelzeiger SP! darf nur bei genauer Kenntnis der Abläufe von Stapeloperationen verändert werden. Wird SP! nicht vorschriftsmäßig verändert, kann es zu einem Fehler in der CPU kommen. Dies bewirkt einen HALT in der CPU; im Statustest wird die Fehlermeldung "STACKFEHLER" ausgegeben.  
Die Veränderung des Programmzählers PC bewirkt einen unbedingten Sprung an die Adresse, mit der PC geladen wurde.



## 3.4. STAPELOPERATIONEN

In diesem Abschnitt werden alle Befehle beschrieben, die den Inhalt von Registern (1 oder 2 Byte) auf dem Stapel ablegen bzw. vom Stapel holen.

Motorola	B&R	Betriebsart			
		PG1000	PG-PC	CP 80	PC 80
PSH	PSH				○
PUL	PUL				○
PSHA	ANS	○	○	○	○
PULA	AVS	○	○	○	○
PSHB	BNS	○	○	○	○
PULB	BVS	○	○	○	○
PSHX	RNS	○	○	○	○
PULX	RVS	○	○	○	○

Motorola		Funktion														CPU Typ A / 6303											
B&R																											
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen														PG1000	
																										PG-PC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	

Motorola	PSH	Funktion	CPU Typ B / 6809																							
B&R	PSH	Inhalt der angegebenen Register auf System- oder Anwender-Stapel speichern.																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div>CP 80</div> <div>PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
54/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
34 / 36																										

Mit diesem Befehl können einzelne oder mehrere vom Anwender angegebene Register auf dem Stapel (System- oder Anwender-Stapel) abgelegt werden.

**Syntax:** **PSH** **s** **,r.....** — Register, die auf den Stapel gelegt werden sollen  
 — Stapel, auf dem die Register abgelegt werden sollen

Der ganze Befehl besteht aus zwei Byte. Das erste Byte (Opcode) gibt an, welcher Stapel angesprochen wird. Im zweiten Byte (Postbyte) werden die Register definiert, die auf dem Stapel abgelegt werden sollen. Werden mehrere Register im Postbyte angegeben, werden diese in einer bestimmten Reihenfolge auf dem Stapel abgelegt. Die Pfeile in den Tabellen geben diese Reihenfolge an.

s = ! (Opcode = \$34) Ablegen der Register auf SP!		
Register	r	Postbyte
SR	C	xxxxxxx1
ERA	A	xxxxxxx1x
ERB	B	xxxxxxx1xx
DP	P	xxxxxxx1xxx
R (X)	I (X)	xxxxxxx1xxxx
Y	Y	xxxxxxx1xxxxx
SPU	U	xxxxxxx1xxxxxx
PC	\$	xxxxxxx1xxxxxxx
Alle außer PC	*	01111111

s = U (Opcode = \$36) Ablegen der Register auf SPU		
Register	r	Postbyte
SR	C	xxxxxxx1
ERA	A	xxxxxxx1x
ERB	B	xxxxxxx1xx
DP	P	xxxxxxx1xxx
R (X)	I (X)	xxxxxxx1xxxx
Y	Y	xxxxxxx1xxxxx
SP!	!	xxxxxxx1xxxxxx
PC	\$	xxxxxxx1xxxxxxx
Alle außer PC	*	01111111

Im **Postbyte** können beliebige Register kombiniert werden.

**Beispiele:**     **PSH    ! ,ABY**     Dieser Befehl bewirkt, daß die Register ERA, ERB und Y auf dem System-Stapel abgelegt werden.

**Eingabe von...**

**PSH    ! ,\***

**PSH    U ,\***

**PSH    U ,IBC**

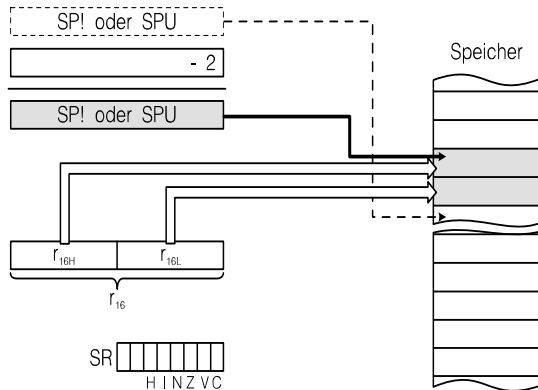
**PROgrammierSYstem ersetzt durch...**

**PSH    ! ,CABPIYU**

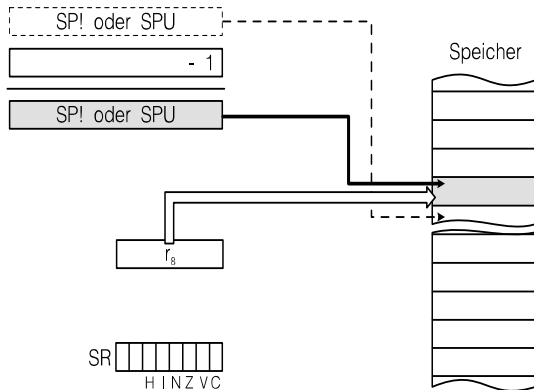
**PSH    U ,CABPIY!**

**PSH    U ,CBI**

Für die 16 Bit Register R (X) Y, SPU/SP!, PC:  
Für jedes der angegebenen Register gilt folgender Datenfluß:



Für die 8 Bit Register SR, ERA, ERB, DP



Motorola		Funktion																CPU Typ A / 6303									
B&R																											
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen														PG1000	
																										PG-PC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	

Motorola	PUL	Funktion										CPU Typ B / 6809											
B&R	PUL	Angegebene Register mit Daten vom System- oder Anwender-Stapel laden.																					
Kurz																							
Adressierungsarten / Opcode						Adreßvorwahlen												CP 80					
																<input type="radio"/>	PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister		
5*/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H   I   N   Z   V   C		
35 / 37																						• • • • • • • • • •	

Mit diesem Befehl können einzelne bzw. mehrere Register mit Daten vom Stapel (System- oder Anwender-Stapel) geladen werden.

**Syntax:** **PUL** **S** **,r.....** — Register, die mit Daten vom Stapel geladen werden sollen  
 — Stapel, von dem die Daten geholt werden sollen

Der ganze Befehl besteht aus zwei Byte. Das erste Byte (Opcode) gibt an, welcher Stapel angesprochen wird. Im zweiten Byte (Postbyte) werden die Register definiert, die mit Daten vom Stapel geladen werden sollen. Werden mehrere Register im Postbyte angegeben, werden diese in einer bestimmten Reihenfolge mit Daten vom Stapel geladen. Die Pfeile in den Tabellen geben diese Reihenfolge an.

s = ! (Opcode = \$35) Laden der Register von SP!		
Register	r	Postbyte
SR	C	xxxxxxx1
ERA	A	xxxxxxx1x
ERB	B	xxxxxxx1xx
DP	P	xxxx1xxxx
R (X)	I (X)	xxx1xxxx
Y	Y	xx1xxxxx
SPU	U	x1xxxxxxx
PC	\$	1xxxxxxx
Alle außer PC	*	01111111

s = U (Opcode = \$37) Laden der Register von SPU		
Register	r	Postbyte
SR	C	xxxxxxx1
ERA	A	xxxxxxx1x
ERB	B	xxxxxxx1xx
DP	P	xxxx1xxxx
R (X)	I (X)	xxx1xxxx
Y	Y	xx1xxxxx
SP!	!	x1xxxxxxx
PC	\$	1xxxxxxx
Alle außer PC	*	01111111

Im **Postbyte** können beliebige Register kombiniert werden.

**Beispiele:**     **PUL**     **I** , **CPI**     Dieser Befehl bewirkt, daß die Register SR, DP und R (X) mit Daten vom System-Stapel geladen werden.

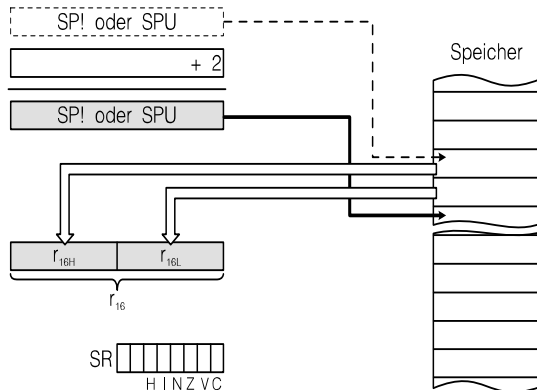
**Eingabe von...**

**PUL**     **I** , **\***  
**PUL**     **U** , **\***  
**PUL**     **U** , **IBC**

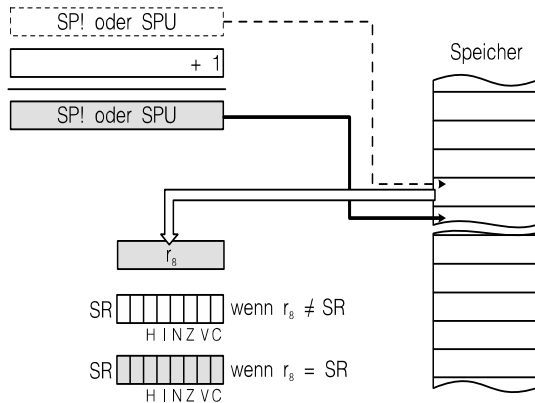
**PROgrammierSYstem ersetzt durch...**

**PSH**     **I** , **CABPIYU**  
**PSH**     **U** , **CABPIY!**  
**PSH**     **U** , **CBI**

Für die 16 Bit Register R (X), Y, SPU/SP!, PC  
 Für jedes der angegebenen Register gilt folgender Datenfluß:



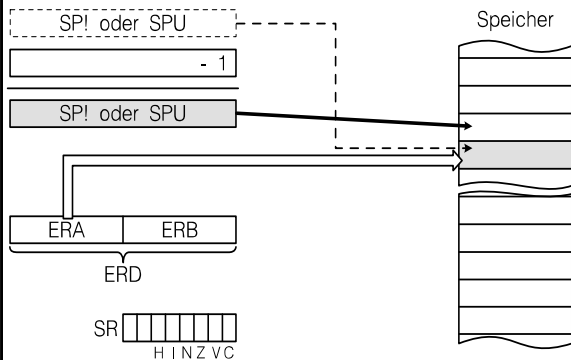
Für die 8 Bit Register SR, ERA, ERB, DP



Motorola	PSHA	Funktion														CPU Typ A / 6303											
B&R	ANS	ERA ⇒ (SP!); SP! - 1 ⇒ SP!																									
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> PG1000 <input type="radio"/> PG-PC			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister						
4/1						E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C	
36																											

Der Inhalt des Ergebnisregisters ERA wird in der Speicherstelle gespeichert, auf die der System-Stapelzeiger SP! zeigt.

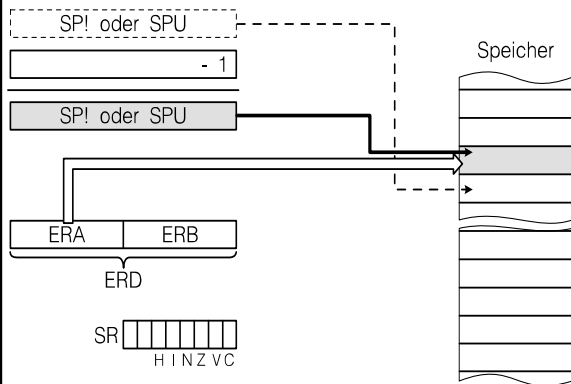
Danach wird SP! um den Wert 1 vermindert, damit SP! wieder auf die nächste freie Stelle des Stapels zeigt.



Motorola	PSHA	Funktion	CPU Typ B / 6809																							
B&R	ANS	SP! - 1 ⇒ SP!; ERA ⇒ (SP!)																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
6/2						E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
34 02																										

Da der System-Stapelzeiger SP! auf die letzte belegte Speicherstelle des Stapels zeigt, muß SP! zuerst um den Wert 1 vermindert werden.

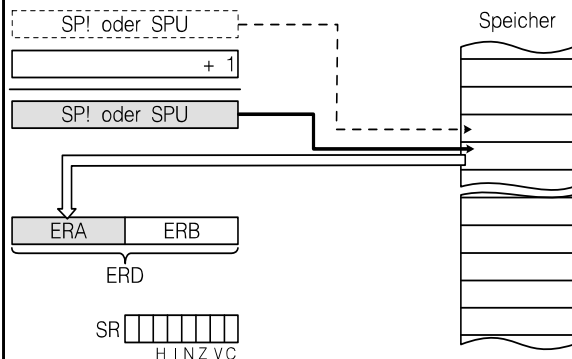
Der Inhalt des Ergebnisregisters ERA wird in der Speicherstelle gespeichert, auf die der System-Stapelzeiger SP! zeigt.



Motorola	PULA	Funktion	CPU Typ A / 6303																							
B&R	AVS	SP! + 1 ⇨ SP!; (SP!) ⇨ ERA																								
Kurz																										
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> PG1000																						
				<input type="radio"/> PG-PC																						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
3/1						E	A	M	F	Z	#	P	C	I	Y	D <td>U<td>I</td><td>B</td><td>G</td><td>H</td><td>I</td><td>N</td><td>Z</td><td>V</td><td>C</td></td>	U <td>I</td> <td>B</td> <td>G</td> <td>H</td> <td>I</td> <td>N</td> <td>Z</td> <td>V</td> <td>C</td>	I	B	G	H	I	N	Z	V	C
32																										

Da der System-Stapelzeiger SP! auf die nächste freie Speicherstelle des Stapels zeigt, muß der SP! zuerst um den Wert 1 erhöht werden.

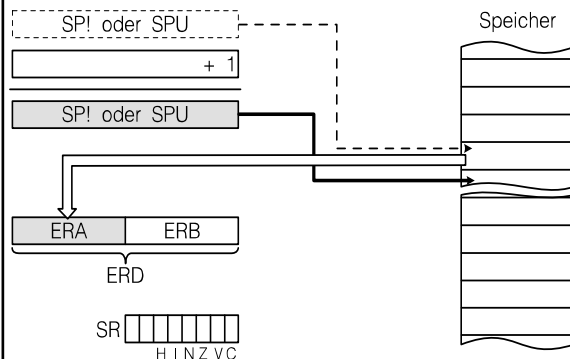
Danach wird das Ergebnisregister ERA mit dem Inhalt der Speicherstelle geladen, auf die SP! zeigt.



Motorola	PULA	Funktion	CPU Typ B / 6809																							
B&R	AVS	(SP!) ⇒ ERA; SP! + 1 ⇒ SP!																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
6/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
35 02																										

Da der System-Stapelzeiger SP! auf die letzte belegte Speicherstelle des Stapels zeigt, wird das Ergebnisregister ERA mit dem Inhalt der Speicherstelle geladen, auf die SP! zeigt.

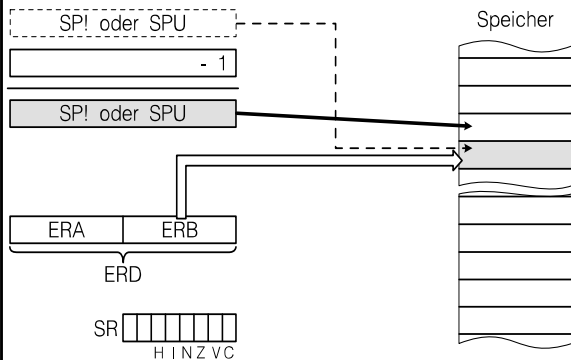
Danach wird SP! um den Wert 1 erhöht, damit SP! wieder auf die letzte belegte Speicherstelle des Stapels zeigt.



Motorola	PSHB	Funktion														CPU Typ A / 6303										
B&R	BNS	ERB ⇒ (SP!); SP! - 1 ⇒ SP!																								
Kurz																										
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> PG1000 <input type="radio"/> PG-PC		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
4/1						E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
37																										

Der Inhalt des Ergebnisregisters ERB wird in der Speicherstelle gespeichert, auf die der System-Stapelzeiger SP! zeigt.

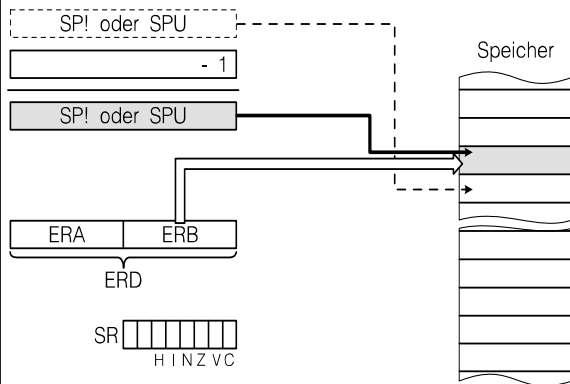
Danach wird SP! um den Wert 1 vermindert, damit SP! wieder auf die nächste freie Stelle des Stapels zeigt.



Motorola	PSHB	Funktion														CPU Typ B / 6809										
B&R	BNS	SP! - 1 ⇒ SP!; ERB ⇒ (SP!)																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80						
																				<input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
6/2						E	A	M	F	Z	#	P	C	I	Y	D	U <td>!</td> <td>B</td> <td>G</td> <td>H</td> <td>I</td> <td>N</td> <td>Z</td> <td>V</td> <td>C</td>	!	B	G	H	I	N	Z	V	C
34 04																						<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Da der System-Stapelzeiger SP! auf die letzte belegte Speicherstelle des Stapels zeigt, muß SP! zuerst um den Wert 1 vermindert werden.

Der Inhalt des Ergebnisregisters ERB wird in der Speicherstelle gespeichert, auf die SP! zeigt.

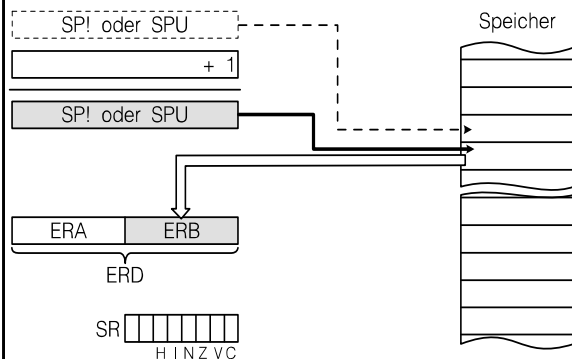




Motorola	PULB	Funktion	CPU Typ A / 6303																							
B&R	BVS	SP! + 1 ⇨ SP!; (SP!) ⇨ ERB																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen						<input type="radio"/> PG1000														
												<input type="radio"/> PG-PC														
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
3/1						E	A	M	F	Z	#	P	C	I	Y	D <td>U<td>I</td><td>B</td><td>G</td><td>H</td><td>I</td><td>N</td><td>Z</td><td>V</td><td>C</td></td>	U <td>I</td> <td>B</td> <td>G</td> <td>H</td> <td>I</td> <td>N</td> <td>Z</td> <td>V</td> <td>C</td>	I	B	G	H	I	N	Z	V	C
33																										

Da der System-Stapelzeiger SP! auf die nächste freie Speicherstelle des Stapels zeigt, muß der SP! zuerst um den Wert 1 erhöht werden.

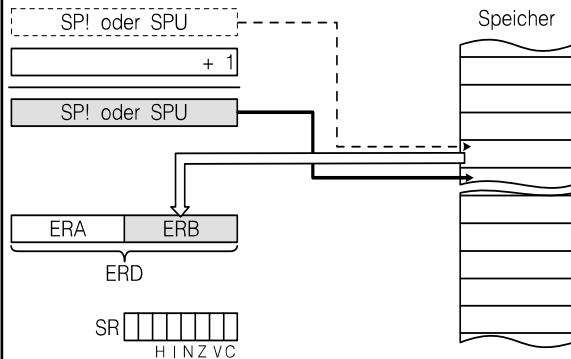
Danach wird das Ergebnisregister ERB mit dem Inhalt der Speicherstelle geladen, auf die SP! zeigt.



Motorola	PULB	Funktion														CPU Typ B / 6809										
B&R	BVS	(SP!) ⇒ ERB; SP! + 1 ⇒ SP!																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80						
																				<input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
6/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
35 04																										

Da der System-Stapelzeiger SP! auf die letzte belegte Speicherstelle des Stapels zeigt, wird das Ergebnisregister ERB mit dem Inhalt der Speicherstelle geladen, auf die SP! zeigt.

Danach wird SP! um den Wert 1 erhöht, damit SP! wieder auf die letzte belegte Speicherstelle des Stapels zeigt.



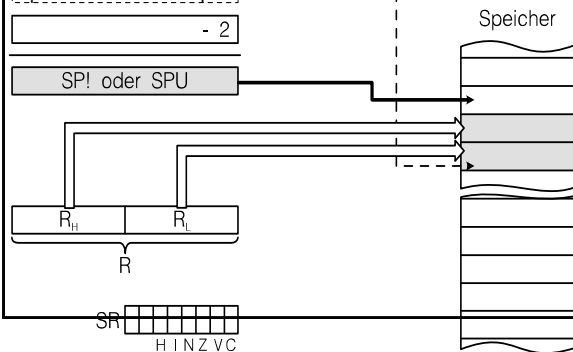
Motorola	PSHX	Funktion	CPU Typ A / 6303																							
B&R	RNS	$R_L \Rightarrow (SP!); SP! - 1 \Rightarrow SP!$																								
Kurz		$R_H \Rightarrow (SP!); SP! - 1 \Rightarrow SP!$																								
Adressierungsarten / Opcode			<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
5/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
3C																										

Der Inhalt des Indexregisters R (X) wird auf dem System-Stapel abgelegt.

Da der System-Stapelzeiger SP! immer auf die nächste freie Speicherstelle des Stapels zeigt, wird der Inhalt von R (X) wie folgt gespeichert:

- 1)  $R_L \Rightarrow (SP!)$
- 2)  $SP! - 1 \Rightarrow SP!$
- 3)  $R_H \Rightarrow (SP!)$
- 4)  $SP! - 1 \Rightarrow SP!$

Nach Ausführung des Befehles zeigt SP! wieder auf die nächste freie Speicherstelle des Stapels.



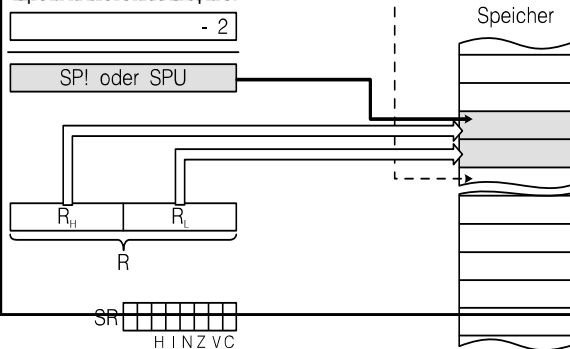
Motorola	PSHX	Funktion	CPU Typ B / 6809																							
B&R	RNS	SP! - 1 ⇒ SP!; R <sub>L</sub> ⇒ (SP!)																								
Kurz		SP! - 1 ⇒ SP!; R <sub>H</sub> ⇒ (SP!)																								
Adressierungsarten / Opcode			<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
7/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
34 10																										

Der Inhalt des Indexregisters R (X) wird auf dem System-Stapel abgelegt.

Da der System-Stapelzeiger SP! auf die letzte belegte Speicherstelle des Stapels zeigt, wird der Inhalt von R (X) wie folgt gespeichert:

- 1)  $SP! - 1 \Rightarrow SP!$
- 2)  $R_L \Rightarrow (SP!)$
- 3)  $SP! - 1 \Rightarrow SP!$
- 4)  $R_H \Rightarrow (SP!)$

SP! zeigt nach der Ausführung des Befehles auf die letzte belegte Speicherstelle des Stapels.

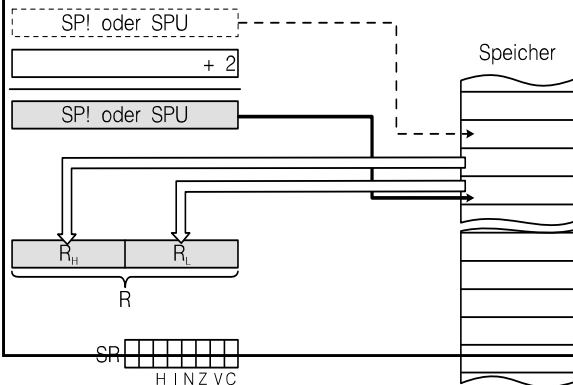


Motorola	PULX	Funktion	CPU Typ A / 6303																							
B&R	RVS	SP! + 1 ⇨ SP!; (SP!) ⇨ R <sub>H</sub>																								
Kurz		SP! + 1 ⇨ SP!; (SP!) ⇨ R <sub>L</sub>																								
Adressierungsarten / Opcode			<input type="checkbox"/> PG1000 <input type="checkbox"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
4/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
38																										

Das Indexregister R (X) wird mit Daten vom Systemstapel geladen.  
Da der System-Stapelzeiger SP! immer auf die nächste freie Speicherstelle des Stapels zeigt, wird R (X) wie folgt vom Stapel geladen:

- 1)  $SP! + 1 \Rightarrow SP!$
- 2)  $R_i \Rightarrow (SP!)$
- 3)  $SP! + 1 \Rightarrow SP!$
- 4)  $R_H \Rightarrow (SP!)$

Nach Ausführung des Befehles zeigt SP! wieder auf die nächste freie Speicherstelle des Stapels.

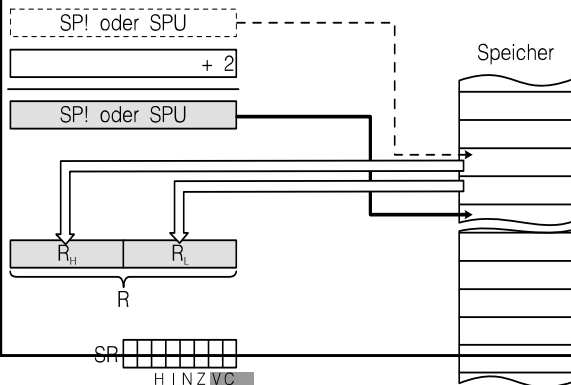


Motorola	PULX	Funktion	CPU Typ B / 6809																							
B&R	RVS	(SP!) ⇒ R <sub>H</sub> ; SP! + 1 ⇒ SP!																								
Kurz		(SP!) ⇒ R <sub>L</sub> ; SP! + 1 ⇒ SP!																								
Adressierungsarten / Opcode			<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
7/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
35 10																										

Das Indexregister R (X) wird mit Daten vom Systemstapel geladen.  
Da der System-Stapelzeiger SP! immer auf die letzte belegte Speicherstelle des Stapels zeigt, wird R (X) wie folgt vom Stapel geladen:

- 1)  $(SP!) \Rightarrow R_L$
- 2)  $SP! + 1 \Rightarrow SP!$
- 3)  $(SP!) \Rightarrow R_H$
- 4)  $SP! + 1 \Rightarrow SP!$

Nach Ausführung des Befehles zeigt SP! auf die letzte belegte Speicherstelle des Stapels.





## 3.5. LOGISCHE VERKNÜPFUNGEN

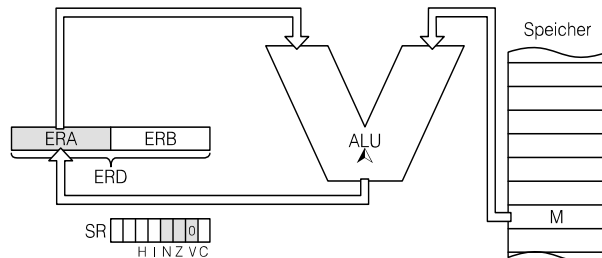
In diesem Abschnitt werden alle Befehle beschrieben, die den Inhalt von Registern und/oder Speicherstellen miteinander verknüpfen und wieder abspeichern.

Motorola	B&R	Betriebsart			
		PG1000	PG-PC	CP 80	PC 80
ANDA	UND	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
ANDB	UB	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
AIM	AIM		<input type="radio"/>		
ORAA	OD	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
ORAB	OB	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
OIM	OIM		<input type="radio"/>		
EORA	EXO	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
EORB	EB	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
EIM	EIM		<input type="radio"/>		

Motorola	ANDA	Funktion														CPU Typ A / 6303											
B&R	UND	ERA $\wedge$ (M) $\Rightarrow$ ERA																									
Kurz	U																										
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> PG1000	<div>Statusregister</div>		
												<input type="radio"/> PG-PC															
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G							
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
	94	B4	84	A4		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Motorola	ANDA	Funktion	CPU Typ B / 6809																							
B&R	UND	ERA $\wedge$ (M) $\Rightarrow$ ERA																								
Kurz	U																									
Adressierungsarten / Opcode		Adreßvorwahlen	<div><input type="radio"/> CP 80</div> <div><input type="radio"/> PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	94	B4	84	A4		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

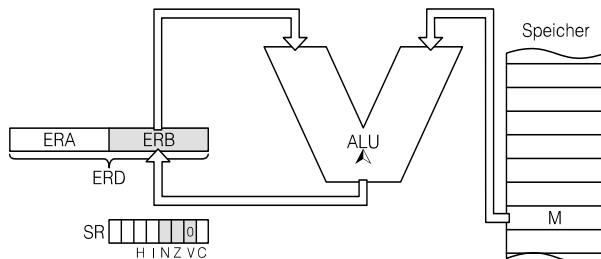
Das Ergebnisregister ERA wird mit dem Inhalt der Speicherstelle M UND-verknüpft. Das Ergebnis wird im Register ERA gespeichert.



Motorola	ANDB	Funktion	CPU Typ A / 6303																							
B&R	UB	ERB ∧ (M) ⇒ ERB																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D4	F4	C4	E4		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	●	●	▼	●

Motorola	ANDB	Funktion	CPU Typ B / 6809																							
B&R	UB	ERB ∧ (M) ⇒ ERB																								
Kurz																										
Adressierungsarten / Opcode			<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D4	F4	C4	E4		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Das Ergebnisregister ERB wird mit dem Inhalt der Speicherstelle M UND-verknüpft. Das Ergebnis wird im Register ERB gespeichert.



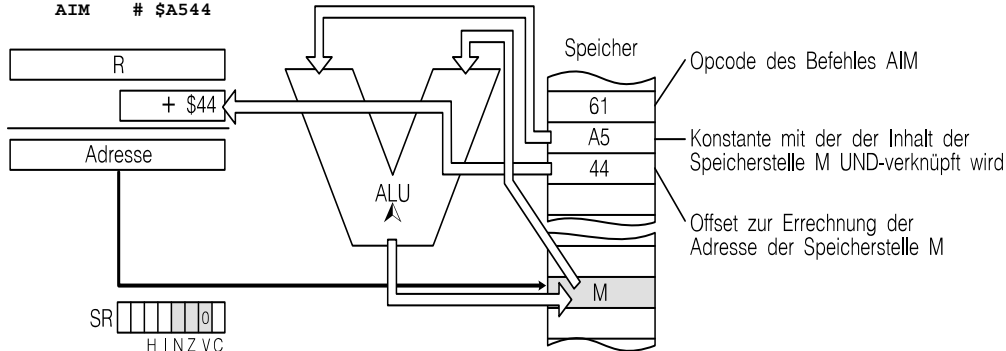
Motorola	AIM	Funktion	CPU Typ A / 6303																							
B&R	AIM	(R + Offset) ^ IMM => (R + Offset)																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div>PG1000</div> <div><input type="radio"/> PG-PC</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
				7/3		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
				61																						

Motorola		Funktion														CPU Typ B / 6809											
B&R																											
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen												CP 80			
																								PC 80			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	

Ein beim Befehl angegebener Immediate Wert \$xx (Konstante) wird mit dem Inhalt der Speicherstelle M UND-verknüpft. Das Ergebnis wird in derselben Speicherstelle gespeichert. Die Adresse M ergibt sich aus der Summe von Indexregister R (X) und dem Offset \$yy, der beim Befehl angegeben werden muß.

**Syntax:** AIM # \$xxyy xx => 1 Byte Konstante  
yy => 1 Byte Offset

**Beispiel:** AIM # \$A544

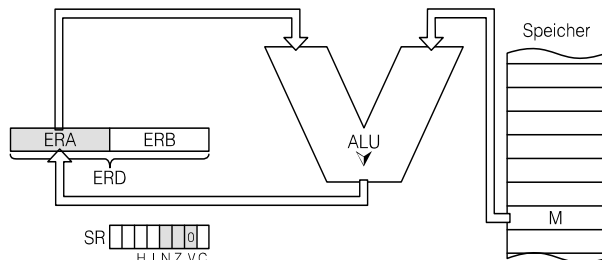




Motorola	ORAA	Funktion	CPU Typ A / 6303																							
B&R	OD	ERA v (M) ⇨ ERA																								
Kurz	O																									
Adressierungsarten / Opcode			<div><input type="radio"/> PG1000</div> <div><input type="radio"/> PG-PC</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	9A	BA	8A	AA		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	ORAA	Funktion														CPU Typ B / 6809														
B&R	OD	ERA v (M) ⇨ ERA																												
Kurz	O																													
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> CP 80 <input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister									
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C				
	9A	BA	8A	AA		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

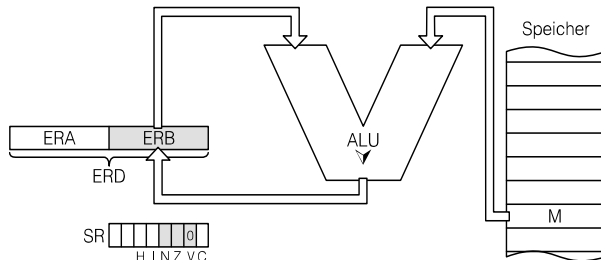
Das Ergebnisregister ERA wird mit dem Inhalt der Speicherstelle M ODER-verknüpft. Das Ergebnis wird im Register ERA gespeichert.



Motorola	ORAB	Funktion	CPU Typ A / 6303																							
B&R	OB	ERB v (M) ⇨ ERB																								
Kurz																										
Adressierungsarten / Opcode			<div><div></div><div></div></div> <div>PG1000 PG-PC</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	DA	FA	CA	EA		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	ORAB	Funktion	CPU Typ B / 6809																							
B&R	OB	ERB v (M) ⇨ ERB																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div><div></div>CP 80</div> <div><div></div>PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	DA	FA	CA	EA		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Das Ergebnisregister ERB wird mit dem Inhalt der Speicherstelle M ODER-verknüpft. Das Ergebnis wird im Register ERB gespeichert.



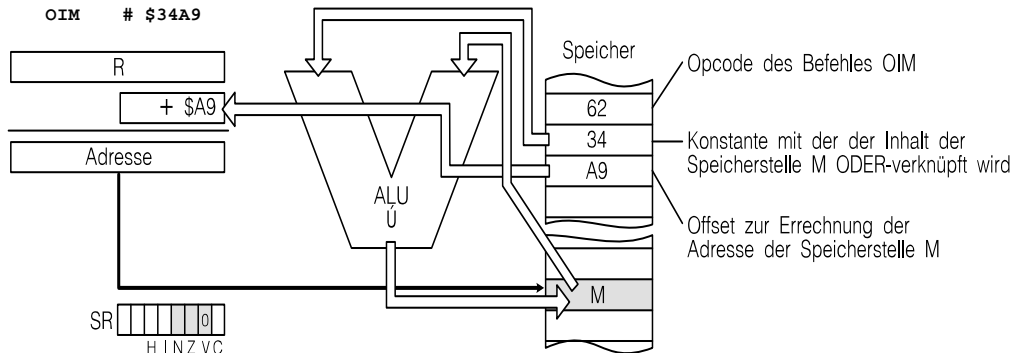
Motorola	OIM	Funktion	CPU Typ A / 6303																							
B&R	OIM	(R + Offset) ∨ IMM ⇒ (R + Offset)																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div>PG1000</div> <div><input type="radio"/> PG-PC</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
				7/3		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
				62																						

Motorola		Funktion														CPU Typ B / 6809											
B&R																											
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen														CP 80	
																										PC 80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	

Ein beim Befehl angegebener Immediate Wert \$xx (Konstante) wird mit dem Inhalt der Speicherstelle M ODER-verknüpft. Das Ergebnis wird in derselben Speicherstelle gespeichert. Die Adresse M ergibt sich aus der Summe von Indexregister R (X) und dem Offset \$yy, der beim Befehl angegeben werden muß.

**Syntax:** OIM # \$xxyy      xx => 1 Byte Konstante  
                                  yy => 1 Byte Offset

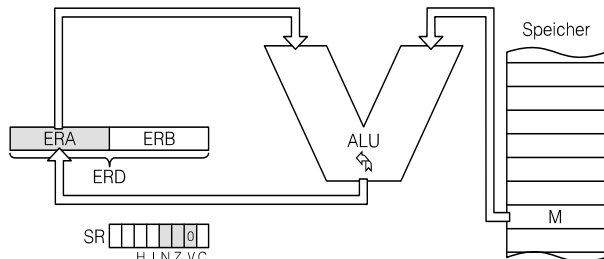
**Beispiel:** OIM # \$34A9



Motorola	EORA	Funktion	CPU Typ A / 6303																							
B&R	EXO	ERA ⊕ (M) ⇒ ERA																								
Kurz	E																									
Adressierungsarten / Opcode		Adreßvorwahlen	<div><input type="radio"/> PG1000</div> <div><input type="radio"/> PG-PC</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	98	B8	88	A8		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	EORA	Funktion	CPU Typ B / 6809																							
B&R	EXO	ERA ⊕ (M) ⇒ ERA																								
Kurz	E																									
Adressierungsarten / Opcode		Adreßvorwahlen	<div><input type="radio"/> CP 80</div> <div><input type="radio"/> PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	98	B8	88	A8		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

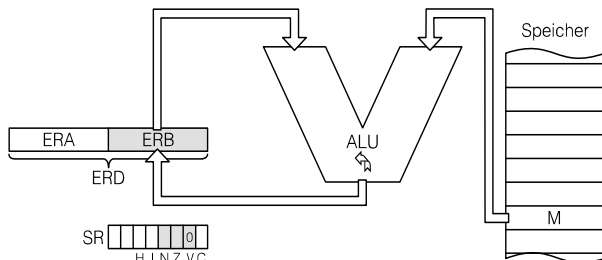
Das Ergebnisregister ERA wird mit dem Inhalt der Speicherstelle M Exklusiv-Oder-verknüpft. Das Ergebnis wird im Register ERA gespeichert.



Motorola	EORB	Funktion	CPU Typ A / 6303																							
B&R	EB	ERB ⊕ (M) ⇨ ERB																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D8	F8	C8	E8		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	EORB	Funktion	CPU Typ B / 6809																							
B&R	EB	ERB ⊕ (M) ⇔ ERB																								
Kurz																										
Adressierungsarten / Opcode			<div><input type="radio"/> CP 80</div> <div><input type="radio"/> PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D8	F8	C8	E8		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Das Ergebnisregister ERB wird mit dem Inhalt der Speicherstelle M Exklusiv-Oder-verknüpft. Das Ergebnis wird im Register ERB gespeichert.



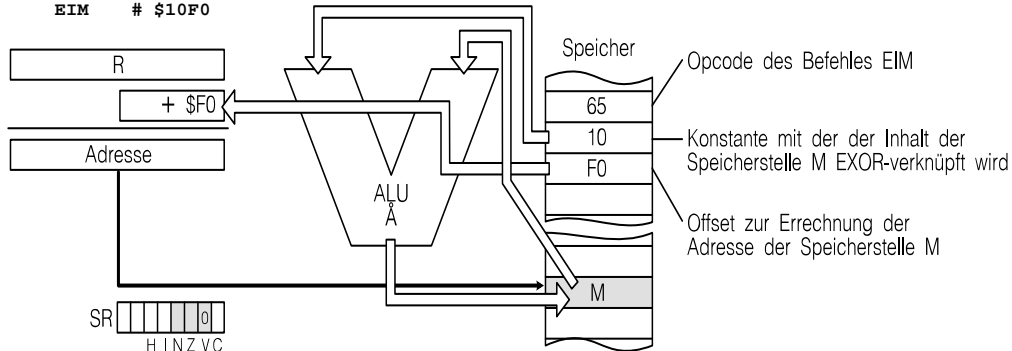
Motorola	EIM	Funktion														CPU Typ A / 6303											
B&R	EIM	(R + Offset) ⊕ IMM ⇒ (R + Offset)																									
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen												PG1000			
																								PG-PC			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
				7/3		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
				65							○											○	○	●	●	▼	○

Motorola		Funktion														CPU Typ B / 6809											
B&R																											
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen												CP 80			
																								PC 80			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	

Ein beim Befehl angegebener Immediate Wert \$xx (Konstante) wird mit dem Inhalt der Speicherstelle M Exklusiv-Oder-verknüpft. Das Ergebnis wird in derselben Speicherstelle gespeichert. Die Adresse M ergibt sich aus der Summe von Indexregister R (X) und dem Offset \$yy, der beim Befehl angegeben werden muß.

**Syntax:** EIM # \$xyy      xx => 1 Byte Konstante  
yy => 1 Byte Offset

**Beispiel:** EIM # \$10F0



## 3.6. ARITHMETISCHE OPERATIONEN

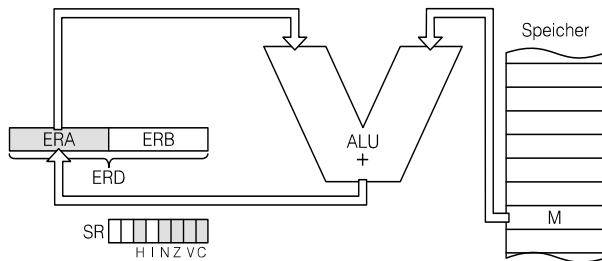
In diesem Abschnitt werden die Befehle beschrieben, die zwei Operanden addieren, subtrahieren oder multiplizieren.

Motorola	B&R	Betriebsart			
		PG1000	PG-PC	CP 80	PC 80
ADDA	+	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
ADCA	ADD	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
ADDB	+B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
ADCB	++B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
ADDD	+D	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
ABA	A+B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
SUBA	-	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
SBCA	SUB	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
SUBB	-B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
SBCB	--B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
SUBD	-D	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
SBA	A-B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
ABX	B+R	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
MUL	A*B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Motorola	ADDA	Funktion														CPU Typ A / 6303										
B&R	+	ERA + (M) ⇨ ERA																								
Kurz																										
Adressierungsarten / Opcode										Adreßvorwahlen										<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
	9B	BB	8B	AB		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	●	●	●	●	●

Motorola	ADDA	Funktion														CPU Typ B / 6809										
B&R	+	ERA + (M) ⇨ ERA																								
Kurz																										
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> CP 80		
																								<input type="radio"/> PC 80		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
	9B	BB	8B	AB		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Die Inhalte von Ergebnisregister ERA und Speicherstelle M werden addiert. Das Ergebnis wird im ERA abgespeichert.

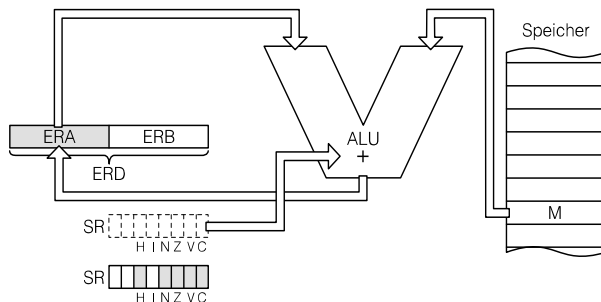




Motorola	ADCA	Funktion	CPU Typ A / 6303																							
B&R	ADD	ERA + (M) + C ⇔ ERA																								
Kurz	A, ++																									
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	99	B9	89	A9		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	ADCA	Funktion	CPU Typ B / 6809																							
B&R	ADD	ERA + (M) + C ⇒ ERA																								
Kurz	A, ++																									
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> CP 80																						
				<input type="radio"/> PC 80																						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	99	B9	89	A9		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

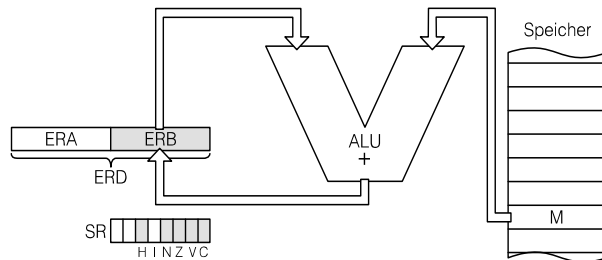
Die Inhalte von Ergebnisregister ERA, Speicherstelle M und das Carry-Flag (möglicher Übertrag einer vorhergehenden Addition) werden addiert. Das Ergebnis wird im ERA abgespeichert.



Motorola	ADDB	Funktion	CPU Typ A / 6303																							
B&R	+B	ERB + (M) ⇨ ERB																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div><input type="radio"/> PG1000</div> <div><input type="radio"/> PG-PC</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
	DB	FB	CB	EB		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	●	●	●	●	●

Motorola	ADDB	Funktion														CPU Typ B / 6809										
B&R	+B	ERB + (M) ⇨ ERB																								
Kurz																										
Adressierungsarten / Opcode										Adreßvorwahlen										<input type="radio"/> CP 80 <input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
	DB	FB	CB	EB		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

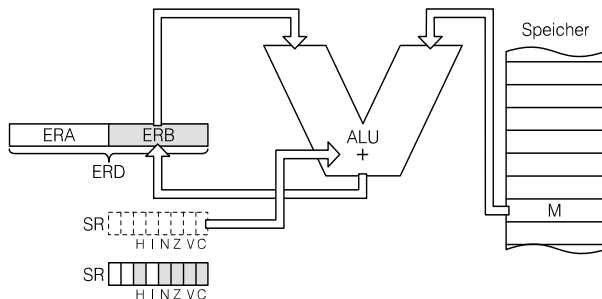
Die Inhalte von Ergebnisregister ERB und Speicherstelle M werden addiert. Das Ergebnis wird im ERB abgespeichert.



Motorola	ADCB	Funktion	CPU Typ A / 6303																							
B&R	++B	ERB + (M) + C ⇨ ERB																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen						<input type="radio"/> PG1000														
												<input type="radio"/> PG-PC														
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D9	F9	C9	E9		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	ADCB	Funktion														CPU Typ B / 6809														
B&R	++B	ERB + (M) + C ⇒ ERB																												
Kurz																														
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> CP 80 <input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister									
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C				
	D9	F9	C9	E9		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

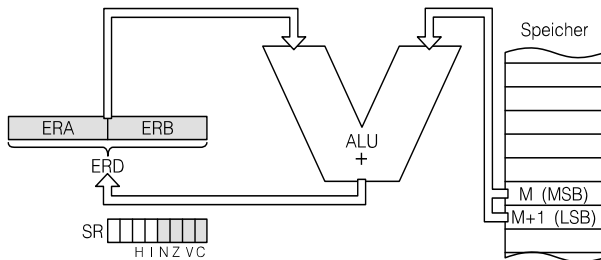
Die Inhalte von Ergebnisregister ERB, Speicherstelle M und das Carry-Flag (möglicher Übertrag einer vorhergehenden Addition) werden addiert. Das Ergebnis wird im ERB abgespeichert.



Motorola	ADDD	Funktion	CPU Typ A / 6303																							
B&R	+D	ERD + (M:M+1) ⇒ ERD																								
Kurz																										
Adressierungsarten / Opcode			<div><input type="radio"/> PG1000</div> <div><input type="radio"/> PG-PC</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
	4/2	5/3	3/3	5/2		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
	D3	F3	C3	E3																						

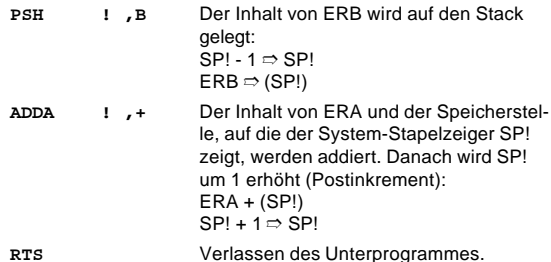
Motorola	ADDD	Funktion	CPU Typ B / 6809																							
B&R	+D	ERD + (M:M+1) ⇨ ERD																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
	6/2	7/3	4/3	6+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
	D3	F3	C3	E3																						

Die Inhalte von Ergebnisregister ERD und den Speicherstellen M und M+1 werden addiert. Das Ergebnis wird im ERD abgespeichert.



Motorola	ABA	Funktion	CPU Typ B / 6809																							
B&R	A+B	ERA + ERB ⇒ ERA																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen										<input type="radio"/> CP 80										
																<input type="radio"/> PC 80										
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		25/3				E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		BD FF 90																			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

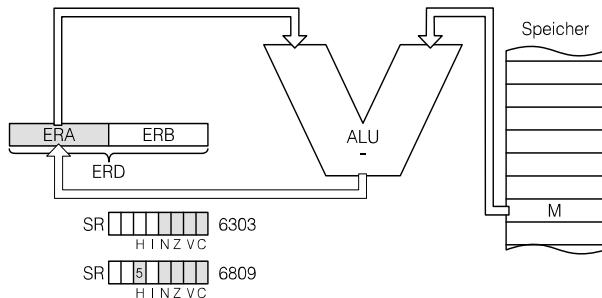
Da der Befehl im 6809 nicht vorhanden ist, wird dieser durch einen Unterprogrammssprung in das Betriebssystem realisiert. Das kleine Unterprogramm, das diesen Befehl simuliert, besteht aus drei Befehlen:



Motorola	SUBA	Funktion	CPU Typ A / 6303																							
B&R	-	ERA - (M) ⇒ ERA																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	90	B0	80	A0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	SUBA	Funktion	CPU Typ B / 6809																							
B&R	-	ERA - (M) ⇒ ERA																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div><input type="radio"/> CP 80</div> <div><input type="radio"/> PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	90	B0	80	A0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0	0	0

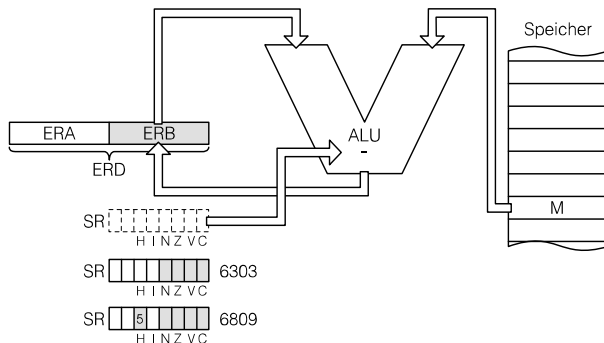
Der Inhalt der Speicherstelle M wird vom Ergebnisregister ERA subtrahiert. Das Ergebnis wird im ERA abgespeichert.



Motorola	SBCA	Funktion	CPU Typ A / 6303																							
B&R	SUB	ERA - (M) - C ⇒ ERA																								
Kurz	--																									
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> PG1000																						
				<input type="radio"/> PG-PC																						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	92	B2	82	A2		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	SBCA	Funktion														CPU Typ B / 6809											
B&R	SUB	ERA - (M) - C ⇒ ERA																									
Kurz	--																										
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> CP 80 <input type="radio"/> PC 80			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
	92	B2	82	A2		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

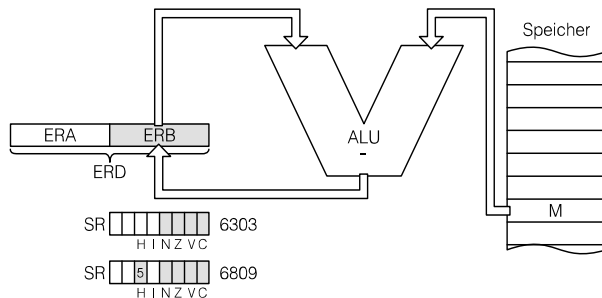
Der Inhalt der Speicherstelle M und das Carry-Flag werden vom Ergebnisregister ERA subtrahiert. Das Ergebnis wird im ERA abgespeichert.



Motorola	SUBB	Funktion	CPU Typ A / 6303																							
B&R	-B	ERB - (M) ⇨ ERB																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div><div><input type="radio"/> PG1000</div><div><input type="radio"/> PG-PC</div></div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D0	F0	C0	E0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	SUBB	Funktion	CPU Typ B / 6809																							
B&R	-B	ERB - (M) ⇨ ERB																								
Kurz																										
Adressierungsarten / Opcode			<div><input type="radio"/> CP 80</div> <div><input type="radio"/> PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D0	F0	C0	E0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0	0	0

Der Inhalt der Speicherstelle M wird vom Ergebnisregister ERB subtrahiert. Das Ergebnis wird im ERB abgespeichert.

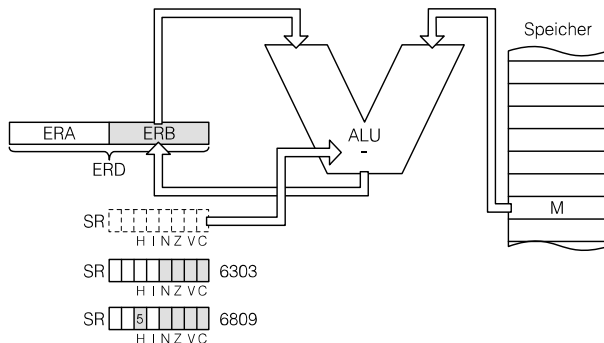




Motorola	SBCB	Funktion	CPU Typ A / 6303																							
B&R	--B	ERB - (M) - C ⇒ ERB																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D2	F2	C2	E2		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	SBCB	Funktion	CPU Typ B / 6809																							
B&R	--B	ERB - (M) - C ⇒ ERB																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div><input type="radio"/> CP 80</div> <div><input type="radio"/> PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D2	F2	C2	E2		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

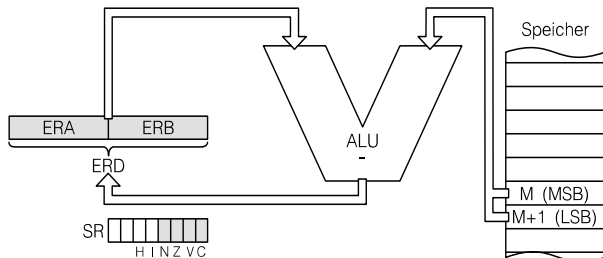
Der Inhalt der Speicherstelle M und das Carry-Flag werden vom Ergebnisregister ERB subtrahiert. Das Ergebnis wird im ERB abgespeichert.



Motorola	SUBD	Funktion														CPU Typ A / 6303										
B&R	-D	ERD - (M:M+1) ⇔ ERD																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000	Statusregister					
																				<input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	H	I	N	Z	V	C
	4/2	5/3	3/3	5/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	93	B3	83	A3																						

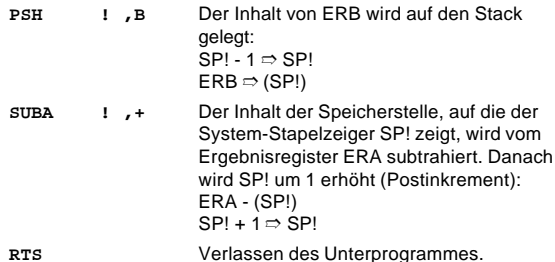
Motorola	SUBD	Funktion	CPU Typ B / 6809																							
B&R	-D	ERD - (M:M+1) ⇔ ERD																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div><div></div>CP 80</div> <div><div></div>PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	6/2	7/3	4/3	6+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	93	B3	83	A3																						

Die Inhalte der Speicherstellen M und M+1 werden vom Ergebnisregister ERD subtrahiert. Das Ergebnis wird im ERD abgespeichert.



Motorola	SBA	Funktion														CPU Typ B / 6809										
B&R	A-B	ERA - ERB ⇒ ERA																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80 <input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		25/3				E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		8D FF 9A																			X	O	•	•	•	•

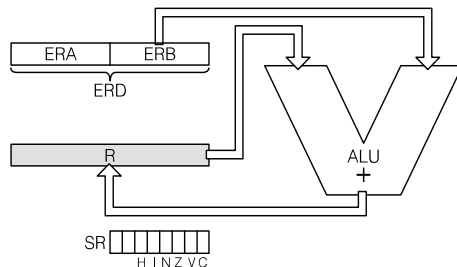
Da der Befehl im 6809 nicht vorhanden ist, wird dieser durch einen Unterprogrammssprung in das Betriebssystem realisiert. Das kleine Unterprogramm, das diesen Befehl simuliert, besteht aus drei Befehlen:



Motorola	ABX	Funktion														CPU Typ A / 6303																		
B&R	B+R	ERB + R ⇨ R																																
Kurz																																		
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/>	PG1000									
																								<input type="radio"/>	PG-PC									
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister													
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C								
3A																																		

Motorola	ABX	Funktion																CPU Typ B / 6809								
B&R	B+R	ERB + R ⇨ R																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80	<input type="radio"/> PC 80					
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
3/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
3A																										

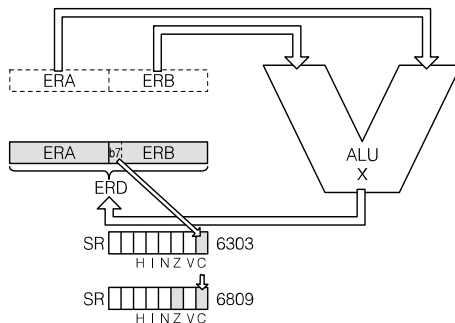
Der Inhalt von Ergebnisregister ERB und Indexregister R (X) wird addiert. Das Ergebnis wird im R (X) abgespeichert.



Motorola	MUL	Funktion														CPU Typ A / 6303										
B&R	A*B	ERA X ERB ⇒ ERD																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/>	PG1000					
																				<input type="radio"/>	PG-PC					
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
7/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
3D																										

Motorola	MUL	Funktion	CPU Typ B / 6809																							
B&R	A*B	ERA X ERB ➔ ERD																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
11/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
3D																										

Die Inhalte von Ergebnisregister ERA und ERB werden multipliziert. Das Ergebnis wird im ERD abgespeichert.





## 3.7. VERGLEICHS- UND TESTBEFEHLE

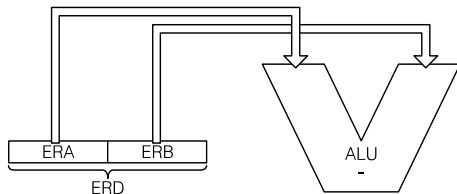
In diesem Abschnitt werden die Befehle beschrieben, die Inhalte von Registern oder Speicherstellen miteinander vergleichen (Subtraktion) oder auf ein bestimmtes Bitmuster testen (UND-Verknüpfung). Die Operationen werden durchgeführt, ohne das Ergebnis zu speichern. Es wird lediglich das Statusregister entsprechend dem Ergebnis der Operation verändert. Aufgrund der Veränderung des Statusregisters können bedingte Entscheidungen (bedingte Sprünge) getroffen werden.

Motorola	B&R	Betriebsart			
		PG1000	PG-PC	CP 80	PC 80
CBA	AVB	○	○	○	○
CMPA	CMP	○	○	○	○
CMPB	VB	○	○	○	○
CPX	VR	○	○	○	○
CPX#	VRK	○	○	○	○
CPY	VY				○
CPY#	VYK				○
BITA	B	○	○	○	○
BITB	BB	○	○	○	○
TIM	TIM		○		

Motorola	CBA	Funktion														CPU Typ A / 6303										
B&R	AVB	ERA - ERB																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
11																					<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input checked="" type="radio"/>	<input checked="" type="radio"/>	<input checked="" type="radio"/>

Motorola	CBA	Funktion														CPU Typ B / 6809											
B&R	AVB	ERA - ERB																									
Kurz																											
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80 <input type="radio"/> PC 80							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
		25/3				E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
		BD FF 95																				X	0	●	●	●	●

Die Inhalte der beiden Ergebnisregister ERA und ERB werden miteinander verglichen, indem eine Subtraktion (ERA - ERB) durchgeführt wird. Das Ergebnis beeinflusst den Zustand des Statusregisters, wird aber nicht abgespeichert.



SR 

--	--	--	--	--	--	--	--

 6303  
H I N Z V C

SR 

	5						
--	---	--	--	--	--	--	--

 6809  
H I N Z V C

### 6809:

Da der Befehl im 6809 nicht vorhanden ist, wird dieser durch einen Unterprogrammssprung in das Betriebssystem realisiert. Das kleine Unterprogramm, das diesen Befehl simuliert besteht aus drei Befehlen:

**PSH**

	!	,B
--	---	----

Der Inhalt von ERB wird auf den Stack gelegt:  
SP! - 1 ⇒ SP!  
ERB ⇒ (SP!)

**CMP**

	!	,+
--	---	----

Der Inhalt von ERA wird mit dem Inhalt der Speicherstelle verglichen, auf die der System-Stapelzeiger SP! zeigt. Danach wird SP! um 1 erhöht (Postinkrement):  
ERA - (SP!)  
SP! + 1 ⇒ SP!

**RTS**

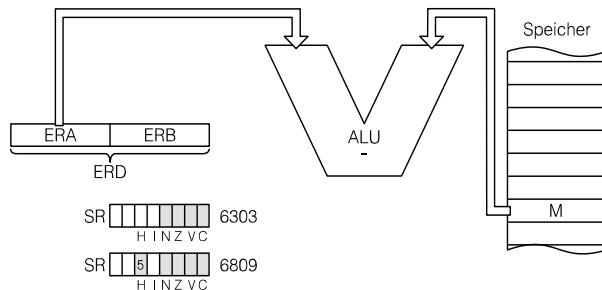
Verlassen des Unterprogrammes.



Motorola	CMPA	Funktion														CPU Typ A / 6303										
B&R	CMP	ERA - (M)																								
Kurz	V																									
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	91	B1	81	A1		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	CMPA	Funktion														CPU Typ B / 6809											
B&R	CMP	ERA - (M)																									
Kurz	V																										
Adressierungsarten / Opcode													Adreßvorwahlen													<input type="radio"/> CP 80 <input type="radio"/> PC 80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
	91	B1	81	A1		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

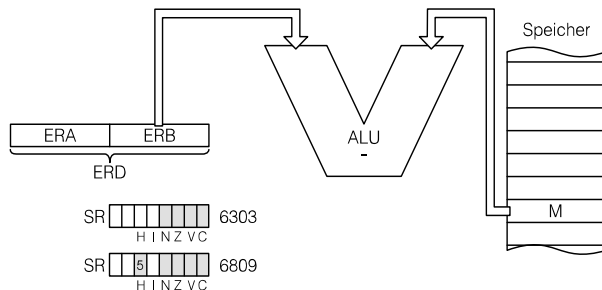
Der Inhalt des Ergebnisregisters ERA wird mit dem Inhalt der Speicherstelle M verglichen, indem eine Subtraktion (ERA - (M)) durchgeführt wird. Das Ergebnis beeinflusst das Statusregister, wird aber nicht abgespeichert.



Motorola	CMPB	Funktion														CPU Typ A / 6303										
B&R	VB	ERB - (M)																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D1	F1	C1	E1		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	CMPB	Funktion														CPU Typ B / 6809											
B&R	VB	ERB - (M)																									
Kurz																											
Adressierungsarten / Opcode													Adreßvorwahlen													<input type="radio"/> CP 80 <input type="radio"/> PC 80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
	D1	F1	C1	E1		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

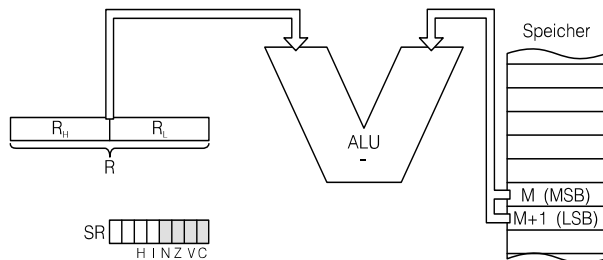
Der Inhalt des Ergebnisregisters ERB wird mit dem Inhalt der Speicherstelle M verglichen, indem eine Subtraktion (ERB - (M)) durchgeführt wird. Das Ergebnis beeinflusst das Statusregister, wird aber nicht abgespeichert.



Motorola	CPX	Funktion														CPU Typ A / 6303										
B&R	VR	R - (M:M+1)																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	3/3	5/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	9C	BC	8C	AC																						

Motorola	CPX	Funktion														CPU Typ B / 6809											
B&R	VR	R - (M:M+1)																									
Kurz																											
Adressierungsarten / Opcode													Adreßvorwahlen													<input type="radio"/> CP 80 <input type="radio"/> PC 80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
	6/2	7/3	4/3	6+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
	9C	BC	8C	AC																							

Der Inhalt des Indexregisters R (X) wird mit dem Inhalt der Speicherstellen M und M+1 verglichen, indem eine Subtraktion ( $R - (M:M+1)$ ) durchgeführt wird. Das Ergebnis beeinflusst das Statusregister, wird aber nicht abgespeichert.

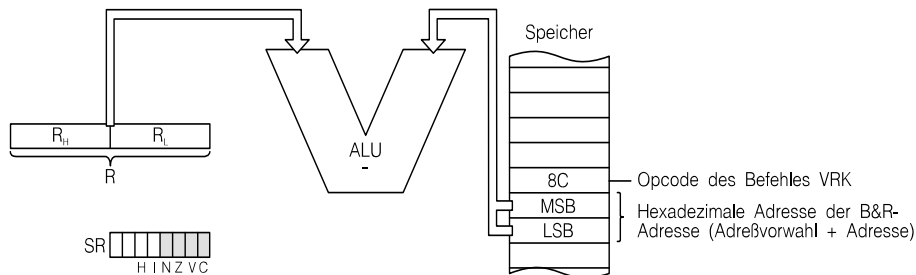


Motorola	CPX#	Funktion														CPU Typ A / 6303										
B&R	VRK	R - M																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/>	PG1000					
																				<input type="radio"/>	PG-PC					
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
			3/3			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
			8C			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	CPX#	Funktion														CPU Typ B / 6809										
B&R	VRK	R - M																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80 <input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
			4/3			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
			8C			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Der Inhalt des Indexregisters R (X) wird mit der angegebenen Adresse M verglichen, indem eine Subtraktion ( $R - M$ ) durchgeführt wird. Das Ergebnis beeinflusst das Statusregister, wird aber nicht abgespeichert.

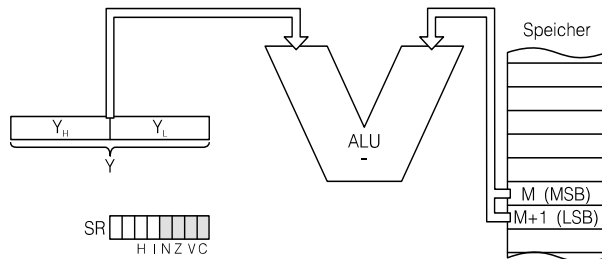
Der Befehl entspricht dem VR # xxxx Befehl. Die Adresse, die der Anwender in die AWL-Eingabezeile eingibt, ersetzt das PROgrammierSYstem durch die effektive Adresse (hexadezimaler Wert), die die B&R Adresse (= Adreßvorwahl + Adreßteil) im Speicher der SPS hat.



Motorola		Funktion																CPU Typ A / 6303								
B&R																										
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen																PG1000				
																						PG-PC				
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C

Motorola	CPY	Funktion														CPU Typ B / 6809										
B&R	VY	Y - (M:M+1)																								
Kurz																										
Adressierungsarten / Opcode										Adreßvorwahlen												CP 80				
																				○		PC 80				
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	7/3	8/4	5/4	7+/3+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	10 9C	10 BC	10 8C	10 AC												●	●	●	●	●	●	○	●	●	●	●

Der Inhalt des Indexregisters Y wird mit dem Inhalt der Speicherstellen M und M+1 verglichen, indem eine Subtraktion ( $Y - (M:M+1)$ ) durchgeführt wird. Das Ergebnis beeinflusst das Statusregister, wird aber nicht abgespeichert.

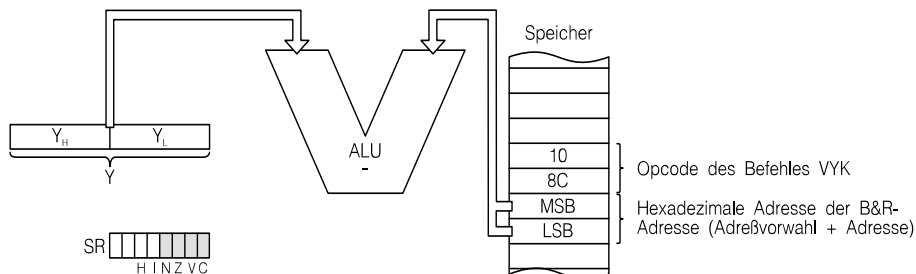


Motorola			Funktion														CPU Typ A / 6303									
B&R																										
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														PG1000						
																				PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
						E	A	M	F	Z	#	P	C	I	Y	D <td>U</td> <td>I</td> <td>B</td> <td>G</td> <td>H</td> <td>I</td> <td>N</td> <td>Z</td> <td>V</td> <td>C</td>	U	I	B	G	H	I	N	Z	V	C

Motorola	CPY#	Funktion															CPU Typ B / 6809													
B&R	VYK	Y - M																												
Kurz																														
Adressierungsarten / Opcode						Adreßvorwahlen															<div>CP 80</div> <div><input type="radio"/> PC 80</div>									
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister									
			5/4			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C				
			10 8C			•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•				

Der Inhalt des Indexregisters Y wird mit der angegebenen Adresse M verglichen, indem eine Subtraktion ( $Y - M$ ) durchgeführt wird. Das Ergebnis beeinflusst das Statusregister, wird aber nicht abgespeichert.

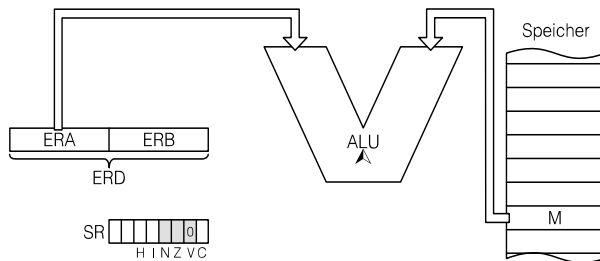
Der Befehl entspricht dem VY # xxxx Befehl. Die Adresse, die der Anwender in die AWL-Eingabezeile eingibt, ersetzt das PROgrammierSYStem durch die effektive Adresse (hexadezimaler Wert), die die B&R Adresse (= Adreßvorwahl + Adreßteil) im Speicher der SPS hat.



Motorola	BITA	Funktion	CPU Typ A / 6303																							
B&R	B	ERA ^ (M)																								
Kurz																										
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> PG1000																						
				<input type="radio"/> PG-PC																						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
	95	B5	85	A5		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	BITA	Funktion														CPU Typ B / 6809											
B&R	B	ERA $\wedge$ (M)																									
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> CP 80 <input type="radio"/> PC 80			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister						
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C	
	95	B5	85	A5		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

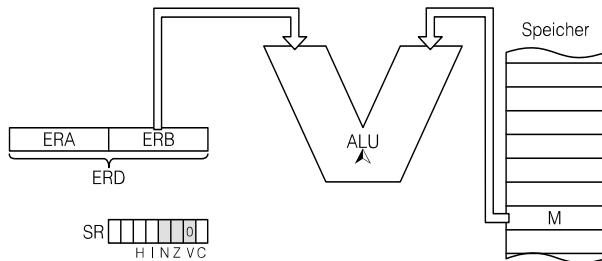
Der Inhalt des Ergebnisregisters ERA wird mit dem Inhalt der Speicherstelle M UND-verknüpft. Das Ergebnis beeinflusst das Statusregister, wird aber nicht abgespeichert.



Motorola	BITB	Funktion	CPU Typ A / 6303																							
B&R	BB	ERB $\wedge$ (M)																								
Kurz																										
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> PG1000																						
				<input type="radio"/> PG-PC																						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	3/2	4/3	2/2	4/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D5	F5	C5	E5		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Motorola	BITB	Funktion	CPU Typ B / 6809																							
B&R	BB	ERB $\wedge$ (M)																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div><div></div>CP 80</div> <div><div></div>PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	4/2	5/3	2/2	4+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	D5	F5	C5	E5		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Der Inhalt des Ergebnisregisters ERB wird mit dem Inhalt der Speicherstelle M UND-verknüpft. Das Ergebnis beeinflusst das Statusregister, wird aber nicht abgespeichert.





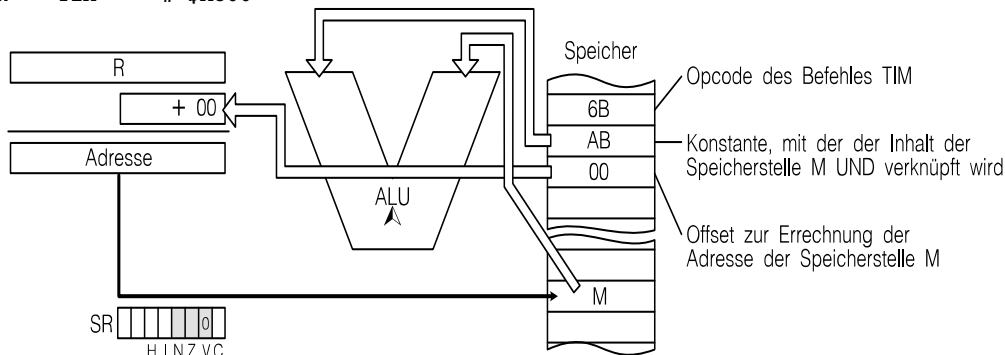
Motorola	TIM	Funktion														CPU Typ A / 6303												
B&R	TIM	(R + Offset) ^ IMM																										
Kurz																												
Adressierungsarten / Opcode												Adreßvorwahlen														PG1000		
																								<input type="radio"/>		PG-PC		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister							
			5/3			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C		
			6B																									

Motorola		Funktion														CPU Typ B / 6809											
B&R																											
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen												CP 80			
																								PC 80			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	

Ein beim Befehl angegebener Immediate Wert \$xx (Konstante) wird mit dem Inhalt der Speicherstelle M UND-Verknüpft. Das Ergebnis beeinflusst das Statusregister, wird aber nicht gespeichert. Die Adresse M ergibt sich aus der Summe von Indexregister R (X) und dem Offset \$yy, der beim Befehl angegeben werden muß.

**Syntax:**      **TIM**      # \$xxyy      xx => 1 Byte Konstante  
    yy => 1 Byte Offset zum Indexregister R (X)

**Beispiel:**      **TIM**      # \$A800





## 3.8. INKREMENT- UND DEKREMENTBEFEHLE

In diesem Abschnitt werden die Befehle beschrieben, die den Inhalt von Registern oder Speicherstellen um den Wert 1 erhöhen oder vermindern.

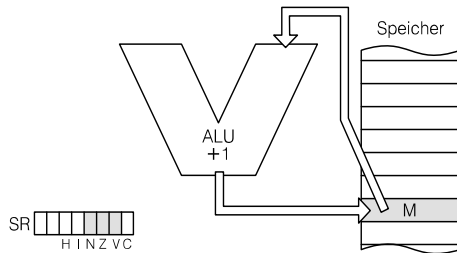
Motorola	B&R	Betriebsart			
		PG1000	PG-PC	CP 80	PC 80
INC	INC	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
INCA	IA	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
INCB	IB	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
INX	IR	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
INS	IS	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
DEC	DEC	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
DECA	DA	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
DECB	DB	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
DEX	DR	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
DES	DS	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Motorola	INC	Funktion														CPU Typ A / 6303										
B&R	INC	(M) + 1 ⇨ (M)																								
Kurz																										
Adressierungsarten / Opcode										Adreßvorwahlen										<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		6/3		6/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		7C		6C		0	0	0				0	0	0							0	0	•	•	•	0

Motorola	INC	Funktion														CPU Typ B / 6809											
B&R	INC	(M) + 1 ⇨ (M)																									
Kurz																											
Adressierungsarten / Opcode													Adreßvorwahlen													<input type="radio"/> CP 80 <input type="radio"/> PC 80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
	6/2	7/3		6+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
	0C	7C		6C		0	0	0				0	0	0	•	•	•	•	•	•	0	0	•	•	•	0	

Der Inhalt der angegebenen Speicherstelle M wird um den Wert 1 erhöht.

Das V-Flag wird auf log. 1 gesetzt, wenn der Inhalt der Speicherstelle von \$7F auf \$80 erhöht wird, andernfalls wird V immer auf log. 0 gesetzt.

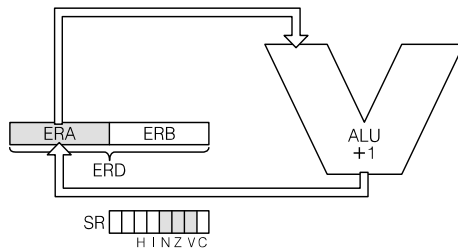


Motorola	INCA	Funktion														CPU Typ A / 6303										
B&R	IA	ERA + 1 ⇒ ERA																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/>	PG1000					
																				<input type="radio"/>	PG-PC					
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
4C																										

Motorola	INCA	Funktion														CPU Typ B / 6809										
B&R	IA	ERA + 1 ➡ ERA																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80						
																				<input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
4C																						<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Der Inhalt des Ergebnisregisters ERA wird um den Wert 1 erhöht.

Das V-Flag wird auf log. 1 gesetzt, wenn ERA von \$7F auf \$80 erhöht wird, andernfalls wird V immer auf log. 0 gesetzt.

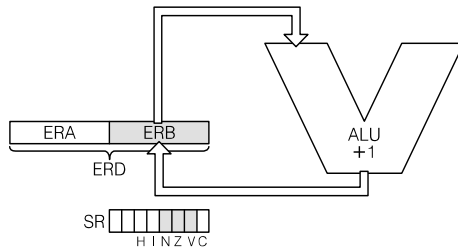


Motorola	INCB	Funktion														CPU Typ A / 6303										
B&R	IB	ERB + 1 ⇨ ERB																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
5C																										

Motorola	INCB	Funktion	CPU Typ B / 6809																							
B&R	IB	ERB + 1 ⇒ ERB																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
5C																										

Der Inhalt des Ergebnisregisters ERB wird um den Wert 1 erhöht.

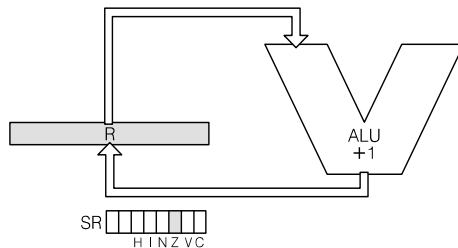
Das V-Flag wird auf log. 1 gesetzt, wenn ERB von \$7F auf \$80 erhöht wird, andernfalls wird V immer auf log. 0 gesetzt.



Motorola	INX	Funktion	CPU Typ A / 6303																							
B&R	IR	R + 1 ⇨ R																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
08																										

Motorola	INX	Funktion														CPU Typ B / 6809										
B&R	IR	R + 1 ⇨ R																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80						
																				<input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
				5/3																	H	I	N	Z	V	C
				30 88 01																		0	1	0	•	0

Der Inhalt des Indexregisters R (X) wird um den Wert 1 erhöht.



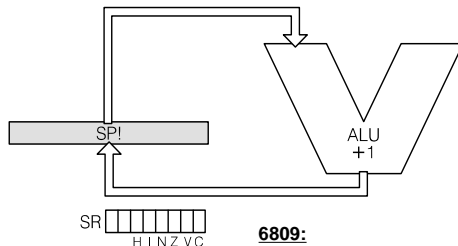
#### 6809:

Der Befehl "IR" existiert nicht im Grundbefehlssatz des 6809. Er wird durch den Befehl "LER I 001" simuliert.

Motorola	INS	Funktion														CPU Typ A / 6303														
B&R	IS	S + 1 ⇒ S																												
Kurz																														
Adressierungsarten / Opcode													Adreßvorwahlen													<input type="radio"/> PG1000 <input type="radio"/> PG-PC				
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister									
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C				
31																														

Motorola	INS	Funktion														CPU Typ B / 6809										
B&R	IS	S + 1 ⇒ S																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80 <input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
				5/3		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
				32 E8 01																						

Der Inhalt des System-Stapelzeigers SP! wird um den Wert 1 erhöht.



#### 6809:

Der Befehl "IS" existiert nicht im Grundbefehlssatz des 6809. Er wird durch den Befehl "LE! ! 001" simuliert.

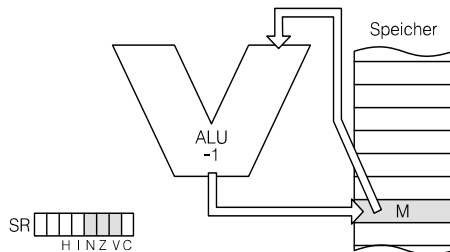


Motorola	DEC	Funktion	CPU Typ A / 6303																							
B&R	DEC	(M) - 1 ⇨ (M)																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
		6/3		6/2		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
		7A		6A		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				<input type="radio"/>	<input type="radio"/>	<input type="radio"/>							<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input checked="" type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>

Motorola	DEC	Funktion	CPU Typ B / 6809																							
B&R	DEC	(M) - 1 → (M)																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
	6/2	7/3		6+2+		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
	0A	7A		6A		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				<input type="radio"/>	<input type="radio"/>	<input type="radio"/>							<input checked="" type="radio"/>	<input checked="" type="radio"/>	<input checked="" type="radio"/>	<input checked="" type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>

Der Inhalt der angegebenen Speicherstelle M wird um den Wert 1 vermindert.

Das V-Flag wird auf log. 1 gesetzt, wenn der Inhalt der Speicherstelle von \$80 auf \$7F vermindert wird, andernfalls wird V immer auf log. 0 gesetzt.

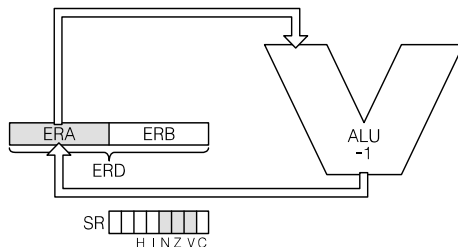


Motorola	DECA	Funktion	CPU Typ A / 6303																							
B&R	DA	ERA - 1 ⇒ ERA																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
4A																										

Motorola	DECA	Funktion	CPU Typ B / 6809																							
B&R	DA	ERA - 1 ⇒ ERA																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
4A																										

Der Inhalt des Ergebnisregisters ERA wird um den Wert 1 vermindert.

Das V-Flag wird auf log. 1 gesetzt, wenn ERA von \$80 auf \$7F vermindert wird, andernfalls wird V immer auf log. 0 gesetzt.

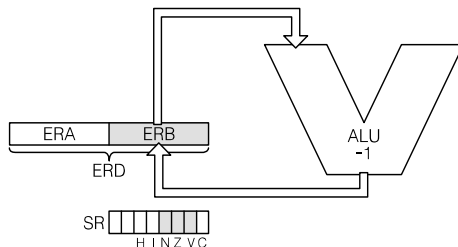


Motorola	DECB	Funktion	CPU Typ A / 6303																							
B&R	DB	ERB - 1 ⇒ ERB																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
5A																										

Motorola	DECB	Funktion	CPU Typ B / 6809																							
B&R	DB	ERB - 1 ⇒ ERB																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen																<input type="radio"/> CP 80				
																						<input type="radio"/> PC 80				
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
5A																					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Der Inhalt des Ergebnisregisters ERB wird um den Wert 1 vermindert.

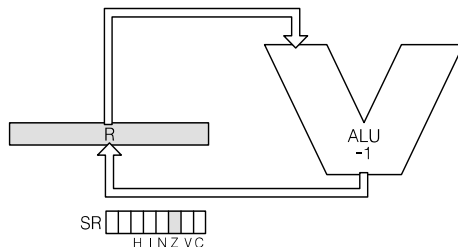
Das V-Flag wird auf log. 1 gesetzt, wenn ERB von \$80 auf \$7F vermindert wird, andernfalls wird V immer auf log. 0 gesetzt.



Motorola	DEX	Funktion														CPU Typ A / 6303											
B&R	DR	R - 1 → R																									
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> PG1000 <input type="radio"/> PG-PC			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister						
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C	
09																											

Motorola	DEX	Funktion														CPU Typ B / 6809										
B&R	DR	R - 1 ➡ R																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80 <input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
				5/3																	H	I	N	Z	V	C
				30 88 FF																		⌋	⌋	●	⌋	⌋

Der Inhalt des Indexregisters R (X) wird um den Wert 1 vermindert.



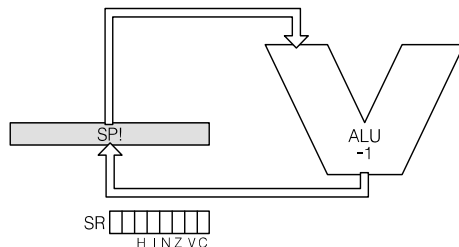
#### 6809:

Der Befehl "DR" existiert nicht im Grundbefehlssatz des 6809. Er wird durch den Befehl "LER I -001" simuliert.

Motorola	DES	Funktion														CPU Typ A / 6303										
B&R	DS	S - 1 → S																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U <td>!</td> <td>B</td> <td>G</td> <td>H</td> <td>I</td> <td>N</td> <td>Z</td> <td>V</td> <td>C</td>	!	B	G	H	I	N	Z	V	C
34																										

Motorola	DES	Funktion														CPU Typ B / 6809										
B&R	DS	S - 1 ➡ S																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80 <input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
				5/3		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
				32 E8 FF																						

Der Inhalt des System-Stapelzeigers SP! wird um den Wert 1 vermindert.



#### 6809:

Der Befehl "DS" existiert nicht im Grundbefehlssatz des 6809. Er wird durch den Befehl "LE! ! -001" simuliert.



### 3.9. SCHIEBE- UND ROTIERBEFEHLE

In diesem Abschnitt werden die Befehle beschrieben, die den Inhalt von Registern oder Speicherstellen um 1 Bit nach links/rechts schieben bzw. rotieren.

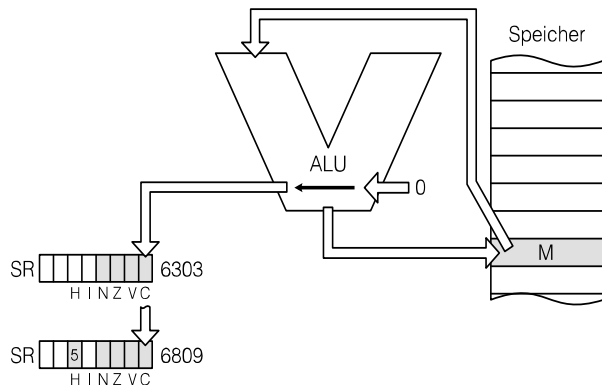
Motorola	B&R	Betriebsart			
		PG1000	PG-PC	CP 80	PC 80
ASL	SL	○	○	○	○
ASLA	SLA	○	○	○	○
ASLB	SLB	○	○	○	○
ASLD	SLD	○	○	○	○
LSR	SR	○	○	○	○
LSRA	SRA	○	○	○	○
LSRB	SRB	○	○	○	○
LSRD	SRD	○	○	○	○
ROL	SLI	○	○	○	○
ROLA	RLA	○	○	○	○
ROLB	RLB	○	○	○	○
ROR	SRE	○	○	○	○
RORA	RRA	○	○	○	○
RORB	RRB	○	○	○	○

Motorola	ASL	Funktion	CPU Typ A / 6303																							
B&R	SL	Schiebe Inhalt von M um 1 Bit nach links																								
Kurz																										
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		6/3		6/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		78		68		0	0	0				0	0	0							0	0	0	0	0	0


Motorola	ASL	Funktion	CPU Typ B / 6809																							
B&R	SL	Schiebe Inhalt von M um 1 Bit nach links																								
Kurz																										
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	6/2	7/3		6+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	08	78		68								0	0	0							0	0	0	0	0	0

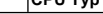
Der Inhalt der Speicherstelle M wird um 1 Bit nach links geschoben. Bit 0 wird durch log. 0 ersetzt und Bit 7 wird in das Carry-Flag geschoben.

Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).



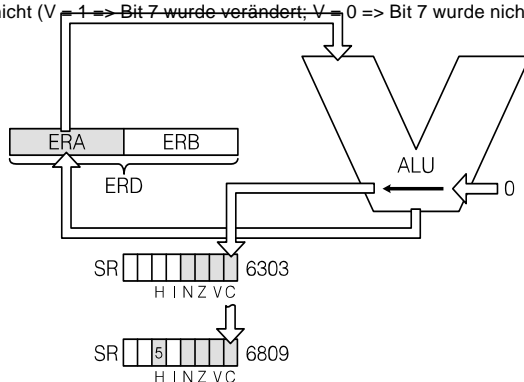


Motorola	ASLA	Funktion	CPU Typ A / 6303																		
B&R	SLA	Schiebe Inhalt von ERA um 1 Bit nach links																			
Kurz																					
Adressierungsarten / Opcode			Adreßvorwahlen																		
			<input type="radio"/> PG1000 <input type="radio"/> PG-PC																		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H I N Z V C
48																					• • • • •

Motorola	ASLA	Funktion	CPU Typ B / 6809																		
B&R	SLA	Schiebe Inhalt von ERA um 1 Bit nach links																			
Kurz																					
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H I N Z V C
48																					x o • • • •

Der Inhalt des Ergebnisregisters ERA wird um 1 Bit nach links geschoben. Bit 0 wird durch log. 0 ersetzt und Bit 7 wird in das Carry-Flag geschoben.

Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).

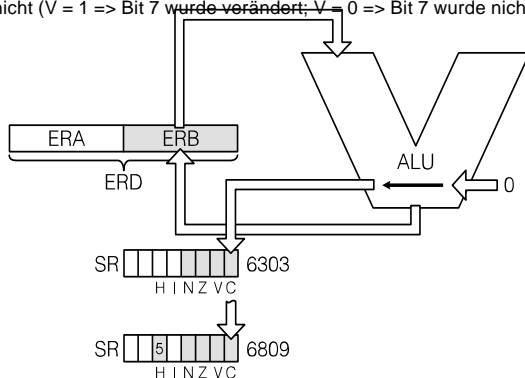


Motorola	ASLB	Funktion	CPU Typ A / 6303																		
B&R	SLB	Schiebe Inhalt von ERB um 1 Bit nach links																			
Kurz																					
Adressierungsarten / Opcode			<div><input type="radio"/> PG1000</div> <div><input type="radio"/> PG-PC</div>																		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H I N Z V C
58																					• • • • •

Motorola	ASLB	Funktion	CPU Typ B / 6809																		
B&R	SLB	Schiebe Inhalt von ERB um 1 Bit nach links																			
Kurz																					
Adressierungsarten / Opcode			Adreßvorwahlen																		
			<input type="radio"/> CP 80 <input type="radio"/> PC 80																		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H I N Z V C
58																					x o . n z v c

Der Inhalt des Ergebnisregisters ERB wird um 1 Bit nach links geschoben. Bit 0 wird durch log. 0 ersetzt und Bit 7 wird in das Carry-Flag geschoben.

Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).

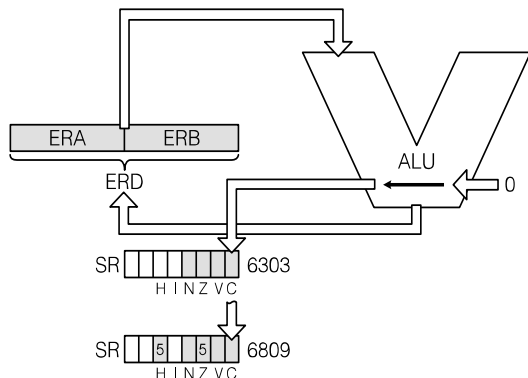


Motorola	ASLD	Funktion	CPU Typ A / 6303																							
B&R	SLD	Schiebe Inhalt von ERD um 1 Bit nach links																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
05																										

Motorola	ASLD	Funktion																	CPU Typ B / 6809								
B&R	SLD	Schiebe Inhalt von ERD																									
Kurz		um 1 Bit nach links																									
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80 <input type="radio"/> PC 80							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
4/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
58 49																						X	○	●	X	●	●

Der Inhalt des Ergebnisregisters ERD wird um 1 Bit nach links geschoben. Bit 0 wird durch log. 0 ersetzt und Bit 7 wird in das Carry-Flag geschoben.

Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).



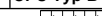
#### 6809:

Da der Befehl im 6809 nicht vorhanden ist, wird er durch zwei andere Befehle realisiert:

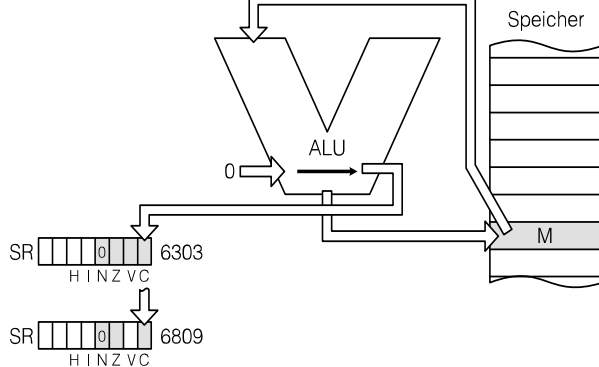
- SLB** Der Inhalt von ERB wird um 1 Bit nach links geschoben. Bit 0 wird durch log. 0 ersetzt. Bit 7 wird in das Carry-Flag geschoben.
- RLA** Der Inhalt von ERA wird um 1 Bit nach links geschoben. Das Carry-Flag ( $\Rightarrow$  Bit 7 von ERB) wird ins Bit 0 geschoben. Bit 7 wird in das Carry-Flag geschoben.

Da der SLD Befehl aus zwei Befehlen realisiert wurde, ist der Zustand des Zero-Flags undefiniert.

Motorola	LSR	Funktion	CPU Typ A / 6303																							
B&R	SR	Schiebe Inhalt von M um 1 Bit nach rechts																								
Kurz																										
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		6/3		6/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		74		64		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0


Motorola	LSR	Funktion	CPU Typ B / 6809																							
B&R	SR	Schiebe Inhalt von M um 1 Bit nach rechts																								
Kurz																										
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	6/2	7/3		6+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	04	74		64		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	


Der Inhalt der Speicherstelle M wird um 1 Bit nach rechts geschoben. Bit 0 wird in das Carry-Flag geschoben. Bit 7 wird durch log. 0 ersetzt.



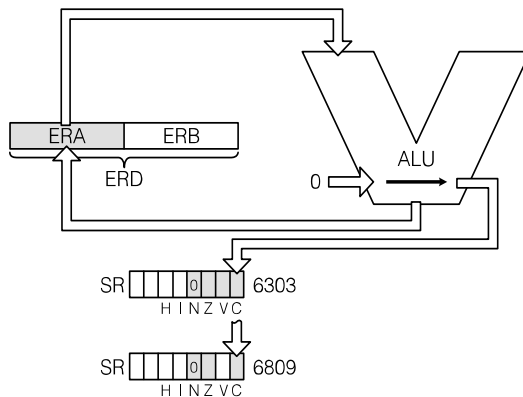
### 6303:

Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).

Motorola	LSRA	Funktion	CPU Typ A / 6303																							
B&R	SRA	Schiebe Inhalt von ERA																								
Kurz		um 1 Bit nach rechts																								
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
44																										

Motorola	LSRA	Funktion	CPU Typ B / 6809																							
B&R	SRA	Schiebe Inhalt von ERA um 1 Bit nach rechts																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
44																										

Der Inhalt des Ergebnisregisters ERA wird um 1 Bit nach rechts geschoben. Bit 0 wird in das Carry-Flag geschoben. Bit 7 wird durch log. 0 ersetzt.



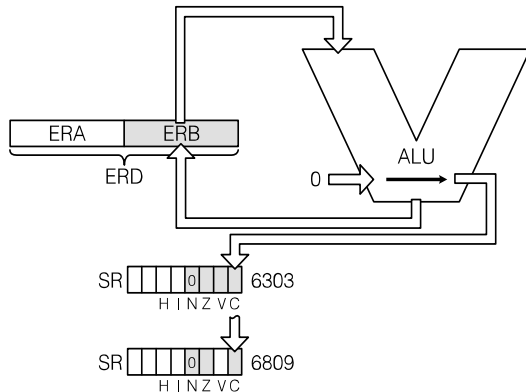
#### 6303:

Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).

Motorola	LSRB	Funktion	CPU Typ A / 6303																		
B&R	SRB	Schiebe Inhalt von ERB um 1 Bit nach rechts																			
Kurz																					
Adressierungsarten / Opcode			Adreßvorwahlen																		
			<input type="radio"/> PG1000 <input type="radio"/> PG-PC																		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H I N Z V C
54																					

Motorola	LSRB	Funktion	CPU Typ B / 6809																							
B&R	SRB	Schiebe Inhalt von ERB um 1 Bit nach rechts																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
54																										

Der Inhalt des Ergebnisregisters ERB wird um 1 Bit nach rechts geschoben. Bit 0 wird in das Carry-Flag geschoben. Bit 7 wird durch log. 0 ersetzt.



### 6303:

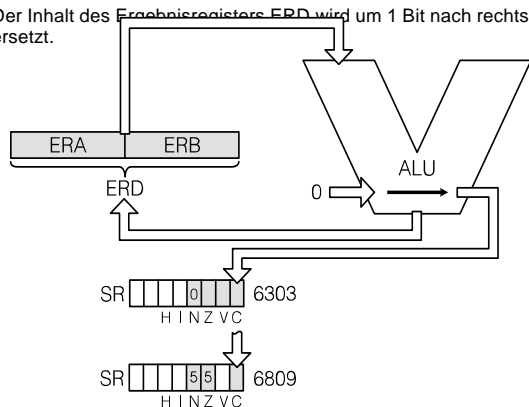
Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).

Motorola	LSRD	Funktion	CPU Typ A / 6303																							
B&R	SRD	Schiebe Inhalt von ERD 																								
Kurz			um 1 Bit nach rechts	d15 d0 C																						
Adressierungsarten / Opcode						Adreßvorwahlen						<input type="radio"/> PG1000 <input type="radio"/> PG-PC														
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
04																										

Motorola	LSRD	Funktion	CPU Typ B / 6809
B&R	SRD	Schiebe Inhalt von ERD um 1 Bit nach rechts	
Kurz			

Adressierungsarten / Opcode						Adreßvorwahlen															<input type="radio"/> CP 80 <input type="radio"/> PC 80					
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
4/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
44 56																										

Der Inhalt des Ergebnisregisters ERD wird um 1 Bit nach rechts geschoben. Bit 0 wird in das Carry-Flag geschoben. Bit 7 wird durch log. 0 ersetzt.



### 6303:

Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).

### 6809:

Da der Befehl im 6809 nicht vorhanden ist, wird er durch zwei andere Befehle realisiert:

- SRA** Der Inhalt von ERA wird um 1 Bit nach rechts geschoben. Bit 0 wird in das Carry-Flag geschoben. Bit 7 wird durch log. 0 ersetzt.
- RRB** Der Inhalt von ERB wird um 1 Bit nach rechts geschoben. Das Carry-Flag ( $\Rightarrow$  Bit 0 von ERA) wird ins Bit 7 geschoben. Bit 0 wird in das Carry-Flag geschoben.

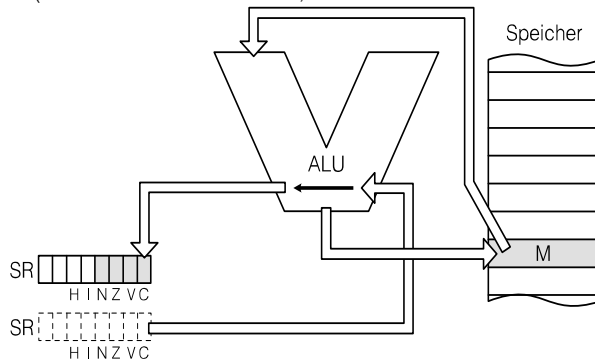
Da der SRD Befehl aus zwei Befehlen realisiert wurde, sind die Zustände von Zero- und Negativ-Flag undefiniert.

Motorola	ROL	Funktion	CPU Typ A / 6303																							
B&R	SLI	Rotiere Inhalt von M um 1 Bit nach links																								
Kurz	RL																									
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		6/3		6/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		79		69																						

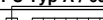
Motorola	ROL	Funktion	CPU Typ B / 6809																							
B&R	SLI	Rotiere Inhalt von M um 1 Bit nach links																								
Kurz	RL																									
Adressierungsarten / Opcode			Adreßvorwahlen																							
			<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	6/2	7/3		6+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	09	79		69																						

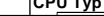
Der Inhalt der Speicherstelle M wird um 1 Bit nach links geschoben. Das Carry-Flag wird in Bit 0 geschoben. Bit 7 wird ins Carry-Flag geschoben.

Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).



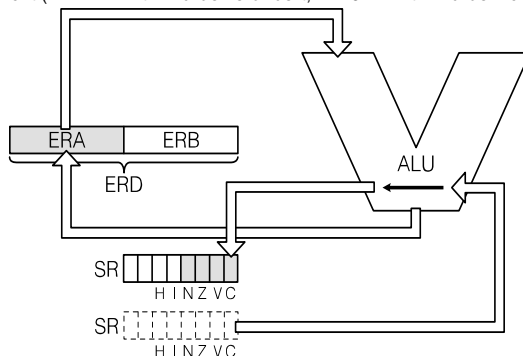


Motorola	ROLA	Funktion	CPU Typ A / 6303																		
B&R	RLA	Rotiere Inhalt von ERA																			
Kurz		um 1 Bit nach links																			
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H I N Z V C
49																					• • • • •

Motorola	ROLA	Funktion	CPU Typ B / 6809																		
B&R	RLA	Rotiere Inhalt von ERA																			
Kurz		um 1 Bit nach links																			
Adressierungsarten / Opcode			<input type="radio"/> CP 80 <input type="radio"/> PC 80																		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H I N Z V C
49																					• • • • •

Der Inhalt des Ergebnisregisters ERA wird um 1 Bit nach links geschoben. Das Carry-Flag wird in Bit 0 geschoben. Bit 7 wird ins Carry-Flag geschoben.

Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).

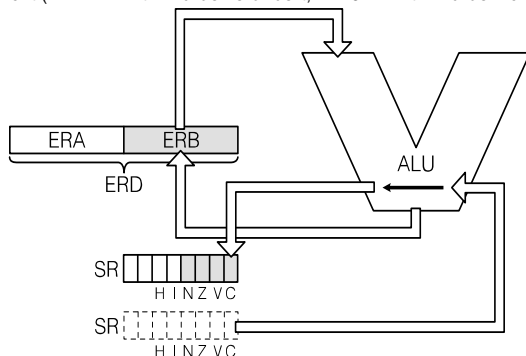


Motorola	ROLB	Funktion	CPU Typ A / 6303																							
B&R	RLB	Rotiere Inhalt von ERB																								
Kurz		um 1 Bit nach links																								
Adressierungsarten / Opcode			PG1000 PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
59																										

Motorola	ROLB	Funktion	CPU Typ B / 6809																							
B&R	RLB	Rotiere Inhalt von ERB																								
Kurz		um 1 Bit nach links																								
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
59																										

Der Inhalt des Ergebnisregisters ERB wird um 1 Bit nach links geschoben. Das Carry-Flag wird in Bit 0 geschoben. Bit 7 wird ins Carry-Flag geschoben.

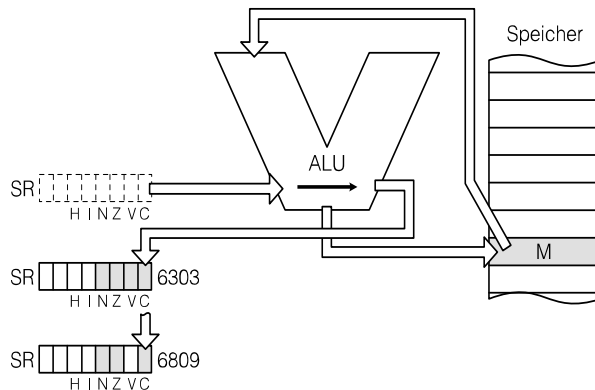
Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).



Motorola	ROR	Funktion	CPU Typ A / 6303																							
B&R	SRE	Rotiere Inhalt von M um 1 Bit nach rechts																								
Kurz	RR																									
Adressierungsarten / Opcode			Adreßvorwahlen																							
			<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		6/3		6/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		76		66																						

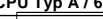
Motorola	ROR	Funktion	CPU Typ B / 6809																							
B&R	SRE	Rotiere Inhalt von M um 1 Bit nach rechts																								
Kurz	RR																									
Adressierungsarten / Opcode			Adreßvorwahlen																							
			<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		6/2	7/3		6+2/2+	E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		06	76		66																0	0				


Der Inhalt der Speicherstelle M wird um 1 Bit nach rechts geschoben. Das Carry-Flag wird ins Bit 7 geschoben. Das Bit 0 wird in das Carry-Flag geschoben.



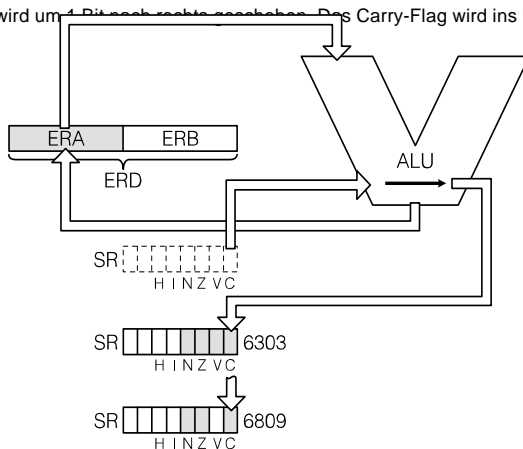
#### 6303:

Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).

Motorola	RORA	Funktion	CPU Typ A / 6303																							
B&R	RRA	Rotiere Inhalt von ERA																								
Kurz		um 1 Bit nach rechts																								
Adressierungsarten / Opcode			Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
46																										

Motorola	RORA	Funktion	CPU Typ B / 6809																							
B&R	RRA	Rotiere Inhalt von ERA																								
Kurz		um 1 Bit nach rechts																								
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
46																										

Der Inhalt des Ergebnisregisters ERA wird um 1 Bit nach rechts geschoben. Das Carry-Flag wird ins Bit 7 geschoben. Das Bit 0 wird in das Carry-Flag geschoben.



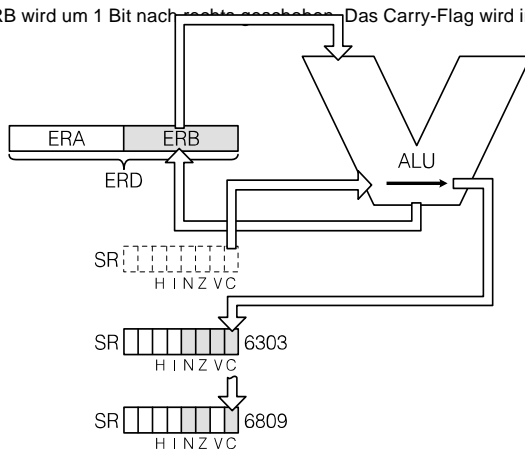
### 6303:

Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).

Motorola	RORB	Funktion	CPU Typ A / 6303
B&R	RRB	Rotiere Inhalt von ERB	
Kurz		um 1 Bit nach rechts	
Adressierungsarten / Opcode		Adreßvorwahlen	Statusregister
IMPL.	DIR.	EXT.	
1/1			
56			

Motorola	RORB	Funktion	CPU Typ B / 6809
B&R	RRB	Rotiere Inhalt von ERB	
Kurz		um 1 Bit nach rechts	
Adressierungsarten / Opcode		Adreßvorwahlen	Statusregister
IMPL.	DIR.	EXT.	
2/1			
56			

Der Inhalt des Ergebnisregisters ERB wird um 1 Bit nach rechts geschoben. Das Carry-Flag wird ins Bit 7 geschoben. Das Bit 0 wird in das Carry-Flag geschoben.



### 6303:

Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von  $N \oplus C$  zugewiesen ( $V = N \oplus C$ ). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht ( $V = 1 \Rightarrow$  Bit 7 wurde verändert;  $V = 0 \Rightarrow$  Bit 7 wurde nicht verändert).



## 3.10. SPRUNGBEFEHLE

Sprungbefehle sind Befehle, die den Programmzähler verändern.  
Es wird zwischen zwei Arten von Sprungbefehlen unterschieden:

- unbedingte Sprungbefehle
- bedingte Sprungbefehle

**Unbedingte** Sprungbefehle werden immer ausgeführt.

**Bedingte** Sprungbefehle werden nur ausgeführt, wenn bestimmte Bedingungen erfüllt sind.

Motorola	B&R	Betriebsart			
		PG1000	PG-PC	CP 80	PC 80
BCC <sup>1)</sup>	JC0 <sup>1)</sup>	○	○	○	○
BCS <sup>1)</sup>	SP< <sup>1)</sup>	○	○	○	○
BPL <sup>1)</sup>	J+ <sup>1)</sup>	○	○	○	○
BMI <sup>1)</sup>	J- <sup>1)</sup>	○	○	○	○
BNE <sup>1)</sup>	SN0 <sup>1)</sup>	○	○	○	○
BEQ <sup>1)</sup>	SP0 <sup>1)</sup>	○	○	○	○
BHI <sup>1)</sup>	SP> <sup>1)</sup>	○	○	○	○
BLS <sup>1)</sup>	J<= <sup>1)</sup>	○	○	○	○
SK0	SK0		○		○
SK1	SK1		○		○
JSR	SPU	○	○	○	○
RTS	RET	○	○	○	○
JMP	SPI	○	○	○	○
NOP	NOP	○	○	○	○
END	END	○	○	○	○

<sup>1)</sup> **Kurze** relative Sprungbefehle (Sprungweite: -128 bis +127).  
Durch Anhängen eines "L" wird aus einem kurzen ein **langer**  
Sprungbefehl (Sprungweite: -32768 bis +32767). Dies ist nur im  
PC 80 Modus möglich!

**Beispiel:** BCCL oder JC0L

Motorola	xxx	Funktion														CPU Typ A / 6303														
B&R	xxx	Springe, wenn Bedingung xxx erfüllt ist																												
Kurz	xxx																													
Adressierungsarten / Opcode												Adreßvorwahlen														<input type="radio"/> PG1000				
																										<input type="radio"/> PG-PC				
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister									
					3/2	E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C				
					xxx																									

“xxx” in obiger Tabelle sind Platzhalter, die in der folgenden Tabelle definiert werden:

Mnemonics			Bedingung	Opcode
Motorola	B&R	Kurz		
BCC	JC0		C = 0	24
BCS	SP<	J<	C = 1	25
BEQ	SP0	J0	Z = 1	27
BHI	SP>	J>	C v Z = 0	22
BLS	J<=		C v Z = 1	23
BMI	J-		N = 1	2B
BNE	SN0	J1	Z = 0	26
BPL	J+		N = 0	2A

Motorola	xxx	Funktion														CPU Typ B / 6809											
B&R	xxx	Springe, wenn Bedingung xxx erfüllt ist																									
Kurz	xxx																										
Adressierungsarten / Opcode												Adreßvorwahlen														<input type="radio"/> CP 80	
																										<input type="radio"/> PC 80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
					3/2	E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
					xxx																<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	

**Kurze relative Sprünge** bestehen aus zwei Bytes:

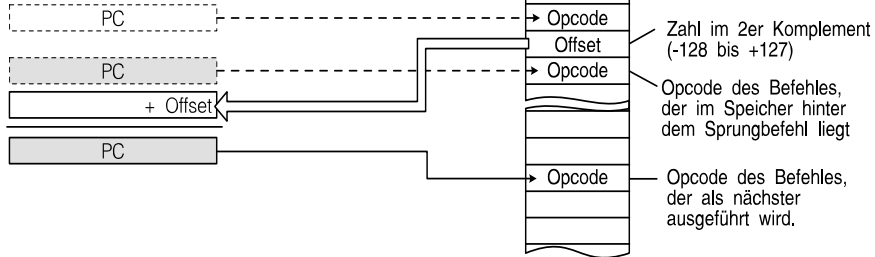
- Das erste Byte ist der Opcode des Befehles.
- Das zweite Byte ist der Offset, der angibt, wo das Programm fortgesetzt werden soll, wenn die Bedingung erfüllt ist. Dieser Offset ist eine Zahl im 2er-Komplement, d.h. er kann Werte zwischen -128 und +127 annehmen. Der Offset bezieht sich immer auf die Adresse des nächsten Befehles (Befehl nach dem bedingten Sprung). In der AWL-Eingabezeile wird als Sprungziel ein Label eingegeben. Das B&R PROgrammierSYstem berechnet den Offset selbst. Sollte ein Sprung nicht möglich sein, weil er außerhalb des Bereiches (-128 bis +127) liegt, wird dies durch ein “+” vor dem Label gekennzeichnet. Das Programm kann in diesem Fall nicht in die SPS überspielt werden. Folgende Fehlermeldung wird ausgegeben:

**E051 Sprung zu groß**



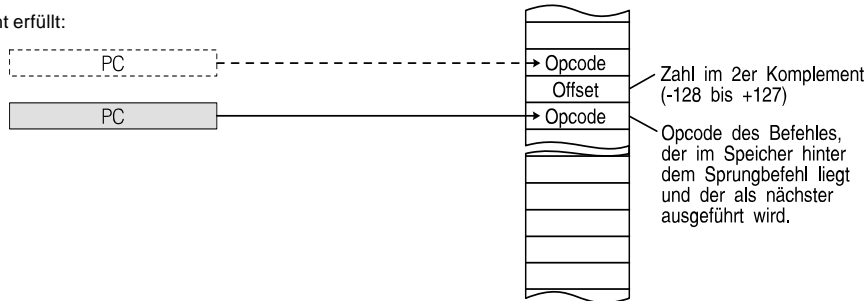
Speicher

Bedingung ist erfüllt:



Speicher

Bedingung ist nicht erfüllt:



Motorola		Funktion										CPU Typ A / 6303									
B&R																					
Kurz																					
Adressierungsarten / Opcode						Adreßvorwahlen						PG1000									
												PG-PC									
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H

“xxx” in obiger Tabelle sind Platzhalter, die in der folgenden Tabelle definiert werden:

Mnemonics			Bedingung	Opcode
Motorola	B&R	Kurz		
BCCL	JC0L		C = 0	10 24
BCSL	SP<L		C = 1	10 25
BEQL	SP0L		Z = 1	10 27
BHIL	SP>L		C ∨ Z = 0	10 22
BLSL	J<=L		C ∨ Z = 1	10 23
BMIL	J-L		N = 1	10 2B
BNEL	SN0L		Z = 0	10 26
BPLL	J+L		N = 0	10 2A

Motorola		Funktion										CPU Typ B / 6809									
B&R		xxx										Springe, wenn Bedingung xxx erfüllt ist									
Kurz																					
Adressierungsarten / Opcode						Adreßvorwahlen						CP 80									
												PC 80									
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister

**Lange** relative Sprünge bestehen aus vier Bytes:

- Die Bytes 1 und 2 sind der Opcode des Befehles.
- Die Bytes 3 und 4 sind der Offset, der angibt, wo das Programm fortgesetzt werden soll, wenn die Bedingung erfüllt ist. Dieser Offset ist eine Zahl im 2er-Komplement, d.h. er kann Werte zwischen -32768 und +32767 annehmen. Der Offset bezieht sich immer auf die Adresse des nächsten Befehles (Befehl nach dem bedingten Sprung). Mit dem Zwei-Byte Offset kann der ganze Speicherbereich einer SPS abgedeckt werden, d.h. es gibt keine Begrenzung wie für kurze relative Sprungbefehle. In der AWL-Eingabezeile wird als Sprungziel ein Label eingegeben. Das B&R PROGRAMMIERSYSTEM berechnet den Offset selbst.

**Ausführungszeit:** 5(6) Taktzyklen

- Bedingung ist erfüllt, Sprung wird ausgeführt.
- Bedingung ist nicht erfüllt, Sprung wird nicht ausgeführt.

Speicher

Bedingung ist erfüllt:

PC

PC

+ Offset

PC

10  
xx

Offset (MSB)

Offset (LSB)

Opcode

Opcode

Opcode des Sprungbefehles

Zahl im 2er Komplement  
(-32768 bis +32767)

Opcode des Befehles,  
der im Speicher hinter  
dem Sprungbefehl liegt

Opcode des Befehles,  
der als nächster  
ausgeführt wird.

Speicher

Bedingung ist nicht erfüllt:

PC

PC

10  
xx

Offset (MSB)

Offset (LSB)

Opcode

Opcode des Sprungbefehles

Zahl im 2er Komplement  
(-32768 bis +32767)

Opcode des Befehles,  
der im Speicher hinter  
dem Sprungbefehl liegt  
und der als nächster  
ausgeführt wird.

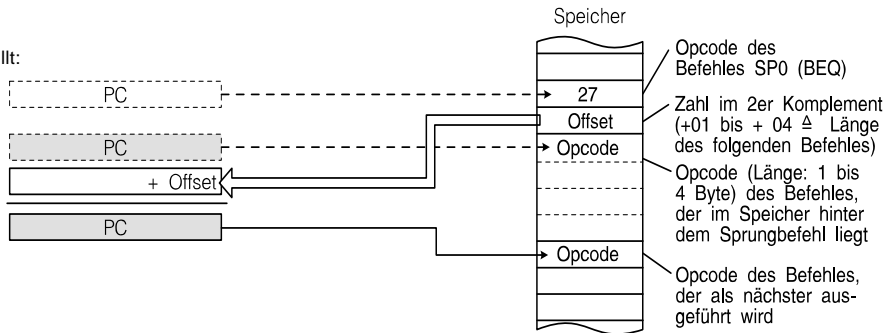
Motorola	SK0	Funktion	CPU Typ A / 6303																							
B&R	SK0	Überspringe nächsten Befehl, wenn Z = 1																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div>PG1000</div> <div><div></div>PG-PC</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
					3/2	E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
					27																					

Motorola	SK0	Funktion	CPU Typ B / 6809																							
B&R	SK0	Überspringe nächsten Befehl, wenn Z = 1																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<div>CP 80</div> <div>PC 80</div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
					3/2	E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
					27																					

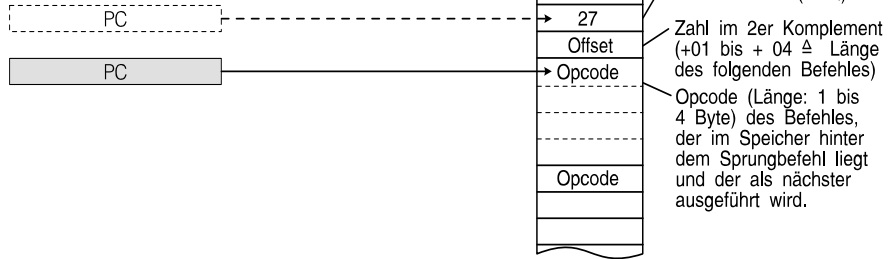
Der folgende Befehl wird übersprungen, wenn das Zero-Flag log. 1 ist (Ergebnis der vorhergehenden Operation war Null).

Dieser Befehl entspricht dem SP0 (BEQ) Befehl. Der Offset für den Sprungbefehl entspricht der Opcode-Länge des folgenden Befehles. Das PROgrammierSYstem setzt diesen Wert automatisch ein.

Bedingung ist erfüllt:



Bedingung ist nicht erfüllt:



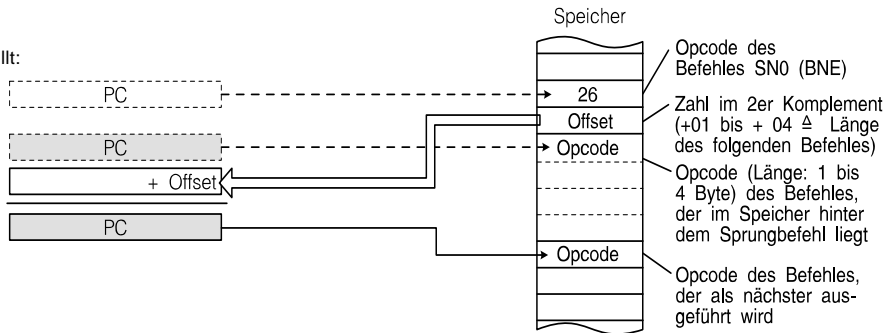
Motorola	SK1	Funktion																CPU Typ A / 6303											
B&R	SK1	Überspringe nächsten Befehl, wenn Z = 0																											
Kurz																													
Adressierungsarten / Opcode												Adreßvorwahlen																<input type="checkbox"/> PG1000 <input checked="" type="checkbox"/> PG-PC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister								
					3/2	E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C			
					26																								

Motorola	SK1	Funktion														CPU Typ B / 6809										
B&R	SK1	Überspringe nächsten Befehl, wenn Z = 0																								
Kurz																										
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="checkbox"/> CP 80 <input checked="" type="checkbox"/> PC 80		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
					3/2	E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
					26																					

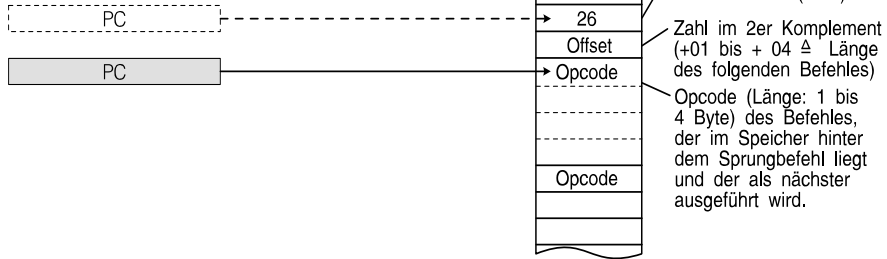
Der folgende Befehl wird übersprungen, wenn das Zero-Flag log. 0 ist (Ergebnis der vorhergehenden Operation war ungleich Null).

Dieser Befehl entspricht dem SN0 (BNE) Befehl. Der Offset für den Sprungbefehl entspricht der Opcode-Länge des folgenden Befehles. Das PROgrammierSYStem setzt diesen Wert automatisch ein.

Bedingung ist erfüllt:



Bedingung ist nicht erfüllt:



Motorola	JSR	Funktion	CPU Typ A / 6303																							
B&R	SPU	Unbedingter Sprung in ein Unterprogramm																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	5/2	6/3		5/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	9D	BD		AD																						

Motorola	JSR	Funktion														CPU Typ B / 6809										
B&R	SPU	Unbedingter Sprung in ein Unterprogramm																								
Kurz																										
Adressierungsarten / Opcode										Adreßvorwahlen										<input type="radio"/> CP 80 <input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
	7/2	8/3		7+2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
	9D	BD		AD																						

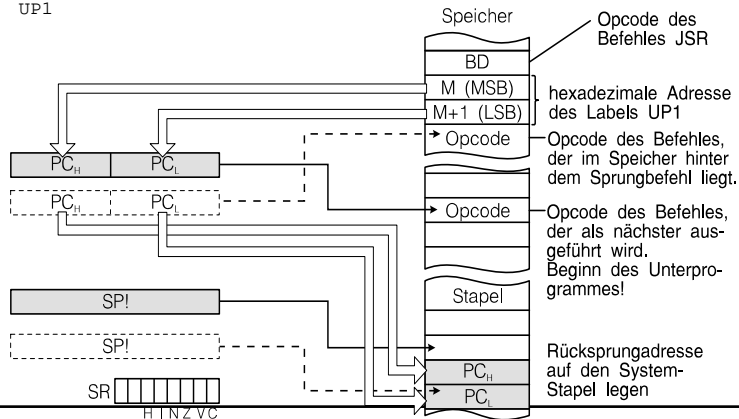
Dieser Befehl veranlaßt eine Verzweigung zu der angegebenen Stelle im Programm. Vor dem Sprung wird die Adresse des Befehles (=> Rücksprungadresse), der im Speicher hinter dem Sprungbefehl liegt, auf den System-Stapel gelegt.

Wird in der AWL-Eingabezeile ein Label eingegeben, zu dem das Programm verzweigen soll, ersetzt das PROgrammierSYSTEM diesen durch die hexadezimale Adresse, die der Label im Speicher der SPS hat.

Die Stapelverarbeitung unterscheidet sich bei den beiden CPU-Typen etwas:

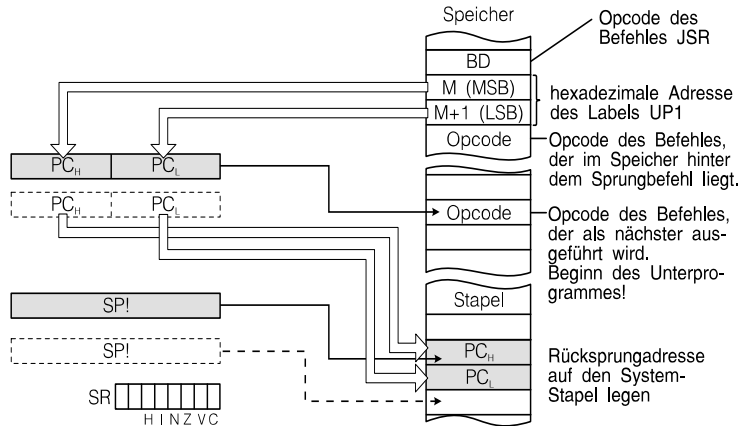
**Beispiel:** SPU UP1

**6303:**





**6809:**



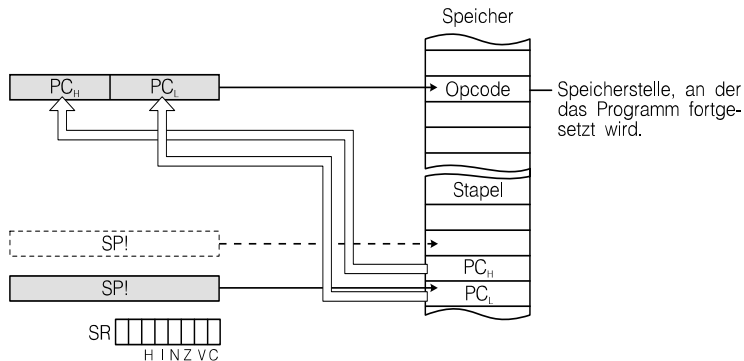
Motorola	RTS	Funktion	CPU Typ A / 6303																							
B&R	RET	Rücksprung vom Unterprogramm																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
5/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
39																										

Motorola	RTS	Funktion	CPU Typ B / 6809																							
B&R	RET	Rücksprung vom Unterprogramm																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
5/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
39																										

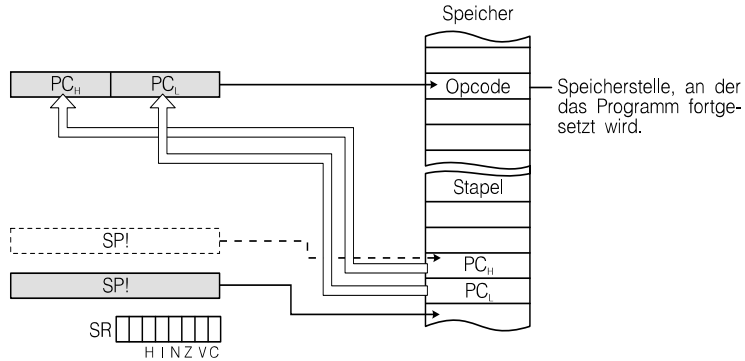
Dieser Befehl veranlaßt den Rücksprung zu der Adresse, die auf dem System-Stapel liegt. Wurde ein sog. Unterprogramm mit dem Befehl JSR aufgerufen, muß dieses mit RTS abgeschlossen werden, da sonst ein "STACKFEHLER" auftreten könnte.

Die Stapelverarbeitung unterscheidet sich bei den beiden CPU-Typen etwas:

### 6303:



**6809:**



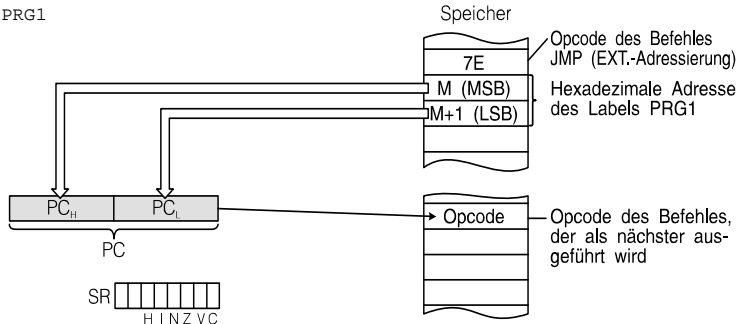
Motorola	JMP	Funktion														CPU Typ A / 6303											
B&R	SPI	Unbedingter Sprung																									
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> PG1000 <input type="radio"/> PG-PC			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister						
		3/3		3/2		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C	
		7E		6E																							

Motorola	JMP	Funktion														CPU Typ B / 6809										
B&R	SPI	Unbedingter Sprung																								
Kurz																										
Adressierungsarten / Opcode										Adreßvorwahlen										<input type="radio"/>	CP 80					
																				<input type="radio"/>	PC 80					
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
	3/2	4/3		3+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
	0E	7E		6E																						

Dieser Befehl veranlaßt eine Verzweigung zu der angegebenen Stelle im Programm.

Wird in der AWL-Eingabezeile ein Label eingegeben, zu dem das Programm verzweigen soll, ersetzt das PROgrammierSYStem diesen durch die hexadezimale Adresse, die der Label im Speicher der SPS hat.

**Beispiel:** SPI PRG1



Motorola	NOP	Funktion	CPU Typ A / 6303																							
B&R	NOP	Keine Funktion																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
01																										

Dieser Befehl hat **keine** Funktion. Er bewirkt lediglich, daß eine bestimmte Zeit ( $\Rightarrow$  1 Prozessorzyklus) nichts getan wird.

Motorola	NOP	Funktion														CPU Typ B / 6809										
B&R	NOP	Keine Funktion																								
Kurz																										
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/>	CP 80	
																								<input type="radio"/>	PC 80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
12																										

Dieser Befehl hat **keine** Funktion. Er bewirkt lediglich, daß eine bestimmte Zeit ( $\Rightarrow$  2 Prozessorzyklen) nichts getan wird.

Motorola	END	Funktion	CPU Typ A / 6303																							
B&R	END	Programmende!																								
Kurz		Beginne in Programmzeile 0																								
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
						E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
			7E C0 00																							

Motorola	END	Funktion	CPU Typ B / 6809																							
B&R	END	Programmende!																								
Kurz		Beginne in Programmzeile 0																								
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
						E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
11 3F																					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Dieser Befehl bewirkt einen Sprung in das Betriebssystem der SPS. Dieser Sprung wird bei den beiden CPU-Typen unterschiedlich realisiert:

**6303:** Mit einem unbedingten Sprung (SPI / JMP) an die Adresse \$C000 wird direkt in das Betriebssystem verzweigt.

**6809:** Durch einen Softwareinterrupt wird das Betriebssystem aufgerufen.

Nach der Durchführung einiger Tests (z.B.: Test auf Stackfehler) verzweigt das Betriebssystem zu Zeile 0 des Anwenderprogrammes.

## 3.11. SONSTIGES

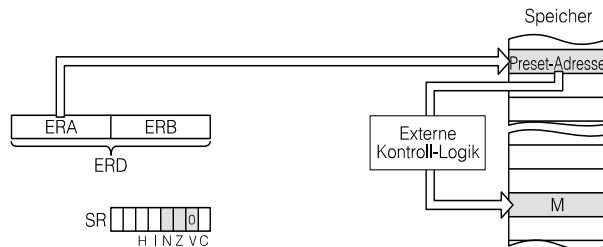
Motorola	B&R	Betriebsart			
		PG1000	PG-PC	CP 80	PC 80
PRS	PRS	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
RST	RST	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
CLR	CLR	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
CLRA	CLA		<input type="radio"/>		<input type="radio"/>
CLRB	CLB		<input type="radio"/>		<input type="radio"/>
SET	SET	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
CLC	CLC	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
SEC	SEC	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
CLI	CLI	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
SEI	SEI	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
COM	K	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
COMA	COA		<input type="radio"/>		<input type="radio"/>
COMB	COB		<input type="radio"/>		<input type="radio"/>
DAA	DK	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Motorola	PRS	Funktion	CPU Typ A / 6303																							
B&R	PRS	Wenn d0 von ERA = 1: 1 ⇒ (M)																								
Kurz	P	Wenn d0 von ERA = 0: (M) bleibt unverändert																								
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="checkbox"/> PG1000 <input type="checkbox"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		4/3				E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		B7				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>													<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Motorola	PRS	Funktion	CPU Typ B / 6809																							
B&R	PRS	Wenn d0 von ERA = 1: 1 ⇒ (M)																								
Kurz	P	Wenn d0 von ERA = 0: (M) bleibt unverändert																								
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		5/3				E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		B7				<input type="radio"/>	<input type="radio"/>	<input type="radio"/>													<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Die 1 Bit Speicherstelle M wird mit log. 1 geladen, wenn das Datenbit 0 von ERA log. 1 ist, andernfalls bleibt der Inhalt von M unverändert.

Die Prozessoren 6303 und 6809 besitzen den Befehl PRS nicht. PRS ist eigentlich ein Speicherbefehl (= / STAA). Die Preset-Funktion wird hardwaremäßig realisiert. Dazu gibt es für die 1 Bit Speicherstellen mit den Adreßvorwahlen A, M und F sogenannte Preset-Adressen. Das PROgrammierSYStem ersetzt beim Überspielen in die SPS die eingegebene Adresse M durch die entsprechende Preset-Adresse.



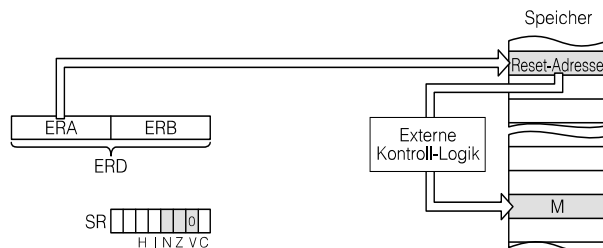
**PRESET für erweiterte Ausgänge (G bis N) ist aus Speichergründen nicht möglich!**



Motorola	RST	Funktion	CPU Typ A / 6303																							
B&R	RST	Wenn d0 von ERA = 1: 0 ⇒ (M)																								
Kurz	R	Wenn d0 von ERA = 0: (M) bleibt unverändert																								
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		4/3				E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		B7				<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Motorola	RST	Funktion	CPU Typ B / 6809																							
B&R	RST	Wenn d0 von ERA = 1: 0 ⇒ (M)																								
Kurz	R	Wenn d0 von ERA = 0: (M) bleibt unverändert																								
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		5/3				E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		B7				<input type="radio"/>	<input type="radio"/>	<input type="radio"/>												<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Die 1 Bit Speicherstelle M wird mit log. 0 geladen, wenn das Datenbit 0 von ERA log. 1 ist, andernfalls bleibt der Inhalt von M unverändert. Die Prozessoren 6303 und 6809 besitzen den Befehl RST nicht. Die Reset-Funktion wird hardwaremäßig realisiert. RST ist eigentlich ein Speicherbefehl (= / STAA) auf eine sogenannte Reset-Adresse. Allen 1 Bit Speicherstellen mit den Adreßvorwahlen M, A und F ist solch eine Reset-Adresse zugeordnet. Das PROgrammierSYSTEM ersetzt beim Überspielen in die SPS die eingegebene Adresse M durch die entsprechende Reset-Adresse.

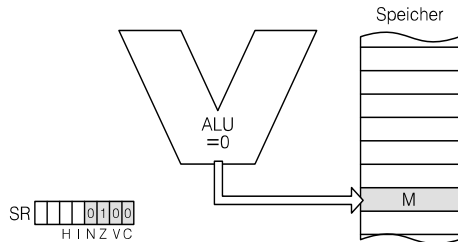


**RESET für erweiterte Ausgänge (G bis N) ist aus Speichergründen nicht möglich!**

Motorola	CLR	Funktion														CPU Typ A / 6303										
B&R	CLR	0 ⇒ (M)																								
Kurz	C																									
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		5/3		5/2		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		7F		6F		⊙	⊙	⊙			⊙	⊙	⊙	⊙							⊙	⊙	▼	▲	▼	▼

Motorola	CLR	Funktion														CPU Typ B / 6809											
B&R	CLR	0 ⇒ (M)																									
Kurz	C																										
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> CP 80 <input type="radio"/> PC 80			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
	6/2	7/3		6+/2+		E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
	0F	7F		6F		⊙	⊙	⊙			⊙	⊙	⊙	●	●	●	●	●	●	●	⊙	⊙	▼	▲	▼	▼	

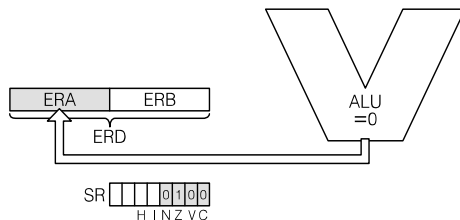
Die Speicherstelle M wird gelöscht, d.h. sie erhält den Wert log. 0.



Motorola	CLRA	Funktion	CPU Typ A / 6303																							
B&R	CLA	0 ⇒ ERA																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
4F																						◁	◃	▼	▲	▷

Motorola	CLRA	Funktion	CPU Typ B / 6809																							
B&R	CLA	0 ⇒ ERA																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
4F																										

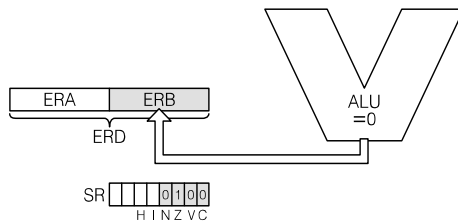
Das Ergebnisregister ERA wird gelöscht, d.h. es erhält den Wert log. 0.



Motorola	CLRB	Funktion	CPU Typ A / 6303																							
B&R	CLB	0 ⇒ ERB																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
5F																					0	0	▼	▲	▼	▼

Motorola	CLRB	Funktion	CPU Typ B / 6809																							
B&R	CLB	0 ⇒ ERB																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
5F																					0	0	▼	▲	▼	▼

Das Ergebnisregister ERB wird gelöscht, d.h. es erhält den Wert log. 0.

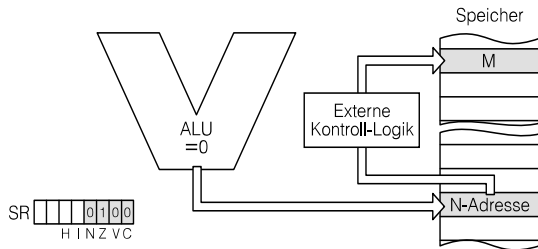


Motorola	SET	Funktion														CPU Typ A / 6303										
B&R	SET	1 ⇨ (M)																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen												<input type="radio"/>	PG1000							
																		<input type="radio"/>	PG-PC							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		5/3				E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		7F				<input type="radio"/>	<input type="radio"/>	<input type="radio"/>													<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Motorola	SET	Funktion														CPU Typ B / 6809										
B&R	SET	1 ⇨ (M)																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80						
																				<input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		7/3				E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		7F				0	0	0													0	0	▲	▼	▲	▼

Die 1 Bit Speicherstelle M wird mit log 1 geladen.

Die Prozessoren 6303 und 6809 besitzen den Befehl SET nicht in ihrem Grundbefehlssatz. Die SET-Funktion wird hardwaremäßig realisiert. SET ist eigentlich ein CLR Befehl, der eine sogenannte N-Adresse (bewirkt eine Negation) anspricht. Allen 1 Bit Speicherstellen mit den Adreßvorwahlen M, A und F ist solch eine N-Adresse zugeordnet. Das PROgrammierSYSTEM ersetzt beim Überspielen in die SPS die eingegebene Adresse M durch die entsprechende N-Adresse.

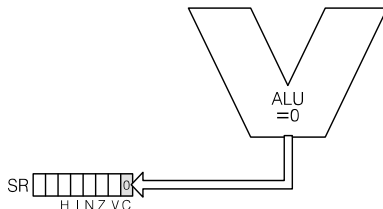


**SET für erweiterte Ausgänge (G bis N) ist aus Speichergründen nicht möglich!**

Motorola	CLC	Funktion														CPU Typ A / 6303											
B&R	CLC	0 ⇒ C																									
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/> PG1000 <input type="radio"/> PG-PC			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
0C																											

Motorola	CLC	Funktion	CPU Typ B / 6809																							
B&R	CLC	0 ⇒ C																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
3/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
1C FE																										

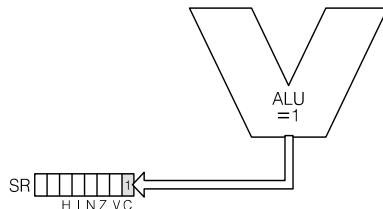
Das Carry-Flag wird gelöscht, d.h. es erhält den Wert log. 0.



Motorola	SEC	Funktion														CPU Typ A / 6303										
B&R	SEC	1 ⇨ C																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000						
																				<input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U <td>I</td> <td>B</td> <td>G</td> <td>H</td> <td>I</td> <td>N</td> <td>Z</td> <td>V</td> <td>C</td>	I	B	G	H	I	N	Z	V	C
0D																										

Motorola	SEC	Funktion														CPU Typ B / 6809										
B&R	SEC	1 ⇨ C																								
Kurz																										
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80						
																				<input type="radio"/> PC 80						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
3/2						E	A	M	F	Z	#	P	C	I	Y	D <td>U<td>I</td><td>B</td><td>G</td><td>H</td><td>I</td><td>N</td><td>Z</td><td>V</td><td>C</td></td>	U <td>I</td> <td>B</td> <td>G</td> <td>H</td> <td>I</td> <td>N</td> <td>Z</td> <td>V</td> <td>C</td>	I	B	G	H	I	N	Z	V	C
1A 01																										

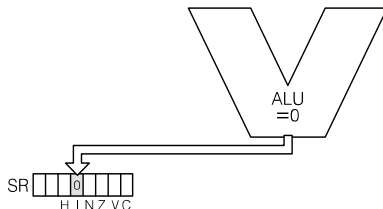
Das Carry-Flag wird gesetzt, d.h. es erhält den Wert log. 1.



Motorola	CLI	Funktion														CPU Typ A / 6303					
B&R	CLI	0 ⇒ I																			
Kurz																					
Adressierungsarten / Opcode						Adreßvorwahlen												<input type="radio"/> PG1000 <input type="radio"/> PG-PC			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H   I   N   Z   V   C
0E																					◊   ▼   ◊   ◊   ◊   ◊

Motorola	CLI	Funktion														CPU Typ B / 6809					
B&R	CLI	0 ⇨ I																			
Kurz																					
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> CP 80 <input type="radio"/> PC 80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister
3/2						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H   I   N   Z   V   C
1C EF																					<input type="radio"/> ▾ <input type="radio"/> <input type="radio"/> <input type="radio"/>

Das IRQ Mask Bit wird gelöscht, d.h. es erhält den Wert log. 0 (Aufheben des Befehles SEI). Auftretende Interrupts (Unterbrechungen) werden nicht verhindert.

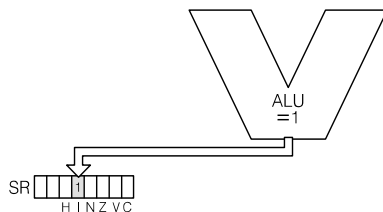




Motorola	SEI	Funktion	CPU Typ A / 6303																							
B&R	SEI	1 ⇨ I																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H	I	N	Z	V	C
0F																										

Motorola	SEI	Funktion	CPU Typ B / 6809																		
B&R	SEI	1 ⇨ I																			
Kurz																					
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	!	B	G	Statusregister
3/2						E	A	M	F	Z	#	P	C	I	Y	D	U	!	B	G	H I N Z V C
1A 10																					◀ ▶ ↻ ↺

Das IRQ Mask Bit wird gesetzt d.h. es erhält den Wert log. 1. Tritt ein Interrupt (Unterbrechung) auf, wird dieser verhindert und es wird nicht in das Interrupt-Programm verzweigt.



#### Hinweis:

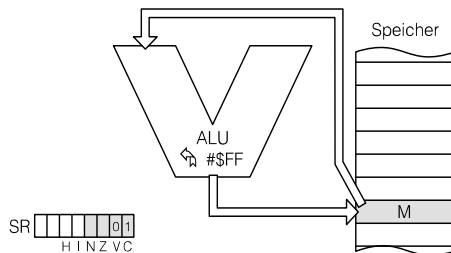
Wird das IRQ Mask Bit I (Bit 4 des Statusregisters) vom Anwender auf log. 1 gesetzt, werden alle Interrupts gesperrt. Diese Technik wird dann verwendet, wenn Programmteile des Anwenderprogrammes in keinem Fall durch einen Interrupt unterbrochen werden dürfen.

**Bevor der Befehl END abgearbeitet wird, muß das Bit I mit dem Befehl CLI unbedingt wieder auf log. 0 gesetzt werden.**

Motorola	COM	Funktion	CPU Typ A / 6303																							
B&R	K	(M) ⊕ #\$FF ⇨ (M) (⇒ Negation)																								
Kurz																										
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
		6/3	6/2			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
		73	63			<input type="radio"/>	<input type="radio"/>	<input type="radio"/>				<input type="radio"/>	<input type="radio"/>	<input type="radio"/>							<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Motorola	COM	Funktion														CPU Typ B / 6809											
B&R	K	(M) ⊕ #\$FF ⇨ (M) (⇒ Negation)																									
Kurz																											
Adressierungsarten / Opcode												Adreßvorwahlen												<input type="radio"/>	CP 80		
																								<input type="radio"/>	PC 80		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
	6/2	7/3	6+2+			E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
	03	73	63			0	0	0				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

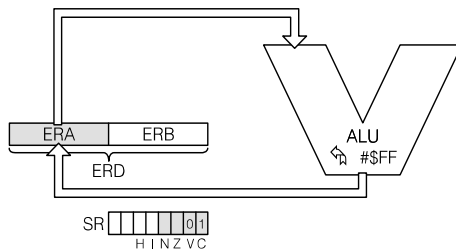
Der Inhalt der Speicherstelle M wird invertiert (= Exklusiv-Oder-Verknüpfung mit  $\$FF$ ).



Motorola	COMA	Funktion														CPU Typ A / 6303										
B&R	COA	ERA ⊕ #\$FF ⇒ ERA (⇒ Negation)																								
Kurz	KA																									
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000 <input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
43																										

Motorola	COMA	Funktion														CPU Typ B / 6809											
B&R	COA	ERA ⊕ #\$FF ⇒ ERA (⇒ Negation)																									
Kurz	KA																										
Adressierungsarten / Opcode										Adreßvorwahlen										<input type="radio"/> CP 80 <input type="radio"/> PC 80							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister						
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C	
43																						<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

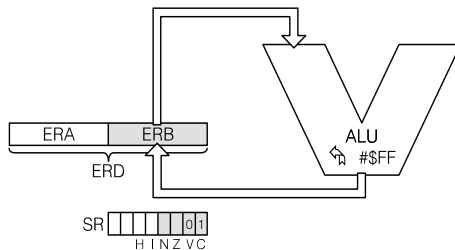
Der Inhalt des Ergebnisregisters ERA wird invertiert (= Exklusiv-Oder-Verknüpfung mit #\$FF).



Motorola	COMB	Funktion														CPU Typ A / 6303										
B&R	COB	ERB ⊕ #\$FF ⇒ ERB (⇒ Negation)																								
Kurz	KB																									
Adressierungsarten / Opcode						Adreßvorwahlen														<input type="radio"/> PG1000						
																				<input type="radio"/> PG-PC						
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
1/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
53																										

Motorola	COMB	Funktion	CPU Typ B / 6809																							
B&R	COB	ERB ⊕ \$FF ⇒ ERB (⇒ Negation)																								
Kurz	KB																									
Adressierungsarten / Opcode		Adreßvorwahlen	<div><div>CP 80</div><div>PC 80</div></div>																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
53																										

Der Inhalt des Ergebnisregisters ERB wird invertiert (= Exklusiv-Oder-Verknüpfung mit #\$FF).

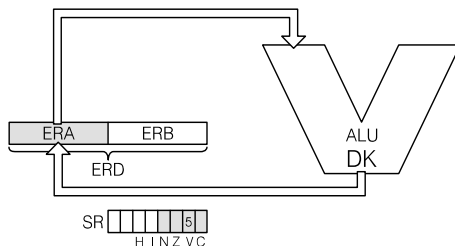


Motorola	DAA	Funktion	CPU Typ A / 6303																							
B&R	DK	Dezimalkorrektur von ERA nach einer Addition																								
Kurz		von BCD Zahlen																								
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> PG1000 <input type="radio"/> PG-PC																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
19																										

Motorola	DAA	Funktion	CPU Typ B / 6809																							
B&R	DK	Dezimalkorrektur von ERA nach einer Addition																								
Kurz		von BCD Zahlen																								
Adressierungsarten / Opcode		Adreßvorwahlen	<input type="radio"/> CP 80 <input type="radio"/> PC 80																							
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	O	F	S	T	#	P	R	X	Y	D	U	I	B	G	Statusregister					
2/1						E	A	M	F	Z	#	P	C	I	Y	D	U	I	B	G	H	I	N	Z	V	C
19																					<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Das nach der Addition von BCD-Zahlen entstandene binäre Ergebnis (in ERA) wird wieder in eine BCD-Zahl umgewandelt und im Ergebnisregister ERA abgelegt. Dieser Befehl funktioniert nur einwandfrei, wenn er unmittelbar nach einer Addition (ADD, + oder A+B) durchgeführt wird.

Das Carry-Flag wird dann auf log. 1 gesetzt, wenn das BCD-Ergebnis der Addition größer als 99 ist.



**Beispiel:** Addition der BCD-Zahlen \$27 und \$96: ERA = \$BD

Dezimalkorrektur:

ERA = \$23

C = 1

(=> Übertrag auf die Hunderter-Stelle)



## 4. MATHEMATIK-ROUTINEN

### 4.1. ALLGEMEINES

Alle Zentraleinheiten und Peripherieprozessoren sind standardmäßig mit Mathematik-Routinen ausgestattet. Diese Routinen sind Bestandteil des Betriebssystems. Sie werden durch **Befehls-Mnemonics** aus der Anweisungsliste aufgerufen. Neben den Grundrechenarten Addition, Subtraktion, Multiplikation, Division und Quadratwurzel stehen zahlreiche Umwandlungs- und Hilfsprogramme zur Verfügung (z.B. zum Vergleichen oder Kopieren). Funktionsblöcke können ebenfalls auf die Mathematik-Routinen zugreifen. Zur Zahlendarstellung wird das genormte 4 Byte **IEEE-Format** verwendet.

**MATHEMATIK-ROUTINEN DÜRFEN NICHT IN INTERRUPTPROGRAMMEN BETRIEBEN WERDEN.**

### Operanden und Speicher

Die Mathematik-Routinen belegen einige Speicher im Anwender-Datenbereich:

C1024	Fehlernummer	} IEEE-Format
C1025	reserviert	
C1026 bis C1029	Operand 1 (OP1)	
C1030 bis C1033	Operand 2 (OP2)	
C1034 bis C1037	Zwischenspeicher 1 (MEM1)	
C1038 bis C1041	Zwischenspeicher 2 (MEM2)	
C1042 bis C1045	Zwischenspeicher 3 (MEM3)	
C1046 bis C1047	Quelladresse	
C1048 bis C1049	Zieladresse	
C1050 bis C1051	Länge	
C1052 bis C1053	Daten	

## FEHLERMELDUNGEN

Tritt bei der Ausführung einer Mathematik-Routine ein Fehler auf, so wird das Carry-Flag gesetzt und die Speicherstelle C1024 enthält eine Fehlernummer:

Fehlernummer	Kurzbezeichnung	Beschreibung
1	MATH_OVERFLOW	Bei einer Berechnung wurde der darstellbare Zahlenbereich überschritten
2	MATH_UNDERFLOW	Bei einer Berechnung wurde der darstellbare Zahlenbereich unterschritten
3	DIV_BY_ZERO	Division durch 0
4	CONV_OVERFLOW	Bereichsüberschreitung beim Umwandeln von Zahlenformaten
5	CUT_LSB	Beschneidung des Lower Significant Byte (LSB) beim Laden von 4 Byte-Mantissen
6	LOAD_OVERFLOW	Bereichsüberschreitung beim Laden von Zahlen
7	LOAD_UNDERFLOW	Bereichsunterschreitung beim Laden von Zahlen
8	NEG_SQRT	Negativer Operand bei Quadratwurzelberechnung
9	INVAL_CHAR	Unzulässiges Zeichen bei Stringumwandlungsroutine
10	NO_FPC	Kein Floating Point-Coprozessor installiert (TRAP-Fehler wird ausgelöst)
11	INVAL_COMMAND	Unzulässiges Kommando (TRAP-Fehler wird ausgelöst)
12	NOT_A_NUMBER	Zahl nicht im Rechenbereich
13	INCH_EXP	Exponentfehler bei Inch-Metrisch- bzw. Metrisch-Inch-Umwandlung
14	INCH_OVERFLOW	Datenüberlauf bei Inch-Metrisch- bzw. Metrisch-Inch-Umwandlung



# ZAHLENFORMATE

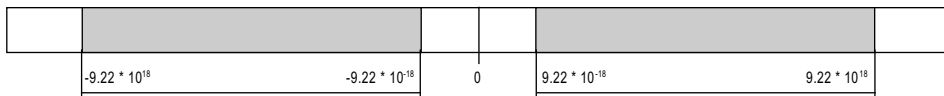
## Einfachgenaues Fließkommaformat



**Umrechnung:**

$$(-1)^S \cdot 2^{(EXP-127)} \cdot 1.\text{Mantisse}$$

**Darstellbarer Zahlenbereich:**

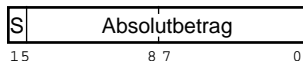


Zahlen von  $-9.22 \cdot 10^{18}$  bis  $+9.22 \cdot 10^{18}$  können mit Ausnahme von 0 nicht dargestellt werden und werden behandelt wie 0.

# Absolut mit Vorzeichen

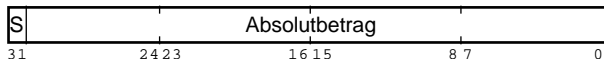
S ... Vorzeichen

## Absolut-Kurz



$\pm 32767$   
 $\pm(2^{15} - 1)$

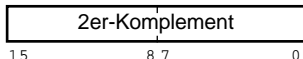
## Absolut-Lang



$\pm 2147483647$   
 $\pm(2^{31} - 1)$

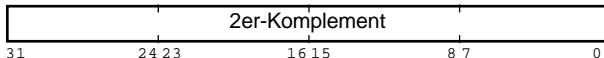
# Integer

## Integer-Kurz



$-32768 (-2^{15})$   
 bis  
 $32767 (2^{15}-1)$

## Integer-Lang



$-2147483648 (-2^{31})$   
 bis  
 $2147483647 (2^{31}-1)$   
 $(\pm 2,15 * 10^9)$

MADD		Addition im Fließkommaformat															
Ausführungszeit in $\mu$ s	6303	209 - 690	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	207 - 503		○	○				○	○					○		
<b>Funktion:</b> Die Operanden OP1 und OP2 werden addiert. Das Ergebnis wird in OP1 gespeichert. OP2 bleibt unverändert.																	
<b>Übergabe:</b> keine																	
<b>Ergebnis:</b> ERD verändert N, Z entsprechend dem Ergebnis der Routine																	

MSUB		Subtraktion im Fließkommaformat															
Ausführungszeit in $\mu$ s	6303	219 - 700	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	213 - 509		○	○				○	○					○		
<b>Funktion:</b> Der Operand OP2 wird von OP1 subtrahiert. Das Ergebnis wird in OP1 gespeichert. OP2 bleibt unverändert.																	
<b>Übergabe:</b> keine																	
<b>Ergebnis:</b> ERD verändert N, Z entsprechend dem Ergebnis der Routine																	



MSQR		Quadratwurzel im Fließkommaformat															
Ausführungszeit in µs	6303	71 - 8065	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	123 - 5100		○					○	○	○				○		
<b>Funktion:</b> Die Quadratwurzel von Operand OP1 wird berechnet und das Ergebnis in OP1 abgespeichert. OP2 bleibt unverändert. Die Rechengenauigkeit ist bei dieser Funktion auf vier Dezimalstellen beschränkt.																	
<b>Übergabe:</b> keine																	
<b>Ergebnis:</b> ERD verändert N, Z entsprechend dem Ergebnis der Routine																	

MSGN		Vorzeichenwechsel von Operand 1															
Ausführungszeit in µs	6303	85	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	126															
<b>Funktion:</b> Das Vorzeichen von OP1 wird invertiert. Die Operation entspricht einer Multiplikation mit -1. OP2 bleibt unverändert.																	
<b>Übergabe:</b> keine																	
<b>Ergebnis:</b> ERD verändert C 0 N, Z entsprechend dem Ergebnis der Routine																	



LAL1		Lade Operand OP1 mit Zahl (Absolut-Lang)															
Ausführungszeit in µs	6303	190 - 339	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	193 - 293						○									
<b>Funktion:</b> Die Binärzahl (Format: Absolut-Lang), auf die das Indexregister R zeigt, wird in IEEE-Format umgewandelt und im Operanden OP1 gespeichert. Sind in der Binärzahl mehr als 24 Bit verwendet, werden nur die höherwertigen 24 Bit in das IEEE-Format umgewandelt und in OP1 gespeichert; der Fehler 5 (CUT_LSB) wird gemeldet.																	
<b>Übergabe:</b> R		Quelladresse der Binärzahl						<b>Beispiel:</b>		Operand OP1 mit Binärzahl aus den 8-Bit Speicherstellen C 320 bis C 323 laden:							
<b>Ergebnis:</b> ERD N, Z		verändert entsprechend dem Ergebnis der Routine						LRK C 0320 Quelladresse der Binärzahl LAL1									

LAL2		Lade Operand OP2 mit Zahl (Absolut-Lang)															
Ausführungszeit in µs	6303	190 - 339	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	193 - 293						○									
<b>Funktion:</b> Die Binärzahl (Format: Absolut-Lang), auf die das Indexregister R zeigt, wird in IEEE-Format umgewandelt und im Operanden OP2 gespeichert. Sind in der Binärzahl mehr als 24 Bit verwendet, werden nur die höherwertigen 24 Bit in das IEEE-Format umgewandelt und in OP2 gespeichert; der Fehler 5 (CUT_LSB) wird gemeldet.																	
<b>Übergabe:</b> R		Quelladresse der Binärzahl															
<b>Ergebnis:</b> ERD N, Z		verändert entsprechend dem Ergebnis der Routine															

LAW1		Lade Operand OP1 mit Zahl (Absolut-Kurz)															
Ausführungszeit in µs	6303	83 - 250	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	125 - 241															
Funktion:	Die Binärzahl (Format: Absolut-Kurz) im Ergebnisregister ERD wird in IEEE-Format umgewandelt und im Operanden OP1 gespeichert.																
Übergabe:	ERD	Binärzahl (Format: Absolut-Kurz)															
Ergebnis:	ERD	verändert															
	N, Z	entsprechend dem Ergebnis der Routine															

LAW2		Lade Operand OP2 mit Zahl (Absolut-Kurz)															
Ausführungszeit in µs	6303	83 - 247	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	125 - 240															
Funktion:	Die Binärzahl (Format: Absolut-Kurz) im Ergebnisregister ERD wird in IEEE-Format umgewandelt und im Operanden OP2 gespeichert.																
Übergabe:	ERD	Binärzahl (Format: Absolut-Kurz)															
Ergebnis:	ERD	verändert															
	N, Z	entsprechend dem Ergebnis der Routine															



LIL1		Lade Operand OP1 mit Zahl (Integer-Lang)															
Ausführungszeit in µs	6303	197 - 381	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	201 - 315						○									
<b>Funktion:</b> Die Binärzahl (Format: Integer-Lang), auf die das Indexregister R zeigt, wird in IEEE-Format umgewandelt und im Operanden OP1 gespeichert. Sind in der Binärzahl mehr als 24 Bit verwendet, so werden nur die höherwertigen 24 Bit in das IEEE-Format umgewandelt und in OP1 gespeichert; der Fehler 5 (CUT_LSB) wird gemeldet.																	
<b>Übergabe:</b> R                      Quelladresse der Binärzahl																	
<b>Ergebnis:</b> ERD                      verändert N, Z                      entsprechend dem Ergebnis der Routine																	

LIL2		Lade Operand OP2 mit Zahl (Integer-Lang)															
Ausführungszeit in µs	6303	194 - 378	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	198 - 315						○									
<b>Funktion:</b> Die Binärzahl (Format: Integer-Lang), auf die das Indexregister R zeigt, wird in IEEE-Format umgewandelt und im Operanden OP2 gespeichert. Sind in der Binärzahl mehr als 24 Bit verwendet, so werden nur die höherwertigen 24 Bit in das IEEE-Format umgewandelt und in OP2 gespeichert; der Fehler 5 (CUT_LSB) wird gemeldet.																	
<b>Übergabe:</b> R                      Quelladresse der Binärzahl																	
<b>Ergebnis:</b> ERD                      verändert N, Z                      entsprechend dem Ergebnis der Routine																	

<b>LIW1</b>		<b>Lade Operand OP1 mit Zahl (Integer-Kurz)</b>															
<b>Ausführungszeit</b> in µs	<b>6303</b>	87 - 260	<b>Mögliche Fehlermeldungen</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	<b>6809</b>	128 - 249															
<b>Funktion:</b> Die Binärzahl (Format: Integer-Kurz) im Ergebnisregister ERD wird in IEEE-Format umgewandelt und im Operanden OP1 gespeichert. <b>Übergabe:</b> ERD Binärzahl <b>Ergebnis:</b> ERD verändert N, Z entsprechend dem Ergebnis der Routine																	
<b>LIW2</b>		<b>Lade Operand OP2 mit Zahl (Integer-Kurz)</b>															
<b>Ausführungszeit</b> in µs	<b>6303</b>	84 - 257	<b>Mögliche Fehlermeldungen</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	<b>6809</b>	126 - 247															
<b>Funktion:</b> Die Binärzahl (Format: Integer-Kurz) im Ergebnisregister ERD wird in IEEE-Format umgewandelt und im Operanden OP2 gespeichert. <b>Übergabe:</b> ERD Binärzahl <b>Ergebnis:</b> ERD verändert N, Z entsprechend dem Ergebnis der Routine																	

LF1		Lade Operand OP1 mit Zahl (IEEE-Format)																	
Ausführungszeit in µs	6303	88 - 125	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14		
	6809	129 - 146		○					○	○					○				
Funktion:		Die Fließkommazahl (IEEE-Format), auf die das Indexregister R zeigt, wird im Operanden OP1 gespeichert. Es wird überprüft, ob die geladene Zahl im erlaubten Zahlenbereich ist.																	
Übergabe:		R                      Quelladresse der Fließkommazahl																	
Ergebnis:		ERD                      verändert N, Z                      entsprechend dem Ergebnis der Routine																	

LF2		Lade Operand OP2 mit Zahl (IEEE-Format)																	
Ausführungszeit in µs	6303	88 - 125	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14		
	6809	127 - 144		○					○	○					○				
Funktion:		Die Fließkommazahl (IEEE-Format), auf die das Indexregister R zeigt, wird im Operanden OP2 gespeichert. Es wird überprüft, ob die geladene Zahl im erlaubten Zahlenbereich ist.																	
Übergabe:		R                      Quelladresse der Fließkommazahl																	
Ergebnis:		ERD                      verändert N, Z                      entsprechend dem Ergebnis der Routine																	



SIL		Operand OP1 im Format Integer-Lang abspeichern															
Ausführungszeit in µs	6303	172 - 424	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	190 - 347					○								○		
<b>Funktion:</b> Der Operand OP1 wird in eine Binärzahl (Format: Integer-Lang) umgewandelt und abgespeichert. Das Indexregister R enthält die Zieladresse, an der die Binärzahl abgespeichert wird. Die Operanden OP1 und OP2 werden dabei nicht verändert. Kann die Zahl mit 31 Bit nicht dargestellt werden, wird der Fehler 4 (CONV_OVERFLOW) gemeldet und der max. darstellbare negative bzw. positive Wert (-2147483647 oder +2147483647) abgespeichert.																	
<b>Übergabe:</b>		R	Zieladresse der Binärzahl														
<b>Ergebnis:</b>		ERD N, Z	verändert entsprechend dem Ergebnis der Routine														

SIW		Operand OP1 im Format Integer-Kurz abspeichern															
Ausführungszeit in µs	6303	158 - 380	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	183 - 321					○								○		
<b>Funktion:</b> Der Operand OP1 wird in eine Binärzahl (Format: Integer-Kurz) umgewandelt im Ergebnisregister ERD abgespeichert. Die Operanden OP1 und OP2 werden dabei nicht verändert. Kann die Zahl mit 15 Bit nicht dargestellt werden, wird der Fehler 4 (CONV_OVERFLOW) gemeldet und der max. darstellbare negative bzw. positive Wert (-32767 oder +32767) abgespeichert.																	
<b>Übergabe:</b>		keine															
<b>Ergebnis:</b>		ERD N, Z	Binärzahl entsprechend dem Inhalt von ERD verändert														



<b>SFM1</b>		<b>Operand OP1 im Zwischenspeicher 1 speichern</b>															
<b>Ausführungszeit</b> in $\mu$ s	<b>6303</b>	60	<b>Mögliche Fehlermeldungen</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	<b>6809</b>	116															
<b>Funktion:</b> Der Operand OP1 wird als Fließkommazahl (IEEE-Format) im Zwischenspeicher 1 (C1034 bis C1037) gespeichert.																	
<b>Übergabe:</b> keine																	
<b>Ergebnis:</b> ERD verändert																	

<b>SFM2</b>		<b>Operand OP1 im Zwischenspeicher 2 speichern</b>															
<b>Ausführungszeit</b> in $\mu$ s	<b>6303</b>	60	<b>Mögliche Fehlermeldungen</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	<b>6809</b>	116															
<b>Funktion:</b> Der Operand OP1 wird als Fließkommazahl (IEEE-Format) im Zwischenspeicher 2 (C1038 bis C1041) gespeichert.																	
<b>Übergabe:</b> keine																	
<b>Ergebnis:</b> ERD verändert																	

<b>SFM3</b>		<b>Operand OP1 im Zwischenspeicher 3 speichern</b>															
<b>Ausführungszeit</b> in $\mu$ s	<b>6303</b>	60	<b>Mögliche Fehlermeldungen</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	<b>6809</b>	116															
<b>Funktion:</b> Der Operand OP1 wird als Fließkommazahl (IEEE-Format) im Zwischenspeicher 3 (C1042 bis C1045) gespeichert.																	
<b>Übergabe:</b> keine																	
<b>Ergebnis:</b> ERD verändert																	

<b>RFM1</b>		<b>Operand OP2 aus Zwischenspeicher 1 laden</b>															
<b>Ausführungszeit</b> in µs	<b>6303</b>	56	<b>Mögliche Fehlermeldungen</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	<b>6809</b>	116															
<b>Funktion:</b>		Der Operand OP1 wird mit der Fließkommazahl (IEEE-Format) aus Zwischenspeicher 1 (C1034 bis C1037) geladen.															
<b>Übergabe:</b>		keine															
<b>Ergebnis:</b>		ERD            verändert															

<b>RFM2</b>		<b>Operand OP2 aus Zwischenspeicher 2 laden</b>															
<b>Ausführungszeit</b> in µs	<b>6303</b>	56	<b>Mögliche Fehlermeldungen</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	<b>6809</b>	116															
<b>Funktion:</b>		Der Operand OP1 wird mit der Fließkommazahl (IEEE-Format) aus Zwischenspeicher 2 (C1038 bis C1041) geladen.															
<b>Übergabe:</b>		keine															
<b>Ergebnis:</b>		ERD            verändert															

<b>RFM3</b>		<b>Operand OP2 aus Zwischenspeicher 3 laden</b>															
<b>Ausführungszeit</b> in µs	<b>6303</b>	56	<b>Mögliche Fehlermeldungen</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	<b>6809</b>	116															
<b>Funktion:</b>		Der Operand OP1 wird mit der Fließkommazahl (IEEE-Format) aus Zwischenspeicher 3 (C1042 bis C1045) geladen.															
<b>Übergabe:</b>		keine															
<b>Ergebnis:</b>		ERD            verändert															



FM2B		2 Byte X 2 Byte Multiplikation																
Ausführungszeit in µs	6303	115 - 191	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
	6809	125 - 175																
Funktion:		Zwei Binärzahlen (Format: Integer-Kurz) werden miteinander multipliziert. Das Ergebnis ist eine Zahl im Integer-Lang Format.																
Übergabe:		R                      Quelladresse des Multiplikanden ERD                      Multiplikator C1048&                      Zieladresse des Ergebnisses																
Ergebnis:		ERD                      verändert C, N, Z                      ungültig R                              unverändert																

FM3B		3 Byte X 2 Byte Multiplikation																
Ausführungszeit in µs	6303	156 - 270	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
	6809	150 - 225																
Funktion:		Eine 3 Byte Integerzahl wird mit einer Zahl im Integer-Kurz Format multipliziert. Das Ergebnis ist eine 5 Byte Integerzahl.																
Übergabe:		R                      Quelladresse des Multiplikanden (3 Byte Integerzahl) ERD                      Multiplikator (Integer-Kurz) C1048&                      Zieladresse des Ergebnisses (5 Byte Integerzahl)																
Ergebnis:		ERD                      verändert C, N, Z                      ungültig R                              unverändert																



CAF		ASCII-String in IEEE-Format umwandeln																	
Ausführungszeit in µs	6303	280 - 2140	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14		
	6809	251 - 1460																	
<b>Funktion:</b> Der ASCII-String, auf den das Indexregister R zeigt, wird in das interne IEEE-Format umgewandelt und in OP1 gespeichert. OP2 wird nicht verändert.																			
<b>Übergabe:</b> R Quelladresse des ASCII-Strings																			
<b>Ergebnis:</b> ERD verändert N, Z ungültig																			
<b>Syntaxregeln für den ASCII-String:5</b>																			
<div>1. Für das <b>Vorzeichen</b> sind &lt;+&gt;, &lt;-&gt; und &lt;Leerzeichen&gt; zulässig. Das Vorzeichen kann auch entfallen.</div> <div>2. Die <b>Mantisse</b> kann mit vorlaufenden Nullen oder Leerzeichen beginnen, kann einen Dezimalpunkt und bis zu sieben signifikante Ziffern enthalten. Zur Trennung von Mantisse und Exponent wird ein &lt;E&gt; oder ein Leerzeichen verwendet.</div> <div>3. Der <b>Exponent</b> ist zweistellig plus Vorzeichen und beginnt mit einem &lt;E&gt; nach der letzten Mantissenziffer bzw. nach dem Trennzeichen (Leerzeichen).</div> <div>4. <b>Zulässige Zeichen</b> sind &lt;0&gt; bis &lt;9&gt;, &lt;Leerzeichen&gt;, &lt;-&gt;, &lt;+&gt;, &lt;Dezimalpunkt&gt; und &lt;E&gt;. Ungültige Zeichen im ASCII-String führen zum Abbruch der Routine und der Fehler 9 (INVAL_CHAR) wird gemeldet. Der Wert, der im Operanden OP1 nach dem Abbruch steht, ist nicht gültig.</div> <div>5. Nach einem &lt;E&gt; werden automatisch max. 3 Zeichen eingelesen, ansonsten muß der String <b>mit &lt;CR&gt; (\$OD) oder &lt;0&gt; (\$00) abgeschlossen</b> sein.</div>																			
<b>Beispiele für gültige ASCII-Strings:</b>																			
<div>" 12.45&lt;CR&gt;"                      "-12.45&lt;CR&gt;"                      "+.23&lt;CR&gt;"                      ".1234567&lt;CR&gt;"                      "0.0022&lt;CR&gt;"</div> <div>"-1.234567 E 09"                      "+.34E-8&lt;CR&gt;"                      "0.022E1&lt;CR&gt;"                      "-12.45 E+01"                      ".3E+14"</div>																			

**CFA****OP1 in ASCII ohne Vornullen wandeln****Ausführungszeit**  
in  $\mu$ s**6303**

352 - 7310

**6809**

352 - 4440

**Mögliche****Fehlermeldungen**

1	2	3	4	5	6	7	8	9	10	11	12	13	14
<input type="radio"/>			<input type="radio"/>		<input type="radio"/>	<input type="radio"/>					<input type="radio"/>		

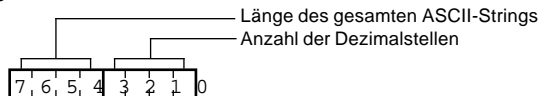
**Funktion:**

Der Inhalt von OP1 wird in einen ASCII-String umgewandelt und ab der im Indexregister R angegebenen Adresse abgespeichert. Die Operanden OP1 und OP2 werden nicht verändert. Die Anzahl der signifikanten Ziffern ist auf max. 7 beschränkt. Der ASCII-String kann daher max. 9 Zeichen (inkl. Vorzeichen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden durch Leerzeichen ersetzt. Kann OP1 nicht im gewünschten ASCII-Format dargestellt werden, wird der String mit ">" aufgefüllt.

**Übergabe:**

R Zieladresse des ASCII-Strings

ERA Format des ASCII-Strings:

**Ergebnis:**

ERA verändert  
 ERB Länge des ASCII-Strings  
 N, Z 0

**Beispiel:**

OP1 soll in einen ASCII-String mit max. 5 signifikanten Stellen (inkl. 2 Dezimalstellen) umgewandelt werden. Der ASCII-String soll ab der 8-Bit Speicherstelle C0250 abgespeichert werden.

```
      :  
      :  
LWK   C 0250      Zieladresse für den ASCII-String in das Indexregister R laden  
      |  
      |  
LAD   # $72      Länge des ASCII-Strings (5 signifikante Stellen + Vorzeichen + Dezimalpunkt = 7)  
CFA   |          Anzahl der Dezimalstellen (2)  
      |          Format des ASCII-Strings in ERA übergeben  
      :  
      :
```

OP1 = 32.123	=>	" 32.12"
OP1 = -0.1	=>	" - 0.10"
OP1 = 4.87	=>	" 4.87"
OP1 = 2300.25	=>	" >>>. >>"
OP1 = -1000.25	=>	" -<<<. <<"

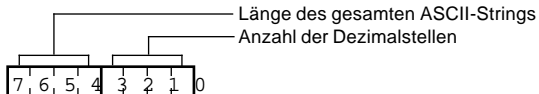
**CFA0****OP1 in ASCII mit Vornullen wandeln**

Ausführungszeit in $\mu$ s	6303	310 - 7190	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	310 - 4320		<input type="radio"/>			<input type="radio"/>		<input type="radio"/>	<input type="radio"/>					<input type="radio"/>		

**Funktion:** Der Inhalt von OP1 wird in einen ASCII-String umgewandelt und ab der im Indexregister R angegebenen Adresse abgespeichert. Die Operanden OP1 und OP2 werden nicht verändert. Die Anzahl der signifikanten Ziffern ist auf max. 7 beschränkt. Der ASCII-String kann daher max. 9 Zeichen (inkl. Vorzeichen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden nicht unterdrückt. Kann OP1 nicht im gewünschten ASCII-Format dargestellt werden, wird der String mit ">" aufgefüllt.

**Übergabe:** R            Zieladresse des ASCII-Strings

ERA            Format des ASCII-Strings:



**Ergebnis:** ERA            verändert  
 ERB            Länge des ASCII-Strings  
 N, Z            0

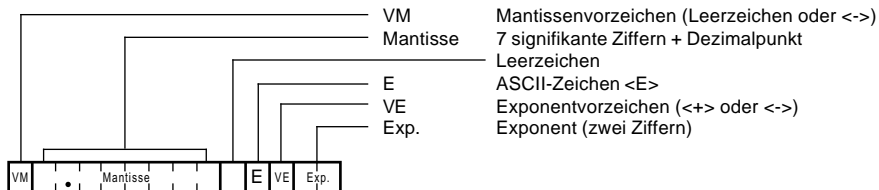
OP1 soll in einen ASCII-String mit max. 4 signifikanten Stellen (inkl. 1 Dezimalstelle) umgewandelt werden. Der ASCII-String soll ab der 8-Bit Speicherstelle C0100 abgespeichert werden.

```
OP1 = 32.123      =>    " 032.1"
OP1 = -0.1        =>    "-000.1"
OP1 = 4.87        =>    " 004.8"
OP1 = 2300.25     =>    " >>>.>"
OP1 = -1000.25    =>    "-<<<.<"
```

**CFEA****OP1 in ASCII mit Exponentdarstellung wandeln**

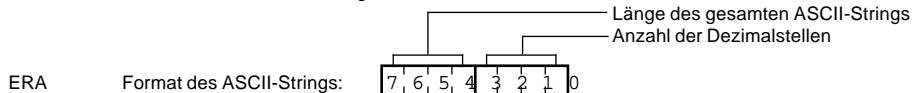
<b>Ausführungszeit</b> in $\mu$ s	<b>6303</b>	570 - 7140	<b>Mögliche</b> <b>Fehlermeldungen</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	<b>6809</b>	267 - 4140		○					○	○					○		

**Funktion:** Der Inhalt von OP1 wird in einen ASCII-String mit Exponentdarstellung umgewandelt und ab der im Indexregister R angegebenen Adresse abgespeichert. Die Operanden OP1 und OP2 werden nicht verändert. Der String hat immer das selbe Format (Länge: 14 Zeichen).



Kann OP1 nicht im gewünschten ASCII-Format dargestellt werden, wird der String mit ">" aufgefüllt.

**Übergabe:** R Zieladresse des ASCII-Strings



**Ergebnis:** ERA verändert  
ERB Länge des ASCII-Strings  
Z entsprechend dem Ergebnis der Routine  
N ungültig



**Beispiel:** OP1 soll in einen ASCII-String mit Exponentendarstellung umgewandelt und ab C0500 abgespeichert werden.

```
      :  
      :  
LRK   C 0500      Zieladresse für den ASCII-String  
CFEA  
      :  
      :
```

OP1 = 32111	=>	" 3.211100 E+04 "
OP1 = -487	=>	" -4.870000 E+02 "
OP1 = 0.0456	=>	" 4.560000 E-02 "
OP1 = 0	=>	" 0.000000 E+00 "

**CIA****Binär in ASCII ohne Vornullen wandeln****Ausführungszeit**  
in  $\mu$ s**6303**

380 - 2020

**6809**

357 - 1370

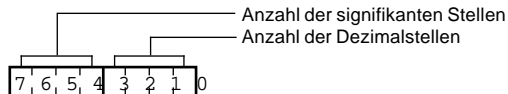
**Mögliche****Fehlermeldungen**

1	2	3	4	5	6	7	8	9	10	11	12	13	14
			○										

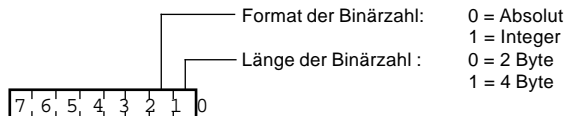
**Funktion:** Eine Binärzahl wird in einen ASCII-String umgewandelt und ab der im Indexregister R angegebenen Adresse abgespeichert. Die Operanden OP1 und OP2 werden nicht verändert.  
Die Anzahl der signifikanten Ziffern ist auf max. 9 beschränkt. Der ASCII-String kann daher max. 11 Zeichen (inkl. Vorzeichen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden durch Leerzeichen ersetzt. Kann die Binärzahl im gewünschten ASCII-Format nicht dargestellt werden, wird der String mit ">" bzw. "<" aufgefüllt.

**Übergabe:** C1046&      Quelladresse der Binärzahl  
R                    Zieladresse des ASCII-Strings

ERA      Format des ASCII-Strings:



ERB      Format der Binärzahl:



**Ergebnis:** ERA      verändert  
ERB      Länge des gesamten ASCII-Strings  
N, Z      0

**Beispiel:**

Die Zahl in den 8 Bit Speicherstellen C0100 bis C0103 (Format Integer-Lang) soll in einen ASCII-String mit max. 6 signifikanten Stellen (davon 2 Dezimalstellen) umgewandelt werden. Der String soll ab C3000 abgespeichert werden.

```

:
:
LRK      C 0100      Quelladresse der Binärzahl
=LR      C 1046      Abspeichern der Quelladresse in der 8 Bit Speicherstelle C1046
LRK      C 3000      Zieladresse für den ASCII-String

LAD      # $62        Anzahl der signifikanten Stellen
                        Anzahl der Dezimalstellen
                        Format des ASCII-Strings in Ergebnisregister ERA übergeben

LB       # %00000011  Format der Binärzahl: 1 => Integer
                        Länge der Binärzahl: 1 => 4 Byte
                        Format der Binärzahl in Ergebnisregister ERB übergeben

CIA
:
:
```

Integer-Lang = 56499	=>	" 564.99 " "
Integer-Lang = -23	=>	" - 0.23 "
Integer-Lang = 1000000	=>	" >>> . >> "

**CIA0****Binär in ASCII mit Vornullen wandeln****Ausführungszeit**  
in  $\mu$ s**6303**

310 - 1960

**6809**

312 - 1320

**Mögliche**

1 2 3 4 5 6 7 8 9 10 11 12 13 14

**Fehlermeldungen**

○

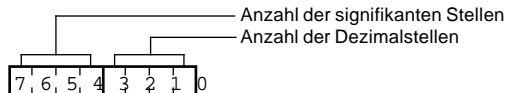
**Funktion:**

Eine Binärzahl wird in einen ASCII-String umgewandelt und ab der im Indexregister R angegebenen Adresse abgespeichert. Die Operanden OP1 und OP2 werden nicht verändert.  
 Die Anzahl der signifikanten Ziffern ist auf max. 9 beschränkt. Der ASCII-String kann daher max. 11 Zeichen (inkl. Vorzeichen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden nicht unterdrückt. Kann die Binärzahl im gewünschten ASCII-Format nicht dargestellt werden, wird der String mit ">" bzw. "<" aufgefüllt.

**Übergabe:**C1046&  
RQuelladresse der Binärzahl  
Zieladresse des ASCII-Strings

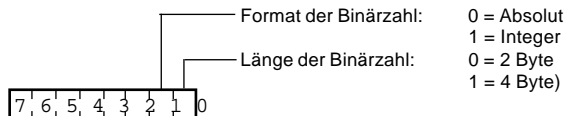
ERA

Format des ASCII-Strings:



ERB

Format der Binärzahl:

**Ergebnis:**ERA  
ERB  
N, Zverändert  
Länge des gesamten ASCII-Strings  
0

**Beispiel:**

Die Zahl in den 8 Bit Speicherstellen C0100 bis C0103 (Format Absolut-Lang) soll in einen ASCII-String mit max. 8 signifikanten Stellen (davon 3 Dezimalstellen) umgewandelt werden. Der String soll ab C0200 abgespeichert werden.

```

:
:
LRK   C 0100      Quelladresse der Binärzahl
=R    C 1046      Abspeichern der Quelladresse in der 8 Bit Speicherstelle C1046
LRK   C 0200      Zieladresse für den ASCII-String

LAD   # $83       Anzahl der signifikanten Stellen
                        Anzahl der Dezimalstellen
                        Format des ASCII-Strings in Ergebnisregister ERA übergeben

LB    # %00000001  Format der Binärzahl: 0 => Absolut
                        Länge der Binärzahl: 1 => 4 Byte
                        Format der Binärzahl in Ergebnisregister ERB übergeben
CIA
:
:
```

Absolut-Lang = 56499	=>	" 00056.499"
Absolut-Lang = -23	=>	" -00000.023"
Absolut-Lang = 1000000	=>	" 01000.000"
Absolut-Lang = 100000000	=>	" >>>>. >>>"



# CBIN

## BCD in Binär wandeln

Ausführungszeit in µs	6303	112 - 223	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	166 258															

**Funktion:** Eine 3 Byte BCD-Zahl wird in eine 3 Byte Binärzahl umgewandelt. Quell- und Zieladresse müssen verschieden sein.

**Übergabe:** ERD Quelladresse der BCD-Zahl  
R Zieladresse der Binärzahl

**Ergebnis:** ERD niederwertigen zwei Byte der Binärzahl  
N, Z entsprechend dem Inhalt von ERD

**Beispiel:** BCD-Zahl = \$ 

0	0	0	4	5
---	---	---	---	---

 0 => Binärzahl = 450  
BCD-Zahl = \$ 

0	0	1	9	5
---	---	---	---	---

 6 => Binärzahl = 1956  
BCD-Zahl = \$ 

9	9	9	9	9
---	---	---	---	---

 9 => Binärzahl = 999999

Die BCD-Zahl, die in den Speicherstellen C0200 bis C0202 gespeichert ist, soll in eine Binärzahl umgewandelt werden. Das Ergebnis soll ab der Speicherstelle C3000 abgespeichert werden.

```
:  
LRK    C 0200      Quelladresse der BCD-Zahl  
DXR                    Quelladresse nach ERD laden  
LRK    C 3000      Zieladresse für Binärzahl  
CBIN  
:
```

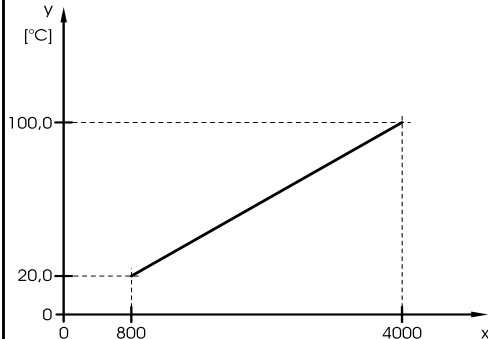
# Umwandlung: Binär <=> Physikalisch (Normierung)

In SPS-Programmen liegen Zahlen meist nicht in ihrer physikalischen Größe und Einheit vor, sondern in einem binären Wert, der einer bestimmten physikalischen Größe entspricht. Zur Darstellung von physikalischen Werten (z.B. an Bedienterminals) muß die vorliegende Binärzahl wieder in die physikalische Größe umgerechnet werden.

**Beispiel:** Eine Temperatur wird im Bereich 20,0 °C bis 100,0 °C gemessen, Der Temperaturfühler liefert ein Analogsignal von 4 mA bis 20 mA, das von einem A/D-Wandler in einen SPS-internen Zahlenwert von 800 bis 4000 umgewandelt wird. Daraus folgt:

$$\begin{array}{lll} 20,0 \text{ °C (4 mA)} & \Rightarrow & 800 \\ 100,0 \text{ °C (20 mA)} & \Rightarrow & 4000 \end{array}$$

Die Umrechnung erfolgt nach der Geradengleichung:  $y = kx + d$



Die Faktoren **k** und **d** lassen sich aus den zwei Geradenstützpunkten (800/200) und (4000/1000) errechnen:

$$\text{I.} \quad 200 = 800k + d \quad \Rightarrow \quad d = 200 - 800k$$

$$\text{II.} \quad 1000 = 4000k + d$$

---


$$\text{II.} \quad 1000 = 4000k + 200 - 800k$$

$$800 = 3200k$$

$$k = \frac{800}{3200} = 0.25$$

$$\Rightarrow \quad d = 200 - 800k = 0$$



Für die **Umwandlung von Binärzahlen in physikalische Größen** stehen folgende Aufrufe zur Verfügung:

- CBPP** Berechnung der Faktoren **k** und **d** aus zwei Geradenstützpunkten bzw. Umrechnung von **k** und **d** aus dem Integer-Kurz Format in das IEEE-Format.
- CBPQ** Umrechnung nach der Formel  $y = kx + d$ . Die Faktoren **k** und **d** müssen im IEEE-Format vorliegen (z.B. berechnet mit der Routine CBPP).
- CBP** Umrechnung nach der Formel  $y = kx + d$ . Es können entweder die Faktoren **k** und **d** oder zwei Geradenstützpunkte ( $x_1/y_1$ ) und ( $x_2/y_2$ ) der Berechnung zu Grunde liegen. Das Zahlenformat (Integer-Kurz oder IEEE-Format) der angegebenen Faktoren kann außerdem vom Anwender gewählt werden.

Oft sollen über Bedienterminals Zahlenwerte in ihrer physikalischen Größe eingegeben werden. Diese müssen dann nach der selben Geradengleichung in den SPS-internen Binärwert umgerechnet werden. Aus der Umkehrung der Geradengleichung ergibt sich:

$$y = kx + d \quad \Rightarrow \quad x = \frac{y - d}{k}$$

Für die **Umwandlung physikalischer Eingabewerte in Binärwerte** stehen folgende Routinen zur Verfügung:

- CBPP** Berechnung der Faktoren **k** und **d** aus zwei Geradenstützpunkten bzw. Umrechnung von **k** und **d** aus dem Integer-Kurz Format in das IEEE-Format.
- CPBQ** Umrechnung von x nach der o.a. Formel. Die Faktoren **k** und **d** müssen im IEEE-Format vorliegen (z.B. berechnet mit der Routine CBPP).
- CPB** Umrechnung von x nach der o.a. Formel. Es können entweder die Faktoren **k** und **d** oder zwei Geradenstützpunkte ( $x_1/y_1$ ) und ( $x_2/y_2$ ) der Berechnung zu Grunde liegen. Das Zahlenformat (Integer-Kurz oder IEEE-Format) der angegebenen Faktoren kann außerdem vom Anwender gewählt werden.

**CBPP****Berechnung der Faktoren  $k$  und  $d$  aus zwei Geradenstützpunkten**

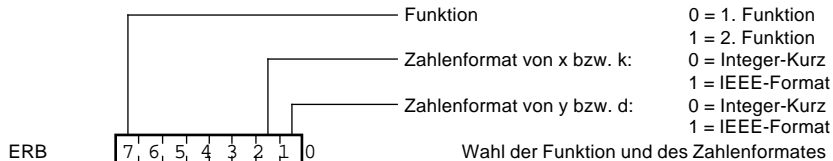
Ausführungszeit in $\mu$ s	6303	2500 - 6700	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	1200 - 4200		<input type="radio"/>					<input type="radio"/>	<input type="radio"/>					<input type="radio"/>		

**Funktion:** Diese Routine hat zwei Funktionen:

1. Berechnung der Faktoren **k** und **d** (im IEEE-Format) aus zwei Geradenstützpunkten  $(x_1/y_1)$  und  $(x_2/y_2)$ . Die Geradenstützpunkte können im Integer-Kurz- oder IEEE-Format vorliegen.
2. Die Faktoren **k** und **d** werden im Integer-Kurz- oder IEEE-Format an die Routine übergeben. Die Faktoren, die im Integer-Kurz Format übergeben wurden, werden in das IEEE-Format umgewandelt.

Die Zahlenformate können beliebig gewählt werden. Siehe dazu das Diagramm auf der folgenden Seite.

**Übergabe:** C1048& Zieladresse für die Faktoren **k** und **d** (IEEE-Format, => 8 Byte)



R Quelladresse der Parameter (x/y oder k/d)

**Ergebnis:** R, ERD verändert  
N, Z ungültig

Abhängig von der gewählten Funktion und des Zahlenformates der Parameter wird unterschiedlich viel Speicherplatz für die Übergabe-Parameter benötigt:

Speicher	Inhalt von Ergebnisregister ERB							
	0xxxx00	0xxxx01	0xxxx10	0xxxx11	1xxxx00	1xxxx01	1xxxx10	1xxxx11
Cxxx	$x_1$	$x_1$	$x_1$	$x_1$	k	k	k	k
Cxxx + 1	Integer-Kurz	Integer-Kurz	IEEE	IEEE	Integer-Kurz	Integer-Kurz	IEEE	IEEE
Cxxx + 2	$y_1$	$y_1$			d	d		
Cxxx + 3	Integer-Kurz	IEEE			Integer-Kurz	IEEE		
Cxxx + 4	$x_2$		$y_1$	$y_1$			d	d
Cxxx + 5	Integer-Kurz		Integer-Kurz	IEEE			Integer-Kurz	IEEE
Cxxx + 6	$y_2$	$y_2$	$x_2$					
Cxxx + 7	Integer-Kurz	Integer-Kurz	IEEE					
Cxxx + 8		$y_2$		$x_2$				
Cxxx + 9		IEEE		IEEE				
Cxxx + 10			$y_2$					
Cxxx + 11			Integer-Kurz					
Cxxx + 12				$y_2$				
Cxxx + 13				IEEE				
Cxxx + 14								
Cxxx + 15								



**CPBQ****Wandle Physikalisch => Binär, schnell**

Ausführungszeit in µs	6303	780 - 1500	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	580 - 950		<input type="radio"/>					<input type="radio"/>	<input type="radio"/>					<input type="radio"/>		

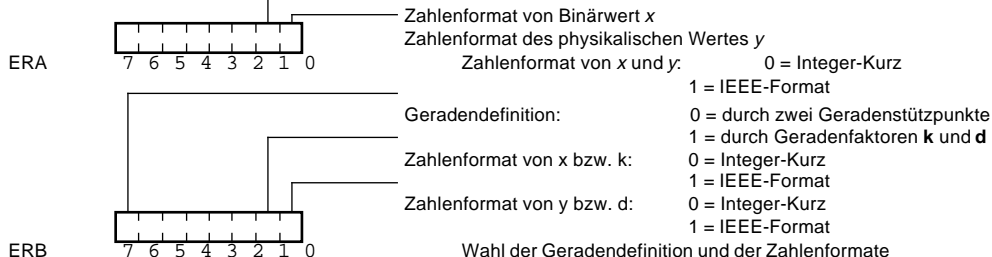
- Funktion:** Umrechnung eines physikalisch normierten Eingabewertes nach der Formel  $x = \frac{y - d}{k}$ .  
Die Faktoren **k** und **d** liegen bereits als Fließkommazahlen (IEEE-Format, z.B. durch die Routine **CBPP** ermittelt) vor.
- Übergabe:** ERD      Physikalischer Wert **y** (Integer-Kurz)  
R      Quelladresse der Geradenfaktoren **k** und **d** (IEEE-Format)
- Ergebnis:** ERD      Binärwert **x** (Integer-Kurz)  
OP1      Binärwert **x** (IEEE-Format)  
R      verändert  
N, Z      entsprechend dem Inhalt von ERD

**CBP****Wandle Binär => Physikalisch**

<b>Ausführungszeit</b> in $\mu$ s	<b>6303</b>	3400 - 8300	<b>Mögliche</b> <b>Fehlermeldungen</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	<b>6809</b>	ca. 4000		<input type="radio"/>					<input type="radio"/>	<input type="radio"/>					<input type="radio"/>		

**Funktion:** Umrechnung einer Binärzahl (Integer-Kurz) in einen physikalischen Wert nach der Formel  $y = kx + d$ . Die Umwandlungsgerade wird wahlweise durch zwei Geradenstützpunkte ( $x_1/y_1$ ) und ( $x_2/y_2$ ) oder durch die Faktoren **k** und **d** definiert. Da die Berechnung der Faktoren **k** und **d** aus den Geradenstützpunkten relativ lange dauert, sollte sie nicht bei jedem Aufruf erfolgen. Die Faktoren **k** und **d** sollten deshalb mit der Routine CBPP in einem Initialisierungs-Programmteil errechnet und zwischengespeichert werden. Bei Aufruf der Routine **CBP** werden diese bereits errechneten Faktoren zur Verfügung gestellt.

**Übergabe:** C1046& Quelladresse der Binärzahl x  
C1048& Zieladresse für den physikalischen Wert y  
R Quelladresse der Geradenstützpunkte ( $x_1/y_1$ ) und ( $x_2/y_2$ ) oder Geradenfaktoren **k** und **d**



**Ergebnis:** ERD      Physikalischer Wert  $y$  (Integer-Kurz)  
 OP1      Physikalischer Wert  $y$  (IEEE-Format)  
 R      verändert  
 N, Z      entsprechend dem Inhalt von ERD

**Beispiel:** Die Geradenfaktoren **k** und **d** wurden bereits mit der Routine **CBPP** berechnet und sind in den 8 Bit Speicherstellen C0300 bis C0307 abgespeichert. Die Binärzahl  $x$  (Format: Integer-Kurz) aus den Speicherstellen C0100& soll normiert und das Ergebnis  $y$  im IEEE-Format ab der Speicherstelle C0120 abgelegt werden.

```

:
:
LRK   C 0100      Laden der Quelladresse der Binärzahl x in das Indexregister R
=R    C 1046      Abspeichern der Quelladresse in C1046&
LRK   C 0120      Laden der Zieladresse für den physikalischen Wert y in das Indexregister R
=R    C 1048      Abspeichern der Quelladresse in C1048&

LAD   # %00000001  Format von x (0 = Integer-Kurz)
                        Format von y (1 = IEEE)
                        Zahlenformat von x und y

LB    # %10000011  Gerade ist durch die Faktoren k und d definiert
                        Format von d (1 = IEEE)
                        Format von k (1 = IEEE)
                        Geradendefinition und Zahlenformate

LRK   C 0300      Quelladresse der Geradenfaktoren k und d
CBP                                Aufruf der Umwandlungs-Routine
:
:
```

**CPB****Wandle Physikalisch => Binär**

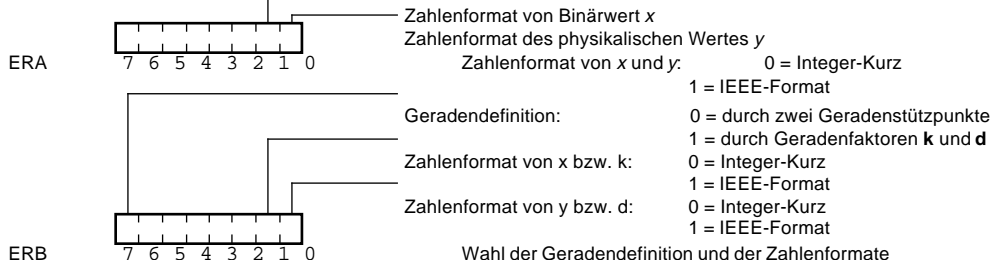
<b>Ausführungszeit</b> in $\mu$ s	<b>6303</b>	3400 - 8300	<b>Mögliche</b> <b>Fehlermeldungen</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	<b>6809</b>	ca. 4000		<input type="radio"/>					<input type="radio"/>	<input type="radio"/>					<input type="radio"/>		

**Funktion:** Umrechnung eines physikalisch normierten Eingabewertes nach der Formel  $x \frac{y - d}{k}$ .

Die Umwandlungsgerade wird wahlweise durch zwei Geradenstützpunkte ( $x_1/y_1$ ) und ( $x_2/y_2$ ) oder durch die Faktoren **k** und **d** definiert.

Da die Berechnung der Faktoren **k** und **d** aus den Geradenstützpunkten relativ lange dauert, sollte sie nicht bei jedem Aufruf erfolgen. Die Faktoren **k** und **d** sollten deshalb mit der Routine CBPP in einem Initialisierungs-Programmteil errechnet und zwischengespeichert werden. Bei Aufruf der Routine **CPB** werden diese bereits errechneten Faktoren zur Verfügung gestellt.

**Übergabe:** C1046& Quelladresse des physikalischen Wertes y  
 C1048& Zieladresse für die Binärzahl x  
 R Quelladresse der Geradenstützpunkte ( $x_1/y_1$ ) und ( $x_2/y_2$ ) oder Geradenfaktoren **k** und **d**



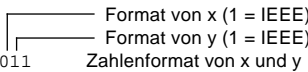
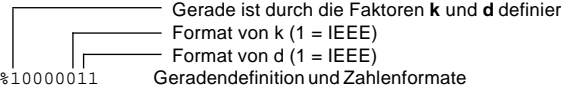


**Ergebnis:** ERD Binärwert x (Integer-Kurz), wenn für x das Format Integer-Kurz gewählt wurde, ansonsten ist ERD undefiniert.  
 OP1 Binärwert x (IEEE-Format)  
 R verändert  
 N, Z entsprechend dem Inhalt von ERD

**Beispiel:** Die Geradenfaktoren **k** und **d** wurden bereits mit der Routine **CBPP** berechnet und sind in den 8 Bit Speicherstellen C0300 bis C0307 abgespeichert. Aus dem physikalische Wert y (IEEE-Format) aus den Speicherstellen C0100 bis C0103 soll der Binärwert x errechnet und ab der Speicherstelle C0120 im selben Format abgelegt werden.

```

:
:
LRK   C 0100      Laden der Quelladresse physikalischen Wertes y in das Indexregister R
=R    C 1046      Abspeichern der Quelladresse in C1046&
LRK   C 0120      Laden der Zieladresse für den Binärwert x in das Indexregister R
=R    C 1048      Abspeichern der Quelladresse in C1048&

LAD   # %00000011  
LB    # %10000011  
LRK   C 0300      Quelladresse der Geradenfaktoren k und d
CPB                                     Aufruf der Umwandlungs-Routine
:
:
```



**Beispiel:**

Über eine Tastatur werden Sollpositionen in Zoll-Einheiten eingegeben. Intern werden alle Ist- und Sollpositionen in mm mit der Auflösung 0.01 gespeichert. Um bei der Umrechnung von Zoll in mm möglichst wenig Genauigkeit zu verlieren, wird die Zoll-Zahl mit vier Nachkommastellen eingegeben.

```

:
:
LRK      C 0100      Quelladresse der Zoll-Zahl
=R       C 1046
LAD      # 004       Exponent der Zoll-Zahl (Genauigkeit = 0.0001; => 4 Nachkommastellen)
LB       # 002       Exponent der mm-Zahl (Genauigkeit = 0.01; => 2 Nachkommastellen)
LRK      C 0200      Zieladresse der für die mm-Zahl
CIM
:
:

```

Binärzahl in C0100 bis C0103	Zoll-Zahl	mm-Zahl	Binärzahl in C0200 bis C0203
3460	0.3460"	8.79mm	879
234500	23.4500"	595.62mm	59562
3937	0.3937"	10.00mm	1000
10000	1.0000"	25.40mm	2540



**Beispiel:**

In einem Positioniersystem wird die Istposition metrisch erfaßt, soll aber in Zoll auf einem Bedienterminal angezeigt werden. Intern werden alle Ist- und Sollpositionen in mm mit Auflösung 0.01 gespeichert. Die Zoll-Zahl wird mit vier Nachkommastellen (=> Genauigkeit = 0.0001 Zoll) angezeigt.

```

:
:
LRK      C 0100      Quelladresse der mm-Zahl
=R       C 1046
LAD      # 002       Exponent der mm-Zahl (Genauigkeit = 0.01; => 2 Nachkommastellen)
LB       # 004       Exponent der Zoll-Zahl (Genauigkeit = 0.0001; => 4 Nachkommastellen)
LRK      C 0200      Zieladresse der für die Zoll-Zahl
CMI
:
:

```

Binärzahl in C0100 bis C0103	mm-Zahl	Zoll-Zahl	Binärzahl in C0200 bis C0203
346	3.46 mm	0.1362 "	1362
20045	200.45 mm	7.8927 "	78927
2540	25.40 mm	1.0001 "	10001
10000	100.00 mm	393.75 "	39375

FCOP		Speicherbereich kopieren																
Ausführungszeit in µs	6303	siehe Tabelle	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
	6809	siehe Tabelle																
<b>Funktion:</b> Ein Speicherbereich der Länge L wird von einer Quelladresse an eine Zieladresse kopiert. Die Routine erkennt aus den Angaben von Quell-, Zieladresse und Länge des Blockes, ob sich Quelle und Ziel überlappen. Dementsprechend wird in der richtigen Reihenfolge (auf- bzw. absteigend) kopiert. Die Quelle wird beim kopieren nicht verändert, außer sie wird vom Ziel überlappt. Folgende Kopiervorgänge sind möglich:																		
<div><div><div><div>Quelle</div><div></div><div>Ziel</div><div></div></div><div>vorwärts</div></div><div><div><div>Ziel</div><div></div><div>Quelle</div><div></div></div><div>rückwärts</div></div><div><div><div></div><div>Quelle</div><div>Ziel</div><div></div></div><div>vorwärts überlappend</div></div><div><div><div></div><div>Ziel</div><div></div><div>Quelle</div></div><div>rückwärts überlappend</div></div></div>																		
<div><div><div><b>Beispiel:</b> Die Tabelle TEXT soll in den Speicher ab C0100 kopiert werden.</div><div><div>: LRK    C   0100 =R    C   1048 LAD    #   000 SPU    TEXT</div><div>FCOP</div></div><div><div>Zieladresse</div><div>ERD enthält die Länge der Tabelle und R die Startadresse der Tabelle Kopieren der Tabelle</div></div></div></div>																		
<b>Übergabe:</b> C1048&ERD R		Zieladresse des Speicherbereiches Länge des Speicherbereiches in Byte Quelladresse des Speicherbereiches		<b>Ausführungszeiten in µs</b>														
				<b>6303</b>							<b>6809</b>							
<b>Ergebnis:</b> ERD N, Z, C		verändert ungültig		Zieladr. > Quelladr.	$82 + (\frac{L}{256} + 1) * 55 + L * 23$							121 + L * 10,5						
				Zieladr. < Quelladr	$58 + (\frac{L}{256} + 1) * 54 + L * 25$							112 + L * 10,5						

# FSMB

## Speicherbereich mit 1 Byte-Wert initialisieren

Ausführungszeit in $\mu$ s	6303	$48 + L \cdot 12$	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	$94 + L \cdot 7,5$															

**Funktion:** Alle Speicherstellen des angegebenen Bereiches (Anfangsadresse, Länge des Bereiches in Byte) werden mit einem 1 Byte-Wert geladen.

**Übergabe:** C1052 1 Byte-Wert  
ERD Länge des Speicherbereiches in Byte  
R Anfangsadresse des Speicherbereiches

**Ergebnis:** ERD verändert  
N, Z, C ungültig

**Beispiel:** Die Speicherstellen C3000 bis C3299 sollen mit dem Wert 255 (\$FF) geladen werden:

```

:
:
LAD # 255          1 Byte-Wert in ERA laden
=   C 1052        1 Byte-Wert in C1052 an die Routine übergeben
LD   # 00300      Länge des Speicherbereiches in Byte (300)
LRK  C 3000       Anfangsadresse des Speicherbereiches
FSMB
:
:

```





FCLR		Speicherbereich löschen																	
Ausführungszeit in µs	6303	48 + L*12	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14		
	6809	94 + L*7,5																	
Funktion:	Alle Speicherstellen des angegebenen Bereiches (Anfangsadresse, Länge des Bereiches in Byte) werden gelöscht, d.h. mit dem Wert Null überschrieben.																		
Übergabe:	ERD	Länge des Speicherbereiches in Byte																	
	R	Anfangsadresse des Speicherbereiches																	
Ergebnis:	ERD	verändert																	
	N, Z, C	ungültig																	
Beispiel:	Der Speicherbereich C0100 bis C0199 soll gelöscht werden																		
	:																		
	:																		
	LD	#	00100	Länge des Speicherbereiches															
	LRK	C	0100	Anfangsadresse des Speicherbereiches															
	FCLR																		
	:																		
	:																		

**MCMP****OP1 und OP2 vergleichen**

<b>Ausführungszeit</b> in $\mu$ s	<b>6303</b>	201 - 223	<b>Mögliche Fehlermeldungen</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	<b>6809</b>	195 - 207													○		

**Funktion:** Die Operanden OP1 und OP2 werden miteinander verglichen und die Flags N, Z, und C entsprechend dem Ergebnis gesetzt. Die Operanden werden dabei nicht verändert.

**Übergabe:** keine

**Ergebnis:** ERD verändert  
C1024 Fehlernummer (0 => kein Fehler)  
N, Z, C entsprechend dem Ergebnis der Routine

Nach dem Vergleich sind bedingt Sprünge möglich:

<b>Springe, wenn...</b>	<b>Sprungbefehl</b>
...OP1 = OP2	SP0 (BEQ)
...OP1 $\neq$ OP2	SN0 (BNE)
...OP1 < OP2	SP< (BCS)
...OP1 $\leq$ OP2	J<= (BLS)
...OP1 > OP2	SP> (BHI)
...OP1 $\geq$ OP2	JC0 (BCC)

Ausführungszeit in $\mu s$	6303	215 - 271	Mögliche Fehlermeldungen	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	6809	205 - 237													○		

**Funktion:** Die Operanden OP1 und OP2 werden miteinander verglichen. Ist OP1 größer als OP2, wird OP1 mit dem Inhalt von OP2 geladen

**Übergabe:** OP1 Wert, der begrenzt werden soll  
OP2 Obergrenze

**Ergebnis:** ERD verändert  
C1024 Fehlernummer (0 => kein Fehler)  
N, Z ungültig  
C gesetzt, wenn OP1 mit OP2 geladen wurde.



## 5. SONDERBEFEHLE

### 5.1. ARITHMETIKPROZESSOR (NUR CP80)

Alle Zentraleinheiten sind mit Fließkomma-Mathematikroutinen ausgestattet (siehe Abschnitt "4. Mathematik-Routinen"). Die Zentraleinheit CP80 verfügt zusätzlich über einen schnellen Fließkomma-Arithmetikprozessor (MC68881, Motorola). Dieser Arithmetikprozessor (im weiteren mit APU abgekürzt) wird verwendet, wenn:

- die Rechengenauigkeit der Fließkomma-Mathematikroutinen nicht ausreicht
- die Rechengeschwindigkeit der Fließkomma-Mathematikroutinen nicht ausreicht
- Rechenarten benötigt werden, die im Befehlssatz der Fließkomma-Mathematikroutinen nicht enthalten sind (z.B. Winkelfunktionen, Logarithmen, ...)

#### 5.1.1. Operanden

Der Arithmetikprozessor verfügt über 8 interne Rechenregister. Diese Rechenregister werden als "interne Operanden" bezeichnet. Im Gegensatz dazu sind "externe Operanden" Zahlen, die im Datenbereich der Zentraleinheit (C0000 bis C7167) liegen. Bei den APU-Befehlen unterscheidet man:

- Befehle zum Laden und Auslesen von APU-Rechenregistern
- Befehle zur Verknüpfung von zwei APU-Rechenregistern (zwei interne Operanden)
- Befehle zur Verknüpfung eines internen Operanden mit einem externen Operanden

## 5.1.2. Aufruf von APU-Befehlen

Mit dem AWL-Befehl "MAT" werden Befehle an den APU übergeben. Indexregister und Ergebnisregister enthalten die folgenden Parameter:

<div><div>70</div><div><div>PTR</div><div>F/REG</div><div>REG</div></div></div>	<div><div>70</div><div><div>EI</div><div>D</div><div>CODE</div></div></div>
<div><div>ERA</div><div><div><div>Zeiger auf externen Operanden</div><div>00 = Indexregister R</div><div>01 = Indexregister Y<sup>1)</sup></div><div>10 = User-Stackpointer<sup>1)</sup></div></div><div><div>F/REG</div><div><div>Wenn der Befehl zwei interne Operanden hat:</div><div>F/REG = Register-Nr. des 2. Operanden (OP2<sup>2)</sup>)</div><div>Wenn der Befehl einen externen Operanden hat:</div><div>F/REG = Formatcode (siehe Tabelle in Abschnitt "Zahlenformate")</div></div><div><div>REG</div><div>Register-Nr. des 1. Operanden (OP1<sup>2)</sup>)</div></div></div></div></div>	<div><div><div>ERB</div><div><div>Befehlsart:</div><div>0 = Zwei interne Operanden</div><div>1 = Ein externer Operand</div></div><div><div>D</div><div>Laden/Abspeichern von internem Operanden:</div><div>0 = Laden</div><div>1 = Abspeichern</div></div><div><div>CODE</div><div>Befehlscode (siehe "Befehlssatz des 68881")</div></div></div></div>
	<div><div><div><div>15870</div><div>Startadresse des externen Operanden</div></div><div>Indexregister R, Y oder User-Stackpointer<sup>1)</sup></div></div></div>

<sup>1)</sup> Indexregister Y und User-Stackpointer können ab Version 5.0 des B&R PROgrammierSYSTEMes verwendet werden.

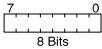
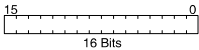


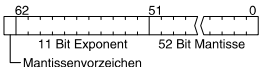
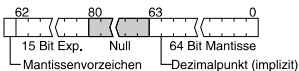
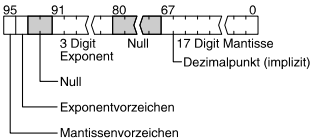
<sup>2)</sup> OP1 und OP2 sind nicht identisch mit den Operanden der Mathematik-Routinen.

### Beispiel eines APU-Aufrufes:

```
LRK    C 2000
LAD    # %00100011
LB      # %10xxxxxx
MAT
```

Zeiger auf externen Operanden  
Datenformat "Word Integer"  
Ein externer Operand, xxxxxx = Befehl

## 5.1.3.Zahlenformate

FORMAT	Bezeichnung	Länge	Formatcode <sup>1)</sup>
	BYTE INTEGER	8 Bit / 1 Byte	110
	WORD INTEGER	16 Bit / 2 Byte	100
	LONG INTEGER	32 Bit / 4 Byte	000
	SINGLE REAL	32 Bit / 4 Byte	001
	DOUBLE REAL	64 Bit / 8 Byte	101
	EXTENDED REAL	96 Bit / 12 Byte	010
	PACKED DEZIMAL REAL	96 Bit / 12 Byte	011

## 5.1.4. Befehlssatz des 68881 Arithmetikprozessors

Befehl	Bezeichnung	Beschreibung	Code (hex.)	Code (binär)
ABS	Absolutwert	$OP1 := \text{abs}(OP1)$	\$18	%011000
ACOS	Arcus Cosinus	$OP1 := \arccos(OP1)$	\$1C	%011100
ADD	Addition	$OP1 := OP1 + OP2$	\$22	%100010
ASIN	Arcus Sinus	$OP1 := \arcsin(OP1)$	\$0C	%001100
ATAN	Arcus Tangens	$OP1 := \arctan(OP1)$	\$0A	%001010
ATANH	Arcus Tangens Hyperbolicus	$OP1 := \tanh(OP1)$	\$0D	%001101
COS	Cosinus	$OP1 := \cos(OP1)$	\$1D	%011101
COSH	Cosinus Hyperbolicus	$OP1 := \cosh(OP1)$	\$19	%011001
DIV	Division	$OP1 := OP1 / OP2$	\$20	%100000
ETOX	$e^x$	$OP1 := e^{OP1}$	\$10	%010000
ETOM1	$e^{x-1}$	$OP1 := e^{OP1-1}$	\$08	%001000
GETEXP		$OP1 := \text{Exponent}(OP1)$	\$1E	%011110
GETMAN		$OP1 := \text{Mantisse}(OP1)$	\$1F	%011111
INT	Integerfunktion	$OP1 := \text{int}(OP1)$	\$01	%000001
INTRZ	Integer mit Rundung	$OP1 := \text{int}(OP1)$	\$03	%000011
LOG10	Logarithmus zur Basis 10	$OP1 := \log_{10}(OP1)$	\$15	%010101
LOG2	Logarithmus zur Basis 2	$OP1 := \log_2(OP1)$	\$16	%010110
LOGN	Logarithmus zur Basis e	$OP1 := \ln(OP1)$	\$14	%010100



LOGNP1		$OP1 := \ln(OP1 + 1)$	\$06	%000110
MOD	Modulofunktion	$OP1 := \text{mod}(OP1)$	\$21	%100001
MOVE	Laden oder Auslesen		\$00	%000000
MOVECR	Laden mit Konstante	$OP1 := \text{const.}$	\$3B	%111011
MUL	Multiplikation	$OP1 := OP1 * OP2$	\$23	%100011
NEG	Negation	$OP1 := 0 - OP1$	\$1A	%011010
SCALE		$OP1 := OP1 * \text{int}(2^{OP1})$	\$26	%100110
SGLDIV	Single Precision Division	$OP1 := OP1 / OP2$	\$24	%100100
SGLMUL	Single Precision Multiplikation	$OP1 := OP1 * OP2$	\$27	%100111
SIN	Sinus	$OP1 := \sin(OP1)$	\$0E	%001110
SINCOS	Sinus und Cosinus	$OP1 := \sin(OP1); \text{Reg. n} := \cos(OP1)$	\$3n	%110nnn
SINH	Sinus Hyperbolicus	$OP1 := \sinh(OP1)$	\$02	%000010
SQRT	Quadratwurzel	$OP1 := \text{sqr}(OP1)$	\$04	%000100
SUB	Subtraktion	$OP1 := OP1 - OP2$	\$28	%101000
TAN	Tangens	$OP1 := \tan(OP1)$	\$0F	%001111
TANH	Tangens Hyperbolicus	$OP1 := \tanh(OP1)$	\$09	%001001
TENTOX		$OP1 := 10^{OP1}$	\$12	%010010
TWOTOX		$OP1 := 2^{OP1}$	\$11	%010001

Die Ausführungszeit der APU-Befehle beträgt ca. 330  $\mu\text{s}$ .

## 5.1.5. KONSTANTEN

Der 68881 Arithmetikprozessor hat die wichtigsten technischen Konstanten bereits fix gespeichert. Mit dem Befehl \$3B (%111011) wird ein Rechenregister mit einer Konstante geladen. Vor dem Aufruf zeigt der in ERA definierte Zeiger (Indexregister R, Y oder User-Stackpointer) auf eine Speicherstelle, die die Nummer der gewünschten Konstante enthält:

Nr.	Konstante	Nr.	Konstante	Nr.	Konstante	Nr.	Konstante
\$00	P	\$30	$\ln(2)$	\$36	$10^8$	\$3C	$10^{512}$
\$0B	$\log_{10}(2)$	\$31	$\ln(10)$	\$37	$10^{16}$	\$3D	$10^{1024}$
\$0C	e	\$32	$10^0$	\$38	$10^{32}$	\$3E	$10^{2048}$
\$0D	$\log_2(e)$	\$33	$10^1$	\$39	$10^{64}$	\$3F	$10^{4096}$
\$0E	$\log_{10}(e)$	\$34	$10^2$	\$3A	$10^{128}$		
\$0F	0	\$35	$10^4$	\$3B	$10^{256}$		

**Beispiel:** Das Rechenregister 2 soll mit der Konstante e (Nr. \$0C) geladen werden.

```

LAD      # $0C          Konstantennummer
=        C 0100         zwischenspeichern
LRK      C 0100         Indexregister auf Konstantennummer
LAD      # %00000010    Registernummer 2
LB       # %00111011    Befehlscode "Konstante laden"
MAT

```

**Beispiel:** Der Inhalt der APU-Rechenregister 1 und 4 soll multipliziert werden.

LAD	#	%00100001	Registernummern der beiden internen Operanden
LB	#	%00100011	Befehlscode für Multiplikation = \$23
MAT			

**Beispiel:** Das APU-Register 2 soll mit der 2 Byte-Integerzahl in C 0100, 0101 geladen werden.

LRK	C	0100	Datenquelladresse
LAD	#	%00100010	Zahlenformat und Registernummer
LB	#	%10000000	Befehlscode für Laden/Abspeichern = \$00
MAT			

**Beispiel:** Das Ergebnis in APU-Rechenregister 6 soll im Format long integer (4 Byte) in den Speicherstellen C 0200 bis C 0203 abgespeichert werden.

LRK	C	0200	Datenzieladresse
LAD	#	%00000110	Zahlenformat und Registernummer
LB	#	%11000000	Befehlscode für Laden/Abspeichern = \$00
MAT			



## 5.2. SCHNITTSTELLENBEFEHLE (NUR CP80, PP60 UND NTCP6#)

Zur Bedienung der Anwender-Schnittstellen der Zentraleinheiten CP80, PP60 und NTCP6# sind folgende AWL-Befehle verfügbar:

<b>SOB</b>	Zeichen ausgeben
<b>SIB</b>	Zeichen einlesen
<b>SC</b>	Schnittstellenstatus anfordern
<b>SF</b>	Schnittstellenfunktionen (z.B.: Initialisierung)

### 5.2.1. SOB - Zeichen ausgeben

Mit dem Befehl SOB wird ein einzelnes Zeichen über die serielle Schnittstelle gesendet. Die Schnittstelle muß vorher initialisiert worden sein (siehe Befehl SF). Das auszugebende Zeichen wird in ERA übergeben. ERA wird durch den SOB-Aufruf nicht verändert. Nach dem SOB-Aufruf zeigt das Carry-Flag an, ob die Ausgabe möglich war:

Carry-Flag = 0	Zeichen ausgegeben
Carry-Flag = 1	Ausgabe war nicht möglich (Sendepuffer voll)

**Beispiel:** Ausgabe des ASCII-Zeichens "A"

LAD	# 'A	Zeichen "A" in Ergebnisregister ERA an die Schnittstellenroutine übergeben
SOB		senden

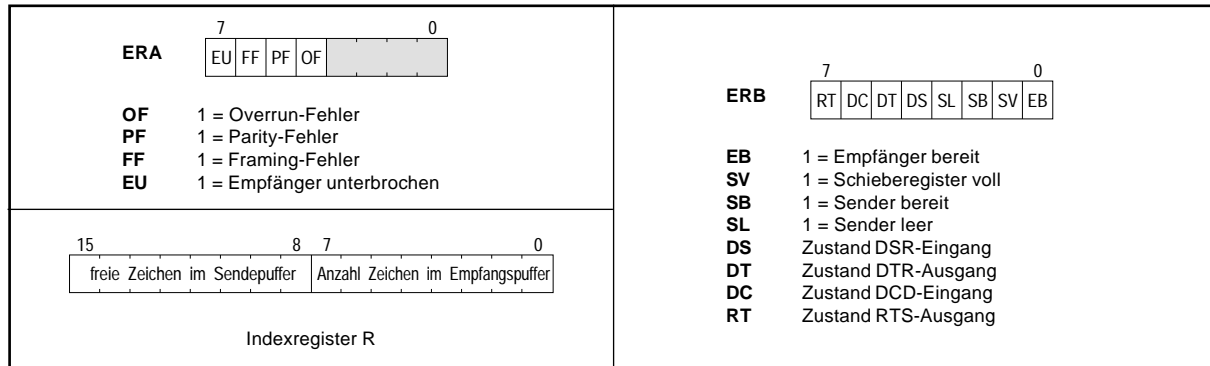
## 5.2.2. SIB - Zeichen einlesen

Mit dem Befehl SIB wird ein Zeichen von der seriellen Schnittstelle eingelesen. Die Schnittstelle muß vorher initialisiert worden sein (siehe Befehl SF). Nach dem SIB-Befehl enthält ERA das eingelesene Zeichen. Wurde kein Zeichen empfangen (Empfangspuffer leer), so wird ERA durch den SIB-Aufruf nicht verändert. Carry- und Zero-Flag zeigen nach dem SIB-Aufruf an, ob ein Zeichen empfangen wurde, oder ob ein Schnittstellenfehler auftrat:

C = 0	C = 1	
gültiges Zeichen empfangen und in ERA gespeichert	kein gültiges Zeichen empfangen	
	Z = 1	Z = 1
	Empfangspuffer leer	<div>Übertragungsfehler; ERA enthält einen Fehlercode:</div> <div><div><div>7</div><div>0</div><div>EU</div><div>FF</div><div>PF</div><div>OF</div><div></div><div></div><div></div><div></div></div><div><div>OF</div><div>1 = Overrun-Fehler</div><div>PF</div><div>1 = Parität-Fehler</div><div>FF</div><div>1 = Framing-Fehler</div><div>EU</div><div>1 = Empfänger unterbrochen</div></div></div>

## 5.2.3. SC - Schnittstellenstatus anfordern

Der Befehl SC liefert Informationen über den Zustand der Schnittstelle und die Sende-/Empfangspuffer. Die Schnittstelle muß vorher initialisiert worden sein (siehe Befehl SF). Nach dem SC-Aufruf enthalten die Ergebnisregister und das Indexregister folgende Informationen:



## 5.2.4. SF - Schnittstellenfunktionen

Der Befehl SF wird verwendet für:

- Schnittstelle initialisieren
- Manuelle Steuerung der Handshake-Leitungen (DTR, RTS)
- Löschen des Sende- bzw. Empfangspuffers
- Voreinstellungen für Senden/Empfangen im Blockmode
- Senden und Empfangen im Blockmode
- Blockmode-Status anfordern

Zusätzlich zu der Bedienung mit den Befehlen SOB (Einzelzeichen senden) und SIB (Einzelzeichen empfangen) kann die Schnittstelle in einem sogenannten Block-Mode betrieben werden. Dadurch kann der Anwender ganze Datenblöcke (Frames) senden und empfangen. Diese Funktion wird auch benötigt, um mit netzwerkfähigen Bedientableaus (z.B. BRRT28) oder dem Massenspeichergerät BRMEC zu kommunizieren.

Die Blockmode-Funktionen können nur verwendet werden, wenn:

- a. das B&R PROgrammierSYStem Version 5.0 oder höher verwendet wird
- oder
- b. Das Systemmodul einer älteren B&R PROgrammierSYStem-Version gegen ein Systemmodul der Version 3.1 oder höher ausgetauscht wurde



# SF - Schnittstelle initialisieren

Die Schnittstelle muß vor der ersten Verwendung initialisiert werden, unabhängig davon, ob mit den Befehlen SOB/SIB (Einzelzeichen senden/empfangen) gearbeitet wird, oder im Blockmode. Parameter:

<div>70</div> <div>ERA</div> <div><div>0</div><div>0</div><div>RS<sub>pp</sub></div><div>RS</div><div>BAUD</div></div>				<div>70</div> <div>ERB</div> <div><div>EI</div><div>SI</div><div>SB</div><div>P</div><div>DI</div><div>OE</div><div>DB</div></div>			
<div>RS<sub>pp</sub><sup>1)</sup></div> <div>1 = Deaktivieren des RS485 Empfängers für Betrieb des PP60 mit RS232-Hardware auf Pegelumsetzer mit RS485-Verhalten.</div>				<div>DB</div> <div>Datenbits:</div> <div>00 = 5 Bit10 = 7 Bit</div> <div>01 = 6 Bit11 = 8 Bit</div>			
<div>RS</div> <div>1 = RS485 Mode <sup>2)</sup></div> <div>0 = RS422/RS232 Mode</div>				<div>OE</div> <div>Parität</div> <div>1 = ungerade0 = gerade</div> <div>DI<sup>3)</sup></div> <div>DSR aktiv</div> <div>1 = inaktiv0 = aktiv</div> <div>P</div> <div>Parität ein/aus</div> <div>1 = aus0 = ein</div> <div>SB</div> <div>Stop-Bits</div> <div>1 = 2 Stop-Bits0 = 1 Stop-Bit</div> <div>SI</div> <div>Sender-Interrupt</div> <div>1 = freigegeben0 = gesperrt</div> <div>EI</div> <div>Empfänger-Interrupt</div> <div>1 = freigegeben0 = gesperrt</div>			
<div>Baud</div> <div>Baudrate</div> <div>Baud</div> <div>Baudrate</div>							
0000 ----1000 1200 Baud							
0001 50 Baud1001 1800 Baud							
0010 75 Baud1010 2400 Baud							
0011 110 Baud1011 3600 Baud							
0100 135 Baud1100 4800 Baud							
0101 150 Baud1101 7200 Baud							
0110 300 Baud1110 9600 Baud							
0111 600 Baud1111 19200 Baud							
<div><div><sup>1)</sup> erst ab Systemmodul Version 3.8 verfügbar</div><div><sup>2)</sup> Im RS485 Mode schaltet sich der Sender automatisch vom Bus, wenn 300 ms kein Zeichen gesendet wird.</div></div>				<div><div><sup>3)</sup> nur bei Punkt zu Punkt Verbindung (RS232 &lt;=&gt; RS232) verwendet.</div></div>			
				<div><div>15870</div><div>Länge Sendepuffer (1 bis 255)</div><div>Länge Empfangspuffer (1 bis 255)</div></div> <div>Indexregister R</div>			

## SF - Manuelle Bedienung der Handshake-Leitungen und Sende-/Empfangspuffer

<div>7 0</div> <div>1 0 0 0 0 0 0 0</div>	LAD SF # \$80	RTS-Leitung auf Low schalten. <sup>1)</sup>
<div>7 0</div> <div>1 0 0 0 0 0 0 1</div>	LAD SF # \$81	RTS-Leitung auf High Schalten. <sup>1)</sup>
<div>7 0</div> <div>1 0 0 0 0 0 1 0</div>	LAD SF # \$82	Automatische DTR-Behandlung ein (Nach Power-ON ist diese immer ausgeschaltet!)
<div>7 0</div> <div>1 0 0 0 0 0 1 1</div>	LAD SF # \$83	DTR-Leitung auf Low schalten. DTR bleibt so lange low, bis mit Befehl \$82 die automatische DTR-Behandlung wieder freigegeben wird und kein Busy-Status mehr vorliegt.
<div>7 0</div> <div>1 0 0 0 0 1 0 0</div>	LAD SF # \$84	Sende- und Empfangspuffer löschen (Pointer zurücksetzen). Der Busy-Status wird gelöscht, falls er nicht mit dem Befehl \$83 gesperrt wurde.
<div>7 0</div> <div>1 0 0 0 0 1 0 1</div>	LAD SF # \$85	Empfangspuffer löschen (Pointer zurücksetzen). Der Busy-Status wird gelöscht, falls er nicht mit dem Befehl \$83 gesperrt wurde.
<div>7 0</div> <div>1 0 0 0 0 1 1 0</div>	LAD SF # \$86	Sendepuffer löschen (Pointer zurücksetzen).

<sup>1)</sup> Bei der Verbindung RS232 auf Buskoppler (ECINT1) wird mit diesem Befehl die DTR-Leitung geschaltet.

## SF - Blockmode initialisieren

Im Blockmode definiert der Anwender einen Sendepuffer und einen Empfangspuffer. Beim Senden wird der gewünschte Datenblock (Frame) in diesen Puffer geschrieben und mit einem SF-Kommando der Sendevorgang gestartet. Das Senden der einzelnen Zeichen erfolgt dann automatisch (timerinterrupt-gesteuert mit dem Usertimerinterrupthandler \$US2).

Das Blocksenden/-Empfangen kann wahlweise auch mit dem B&R Standardprotokoll erfolgen (MININET-Protokoll). Dieser Protokollmode wird z.B. benötigt, wenn mit einem BRRT28-Bedientableau oder einem BRMEC Massenspeicher kommuniziert wird. In diesem Fall müssen die Einträge im Sendepuffer (Frames) der Syntax des B&R Protokolls entsprechen:

<b>Befehl vom Master</b>	STX	LEN	NODE	INDEX	DATA	...	CHK
<b>Antwort vom Slave ohne Daten (kurze Antwort)</b>	ACK						
<b>Antwort vom Slave mit Daten (lange Antwort)</b>	STX	LEN	NODE	INDEX	DATA	...	CHK

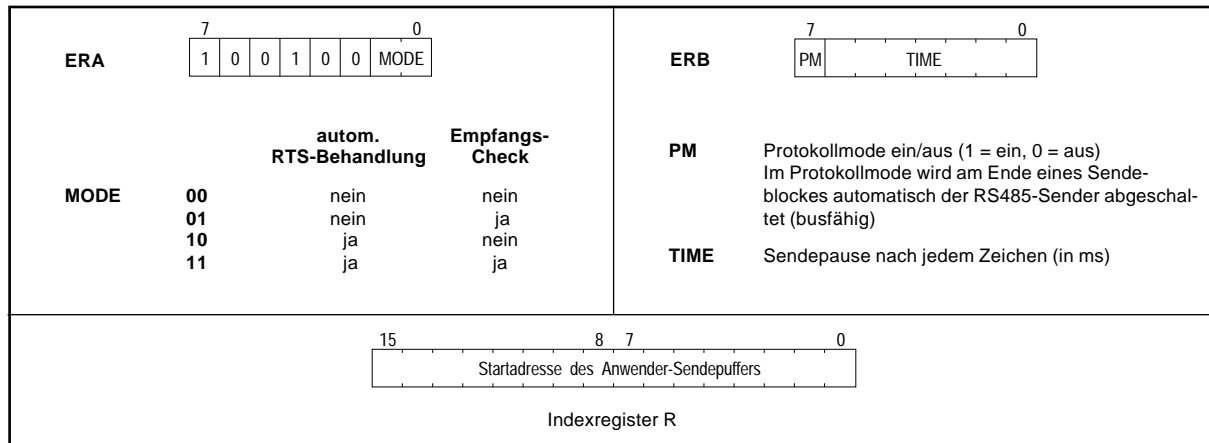
STX	.....	Startzeichen, das den Beginn eines Frames anzeigt (\$02)
LEN	.....	Länge des gesamten Frames
NODE	.....	Stationsnummer der Station, an die gesendet wird
INDEX	.....	Indexnummer zur Identifizierung zusammengehörender Frames
DATA	.....	Datenbytes. Nach jedem Datenbyte \$02 wird ein Füllbyteeingefügt und gesendet, um \$02 (=> STX) als Datenbyte zu ermöglichen.
CHK	.....	Prüfsumme (Checksum) über den Frame. Diese Prüfsumme wird automatisch vom Betriebssystem errechnet und muß nicht vom Anwender eingetragen werden.
ACK	.....	Bestätigung, daß ein Frame fehlerfrei empfangen wurde (\$06)

*B&R Standardprotokoll (MININET)*

Für die Initialisierung des Blockmodes ist folgender Vorgang einzuhalten:

- a. Initialisieren des Blockmode-Sendens mit den SF-Kommandos \$90 bis \$93. Dabei wird festgelegt:
  - die Startadresse des Anwender-Sendepuffers
  - eine Sendepause nach jedem Zeichen
  - ob für den Handshake die RTS-Leitung verwendet werden soll
  - ob ein Empfangscheck durchgeführt werden soll
  - ob die Sendeblocke im Protokollmode gesendet werden sollen
- b. Starten der Usertimerinterruptroutine mit dem Timerinterrupthandler \$US2. Die benötigten Parameter werden vom SF-Aufruf (a.) automatisch festgelegt.
- c. Initialisieren des Blockmode-Empfangens mit dem SF-Kommando \$98. Dabei wird festgelegt:
  - die Startadresse des Anwender-Empfangspuffers
  - ein Timeout für das Empfangen

## a. Initialisierung für Blockmode-Senden



## b. User-Timerinterruptroutine starten

Mit dem User-Timerinterrupthandler \$US2 wird die Sende-Blockmodeinitialisierung abgeschlossen. Der unter a. beschriebene SF-Aufruf definiert automatisch die Parameter für \$US2.

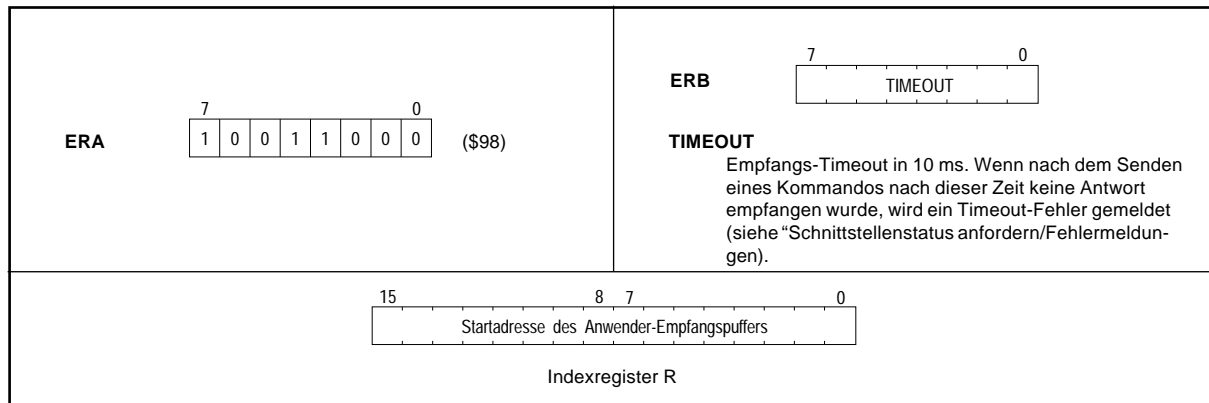
**Beispiel:** Die Anwenderschnittstelle wird für Blockmode ohne B&R-Protokoll initialisiert. Der User-Sendepuffer beginnt bei C 3000. Da der Empfänger nicht über Handshake-Leitungen verfügt, wird "autom. RTS-Behandlung" ausgeschaltet und nach jedem Zeichen eine 2 ms-Pause eingelegt. Baudrate, Stop-Bits und Parity müssen vorher initialisiert worden sein.

LRK	C	3000	Startadresse User-Sendepuffer
LB	#	002	2 ms Sendepause nach jedem Zeichen
LAD	#	%10010000	Autom. RTS-Behandlung aus, Check aus
SF			
SPU		\$US2	Starten der Timerinterruptroutine

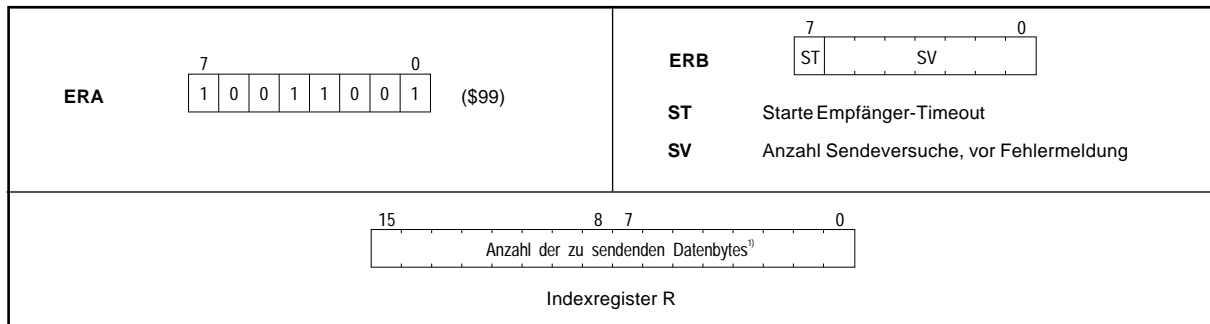
Zwischen dem SF-Aufruf und dem Starten der Timerinterruptroutine mit "SPU \$US2" dürfen keine Befehle stehen, die das Indexregister oder ERA verändern.

## c. Initialisierung für Blockmode-Empfangen

Dieser Befehl muß nach dem Initialisieren des Blockmode-Sendens aufgerufen werden. Er definiert die Startadresse des User-Empfangspuffers. Durch wiederholtes Absetzen dieses Befehles mit unterschiedlichen Puffer-Startadressen kann alternierend mit mehreren Empfangspuffern gearbeitet werden.



## SF - Daten im Blockmode senden

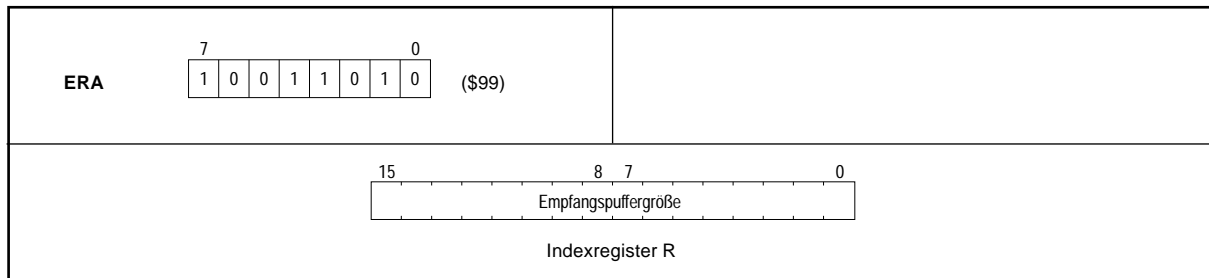


<sup>1)</sup> Im Protokollmode wird die Anzahl der zu sendenden Zeichen aus dem Frame bestimmt und die Angabe im Indexregister R ignoriert

Steht bei eingeschaltetem Protokollmode an der ersten Stelle im Sendepuffer kein STX (\$02), so wird der Datenblock nicht im Protokollmode gesendet. Diese Funktion kann verwendet werden, um Sonderzeichen zu senden, die Protokoll-Steuerzeichen sind (z.B. \$06).



## SF - Daten im Blockmode empfangen



Die Empfangspuffergröße muß mindestens 32 Zeichen betragen. Sie muß um 2 Zeichen größer sein, als die eigentliche Datenblocklänge. Im Protokollmode wird die Empfangspuffergröße mit 255+2 Zeichen vorausgesetzt, d.h. die Angabe im Indexregister wird nicht berücksichtigt, der Anwender muß einen Pufferbereich von 255+2 Zeichen zur Verfügung stellen.

**Beispiel:**      Datenblock empfangen. Empfangspuffergröße = 128 Bytes:

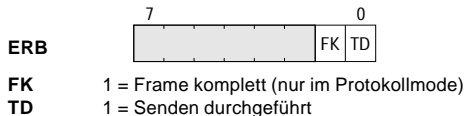
LD	#	00128	Empfangspuffergröße
DXR			
LAD	#	\$9A	Kommando "Daten empfangen"
SF			

## SF - Schnittstellenstatus anfordern (\$9F)

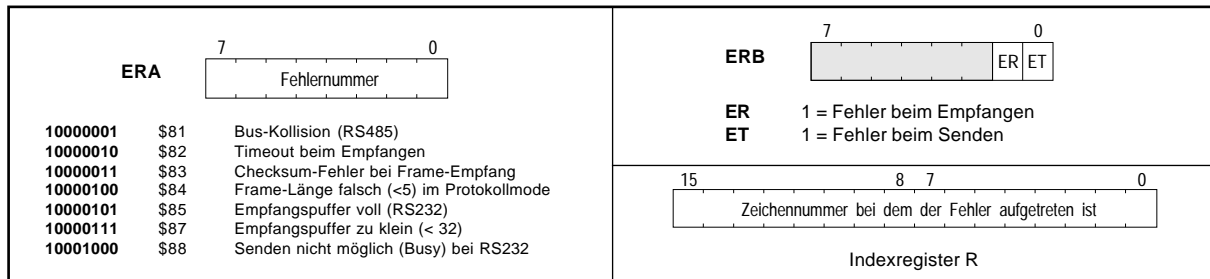
Mit diesem Kommando werden Informationen über den Zustand der Schnittstelle angefordert (nur bei Blockmode/Protokollmode relevant).

LAD      # \$9F  
SF

Die Antwort ist abhängig davon, ob ein Fehler aufgetreten ist: Ist nach dem SF-Aufruf das Carry-Flag = 0, so ist kein Fehler aufgetreten und ERB enthält die folgenden Informationen:



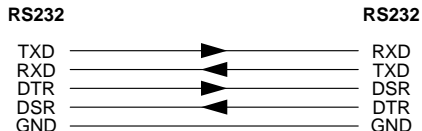
Im Fehlerfall ist das Carry-Flag nach dem SF-Aufruf gesetzt und ERA enthält einen Fehlercode:



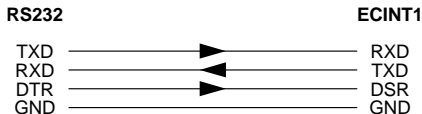
## 5.2.5. Unterschiedliche Behandlung der Handshake-Leitungen

Bei der Initialisierung sind die jeweiligen Interrupts, falls Sender und/oder Empfänger verwendet werden, zu setzen. Bei der Behandlung der Handshake-Leitungen sind folgende Verbindungen zu unterscheiden.

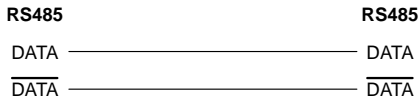
### Punkt zu Punkt RS232 <=> RS232 (SIB/SOB, Blockmode)



### Punkt zu Punkt RS232 <=> ECINT1 (SIB/SOB, Blockmode)



### RS485 (SIB/SOB, Blockmode)



Im folgenden werden alle möglichen Verbindungen und die verschiedenen Betriebsarten für Senden und Empfangen behandelt. Zu jeder Betriebsart wird erklärt, wie sich die Veränderung der Datenbits DSR <sup>1)</sup>, DTR <sup>2)</sup> und RTS <sup>2)</sup> auf die Schnittstelle und die Datenleitungen (Hardware) auswirken.

## a) RS232 (RS422) Punkt zu Punkt

**Schnittstelleninitialisierung:** RS Bit = 0 (RS232/422)

### SIB/SOB:

DSR: DSR: inaktiv: Das Signal "nicht empfangsbereit" der Gegenstelle (DTR-Leitung auf low) wird nicht berücksichtigt.

DSR aktiv: Meldet die Gegenstelle "nicht empfangsbereit" (DTR-Leitung auf low), so können noch bis zu 2 Zeichen gesendet werden (Zeichen, die noch im internen Buffer des Schnittstellenbausteins stehen).

DTR: Nach dem Power-On ist die automatische DTR-Behandlung ausgeschaltet. Mit dem Befehl \$82 wird die DTR Leitung automatisch behandelt (ist der Puffer zu 80% bis 90% voll, wird das Signal "nicht empfangsbereit" gesendet => DTR-Leitung auf low). Mit \$83 kann man übergeordnet ein "nicht empfangsbereit" Signal für die Gegenstelle generieren.

RTS: Wird nicht verwendet

### Blockmode ohne Protokoll:

DSR: wie SIB/SOB

DTR: wie SIB/SOB

RTS: Mit dem Befehl "Daten im Blockmode senden" wird die RTS Leitung auf high geschaltet. Die RTS-Leitung kann mit dem Befehl \$80 wieder auf Low geschaltet werden. Der Anwender muß dafür sorgen, daß die RTS-Leitung auf Low geschaltet wird, wenn alle Zeichen in Puffer des Schnittstellenbausteins eingetragen sind und die "Datenübertragungszeit" abgelaufen ist. Wird 300 ms kein Zeichen gesendet, wird automatisch abgeschaltet.

### Blockmode mit Protokoll:

DSR: wie SIB/SOB

DTR: keine Behandlung (String muß in Empfangspuffer passen)

RTS: wie Blockmode ohne Protokoll.

Datenformat im Sendepuffer:



Die Checksumme wird vom Betriebssystem errechnet. Ebenso werden die Füllbytes (nach jedem \$02 im Datenstrom) vom Betriebssystem eingefügt.

<sup>1)</sup> siehe "SF - Schnittstelle initialisieren"

<sup>2)</sup> siehe "SF - Manuelle Bedienung der Handschake-Leitungen"

## b) RS232 auf Buskoppler

**Schnittstelleninitialisierung:** RS Bit = 1 (RS485)

### SIB/SOB:

DSR: wird nicht verwendet

DTR: keine Behandlung

RTS: Wird im RS485 Mode mit RS232 Hardware ein Buskoppler angesteuert, so wird die DTR Leitung für Bus Aktiv- bzw. Passiv-Schalten verwendet. Dies erfolgt mit den Befehlen \$80 (DTR-Leitung auf Low) und \$81 (DTR-Leitung auf High). Wird ein Fremdgerät angesteuert, so muß dieses ein Echobyte generieren können. Das vom Buskoppler oder Fremdgerät zurückgesendete Echobyte muß vom Programm ausgewertet werden (durch Auslesen oder Löschen des Empfangspuffers).

Um auch beim PP60 diese Funktion zu ermöglichen, muß der RS485 Empfänger deaktiviert werden, damit sich die Empfangsstufen von TTY, RS232 und RS485 nicht beeinflussen.

	7					0
ERA	0	0	RS <sub>pp</sub>	RS		BAUD
RS <sub>pp</sub>	1 = Deaktivieren des RS485 Empfängers für Betrieb des PP60 mit RS232-Hardware auf Pegelumsetzer mit RS485-Verhalten (ab Systemmodulversion 3.8.)					
RS	1 = RS485 Mode					

### Blockmode ohne Protokoll:

DSR: wird nicht verwendet

DTR: wird nicht verwendet

RTS: Prinzip wie SIB/SOB, das Echobyte wird jedoch vom Betriebssystem selbst behandelt. Der Bus wird mit dem Befehl "Daten im Blockmode senden" aktiv und mit Erhalt des letzten Echobytes passiv geschaltet.

**!! KEINE MANUELLE BEDIENTUNG !!**

### Blockmode mit Protokoll:

Im Sendepuffer müssen folgende Einträge sein:

STX LEN NODE INDEX DATA ..... DATA

sonst wie RS232 - Punkt zu Punkt Verbindung

## c) RS485 (Busfähig)

**Schnittstelleninitialisierung:** RS Bit = 1 (RS485)

### SIB/SOB:

DSR: wird nicht verwendet

DTR: keine Behandlung

RTS: Mit den Befehlen \$80 und \$81 wird der Bus aktiv/passiv geschaltet. Das Echobyte muß jedoch ausgewertet werden. Wird 300 ms kein Zeichen gesendet, so wird der Sender automatisch wieder vom Bus geschaltet.

### Blockmode ohne Protokoll:

DSR: wird nicht verwendet

DTR: keine Behandlung

RTS: Das Echobyte wird vom Betriebssystem aus behandelt. Der Bus wird mit dem Befehl "Daten im Blockmode senden" aktiv und mit dem Empfang des letzten Echobytes wieder passiv geschaltet.

**!! KEINE MANUELLE BETEDIENUNG !!**

Das RS<sub>pp</sub> Bit muß 0 sein.

### Blockmode mit Protokoll:

DSR, DTR, RTS: wie Blockmode ohne Protokoll

Im Sendepuffer müssen folgende Einträge sein:

STX	LEN	NODE	INDEX	DATA	.....	DATA
-----	-----	------	-------	------	-------	------

Die Checksumme wird vom Betriebssystem errechnet. Ebenso werden die Füllbytes (nach jedem \$02 im Datenstrom) vom Betriebssystem eingefügt.

Das RS<sub>pp</sub> Bit muß 0 sein.

## 6. ANHANG

### 6.1. Alphabetische Übersicht der B&R Mnemonics

+	92	COA	175	LEI	46	SEI	173
++B	95	COB	176	LER	48	SET	169
+B	94	DA	126	LEU	47	SK0	152
+D	96	DB	127	LEY	49	SK1	154
-	98	DEC	125	LR	39	SL	132
SUB	99	DK	177	LRK	40	SLA	133
- -B	101	DR	128	LRL	41	SLB	134
-B	100	DS	129	LS	45	SLD	135
-D	102	DXR	66	LY	42	SLI	140
=	52	EB	89	LYK	43	SN0	148
=B	53	EIM	90	LYL	44	SN0L	150
=D	54	END	162	MAB	60	SP<	148
=R	55	EXG	68	MAC	62	SP<L	150
=S	57	EXO	88	MBA	61	SP>	148
=Y	56	IA	121	MCA	63	SP>L	150
A*B	105	IB	122	MRS	65	SP0	148
A+B	97	INC	120	MSR	64	SP0L	150
A-B	103	IR	123	NOP	161	SPI	160
ADD	93	IS	124	OB	86	SPU	156
AIM	84	J+	148	OD	85	SR	136
ANS	74	J+L	150	OIM	87	SRA	137
AVB	108	J-	148	PRS	164	SRB	138
AVS	75	J-L	150	PSH	70	SRD	139
B	115	J<=	148	PUL	72	SRE	143
B+R	104	J<=L	150	RET	158	TFR	67
BB	116	JC0	148	RLA	141	TIM	117
BNS	76	JCOL	150	RLB	142	UB	83
BVS	77	K	174	RNS	78	UND	82
CLA	167	LAD	34	RRA	144	VB	110
CLB	168	LB	35	RRB	145	VR	111
CLC	170	LD	36	RST	165	VRK	112
CLI	172	LDK	37	RVS	79	VY	113
CLR	166	LDL	38	SEC	171	VYK	114
CMP	109						

## 6.2. Alphabetische Übersicht der Motorola Mnemonics

ABA	97	CLI	172	LDK	37	ROR	143
ABX	104	CLR	166	LDL	38	RORA	144
ADCA	93	CLRA	167	LDS	45	RORB	145
ADCB	95	CLRB	168	LDX	39	RST	165
ADDA	92	CMPA	109	LDX#	40	RTS	158
ADDB	94	CMPB	110	LDXL	41	SBA	103
ADD	96	COM	174	LDY	42	SBCA	99
AIM	84	COMA	175	LDY#	43	SBCB	101
ANDA	82	COMB	176	LDYL	44	SEC	171
ANDB	83	CPX	111	LEA!	46	SEI	173
ASL	132	CPX#	112	LEAU	47	SET	169
ASLA	133	CPY	113	LEAX	48	SK0	152
ASLB	134	CPY#	114	LEAY	49	SK1	154
ASLD	135	DAA	177	LSR	136	STAA	52
BCC	148	DEC	125	LSRA	137	STAB	53
BCCL	150	DECA	126	LSRB	138	STAD	54
BCS	148	DECB	127	LSRD	139	STS	57
BCSL	150	DES	129	MUL	105	STX	55
BEQ	148	DEX	128	NOP	161	STY	56
BEQL	150	EIM	90	OIM	87	SUBA	98
BHI	148	END	162	ORAA	85	SUBB	100
BHIL	150	EORA	88	ORAB	86	SUBD	102
BITA	115	EORB	89	PRS	164	TAB	60
BITB	116	EXG	68	PSH	70	TAP	62
BLS	148	INC	120	PSHA	74	TBA	61
BLSL	150	INCA	121	PSHB	76	TFR	67
BMI	148	INCB	122	PSHX	78	TIM	117
BMIL	150	INS	124	PUL	72	TPA	63
BNE	148	INX	123	PULA	75	TSX	64
BNEL	150	JMP	160	PULB	77	TXS	65
BPL	148	JSR	156	PULX	79	XGDX	66
BPLL	150	LDAA	34	ROL	140		
CBA	108	LDAB	35	ROLA	141		
CLC	170	LDD	36	ROLB	142		



## 6.3. Alphabetische Übersicht der Mathematik-Routinen

CAF	199	LF2	191
CBCD	210	LIL1	189
CBIN	211	LIL2	189
CBP	218	LIW1	190
CBPP	214	LIW2	190
CBPQ	216	MADD	183
CFA	200	MCMP	230
CFA0	202	MCOP	186
CFEA	204	MDIV	184
CIA	206	MEXG	186
CIA0	208	MHIL	231
CIM	222	MLOL	232
CMI	224	MMUL	184
CPB	220	MSGN	185
CPBQ	217	MSQR	185
FCLR	229	MSUB	183
FCOP	226	RFM1	196
FM2B	197	RFM2	196
FM3B	197	RFM3	196
FM4B	198	SAL	192
FSMB	227	SAW	192
FSMW	228	SFM1	195
LAL1	187	SFM2	195
LAL2	187	SFM3	195
LAW1	188	SFX	194
LAW2	188	SIL	193
LF1	191	SIW	193

## 6.4. Alphabetische Übersicht der B&R Kurz Mnemonics

--	99	KA	175
++	93	KB	176
A	93	L	34
C	166	O	85
E	88	P	164
I	52	R	165
J<	148	RL	140
J>	148	RR	143
J0	148	U	82
J1	148	V	109

## 6.5. Stichwortverzeichnis

### A

Absolute Adressierung	21
Adressierungsarten	12, 17
Absolute Adressierung	21
Direkte Adressierung	19
Implizite Adressierung	17
Indirekte Adressierung	30
Indizierte Adressierung	25
Negation	32
Relative Adressierung	29
Unmittelbare Adressierung	23
Adreßvorwahlen	13
Arithmetikbefehle	233
Arithmetikprozessor	233

### B

Befehlsabkürzung	10
Befehlssatz des 68881 - Arithmetikprozessors	236

### C

Carry-Flag	14
CPU-Typ	9
CPU-Typ A	15
CPU-Typ B	15

### D

Direkte Adressierung	19
----------------------	----

### E

Einfachgenaues Fließkommaformat	181
---------------------------------	-----

### H

Half-Carry	14
------------	----

### I

IEEE-Format	181
Implizite Adressierung	17
Indirekte Adressierung	30
Indizierte Adressierung	25
IRQ Mask	14

### M

Mathematik-Routinen	179
Mnemonic	10

### N

Negation	32
Negativ-Flag	14

### O

Opcode	12
Overflow-Flag	14

### R

Relative Adressierung	29
-----------------------	----

### S

Schnittstellenbefehle	241
SC - Schnittstellenstatus anfordern	243
SF - Schnittstellenfunktionen	244
Blockmode initialisieren	247
Daten im Blockmode empfangen	253
Daten im Blockmode senden	252
Manuelle Bedienung der Handshakeleitungen und Sende-/Empfangspuffer	246
Schnittstelle initialisieren	245
Schnittstellenstatus anfordern (\$9F)	254
SIB - Zeichen einlesen	242
SOB - Zeichen ausgeben	241
Statusregister	14
Syntax	9

### U

Unmittelbare Adressierung	23
---------------------------	----

### Z

Zahlenformate	181
Absolut mit Vorzeichen	182
Einfachgenaues Fließkommaformat	181
IEEE-Format	181
Integer	182
Zahlenformate (APU)	235
Zero-Flag	14