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		
1.3.5. Adressierungsarten / Opcode	12	MSQR - Quadratwurzel im Fließkommaformat	185
1.3.6. Adreßvorwahlen	13	4.3. Operandenmanipulation	405
1.3.7. Statusregister	14	MSGN - Vorzeichenwechsel von Operand 1	185
1.4. Genauere Beschreibung	15	MCOP - Operand OP1 nach Operand OP2 kopieren	186
2. Adressierungsarten	17	MEXG - Operanden OP1 und OP2 miteinander vertauschen	186
2.1. IMPL Implizite Adressierung	17	4.4. Operanden laden	
2.2. DIR Direkte Adressierung	19	LAL1 - Lade Operand OP1 mit Zahl (Absolut-Lang)	187
2.3. EXT Absolute Adressierung	21	LAL2 - Lade Operand OP2 mit Zahl (Absolut-Lang)	187
2.4. IMMED Unmittelbare Adressierung	23	LAW1 - Lade Operand OP1 mit Zahl (Absolut-Kurz)	188
2.5. IND Indizierte Adressierung 2.6. REL Relative Adressierung	25 29	LAW2 - Lade Operand OP2 mit Zahl (Absolut-Kurz)	188
2.7. Indirekte Adressierung	30	LIL1 - Lade Operand OP1 mit Zahl (Integer-Lang)	189
2.7. Negation	32	LIL2 - Lade Operand OP2 mit Zahl (Integer-Lang)	189
3. AWL-Befehle	33	LIW1 - Lade Operand OP1 mit Zahl (Integer-Kurz)	190
3.1. Ladebefehle	33	LIW2 - Lade Operand OP2 mit Zahl (Integer-Kurz)	190
3.2. Speicherbefehle	51	LF1 - Lade Operand OP1 mit Zahl (IEEE-Format)	191
3.3. Datenaustausch zwischen Registern	59	LF2 - Lade Operand OP2 mit Zahl (IEEE-Format)	191
3.4. Stapeloperationen	69	4.5. Operand abspeichern	
3.5. Logische Verknüpfungen	81	SAL - Operand OP1 im Format Absolut-Lang abspeichern	192
3.6. Arithmetische Operationen	91	SAW - Operand OP1 im Format Absolut-Kurz abspeichern	192
3.7. Vergleichs- und Testbefehle	107	SIL - Operand OP1 im Format Integer-Lang abspeichern	193
3.8. Inkrement- und Dekrementbefehle	119	SIW - Operand OP1 im Format Integer-Lang abspeichern	193
3.9. Schiebe- und Rotierbefehle	131	SFX - Operand OP1 im IEEE-Format abspeichern	194
3.10. Sprungbefehle	147	SFA - Operand OF Fill IEEE-Format abspectient	194
3.11 Sonstines	163		

	4.10. Hilfsroutinen	
195	FCOP - Speicherbereich kopieren	226
195	FSMB - Speicherbereich mit 1 Byte-Wert initialisieren	227
195	FSMW - Speicherbereich mit 2 Byte-Wert initialisieren	228
196	FCLR - Speicherbereich löschen	229
196	MCMP - OP1 und OP2 vergleichen	230
196	MHIL - Begrenzen auf Obergrenze;	
	Wenn OP1 > OP2, dann OP2 ⇒ OP1	231
197	MLOL - Begrenzen auf Untergrenze;	
197	Wenn OP1 < OP2, dann OP2 ⇒ OP1	232
198		
	5. Sonderbefehle	233
199	5.1. Arithmetikprozessor (nur CP80)	233
200	5.1.1. Operanden	233
202		234
204	• • • • • • • • • • • • • • • • • • • •	235
206	5.1.4. Befehlssatz des 68881 Arithmetikprozessors	236
208	5.1.5. Konstanten	238
210		241
211	3	24′
		242
212	5.2.3. SC - Schnittstellenstatus anfordern	243
	5.2.4. SF - Schnittstellenfunktionen	244
214	5.2.5. Unterschiedliche Behandlung der Handshake-Leitungen	255
216		
217		259
218		259
220		260
222		261
224	•	261
	6.5. Stichwortverzeichnis	262
	195 196 196 196 197 197 198 199 200 202 204 206 208 210 211 212 214 216 217 218 220 222	FCOP - Speicherbereich kopieren FSMB - Speicherbereich mit 1 Byte-Wert initialisieren FSMW - Speicherbereich mit 2 Byte-Wert initialisieren FSMW - Speicherbereich mit 2 Byte-Wert initialisieren FCLR - Speicherbereich löschen MCMP - OP1 und OP2 vergleichen MHIL - Begrenzen auf Obergrenze; Wenn OP1 > OP2, dann OP2 ⇒ OP1 MLOL - Begrenzen auf Untergrenze; Wenn OP1 < OP2, dann OP2 ⇒ OP1 S. Sonderbefehle S.1. Arithmetikprozessor (nur CP80) S.1.1. Operanden S.1.2. Aufruf von APU-Befehlen S.1.3. Zahlenformate S.1.4. Befehlssatz des 68881 Arithmetikprozessors S.1.5. Konstanten S.2. Schnittstellenbefehle (nur CP80, PP60 und NTCP6#) S.2.1. SOB - Zeichen ausgeben S.2.2. SIB - Zeichen einlesen S.2.3. SC - Schnittstellenstatus anfordern S.2.4. SF - Schnittstellenfunktionen S.2.5. Unterschiedliche Behandlung der Handshake-Leitungen 6. Anhang 6. Anhang 6. Anhang 6. Alphabetische Übersicht der B&R Mnemonics 6. Alphabetische Übersicht der Motorola Mnemonics 6. Alphabetische Übersicht der Mathematik-Routinen

Register

Arithmetikprozessor (nur CP80) Schnittstellenbefehle (nur CP80, PP60 und NTCP6#)	Sonderbefehle
Hilfsroutinen	
Binär <=> Physikalisch-Wandlung	
Zahlenumwandlungen	
Schnelle Multiplikationen	
Operandenzwischenspeicher	
Operand abspeichern	
Operanden laden	
Operandenmanipulation	
inen Grundrechenarten	Mathematik-Routinen
Sonstiges	
Sprungbefehle	
Schiebe- und Rotierbefehle	
Inkrement- und Dekrementbefehle	
Vergleichs- und Testbefehle	
Arithmetische Operationen	
Logische Verknüpfungen	
Stapeloperationen	
Datenaustausch zwischen Registern	
Speicherbefehle	
Ladebefehle	AWL-Befehle
Adressierungsarten	
Erklärung der Syntax	

1. ERKLÄRUNG DER SYNTAX

1.1. CPU-TYP

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

Prozessor	CPU-TYP	Modul
6303	Α	CP40, CP41, NTCP33, NTCP34, PSCP35, PP40,
	_	CP30, CP31, CP32
6809	В	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:

Mot	orola			Fun	ktion												•	CF	٦٢	ΙŢ	У	o /	۹ /	6	30	3
B&F	?																									
Kur	z																									
Adressierungsarten / Opcode						Γ			Δ	dr	ام	۷۷	or	۱۸/	ah	ما	n					Ι	P	G10	000	Γ
Aui	63316	ungs	ai teii i	Орсс	,ue	L				uı	٠.	3 V	٠.	***	an								Ρ	G-F	S	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	S	Т	#	Р	R	Х	Υ	D	U		В	G	St	atı	ISI	eg	ist	er
					E A M F Z # P C I Y D U ! B G H I								Ν	z	٧	С										
						Γ	Г	Г				Г	Г								Г			П		

Mote	orola			Fun	ktion												1	CI	2 U	ΙT	УI	o E	3 /	68	30	9
B&R	1																									
Kurz	2																									
Adressierungsarten / Opcode Adreßvorwahlen											Ι	С	Pε	30												
_ Au	63316	ungs	arterr	Орсс	Jue				_	uı	٠.	3 V	٠.	***	anı								P	CE	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	н	I	N	Z	٧	С
						Γ	Г	Г				Г	Г					Г								П

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-

 $\label{eq:model} \mbox{Mnemonic (im B\&R Modus) bzw. die MOTOROLA®-Mnemonic (im MIXM Modus) eingesetzt.}$

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)	r _s	8-Bit Register (ERA, ERB, SR, DP)
ERB	Ergebnisregister B (8 Bit)	r ₁₆	16-Bit Register (ERD, R, Y SP!, SPU)
ERD	Ergebnisregister D (16 Bit; ERA = MSB, ERB = LSB)	l ^{'°}	IRQ Mask
SR	Statusregister (8 Bit)	^	Logische UND Verknüpfung
DP	Register für direkte Adressierung (8 Bit)	V	Logische ODER Verknüpfung
	(Direct Page Register)	\oplus	Logische Exklusiv-Oder (EXOR) Verknüpfung
R	Indexregister X (16 Bit)	MSB	Höherwertiges Byte eines 2 Byte Wertes
Υ	Indexregister Y (16 Bit)	LSB	Niederwertiges Byte eines 2 Byte Wertes
SP!	System-Stapelzeiger (16 Bit)	EA	Effektive Adresse
SPU	Anwender- oder User-Stapelzeiger (16 Bit)	IMM	Immediate Wert (Konstante)
PC	Programmzähler (Program counter)	dn	Datenbit n eines Registers oder einer
M	Adresse einer Speicherstelle (Memory)		Speicherstelle
(M)	Inhalt der Speicherstelle mit der Adresse M (8 Bit)		•

1.3.4. Betriebsart

Das Symbol ": 1" zeigt an, daß ein Befehl in dieser Betriebsart des PROgrammierSYStemes verfügbar ist.

Beispiel:



(M:M+1) Inhalt der beiden Speicherstellen mit den Adressen M und M+1 (16 Bit)

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:

IMPL. Implizit

Der Befehl braucht keine zusätzlichen Parameter. Der Opcode enthält bereits alle nötigen Adreßinformationen zur Ausführung des Befehles.

DIR.1) Direkt

Zusätzlich zum Befehl wird das niederwertige Byte der Adresse angegeben. Das höherwertige Byte wird durch das DP-Register (Direct Page Register) definiert

EXT. Absolut (Indirekt Absolut²⁾)

Zusätzlich zum Befehl wird die ganze Adresse (2 Byte) eingegeben.

IMMED. Unmittelbar (Konstante)

Der Wert (1 oder 2 Byte), den der Befehl verarbeitet, wird unmittelbar mit dem Befehl eingegeben.

IND. Indiziert (Indirekt Indiziert 2)

Bei der indizierten Adressierung wird die effektive Adresse aus dem Offset, der zusätzlich zum Befehl eingegeben wird, und einem der Indexregister (I, Y¹), SPI¹). SPU¹) berechnet.

REL. Relativ

Die relative Adressierung wird bei bedingten Sprüngen verwendet. Die Zieladresse berechnet sich aus dem aktuellen PC (Program counter) und dem angegebenen 1 oder 2¹⁰ Byte Offset.

- nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar.
- 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	
O F S T # P R X Y U ! D G B	E A M F Z # P C I Y U ! D G B	Eingang Ausgang 1-Bit Speicherstelle Freigabe einer Zeit Signal einer abgelaufenen Zeit Konstante Peripherie-Adresse 8-Bit Speicherstelle Indexregister R (X) Indexregister Y Anwender Stapelzeiger System Stapelzeiger Direkte Adressierung Globales RAM im PP60 (erweiterter Koppel-RAM Bereich) Blockspeicher im PP60)
_	_	

Weiters werden noch folgende Symbole verwendet:

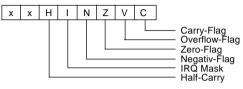
- verfügbar in den angegebenen Betriebsarten
- verfügbar ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80

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:



Half-Carry:

Bit 6 und 7:

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 Rechenre-

gister 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.

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.

immer auf log. 1 gesetzt.

Im 6809 sollten diese Bits nicht vom Anwender verändert werden!

Folgende Symbole werden verwendet: Flag wird nicht beeinflußt.

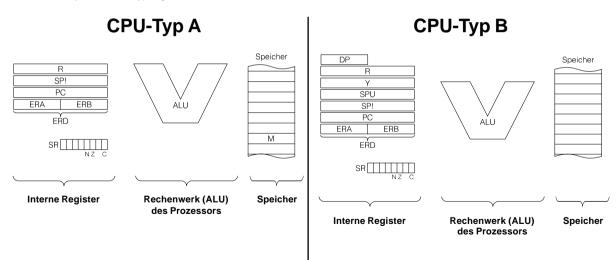
- Flag wird entsprechend der Operation verändert.
- ▲ Flag wird auf log. 1 gesetzt.
- ▼ Flag wird auf log. 0 gesetzt.
- X Zustand des Flags ist nicht definiert.

1.4. Genauere Beschreibung

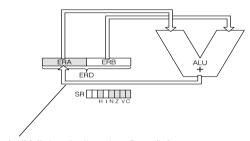
Nach der Kurzinformation über ieden Befehl folgt:

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

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

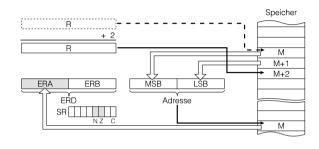


Register, Speicher oder Statusregisterbits, die bei der Ausführung des Befehles verändert werden könnten, werden **grau hinterlegt** gekennzeichnet.



Breite Pfeile beschreiben einen Datenfluß.

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

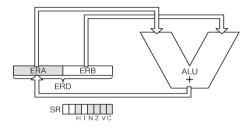


Strichlierte Register und Zeiger stellen den Zustand vor Ausführung des Befehles dar.

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 ¹⁾	EXG ¹⁾	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 ²⁾	CLA ²⁾	LSRD	SRD	SEI	SEI
CLRB ²⁾	CLB ²⁾	MUL	A*B	TAB	MAB
COMA ²⁾	COA ²⁾	NOP	NOP	TBA	MBA
COMB ²⁾	COB ²⁾	PSHA	ANS	TFR ¹⁾	TFR ¹⁾
DAA	DK	PSHB	BNS	TPA	MCA
DECA	DA	PSHX	RNS	TSX	MSR
DECB	DB	PULA	AVS	TXS	MRS
DES	DS	PULB	BVS	XGDX	DXR

nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar

nur ab PROgrammierSYStem Version 5.00 in den Betriebsarten PG-PC oder PC 80 verfügbar

2.2. DIR. - DIREKTE ADRESSIERUNG

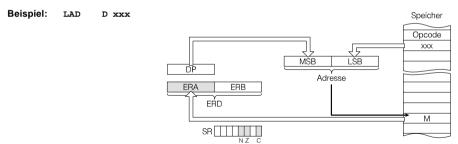
Diese Adressierungsart ist vergleichbar mit der indizierten Adressierung. Die Adresse, auf die ein Befehl zugreift, setzt sich aus zwei Teilen zusammen:



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.



Befehlsübersicht / Direkte Adressierung:

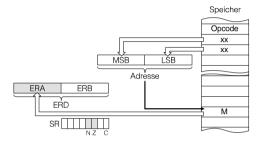
Diese Adressierungsart ist nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar.

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



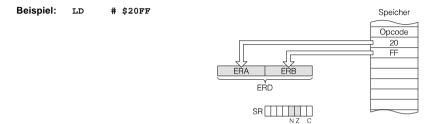
Befehlsübersicht / Absolute Adressierung

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	В	LDD	LD	STD	=D
BITB	BB	LDS	LS	STS	=S
CLR	CLR	LDX	LR	STX	=R
CMPA	CMP	LDY ¹⁾	LY ¹⁾	STY ¹⁾	=Y1)
CMPB	VB	LSR	SR	SUBA	-
COM	K	ORAA	OD	SUBB	-B
CPX	VR	ORAB	ОВ	SUBD	-D
CPY ¹⁾	VY ¹⁾	PRS	PRS		

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.



Befehlsübersicht / Unmittelbare Adressierung

MOTOROLA	B&R	MOTOROLA	B&R	MOTOROLA	B&R
ADCA	ADD	CPX#	VRK	LDY#1)	LYK¹)
ADCB ADDA	++B +	CPY#¹) EIM²)	VYK¹) EIM²)	LDYL ¹⁾ OIM ²⁾	LYL ¹⁾ OIM ²⁾
ADDB	+В	EORA	EXO	ORAA	OD
ADDD	+D	EORB	EB	ORAB	OB
AIM ²⁾	AIM ²⁾	LDAA	LAD	SBCA	SUB
ANDA	UND	LDAB	LB	SBCB	B
ANDB	UB	LDD	LD	SUBA	-
BITA	В	LDK ³⁾	LDK ³⁾	SUBB	-B
BITB	BB	LDX#	LRK	SUBD	-D
CMPA	CMP	LDXL ³⁾	LDL ³⁾	TIM ²⁾	TIM ²⁾
CMPB	VB				

¹⁾ nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar

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

³⁾ 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

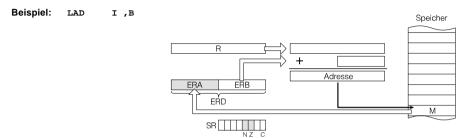
Die Offsetwerte werden in der AWL-Eingabezeile unmittelbar nach dem Befehl und der Adreßvorwahl eingegeben.

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.

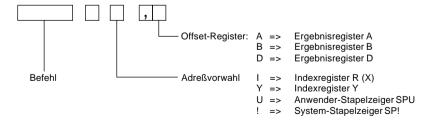
Offsetwert	-16 bis +15	-128 bis -17 +16 bis +127	-32768 bis -129 +128 bis +32767
Befehlslänge	2 + 0 = 2 Byte	2 + 1 = 3 Byte	2 + 2 = 4 Byte

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.

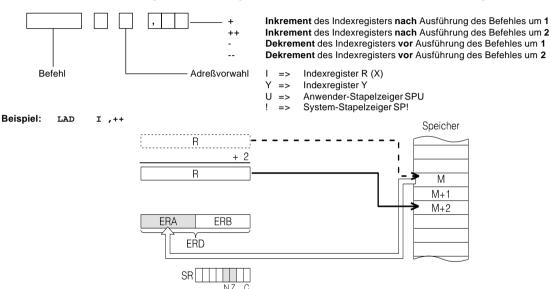


Für die indizierte Adressierung mit Registeroffset gibt es folgende Möglichkeiten:



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.



Befehlsübersicht/Indizierte Adressierung

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

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 ¹⁾
NUNZ	LANG

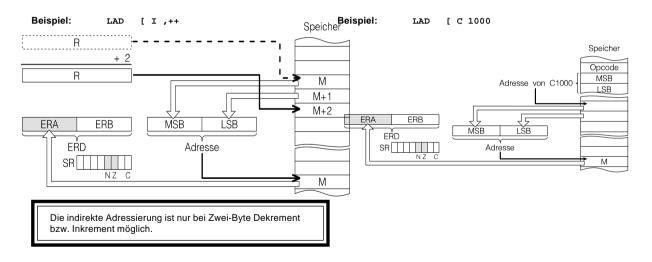
MOTOROLA	B&R	MOTOROLA	B&R
BCC	JC0	BCCL	JC0L
BEQ	SP0	BEQL	SP0L
BHI	SP>	BHIL	SP>L
BCS	SP<	BCSL	SP <l< th=""></l<>
BLS	J<=	BLSL	J<=L
BMI	J-	BMIL	J-L
BNE	SN0	BNEL	SN0L
BPL	J+	BPLL	J+L

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".



Befehlsübersicht / Indirekte Adressierung

MOTOROLA	B&R	MOTOROLA	B&R	MOTOROLA	B&R
ADCA ADCB ADDA ADDB ADDD ANDA ANDB ASL BITA BITB CLR CMPA CMPB	ADD ++B + +B +D UND UB SL B BB CLR CMP VB	DEC EORA EORB INC JMP JSR LDAA LDAB LDD LDS LDX LDY¹¹¹ LEA!¹¹²²	DEC EXO EB INC SPI SPU LAD LB LD LS LR LY'1 LE! ^{1) 2)}	LSR ORAA ORAB ROL ROR SBCA SBCB STAA STAB STD STS STX STY')	SR OD OB SLI SRE SUBB = B = D = S = R = Y ¹
COM CPX CPY ¹⁾	K VR VY¹)	LEAU ^{1) 2)} LEAX ^{1) 2)} LEAY ^{1) 2)}	LEU ^{1) 2)} LER ^{1) 2)} LEY ^{1) 2)}	SUBA SUBB SUBD	- -B -D
CPY	V Y ''	LEAT '/-/	LE Y '/ =/	2000	-ט

nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar

²⁾ 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#1)	VYK ¹⁾	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	В	LDD	LD	SET	SET
BITB	BB	LDK	LDK	STAA	=
CLR	CLR	LDX#	LRK	STAB	=B
CMPA	CMP	LDY'1)	LYK ¹⁾	STD	=D
CMPB	VB	LSR	SR	SUBA	-
COM	K			SUBB	-B

nur ab PROgrammierSYStem Version 5.00 in Betriebsart PC 80 verfügbar

3. AWL-BEFEHLE

3.1. LADEBEFEHLE

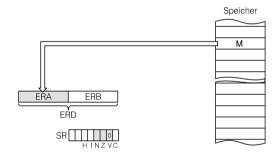
In diesem Abschnitt werden alle Befehle beschrieben, die Daten (1 oder 2 Byte) von einer angegebenen Speicherstelle in ein Register laden.

Motorola	B&R		Betri	ebsart	
Wiotorola	Dak	PG1000	PG-PC	CP 80	PC 80
LDAA	LAD	0	0	0	0
LDAB	LB	0	0	0	0
LDD	LD	0	0	0	
LDK	LDK		0		0
LDL	LDL		0		0
LDX	LR	0	0	0	0
LDX#	LRK	0	0	0	0
LDXL	LRL	0	0	0	0
LDY	LY				0
LDY#	LYK				0
LDYL	LYL				0
LDS	LS	0	0	0	0
LEA!	LE!				0
LEAU	LEU				0
LEAX	LER			0	0
LEAY	LEY				0

Mot	orola	LDA	AΑ	Fun	ktion									(CP	U T	Гуј	о А	//	630	3	П	Moto	orola	LDA	AΑ	Fun	ktion										CPU Typ B / 6809							
B&F	₹	LAD		(M)	⇒ ER	A															7	Г	B&R		LAI)	(M)	⇒ ER	A																
Kur	z	L				lo ro											1	Г	Kurz	:	L																								
Ad	ressie	rungs	arten /	Орсо	de			4	١dr	eßv	or/	wa	hle	n			0	-	_	1000 -PC	_	Г	Adr	essie	rungs	arten	/ Opc	ode	Γ		4	١dr	eß	vo	rw	ahl	er	1		0)	-	80	7	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	FS	т	# P	R	X,	/ D	U	! !	в С	St	_	_	gist	_	IN	MPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	FS	т	# F	R	x	Υ	DΙ	. l	В	G S	tatı	_	giste	r	
	3/2	4/3	2/2	4/2		Ε	A I	ИF	z	# P	С	1	/ D	U	! !	в G	Н	I	N Z	. V	С	Г		4/2	5/3	2/2	4+/2+		Ε	ΑI	ИF	Z	# F	c	1	ΥI	DΙ	J!	В	G F	I I	N Z	z v	С	
	96	В6	86	A6		Э	0)	0	o c	ं	0		П	T	T	0	0	• •	•)			96	B6	86	A6		ာ	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.

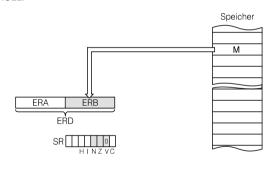


Mot	orola	LDA	λB	Fun	ktion												(CI	9٤	J T	УI	0 /	4 /	6	30	3
B&F	₹	LB		(M)	⇒ ERE	3																				
Kur	z																									
۸4	roccio	runae	arten	/ Once	,do	Г			^	4,	٠,٠١	2 . ,	or	١	ah	ı	n				0	1	P	G1 (000	,
Au	CSSIC	ungs	arterr	Opce	ue				^	uı	CI	3 V	UI.	vv.	aıı	116	••				0	•	Р	G-F	'n	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
	3/2	4/3	2/2	4/2		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С
	D6	F6	C6	E6		0	0	0	0	0	0	0	0	0							0	0	•	•	•	0

Mot	orola	LDA	В	Fun	ktion												1	CI	٧,	ΙŢ	y	o E	3 /	6	В0	9
B&F	t	LB		(M)	⇒ ERI	3																				
Kur	z																									
Δd	rossio	runae	arten	/ Once	nde	Γ			Δ	dr	ام:	ŝν	۸r	۱۸/	ah	ما	n				O	1	С	P	30	
Au	63316	ungs	arterr	Opce	Jue	L			_	uı	٠.	3 V	٠.	***	u		••				С	•	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	ısr	eg	ist	er
	4/2	5/3	2/2	4+/2+		Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
	D6	F6	C6	E6		Э	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.

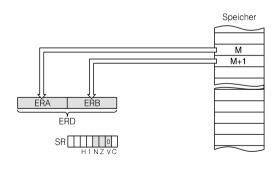


Mot	orola)	Fun	ktion												1	CF	?(Ţ	УI	р	Α/	6	30	3
B&F	₹	LD		(M:N	VI+1) ⊏	۱ (ER	D																		٦
Kur	z																									
۸4	roccio	runae	arton	/ Once	ode Adreßvorwahlen O PG1000														П							
Kui	CSSIC	ungs	arterr	Орсс	Jue				^	uı	CI	3 V	UI.	w	aii	iie					О	•	Ρ	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	usi	reg	ist	er
	4/2	5/3	3/3	5/2		Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Ι	ı	N	Z	٧	С
	DC	FC	cc	EC		Э	0	0	0	0	0	0	0	0			П		П		0	0	•	•	▼	0

Mot	orola	LDE)	Fun	ktion													CI	٦,	ĮΤ	У	рΙ	3 /	6	В0	9
B&F	t	LD		(M:N	/l+1) =	> I	ΞR	D																		
Kur	z																									
Δd	rossio	runas	arton	/ Once	nde	Г			Δ	dr	eßvorwahlen										С	1	С	P	30	
	63316	unga	arterr	Орсс	Jue				_	uı	٠.	3 V	٠.	an						С)	Р	C	30		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
	5/2	6/3	3/3	5+/2+		Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
	DC	FC	CC	EC		0	0	0	0	0	0	ာ	0	0	•	•	•	•	•	•	ာ	0	•	•	•	0
	B&F Kurz Adı	IMPL. DIR.	B&R LD Kurz Adressierungs	B&R LD	B&R LD (M:N Kurz Adressierungsarten / Opco IMPL. DIR. EXT. IMMED. IND. 5/2 6/3 3/3 5+/2+	B&R LD (M:M+1) C	B&R LD (M:M+1) □ B	B&R LD (M:M+1) ⇒ ER Kurz Adressierungsarten / Opcode IMPL. DIR. EXT. IMMED. IND. REL. I 0 5/2 6/3 3/3 5+/2+ E A	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode IMPL. DIR. EXT. IMMED. IND. REL. I 0 F 5/2 6/3 3/3 5+/2+ E A M	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode A IMPL. DIR. EXT. IMMED. IND. REL. I 0 F S S 5/2 6/3 3/3 5+/2+ E A M F B A M F	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adr IMPL. DIR. EXT. IMMED. IND. REL. I 0 F S T S T S S T S T S T S S T S	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adrel IMPL. DIR. EXT. IMMED. IND. REL. I OF S T W S T W 5/2 6/3 3/3 5+/2+ E A M F Z W S T W </th <th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßv IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P 5 Z # P 5/2 6/3 3/3 5+2+ E A M F Z # P 2 # P</th> <th>B&R LD (M:M+1) □ ERD Kurz Adressierungsarten / Opcode Adreßvor IMPL. DIR. EXT. IMMED. IND. REL. I OFS TFSTFP FSTFP PR 5/2 6/3 3/3 5+/2+ EA MFZ FSTFP PR</th> <th> B&R LD (M:M+1) □ ERD </th> <th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwah IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y 5/2 6/3 3/3 5+/2+ E A M F Z # P C I Y</th> <th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahle IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D I Y D<!--</th--><th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D U IV D U 5/2 6/3 3/3 54/2+ E A M F Z # P C I Y D U IV D U</th><th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I V D U I 5/2 6/3 3/3 5+/2+ E A M F Z # P C I Y D U I V D U I</th><th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D U I B S / 2 8 / 3 3/3 5+/2+ E A M F Z # P C I Y D U I B</th><th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I B G 5/2 6/3 3/3 5+/2+ E A M F Z # P C I Y D U I B G</th><th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adrefvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D U I B G SI I V D U I B G SI S G SI 5/2 6/3 3/3 5+/2+ E A M F Z # P C I V D U I B G H S G SI S G SI</th><th>B&R LD (M:M+1) □ ERD Kurz Adressierungsarten / Opcode Adresvorwahlen O IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I B G State 5/2 6/3 3/3 54/2+ E A M F Z # P C I Y D U I B G H I</th><th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen O CO P IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I B G Statusr S12 6/3 3/3 5+/2+ E A M F Z # P C I Y D U I B G H I N</th><th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen O PC IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D U I B G Statusreg Statusreg 5/2 6/3 3/3 54/2+ E A M F Z # P C I Y D U I B G H I N Z</th><th>B&R LD (M:M+1) □ ERD Kurz Adressierungsarten / Opcode Adrefsvorwahlen □ CP 80 IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I B G Statusregist Statusregist 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</th></th>	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßv IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P 5 Z # P 5/2 6/3 3/3 5+2+ E A M F Z # P 2 # P	B&R LD (M:M+1) □ ERD Kurz Adressierungsarten / Opcode Adreßvor IMPL. DIR. EXT. IMMED. IND. REL. I OFS TFSTFP FSTFP PR 5/2 6/3 3/3 5+/2+ EA MFZ FSTFP PR	B&R LD (M:M+1) □ ERD	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwah IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y 5/2 6/3 3/3 5+/2+ E A M F Z # P C I Y	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahle IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D I Y D </th <th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D U IV D U 5/2 6/3 3/3 54/2+ E A M F Z # P C I Y D U IV D U</th> <th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I V D U I 5/2 6/3 3/3 5+/2+ E A M F Z # P C I Y D U I V D U I</th> <th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D U I B S / 2 8 / 3 3/3 5+/2+ E A M F Z # P C I Y D U I B</th> <th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I B G 5/2 6/3 3/3 5+/2+ E A M F Z # P C I Y D U I B G</th> <th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adrefvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D U I B G SI I V D U I B G SI S G SI 5/2 6/3 3/3 5+/2+ E A M F Z # P C I V D U I B G H S G SI S G SI</th> <th>B&R LD (M:M+1) □ ERD Kurz Adressierungsarten / Opcode Adresvorwahlen O IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I B G State 5/2 6/3 3/3 54/2+ E A M F Z # P C I Y D U I B G H I</th> <th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen O CO P IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I B G Statusr S12 6/3 3/3 5+/2+ E A M F Z # P C I Y D U I B G H I N</th> <th>B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen O PC IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D U I B G Statusreg Statusreg 5/2 6/3 3/3 54/2+ E A M F Z # P C I Y D U I B G H I N Z</th> <th>B&R LD (M:M+1) □ ERD Kurz Adressierungsarten / Opcode Adrefsvorwahlen □ CP 80 IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I B G Statusregist Statusregist 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</th>	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D U IV D U 5/2 6/3 3/3 54/2+ E A M F Z # P C I Y D U IV D U	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I V D U I 5/2 6/3 3/3 5+/2+ E A M F Z # P C I Y D U I V D U I	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D U I B S / 2 8 / 3 3/3 5+/2+ E A M F Z # P C I Y D U I B	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I B G 5/2 6/3 3/3 5+/2+ E A M F Z # P C I Y D U I B G	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adrefvorwahlen IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D U I B G SI I V D U I B G SI S G SI 5/2 6/3 3/3 5+/2+ E A M F Z # P C I V D U I B G H S G SI S G SI	B&R LD (M:M+1) □ ERD Kurz Adressierungsarten / Opcode Adresvorwahlen O IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I B G State 5/2 6/3 3/3 54/2+ E A M F Z # P C I Y D U I B G H I	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen O CO P IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I B G Statusr S12 6/3 3/3 5+/2+ E A M F Z # P C I Y D U I B G H I N	B&R LD (M:M+1) ⇒ ERD Kurz Adressierungsarten / Opcode Adreßvorwahlen O PC IMPL. DIR. EXT. IMMED. IND. REL. I O F S T # P R X Y D U I B G Statusreg Statusreg 5/2 6/3 3/3 54/2+ E A M F Z # P C I Y D U I B G H I N Z	B&R LD (M:M+1) □ ERD Kurz Adressierungsarten / Opcode Adrefsvorwahlen □ CP 80 IMPL. DIR. EXT. IMMED. IND. REL. I OF S T # P R X Y D U I B G Statusregist Statusregist 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

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.



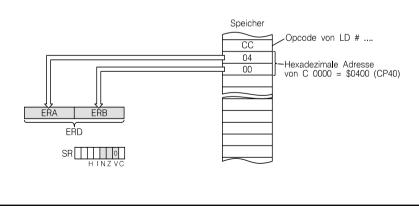
Mot	orola	LD	(Fun	ktion												-	CI	2	J T	у	р	A A	6	30	3
B&F	₹	LDF	(Mఐ	ERD																					
Kur	z																									
۸4	roccio	runae	arten	/ Once	,do	Γ			^	4	٠,٠١	21/	or	١	ak	ı	n				Г	T	P	G1(000)
Au	CSSIC	ungs	arterr	Орсс	ue				^	uı	CI	3 V	UI.	vv.	aı		•••				C	,	Р	G-F	S	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	us	reg	ist	er
			3/3			Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
			cc			Э	0	0	0	0	Г	0	0	0	Г		П		П		0	0	•	•	▼	0

Mot	orola	LDK	(Fun	ktion												1	CF	٦,	ΙŢ	y	o E	3 /	68	B0	9
B&F	₹	LDK		Mఐ	ERD																					
Kur	z																									
Δd	ressie	runas	arten	/ Once	nde	Г			Δ	dr	eí	۱۷	٥r	w	ah	le	n					Ι		P		
~~	. 000.0	ungo	ui toii	Орос	Juc				_	۵.	٠.	•	٠.	•••	u.,	٠.	••				С	1	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
			3/3			Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
			cc			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)



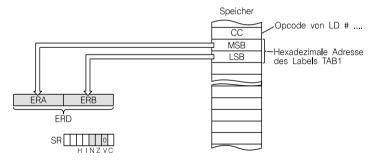
Mot	orola	LDL		Fun	ktion												1	CF	P	J T	y	р	۹ /	6	30	3
B&F	₹	LDL		Mఐ	ERD																					
Kur	z																									
۸.4	roccio	runae	arten	/ Once	do	Γ			^	4	٠,٠١	2.,	or	١	ah	ı	n					Τ	P	G1 (000	,
κu	CSSIC	ungs	arterr	Орсс	ue				_	uı			-								O	_		G-F		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
			3/3			Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	ī	N	z	٧	С
			CC			Г	Г	Г	Г	П	Г	Г	П	П			П			П	0	0	•	•	▼	0

		_																									
Mot	torola	LDL		Fun	ktion												(CF	P	JΤ	УI	o E	3 /	6	В0	9	
B&I	R	LDL		М⇨	ERD																						
Kur	z																										
Δd	urz Adressierungsarten / Opcode Adreßvorwah										le	n					Ι	_	Pδ	_							
		unge	ar torr	Орос	Juc	L			_	۵.	٠.	•	٠.	•••	۵.,						О	ı		C			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	Ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er	
			3/3			Ε	Α	М	F	Z	#	Р	С	-	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С	
			CC																		0	0	•	•	•	0	

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

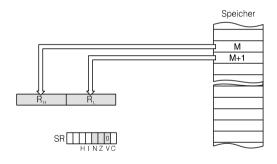


Anmerkung: Wird für den Label ein Tabellenname eingegeben, enthät das Ergebnisregister ERD nach der Ausführung des Befehles die Adresse des Tabellen-Headers. Das erste Datenbyte ist effektiv in der Speicherstelle mit der Adresse ERD + 15 gespeichert (siehe Bedienerhandbuch PROgrammierSYStem, Kapitel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

Mot	orola	LDX	(Fun	ktion												-	CI	9٤	ľ	У	ρ	A /	6	30	3
B&F	₹	LR		(M:N	/I+1) =	۱ (₹																			٦
Kur	z																									-
٨٨	roccio	runae	arten	Once	,do	Г			_	4,	٠,	2.,	or	\A/-	ah	10	n				0	₹ Т	P	G10	000	Л
Kui	CSSIC	ungs	arterr	Орсс	ue				^				-								0	ī		G-F		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	usi	reg	ist	er
	4/2	5/3	3/3	5/2		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	T	Ν	Z	٧	С
	DE	FE	CE	EE			П					ာ	0	ာ							0	0	•	•	•	ା

Mot	orola	LDX		Fun	ktion												1	CF	Pι	ΙŢ	УI	o E	3 /	6	80	9
B&F	t	LR		(M:N	/l+1) =	> F	₹																			
Kur	z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	el	ŝν	or	w	ah	le	n				О	_		P		
Au.	00010	ungo	ui toii ,	Орос	, , ,	L															О			C		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
	5/2	6/3	3/3	5+/2+		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
	9E	BE	8E	AE		Γ	Г	П	П			0	0	0	•		•	•	•	•	0	0	•	•	▼	0

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



WIOT	oroia	ᄓᄓ	.#	run	Ktion												- ['	U	- (, ,	УΙ	,	4 /	0	JU.	ગ
B&F	₹	LR		Mఐ	R																					٦
Kur	z																									١
٨٨	roccio	runae	arton	rten / Opcode Adreßvorwahlen O PG1000																						
Au	CSSIC	ungs	ai teii /	Орсс	ue				^	uı	CI	3 V	U.	w	aıı	116	••				О	•	Р	G-F	S	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	IS	eg	ist	er
			3/3			Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	N	z	٧	С
			CE			5	0	0	0	\circ	П	0	0	П			П		П	П	0	0	•	•	▼	\circ

Γ	Mot	orola	LDX	(#	Fun	ktion													CI	?(JΤ	УI	o E	3 /	68	30	9
l	B&F	₹	LRK	(M□	R																					
l	Kur	z																									
1	Δ.	roccio	runas	arten	/ Once	nde	Γ			Δ	di	ام:	ßν	۸r	۱۸/	ah	ما	n				О	1	С	Р8	30	_
l	ا م	163316	ungs	arterr	Орсс	Jue				_	u	٠.	3 4	٠.	***	u		••				О	•	Ρ	C 8	30	
1	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	e
1				3/3			Е	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	I	N	Z	٧	c
1				8E			ં	0	0	0	0		0	0			П		Г	•	•	0	0	•	•	•	C

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

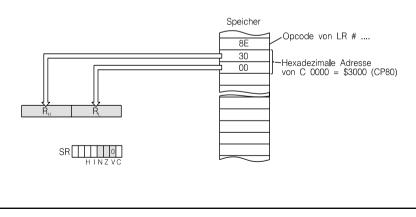
Eunktion

Motorolo I DV#

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.

CBII Tun A / 6202

Beispiel: LRK C 0000 (in einer CP80 \Rightarrow CPU Typ B / 6809)



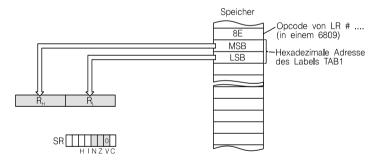
Mot	orola	LD	(L	Fun	ktion													CF	2 U	Ţ	УI	o /	4 /	6	30	3
B&F	₹	LRL	-	M⊏	R																					
Kur	z																									
Δdı	rossio	runas	arten	/ Once	nda	Γ			Δ	dr	ام:	۱۷	or	۱۸/	ah	ما	n				0	1	P	31 0	000	
Au	COOL	ungs	arterr	Орсс	Jue	L			^	u	٠.	3 V	٠.	***	u		••				О	•	Р	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
			3/3			Ε	Α	М	F	z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
			CF			Г	Г		Г	П	Г	Г		П	П	П	П			П	0	0	•	•	▼	0

Mot	orola	LD	(L	Fun	ktion													CI	٦,	JΤ	y	рE	3 /	6	80	9
B&F	ł	LRI	-	M⊏	R																					
Kur	z																									
Δd	ressie	runa	arten	/ Once	nde	Γ			Δ	dr	e í	۱۷	٥r	w	ah	le	n				O	_		P		
~~	000.0	ung	ou to ii	, opo	Juc	L			_	۵.	٠.	•	٠.	•••	u.,		••				С)	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	1	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
			3/3			Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
			8E			Г	П	Г	Г			Г	Г			Г		Г	П	П	0	0	•	•	▼	0

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

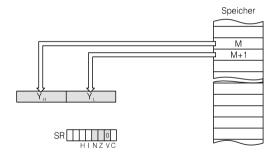


Anmerkung: Wird für den Label ein Tabellenname eingegeben, enthät das Indexregister R (X) nach der Ausführung des Befehles die Adresse des Tabellen-Headers. Das erste Datenbyte ist effektiv in der Speicherstelle mit der Adresse ERD + 15 gespeichert (siehe Bedienerhandbuch PROgrammierSYStem, Kapitel 7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

Mot	orola			Fun	ktion													CI	P	J 1	УΙ	р	4	6	30	3
B&F	₹																									٦
Kur	z																									-
٨٨	roccio	runae	arten	Once	,do	Γ			_	4,	eí	2.,		\A/-	ah	ı	n				Г	Τ	P	G10	000	Л
Au	CSSIC	ungs	arterr	Орсс	ue				^	uı	CI	3 V	U.	vv.	aıı		••				Г	Τ	Р	G-F	S	П
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	IS	reg	ist	er
						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
						Г															Г			П		П

T	Mote	orola	LDY	,	Fun	ktion												1	CI	2	JΤ	y	o E	3 /	6	80	9
1	B&R	ł	LY		(M:N	/l+1) ⊏	۰,	Y																			
l	Kurz	z																									
]	Δdı	rossio	runas	arten	/ Once	nde				Δ	dr	ام	ŝν	۸r	\A/:	ah	ما	n						С	P	30	
]	Aui	63316	unga	arterr	Орсс	Jue				_	uı	٠.	3 V	<u>.</u>	***							U			C		
7	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
1		6/3	7/4	4/4	6+/3+		Е	Α	М	F	Z	#	Р	С	-	Υ	D	U	!	В	G	Н	I	N	z	٧	С
]		10 9E	10 BE	10 8E	10 AE								•	•	•	•		•	•	•	•	0	0	•	•	•	0

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



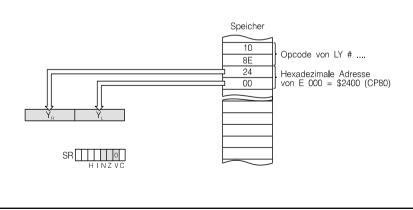
Mot	orola			Fun	ktion													CI	2	J T	у	р	A /	6	30	3
B&F	₹																									٦
Kur	z																									-
Δdı	roccio	runas	arten	Once	nda	Г			Δ	di	ام:	٠,	or	۱۸/	ah	ما	n					Τ	P	G10	000	╗
ď	CSSIC	rungs	arterr	Орсс	ue				^	u												T		G-F		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	usi	reg	ist	er
						Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	ı	Ν	Z	٧	С
																					Г					П

Mot	orola	LDY	"#	Fun	ktion												1	CI	٦,	J T	y	o E	3 /	6	80	9
B&F	₹	LYK		M□	Y																					
Kur	z																									
hΔ	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	اع:	ßν	٥r	w	ah	le	n					Ι		P		
7.4	. 000.0	· ugo	ui toii	Орос	Juc				_	۵.	٠.		٠.	•••	u						С	1	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
			4/4			Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	N	z	٧	С
			10 8E			•	•	•	•	•		•	•						•	•	0	0	•	•	▼	0

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)



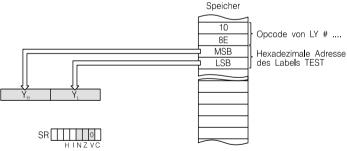
Mot	orola			Fun	ktion												1	CI	2	ן נ	ΊУΙ	р	4 /	6	30	3
B&F	₹																									٦
Kur	z																									
٨٨	roccio	runae	arten	/ Once	,do	Γ			^	4,	eí	2 . ,	^-	١	a k	ı۱۸	n				Г	Τ	P	G1 (000	П
Au	CSSIC	ungs	arterr	Орсс	ue				^	uı	CI	3 V	U.	w	aı		••				Г	Τ	Р	G-F	SC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
						Γ	Γ	Γ	Γ		Γ	Γ	Г			Γ					Γ	Γ	Γ			П

Mote	orola	LDY	Ľ	Fun	ktion												1	CI	٧,	JΤ	УI	o E	3 /	6	80	9
B&R	1	LYL		M□	Y																					
Kurz	z																									
Δdı	essie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	ŝν	or	w	ah	le	n					Ι	_	P		
Au.	000.0	ungo	ui toii i	Орос	Juc				_	۵.	٠.	••	٠.	•••	٠.,						0	1	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	ısr	eg	ist	er
			4/4			Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
			10 8E																		0	0	•	•	•	0

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

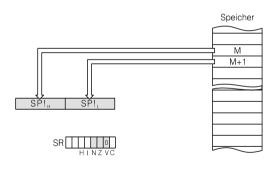


Anmerkung: Wird für den Label ein Tabellenname eingegeben, enthät das Indexregister Y nach der Ausführung des Befehles die Adresse des Tabellen-Headers. Das erste Datenbyte ist effektiv in der Speicherstelle mit der Adresse ERD + 15 gespeichert (siehe Bedienerhandbuch PROgrammierSYStem, Kapitel 7 Tabellen-Editor).

Mot	orola	LDS	3	Fun	ktion													CI	PU	J	y	۵,	A A	6	30	3
B&F	₹	LS		(M:N	/ 1+1) ⊏	> ;	SP	P!																		٦
Kur	z																									
٨٨	roccio	runae	arten	/ Once	,do	Г			^	4,	٠,٠١	21/	or		ah	ı	n				O	7	P	G1 (000	П
ť	CSSIC	ungs	arterr	Орсс	ue				^	uı	CI	3 V	U.	vv.	aii	iie					O	1	Р	G-F	c	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	us	eg	ist	er
	4/2	5/3	3/3	5/2		Е	Α	М	F	Z	#	Ρ	С	1	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
	9E	BE	8E	AE		Γ	Γ	Γ			Γ	0	0	0							0	0	•	•	•	0

Mot	orola	LDS	;	Fun	ktion												1	CI	٦,	JΤ	УI	p E	B /	6	В0	9
B&F	1	LS		(M:N	/ 1+1) ⊏	> ;	SP	!																		
Kur	z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	ßν	٥r	w	ah	le	n				О	<u> </u>	_	P		
, Au	000.0	· ugo	u. to	Орос	Juc	L			_	۵.	٠.	•	٠.	•••	u.,		••				О)	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
	6/3	7/4	4/4	6+/3+		Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
	10 DE	10 FE	10 CE	10 EE		Г	Г	Г	Г			0	0	0	•		•	•	Г	•	0	0	•	•	•	0

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



Mot	orola			Fun	ktion													CI	Pυ	JT	y	p /	۹ /	6	30)3	ı	М	ot	orola	LE/	\!
B&F	₹																										1	В	&F	ł	LE!	
Kur	z																											K	urz	z		
Adı	ressie	rungs	arten /	/ Opco	ode				Α	dr	eß	3v	or	wa	ah	le	n				E	Ŧ	P(_	00 PC	_	}	-	۱d	ressie	rungs	arte
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ISI	reç	gis	ter	7	IMP	L.	DIR.	EXT.	IMME
						Е	Α	М	F	z	#	Ρ	С	_	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С	1					4+/2
]					32

Mot	orola	LEA	.!	Fun	ktion												•	CF	٧U	ΙT	уı	o E	3 /	68	ВО	9
B&F	1	LE!		EA :	⇒ SP!																					
Kur	z																									
Δd	rossio	runae	arten	Once	nde	Γ			Δ	dr	ام:	21/	۸r	۱۸/	ah	ما	n					Τ	С	Pε	30	
Au	63316	ungs	arterr	Орсс	Jue				^	uı	٠.	3 V	٠.	***	411						0	1	Р	CE	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atι	ısr	eg	ist	er
			4+/2+			Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
			32			Γ	Π	Γ	Г				Г	•	•		•	•			0	0	0	0	0	0

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

Beispiele:

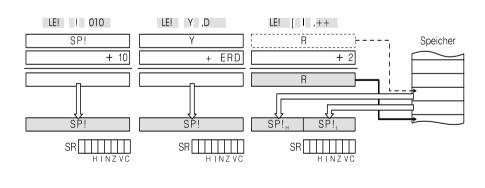
LE! LE!

! 010 Y,D

SP! + 10 ⇒ SP! Y + ERD ⇒ SP!

LE! [I ,++

(R:R+1) ⇒ SP!; R + 2 ⇒ R



Mot	orola			Fun	ktion													CF	٧	1	УI	0 /	4 /	63	30	3
B&F	ł																									٦
Kur	z																									-
Αdı	essie	runas	arten	Onco	ode	Γ			Δ	dı	eí	٧٧	or	w	ah	le	n					Ι	_	G10	_	_
,	000.0	90	u	Opoc		L			•		٠.	••	٠.	•••			••						Р	G-F	УC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	ısı	reg	ist	er
						Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	1	Ν	Z	٧	С

Mot	orola	LEA	'n	Fun	ktion												-	CI	? (ΙT	уı	o E	3 /	6	во	9
B&F	t	LEU	1	EA :	⇒ SPl	J																				
Kur	z																									
Δd	ressie	runae	arton	Once	nde	Γ			Δ	dr	ام:	ŝν	۸r	۱۸/	ah	ما	n					Τ	С	P	30	
, Au	63316	ungs	arterr	Орсс	Jue				_	u	٠.	3 V	٠.	***	anı						0	1	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	ısr	eg	ist	er
			4+/2+			Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
			33			Г		Γ					Γ	•	•		•	•			0	0	0	0	0	0

Die effektive Adresse EA wird aus der beim Befehl angegebenen indizierten Adressierung errechnet und im Anwender-Stapelzeiger SPU gespeichert.

Beispiele:

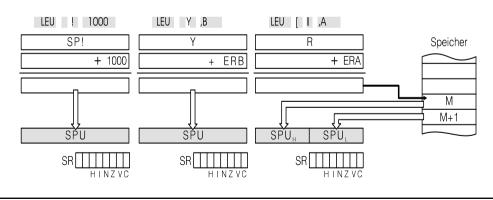
LEU ! 1000

SP! + 1000 ⇒ SPU

LEU Y ,B

Y + B ⇒ SPU

LEU [I ,A (R+ERA:R+ERA+1) ⇒ SPU



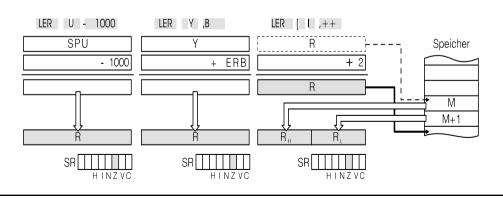
Mot	torola			Fun	ktion										С	Pι	JΤ	yp	Α	/ 6	303	3	Mot	orola	LE	AX	F	unl	ktior	1										(CP	U T	уp	В	/ 6	B09
B&F	R																					1	B&F	₹	LE	R	E	A	⇒R																	
Kur	z																					ı	Kur	z			1																			
Ad	ressie	runas	arten /	Opco	ode	Г		Α	dr	eß	vo	rw	ah	ler					_	_	000]	Ad	ressie	runa	sarten	/0	occ	ode	T			Α	dr	eß	vc	orv	val	nle	n			0		CP	
																				PG-	PC	1			9								-										0		PC 8	<i>i</i> 0
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	F	s	Т	# 6	R	x	Υ	DΙ	J !	В	G	Sta	itu	sre	giste	er	IMPL.	DIR.	EXT.	IMMED	. IN	D.	REL	. [ıo	F	s	Т	#	ΡĮ	R >	(Y	D	U	! E	3 G	Sta	tus	reg	iste
						E/	N N	F	Z	# 8	c	I	Υ	DΙ	J!	В	G	Н	1	ΝZ	۷	С				4+/2+					ΕA	M	F	Z	#	P (C	ΙY	D	U	! E	3 G	Н	I N	ΙZ	v c
							Τ		П		Τ				Τ			П			П	1				30					T	Τ					C	•		•	•		0)	•	0
																						_																								

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 ⇒ R

LER Y,B Y+B⇒R

LER [I ,++ $(R:R+1) \Rightarrow R; R+2 R$



Mot	orola			Fun	ktion													CI	?(J T	Ίу	p /	4 /	6	30	3
B&F	₹																									
Kur	z																									
Δdı	roccio	rungs	arten	Once	nda	Γ			Δ	di	reí	٠.		۱۸/	ak	ماد	n					Τ	P	G1(000	5
Au	163316	unga	arterr	Орсс	Jue	L				u	01	3 V	٠.	**	a.		••					Ι	Р	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	IS	reg	ist	er
						Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
						Γ	Γ	Γ	Γ			Γ	Γ	Γ	Γ	Γ	Π				Γ	Γ	Γ	Γ	Г	П

Mot	orola	LEA	Y	Fun	ktion												1	CI	2 U	T	уı) E	3 /	6	ВО	9
B&F	t	LEY	,	EA :	⇒Y																					
Kur	z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	٩v	٥r	w	ah	le	n					Ι	_	P		
,		90	u	Opor							٠.	•	٠.	•			•				О	L	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
			4+/2+			Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	1	Ν	Z	٧	С
			31			Г	Γ	Γ				Γ	Γ	•	•		•	•			0	0	0	•	0	0

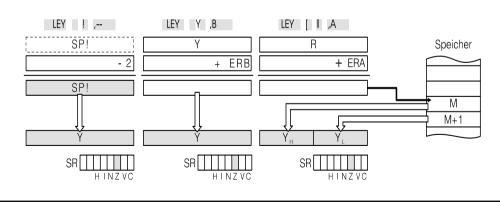
Die effektive Adresse EA wird aus der beim Befehl angegebenen indizierten Adressierung errechnet und im Indexregister Y gespeichert.

Beispiele:

$$SP! -2 \Rightarrow SP!; SP! \Rightarrow Y$$

 $Y + B \Rightarrow Y$

Y,B



3.2. SPEICHERBEFEHLE

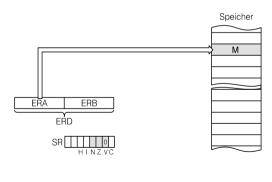
In diesem Abschnitt werden alle Befehle beschrieben, die Daten (1 oder 2 Byte) aus einem Register in einer angegebenen Speicherstelle speichern.

	505		Betr	iebsart	
Motorola	B&R	PG1000	PG-PC	CP 80	PC 80
STAA	=	0	0	0	0
STAB	=B	0	0	0	0
STD	=D	0	0	0	0
STX	=R	0	0	0	0
STY	=Y				0
STS	=S	0	0	0	0

Mot	orola	STA	A	Fun	ktion													CI	2	JΤ	УI	o /	4 /	6	30	3
B&F	₹	=		ERA	. ⇔ (M)																				
Kur	z	ı																								
Δdı	rossio	runae	arten	Onco	do	Г			Δ	dr	ام:	۱۷	or	۱۸/	ah	ما	n				U)	P	31 (000)
Au	COOL	ungs	arterr	Opce	ue				_	u			-								U				PC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	JSI	eg	jist	er
	3/2	4/3		4/2		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	N	z	٧	С
	97	B7		A7			0	0	0			0	0	0							0	0	•	•	•	0

Мо	torola	STA	A	Fun	ktion												1	CI	٧,	JΤ	УĮ	o E	3 /	6	В0	9
В&	R	=		ERA	\ ⇒ (M	I)																				
Ku	rz	ı																								ļ
	dressie	runas	arton	/ Once	nde	Γ			Δ	dr	ام:	ßν	۸r	١٨/:	ah	ما	n				0	1	С	P	30	
l ^`	ai 633i6	unga	arterr	Орсс	Jue				_	u	٠.	3 4	٠.	***	u						О	•	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
	4/2	5/3		4+/2+		Е	Α	М	F	Z	#	Р	С	Ι	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
	97	В7		A7			0	0	0			0	0	0	•	•	•	•		•	0	0	•	•	•	0

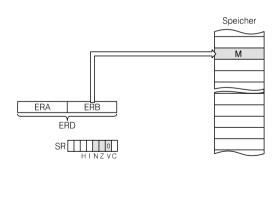
Der Inhalt des Ergebnisregisters ERA wird in der Speicherstelle mit der Adresse M gespeichert. ERA wird dabei nicht verändert. Ist die Zieladresse eine 1 Bit-Speicherstelle (A, M, F), wird nur das Datenbit 0 von ERA gespeichert.



Mot	orola	STA	В	Fun	ktion												-	CI	2	JΤ	y	р	4 /	6	30	3
B&F	₹	=B		ERE	3 ⇒ (M)																				
Kur	z																									
Δdı	rossio	runae	arten	Once	nda	Γ			Δ	dr	ام:	۱۷	or	۱۸/	ah	ما	n				U)	P	G1 (000	,
Au	COOL	ungs	arterr	Opce	ue					u	٠.	3 V	٠.	***	an		•••				\cup)	Ρ	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
	3/2	4/3		4/2		Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
	D7	F7		E7		L	0	0	0			0	0	0							0	0	•	•	•	0

Т	Mot	orola	STA	В	Fun	ktion												1	CF	٦(JΤ	y	рE	3 /	6	80	9
ſ	B&F	ł	=B		ERE	3 ⇔ (M)																				
Ī	Kur	z																									
ı	Δd	rossio	runae	arten	/ Once	nde	Γ			Δ	dr	ام:	ßν	۸r	۱۸/	ah	ما	n				O	7	С	P	30	
L	Au	CSSIC	ungs	aiteii	Opce	Jue				_	uı	CI	3 V	UI	w	all	16					O)	Р	С	30	
-[IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
Ī		4/2	5/3		4+/2+		Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
I		D7	F7		E7		Γ	0	0	0			0	0	0	•	•	•	•	Г	•	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.

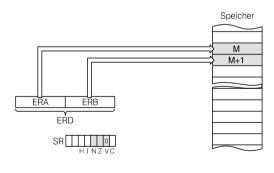


Mot	orola	STA	D	Fun	ktion												1	CF	?U	Ţ	УĮ	р	4 /	6	30	3
B&F	₹	=D		ERD) ⇔ (M	: N	/+	1)																		1
Kur	z																									-
۸.	Adressierungsarten / Opcode Adreßvorwahlen O PGPC															Л										
Au	CSSIC	ungs	arterr	Opce	ue				^	uı	CI	3 V	U.	vv.	aıı	116	••				O	7	Р	G-F	S	\neg
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
	4/2	5/3		5/2		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С
	DD	FD		ED			0	0	0			ာ	0	0							0	0	•	•	•	0

T	Mot	orola	STA	D	Fun	ktion												1	CF	٦,	ΙT	УI	o E	3 /	68	В0	9
Γ	B&R	l	=D		ERD) ⇔ (M	:N	/+	1)																		
	Kurz	z																									
Γ	Δdı	essie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	3v	٥r	w	ah	le	n				0	1		Pδ		
L	7,01	00010	ungo	ui toii ,	Орос	, , ,	L			_	۵.	٠.	•	٠.	•••	۵.,						О	ı	Ρ	CE	30	
Γ	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
Γ		5/2	6/3		5+/2+		Ε	Α	М	F	Z	#	Ρ	С	1	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
I		DD	FD		ED			0	0	0			0	0	0	•	•	•	•		•	0	0	•	•	•	0

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

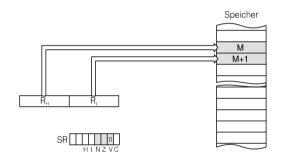
Ist die Zieladresse eine 1 Bit-Speicherstelle (A, M, F), wird das Datenbit 0 von ERA in der Speicherstelle M und das Datenbit 0 von ERB in der Speicherstelle M+1 gespeichert.



Mot	orola	STX	(Fun	ktion													CI	Pι	J 1	УΙ	р	A A	6	30	3
B&F	₹	=R		R⇒	(M:M	+1)																			1
Kur	z																									-
Δdı	roccio	runas	arten	Once	nda	Г			Δ	di	eí	٠,	۸r	۱۸/	ah	ما	n				C)	P	G10	000	╝
Kui	CSSIC	rungs	arterr	Орсс	ue				_	uı	CI	3 V	UI.	w	aii						C)		G-I		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	us	reg	ist	er
	4/2	5/3		5/2		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
	DF	FF		EF		Γ	Γ			Г		0	0	0							b	0	•	•	▼	0

Mot	orola	STX	(Fun	ktion												1	CI	٦,	JΤ	УI	рI	B /	6	80	9
B&F	t	=R		R ⇒	(M:M	+1)																			
Kur	z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	ßν	٥r	w	ah	le	n				0	1		P		
Au	00010	ungo	u. to	Орос	Juc	L			_	۵.	٠.	•	٠.	•••			••				О)	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
	5/2	6/3		5+/2+		Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ī	N	Z	٧	С
	9F	BF		AF		Г	Г	П	Г			0	0	0	•		•	•	П	•	0	0	•	•	▼	0

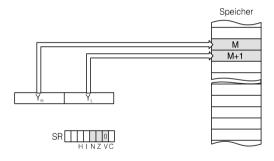
Der Inhalt des Indexregisters R (X) wird in den Speicherstellen mit den Adressen M und M+1 gespeichert. R (X) wird dabei nicht verändert.



Mot	orola			Fun	ktion												1	CI	2 U	ד נ	ΊУ	р	A /	6	30	3
B&F	t																									
Kur	z																									
Adı	essie	runas	arten	/ Opco	ode	Γ			Α	dı	reí	ßν	or	w	ah	le	n				Г	I		G10		_
		5 -				ı															ı		Ρ	G-F	ъC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	usi	reg	ist	er
						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
						Г	П		Г		П	Г		П	Г		П				Г	П	Г	П		

	Mot	orola	STY	,	Fun	ktion												1	CF	٧,	ΙT	УI	o E	3 /	6	80	9
Г	B&R	1	=Y		Y⇔	(M:M	+1)																			
	Kurz	z																									
Г	۸۵۰	neeln	runae	arten	/ Once	ndo.	Г			^	dr	۰,	ßν	۸r	\A/-	a h	ı۸	n					Т	С	P	30	
	Aui	CSSIC	ungs	arterr	Opce	Jue				^	uı	C.	3 V	U.	w	all	16					0	7	Р	C	30	
IN	MPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atι	ısr	eg	ist	er
		6/3	7/4		6+/3+		Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	1	N	Z	٧	С
		10 9F	10 BF		10 AF								0	0	0	•		•	•		•	0	0	•	•	•	0

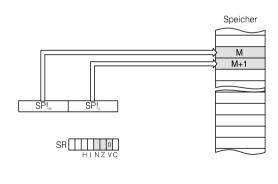
Der Inhalt des Indexregisters Y wird in den Speicherstellen mit den Adressen M und M+1 gespeichert. Y wird dabei nicht verändert.



Mot	orola	STS	3	Fun	ktion												1	CI	2	J	y	0 /	4 /	6	30	3
B&F	₹	=S		SP!	⇔ (M:	M	+1	1)																		
Kur	z																									
Δdı	ressie	runas	nda	Γ			Δ	di	ام:	21/	or	۱۸/	ah	ماد	n				U)	P	G1(000	П		
Au	63316	runga	arterr	Opce	ue					u	٠.	. ·	01	***	an		•••				\cup)	Ρ	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	JSI	reg	ist	er
	4/2	5/3		5/2		Ε	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
	9F	BF		AF		Γ	Γ		Γ	Γ	Γ	0	0	0		Γ					0	0	•	•	•	0

Mot	orola	STS	i	Fun	ktion												1	CF	Pυ	ĮΤ	Γу	рE	B /	6	80	9	
B&F	ł	=S		SP!	⇒ (M:	M	+1)																			
Kur	z																										
Δd	rossio	runas	arten	/ Once	nde	Γ			Δ	di	ام:	ŝν	۸r	\A/:	ah	ما	n				С	7	С	P	80		_
Au	63316	unga	arterr	Орсс	Jue					u	٠.	3 V	٠.	***	an						С			C			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	e	r
	6/3	7/4		6+/3+		Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	ı	N	z	٧	С	;
	10 DF	10 FF		10 EF		Γ			Г			0	0	0	•	П	•	•		•	ા	0	•	•	▼	С)

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

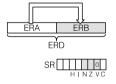
In diesem Abschnitt werden alle Befehle beschrieben, die Daten (1 oder 2 Byte) aus einem Register in einem anderen speichern oder zwischen zwei Registern austauschen.

Mark	B0B		Betri	iebsart	
Motorola	B&R	PG1000	PG-PC	CP 80	PC 80
TAB	MAB	0	0	0	0
TBA	MBA	0	0	0	0
TAP	MAC	0	0	0	0
TPA	MCA	0	0	0	0
TSX	MSR	0	0	0	0
TXS	MRS	0	0	0	0
XGDX	DXR	0	0	0	0
TFR	TFR				0
EXG	EXG				О

Motorola	TAB	Funktion	CPU Typ A / 6303
B&R	MAB	ERA ⇒ ERB	
Kurz			
			DC1000

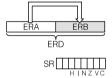
٨٨	roccio	runae	arten	Once	do				۸	4.	eí	2.,	^-		ah	ı	n				Ç	וי	P	ااذ	υu	,
Au	CSSIC	ungs	arterr	Opce	ue				~	uı	CI	3 V	U.	vv.	aii	116	•••				O	7	Р	G-F	Š	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	-	0	F	s	Т	#	Р	R	х	Υ	D	U	-	В	G	St	atı	ısı	eg	ist	er
1/1						Е	Α	М	F	Z	#	Р	С	1	Υ	D	U		В	G	Ι	1	Ν	Z	<	С
16																					0	0	•	•	•	0

Der Inhalt des Ergebnisregisters ERA wird in das Ergebnisregister ERB kopiert.



Mot	orola	TAE	3	Fun	ktion												1	CF	2 U	T	уı	o E	3 /	6	В0	9
B&F	₹	MAI	В	ERA	\ ⇔ EF	RB	;																			
Kur	z																									
Δd	ressie	runas	arton	/ Once	nde	Γ			Δ	dr	ام	٠.	۸r	۱۸/	ah	ما	n			П	0	7	С	P	30	
Α	. 000.0	ungo	ui toii i	Орос	Juc	L			_	۵.	٠.	•	٠.	•••	u.,		••				a	1	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
6/2						Е	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
0/2																										

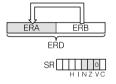
Der Inhalt des Ergebnisregisters ERA wird in das Ergebnisregister ERB kopiert.



Motorola	TBA	Funktion	CPU Typ A / 6303
B&R	MBA	ERB ⇒ ERA	
Kurz			

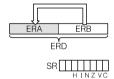
٨٨	roccio	runae	arten	Onco	do	ı			۸	۸,	eí	2.,	^-		ah	ı	n				O)	P	G10)00)	ı
Au	CSSIC	ungs	arterr	Орсс	ue				~	u	CI	3 V	U.	vv.	aıı	116	••				O	7	Р	G-F	Š		1
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	-	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er	
1/1						Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С	ı
17						Γ			Γ		Γ										0	0	•	•	•	0	1

Der Inhalt des Ergebnisregisters ERB wird in das Ergebnisregister ERA kopiert.



Mot	orola	TBA	١	Fun	ktion												1	CF	2 U	ΙT	·уІ	o E	3 /	68	309	9
B&F	~	MB	A	ERE	B ⇔ EF	RA																				
Kur	z																									
Δd	ressie	runae	arton	/ Once	nde	Γ			Δ	dr	ام:	ßν	۸r	\A/:	ah	ما	n				О	1	С	Р8	0	_
Au	633161	ungs	arterr	Opce	Jue	L			^	uı	٠.	3 4	٠.	***	u		••				О	•	P	8	0	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	egi	iste	er
6/2						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	N	z	٧	c
1F 98						Г	Γ	Γ	П	П	Г	Г	П		П	П	П	П	П	П	0	0	0	0	0	0

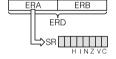
Der Inhalt des Ergebnisregisters ERB wird in das Ergebnisregister ERA kopiert.



Motorola	TAP	Funktion	CPU Typ A / 6303
B&R	MAC	ERA ⇒ SR	
Kurz			

۸.4	roccio	runae	arten	Once	do	ı			۸	۸,	٠~١	2.,	or		a k	ı	n				Ç		PC	310)00	,
Au	essie	ungs	arterr	Орсс	ue				~	u	CI	3 V	U	vv.	aı		••				O	7	P	G-F	'n	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
1/1						Ε	Α	М	F	Z	#	Ρ	С	1	Υ	D	U	!	В	G	Η	1	Ν	Z	٧	С
06						Γ	Γ					Γ			Г						•	•	•	•	•	•

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



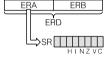
Hinweis:

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 Programmteile 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.

Mote	orola	TAP	•	Fun	ktion													CI	Pι	JΤ	y	рI	3 /	6	В0	9
B&R	₹	MA	С	ER.	\ ⇒ SF	?																				
Kurz	Z																									
Δdı	rossio	runas	arten	/ Once	nde	Γ			Δ	dr	ام:	R v	۸r	۱۸/	ah	ما	n				O)	С	P	30	
~~	63316	unga	arterr	Opci	Jue	L			_	u	٠.	3 4	٠.	**	u	10	••				С)	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	υ	!	В	G	St	atı	ısr	eg	ist	er
6/2						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
1F 8A						Γ		Γ	Γ			Г	Π								•	•	•	•	•	•

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 Programmteile des Anwenderprogrammes in keinem Fall durch einen Interrupt unterbrochen werden dürfen.

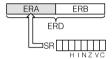
Bevor der Befehl END abgearbeitet wird, muß das Bit I unbedingt wieder auf log. 0 gesetzt werden.

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

Mot	orola	TPA	١	Fun	ktion												(CF	9٤	J T	уı	o /	4 /	6	30	3
B&F	₹	МС	A	SR s	⇒ ERA																					٦
Kurz	z																									
۸۵۰	roccio	runae	gsarten / Opcode Adreßvorwah		ı	n				O	7	P	G10	000	П											
Aui	CSSIC	ungs	ai teii /	Opco	ue				^	uı	CI	3 V	U.	vv.	211	116	••				O	7	Ρ	G-F	Š	П
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ISI	eg	ist	er
1/1						Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
07																					0	0	0	0	0	0

Mot	orola	TPA	١	Fun	ktion												1	CI	Pι	JΤ	у	рE	B /	6	80	9
B&F	ł	MC	A	SR	⇒ ER	4																				
Kur	z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dı	اع:	ßν	٥r	w	ah	le	n				C	1		P		
Au	00010	· ugo	ui toii	Орос	Juc	L			_	· · ·	٠.	•	٠.	•••			••				C)	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
6/2						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	ı	N	z	٧	С
1F A8						Г	Г	Г	Г	Г		Г	П	П	П			П	Г		ા	0	0	0	0	0

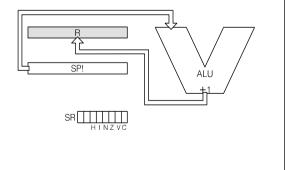
Der Inhalt des Statusregisters SR wird in das Ergebnisregister ERA kopiert.



Motorola	TSX	Funktion	CPU Typ A / 6303
B&R	MSR	SP! + 1 ⇒ R	
Kurz			

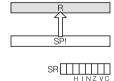
۸ ۵۰	roccio	runae	arten /	/ Onco	do				^	۸,	eí	2.,	^-		ah	ı	n				_	1	г	3 10	JUU	_
Aui	CSSIC	ungs	ai teii /	Орсс	ue				~	u	CI	3 V	U.	w	211	116	••				O	·	P	G-F	SC	٦
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	-	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	O	St	atı	ısr	eg	ist	er
1/1						Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
30																					0	0	0	0	0	ା

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.



Mot	orola	TSX	(Fun	ktion												1	CF	PU	T	УĮ) E	3 /	68	30	9
B&F	t	MSF	₹	SP!	⇒ R																					
Kur	z																									
Δ.	rossio	runae	arten	/ Once	nde	Γ			Δ	dr	ام	21/	۸r	\A/:	a h	ما	n				0	1	С	Р8	0	
^"	63316	ungs	arterr	Орсс	Jue	L			_	uı	٠.	3 .	٠.	***	a 1 1						0	1	P	8	0	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	egi	ist	er
6/2						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	N	z	٧	С
1F 41						Г	Γ	Γ	Г	П	Г	Γ	П					П			0	0	0	0	0	$\overline{\circ}$

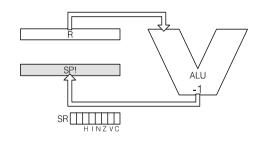
Das Indexregister R (X) wird mit der Adresse der Speicherstelle geladen, die den letzten gültigen Wert des Stapels enthält. Da im Prozessor 6809 der Stapelzeiger auf den letzten gültigen Wert im Stapel zeigt, muß keine Korrektur vorgenommen werden, wie es beim 6303 notwendig ist, d.h. SP! wird nach R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



Motorola	TXS	Funktion	CPU Typ A / 6303
B&R	MRS	R - 1 ⇒ SP!	
Kurz			

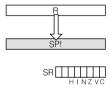
۸.	roccio	runae	arten	/ Once	do	ı			۸	۸,	٠~١	2.,	or		ak	ı۸	n				Ų	<u>'</u> L	P	ااد	JUU	,
Au	63316	rungs	arterr	Орсс																	U	Ī		G-F		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atι	ısı	eg	ist	er
1/1						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Η	Т	Ν	Z	٧	С
35						Г	Π														0	0	0	0	0	0

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



Mot	orola	TXS	;	Fun	ktion												1	CF	٦,	ΙŢ	УI	o E	3 /	6	В0	9
B&F	l	MRS	9	R ⇒	SP!																					
Kurz	Z																									
Δd	neein	runae	arten	/ Once	nde	Γ			Δ	dr	ام	ì.	۸r	۱۸/	ah	ما	n				О	1	С	P	30	
L	63316	unga	arterr	Орсс	Jue				_	uı	٠.	3 V	٠.	***	an						О	•	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
6/2						Ε	Α	М	F	Z	#	Ρ	С	T	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
1F 14						Γ	Г	Г	П				Г								0	0	0	0	0	0

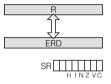
Dieser Befehl ist die Umkehrung zum Befehl MSR (TSX). Der System-Stapelzeiger wird mit dem Inhalt des Indexregisters R (X) geladen.



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.

Mot	orola	XGI	X	Fun	ktion									С	Pl	JΤ	yr	A	/ 6	303	3	Mot	orola	XG	DX	Fun	ktion								С	Pι	J T	ур I	3 /	680	19
B&F	'n	DXF	٧	ERD) ⇔ R	1															1	B&F	1	DXI	R	ERI) ⇔ R	1													
Kur	z																					Kur	z]															
Δdı	ressie	runae	arton	Onco	nde	Г		Δ	dro	Rv	orv	/ak	lor	$\overline{}$			0	Ī	PG1	000]	Δd	roccio	runas	sarten	/ Onc	nde			Adı	-AR	vor	wa	hle	'n		T	<u> </u>	CF	80	
Au	1633161	ungs	arteri /	Орсс	Jue			^'	uie	13 4	0. 1	rai					0		PG-	PC		L	63316	ung	sarteri	Opc	Jue			Aui	CIS	V O:	wa					<u> </u>	PC	80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	П) F	s	T #	P	R >	Υ	DΙ	!	В	G	Sta	atu	sre	giste	er	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	I C	F	S T	# F	R	X,	Y D	U !	! B	G	Statu	ısre	gis	er
2/1						E	A M	F	Z #	P	CI	Υ	DΙ	J !	В	G	Ι	I	N Z	V	С	7/2						E A	М	F Z	# F	c	1	Y D	U!	! В	G	1 1	N Z	Z V	С
18						П											၀)	ာ	0)	1E 10)	0)	0

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



Mot	orola			Fun	ktion												(CP	U	Т	УF	Э.	Α	/ 6	303	3
B&F	₹																									
Kur	z																									
	roccio	runa	sarten	/ Once	,do	Γ			^	4,	۰~۱	2 . ,	or		a h	ı۸	n					T	Р	G1	000	
Au	CSSIC	ung	sai terr	Орсс	ue				^	uı	CI	3 V	U	vv	211	16	"					Τ	F	G-	PC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	Sta	at	us	reç	iste	er
						Е	Δ	M	Е	7	44	٥	\sim	_	V	٦	11		В	6	н	1	N	7	V	$\overline{}$

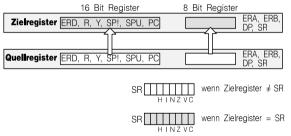
Mot	orola	TFR		Fun	ktion												1	CI	٧,	ΙŢ	уı	o E	3 /	68	30	9
B&F	ł	TFR		r _{Quell}	e ⇔ r _{zi}	el																				
Kurz	z																									
Δd	ressie	runae	arten	/ Once	nde	Γ			Δ	dr	ام	٠.	۸r	۱۸/	ah	ما	n				0	7	С	Pδ	30	
	63316	ungs	arterr	Орсс	Jue				^	uı	٠.	3 V	٠.	***	a 1 1						О	1	Р	CE	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atι	ısr	eg	ist	er
6/2						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
1F						Γ	Π	Г													•	•	•	•	•	•

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.

		Post	byte
Ziel-/Quellregister	r	Quelle	Ziel
ERD R (X) SPU SP! PC ERA ERB DP SR	D(X) YU!\$ABPC	0000 0001 0010 0011 0100 0101 1000 1001 1011 1010	0000 0001 0010 0011 0100 0101 1000 1001 1011 1010



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.

Mot	orola			Fun	ktion												-	CF	PU	ΙŢ	уı	0 /	A /	6	30	3
B&F	₹																									
Kur	z																									
Δdı	rossio	runas	arten	Once	nda	Γ			Δ	dr	ام:	۱۷	or	۱۸/	ah	ما	n						P	31 0	000	
Au	63316	ungs	arterr	Opce	ue				_	u	٠.	3 V	٠.	***	a 1 1		••					Ι	Ρ	G-F	С	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	usi	eg	ist	er
						E	Α	М	F	z	#	Р	С	ī	Υ	D	U	!	В	G	н	ı	N	z	٧	С

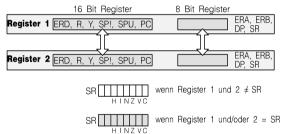
Mot	orola	EXG	;	Fun	ktion												(CF	'n	ΙT	уı	o E	3 /	68	во	9
B&F	ł	EXC	,	r₁	⇒r ₂																					
Kurz	z																									
Αd	ressie	runas	arten	/ Once	ode	Γ			Α	dr	eí	ßν	or	w	ah	le	n				0	_		P8	_	
,		90	u	Орос					•		٠.		٠.								0	1	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
7/2						Е	Α	М	F	Z	#	Р	С	1	Υ	D	U		В	G	Η	I	Ν	Z	٧	С
1E																					•	•	•	•	•	•

Der Inhalt des angegebenen Register wird vertauscht. Beide Register müssen die selbe Größe haben, d.h. entweder 1 Byte Register (ERA, ERB, DP, SR) oder 2 Byte Register (ERD, R, Y, SP!, SPU, PC).

Syntax: EXG r,r—Register 2

Der ganze Befehl besteht aus zwei Byte. Das erste Byte ist der Opcode (Befehl). Im zweiten Byte (Postbyte) wird angegeben, welche Register ausgetauscht werden sollen.

Register 1 / 2	r	Post Register 1	byte Register 2
ERD R (X) SPU SP! PC ERA ERB DP SR	D(X) YU!\$ABPC	0000 0001 0010 0011 0100 0101 1000 1001 1011 1010	0000 0001 0010 0011 0100 0101 1000 1001 1011 1010



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		Betr	iebsart	
Wiotorola	Dak	PG1000	PG-PC	CP 80	PC 80
PSH	PSH				0
PUL	PUL				0
PSHA	ANS	0	0	0	0
PULA	AVS	0	0	0	0
PSHB	BNS	0	0	0	0
PULB	BVS	0	0	0	0
PSHX	RNS	0	0	0	0
PULX	RVS	0	0	0	О

Mot	orola			Fun	ktion												1	CF	Pι	1	УI	Э.	A	6	303	ŀ
B&F	₹																									1
Kur	z																									
٨٨	roccio	runae	arten	/ Once	,do	Г			^	4	eí	2.,	^-	١	a h	ı	_					T	P	G1(000	٦
Au	CSSIC	ungs	arterr	Орсс	ue				~	u	CI	3 V	U.	vv.	211	ıeı	•					Τ	Р	G-F	Š	1
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	at	us	reg	iste	7
						-	Α.		F	7	44	_	_		V				_				N	7	V/ /	╗

																	_									
Mote	orola	PSH	ı	Fun	ktion												1	CI	2	T	УI) E	3 /	68	B0	9
B&R	ł	PSH	ı	Inha	It der	ar	nge	eg	eb	er	ne	n	Re	egi	st	er	aı	uf	S	/S	ter	n-	00	de	r	
Kurz	z			Anw	ender-	·S	ta	ре	ıl s	spe	eic	ch	erı	٦.												
Adı	ressie	rungs	arten	/ Opco	ode				Α	dr	eí	ŝν	or	w	ah	le	n				0	Ŧ	_	P 8		
IMPL.	DIR.	FXT.	IMMED.	IND.	REL.	H	l_	Ī-	l.	-		Р	lR	.,	.,	_		١.	_	_	_	_		_	ist	
IMPL.	DIK.	EAI.	IIMMED.	IND.	KEL.	1	ľ	F	S	1	#	۲	K	Á	T	D	U	1	R	G	J	all	ıoı	ъy	ıοι	51
5+/2						Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	T	N	Z	٧	С
34 / 36							Π	Γ					Π				•	•			0	0	0	0	0	0

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

Syntax:

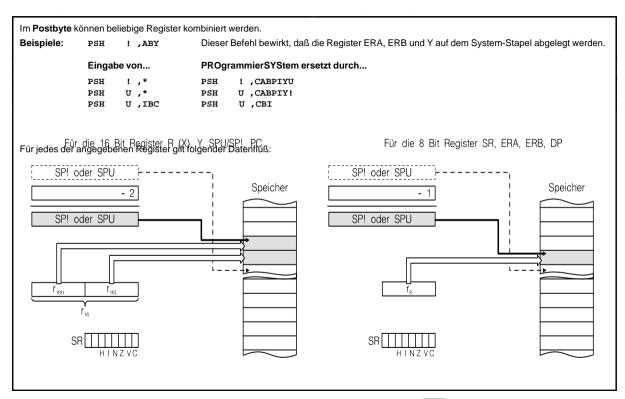
PSH | s ,r..... Register, die auf den Stapel gelegt werden sollen

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.

	= ! (Opcode = \$3 en der Register a	,
Register	r	Postbyte
SR ERA ERB DP R (X) Y SPU PC Alle außer PC	C A B P I (X) Y U \$	xxxxxxx1 xxxxxx1xx xxxxx1xxx xxx1xxxx xx1xxxxx x1xxxxx x1xxxxx 1xxxxxx

	U (Opcode = \$36 n der Register au	•
Register	r	Postbyte
SR ERA ERB DP R (X) >PI PC Alle außer PC	C A B P I (X) Y ! \$	XXXXXXXI XXXXXXIXX XXXXIXXXX XXXIXXXX XXIXXXXX X1XXXXXX X1XXXXXX 1XXXXXXX 01111111



Mot	orola			Fun	ktion													CI	?(J T	у	p.	A A	6	30	3	I
B&F	₹																									٦	I
Kur	B&R Kurz Adressierungsarten / Opcode Adreßvorwahlen PG1000																										
hΔ	ressie	runas	arten	arten / Oncode I Adrekvorwahlen IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII														_	ſ								
Au	. 000.0	ungo	ui toii ,	Орос	uu				_	· · ·	٠.	•	٠.	••	u .		••				L		Ρ	G-	PC		l
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	S	tat	us	reç	gist	ter	
						Е	Α	М	F	Z	#	Р	С	ı	Υ	D	U	!	В	G	Н	Ti	Ν	z	٧	С	ſ

_																											
	Mot	orola	PUL		Fun	ktion													CI	Pι	JΤ	УI	рI	B /	6	80	9
	B&F	ł	PUL		Ang	egebe	ne	F	e,	gis	ste	r	mi	t [Da	te	n '	vo	m	S	ys	te	m	- 0	de	er	
	Kur	z			Anw	ender	-S	ta	ре	П	ac	lei	n.														
	Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	ßν	٥r	w	ah	le	n					Ι	_	P		
		5+/2		ui toii	Орос	,uc	L			_	۵.	٠.	"	٠.	•••	u		••				О	<u>\</u>	Р	C	30	
	Kurz Adressie		EXT.	IMMED.	IND.	REL.	ı	0	F	s	т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
	5+/2						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	ī	N	Z	٧	С
	35 / 37						Γ	Γ	Γ				Γ	Γ				•	•	Γ		•	•	•	•	•	•

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.

	=! (Opcode = \$39 n der Register vo	,
Register	r	Postbyte
SR ERA ERB DP R (X) Y SPU PC Alle außer PC	C A B P (X) Y U \$ *	xxxxxx1 xxxxx1xx xxxx1xxx xxx1xxxx xx1xxxx xx1xxxxx x1xxxxxx

	U (Opcode = \$37 der Register von	
Register	r	Postbyte
SR ERA ERB DP R (X) Y SP! PC Alle außer PC	C A B P (X) Y ! \$ *	XXXXXXX1 XXXXXXXX XXXXXXXX XXXXXXX XXXXXXX XXXXXX

Im Postbyte können beliebige Register kombiniert werden. Beispiele: PUL ! ,CPI Dieser Befehl bewirkt, daß die Register SR, DP und R (X) mit Daten vom System-Stapel geladen werden Eingabe von... PROgrammierSYStem ersetzt durch... PUL PSH ! ,CABPIYU PUL U .* PSH U , CABPIY! U ,CBI PUL U ,IBC PSH 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 SP! oder SPU SP! oder SPU Speicher Speicher SP! oder SPU SP! oder SPU r_{16H} r₁₆₁ wenn $r_8 \neq SR$ HINZVC \square wenn $r_8 = SR$ HINZVC

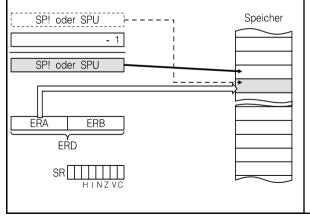
HINZVC

Motorola	PSHA	Funktion	CPU Typ A / 6303
B&R	ANS	ERA ⇒ (SP!); SP! - 1 ⇒ S	P!
Kurz			
			IO PG1000

٨٨	roccio	runae	arten	Once	do	ı			۸	۸,	eí	2.,	_		ah	ı۸	n				Q	1	PC	310)00	,
Au	CSSIC	ungs	arterr	Орсс	ue				~	u	CI	3 V	U	w	aii	116	•••				О	ī	P	G-F	'n	٦
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
4/1						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	1	Ν	Z	٧	С
36						Г	Г	Г	Г		П	П	П	Г			П		П		0	0	0	0	0	0

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

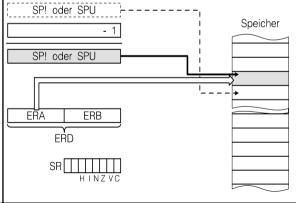
Danach wird SP! um den Wert 1 vermindert, damit SP! wieder auf die nächste freie Stelle des Stapels zeigt.



Mot	orola	PSH	IA	Fun	ktion												1	CF	Pυ	ΙT	Уļ) E	3 /	6	80	9
B&F	t	ANS	3	SP!	-1⇔	SI	P!;	E	R	Α	₽	(5	SP	!)												
Kur	z																									
Δd	rossio	runas	nde.	Γ			Δ	dr	ام	٠.	۸r	١٨/٠	ah	ما	n				0		С	P	30			
Α	Adressierungsarten / Opcode								_	۵.	٠.	•	٠.	•••		٠.	••				0	ı	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısı	eg	ist	er
6/2						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U		В	G	Н	1	Ν	Z	٧	С
34 02				Г	П	Г			Г		Г	П					П		0	0	0	0	0	0		

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.

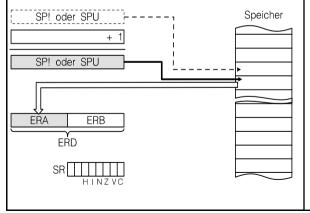


Motorola	PULA	Funktion	CPU Typ A / 6303
B&R	AVS	SP! + 1 ⇒ SP!; (SP!) ⇒ ERA	
Kurz			

٨٨	roccio	runae	arten	/ Once	,do	Г			^	4	eſ	2 . ,	^-	١	a h	ı	n				0	1	PC	310	000	٦
Au	CSSIC	ungs	arterr	Орсс	ue				^	u	e.	3 V	U.	vv.	211	116	"				О	ī	P	G-F	Š	٦
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	O	St	atı	ısr	eg	ist	ər
3/1						Е	Α	М	F	Z	#	Р	С	1	Υ	D	U		В	O	I	1	Z	Z	<	С
32						Γ	Γ										П				0	0	0	0	0	ା

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

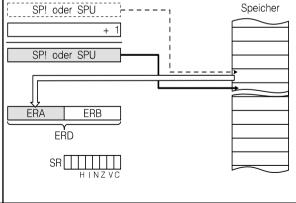
Danach wird das Ergebnisregister ERA mit dem Inhalt der Speicherstelle geladen, auf die SP! zeigt.



Mote	orola	PUL	Α	Fun	ktion												(CF	PU	ΙT	УI	рE	3 /	68	80	9
B&R	₹	AVS	;	(SP!) ⇒ El	R	۸;	SI	P!	+	1	₽	s	P!												
Kurz	z																									
Δdı	ressie	runas	arton	nde.	Г			_	dr	ام	R v	۸r	۱۸/	ah	ما	n				0	1	С	Pε	30		
_ ^u	63316	ungs	,ue				_	uı	٠.	3 4	٠.	***	a 1 1						a	1	Р	C	30			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atu	ısr	eg	ist	er
6/2						Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U		В	G	н	_	N	Z	٧	С
35 02						Г		Г	П				Г	П							0	0	0	0	0	0

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.

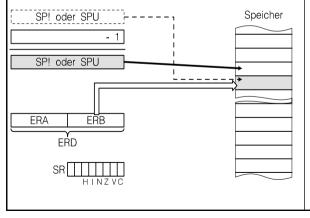


Mot	orola	PSF	IB	Fun	ktion									С	PU	J T	уp	A / 6303
B&F	₹	BNS	3	ERE	3 ⇒ (S	P!)); :	SP!	- 1	⇒	SP!							
Kur	z																	
۸.	roccio	runae	arten /	Once	,do	Г		_	dre	α.,	oru	ah	lor	$\overline{}$		П	0	PG1000
Au	CSSIC	ungs	ai teii /	Opce	ue				uie	13 V	OI W	an	iei	٠			0	PG-PC
IMPI	DIR	FXT	IMMED	IND	RFI	пΤ	οТ	FS	Т #	P	R X	IvI	ьTi	ıΓ	I _B	G	Stat	tusreaister

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

37

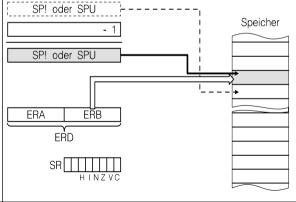
Danach wird SP! um den Wert 1 vermindert, damit SP! wieder auf die nächste freie Stelle des Stapels zeigt.



Mot	orola	PSH	IB	Fun	ktion												1	CF	? U	ΙT	УI	o E	3 /	68	В0	9
B&F	ł	SI	P!;	E	R	В	⇨	(\$	βP	!)																
Kurz	Kurz																									
Δ.	rossio	runas	nda	Γ			Δ	dr	ام	ŝν	۸r	\A/:	ah	ما	n				0	1	С	Pε	30			
^~	63316	, ac	L			_	uı	٠.	3 V	٠.	***	411		••				a	•	Ρ	CE	30				
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
6/2						Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
34 04				Γ		Г	П		Г	Г	П					П	П	П	0	0	0	0	0	0		

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.

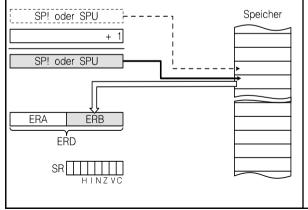


Motorola	PULB	Funktion	CPU Typ A / 6303
B&R	BVS	SP! + 1 ⇒ SP!; (SP!) ⇒ ERB	
Kurz			

٨٨	roccio	runae	arten /	/ Once	,do	Г			^	4	٠, ۲	2 . ,	^-	١	ah	ı	n				0	T	P	G10	00	П
Au	CSSIC	ungs	ai teii /	Орсс	ue	de Adreßvorwahlen O P									Р	G-F	'n	٦								
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	ısı	eg	ist	er
3/1						Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
33						Γ	Γ		Γ		Π	Γ									0	0	0	0	0	0

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

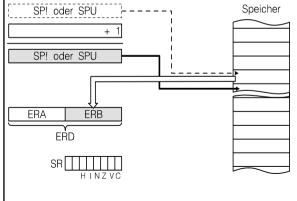
Danach wird das Ergebnisregister ERB mit dem Inhalt der Speicherstelle geladen, auf die SP! zeigt.



Mot	orola	PUL	В	Fun	ktion												1	CF	PU	T	уį) E	3 /	68	В0	9
B&F	t	BVS		(SP) ⇒ El	RI	3;	S	P!	+	1	₽	s	P!												
Kur	z																									
Adı	Δ	dr	ام:	ŝν	۸r	١٨/:	a h	ما	n			П	0	1	С	Pε	30									
Au	63316	ungs	arterr	Орсс	Jue				^	uı	٠.	3 .	٠.	***	411						0	1	Ρ	CE	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
6/2						Ε	Α	М	F	Z	#	Р	С	Ι	Υ	D	U		В	G	Н	1	Ν	Z	٧	С
35 04						Г	Г	Г	П			П	П							П	0	0	0	0	0	0

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.



	DOLLY			00UT 4 / 0000		20117			OD!! T D / 0000
Motorola	PSHX	Funktion		CPU Typ A / 6303	Motorola	PSHX	Funktion		CPU Typ B / 6809
B&R	RNS		; SP! - 1 ⇒ SP!		B&R	RNS		SP!; R _∟ ⇒ (SP!)	
Kurz		R _H ⇒ (SP!)	; SP! - 1 ⇒ SP!		Kurz		SP! - 1 ⇒	SP!; R _H	
Adressier	rungsarten <i>i</i>	Opcode	Adreßvorwahler	O PG1000 O PG-PC	Adressie	rungsarten	/ Opcode	Adreßvorwahle	O CP 80 O PC 80
IMPL. DIR. 5/1 3C	EXT. IMMED.	IND. REL.	I O F S T # P R X Y D U E A M F Z # P C I Y D U		IMPL. DIR. 7/2 34 10	EXT. IMMED.	IND. REL.	I O F S T # P R X Y D E A M F Z # P C I Y D	
abgelegt. Da der Sy Speichers gespeiche	vstem-Stap stelle des S ert:	elzeiger SP tapels zeig 1) R 2) S 3) R 4) S s.Befehles tapels:	(X) wird auf dem Sys e! immer auf die nächs t, wird der Inhalt von k E C SP! F - 1 ⇒ SP! F - 1 ⇒ SP! Zeigt SP! wieder auf E SP! wieder auf	ste freie R (X) wie folgt	abgelegt. Da der Sydes Stape	vstem-Stap els zeigt, w	elzeiger SF ird der Inha 1) S 2) F 3) S 4) F psführung tapels.	(X) wird auf dem Sys P! auf die letzte belegt Ilt von R (X) wie folgt SP! - 1 ⇒ SP! R₁ ⇒ (SP!) SP! - 1 ⇒ SP! SH ⇒ (SP!) des Befehles auf die	te Speicherstelle gespeichert:
	SR III	NZ VC				SR III	NZ VC		

Motorola	PULX	Funktion	CPU Typ A / 6303
B&R	RVS	SP! + 1 ⇒ SP!; (SP!) ⇒ R _H	
Kurz		SP! + 1 ⇒ SP!; (SP!) ⇒ R _L	

٨.	roccio	runae	arten	/ Once	Ndo.	ı			۸	d,	eí	2,,	^-		ah	ı	n				J	<u>'</u> L	P	5 10	JUU	<u>' I</u>
Au	CSSIC	luligs	arterr	Орсс	Jue				~	u	CI	3 V	U.	vv.	aıı	116	••				O	ī	P	G-F	SC	٦
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	er	
4/1						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	1	Ν	Z	٧	С
38						Г															0	0	\circ	0	0	\circ

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:

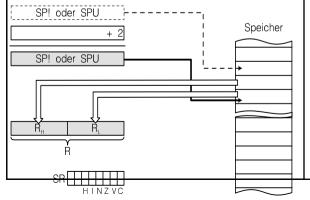
1) SP! + 1 □ SP!

2) R_L ⇒ (SP!)

3) SP! +1 ⇒ SP!

R_H ⇒ (SP!)

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



Mot	orola	PUL	.Х	Fun	ktion												1	CI	٦,	ĮΤ	У	o E	3 /	6	В0	9
B&F	t	RVS	3	(SP) ⇒ R	н;	SI	?!	+	1	₽	s	P!													
Kur	z			(SP	(SP!) ⇒ R _L ; SP! + 1 ⇒ SP!																					
۸.	dressierungsarten / Opcode Adreßvorwal														ah	ı	n				0	7	С	P	30	
Au	CSSIC	ungs	arterr	Opce	Jue				^	uı	e,	3 V	U.	vv.	aıı		••				0	7	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	ısı	eg	ist	er
7/2						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
35 10						Г			П			Г	Г								0	0	0	0	0	0

Das Indexregister R (X) wird mit Daten vom Systemstapel geladen.

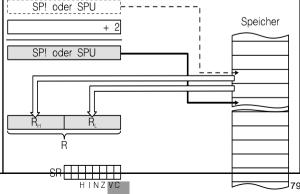
Da der System-Stapelzeiger SP! immer auf die letzte belegte
Speicherstelle des Stapels zeigt, wird R (X) wie folgt vom Stapel
geladen:

1) (SP!) ⇒ R L

2) SP! + 1 ⇒ SP! 3) (SP!) ⇒ R_H

4) SP! + 1 ⇒ SP!

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



3.5. LOGISCHE VERKNÜPFUNGEN

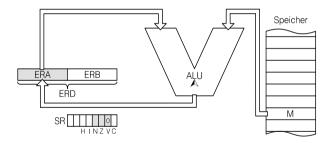
In diesem Abschnitt werden alle Befehle beschrieben, die den Inhalt von Registern und/oder Speicherstellen miteinander verknüpfen und wieder abspeichern.

Motorola	B&R		Betr	iebsart	
Motorola	Dak	PG1000	PG-PC	CP 80	PC 80
ANDA	UND	0	0	0	0
ANDB	UB	0	0	0	0
AIM	AIM		0		
ORAA	OD	0	0	0	0
ORAB	ОВ	0	0	0	0
OIM	OIM		0		
EORA	EXO	0	0	0	0
EORB	EB	0	0	0	0
EIM	EIM		0		

Mot	orola	ANI)A	Fun	ktion												-	CF	7	J	У	3	4 /	6	30	3
B&F	₹	UNI)	ERA	A _ (M)) :	⇨	E	R/	١																٦
Kur	z	U																								-
۸.4	roccio	runae	arten	Once	ndo.	Γ			_	4,	٠,	2.,	or	\A/-	ah	10	n				0	ī	P	G1 (000	П
Au	162216	rungs	arterr	Opco	oue				^	uı	eı	٥v	Oi	w	all	iie	"				О	ıŢ	Р	G-F	С	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
	3/2	4/3	2/2	4/2		Ε	Α	М	F	z	#	Р	С	I	Υ	D	U	!	В	G	Ι	T	N	z	٧	С
	94	B4	84	A4		b	0	0	0	0	0	0	0	0							0	0	•	•	▼	0

I	Mot	orola	AND	PΑ	Fun	ktion												1	CF	Pι	JΤ	УĮ	o E	3 /	6	В0	9
ſ	B&F	ł	UNE)	ERA	(M) ۸) [⇒	EI	R/	١.																
	Kur	z	U																								
ſ	Δd	ressie	nde	Γ			Δ	dr	e í	3v	or	w	ah	le	n				0	1	С	P	30				
ı	Au	000.0	ungo	ui toii i	Орос	Juc				_	۵.	٠.	•	٠.	•••	٠.,		••				0	1	Ρ	C	30	
ſ	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
ſ		4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Ρ	С	Ι	Υ	D	U		В	G	Η	I	N	Z	٧	С
ſ	94 B4 84 A4						0	0	0	0	0	0	0	0	0	•	•	•	•	•	•	0	0	•	•	•	0

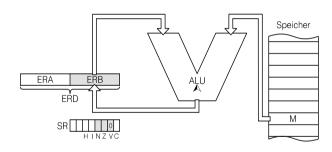
Das Ergebnisregister ERA wird mit dem Inhalt der Speicherstelle M UND-verknüpft. Das Ergebnis wird im Register ERA gespeichert.



Mot	orola	ANI	OB	Fun	ktion												1	CI	2	J	УΙ	0 /	A A	6	30	3
B&F	₹	UB		ERE	3 ^ (M)) 1	⇨	E	RE	3																1
Kur	z																		-							
٨٨	roccio	runae	arton	Once	pcode Adreßvorwahlen O PG1000														П							
Au	CSSIC	ungs	arterr	Opce	ue	L			~	uı	CI	3 V	U.	vv.	aıı	116	"				O	•	Р	G-F	C	٦
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	us	reg	ist	er
	3/2	4/3	2/2	4/2		Е	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Н	I	N	z	٧	С
	D4	F4	C4	E4		О	0	0	0	0	0	0	0	0			П		П		0	0	•	•	▼	0

Mot	orola	ANI	В	Fun	ktion												1	CI	٦(JΤ	у	рE	B /	6	80	9
B&F	1	UB		ERE	3 ^ (M)) :	⇒	EI	RE	3																
Kurz	Z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	۱۷	٥r	w	ah	le	n				С	<u> </u>	_	P		
Au	000.0	· ugo	u. to	n / Opcode Adreßvorwal										u.,		•••				C)	Ρ	С	30		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	ısr	eg	ist	er
	4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
	D4	F4	C4	E4		Ь	0	0	0	0	0	0	0	0	•	•	•	•	•	•	0	0	•	•	▼	0

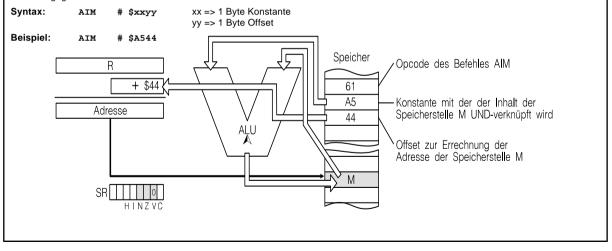
Das Ergebnisregister ERB wird mit dem Inhalt der Speicherstelle M UND-verknüpft. Das Ergebnis wird im Register ERB gespeichert.



Mot	orola	AIN	ı	Fun	ktion													CI	? U	J T	Ίу	р	4 /	6	30	3
B&F	₹	AIN	ı	(R +	Offse	t)	^	IN	11	1 :	⇔	(F	+	C	ff	se	t)									
Kur	Kurz																									
Δdı	rossio	runas	arten	nda	Γ			Δ	di	reí	21/	or	۱۸/	ah	ما	n					Ι	P	G1(000	\Box	
ζū	63316	runga	arterr	Opce	ue	L				u	01		01	***	a 1 1						С)	Ρ	G-I	PC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	JSI	reg	jist	er
				7/3		Е	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
				61		Γ	Г				0										0	0	•	•	▼	0

Mot	orola			Fun	ktion												1	CF	2 U	ΙŢ	·уІ	o E	3 /	68	30	9
B&F	ł																									
Kurz	z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	ام	١٧	or	w	ah	le	n					Ι	_	Pε		
	000.0	ungo	ui toii	Орос	, , ,				_	۵.	٠.	•	٠.	•••	۵.,									CE		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
						Ε	Α	М	F	Z	#	Р	С	Ι	Υ	D	U	!	В	G	Н	1	Ν	Z	٧	С
						Г	Г	Г	П		Г		П				П	П			П					Г

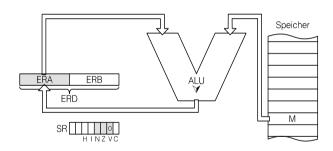
Ein beim Befehl angegebener Immediate Wert \$xx (Konstante) wird mit dem Inhalt der Speicherstelle M UND-verknüpft. Das Ergebnis wird in derselben Speicherstelle gespeichert. Die Adresse M ergibt sich aus der Summe von Indexregister R (X) und dem Offset \$yy, der beim Befehl angegeben werden muß.



Mot	orola	OR	AA	Fun	ktion													CI	2 U	ı	уı	0 /	4 /	6	30	3
B&F	₹	OD		ERA	√ (M)) :	⇨	Е	R/	١																٦
Kur	Surz O																					-				
۸dı	Kurz O Adressierungsarten / Opcode Adreßvorwahl																n				0	7	P	G1 (000	П
Aui	CSSIC	rungs	arterr	Орсс	Jue																О	ŀ		G-F		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
	3/2	4/3	2/2	4/2		Е	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
	9A	BA	8A	AA		b	0	ာ	0	0	0	0	0	0							0	0	•	•	•	0

Mot	orola	ORA	λA	Fun	ktion												1	CI	٦(JΤ	у	рE	B /	6	80	9
B&F	ł	OD		ERA	A ∨ (M)) :	₽	ΕI	R/	١																
Kur	z	0																								
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	ŝν	٥r	w	ah	le	n				С	<u> </u>	_	P		
		90		Орос					•		٠.	••	٠.	•••			••				C)	Ρ	С	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
	4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
	9A	BA	8A	AA		0	0	0	0	0	0	0	0	0	•	•	•	•	•	•	ા	0	•	•	▼	0

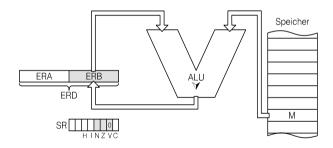
Das Ergebnisregister ERA wird mit dem Inhalt der Speicherstelle M ODER-verknüpft. Das Ergebnis wird im Register ERA gespeichert.



Mot	orola	OR	AB	Fun	ktion												1	CF	?(Ţ	УI	0 /	4	6	30	3
B&F	₹	ОВ		ERE	3 ∨ (M)) :	⇒	EI	RE	3																٦
Kur	z																									
٨٨	roccio	runae	arton	/ Once	Opcode Adreßvorwahlen O PG1000														П							
Au	CSSIC	ungs	arterr	Opce	ue				^	uı	CI	3 V	U.	vv	211	116	"				О	•	Р	G-F	SC	П
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	IS	reg	ist	er
	3/2	4/3	2/2	4/2		Ε	Α	М	F	Z	#	Р	С	_	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
	DA	FA	CA	EA		Ь	0	0	0	0	0	0	0	0			П				0	0	•	•	▼	0

٦	84-4	1-	00.		F	latina.												П.	~-					· ·	$\overline{}$		ᇧ
	Mot	orola	OR/	/R	Fun	ktion												- ['	Cr	'n	, ,	У	э 6	3 /	66	30	9
	B&R	ł	ОВ		ERE	3 v (M)) 1	⇒	E	RE	3																
	Kurz	z																									
ı	Δdı	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	eſ	ŝν	or	w	ah	le.	n				0	_		Р8		
	, Au	000.0	ungo	ui toii i	Орос	Juc	L			_	۵.	٠.	•	٠.	•••		٠.	•••				0	1	P	C8	30	
	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	S	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	egi	ist	er
		4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Ρ	С	\perp	Υ	D	U		В	G	Ι	I	Ν	z	٧	С
		DA	FA	CA	EA		ि	0	0	0	0	0	0	0	0	•	•	•	•	•	•	0	0	•	•	•	0

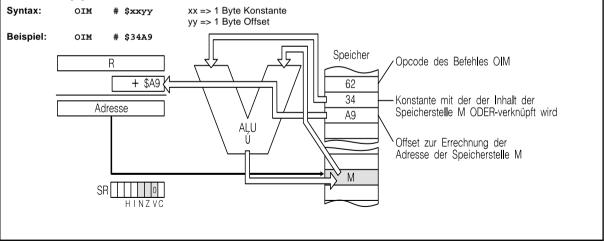
Das Ergebnisregister ERB wird mit dem Inhalt der Speicherstelle M ODER-verknüpft. Das Ergebnis wird im Register ERB gespeichert.



Mot	orola	OIN	l	Fun	ktion												(CI	?(J T	у	p.	A	6	30	3
B&F	₹	OIN	ı	(R +	Offse	t)	v	I۱	٨N	1	⇨	(R	+	С	ff	se	t)									
Kur	Kurz																									
٨٨	Adressierungsarten / Opcode Adreßvorwahlen															Г		P	G1(000	Л					
Aui	essie	rungs	arterr	Opce	ue				^	u	eı	٥v	OI.	w	all	iie	"				С	7	Р	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	at	us	reg	ist	er
				7/3		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	T	N	Z	٧	С
				62		Γ	Γ				0										b	0	•	•	•	0

Mote	orola			Fun	ktion												1	CI	٦,	J 7	у	рE	3 /	68	80	9
B&R	t																									
Kurz	Z																									
Δdı	ressie	runas	arten /	/ Once	ode	Г			Δ	dr	-66	Rv	or	w	ah	le	n					Ι	_	P		
''	000.0.	ungo	uc ,	Option	,40	L				۵.	٠.	•	٠.	•••			••				L		Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.		0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
						Е	Α	М	F	Z	#	Ρ	С	1	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
			\Box			Г	Г	Г	Г	П	П	Г	П			П		Г	П		Г	Г		П	П	Г

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.



IVIOL	UlUla	יים	\A	I un	KUUII												-1'	u		' '	УI	,	٠,	U.	30	ีไ
B&F	₹	EXC)	ERA	M) ⊕ A)	⇔	Е	R/	٩																1
Kur	z	Е																								
٨٨	roccio	runae	arten	/ Once	ndo.	Γ			^	4,	۰~۱	2 1/	or	١	ah	ı	n				0	T	P	310	000	П
ť	CSSIC	ungs	arterr	Орсс	Jue				^	uı	CI	3 V	UI.	w	aii	iie					О	ī	P	G-I	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ISI	eg	ist	er
	3/2	4/3	2/2	4/2		Ε	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С
	98	B8	88	A8		Ь	0	0	0	0	0	0	0	0							0	0	•	•	•	0

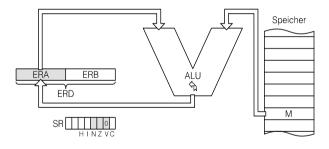
Eunktion

Motorolo FORA

Mot	orola	EOF	RA	Fun	ktion												1	CF	P	Ţ	УI	p E	3 /	6	В0	9
B&F	₹	EXC)	ERA	⊕ (M)	₽	Ε	R	4																
Kur	z	E																								
hΔ	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	ŝν	٥r	w	ah	le	n				0	<u> </u>		P		
Au	. 000.0	· ungo	u. to	Орос	,uc	L			_	۵.	٠.	•	٠.	•••	u						О	<u> </u>	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
	4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
	98	В8	88	A8		ि	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.

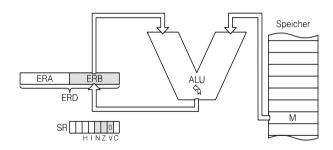
CBII Tup A / 6202



Mot	orola	EOF	₹В	Fun	ktion												1	CI	?(Ţ	УI	0 /	4	6	30	3
B&F	₹	ЕВ		ERE	B ⊕ (M)	₽	Е	RE	3																٦
Kur	z																									
٨٨	Adressierungsarten / Opcode Adressvorwahlen O PG1000																									
Au	CSSIC	ungs	arterr	Opce	ue	L			^	uı	CI	3 V	U.	vv.	aıı	116	"				О	•	Р	G-F	S	П
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	IS	reg	ist	er
	3/2	4/3	2/2	4/2		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
	D8	F8	C8	E8		b	0	0	0	0	0	0	0	0			П				0	0	•	•	▼	0

Mot	orola	EOF	RB	Fun	ktion												1	CI	Pι	ĮΤ	У	p E	B /	6	80	9
B&F	1	ЕВ		ERE	B ⊕ (M)	⇔	Ε	RI	В																
Kur	z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	ßν	٥r	w	ah	le	n				С	_	С	P	30	
, Au	000.0	· ugo	u. to	Орос	Juc	L			_	۵.	٠.	•	٠.	•••	u.,		••				C)	Ρ	С	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
	4/2	5/3	2/2	4+/2+		Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
	D8	F8	C8	E8		Э	0	0	0	0	0	0	0	0	•	•	•	•	•	•	ા	0	•	•	▼	0

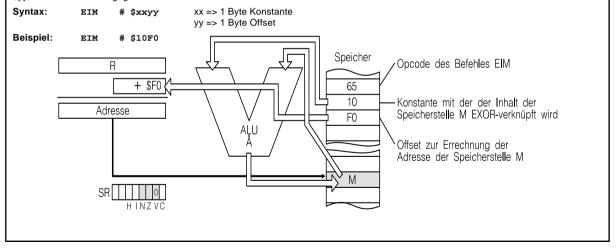
Das Ergebnisregister ERB wird mit dem Inhalt der Speicherstelle M Exklusiv-Oder-verknüpft. Das Ergebnis wird im Register ERB gespeichert.



Mot	orola	EIM		Fun	ktion													CI	P U	ı	уı	0 /	A /	6	30	3
B&F	₹	EIM		(R +	Offse	t)	Ф	II	۷N	Л	⇒	(F	۲ ا	. (Off	se	et)									
Kur	z																									
Δdı	rossio	runas	arten /	Once	nda	Adreßvorwahlen																Τ	P	G1(000	5
Aui	63316	unga	arteri /	Орсс	ue	L				u	٠.	3 V	٠.	**	an						О	·	Ρ	G-I	PC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	usi	reg	jist	er
				7/3		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	1	Ν	Z	٧	С
				65		Γ	Γ		Γ		0										0	0	•	•	•	0

Mot	orola			Fun	ktion												1	CF	2 U	T	·уІ	o E	3 /	68	В0	9
B&F	₹																									
Kur	z																									
Δd	ressie	runas	arten	/ Once	nde	Г			Δ	dr	eſ	١٧	or	w	ah	le	n					Ι	_	Pε		
الم	. 000.0	ungo	ui toii	Орос	Juc				_	۵.	٠.	•	٠.	•••		٠.	••						Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	-	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atu	ısr	eg	ist	er
						Е	Α	М	F	Z	#	Р	С	_	Υ	D	U	!	В	G	Ι	_	Ν	Z	٧	С
						Г	Г	Г	П	П			П					П			П		П	П	П	Г

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.



3.6. ARITHMETISCHE OPERATIONEN

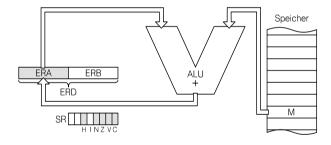
In diesem Abschnitt werden die Befehle beschrieben, die zwei Operanden addieren, subtrahieren oder multiplizieren.

Matauala	D.O.D.		Betr	iebsart	
Motorola	B&R	PG1000	PG-PC	CP 80	PC 80
ADDA	+	0	О	0	О
ADCA	ADD	0	0	0	0
ADDB	+B	0	0	0	0
ADCB	++B	0	0	0	0
ADDD	+D	0	0	0	0
ABA	A+B	0	0	0	0
SUBA	-	0	0	0	0
SBCA	SUB	0	0	0	0
SUBB	-B	0	0	0	0
SBCB	B	0	0	0	0
SUBD	-D	0	0	0	0
SBA	A-B	0	0	0	0
ABX	B+R	0	0	0	0
MUL	A*B	0	0	0	О

Mot	orola	ADI	DA	Fun	ktion													CF	P	ΙT	УĮ	o /	۸ /	6	30	3
B&F	₹	+		ERA	+ (M)) 1	⇒	ΕI	R/	١																
Kur	z																									
Λ.	roccio	runae	arton	Onco	ode Adreßvorwahle											n				0	T	P	G10	000	\Box	
Au	CSSIC	ungs	ai teii /	Орсс	ue																a	1	P	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.											D	U	!	В	G	St	atı	ısı	eg	ist	er
	3/2	4/3	2/2	4/2		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
	9B	BB	8B	AB		0	0	0	0	0	0	0	0	X Y D							•	0	•	•	•	•

_						_																						
I		Mot	orola	ADE	DΑ	Fun	ktion												1	CI	٦(JΤ	УI	o E	3 /	68	30	9
1		B&F	ł	+		ERA	+ (M)) 1	⇒	ΕI	R.A	١																
I		Kur	z																									
1	ſ	Αd	ressie	runas	arten	/ Once	ode	Γ			Δ	dr	eſ	۱۷	or	w	ah	le	n				О	1		Р8	_	
J	ı	, , , ,		90	· a	Орос		L			•	۳.	٠.	••	•								О	l		C 8		
1	- [IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atι	ısr	egi	ist	er
1	ı		4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	Ν	z	٧	С
1	ı		9B	BB	8B	AB		ા	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.

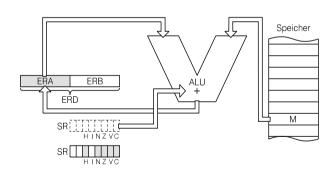


Motorola	ADCA	Funktion	CPU Typ A / 6303
B&R	ADD	ERA + (M) + C ⇒ ERA	
Kurz	A, ++		

Δdı	ressie	runas	arten /	Onco	nde	Г			Δ	dr	eí	١٧	٥r	w	ah	le.	n				О	1	PC	310	00	╗
Au.	000.0	ungo	ui toii /	Opoc	, uc				_	۵.	٠.	•	٠.	•••	u						О	ı		G-F		╝
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
	3/2	4/3	2/2	4/2		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
	99	B9	89	A9		0	ာ	0	0	0	0	0	0	ာ							•	0	•	•	•	•

Mot	orola	ADO	CA	Fun	ktion												1	CF	Pυ	ĮΤ	·уІ	o E	3 /	6	80	9
B&I	₹	ADE		ERA	+ (M)	+ (· C	; :	⇒	E	R.A	١														
Kur	z	A, +	+		de Adreßvorwah																					
Ad	ressie	runas	arten	/ Opco	ode	Γ		Adreßvorwahl								le	n				О	_		Pδ		
		9-				L	Adreßvorwa												О	l	Р	C	30			
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
	4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Р	С	1	Υ	D	U		В	G	Н	I	Ν	Z	٧	С
	99	B9	89	A9		Ь	0	0	0	0	0	0	0	0	•	•	•	•	•	•	•	0	•	•	•	•

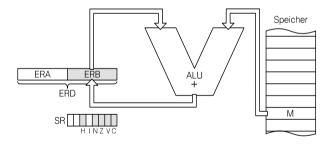
Die Inhalte von Ergebnisregister ERA, Speicherstelle M und das Carry-Flag (möglicher Übertrag einer vorhergehenden Addition) werden addiert. Das Ergebnis wird im ERA abgespeichert.



Mot	orola	ADE	DВ	Fun	ktion												1	CF	PU	J T	УI	р	4 /	6	30	3
B&F	₹	+B		ERB	+ (M)) :	⇒	E	RE	3																
Kur	z																									
	roccio	runae	arten	Onco	do	Adreßvorwa										ı	n				C	7	P	G1 (000)
Au	CSSIC	ungs	arterr	Орсс	ue	Adreßvorwah										16	•••				O)	Ρ	G-F	c	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
	3/2	4/3	2/2	4/2		Ε	Α	М	F	Z	#	Р	С	_	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
	DB	FB	CB	EB		Ь	0	0	0	0	0	0	0	0	П		Π				•	0	•	•	•	•

Motorola ADDB Funktion CPU Typ B / 68 B&R							_																						
Note	I		Mot	orola	ADE	В	Fun	ktion													CI	٦(JΤ	УI	o E	3 /	6	В0	9
Adressierungsarten / Opcode	1		B&F	ł	+B		ERE	3 + (M)) 1	⇒	ΕI	RE	3																
Adressierungsarten / Opcode Adresvorwahlen Opcode Adresvorwahlen Opcode Adresvorwahlen Opcode O	I		Kur	z																									
MPL DIR. EXT. MMED. IND. REL.	1		Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	<u>ام</u>	١٧	٥r	w	ah	le.	n				О	1		_		
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	J	l	Au.	000.0	ungo	u. to	Орос	,uc	L			_	۵.	٠.	•	٠.								О	ı				
	1		IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
DB FB CB EB 00000000000000000000000000000000	1			4/2	5/3	2/2	4+/2+		Е	Α	М	н	Z	#	Р	С	I	Υ	D	U	!	В	G	Ι	I	N	Z	٧	С
	1	l		DB	FB	CB	EB		ं	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.



																					٠.					
B&F	₹	++B	3	ERE	3 + (M)) +	- C	;	⇒	E	RE	3														
Kur	z																									
٨٨	roccio	runae	arten	/ Once	,do	Γ			^	dreßvorwahlen O			7	P	G1 (000	\Box									
ť	63316	ungs	arterr	Орсс	ue				_	uı	e.	3 V	UI.	w	aii						О	1	Ρ	G-F	2	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	usi	eg	ist	er
	3/2	4/3	2/2	4/2		Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
	D9	F9	C9	E9		b	0	0	0	0	0	0	0	0							•	0	•	•	•	•

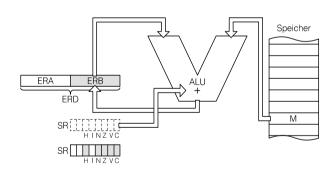
Funktion

Motorola ADCB

Mot	orola	ADO	В	Fun	ktion												1	CI	٦(JΤ	УI	рE	B /	6	80	9
B&F	1	++B		ERE	3 + (M)	+ (- C	; 1	⇒	E	RE	3														
Kur	z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	١٧	٥r	w	ah	le	n				0	<u> </u>	_	P		
		gc		Орос		L			•		٠.	••	٠.	•••			••				О)	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	en v v !		В	G	St	atı	ısr	eg	ist	er
	4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
	D9	F9	C9	E9		b	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.

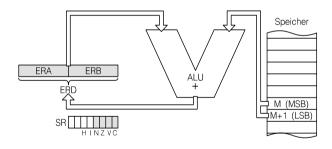
CPU Typ A / 6303



Mot	orola	ADI	טט	Fun	ktion												- 1'	CI	7	, ,	у	0 /	4 /	6	30	3
B&F	₹	+D		ERD	+ (M:	M	+1	I)	=	E	R	D														
Kur	z																									
٨٨	Adressierungsarten / Opcode Adreßvo														ah	ı	n				0	1	P	G10	000	П
Au	CSSIC	ungs	arterr	Орсс	ue				^	uı	e,	3 V	U.	vv.	aıı	116	••				0	•	Р	G-F	S	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	JSI	eg	ist	er
	4/2	5/3	3/3	5/2		Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
	D3	F3	C3	E3		Г	П		П		0	0	0	0					П		0	0	•	•	•	•

	Mot	orola	ADI	D	Fun	ktion												1	CF	Pυ	ΙŢ	y	рE	3 /	6	80	9
	B&R	ł	+D		ERD	+ (M:	M	+1)	Ξ	> E	R	D														
	Kurz	z																									
ı	Δd	ressie	runas	arten	/ Once	nde				Δ	dr	eſ	ŝν	or	w	ah	le.	n				O)		Pδ		
	Α	000.0	· ugo	ui toii i	Орос	Juc				_	۵.	٠.	•	٠.	•••		٠.	••				С)	Ρ	C	30	
	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	_	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
		6/2	7/3	4/3	6+/2+		Е	Α	М	F	Z	#	Ρ	С	-	Υ	D	U		В	G	Н	ı	Ν	Z	٧	С
		D3	F3	C3	E3			П	П	П				\overline{a}	\circ	•	•	•		•	•	0	0	•	•	•	

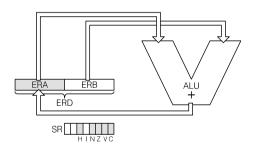
Die Inhalte von Ergebnisregister ERD und den Speicherstellen M und M+1 werden addiert. Das Ergebnis wird im ERD abgespeichert.



Mot	orola	ABA	4	Fun	ktion												-	CI	7	1	УΙ	, د	A /	6	30	3
B&F	₹	A+E	3	ERA	+ ER	В	Ξ	> I	ER	Α																٦
Kur	z																									١
۸.4	roccio	runae	arten	ndo.	Γ			_	4,	۰~۱	2.,	or	\A/-	ah	ı	n				С	7	Р	G10	000	П	
Au	CSSIC	ungs	arterr	Орсс	Jue				^	uı	CI	3 V	U.	vv.	aıı		••				O	7	Р	G-F	SC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	us	reg	ist	er
1/1						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
1B						Г	Г	П	П			П	П	П					П		•	0	•	•	•	•

1	Mot	orola	ABA	,	Fun	ktion												1	CF	٧U	ΙT	УĮ	o E	3 /	6	В0	9
	B&F	ł	A+E		ERA	+ ER	В	Б	> [ER	Α																
	Kur	z																									
ſ	Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	ŝν	or	w	ah	le	n				0	_		P		
ı	7.0	000.0	· u.i.go	ui toii i	Орос	Juc	L			_	۵.	٠.	•	٠.	•••		٠.	•••				0	1	Ρ	C	30	
I	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
ı			25/3				Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U		В	G	Η	I	Ν	Z	٧	С
ı			BD FF 90				Γ														П	•	0	•	•	•	•

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



Da der Befehl im 6809 nicht vorhanden ist, wird dieser durch einen Unterprogrammsprung 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!)

ADDA ! ,+ Der Inhalt von ERA und der Speicherstelle, auf die der System-Stapelzeiger SP! zeigt, werden addiert. Danach wird SP!

um 1 erhöht (Postinkrement): ERA + (SP!)

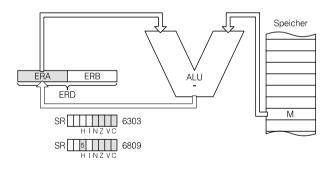
SP! + 1 ⇒ SP!

RTS Verlassen des Unterprogrammes.

Mot	orola	SUE	ЗА	Fun	ktion												1	CI	2	Ţ	УI	р	4	6	30	3
B&F	₹	-		ERA	A - (M)	Ξ	>	ER	RA																	٦
Kur	z																									
۸.4	roccio	runae	arten	/ Once	ndo.	Γ			^	4,	۰~۱	2 . ,	or	١	ah	ı	n				0	7	Р	G1(000	П
Au	CSSIC	ungs	arterr	Орсс	Jue				^	uı	CI	3 V	U.	vv.	aıı		••				0	7	Р	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	IS	reg	ist	er
	3/2	4/3	2/2	4/2		Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Ι	I	N	Z	٧	С
	90	B0	80	A0		Ь	0	0	0	0	0	0	0	0							0	0	•	•	•	•

1	Mot	orola	SUE	BA	Fun	ktion													CI	٦(JΤ	УI) E	3 /	68	30	9
1	B&F	ł	-		ERA	- (M)	п	> [ER	A																	
ı	Kur	z																									
]	Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e۱	١٧	or	w	ah	le	n				0	1	_	P8		
╛		000.0	ungo	ui toii ,	Орос	Juc	L			_	۵.	٠.		٠.	•••	u		•••				О	ı	Ρ	CE	30	
r	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	1	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
7		4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Н	1	Ν	Z	٧	С
•		90	B0	80	A0		О	0	0	0	0	0	0	0	0	•	•	•	•	•	•	х	0	•	•	•	•

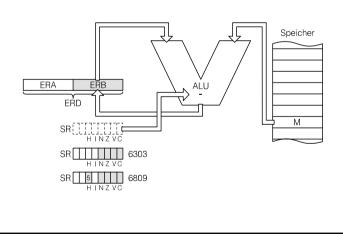
Der Inhalt der Speicherstelle M wird vom Ergebnisregister ERA subtrahiert. Das Ergebnis wird im ERA abgespeichert.



Mot	orola	SBC	jΑ	Fun	ktion												- 1'	CI	7	, ,	у	0 /	4 /	6	30	3
B&F	₹	SUI	В	ERA	· - (M)	-	С	₽	E	R	Α															٦
Kur	z																									
٨٨	roccio	runae	arten	Once	,do	Γ			_	4,	٠, ۲	2 1,	or	\A/-	a h	10	n				0	7	P	G10	000	П
Kui	CSSIC	rungs	arterr	Орсс	ue				^	uı	e.	3 V	UI.	w	211	iie					О	ŀ	Ρ	G-F	C	┒
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	JSI	reg	ist	er
	3/2	4/3	2/2	4/2		Ε	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С
	92	B2	82	A2		Ь	0	0	0	0	0	0	0	0							0	0	•	•	•	•

		1 1																								
Mot	orola	SBC	CA	Fun	ktion												1	CF	٦,	JΤ	УI	рE	B /	6	80	9
B&F	1	SUE	3	ERA	· - (M)	- 1	С	Ξ	• E	R	Α															
Kur	z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	ßν	٥r	w	ah	le	n				0	<u> </u>	_	P		
, Au	000.0	· ugo	u. to	Орос	Juc	L			_	۵.	٠.	•	٠.	•••	u.,		•••				О)	Ρ	С	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
	4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
	92	B2	82	A2		E A M F Z # P C		0	•	•	•	•	•	•	x	0	•	•	•	•						

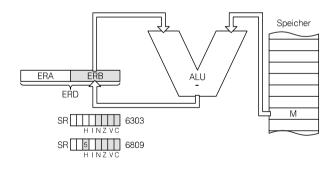
Der Inhalt der Speicherstelle M und das Carry-Flag werden vom Ergebnisregister ERA subtrahiert. Das Ergebnis wird im ERA abgespeichert.



Mot	orola	SUE	3B	Fun	ktion												1	CI	PU	Ţ	УI	р	A /	6	30	3
B&F	₹	-В		ERE	3 - (M)	Ξ	>	ER	RВ																	٦
Kur	z																									
٨٨	Adressierungsarten / Opcode Adreßvorwahlen																0	7	P	G1(000	П				
ť	CSSIC	ungs	arterr	Орсс	ue				^	uı	e.	3 V	UI.	w	aii	iie					О)	Р	G-I	2	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Ρ	R	х	Υ	D	U	!	В	G	St	atı	us	reg	ist	er
	3/2	4/3	2/2	4/2		Ε	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Ι	I	N	Z	٧	С
	D0	F0	C0	E0		Ь	0	0	0	0	0	0	0	0					Γ		0	0	•	•	•	•

_																											
	Mot	orola	SUE	3B	Fun	ktion												1	CF	PU	ΙT	УI	o E	3 /	6	80	9
	B&F	?	-В		ERB	- (M)	Σ	> E	ER	В																	
	Kurz	z																									
	Ad	ressie	runas	arten	/ Opco	ode	Γ			Α	dr	eſ	3v	or	wa	ah	le	n				О	-	_	P		
							L															О	1	Р	C	30	
	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
		4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Ρ	С	1	Υ	D	U		В	G	Н	I	Ν	Z	٧	С
		D0	F0	C0	E0		b	0	0	0	0	0	0	0	0	•	•	•	•	•	•	х	0	•	•	•	•

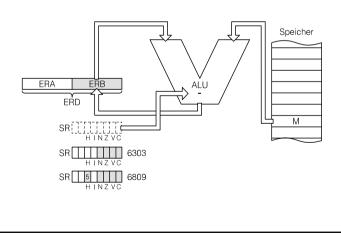
Der Inhalt der Speicherstelle M wird vom Ergebnisregister ERB subtrahiert. Das Ergebnis wird im ERB abgespeichert.



Mot	orola	SBC	СВ	Fun	ktion													CI	PU	J 1	у	р	A A	6	30	3
B&F	₹	В		ERE	3 - (M)	-	С	D	E	R	В															٦
Kur	z																									
۸4	roccio	runae	arten	,do	Γ			^	41	eſ	2 1/	^-	١	ah	ı۱۸	n				C)	P	G1(000	П	
Au	CSSIC	ungs	arterr	Opce	ue				^	uı	e.	3 V	U.	vv.	aıı		•••				C	,	Р	G-F	S	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	us	reg	ist	er
	3/2	4/3	2/2	4/2		Ε	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
	D2	F2	C2	E2		b	0	0	0	0	0	0	0	0		Γ					Ь	0	•	•	•	•

Mot	orola	SBC	В	Fun	ktion												1	CI	Pι	ΙŢ	yı	p E	3 /	6	80	9
B&F	t	В		ERE	3 - (M)	- 1	С	п	E	R	В															
Kurz	z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	۱۷	٥r	w	ah	le	n				0	<u> </u>	_	P		
7	00010	ungo	u. to	Орос	,uc	L			_	۵.	٠.	•	٠.	•••	u						0	<u> </u>	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
	4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
	D2	F2	C2	E2		О	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.



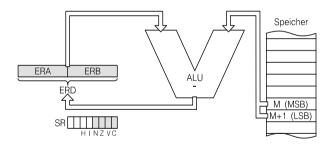
IVIOL	oroia	300	טכ	Full	KUOII													C I	- u	<u>' '</u>	у	μ,	٩,	0	JU	ᆀ
B&F	₹	-D		ERD) - (M:	M-	+1)	⇨	Ε	RI)														
Kur	z																									
Δdı	roccio	runae	arten /	Onco	nda	Г			Δ	dr	ام:	١٧	or	۱۸/	ah	ما	n				C)	P	G1 (000	╝
ζ	163316	ungs	ai teii /	Opce	ue					u	٠.	. ·	٠.	***	an						С)	Ρ	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Ρ	R	х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
	4/2	5/3	3/3	5/2		Ε	Α	М	F	Z	#	Ρ	С	1	Υ	D	U	!	В	G	Н	ī	Ν	Z	٧	С
	93	В3	83	A3							0	0	0	0							0	0	•	•	•	•

Eunktion

Motorolo SUBD

	_										 																									
	C	F	'	J 7	у	р	A	/ 6	30	3	Mote	orola	SUE	3D	Fun	ktion											•	CF	٧U	T	УĮ	ьВ	1	68	09	,
											B&R	1	-D		ERD	- (M:	M-	+1) 1	⇒	ER	D														
											Kurz	z																								
_	_				C	7	Р	G1	000)		occio	runac	arten	/ Once	do	Г			۸,	ire	o.,			. h	١	_			П	0	ī	CI	8	5	
eı	•				C)	P	G-	PC		Aui	62216	ungs	arterr	Opce	ue				AL	пe	13 V	Oi	w	111	ie	"				0		P	8	0	
ı	υ	!	В	G	SI	at	us	reg	gis	er	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	S	Т #	Р	R	х	Υ	D	U	!	В	G	St	atu	sre	gi	ste	r
1	U	!	В	G	Н	I	N	Z	٧	С		6/2	7/3	4/3	6+/2+		Ε	Α	М	F.	Z #	Р	С	1	Υ	D	U	!	В	G	Н	1	N.	z١	/ (5
I	T				0	0	•	•	•	•		93	В3	83	А3						0	0	0	0	•	•	•	•	•	•	0	0	•	•	•	ē

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



Mot	orola	SBA	4	Fun	ktion												1	CI	2	ľ	y	р	4	6	30	3
B&F	₹	A-B		ERA	- ERI	В	п	· E	R	Α																٦
Kur	z																									
٨٨	roccio	runae	arten	/ Once	ndo.	Γ			^	4,	٠,٠١	2 1/	or	١	ah	ı	n				O	7	P	G1(000	П
Au	CSSIC	ungs	arterr	Opce	Jue				~	uı	CI	3 V	U	vv.	aıı	116	••				O	7	Р	G-F	PC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	IS	reg	ist	er
1/1						Ε	Α	М	F	z	#	Р	С	I	Υ	D	U	!	В	G	Н	ī	N	Z	٧	С
10						Г	Г		Г			Г		П			П		П		0	0	•	•	•	•

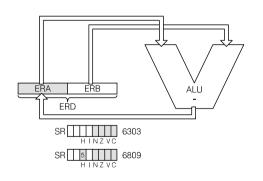
Mot	orola	SBA	١	Fun	ktion												1	CF	Pι	ΙT	УI	рE	3 /	6	80	9
B&F	₹	A-B		ERA	- ERI	В	₽	Е	R	Α																
Kur	Z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	ŝν	٥r	w	ah	le	n				0	_		P		
Au	000.0	· u.i.go	ui toii	Орос	Juc	L			_	۵.	٠.	•	٠.	•••		٠.	••				О)	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
		25/3				Е	Α	М	F	Z	#	Р	С	Ι	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
		BD FF 9A				Γ														П	×	0	•	•	•	•

Der Inhalt des Ergebnisregisters ERB wird vom ERA subtrahiert. Das Ergebnis wird im ERA abgespeichert.

PSH

RTS

! ,B



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

Der Inhalt von ERB wird auf den Stack

Verlassen des Unterprogrammes.

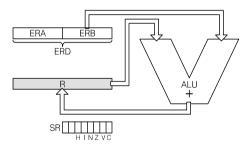
gelegt:
SP! - 1 ⇒ SP!
ERB ⇒ (SP!)

SUBA
! ,+
Der Inhalt der Speicherstelle, auf die der System-Stapelzeiger SP! zeigt, wird vom Ergebnisregister ERA subtrahiert. Danach wird SP! um 1 erhöht (Postinkrement):
ERA - (SP!)
SP! + 1 ⇒ SP!

Mot	orola	AB	(Fun	ktion													CI	PU	ΙT	уı	o A	۹/	6	30	3		Mot	or
B&F	₹	B+F	₹	ERE	8 + R	\Rightarrow	R																					B&F	₹
Kur	z																											Kur	z
Adı	ressie	rungs	arten	Орсс	de				Α	dı	eí	ŝν	or	w	aŀ	ıle	n				00	_		31(G-F				Ad	res
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atι	ısı	eg	jist	er		IMPL.	С
1/1						Ε	Α	М	F	z	#	Ρ	С	I	Υ	D	U	!	В	G	Η	1	Ν	Z	٧	С		3/1	
3A						Γ	Т	Γ	Τ	Γ	Γ	Г	Γ	Γ	Γ	Γ	Γ	Γ	Γ		0	0	0	0	0	0	ı	3A	Γ

Mot	orola	AB)	(Fun	ktion												1	CI	٦(J 7	у	рE	3 /	6	В0	9
B&F	₹	B+F	₹	ERE	3 + R	\Rightarrow	R																			
Kur	z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	اع:	ßν	٥r	w	ah	le	n				C	1		P		
	. 000.0	· ugo	u. to	Орос	Juc	L			_	۵.	٠.	•	٠.	•••	u.,		••				C)	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
3/1						Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
3A						Г	Г		Г		Г	Г	П	П				П	Г		ા	0	0	0	0	0

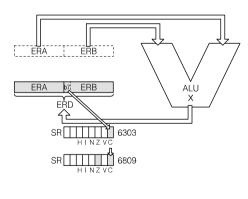
Der Inhalt von Ergebnisregister ERB und Indexregister R (X) wird addiert. Das Ergebnis wird im R (X) abgespeichert.



Mot	orola	MUI	-	Fun	ktion												1	CI	P	J T	y	o /	4 /	6	30	3
B&F	₹	A*B		ERA	X ER	В	ε	⇒	EF	RD)															
Kur	z																									
۸4	roccio	runae	arten	/ Once	,do	Г			^	4,	٠,٠١	2 . ,	or	١	ah	ı	n				C	T	P	G1 (000)
Kui	CSSIC	ungs	ai teii i	Орсс	ue				^	uı	CI	3 V	UI.	w	aii		•••				O	1	Ρ	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
7/1						Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
3D						Γ	Γ	Γ			Γ		Γ				Π)	0	0	0	0	•

	Mot	orola	MUI	L	Fun	ktion												1	CI	?(JΤ	УI	o E	B /	6	В0	9
ı	B&F	₹	A*B		ER#	XER	В	٤	⇒	EF	RD)															
ı	Kur	z			1																						
ı	Δd	roccio	runas	arten	/ Once	nde	Γ			Δ	dr	ام:	ßν	۸r	۱۸/	ah	ما	n				0	1	С	P	30	
ı	ť	163316	unga	arterr	Opce	Jue					u	٠.	3 4	٠.	***	an						О	•	Ρ	C	30	
ı	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
ı	11/1						Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
ı	3D						Г	П	П	П		Г	П		П					Г	П	0	0	0	•	0	•

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



3.7. VERGLEICHS- UND TESTBEFEHLE

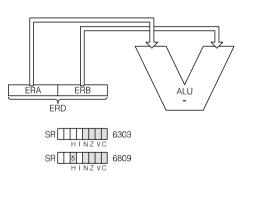
In diesem Abschnitt werden die Befehle beschrieben, die Inhalte von Registern oder Speicherstellen miteinander vergleichen (Subtraktion) oder auf ein bestimmtes Bitmuster testen (UND-Verknüpfung). Die Operationen werden durchgeführt, ohne das Ergebnis zu speichern. Es wird lediglich das Statusregister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

Matanala	Den		Beti	iebsart	
Motorola	B&R	PG1000	PG-PC	CP 80	PC 80
СВА	AVB	0	0	0	0
CMPA	CMP	0	0	0	0
CMPB	VB	0	0	0	0
CPX	VR	0	0	0	0
CPX#	VRK	0	0	0	0
CPY	VY				0
CPY#	VYK				0
BITA	В	0	0	0	0
BITB	BB	0	0	0	0
TIM	TIM		0		

Mot	orola	CB/	4	Fun	ktion													CF	2	J T	УI	0 /	۹/	6	30	3
B&F	₹	AVE	3	ERA	- ERI	В																				
Kur	z																									
Δdı	rossio	runae	arten	Once	nda	Γ			Δ	dr	ام:	۱۷	or	۱۸/	ah	ما	n				0	1	P	31 0	000	П
Au	63316	ungs	arterr	Opce	ue				_	u	٠.	3 V	٠.	***	a 1 1		•••				О	•	Р	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
1/1						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С
11						Γ	Γ														0	0	•	•	•	•

_							_											_	_								_	۰
ı	Mote	orola	CBA	١.	Fun	ktion												1	CI	2	J	УΙ	o E	В/	6	80	9	
	B&R	?	AVE	3	ERA	- ERI	3																					ĺ
	Kurz	z																										
I	۸۸۰	roccio	runae	arten	/ Once	ndo.	Г			^	4,	٠~١	ŝν	۸r		a h	ı	n				0	T	С	P	30		Ì
l	Aui	CSSIC	rungs	anten	Opce	Jue				_	uı	CI	3 V	UI.	vv	211	16					0	1	Р	C	30		
I	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	eı	•
I			25/3				Е	Α	М	F	Z	#	Р	С	-	Υ	D	U	!	В	G	Н	I	N	Z	٧	С	
I			BD FF 95				Γ						Γ									×	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.



6809:

Da der Befehl im 6809 nicht vorhanden ist, wird dieser durch einen Unterprogrammsprung 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.

IVIOL	UlUla	CIVII		Full	KUUII													Ç.		<u>, , </u>	УI	,	٦,	U.	30	٠
B&F	₹	СМ	P	ERA	4 - (M)																					
Kurz	z	٧																								
۸۵۰	ressie	, do	Γ			^	۸.	٠, ١	· · ·	or		- L	۱,	<u>_</u>				0	ī	P	G1 (000	,			
Aui	62216	rungs	Opco	oue				^	uı	eı	٥v	Oi	w	411	ie	"				О	1	Р	G-F	C		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
	3/2	4/3	2/2	4/2		Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Η	Ι	Ν	z	٧	С
	91	B1	81	A1		О	0	0	0	0	0	0	0	0	П				П	П	0	0	•	•	•	•

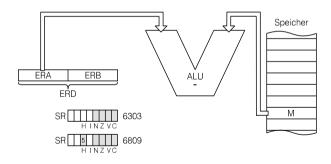
Funktion

Motorola CMPA

T	Mot	orola	СМІ	PA	Fun	ktion												1	CF	٧,	ΙŢ	y	рE	3 /	6	80	9
ſ	B&F	₹	СМІ	-	ERA	· - (M)																					
	Kur	z	٧																								
ı	Δd	roccio	nde	Γ			Δ	dr	ام:	ßν	۸r	۱۸/	ah	ما	n				O	7	С	Pδ	30				
L	Adressierungsarte				Орсс	Jue	L			_	u	٠.	3 4	٠.	***	an						U)	Р	C	30	
ſ	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
ſ		4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
Ī		91	B1	81	A1		0	0	0	0	0	0	0	0	0	•	•	•	•	•	•	X	ာ	•	•	•	•

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.

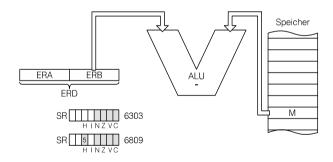
CPII Typ. A / 6303



Mot	orola	СМІ	РВ	Fun	ktion												1	CI	2	J T	УĮ	o /	4 /	6	30	3
B&F	₹	VВ		ERE	3 - (M)																					
Kur	z																									
۸,4	Adressierungsarten / Opcode								^	4	٠~١	2 . ,	or	١	ah	ı	n				0	T	P	G1 (000)
Au	Adressierungsarten / Opcode								^	uı	CI	3 V	UI.								О			G-F		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ISI	eg	ist	er
	3/2	4/3	2/2	4/2		Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С
	D1	F1	C1	E1		b	0	0	0	0	0	0	0	0							0	0	•	•	•	•

_					_													_									
	Mot	orola	СМІ	РВ	Fun	ktion												1	CF	P	ľ	уı	ρE	3 /	6	80	9
	B&F	₹	VВ		ERE	3 - (M)																					
	Kur	z																									
	Δd	ressie	nde	Γ			Δ	dr	ام:	3v	۸r	\A/:	ah	ما	n				0	1	С	P	30				
	Au	163316	runga	arterr	Орсс	Jue	L			_	u	٠.	3 V	٠.	***	a 1 1						О)	Р	C	30	
	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
		4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Р	С	-	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
		D1	F1	C1	E1		0	0	0	0	0	0	ာ	0	0	•	•	•	•	•	•	×	ာ	•	•	•	•

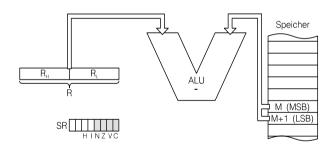
Der Inhalt des Ergebnisregisters ERB wird mit dem Inhalt der Speicherstelle M verglichen, indem eine Subtraktion (ERB - (M)) durchgeführt wird. Das Ergebnis beeinflußt das Statusregister, wird aber nicht abgespeichert.



Mot	orola	(CP)	(Fun	ktion												1	CI	?(J T	УI	p /	4 /	6	30	3
B&F	₹	VR		R - ((M:M+	1)																				
Kur	z																									
	roccio	,do	Γ			^	4	٠,٠١	2 1/	or	١	ah	ı	n				C	7	P	G1 (000	\Box			
Au	Adressierungsarten / Opcode								^	uı	CI	3 V	UI.	w	aii						С)	Ρ	G-F	c	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
	4/2	5/3	3/3	5/2		Ε	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Н	1	Ν	Z	٧	С
	9C	BC	8C	AC		Γ	Γ	Γ	Γ		Γ	0	0	0							0	0	•	•	•	•

	Mot	orola	СРХ	(Fun	ktion												1	CF	Pι	JΤ	УI	рE	3 /	6	В0	9
I	B&F	1	VR		R - (M:M+	1)																				
I	Kur	Z																									
	Δd	ressie	nde	Γ			Δ	dr	ام:	3v	۸r	\A/:	a h	ما	n				О	7	С	P	30				
	Au	63316	unga	arterr	Орсс	Jue				_	uı	٠.	3 V	٠.	***	a 1 1						О)	Р	C	30	
	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
1		6/2	7/3	4/3	6+/2+		Ε	Α	М	F	Z	#	Ρ	С	1	Υ	D	U		В	G	Н	ı	N	Z	٧	С
		9C	BC	8C	AC		Γ						0	0	0	•		•	•	•	•	0	0	•	•	•	•

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.

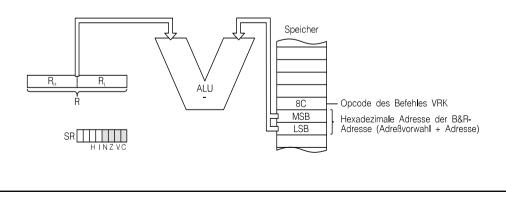


Mot	orola	CP	(#	Fun	ktion												1	CF	2	JT	УI	0 /	4 /	6	30	3
B&F	₹	VR	`	R - I	VI																					٦
Kurz	z																									
Δdı	ressie	runae	nda	Γ			Δ	dr	ام:	٠,	۸r	۱۸/	ah	le	n				О	1	P	G1 (000	Ĭ		
Aui	COOL	ue	L			^	u	٠.	3 V	٠.	***	u						О	•	Ρ	G-F	PC				
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
			3/3			Ε	Α	М	F	z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	N	z	٧	С
			8C			b	0	0	0	0		0	0								0	0	•	•	•	•

I	Mot	orola	CPX	(#	Fun	ktion													CI	٦,	JΤ	у	p E	3 /	6	В0	9
1	B&F	₹	VR	`	R - I	VI																					
l	Kur	z																									
]	Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	ام	3v	or	w	ah	le.	n				C	1		P	_	
]	^~	163316	unga	arterr	Орсс	Jue	L			_	uı	٠.	3 V	٠.	***	an						С)	Р	C	30	
7	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
1				4/3			Е	Α	М	F	Z	#	Р	С	-	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
1				8C			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.

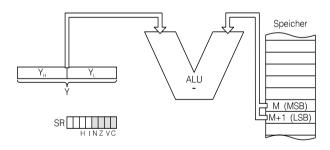
Der Befehl entspricht dem VR #xxxx Befehl. Die Adresse, die der Anwender in die AWL-Eingabezeile eingibt, ersetzt das PROgrammierSYStem durch die effektive Adresse (hexadezimaler Wert), die die B&R Adresse (= Adreßvorwahl + Adreßteil) im Speicher der SPS hat.



Mot	orola			Fun	ktion												•	CF	2	ΙT	УΙ	o /	A /	63	30:	3
B&F	₹																									٦
Kur	z																									١
Δdı	ressie	runas	arten /	/ Onco	nde	Г			Δ	dr	e í	۱۷	or	w	ah	le	n						PC	310	00	┒
, Au	000.0	ungo	u	Орос	uc				_	۵.	٠.	•	٠.	•••			••						P	G-P	C	_
							П		I .	_		_	-			_			I		<u> </u>	-4.	usr	~~	nt.	_
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	1	l٥	F	S	Т	#	Р	R	Х	Y	D	υl	!	В	G	ગ	au	uSi	eg	Su	er

Mote	orola	CPY	,	Fun	ktion												1	CI	٧,	ΙŢ	уı	o E	3 /	6	80	9
B&R	1	VY		Y - (M:M+	1)																				
Kurz	<u> </u>																									
٨٨	neein	runae	arten	/ Once	odo	Γ			^	d١	٠,	ßν	^-	١	ah	ı	n					Τ	С	P	30	
Aui	CSSIC	ungs	arterr	Opce	Jue				^	uı	C1	3 V	U.	w	all	16					О	7	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
	7/3	8/4	5/4	7+/3+		Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
	10 9C	10 BC	10 8C	10 AC								•	•	•	•		•	•	•	•	0	0	•	•	•	•

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.



WOT	oroia			run	Ktion													u	-	' '	У	p /	4 /	ю.	30	ગ
B&F	₹																									
Kur	z																									-
۸۵۰	roccio	runae	arten	Once	,do	Γ			_	4,	٠, ۲	2.,	or	\A/-	ah	10	n				Г	T	P	G10	000	П
Aui	CSSIC	ungs	arterr	Opce	ue				^	uı	CI	3 V	U.	vv.	aıı	116	••				Г	Τ	Р	G-F	SC	٦
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	JSI	reg	ist	er
						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
						Γ															Г					П

Franklikan

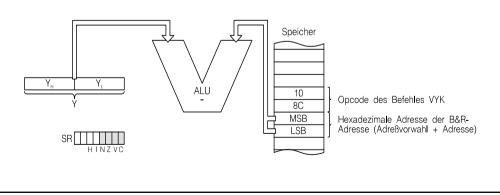
Matanala

T	Mot	orola	CPY	' #	Fun	ktion													CI	٦,	JΤ	y	o I	в/	6	80	9
Γ	B&R	t	VYK	7	Y - I	И																					
ſ	Kurz	z																									
Г	۸۵۰	roccio	runae	arten	/ Once	ndo.	Г			_	4,		ßν	۸r	\A/-	a h	ı۸	n				Г	T	С	P	30	
L	Aui	CSSIC	ungs	anten	Орсс	Jue				^	uı	C1	3 V	UI	w	all	16	•••				O	ŀ	Р	C	30	
Г	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
				5/4			Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	1	N	z	٧	С
				10 8C			•	•	•	•	•		•	•						•	•	0	0	•	•	•	•

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.

ODU T. ... A / COOO

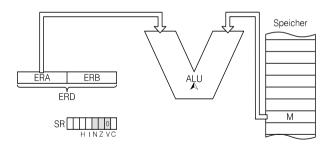
Der Befehl entspricht dem VY # xxxx Befehl. Die Adresse, die der Anwender in die AWL-Eingabezeile eingibt, ersetzt das PROgrammierSYStem durch die effektive Adresse (hexadezimaler Wert), die die B&R Adresse (= Adreßvorwahl + Adreßteil) im Speicher der SPS hat.



Mot	orola	BIT	A	Fun	ktion												1	CI	PU	J	y	р	A /	6	30	3
B&F	₹	В		ERA	∧ (M))																				
Kur	z																									
٨٨	roccio	runae	arten	/ Once	,do	Γ			^	4,	٠, ۲	2 1/	or	١	ak	ı	n				O	7	P	G1(000	П
ť	63316	ungs	arterr	Орсс	ue				^	uı	e.	3 V	UI.	w	aı	iie	•••				O)	P	G-F	c	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Ρ	R	х	Υ	D	U	!	В	G	St	atı	usi	reg	ist	er
	3/2	4/3	2/2	4/2		Ε	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
	95	B5	85	A5		Ь	0	0	0	0	0	0	0	0	Г						0	0	•	•	•	0

Т	Mot	orola	BIT	A	Fun	ktion												1	CF	٦,	JΤ	УI	рE	3 /	6	80	9
ſ	B&F	ł	В		ERA	√ (M))																				
Ī	Kur	z																									
ı	Δd	rossio	runas	arten	/ Once	nde	Γ			Δ	dr	ام:	ßν	۸r	۱۸/	ah	ما	n				О	7	С	Pδ	30	
L	Au	63316	unga	arterr	Орсс	Jue				_	u	٠.	3 4	٠.	***	an						О)	Р	C	30	
-[IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
Ī		4/2	5/3	2/2	4+/2+		Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
I		95	B5	85	A5		0	0	0	0	0	0	0	0	0	•	•	•	•	•	•	0	ာ	•	•	•	0

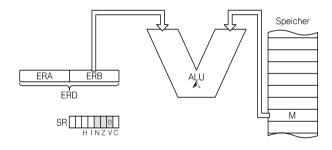
Der Inhalt des Ergebnisregisters ERA wird mit dem Inhalt der Speicherstelle M UND-verknüpft. Das Ergebnis beeinflußt das Statusregister, wird aber nicht abgespeichert.



Mot	orola	BITI	В	Fun	ktion												1	CF	2	J T	УI	o /	۸ /	6	30	3
B&F	₹	ВВ		ERE	3 ^ (M))																				
Kur	z																									
Δdı	roccio	runas	arten	Onco	nda	Г			Δ	dr	ام:	۱۷	or	۱۸/	ah	ما	n				0	1	P	31 (000	7
Adressieru		ungs	arterr	Opce	ue				_	u	٠.	3 V	٠.	***	an						О	•	Р	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
	3/2	4/3	2/2	4/2		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С
	D5	F5	C5	E5		0	0	0	0	0	0	0	0	0							0	0	•	•	•	0

	Mote	orola	BITI	В	Fun	ktion												1	CI	٦,	ΙŢ	УI	o E	3 /	6	80	9
	B&R	t	ВВ		ERE	3 ^ (M))																				
	Kurz	z																									
ı	Δdı	rossio	runas	arten	/ Once	nde	Γ			Δ	dr	ام:	ŝν	۸r	\A/:	a h	ما	n				0	1	С	Pδ	30	П
ı	7.4.	00010	ungo	ui toii i	Орос	Juc	L			_	۵.	٠.	•	٠.	•••	٠.,		••				О	1	Ρ	C	30	
	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
		4/2	5/3	2/2	4+/2+		Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
		D5	F5	C5	E5		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.



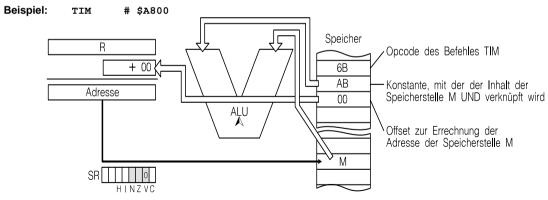
Mot	orola	TIM		Fun	ktion													CF	٧	Т	УI	0 /	۹ /	6	30	3
B&F	₹	TIM	(R +	Offse	t)	^	IN	١N	ı																П	
Kurz	z																									
۸۵۰	roccio	runae	arten	/ Once	,do	Γ			^	di	٠~١	21/	^-	۱	ah	le	n						P	G1 (000	Л
Aui	CSSIC	ungs	arterr	Орсс	ue				^	uı	e,	3 V	U.	w	211	116					0	7	Ρ	G-F	S	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ISI	eg	ist	er
			5/3			Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	1	Ν	Z	٧	С
			6B			Γ	Г				0										0	0	•	•	▼	0

Mote	orola			Fun	ktion												1	CF	2U	ΙŢ	yı	ρE	3 /	68	80	9
B&R	ŧ					_	_	_	_			_			_	_	_		_		_		_	_	_	
Kurz	z			L																						
Δd	ressie	runas	arten /	/ Once	ode	Γ			Δ	dr	eſ	ßν	or	w	ah	le	n		_			I	_	Pε		
	000.0.	ungo	uc ,	Opti	,40	L			-	٠.	٠.	•	٠.	•••	٠								P	C8	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	П	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atu	ısr	eg	ist	er
						Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	П	N	z	٧	С
						Г	Г	Г	Г	П		П	П			П	П	П			П	П	П	П	Г	Г

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)



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.

Matauala	D.O.D.		Betr	iebsart	
Motorola	B&R	PG1000	PG-PC	CP 80	PC 80
INC	INC	0	0	0	0
INCA	IA	0	0	0	0
INCB	IB	0	0	0	0
INX	IR	0	0	0	0
INS	IS	0	0	0	0
DEC	DEC	0	0	0	0
DECA	DA	0	0	0	0
DECB	DB	0	0	0	0
DEX	DR	0	0	0	0
DES	DS	0	0	0	0

B&F	₹	INC		(M)	+1 ⇔	(1	M)																			
Kur	z																									
	roccio	runae	arten	/ Once	,do	Γ			^	4	eſ	2 . ,			ah	ı	n				0	1	Р	G10	000	
Au	CSSIC	ungs	arterr	Орсс	ue				~	uı	e.	3 V	U	w	aii	116	••				0	ī	Р	G-F	S	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	us	reg	iste	r
		0.40		0/0		F	Α.		F	7	44	_	_	١.	V	٥			2	(-	NI	7	\/	$\overline{}$

Funktion

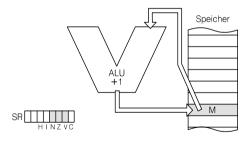
Motorola INC

Mot	orola	INC		Fun	ktion												1	CF	Pι	ΙT	УI	o E	3 /	6	В0	9
B&F	ł	INC		(M)	+1 ⇨	(1	VI)																			
Kur	z																									
Δд	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	<u>ام</u>	3v	٥r	w	ah	le	n				О	1		P		
~~	000.0	ungo	ui toii i	Орос	Juc	L			_	۵.	٠.	•	٠.	•••	u.,		••				О	1	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısı	eg	ist	er
	6/2	7/3		6+/2+		Е	Α	М	F	Z	#	Ρ	С	T	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
	0C	7C		6C		Γ	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.

CPU Typ A / 6303

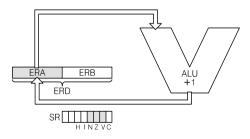


Mot	orola	rola	INC	A	Fun	ktion												1	CP	U	1	уŗ) /	A /	63	303
B&F	₹		IA		ERA	+1 =	⇒	E	RA																	
Kur	z																									
	roccio	ccior	unae	arten	Once	,do	Γ			^	dr	~(2 . ,	or	١	ah	ı	n				0	1	PG	10	00
Au	CSSIC	33161	ungs	ai teii i	Opce	ue	L			^	uı	CI	3 V	UI.	w	aii						0	I	PC	-P	C
IMPL.	DIR.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Ρ	R	Х	Υ	D	U	! [3	G	Sta	atı	usre	egi	ister

Mot	orola	INC	Α	Fun	ktion												1	CI	2 U	T	УĮ	o E	3 /	6	30	9
B&F	₹	IA		ERA	1 + 1	⇒	EI	R/	١																	
Kur	z																									
Δd	ressie	runas	arton	/ Once	nde	Γ			Δ	dr	ام:	ŝν	۸r	۱۸/	ah	ما	n				0	7	С	P	30	
Au	163316	unga	arterr	Орсс	Jue	L			_	uı	٠.	3 .	٠.	***	u						О	•	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atι	ısr	eg	ist	er
2/1						Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
4C						Г	Г	Г	П			Г	Г	П		Г					0	0	•	•	•	0

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.

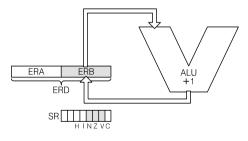


Mot	orola	INC	В	Fun	ktion												-	CF	٦,	ΙŢ	y	o /	۸/	6	30	3
B&F	₹	IB		ERE	3 + 1	⇒	EI	RE	3																	
Kur	z																									
Δdı	ressie	runas	arten	Onco	nde	Γ			Δ	dr	اع:	٩v	or	w	ah	ıle	n				O	1	PC			_
, Au	. 000.0	ungo	ui toii ,	Орос	uc	L			_	۵.	٠.		٠.	•••			••				С	1	P	G-F	c	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	1	0	F	s	Т	#	Р	R	Х	Υ	D	U	-	В	G	St	atı	ısr	eg	ist	er
1/1						Ε	Α	М	F	z	#	Р	С	I	Υ	D	U	!	В	G	Н	1	Ν	Z	٧	С
5C						Г	Γ	Г	Г	П	Γ	Γ	П	Г	П		П	П	Г	П	0	0	•	•	•	0

_			_		_																						
	Mot	orola	INC	В	Fun	ktion												1	CF	PU	J T	УI	o E	3 /	68	30	9
	B&F	t	IB		ERE	3 + 1	⇒	EF	₹E	3																	
	Kurz	z																									
	Δd	rossio	runas	arten	/ Once	nde	Γ			Δ	dr	ام	3v	or	M/:	a h	ما	n				0	7	С	Pε	30	
	, a	63316	unga	arterr	Орсс	Jue	L						•	•				•••				О			CE		
	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
	2/1						Е	Α	М	F	Z	#	Ρ	С	Т	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
	5C						Г	П	Г	П					٦							0	0	•	•	•	0

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.

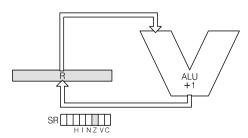


Motorola	INX	Funktion	CPU Typ A / 6303
B&R	IR	R + 1 ⇒ R	
Kurz			

٨٨	roccio	runae	arten	Once	ndo.	Г			^	4.	eí	2.,			a h	ı	n				O	ī	PC	G10	000	П
Au	CSSIC	ungs	arterr	Opce	ue				~	uı	CI	3 V	U.	vv.	aıı	116	••				O	ī	P	G-F	SC	П
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
1/1						Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Ι	1	N	Z	٧	С
08						Γ	Γ														0	0	0	•	0	0

Mote	orola	INX		Fun	ktion												1	CF	? (ĮΤ	·уІ	o E	3 /	6	80	9
B&R	1	IR		R+	1 ⇒ F	₹																				
Kurz	z																									
Δdı	essie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	ŝν	٥r	w	ah	le	n				0	1	С	P	30	
Au.	000.0	ungo	u. to	, opoc	Juc	L			_	۵.	٠.	•	٠.	•••	u.,		••				О	1	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
				5/3		Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	N	z	٧	С
				30 88 01		Г	Г		Г			П	П						П		0	0	0	•	0	0

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



6809:

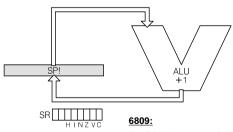
Der Befehl "IR" existiert nicht im Grundbefehlssatz des 6809. Er wird durch den Befehl "LER I 001" simuliert.

Motorola	INS	Funktion		CPU Typ A / 6303
B&R	IS	S+1 ⇒ S	•	
Kurz				
				DC4000

٨٨	roccio	runae	arten /	Once	ndo.	Г			^	4	eí	2.,			a h		n				O	7	P	31 (000	П
Au	CSSIC	ungs	ai teii /	Opce	ue				~	u	CI	3 V	U.	vv.	aıı		•••				O	7	Р	G-F	'n	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
1/1						Е	Α	М	F	z	#	Р	С	I	Υ	D	U	!	В	G	Н	Ι	Ν	Z	٧	С
31																					0	0	0	0	0	0

Ī	T	Mot	orola	INS		Fun	ktion												1	CI	٦,	JΤ	Уļ	o E	3 /	68	В0	9
1		B&F	₹	IS		S+	1 ⇒ 8	3																				
l		Kur	z			1																						
1	ı	Δd	ressie	runas	arton	/ Once	nde	Г			Δ	dr	ام	21/	۸r	\A/:	ah	ما	n				0	1	С	Pε	30	
l	L	Au	163316	unga	arten	Opce	Jue	L			_	uı	٠.	3 V	٠.	***	an						О	1	Ρ	CE	30	
1	-[IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	1	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	eı
1	[5/3		Е	Α	М	F	Z	#	Ρ	С	-	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С
1	- [32 F8 01		Г	Г	Г	Г	П					П					П	\circ	0	0	0	0	c

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



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

Motorola	DEC	Funktion		СРО ТУР	A / 6303
B&R	DEC	(M) - 1 ⇒	(M)		·
Kurz					
Adrossio	rungsarten	/ Oncodo	Adreßvorwahlen	0	PG1000
Auressie	ungsanten	Opcode	Aureisvorwanien	O	PG-PC

Front of the co

IND.

6/2

Maria de DEO

DIR

EXT. IMMED.

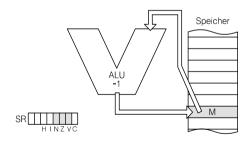
6/3

N	/lot	orola	DEC	;	Fun	ktion													CI	9٤	ľ	уı	ρE	3 /	6	80	9
В	8&R		DEC	;	(M)	-1 ⇨	(N	A)																			
К	(urz																										
	Δdı	occio	runas	arten	/ Once	nde	Г			Δ	dr	ام	٠.	۸r	\A/:	ah	ما	n				0	1	С	P	30	
Ľ	_u	63316	unga	arterr	Орсс	Jue				_	uı	٠.	3 V	٠.	***	a 1 1						О)	Р	C	30	
IME	PL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
		6/2	7/3		6+/2+		Ε	Α	М	F	Z	#	Р	С	_	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
		0A	7A		6A			0	0	0			0	0	0	•	•	•	•		•	0	0	•	•	•	0

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.

ODU T. ... A / COOO



B&F	₹	DA		ER#	1 ⊏	>	EF	RA																		١
Kurz	z																									
۸4،	roccio	runae	arten	/ Once	,do	Γ			^	4	٠~١	2 . ,	or		a h	ı	n				0	7	P	310	000	
Aui	CSSIC	ungs	ai teii i	Орсс	ue				_	uı	CI	3 V	UI.	vv	211	iie					О	ŀ	P	G-P	'n	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	egi	ist	er
1/1						Ε	Α	М	F	Z	#	Р	С	_	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С

Funktion

Motorola

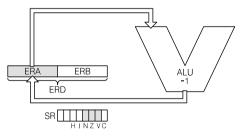
DECA

	Mot	orola	DEC	A	Fun	ktion												•	CF	Pι	ΙT	УĮ	o E	3 /	68	В0	9
	B&F	t	DA		ERA	\-1 ⊏	>	ER	A																		
	Kur	z																									
ı	Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	eſ	3v	or	w	ah	le.	n				0	1		Pε		
	Au	00010	ungo	ui toii i	Орос	Juc	L			_	۵.	٠.	•	٠.	•••		٠.					0	1	P	C	30	
	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
	2/1						Ε	Α	М	F	Z	#	Ρ	С	-	Υ	D	U		В	G	Ι	I	N	Z	٧	С
	4A						Г	Г	П	П												0	0	•	•	•	ာ

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.

CPU Typ A / 6303



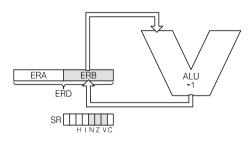
Mot	orola	DE	СВ	Fun	ktion											С	Pι	J٦	Τур	Α	/ 6303
B&F	₹	DB		ERE	3-1 ⊏	>	EF	≀В													
Kur	z																				
	roccio	runas	arten	/ Once	,do	Γ			١,	rol	2.,	orv	12	hl	'n				0	П	PG1000
Au	CSSIC	unga	arterr	Орсс	ue	L			١u	161	3 V	OI V	vai		711				0		PG-PC
IMDI	DIB	EVT	IMMED	INID	DEI	L	1~	ι-Ta	Ī	- 14	ь	ьΤ	/ _	1	To	Ιī	L	0	Sta	ıtıı	ereniete

Mot	orola	DEC	В	Fun	ktion												•	CI	٧,	ΙŢ	УI	o E	3 /	6	30	9
B&F	₹	DB		ERE	3-1 ⊏	>	EF	RВ																		
Kurz	z																									
Δ.	ressie	runas	arton	/ Once	nde	Γ			Δ	dr	ام	21/	۸r	١٨/:	a h	ما	n				О	1	С	Pδ	30	Т
^~	163316	ungs	arterr	Орсс	Jue	L			_	uı	٠.	3 V	٠.	***	a 1 1						О	•	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	ļ	В	G	St	atι	ısr	eg	ist	er
1/1						Ε	Α	М	F	Z	#	Р	С	Ι	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
5A						Г	Г	Г	П		Г	Г	Г			П			П		0	0	•	•	•	0

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

5A

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

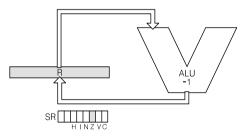


Motorola	DEX	Funktion		CPU Typ	A / 6303
B&R	DR	R - 1 ⇒ R			
Kurz					
Adrossio	rungsarten /	Oncodo	Adreßvorwahlen	0	PG1000
Auressie	ungsanten	Opcode	Aureisvorwanien		PG-PC

roccio	runae	arton	Once	do	ı			Δ	di	ام.	21/	۸r	\A/-	ah	ما	n				C	<u> </u>	P	31 ()00)
COOL	ungs	arterr	Opce	ue	L			_	u	٠.	3 V	٠.	***	an						С	<u> </u>	Ρ	G-I	'n	
DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	usi	eg	ist	er
					Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	Ι	N	Z	٧	С
					Γ				Г										Г	0	0	0	•	0	0
					ressierungsarten / Opcode DIR. EXT. IMMED. IND. REL.				· · · · · · · · · · · · · · · · · · ·			· · · · · · · · · · · · · · · · · · ·		·							·	<u> </u>	ressierungsarten / Opcode Adreßvorwahlen O P	ressierungsarten / Opcode Adreßvorwahlen O PG-F	O PG-PC

	Mot	orola	DEX	(Fun	ktion													CI	Pι	JΤ	УI	p E	3 /	68	30	9
	B&F	₹	DR		R - ′	1 ⇒ R																					
	Kur	z																									
		ressie	runae	arton	/ Once	odo	Γ			^	dr	٠~١	2.,	^-	۱۸/	ah	ı	n				0	7	С	Р8	30	_
	Au	CSSIC	ungs	arten	Opce	Jue	L			^	ui	CI	3 V	UI	w	aii	iie	•••				О			C 8		
	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	egi	ist	e
ı					5/3		Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	ı	N	z	٧	C
1					30 88 FF		Г	Г	Г	П				Г	П				Г	Г		0	0	0	•	0	C

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



6809:

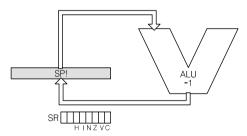
Der Befehl "DR" existiert nicht im Grundbefehlssatz des 6809. Er wird durch den Befehl "LER I -001" simuliert.

Motorola	DES	Funktion	CPU Typ A / 6303
B&R	DS	S-1 ⇒S	
Kurz			

٨٨	raccia	runaa	arten	/ Once	do					۵.	eí	, ,			- h		_				O	1	P	G10)00	П
Au	essie	rungs	arterr	Opco	ue				^	uı	eı	٥v	Oi	w	all	iie	"				O	1	Р	G-F	C	٦
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	JSI	reg	ist	er
1/1						Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
34						Γ	Γ														0	0	0	0	0	0

I	Mote	orola	DES	;	Fun	ktion												-	CF	2	ΙT	УI	o E	3 /	6	80	9
I	B&R	}	DS		S - 1	⇒ S																					
	Kurz	z																									
ſ	Δdı	neein	runae	arton	/ Opco	nda	Г			Δ	dr	۵.	R v	۸r	۱۸/	ah	ما	n				0	1	С	Pδ	30	
1	Aui	63316	ungs	arterr	Орсс	, ac				_	u	٠.	3 4	٠.	***	u		••				О	•	Р	C	30	
ſ	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
ſ					5/3		Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
[32 E8 FF																	0	0	0	0	0	0

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

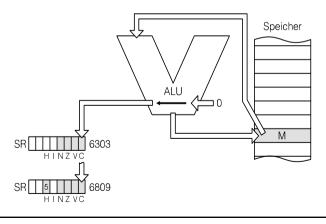
In diesem Abschnitt werden die Befehle beschrieben, die den Inhalt von Registern oder Speicherstellen um 1 Bit nach links/rechts schieben bzw. rotieren.

	505		Betr	iebsart	
Motorola	B&R	PG1000	PG-PC	CP 80	PC 80
ASL	SL	0	0	0	0
ASLA	SLA	0	0	0	0
ASLB	SLB	0	0	0	0
ASLD	SLD	0	0	0	0
LSR	SR	0	0	0	0
LSRA	SRA	0	0	0	0
LSRB	SRB	0	0	0	0
LSRD	SRD	0	0	0	0
ROL	SLI	0	0	0	0
ROLA	RLA	0	0	0	0
ROLB	RLB	0	0	0	0
ROR	SRE	0	0	0	0
RORA	RRA	0	0	0	0
RORB	RRB	0	0	0	0

Mot	orola	ASL	-	Fun	ktion												(CF	P U	ΙT	УI	0 /	A /	6	30	3
B&F	₹	SL		Sch	iebe lı	nh	al	t١	/ 0	n	M							٦.	4	Ŧ	1	Ţ	+	7	-	-0
Kurz	z			um	1 Bit n	ıa	ch	li	nk	s								C	d	7				d	j	
Δdı	ressie	runas	arten	/ Onco	um 1 Bit nach links Dpcode Adreßvorwal												n				C)			000	╗
Au.	000.0	ungo	ui toii ,	Орос	uc	L			_	۵.	٠.	"	٠.	•••			••				O)	Ρ	G-F	c	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	1	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	usi	eg	ist	er
		6/3		6/2		Ε	Α	М	F	z	#	Р	С	ı	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
		78		68		Г	0	0	0			0	0	0							0	0	•	•	•	•

Mot	orola	ASL		Fun	ktion												1	CI	٦,	ΙŢ	y	o E	3 /	6	80	9
B&F	₹	SL		Sch	iebe lı	٦h	al	t١	/ 0	n	М							Ī.		Ŧ	F	Ţ	Ŧ,	7	7	-0
Kur	urz um 1 Bit nach links															С	(17				d	ď			
Δ.	Kurz um 1 Bit nach links Adressierungsarten / Opcode Adreßvorwah												ما	n				V)	С	P	30				
L	163316	unga	arterr	Орсс	Jue				_	uı	٠.	3 V	٠.	***	an						V)	Р	С	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
	6/2	7/3		6+/2+		Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
	08	78		68			0	0	0			0	0	0	•	•	•	•		•	×	0	•	•	•	•

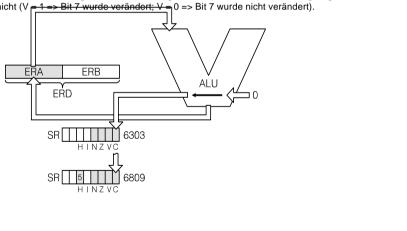
Der Inhalt der Speicherstelle M wird um 1 Bit nach links geschoben. Bit 0 wird durch log. 0 ersetzt und Bit 7 wird in das Carry-Flag geschoben.



Mot	orola	ASL	Α	Fun	ktion												(CI	٦,	ı	уı	0 /	4 /	6	30	3
B&F	₹	SLA	`	Sch	iebe Ir	۱h	al	t١	/ 0	n	EF	RA						7	-[Ŧ	ľ	1	+	Ŧ	F	-0
Kurz	z	um 1 Bit nach links																С	d	7				d	õ	
Δdı	ressie	um 1 Bit nach links														le	n				U)			000	╗
Au.	000.0	ungo	ui toii /	Opoc	uc				_	۵.	٠.	•	٠.	•••			••				С)	Ρ	G-F	'n	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	JSI	eg	ist	er
1/1						Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
48						Г	Г		П			Г									0	0	•	•	•	•

Mot	orola	SLA Schiebe Inhalt von ERA um 1 Bit nach links																CI	9٤	J T	y	ρE	3 /	6	80	9
B&F																		Π		Ŧ,	F	Ţ	+	7	7	- (
Kurz	z		Schiebe Inhalt von ERA um 1 Bit nach links arten / Opcode Adreßvo															C	Ċ	17			_	d	ď	
Δd	ressie	runas	arten	/ Once	nde	ebe Inhalt von ERA Bit nach links Bit nach links Adreßvo									ah	le	n				$^{\circ}$)		P		
'		90	u	Орос		L			•		٠.	••	٠.	•••			••				C)	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	1	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
2/1						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
48						Γ	Π	Γ	Γ	Γ	Γ	Γ	Γ	Г	Г	Г		Γ	Г	П	x	0	•	•	•	•

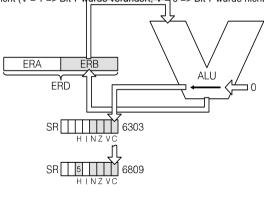
Der Inhalt des Ergebnisregisters ERA wird um 1 Bit nach links geschoben. Bit 0 wird durch log. 0 ersetzt und Bit 7 wird in das Carry-Flag geschoben.



Mot	orola	ASL	В	Fun	ktion													С	Pι	ΙŢ	УĮ) <i>A</i>	۸/	6	30	3	Γ	Mot	orola	Γ
B&F	₹	SLE	3	Sch	iebe l	nh	ıal	lt '	vo	n	Е	RE	3					Π		Ŧ	ľ	Ī	Ę	7	7	- 0		B&F	ł	Γ
Kurz	z			um	1 Bit r	na	ch	ı	in	ks	3							C	Ċ	7			_	d	Ő			Kurz	z	Г
Adı	ressie	rungs	arten	Орсс	ode		halt von ERB ach links Adreßvorw							ał	ηle	en				00	_			C 000	_		Adı	ressie	ru	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	- #	Р	R	x	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er	1	IMPL.	DIR.	E
1/1						Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	Ν	z	٧	С		2/1		Ī
58						Г				Γ		Г		Γ		Γ	Π				0	0	•	•	•	•	1	58		Ē

Mot	orola	ASL	В	Fun	ktion												1	CF	٦,	ΙŢ	УI	o E	3 /	6	80	9
B&F	₹	SLB	-	Sch	iebe lı	۱h	al	t١	/0	n I	EF	RB						Π.	4	Ŧ,	ľ	Ţ	Ŧ.	7	7-	-0
Kur																		C	(17	_			d	ď	
Δ.	urz um 1 Bit nach links dressierungsarten / Opcode Adreßvorwa														ah	ما	n				V)	С	P	30	
L	163316	unga	arterr	Орсс	Jue				_	uı	٠.	3 V	٠.	**	anı						U)	Р	С	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
2/1						Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
58																					×	0	•	•	•	•

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.

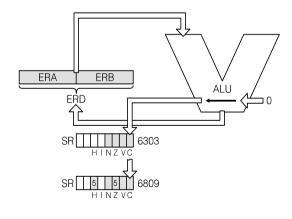


Mot	orola	ASL	D	Fun	ktion													CI	PU	JΤ	y	р	Α/	6	30	3
B&F	₹	SLE	•	Sch	iebe lı	٦h	al	t١	0	n	EF	RD	٦	-	1		-	+	1	ŀ	-	+	1	1	7	-0
Kurz	z			um	1 Bit r	a	ch	li	nŀ	s			C	(115									(<u> 10</u>	
۸ ۵۰	roccio	runae	arten /	/ Once	,do	Г			^	4,	۰~۱	2 . ,	or	١	ah	ı	n				O)	P	G1 (000	Л
Aui	CSSIC	ungs	ai teii /	Opce	ue				^	uı	CI	3 V	U.	w	aıı		••				U	7	Р	G-F	PC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	usi	eg	ist	er
1/1						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	ı	N	z	٧	С
05																					0	0	•	•	•	•

Mot	orola	ASL	D	Fun	ktion												1	CF	2	Ţ	УI	o E	3 /	68	30	9
B&F	ł	SLD)	Sch	iebe lı	nh	al	t١	0	n I	EF	RD	۱ [7	-	1	۲,		1	7	1	7.			7	-(
Kur	Z			um	1 Bit r	ıa	ch	li	nŀ	s			-	5	d15									-	<u>10</u>	
Δd	ressie	runas	arton	/ Once	nde	Γ			Δ	dr	ام	٠.	۸r	۱۸/	ah	ما	n				V	7	С	Pε	30	
L	63316	unga	arterr	Орсс	Jue				_	uı	٠.	3 V	٠.	***	an						U)	Р	CE	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
4/2						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С
58 49																					X	0	•	×	•	•

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 aeschoben.

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



6809:

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

Der Inhalt von ERB wird um 1 Bit nach links ST.B

geschoben. Bit 0 wird durch log. 0 ersetzt. Bit 7

wird in das Carry-Flag geschoben.

Der Inhalt von ERA wird um 1 Bit nach links RT.A geschoben. Das Carry-Flag (=> Bit 7 von ERB) wird ins Bit 0 geschoben. Bit 7 wird in das Carry-

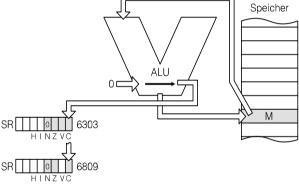
Flag geschoben.

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

Mot	orola	LSR	1	Fun	ktion												•	CI	٦(JΤ	УI	0 /	4 /	6	30	3
B&F	₹	SR		Sch	iebe lı	nh	al	t١	/ 0	n	M							0-	4	7-	Ţ	1	_	+	F	П
Kur	z			um															C	17				d	j	С
٨٨	roccio	runae	arton	Once	um 1 Bit nach rechts pcode Adreßvorwahl																С	7	P	G1 (000	П
ť	CSSIC	ungs	ai teii i	Opce	ue				^	uı	CI	3 V	UI.	w	aii						U)	Ρ	G-F	Š	⅃
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Ρ	R	х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
		6/3		6/2		Ε	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
		74		64		Γ	0	0	0			0	0	0							0	0	▼	•	•	•

Mot	orola	LSF	ł	Fun	ktion													CI	Pι	J 1	у	рl	B /	6	80	9
B&F	₹	SR		Sch	iebe l	nh	al	t١	vo	n	M							0-	+	7	-	1	-	7	7	₹
Kurz	z			um	1 Bit r	na	ch	r	ec	ht	s								(17				d	Ō	C
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	·el	R٧	٥r	w	ah	le	n				0)	С	P	30	
_ ^~	. 000.0	· unge	ar torr	, Opor	Juc	L				۵.	٠.		٠.	•••	u		••				C)	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	1	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
	6/2	7/3		6+/2+		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
	04	74		64		Γ	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.

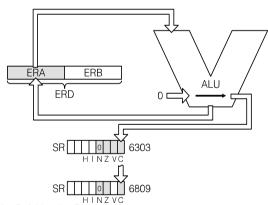


6303:

Mot	orola	LSF	RA	Fun	ktion													CF	·υ	Т	Уľ	,	۱,	63	30	3
B&F	₹	SRA	4	Sch	iebe lı	nh	al	t١	0	n	EF	RA						0-	┩	-		-	۲,	T	١.	Г
Kur	z			um	1 Bit r	ıa	ch	re	ec	ht	s								ď	7				dC	ĺ	C
Αdı	ressie	runas	arten	/ Once	ode	Γ			Δ	dr	eſ	۱۷	or	wa	ah	le	n				0	1	PG			_
1		90	u	Opoc		L			•		٠.	••	٠.				••				Q	1	P	3-P	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	1	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	Sta	atı	ısr	egi	ist	er
1/1						Ε	Α	М	F	z	#	Р	С	1	Υ	D	U	!	В	G	н	I	N	z	٧	С

Mot	orola	LSR	A	. IND. REL. I O F S T # P R X													1	CF	PU	ΙT	УI	o E	3 /	6	В0	9
B&F	ł	SRA	١.	Sch	iebe lı	nh	al	t١	/ 0	n	EF	RA						0-	1	7_	J	1	-	,	F	F
Kurz	z																		d	17	_			d	Õ	C
Δd	rossio	runas	um 1 Bit nach rechts ungsarten / Opcode Adreßvorwa												ah	ما	n				C	7	С	P	30	
L	63316	unga	arterr	Schiebe Inhalt von ERA um 1 Bit nach rechts 1/ Opcode Adreßvorwah 1. IND. REL I OFST # PRXY														C)	Р	C	30				
IMPL.	DIR.	EXT.	IMMED.	n / Opcode Adreßvorwahle								D	U	!	В	G	St	atι	ısı	eg	ist	er				
2/1						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U		В	G	Н	I	Ν	Z	٧	С
44						Γ	Γ	Γ	Γ				Γ								0	0	▼	•	0	•

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.

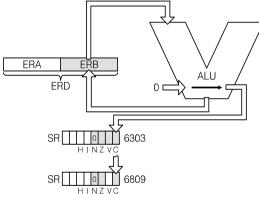


6303:

Mot	orola	LSF	(B	Fun	ktion												-	CI	PU	, ,	У	р	4	6	30	3	ı
B&F	₹	SRE	3	Sch	iebe lı	nh	al	t١	/ 0	n	EF	RB	;					0-	•	-	F	1	-	-	\vdash	ſ	l
Kur	z			um	1 Bit n	ıa	ch	r	ec	ht	ts								ď	7				d	Õ	C	l
Adı	ressie	rungs	arten	/ Opco	ode				Α	dı	reí	Зv	or	w	ah	ıle	n				0))		G1(G-F	_	_	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	SI	atı	JS	reg	ist	er	ı
1/1						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	N	Z	٧	С	ı
54						Γ	Γ	Γ	Γ	Γ	Π	Γ		Г	Г		Γ	Г		Г	b	0	▼	•	•	•	ı

Mot	orola	LSR	В	Fun	ktion												1	CI	٦,	ΙŢ	y	рE	3 /	6	80	9
B&F	₹	SRE	3	Sch	iebe lı	٦h	al	t١	/ 0	n	EF	RВ						0-	-	+	J		-	,	7	Ŧ
Kurz	z			um	1 Bit r	a	ch	r	ec	ht	s								Ċ	17				d	Ō	C
ЬA	ressie	runas	arten	/ Once	ode	Г			Α	dr	eí	ßν	or	w	ah	le	n				\circ)		P		
1		90		Орос					•		٠.		٠.	•••			••				C)	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
2/1						Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
54						Г	Γ	Г	П												0	0	▼	•	0	•

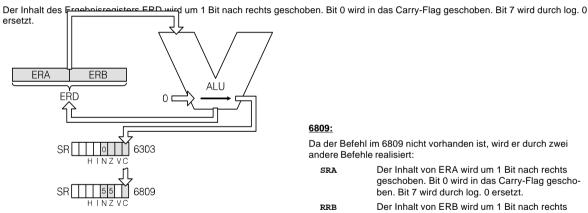
Der Inhalt des Ergebnisregisters ERB wird um 1 Bit nach rechts geschoben. Bit 0 wird in das Carry-Flag geschoben. Bit 7 wird durch log. 0 ersetzt.



6303:

Mot	orola	LSF	₹D	Fun	ktion												-	CI	PU	1	у	р	4 /	6	30	3
B&F	₹	SRI)	Sch	iebe lı	nh	al	t١	/ 0	n	EF	RD)	0-	F	1	1		Ţ	1	1		Ţ	-	7	┦
Kur	z			um	1 Bit n	ıa	ch	r	ec	ht	s				d15	;									d0	C
Αdı	ressie	runas	arten	/ Once	ode	Γ			Α	dı	eí	۱۷	or	w	ah	ıle	n				Ç	7		G1 (_
, , , ,	000.0	90		Орос		L					٠.		•	•••			••				Ç)	Ρ	G-F	C	╝
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Ρ	R	х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
1/1						Ε	Α	М	F	Z	#	Ρ	С	1	Υ	D	U	!	В	G	Н	ī	N	z	٧	С
04						Г				Г		П		П			П			Г	О	0	▼	•	•	•

Mot	orola	LSR	D	Fun	ktion												1	CF	2	ΙT	УI	E	3 /	68	30	9
B&F	ł	SRE		Sch	iebe lı	nh	al	t١	0	n I	EF	RD	, ,)—	F	1	1		Ţ	Ţ	*_		Ţ	7	7	₹
Kur	Z			um	1 Bit r	ıa	ch	re	ес	ht	s				d15									(0E	C
Δd	ressie	runas	arton	/ Once	nde	Γ			Δ	dr	ام:	ŝν	۸r	\A/:	ah	ما	n				V)	С	Pε	30	
Au	CSSIC	ungs	anten	Opce	Jue				_	uı	CI	3 V	UI.	vv	211	16					V)	Р	CE	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atu	ısr	eg	ist	er
4/2						Ε	Α	М	F	Z	#	Р	С	-	Υ	D	U	!	В	G	Ι	Ι	Ν	Z	٧	С
44 56						Γ															0	0	×	×	0	•



6303:

Das V-Flag erhält nach Ausführung des Befehles das Ergebnis von N ⊕ C zugewiesen (V = N ⊕ C). Das V-Flag zeigt an, ob das Bit 7 durch die Operation verändert wurde oder nicht (V = 1 => Bit 7 wurde verändert: V = 0 => 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 gescho-

ben. Bit 7 wird durch log. 0 ersetzt.

RRR

Der Inhalt von ERB wird um 1 Bit nach rechts. geschoben. Das Carry-Flag (=> Bit 0 von ERA) wird ins Bit 7 geschoben. Bit 0 wird in das Carry-

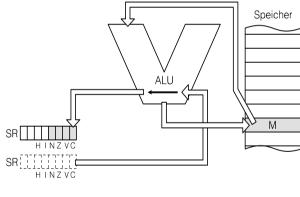
Flag geschoben.

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

Mot	orola	ROL	-	Fun	ktion												•	CF	2	JΤ	Γу	р	A .	6	30	3
B&F	₹	SLI		Roti	ere In	ha	alt	V	on	١	1								ħ.	[1,	ľ	F	1.	1	7
Kur	z	RL		um	1 Bit n	a	ch	li	nk	s									C	(17	_		1.	d	ď
Αdı	ressie	runas	arten	/ Once	ode	Г			Δ	dr	eí	۱۷	or	w	ah	le	n				Ç	7	_	G1(_
,		90	u	Opoc					•		٠.	••	٠.	•••			••				ļÇ)	Ρ	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	-	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	us	reg	ist	er
		6/3		6/2		Ε	Α	М	F	z	#	Р	С	1	Υ	D	U	!	В	G	Н	ī	Ν	Z	٧	С
		79		69		Г	0	0	0	П		0	0	0	П		П		Г	П	0	0	•	•	•	•

Mot	orola	ROL		Fun	ktion												1	CF	٦	ΙT	УI	o E	3 /	6	В0	9
B&F	₹	SLI		Roti	ere In	ha	alt	٧	on	N	1								Т	4	1	ľ	1	Ŧ.	[*	7
Kurz	z	RL		um	1 Bit r	a	ch	li	nŀ	s									C	(17		7		C	0
	ressie	runae	arton	/ Once	ndo.	Г			^	dr	۰,	2.,	٥r		ah	ı	n				C	7	С	P	30	
^u	CSSIC	ungs	arterr	Opce	Jue				^	uı	C.	3 V	U	vv.	all	16	••				0	7	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
	6/2	7/3		6+/2+		Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
	09	79		69			0	0	0			0	0	0	•	•	•	•		•	0	0	•	•	•	•

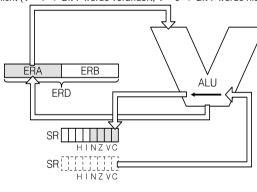
Der Inhalt der Speicherstelle M wird um 1 Bit nach links geschoben. Das Carry-Flag wird in Bit 0 geschoben. Bit 7 wird ins Carry-Flag geschoben.



Mot	orola	ROI	_A	Fun	ktion												(CF	'n	Т	уı	р /	4 /	6	30	3	Γ	Mot	orola	ROL	Α.	Ē
B&F	₹	RLA	4	Roti	ere In	ha	alt	V	on	Ε	R	A						ſ	٦.	4	₹.	ľ	Ţ	+.	[*	7.	1	B&F	ł	RLA	,	R
Kur	z			um	1 Bit r	na	ch	ı li	nk	s								·	С	d	17	_	-1	1.	_	70 0k		Kurz	z			u
Adı	ressie	rungs	arten /	Орсс	de				A	dr	eß	Sve	or	Wa	ah	lei	n				OO))		31(G-F		_		Adı	ressie	rungs	arten	Op
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	ter	1	IMPL.	DIR.	EXT.	IMMED.	INE
1/1						Е	Α	М	F	z	#	Р	С	1	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С	1	2/1				
49						Γ															0	0	•	•	•	•	1	49				

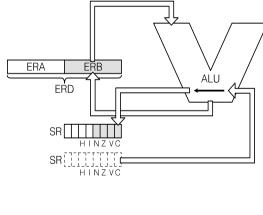
Mot	orola	ROL	-A	Fun	ktion												1	CF	יַיַ	J T	уı	o E	3 /	6	В0	9
B&F	₹	RLA	١	Roti	ere In	ha	alt	٧	on	Е	R	Α							ή	[1	F	Ţ	Ŧ.	1	7
Kur	z			um	1 Bit r	ıa	ch	li	nŀ	s									c	(17	_		_		ĺ
Δ.	ressie	runas	arton	/ Once	nde	Γ			Δ	dr	ام	٠.	۸r	١٨/٠	ah	ما	n				V	7	С	P	30	
~	163316	unga	arterr	Орсс	Jue				_	uı	٠.	3 V	٠.	**	anı						U)	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısı	eg	ist	er
2/1						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С
49						Г	Г	Г	П		Г	П	Г						П		0	0	•	•	•	•

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.



Mot	orola	ROL	В	Fun	ktion									(CP,	U 1	Гур	Α	/ 6	303	3	Mot	orola	ROL	В	Fun	ktion										CI	P <u>U</u>	Туј	В	/ 6	809
B&F	₹	RLE		Rot	iere In	ha	alt	vo	n E	RB					7	$\overline{\mathbb{H}}$	+_	+	I+	I+	ļ.	B&F	₹	RLE	3	Rot	iere In	ha	lt۱	or/	ı E	RE	3					ħ.	1	I+	I+	Ŧ
Kur	z			um	1 Bit r	na	ch	lin	ks						C	; ;	17		1	d(3	Kur	z			um	1 Bit r	ac	ch I	linl	ks							С	d7			dſ
Adı	ressie	rungs	arten /	Орсо	ode	Γ		_	۱dr	eß	voi	wa	hle	n			0	_	_	000]	Ad	ressie	rungs	arten	/ Opc	ode	Γ		Α	۱dr	eß	vo	rw	ahl	en			2	-	CP	
	DIR.	FXT	IMMED.	IND	Lasi	ł.	اما	- L	. [- [I.	Ja	I., I.	ر ا ہ	11	. I.	Ja	_	_	PG-	giste	_	IMPL.	Laua	Leve	IMMED.	LIND	REL.	 	م آ ہ	- I.a	T-1	I., I.		T _v	I., I	. I.	Л.	I. I.	- 64	_	PC	80 giste
IMPL.	DIK.	EXI.	IMMED.	IND.	REL.	-	-	_	$\overline{}$	-	-	-	—	-	_	-	-	_		-	_		DIR.	EXT.	IMMED.	IND.		$\boldsymbol{\vdash}$	_	_	_	—	_	-	\vdash	-	-	-	-	_	`	-
1/1						E	А	M F	Z	# F	, lc	1	r ID	ΙU	! E	3 G	Н	-	_	٧	_	2/1						E	ΑN	ΛF	Z	# E	PC	11	Υ	DΙ	<u>' !</u>	В		_	_	V
59						ı	Н					Ш		Ш			o l	⊃ •	• •	•	•	59						П				Ш						Ш	0	0	•	• • •

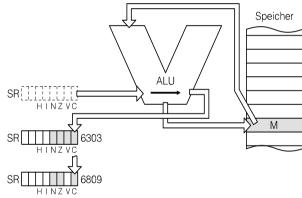
Der Inhalt des Ergebnisregisters ERB wird um 1 Bit nach links geschoben. Das Carry-Flag wird in Bit 0 geschoben. Bit 7 wird ins Carry-Flag geschoben.



Mot	orola	ROI	₹	Fun	ktion												•	CF	2	JΤ	УІ	0 /	4 /	6:	30	3
B&F	₹	SRE		Roti	ere In	ha	alt	V	on	١	1								ħ	- [Ŧ	F	1	1	7	\downarrow
Kur	z	RR		um	1 Bit n	a	ch	re	ec	ht	s								C	L	17	_	_		d	ď
Δdı	ressie	runas	arten	/ Onco	nde	Г			Δ	dr	eí	۱۷	٥r	w	ah	le	n				U)		G10		
,		90	u	Opoc					•		٠.	••	٠.	•••			••				Ç		Ρ	G-F	c	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	-	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	JSI	reg	ist	er
		6/3		6/2		Ε	Α	М	F	z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
		76		66		Г	0	0	0	П		0	0	0	П		П		Г	П	0	0	•	•	•	•

Mot	orola	ROF	₹	Fun	ktion												- 1	CF	2	JΤ	УI	рE	3 /	68	80	9
B&F	ł	SRE	:	Roti	iere In	ha	alt	V	on	N	1								ή	H	Ŧ	F	Ī	Ŧ	F	7
Kurz	z	RR		um	1 Bit n	ia	ch	re	ec	ht	s								C	(d7	_		_		Į0
Δd	roccio	runas	arten /	/ Once	ode	Г		Τ	Δ	dr	ام	21	or	\A/-	ah	ما	n				C	<u>1</u>	С	PE	30	
Au	CSSIC	ungs	ai teii /	Орсс	,ue	L						-									0			33		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.		0	F	s	т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	reg	ist	er
	6/2	7/3		6+/2+		Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
	06	76		66		Г	0	0	0	П	П	0	0	0	•	•	•	•		•	0	0	•	•	0	•

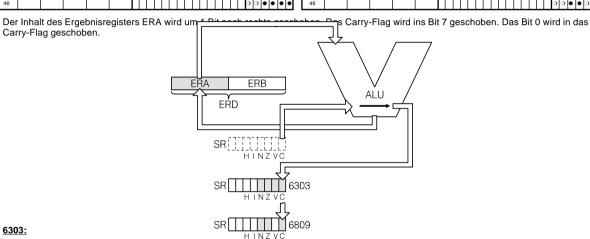
Der Inhalt der Speicherstelle M wird um 1 Bit nach rechts geschoben. Das Carry-Flag wird ins Bit 7 geschoben. Das Bit 0 wird in das Carry-Flag geschoben.



6303:

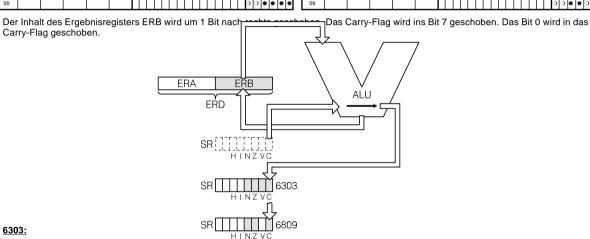
Mot	orola	ROI	RA	Fun	ktion												•	CF	٧	Т	Уľ) <i>A</i>	۸/	63	30:	3
B&F	₹	RRA	4	Roti	ere In	ha	alt	V	on	Е	R	Α							ή.	→F	Ŧ	H	I	1	Ŧ	\mathbb{I}
Kurz	z			um	1 Bit n	ıa	ch	r	ec	ht	s								C	ď	7			ш	d	ď
Δdı	ressie	runas	arten	/ Once	ode	Γ			Α	dr	eſ	۱۷	or	w	ah	le	n			٦	0	1		310		
,		90		Opoc		L			•		٠.	•	•								Q	1	P	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	Sta	atı	ısr	eg	ist	er
1/1						Ε	Α	М	F	z	#	Ρ	С	Ι	Υ	D	U	!	В	G	н	Τ	Ν	Z	٧	С
						•	_	-	$\overline{}$				\neg	\neg			\neg		\neg	_	$\overline{}$	$\overline{}$	_	•	_	ᆿ

Mote	orola	ROF	RA	Fun	ktion													CI	2	JΤ	y	рΙ	B /	6	80	9				
B&R	?	RRA	١	Roti	iere In	ha	alt	v	or	E	R	Α							ή	Ļ	+	F	1	+	H	7				
Kurz	z			um	1 Bit r	ıa	ch	r	ec	ht	s								C	(17	_			- (ر 10				
Adressierungsarten / Opcode						Γ			Α	dr	el	٩v	or	w	ah	le	n				$^{\circ}$)		P						
/	Adressierungsarte					L			•		٠.	•	٠.	•••			•••				C)	Ρ	C	80					
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	usregiste							
2/1						Ε	Α	М	F	Z	#	Ρ	С	T	Υ	D	U	!	В	G	Н	ī	N	z	٧	С				
46						Г	Г	Г	Г		Г	Г	Г	Г			Г	П	П		0	0	•	•	0	•				



Mot	orola	ROF	RB	Fun	ktion												•	CF	טַי	T	yŗ	A	۸/	63	30:	3							
B&F	₹	RRE	3	Roti	ere In	ha	alt	V	on	Е	R	В							1	F	<u> </u>	H	Ţ]	7	\mathbb{I}							
Kur	z			um	1 Bit n	a	ch	r	ec	ht	s								C	ď	7			_	d	j							
Δdı	Adressierungsarten / Opcode								Δ	dr	e í	۱۷	٥r	wa	ah	le	n			I	0	_	PG			┒							
Au.	000.0	ungo	Орос	uc				_	۵.	٠.	•	٠.	•••	4		••			1	0	1	PC	3-P	C									
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	Sta	atı	tusregister										
1/1						Ε	Α	М	F	z	#	Ρ	С	I	Υ	D	U	!	В	G	Н	I	N	z	٧	С							
						г	т	$\overline{}$				П		П			П	\neg	\neg	_	<u> </u>	$\overline{}$		_	_	⇁							

Mote	orola	ROF	₹B	Fun	ktion													CI	<u> </u>	ΙŢ	у.	рE	3 /	6	80	9
B&R	2	RRE	3	Roti	ere In	ha	alt	V	on	E	R	В							ή	_[+	F	Ī	Ŧ	F	7
Kurz	z			um	1 Bit r	ıa	ch	r	ec	ht	s								C	- (d7	_			d	0
Δdı	Adressierungsarten / Opcode								Δ	dr	el	١٧	or	w	ah	le	n				0)	_	P		
~~.	000.0	ungo	Орос	Juc	L			_	۵.	٠.	•	٠.	•••	u.,		••				C)	Ρ	C	30		
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	υ	!	В	G	St	atı	ısr	eg	ist	er
2/1						Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	Ν	Z	٧	С
56						Г	П	Г					Г	П		Г		Г	П		0	0	•	•	0	•



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		Betr	iebsart	
Wotorola	Dak	PG1000	PG-PC	CP 80	PC 80
BCC 1)	JC0 1)	0	0	0	0
BCS ¹⁾	SP<1)	0	0	0	0
BPL ¹⁾	J+ 1)	0	0	0	
BMI ¹⁾	J- 1)	0	0	0	0
BNE ¹⁾	SN0 1)	0	0	0	0
BEQ 1)	SP01)	0	0	0	0
BHI ¹⁾	SP>1)	0	0	0	0
BLS ¹⁾	J<= 1)	0	0	0	0
SK0	SK0		0		0
SK1	SK1		0		0
JSR	SPU	0	0	0	0
RTS	RET	0	0	0	
JMP	SPI	0	0	0	0
NOP	NOP	0	0	0	
END	END	0	О	0	О

Wurze 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 er	rfüllt ist
Kurz	xxx		

۸.	roccio	runae	arton	Once	,do				^	۸,	٠,	2.,	or		ak	ı۱۸	n				O	1	P	310	000	1			
Α α	Adressierungsarten / Opcode								^	u	CI	3 V	U.	vv.	aı		•••				O	•	Р	G-F	Š				
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	sregister					
					3/2	Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С			
					xxx																0	0	0	0	0	0			

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

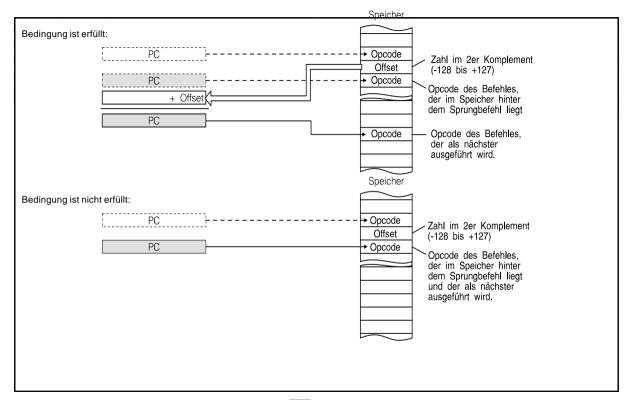
	Mnemonics		Bedingung	Opcode
Motorola	B&R	Kurz	Bealinguing	Opcode
BCC	JC0		C = 0	24
BCS	SP<	J<	C = 1	25
BEQ			Z = 1	27
BHI	SP>	J>	$C \vee Z = 0$	22
BLS	J<=		C ∨ Z = 1	23
BMI	J-		N = 1	2B
BNE	SN0	J1	Z = 0	26
BPL	J+		N = 0	2A

Mot	orola	xxx		Fun	ktion												1	CF	٧,	JΤ	УI	o E	3 /	6	80	9							
B&F		xxx		Spri	nge, v	ve	nı	n I	Зе	di	nç	gu	ηç	ı x	X	(6	erf	ül	lt	is	ŧ												
Kurz xxx																																	
Adressierungsarten / Opcode									Δ	dr	ام	21/	۸r	۱۸/	ah	ما	n				O	1	С	P	30								
, Au	000.0.	ui toii i	Орос	Juc	L			_	۵.	٠.	•	٠.	•••	u.,	٠.	••				a		Ρ	С	30									
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	atusregiste										
					3/2	Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	н	I	N	Z	٧	С							
					xxx	Г		Г	Г	Г	Г	Γ	Г		Г			П		П	0	0	0	0	0	0							

Kurze relative Sprünge bestehen aus zwei Bytes:

- 1) Das erste Byte ist der Opcode des Befehles.
- 2) 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



Motorola	Funktion	CPU Typ A / 6303
B&R		
Kurz		

٨٨	rossio	runae	arten	Once	ndo.	Γ			^	4	٠,	2.,	or		- h		n					Τ	P	G10	000	
Au	CSSIC	ungs	arterr	Opce	ue				^	u	CI	3 V	U	vv.	aı		••					Τ	Р	G-F	Š	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
						Γ	Π																			

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

	Mnemonics		Dadinauma	Oncode
Motorola	B&R	Kurz	Bedingung	Opcode
BCCL	JC0L		C = 0	10 24
BCSL	SP <l< td=""><td></td><td>C = 1</td><td>10 25</td></l<>		C = 1	10 25
BEQL	SP0L		Z = 1	10 27
BHIL	SP>L		$C \vee Z = 0$	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

Mot	orola	xxx		Fun	ktion												1	CI	٧,	JΤ	уı	o E	3 /	6	80	9
B&F	1	xxx		Spri	nge, v	ve	nı	n E	Зе	di	nç	gu	ηç	Į X	(X)	(6	erf	ül	lt	is	t					
Kur	z																									
۸.	neeln	runae	arten	Once	ndo.	Γ			^	dr	۰,	21,	۸r		a h	ı۸	n					Τ	С	P	80	
Au	CSSIC	ungs	anten	Орсс	Jue				^	uı	CI	3 V	UI	w	all	16					О	1	Р	C	80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısı	eg	jist	er
					5(6)/4	Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U		В	G	Η	I	N	Z	٧	С
					xxx	Γ	Г	Г					Г								0	0	0	0	0	0

Lange relative Sprünge bestehen aus vier Bytes:

- 1) Die Bytes 1 und 2 sind der Opcode des Befehles.
- 2) 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 sieht auf internationaler

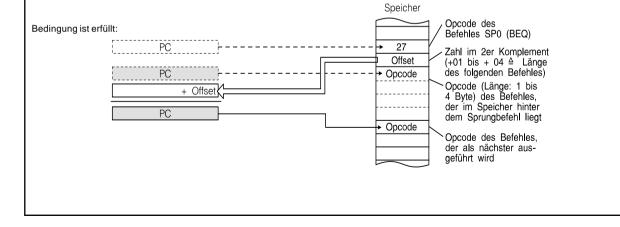
 Bedingung ist nicht erfüllt, Sprung wird nicht ausgeführt.

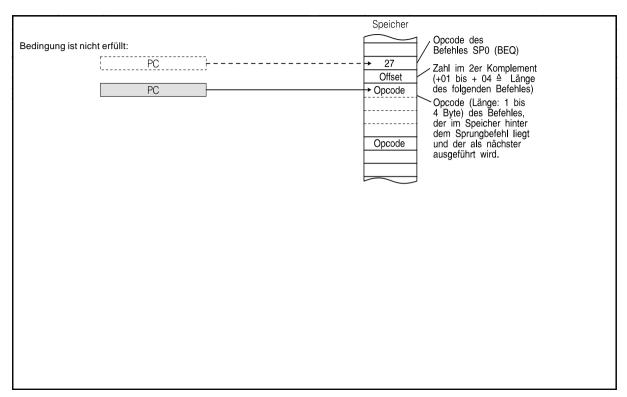
Speicher Bedingung ist erfüllt: 10 Opcode des Sprungbefehles XX Offset (MSB) Zahl im 2er Komplement (-32768 bis +32767) Offset (LSB) ➤ Opcode Opcode des Befehles, + Offset der im Speicher hinter dem Sprungbefehl liegt PC → Opcode Opcode des Befehles, der als nächster ausgeführt wird. Speicher Bedingung ist nicht erfüllt: 10 Opcode des Sprungbefehles ХX Offset (MSB) Zahl im 2er Komplement (-32768 bis +32767) Offset (LSB) PC → Opcode Opcode des Befehles. der im Speicher hinter dem Sprungbefehl liegt und der als nächster ausgeführt wird.

Mot	orola	SKO)	Fun	ktion										С	PU	T	yp	A /	63	303	Γ	Mote	orola	sĸ	0	Fun	ktion										С	PU	J T	ур	В/	68	09
B&F	₹	SKO)	Übe	rsprir	nge	n	äcł	ıst	en	Ве	fel	hI,	we	en	n Z	<u> </u>	1				1	B&R		sĸ	0	Übe	rsprir	nge	n	äc	hs	ter	ı B	efe	ehl	l, w	/en	ın Z	Z =	1			
Kur	z																					П	Kurz	<u>.</u>			1																	
۸ ط	ressie	runae	arton	Once	ndo.	Т		_	dr	~ R ·	vor	14/2	h	۸n			Т		P	G10	000	1	Λ.α.	occio	runa	sarten	/ Onc	ada	Т			٧4	-06	11/6			nler	_		Т		С	P 80	<u>, </u>
Aui	CSSIC	ungs	arterr	Орсс	Jue				uii	CIS	VOI	vv c		CII				\circ	Ρ	G-P	Č]	Aui	C 331C	ung	Saiteii	/ Opci	Jue				٦u		340	/I V\	aı	IIEI				<u> </u>	P	C 80)
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	O F	s	Т	# F	R	х	ΥĮ	U	!	В	G	Stat	usi	eg	ister	7	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	П	0	FS	зТ	#	P	₹ X	ίY	D,	U !	В	G	Stat	usr	egis	ste
					3/2	Е	ΑN	1 F	Z	# F	С	1	ΥĮ) U	!	В	G	ΗΙ	N	Z	v c	1						3/2	Е	Α	M F	Z	#	P (0 1	Y	D	U!	В	G	ΗΙ	Ν	Z۱	/ (
					27	Π	T			Ī					Γ)	0	0	ာ]						27	Π						T	I					၁	0	0)

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.





Mot	orola	SK	ı	Fun	ktion													C	P	וי	УΙ	0 /	4 /	6	30	3	ı	Mot	oro
B&F	₹	SK.	1	Übe	rsprin	g	e i	nä	cl	hs	te	n I	Вє	fe	hl	۱, ۷	ve	n	n Z	Z :	= 0)					1	B&F	₹
Kur	z			1																							l	Kur	z
Adı	ressie	rungs	arten	/ Opco	ode				A	١d١	rei	ßv	OI	w	ah	ıle	n				C)	P(_)00 2C		1	Ad	res
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	υ	!	В	G	St	atı	ısı	eg	ist	er	1	IMPL.	DI
					3/2	Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С	1		
					26	Γ	Γ	Γ	Γ	Γ	Γ	Γ	Γ	Γ	Γ	Γ		Γ	Γ	Γ	0	0	0	0	0	0	1		

Franklikan

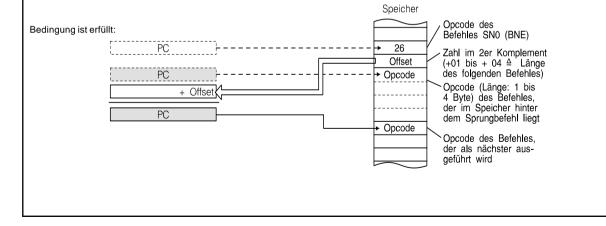
Madanala OKA

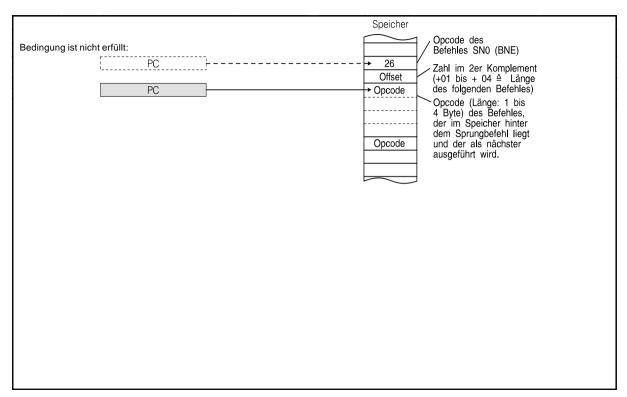
Mot	orola	SK1		Fun	ktion												1	CF	٧,	ΙT	УĮ) E	3 /	6	В0	9
B&F	t	SK1		Übe	rsprin	ıg	e i	۱ä	ch	st	eı	n E	Зе	fe	hl	, v	ve	nr	1 2	Z =	0					
Kur	z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	eſ	١٧	٥r	w	ah	le	n					Ι		P		
	00010	ungo	ui toii ,	Орос	,uc	L			_	۵.	٠.	•	٠.	•••	۵						C)	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
					3/2	Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	1	N	Z	٧	С
					26	Г	Г	Г	П			Г	П								0	0	0	0	0	0

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

ODU T. ... A / 6000

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.





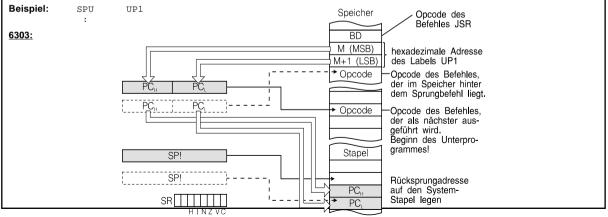
Mot	orola	JSR	l	Fun	ktion												1	CF	2	J	УI	o /	4 /	6	30	3
B&F	₹	SPL	J	Unb	eding	te	r S	Sp	ru	ın	g i	n	ei	n I	Jr	nte	erp	rc	og	ra	m	m				1
Kurz	z																									
۸۵۰	roccio	runae	arton	/ Onco	,do	Г			^	4	٠,٠١	2 . ,	^-	١	a h	ı۱۸	n				0	T	P	310	000	П
Aui	CSSIC	ungs	arterr	Орсс	ue				~	u	CI	3 V	U.	vv.	211	116	••				О	7	Ρ	G-F	SC	П
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ISI	eg	ist	er
Kurz Adressierungsarten / Opcode Adreßvorwahlen O PG1000 O PG-PC											С															
	9D	BD		AD		Γ	Γ	Γ	Γ		Γ										0	0	0	0	0	\circ

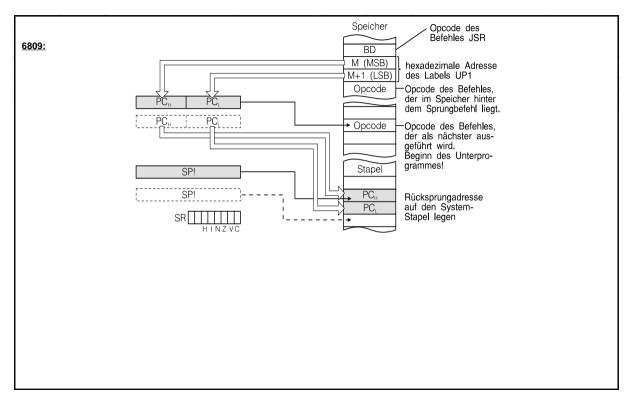
Mot	orola	JSR		Fun	ktion												1	CF	2	ΙT	УΙ	o E	3 /	6	В0	9
B&F	₹	SPU		Unb	eding	te	r S	Sp	ru	ng	j i	n	eiı	n I	Jn	te	erp	rc	og	ra	m	m				
Kur	z																									
Δd	ressie	runas	arton	/ Once	nde	Γ			Δ	dr	ام	21/	۸r	\A/:	ah	ما	n			П	a	1	С	Pδ	30	
	163316	ungs	arterr	Opce	Jue	L			_	uı	٠.	3 V	٠.	***	411						a	1	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
	7/2	8/3		7+/2+		Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
	9D	BD		AD		т	_	_												_	_					

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:



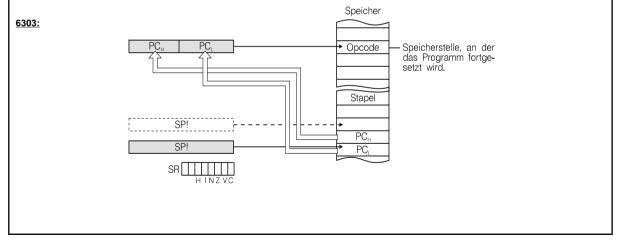


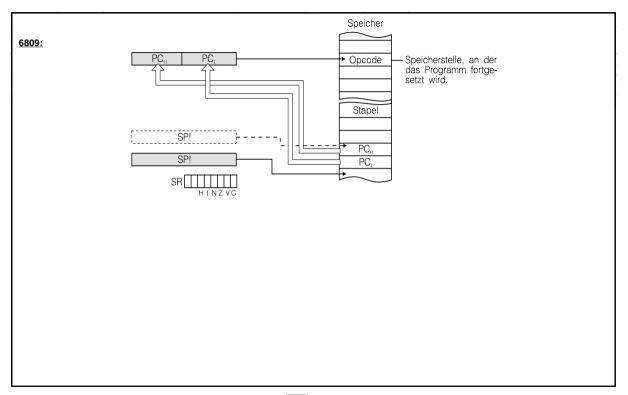
Mot	orola	RTS	;	Fun	ktion												T	CI	?(JΤ	y	o /	۹ /	6	3	03	Ī		Mote	orola	RTS
B&F	₹	RET	•	Rüc	kspru	ın	g١	/ 0l	m	U	nt	erį	pr	og	ıra	am	ım	1									1		B&R	ł	RET
Kur	z																												Kurz	z	
Adı	ressie	rungs	arten	/ Opco	de				Α	dr	eí	3v	or	Wa	ah	le	n				OO	,	P(_	1		Adı	ressie	rungs
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ISI	reç	gis	te	7		IMPL.	DIR.	EXT.
5/1						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	' (2	1	5/1		
39						Г	Π	Γ	Γ			П									0	0	0	0	C))		39		

Mot	orola	RTS	;	Fun	ktion												1	CF	2 U	T	Уļ	o E	3 /	68	30	9
B&F	1	RET		Rüc	kspru	ng	g١	0	m	Uı	nt	er	pr	οç	ıra	ım	ım									
Kur	z																									
Δd	ressie	runae	arton	/ Once	nde	Γ			Δ	dr	ام:	ßν	۸r	\A/:	a h	ما	n			П	O	1	CI	8	0	
Au	63316	ungs	arterr	Opce	Jue	L			_	uı	٠.	3 4	٠.	***	411						O)	P	8	0	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atu	ısre	egi	st	er
5/1						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Н	I	N.	z	٧	С
39						Г	Г	Г	П			Г	П							П	0	0	0)	0	0

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:

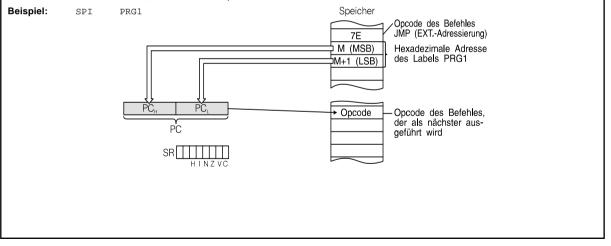




Moto	orola	JMF	•	Fun	ktion										CP	U	Ту	p A	A / 6	303	П	Mot	orola	JM	Р	Fun	ktion										С	PU	Ту	рE	3 / 6	6809
B&R		SPI		Unb	eding	jte	r S	pr	un	g											11	B&F	₹	SPI		Uni	eding	jte	r S	pri	ung)										
Kurz	z																				П	Kur	z																			
Adr	essie	rungs	arten /	Орсс	ode	Γ		1	٩d	eß	vo	rwa	hl	en			0	_	PG1		7	Ad	ressie	rung	sarten	/ Opc	ode	Γ		,	٩dr	eß	vo	rw	ah	lei	n			-	CP PC	80
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	т	#	R	х	ΥC	U	!	В	St	atı	usreç	giste	r	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F S	т	# 6	P F	R X	Υ	D	U !	В			sre	giste
		3/3		3/2		Ε	А	М	z	#	c	T	ΥC	U	!	В	ЭН	I	N Z	V C	7		3/2	4/3		3+/2+		Ε	А	M F	Z	# 8	P	0 1	Υ	D	U!	В	GН	1	N Z	2 V C
		7E		6E		Г	П	Т	Т	П	Τ	П		П	П		0	0	၁၁	o c	7		0E	7E		6E		Т	П	Т	П	П	T		П	П			0	0	0))

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.



Motorola	NOP	Funktion		CPU Typ	A / 6303
B&R	NOP	Keine Fun	ktion		
Kurz					
				o	PG1000

٨٨	roccio	runae	arten	Once	do				^	۸,	٠,	2.,	PRXYDU!					С)	P	310)00	,			
Au	CSSIC	ungs	arterr	Орсс	ue				~	u	CI	3 V	U	vv.	aı		••				O	7	P	G-F	SC	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
1/1						Е	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
01						Γ	Γ								Г						0	0	0	0	0	0

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

Mot	orola	NOF	•	Fun	ktion												1	CF	P	ΙT	УI	o E	3 /	6	В0	9
B&R	1	NOF	•	Keir	ne Fun	kt	tio	n																		
Kurz	z																									
۸۵۰	roccio	runae	arten	/ Once	,do	Г			^	dr	٠~١	2 . ,		١	ah	ı	n				a	ī	С	P	30	
Aui	COOLC	ungs	arterr	Орсс	ue				_	uı	CI	3 V	Ui	w	all	16					0		Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
2/1						Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	н	I	Ν	Z	٧	С
12						Γ							Γ								0	0	0	0	0	ာ

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

Mot	Adressierungsartei PL. DIR. EXT. IMME			Fun	ktion									-	СР	U T	Гур	ь А	/ 6	303	П	Mot	orola	Т	END)	Fur	ktion										С	Pι	JΤ	уp	В	68	309
B&F	2	END	•	Prog	gramn	ner	de	!													П	B&F		I	END)	Pro	gram	me	enc	le!													
Kur	Kurz			Beg	inne i	n F	ro	gra	ımı	mz	eile	0									П	Kurz	:				Beg	ginne	in	Pr	og	ran	nm	ze	ile	0								
Δ.	Kurz Adressierungsarter			Once	ndo.	Г		Δ	dre	Rv.	orv	val	ale	n			0	1	PG1	000]	Δd	000	oru	ınae	arten	/ Onc	ode	Т			Δdı	r a fi	l v) rv	vah	اماد	_		П	0	С	P 8	0
	163316	ungs	arterr,	Opce	ue			^	uie	13 V	0	vai					0		PG-	PC	Ы	ď	633	616	iliga	arterr	Орс	oue				-u	CI.	,,,	J. V	vai					0	Р	C8	0
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı () F	s	T #	P	R 2	ΚY	D	U	! !	з С	Sta	atu	sreç	gister	1	IMPL.	DIR.	E	EXT.	IMMED.	IND.	REL.	ŀ	0	F	зΤ	#	Р	R >	(Y	D	U !	В	G	Stat	usi	regi	ster
	MPL. DIR. EXT. IMMED.				E /	A M	F	Z #	P	С	ΙY	D	U	! !	в С				v c									E	Α	М	F Z	#	Р	С	ΙY	D	U!	В	G	ΗΙ	N	Z	v c	
		7E C0 00															0	o '	▼ ▲	. 🔻 🔻] [11 3F							I												ા	0	0	ာ

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.

<u>**6809:**</u> 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.

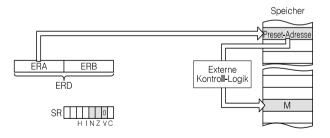
3.11. SONSTIGES

Motorola	B&R		Betr	iebsart	
Wiotorola	Dak	PG1000	PG-PC	CP 80	PC 80
PRS	PRS	0	0	0	0
RST	RST	0	0	0	0
CLR	CLR	0	0	0	0
CLRA	CLA		0		0
CLRB	CLB		0		0
SET	SET	0	0	0	0
CLC	CLC	0	0	0	0
SEC	SEC	0	0	0	0
CLI	CLI	0	0	0	0
SEI	SEI	0	0	0	0
COM	K	0	0	0	0
COMA	COA		0		0
COMB	COB		0		0
DAA	DK	0	0	0	О

Mot	orola	PRS	3	Fun	ktion													CI	٧,	JΤ	УI	р.	Α/	6	30	3
B&F	₹	PRS	3	Wer	n d0	vc	n	E	R.A	۱ =	: 1	: '	1 ε	⇒	(N	1)										
Kur	z	Р		Wer	n d0	vo	n	EI	R.A	۱ =	: 0): ((M) I	ole	eik	t	ur	ıv	er	än	d	ert	Ċ		
٨٨	roccio	runaa	arten	/ Onno	,do	Γ		CPU Typ A / 6303 ERA = 1: 1 □ (M) ERA = 0: (M) bleibt unverändert O PG1000 O PGPC O PGPC O PGPC PGPC																		
Au	essie	rungs	arterr	Opco	ue				^	RA = 1: 1 D (M) RA = 0: (M) bleibt unverändert Adreßvorwahlen O PG1000																
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	t unverändert O PG1000 O PG-PC U I B G Statusregiste				er				
		4/3				Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	ī	N	Z	٧	С
		B7				Γ	0	0	0			Г									0	0	•	•	▼	0

Mote	orola	PRS	3	Fun	ktion												(CF	٧U	ΙT	уı	o E	3 /	6	во	9
B&R	ł	PRS		Wen	ın d0 v	/0	n	EI	RA	= ۱	: 1	: 1	1 2	⇒	(N	I)										7
Kurz	z	Р		ın d0 v	0	n	ΕI	RA	= ۱	: 0	: (M) I	ole	ik	t ı	un	V	er	än	de	ert				
Δdı	ressie	runas	arten	nde				Δ	dr	eſ	١٧	or	w	ah	le	n				С	1	С	P	30		
Au.	00010	ungo	ui toii i	Орос	Juc				_	۵.	٠.	•	٠.	•••	u.,						С)	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	-	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
		5/3				Е	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
		В7					0	0	0											П	0	0	•	•	•	0

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.

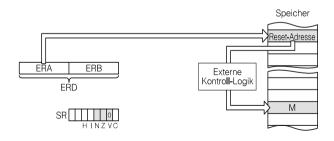


PRESET für erweiterte Ausgänge (G bis N) ist aus Speichergründen nicht möglich!

Mot	orola	RST		Fun	ktion												(CI	٦,	ן ו	УI	0 /	A .	6	30	3
B&F	₹	RST	_	Wer	ın d0 v	/0	n	ΕI	R.A	١ =	= 1	: (0 ε	⇒	(N	1)										٦
Kur	z	R		Wer	n d0 v	0	n	ΕI	R.A	١ =	= 0	: ((M) I	οle	eik	t	ur	ıv	er	än	de	er	t		-
Δdı	rossio	runae	arten /	Once	nda	Г			Δ	di	eí	١,	۸r	۱۸/	ah	ما	n				0	1	Р	G1(000	╝
Aui	COOL	ungs	ai teii /	Opce	ue				^	u	٠.	. v	٠.	***	u						a	•	Р	G-F	С	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	us	reg	ist	er
		4/3				Ε	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Н	I	N	Z	٧	С
		B7				Г	0	0	0	Г	Γ		Г	П	П				П	Г	0	0	•	•	▼	0

Mot	orola	RST		Fun	ktion												(CF	2 U	T	УI	o E	3 /	68	30	9
B&F	₹	RST		Wen	ın d0 v	/0	n	ΕI	R/	= ۱	: 1	: () E	⇒	(M)										
Kur	z	R		Wen	n d0 v	0	n	ΕI	RA	= ۱	: 0	: (M) k	ole	ik	t ı	un	ıve	era	än	de	ert			
Δd	ressie	runas	arten	/ Once	nde	Г			Δ	dr	eſ	١٧	٥r	w	ah	le	n				С	<u> </u>	С	Р8	30	
7.0	. 000.0	ungo	ui toii i	Орос	Juc				_	۵.	٠.	•	٠.	•••	4	٠.					С)	P	8	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atι	ısr	egi	ist	er
		5/3				Ε	Α	М	F	Z	#	Ρ	С	1	Υ	D	U	!	В	G	Η	I	N	z	٧	С
		B7				Г	0	0	0		П		Г								0	0	•	•	•	0

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.



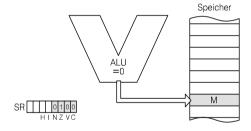
RESET für erweiterte Ausgänge (G bis N) ist aus Speichergründen nicht möglich!

Motorola	CLR	Funktion		CPU Typ	A / 6303
B&R	CLR	0 ⇔ (M)			
Kurz	С				
Adrossio	rungearton	Opcodo	Adroßvorwahlen	0	PG1000

٨٨	roccio	runae	arten /	Once	do	ı			۸	۸,	٠.,١	2.,	^-		ah	ıle	n				IC)	PC	G1()00)
Au	CSSIC	ungs	ai teii /	Opce	ue				~	uı	CI	3 V	U.	w	aıı		•••				C	,	P	G-F	S	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	О	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	Si	atı	ısı	reg	ist	er
		5/3		5/2		Е	Α	М	F	Z	#	Р	С	ı	Υ	D	U	!	В	G	н	Ι	Ν	Z	٧	С
		7F		6F			0	0	0			0	0	0							0	0	•	•	•	•

				_																						
Mot	orola	CLR	1	Fun	ktion												1	CI	٦,	JΤ	УI	рE	3 /	6	80	9
B&F	ł	CLR	ł	0 ⇒	(M)																					
Kurz	z	С																								
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	e í	ßν	٥r	w	ah	le	n				O	<u> </u>	С	P	30	
Au	000.0	ungo	ui toii	Орос	Juc				_	۵.	٠.	•	٠.	•••			••				С)	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	ısr	eg	ist	er
	6/2	7/3		6+/2+		Е	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
	0F	7F		6F		Г	0	0	0			0	0	0	•	•	•	•	П	•	0	0	▼	•	▼	▼

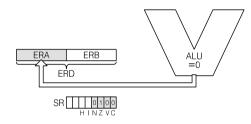
Die Speicherstelle M wird gelöscht, d.h. sie erhält den Wert log. 0.



Mot	orola	CLF	₹A	Fun	ktion												1	CP	U	T	У	o /	A /	6	30	3
B&F	₹	CLA		0 ⇒	ERA																					
Kur	z																									
Adı	ressie	runas	arten /	Opco	de	Г			Α	dr	eſ	ŝν	or	wa	ah	le	n				0	4		31 (_
																								C .	S	
																					О	'Щ		5	_	_
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	-	0	F	s	т	#	Р	R	х	Υ	D	U	!	В	G	١			eg	-	

Mot	orola	CLR	RA	Fun	ktion												1	CF	2	JΤ	y	p E	3 /	6	80	9
B&F	ł	CLA	١	0 ⇒	ERA																					
Kur	z																									
Δd	rossio	runas	arten	/ Once	nde	Γ			Δ	di	۵,	ßν	۸r	۱۸/	ah	ما	n				\circ	7	С	P	30	
ď	63316	unga	arterr	Орсс	Jue					u	٠.	3 4	٠.	***	an						\cup)	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
2/1						Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	N	z	٧	С
4F						Γ			Γ		Γ										0	0	▼	•	▼	▼

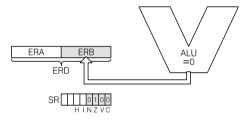
Das Ergebnisregister ERA wird gelöscht, d.h. es erhält den Wert log. 0.



Mot	orola	CLF	RB	Fun	ktion												1	CI	2	J T	y	o /	4 /	6	30	3
B&F	₹	CLE	3	0 ⇒	ERB																					
Kurz	z																									
۸ ۵۰	roccio	runae	arten	/ Once	,do	Г			^	4,	٠,٠١	21/	or	١	ah	ı۱۸	n				С	ī	P	G1 (000)
Aui	CSSIC	ungs	arterr	Орсс	ue				^	uı	CI	3 V	U	vv.	aıı		•••				С	7	Р	G-F	S	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	JSI	eg	ist	er
1/1						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	Ι	N	z	٧	С
5F						Г	Г	Г	П		Г	Г		П		Г	П		П	П	0	0	▼	•	▼	▼

Mot	orola	CLR	₹B	Fun	ktion												1	CI	٧,	ΙŢ	УI	рE	3 /	6	В0	9
B&F	ł	CLE	3	0 ⇒	ERB																					
Kurz	Z																									
Δd	ressie	runas	arten	/ Once	nde	Γ			Δ	dr	اع:	ßν	٥r	w	ah	le	n				C	<u> </u>	_	P		_
, Au	000.0	ungo	u. to	Орос	Juc	L			_	٠	٠.		٠.	•••	u.,		••				С)	Ρ	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	ısr	eg	ist	ei
2/1						Ε	Α	М	F	Z	#	Ρ	С	T	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С
5F						Γ	Г		Г	Г	Г	Г	Г	П					П		0	0	•	•	•	7

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



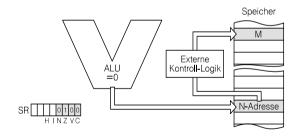
Motorola	SET	Funktion	CPU Typ A / 6303
B&R	SET	1 ⇒ (M)	
Kurz			

۸4	roccio	runae	arten	/ Once	,do	Г			^	4	eí	2 . ,	^-	١	ah	ı	n				О	•	P	310	000	
Au	CSSIC	ungs	arterr	Орсс	ue				^	uı	CI	3 V	U.	vv.	aıı	116	•••				О	•	Р	G-F	Š	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
		5/3				Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	Ν	Z	٧	С
		7F				Г	0	0	0	Г	П	П	П	П			П		П		0	0	▼	▲	•	₹

Mot	orola	SET		Fun	ktion												1	CP	U	T	УI) E	3 /	68	В0	9
B&F	ł	SET		1 ⇨	(M)																					
Kurz	z																									
ЬA	ressie	runas	arten	/ Once	ode	Γ			Α	dr	eí	۱۷	or	w	ah	le	n				0			P8		
						L							-								О	L	P	33	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
		7/3				Е	Α	М	F	Z	#	Р	С	_	Υ	D	U	!	В	G	Н	1	Ν	Z	٧	С

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!

B&F	₹	CLC	;	0 ⇒	С																					
Kurz	z																									
۸ ۵۰	roccio	runae	arten	/ Once	ndo.	Γ			^	4	eí	2 . ,			a k	ı	n				О	,	PC	310	000	
Aui	CSSIC	ungs	arterr	Орсс	Jue				^	u	CI	3 V	U.	vv.	aı		•••				O	,	Р	G-F	Š	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	usr	eg	ist	er
1/1						Е	Α	М	F	z	#	Р	С	1	Υ	D	U	!	В	G	Н	ı	N	Z	٧	С

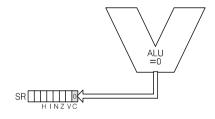
I		Mot	orola	CLC	:	Fun	ktion												1	CI	P	JΤ	У	o E	3 /	68	30	9
1	Г	B&R	1	CLC	;	0 ⇒	С																					
l	Г	Kurz	Z																									
1	Г	Δdı	essie	runas	arten	/ Once	nde	Γ			Δ	dr	eſ	١٧	٥r	w	ah	le	n				О	<u> </u>		Р8	_	
ı		7.01	000.0	ungo	ui toii i	Орос	Juc	L			_	۵.	٠.	•	٠.	•••		٠.	••				О)	Ρ	C8	0	
1	П	MPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atι	ısr	egi	ist	ei
1	Г	3/2						Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	I	Ν	z	٧	С
1	1	C FE						Г	Г	Γ	Г	Г	П	Γ	П								0	0	0	0	0	٧

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

Funktion

CLC

Motorola



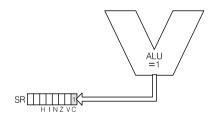
CPU Typ A / 6303

Motorola	SEC	Funktion		CPU Typ	A / 6303
B&R	SEC	1 ⇒ C			
Kurz					
			 	0	PG1000

٨٨	roccio	runaa	arten	/ Once	, do	Г			_	٦.	eſ	· · ·			-h		<u>_</u>				О	7	P	31 0	000	Л
Au	essie	rungs	arterr	Opco	ue				^	uı	eı	٥v	Oi	w	411	iie	"				O	•	Р	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	ısı	eg	ist	er
1/1						Ε	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Н	1	Ν	Z	٧	С
0D						Г	П	Г	Г	Г		П	П	П					П		0	0	0	0	0	▲

Mot	orola	SEC	;	Fun	ktion												•	CF	٧,	ΙT	УĮ	o E	3 /	68	309	9
B&F	₹	SEC	;	1 ⇨	С																					
Kur	z																									
Ad	ressie	runas	arten	/ Opco	ode	Γ			Α	dr	eí	3v	or	wa	ah	le	n				О	-		Р8	_	
						L						•					•••				O	<u> </u>	P	C 8	0	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atι	ısr	egi	iste	er
3/2						Ε	Α	М	F	Z	#	Ρ	С	1	Υ	D	U	!	В	G	Н	I	N	z	٧	С
1 0 01						г	П	Т	Т	г	г	П	П					П	П		\sim	$\overline{}$	\overline{a}	$\overline{}$	$\overline{}$	_

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



	B&F	₹	CLI		0 ⇒	ı																						l	B&F	1	CLI	
	Kurz	z																											Kurz	z		
	Adı	ressie	rungs	arten	/ Opco	ode	Γ			Α	dr	eß	Sve	or	wa	ah	le	n				0	F	P			_	1	Adı	ressie	rungs	arte
1	IMPL.	DIR.	FXT.	IMMED.	IND.	REL.	١.	_	-	s	₊ I	4	ы	ы	νI	νI	ьΙ		. 1	ы		St) of i			-		1	IMPL.	DIR.	EXT.	IMME
Į		DIK.	EAI.	IIVIIVIED.	IND.		┺	Ľ	_		_	_	_	_	_	_	_	-	_	_		-		_		<u>, </u>	-	1		DIK.	EXI.	IIVIIVIE
	1/1						Ε	Α	М	F	ΖĮ	#	P	ျ	1	Υ	ᄓ	미	!	В	G	н	1	Ν	Z	V	lc	ı	3/2		1	

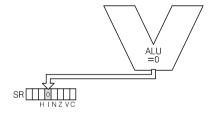
Motorola CLI

Funktion

Mot	orola	CLI		Fun	ktion												1	CF	P	ΙT	УI	o E	3 /	6	В0	9
B&F	ł	CLI		0 ⇒	I																					
Kur	z																									
Δd	rossio	runas	arten	/ Once	nde	Γ			Δ	dr	ام:	ŝν	۸r	۱۸/	ah	ما	n				С)	С	P	30	
Au	CSSIC	ungs	anten	Opce	Jue				_	uı	CI	3 V	UI	w	211	16					U)	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
3/2						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Ι	I	Ν	Z	٧	С
1C EF																					0	•	0	0	0	0

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

CPU Typ A / 6303

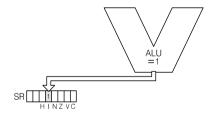


Motorola	SEI	Funktion	CPU Typ A / 6303
B&R	SEI	1⊅1	
Kurz			

۸4	roccio	runae	arten	/ Once	,do	Г			^	4,	eí	2 . ,	^-	١	ah	ı	n				О	ī	P	310	000	П
Au	CSSIC	ungs	arterr	Opce	ue				^	uı	CI	3 V	U.	vv.	aıı	16	"				О	7	Р	G-F	Š	П
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	ļ	В	G	St	atı	ısı	er		
1/1						Е	Α	М	F	Z	#	Р	С	I	Υ	D	U	!	В	G	Η	I	Ν	Z	٧	С
0F						Г	Г	П	П		Г	П	П	П			П				0	•	0	0	0	ା

Mote	orola	SEI		Fun	ktion													CI	٦,	JΤ	y	рE	3 /	6	80	9
B&R		SEI		1 ⇨	I																					
Kurz	Z																									
۸۵.	ressie	unae	arton	Once	,do	Γ			_	dr	~1	2.,		\A/-	ah	ı	n				C	<u>, </u>	С	P	80	
^u	CSSICI	ungs	arterr	Opce	ue				^	uı	C.	3 V	U.	vv.	aıı	16	•••				O)	Р	C	80	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	υ	!	В	G	St	atı	ısı	eg	ist	er
3/2						Ε	Α	М	F	Z	#	Р	С	T	Υ	D	U	!	В	G	Н	ı	N	z	٧	С
1A 10						Γ		Г	П		Г		Γ			Г					0	•	0	0	0	ા

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.

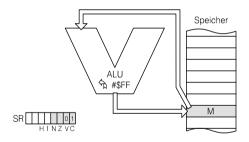
Bevor der Befehl END abgearbeitet wird, muß das Bit I mit dem Befehl CLI unbedingt wieder auf log. 0 gesetzt werden.

Mot	orola	ola	COV	1	Fun	ktion												1	CF	PU	ı T	уp	F	4 / 6303
B&F	₹		K		(M)	⊕ #\$FI	F	⇨	(N	1)	(=	=>	N	le	ga	ti	or	1)						
Kurz	z																							
۸ ۵۰	roccio	cior	unae	arten	Once	,do	Г			۸.	dra	eß	.,,	\r		, h	ı	n				0		PG1000
Aui	CSSIC	SICI	ungs	ai teii i	Opce	ue																0		PG-PC
IMPL.	DIR.	IR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	тТ	#	PΤ	R	x	Υ	D	U	!	В	G	Sta	tυ	sregiste

6/3 6/2 73 63

	Mote	orola	CON	/	Funi	ktion												(CP	U	T	Уľ	E	3 /	68	309	Э
١	B&R		K		(M)	⊕ #\$F	F	⇒	(N	1)	(=>	N	leç	ga	tic	on)									
	Kurz																										
	Δdı	ressiei	runas	arten /	Once	nde	Γ			Δ	dr	e (lve	٦r١	MS	ah	lei	n			╗	0	Ī		P 8	_	
																						0	١.	Р	C 8	80	
	IMPL.	DIR.		IMMED.	IND.	REL.	ı	0	F	S	Т	#	Р	R	х	Υ	D	U	!	3	G	_		P(er
	IMPL.	DIR. 6/2		IMMED. 6+/2+	IND.		ш	O A	ш			ш	_	_	-	_	D D	-	_	_	G G	Sta	atu			iste	er

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

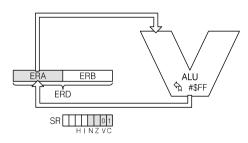


Motorola	COMA	Funktion			CPU Typ	A / 6303
B&R	COA	ERA ⊕ #\$FF	⇒ ERA	(=> Negation	n)	
Kurz	KA					
						DC1000

٨٨	Adressierungsarten / Opcode								^	4	reí	2 1/			ak	ı۱۸	n				С	√	P	G1(000)
Au									~	u	CI	3 V	U	vv.	aı		•••				C	,	P	G-I	2	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	!	В	G	St	atı	usi	eg	ist	tei
1/1						Ε	Α	М	F	z	#	Р	С	I	Υ	D	U	!	В	G	Н	Ι	Ν	z	٧	С
43						Г	Г	Г	Г	Г	П	Г	Г	П	Г	Г	Г	П	П		0	0	•	•	▼	4

_																											
	Mot	orola	COI	VΙΑ	Fun	ktion												1	CI	?(J 1	У	o E	B /	6	80	9
	B&F	₹	CO	4	ER#	₩\$	FI	= =	⇒	EF	RA		(=	>	Ne	g	at	io	n)								
	Kurz	z	KA		1																						
	Δd	roccio	runas	arton	n / Opcode Adreßvorwahlen O CP 8													30									
	Λu	163316	unga	arten	Opce	Jue					u	٠.	3 4	٠.	**	an						C)	Ρ	C	30	
	IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atı	ısr	eg	ist	er
	2/1						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	н	I	N	Z	٧	С
	43						Г	П	Г	Г			П	Г	Г					Г		0	0	•	•	▼	•

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



WIOL	UlUla	- 001	VID.	Full	KUUII													Ç.		<u>' '</u>	УI	,	٦,	U.	30	ച
B&F	₹	COI	3	ERE	3 ⊕ #\$	FI	= =	⇒	EF	RВ		(=	>	Ne	eg	at	io	n)								
Kurz	z	КВ																								
Δdı	rossio	runas	arten	/ Opcode Adreßvorwahlen O PG1000														╝								
Aui	COOL	unga	arterr	Орсс	Jue					u	٠.	3 V	٠.	***	an						О	•	Р	G-F	C	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	Т	#	Ρ	R	Х	Υ	D	U	!	В	G	St	atı	ısı	eg	ist	er
1/1						Е	Α	М	F	Z	#	Ρ	С	I	Υ	D	U	!	В	G	Ι	Ι	Ν	Z	<	С

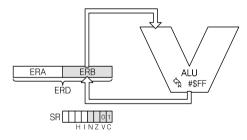
Eunktion

Motorolo COMP

Mot	orola	CON	ИΒ	Fun	ktion												1	CI	٦,	JΤ	УI	рE	B /	6	80	9
B&F	1	COE	3	ERE	3 ⊕ #\$	FI	= =	>	EF	RВ		(=	>	Ne	g	at	io	n)								
Kurz	Z	КВ		1																						
Adressierungsarten / Opcode Adreßvorwahle													ما	n				O	7	С	P	30				
Au	63316	ungs	arterr	Opci	Jue	L				u	٠.	3 4	٠.	**	u						С)	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	х	Υ	D	U	ļ	В	G	St	atı	ısı	eg	ist	er
2/1						Ε	Α	М	F	Z	#	Р	С	1	Υ	D	U	!	В	G	Η	I	N	Z	٧	С
53						Г															0	0	•	•	▼	•

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

CBIL Typ. A / 6202



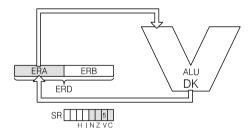
Mot	orola	DA	Α	Fun	ktion												•	CI	? (ΙŢ	уp	Α	/ 6303
B&F	₹	DK		Dez	Dezimalkorrektur von ERA nach einer Addition																		
Kurz	z			von BCD Zahlen																			
۸ ۵۰	roccio	runae	arten /	Once	,do	Γ			^	dr	~u		۸r		a h	ı	n				0		PG1000
Aui	C3516	ungs	ai teli /	Opco	Jue				^	uI	C 13	٧,	UI	w	211	ıe	•••				0		PG-PC
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ı	0	F	s	т	#	Р	R	х	Υ	D	U	!	В	G	Sta	ıtu	sregiste

19

Mot	orola	DAA	١	Fun	ktion												1	CI	2 U	T	УĮ	o E	3 /	6	B0	9
B&F	t		Dezimalkorrektur von ERA nach einer Addition																							
Kurz von BCD Zahlen																										
Adressierungsarten / Opcod					ode	Γ			Α	dr	eí	ŝν	or	wa	ah	le	n				С	-		Pδ		
		9-				L															U	<u> </u>	Р	C	30	
IMPL.	DIR.	EXT.	IMMED.	IND.	REL.	ī	0	F	s	Т	#	Р	R	Х	Υ	D	U	!	В	G	St	atι	ısr	eg	ist	er
2/1						Ε	Α	М	F	Z	#	Р	С	_	Υ	D	U	!	В	G	Н	1	N	Z	٧	С
19						Г	Г	Г	Г			Г	Г			П		Г			0	0	•	•	×	•

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

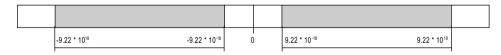
Einfachgenaues Fließkommaformat



Umrechnung:

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

Darstellbarer Zahlenbereich:



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

± 32767

±(2¹⁵ - 1)

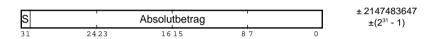
Absolut mit Vorzeichen

S ... Vorzeichen

Absolut-Kurz

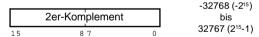


Absolut-Lang



Integer

Integer-Kurz



Integer-Lang



MADD

Addition im Fließkommaformat

Ausführungszeit	6303	209 - 690	Mögliche	1	2	3	4	5	6	7	8	9	10	12	13	14
in µs	6809	207 - 503	Fehlermeldungen	0	0				0	0				0		

Funktion: Die Operanden OP1 und OP2 werden addiert. Das Ergebnis wird in OP1 gespeichert. OP2 bleibt unverändert.

Übergabe: keine

Ergebnis: ERD verändert

N, Z entsprechend dem Ergebnis der Routine

MSUB

Subtraktion im Fließkommaformat

Ausführungszeit	6303	219 - 700	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	213 - 509	Fehlermeldungen	0	0				0	0					0		

Funktion: Der Operand OP2 wird von OP1 subtrahiert. Das Ergebnis wird in OP1 gespeichert. OP2 bleibt unverändert.

Übergabe: keine

Ergebnis: ERD verändert

N, Z entsprechend dem Ergebnis der Routine

MMUL Multiplikation im Fließkommaformat

Ausführungszeit	6303	209 - 803	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	209 - 563	Fehlermeldungen	0	0				0	0					0		

Funktion: Die Operanden OP1 und OP2 werden multipliziert. Das Ergebnis wird in OP1 gespeichert. OP2 bleibt unverändert.

Übergabe: keine

Ergebnis: ERD verändert

N, Z entsprechend dem Ergebnis der Routine

MDIV Division im Fließkommaformat

Ausführungszeit	6303	190 - 1980	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	227 - 1390	Fehlermeldungen	0	0	0			0	0					0		

Funktion: Der Operand OP1 wird durch OP2 dividiert. Das Ergebnis wird in OP1 gespeichert. OP2 bleibt unverändert.

Übergabe: keine

Ergebnis: ERD verändert

N, Z entsprechend dem Ergebnis der Routine

MSQR

Quadratwurzel im Fließkommaformat

Ausführungszeit	6303	71 - 8065	Mögliche	1	2	3	4	5	6	7	8	9	10	12	13	14
in µs	6809	123 - 5100	Fehlermeldungen	0					0	0	0			0		

Funktion: 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.

Übergabe: keine

Ergebnis: ERD verändert

N, Z entsprechend dem Ergebnis der Routine

MSGN

Vorzeichenwechsel von Operand 1

Ausführungszeit	6303	85	Mögliche	1	2	3	4	5	6	7	8	9	10	12	13	
in µs	6809	126	Fehlermeldungen													

Funktion: Das Vorzeichen von OP1 wird invertiert. Die Operation entspricht einer Multiplikation mit -1. OP2 bleibt unverändert.

Übergabe: keine

Ergebnis: ERD verändert

C N. Z

MCOP Operand OP1 nach Operand OP2 kopieren

Ausführungszeit	6303	46	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	115	Fehlermeldungen														

Funktion: Der Operand OP1 wird nach OP2 kopiert.

Übergabe: keine

Ergebnis: ERD verändert

MEXG Die Operanden OP1 und OP2 miteinander vertauschen

Ausführungszeit	6303	76	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	125	Fehlermeldungen														

Funktion: Die Operanden OP1 und OP2 werden miteinander vertauscht

Übergabe: keine

Ergebnis: ERD verändert

LAL1 Lade Operand OP1 mit Zahl (Absolut-Lang)

Ausführungszeit	6303	190 - 339	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	193 - 293	Fehlermeldungen					О									

Funktion: 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.

Übergabe: R Quelladresse der Binärzahl Beispiel: Operand OP1 mit Binärzahl aus den 8-Bit Speicherstellen C 320 bis C 323 laden:

Ergebnis: ERD verändert

N, Z entsprechend dem Ergebnis der Routine

LRK C 0320 Quelladresse der Binärzahl

LAL1

LAL2 | Lade Operand OP2 mit Zahl (Absolut-Lang)

Ausführungszeit	6303	190 - 339	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	193 - 293	Fehlermeldungen					0									

Funktion: 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.

Übergabe: R Quelladresse der Binärzahl

Ergebnis: ERD verändert

LAW1 Lade Operand OP1 mit Zahl (Absolut-Kurz)

Ausführungszeit	6303	83 - 250	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	125 - 241	Fehlermeldungen														

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 entsprech

entsprechend dem Ergebnis der Routine

LAW2 Lade Operand OP2 mit Zahl (Absolut-Kurz)

Ausführungszeit	6303	83 - 247	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	125 - 240	Fehlermeldungen														

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

LIL1 Lade Operand OP1 mit Zahl (Integer-Lang)

Ausführungszeit	6303	197 - 381	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	201 - 315	Fehlermeldungen					0									

Funktion: 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.

Übergabe: R Quelladresse der Binärzahl

Ergebnis: ERD verändert

N, Z entsprechend dem Ergebnis der Routine

LIL2 | Lade Operand OP2 mit Zahl (Integer-Lang)

Ausführungszeit	6303	194 - 378	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	198 - 315	Fehlermeldungen					О									

Funktion: 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.

Übergabe: R Quelladresse der Binärzahl

Ergebnis: ERD verändert

LIW1 Lade Operand OP1 mit Zahl (Integer-Kurz)

Ausführungszeit	6303	87 - 260	Mögliche	1	2	3	4	5	6	7	8	9	10	12	13	14
in µs	6809	128 - 249	Fehlermeldungen													

Funktion: Die Binärzahl (Format: Integer-Kurz) im Ergebnisregister ERD wird in IEEE-Format umgewandelt und im Operanden OP1

gespeichert.

Übergabe:ERDBinärzahlErgebnis:ERDverändert

N, Z entsprechend dem Ergebnis der Routine

LIW2 | Lade Operand OP2 mit Zahl (Integer-Kurz)

Ausführungszeit	6303	84 - 257	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	126 - 247	Fehlermeldungen														

Funktion: Die Binärzahl (Format: Integer-Kurz) im Ergebnisregister ERD wird in IEEE-Format umgewandelt und im Operanden OP2

gespeichert.

Übergabe:ERDBinärzahlErgebnis:ERDverändert

LF1 Lade Operand OP1 mit Zahl (IEEE-Format)

Ausführungszeit	6303	88 - 125	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	129 - 146	Fehlermeldungen	0					0	0					0		

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	6303	88 - 125	Mögliche	1	2	3	4	5	6	7	8	9	10	12	13	14
in µs	6809	127 - 144	Fehlermeldungen	0					0	0				0		

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 entsprech

SAL	Operai	nd OP1 im	Format Absolut-l	ang abspeichern														
Ausführu	ngszeit	6303	169 - 408	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µ	s	6809	189 - 339	Fehlermeldungen				0								0		

Funktion: Der Operand OP1 wird in eine Binärzahl (Format: Absolut-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.

Übergabe: R Zieladresse der Binärzahl

Ergebnis: ERD verändert N. Z entsprech

entsprechend dem Ergebnis der Routine

SAW Operand OP1 im Format Absolut-Kurz abspeichern

Ausführungszeit	6303	158 - 373	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	183 - 316	Fehlermeldungen				0								0		

Funktion: Der Operand OP1 wird in eine Binärzahl (Format: Absolut-Kurz) 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 (+32767 oder -32767) abgespeichert.

Übergabe: keine

Ergebnis: ERD Binärzahl

N. Z entsprechend dem Inhalt von ERD verändert

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

Funktion: 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.

darstellbare flegative bzw. positive wert (-2147403047 oder +2147403047) abgespeiche

Übergabe: R Zieladresse der Binärzahl

Ergebnis: ERD verändert

N, Z entsprechend dem Ergebnis der Routine

SIW Operand OP1 im Format Integer-Kurz abspeichern

Ausführungszeit	6303	158 - 380	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	183 - 321	Fehlermeldungen				0								0		

Funktion: 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) abgespei-

chert.

Übergabe: keine

Ergebnis: ERD Binärzahl

N. Z entsprechend dem Inhalt von ERD verändert

SFX	Operai	nd OP1 im	IEEE-Format abspe	ichern														
Ausführu	ngszeit	6303	43	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µ	s	6809	112	Fehlermeldungen														

Funktion: Der Operand OP1 wird im Fließkommaformat (IEEE-Format) abgespeichert. Das Indexregister R enthält die Zieladresse, an der die Fließkommazahl abgespeichert wird. Die Operanden OP1 und OP2 werden dabei nicht verändert.

Übergabe: R Zieladresse der Fließkommazahl

Ergebnis: ERD verändert

C, N, Z 0

SFM1 Operand OP1 im Zwischenspeicher 1 speichern

Ausführungszeit	6303	60	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	116	Fehlermeldungen														

Funktion: Der Operand OP1 wird als Fließkommazahl (IEEE-Format) im Zwischenspeicher 1 (C1034 bis C1037) gespeichert.

Übergabe: keine

Ergebnis: ERD verändert

SFM2 Operand OP1 im Zwischenspeicher 2 speichern

Ausführungszeit	6303	60	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	116	Fehlermeldungen														

Funktion: Der Operand OP1 wird als Fließkommazahl (IEEE-Format) im Zwischenspeicher 2 (C1038 bis C1041) gespeichert.

Übergabe: keine

Ergebnis: ERD verändert

SFM3 Operand OP1 im Zwischenspeicher 3 speichern

Ausführungszeit	6303	60	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	116	Fehlermeldungen														

Funktion: Der Operand OP1 wird als Fließkommazahl (IEEE-Format) im Zwischenspeicher 3 (C1042 bis C1045) gespeichert.

Übergabe: keine

Ergebnis: ERD verändert

RFM1 Operand OP2 aus Zwischenspeicher 1 laden

Ausführungszeit	6303	56	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	116	Fehlermeldungen														

Funktion: Der Operand OP1 wird mit der Fließkommazahl (IEEE-Format) aus Zwischenspeicher 1 (C1034 bis C1037) geladen.

Übergabe: keine

Ergebnis: ERD verändert

RFM2 Operand OP2 aus Zwischenspeicher 2 laden

Ausführungszeit	6303	56	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	116	Fehlermeldungen														

Funktion: Der Operand OP1 wird mit der Fließkommazahl (IEEE-Format) aus Zwischenspeicher 2 (C1038 bis C1041) geladen.

Übergabe: keine

Ergebnis: ERD verändert

RFM3 Operand OP2 aus Zwischenspeicher 3 laden

Ausführungszeit	6303	56	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	116	Fehlermeldungen														

Funktion: Der Operand OP1 wird mit der Fließkommazahl (IEEE-Format) aus Zwischenspeicher 3 (C1042 bis C1045) geladen.

Übergabe: keine

Ergebnis: ERD verändert

FM2B	2 Byte	X 2 Byte	Multiplikation														
Ausführu	ngszeit	6303	115 - 191	Mögliche	1	2	3	4	5	6 7	' 8	9	10	11	12	13	14
in µ	s	6809	125 - 175	Fehlermeldungen													
Funktion:	Zwei Bina Format.	ärzahlen (Forn	nat: Integer-Kurz) we	rden miteinander multipliziert. I	Das	Erg	ebn	is is	t ein	e Za	hl im	Int	egei	-Laı	ng		
Übergabe:	R ERD C1048&	Multiplikat	sse des Multiplikand or se des Ergebnisses	en													
Ergebnis:	ERD C, N, Z R	verändert ungültig unverände															

FM3B	3 Byte X 2 Byte Multiplikation
	3 byte A 2 byte widitiplikation

Ausführungszeit	6303	156 - 270	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	150 - 225	Fehlermeldungen														П

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

																		198
FM4B	4 Byte	X 2 Byte	Multiplikation															
Ausführur	ngszeit	6303	192 - 344	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	s [6809	172 - 272	Fehlermeldungen														
Funktion:		im Integer-La te Integerzahl	. , ,	rd mit einer Zahl im Integer-Ku	ırz F	orn	nat	(2 E	Byte:) mı	ultip	olizie	ert.	Das	Er	gebi	nis is	st
Übergabe:	R ERD C1048&	Multiplika	sse des Multiplikande tor (Integer-Kurz) se des Ergebnisses (6	· · · · · · · · · · · · · · · · · · ·														
Ergebnis:	ERD	verändert																

C, N, Z R ungültig unverändert

Ascil-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

Funktion: Der ASCII-String, auf den das Indexregister R zeigt, wird in das interne IEEE-Format umgewandelt und in OP1 gespeichert.

OP2 wird nicht verändert.

Übergabe: R Quelladresse des ASCII-Strings

Ergebnis: ERD verändert

N, Z ungültig

Syntaxregeln für den ASCII-String:5

- 1. Für das Vorzeichen sind <+>, <-> und <Leerzeichen> zulässig. Das Vorzeichen kann auch entfallen.
- 2. Die *Mantisse* 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 <E> oder ein Leerzeichen verwendet.
- 3. Der *Exponent* ist zweistellig plus Vorzeichen und beginnt mit einem <E> nach der letzten Mantissenziffer bzw. nach dem Trennzeichen (Leerzeichen).
- Zulässige Zeichen sind <0> bis <9>, <Leerzeichen>, <->, <+>, <Dezimalpunkt> und <E>. 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.
- Nach einem <E> werden automatisch max. 3 Zeichen eingelesen, ansonsten muß der String mit <CR> (\$0D) oder <0> (\$00) abgeschlossen sein.

Beispiele für gültige ASCII-Strings:

																	20
CFA	OP1 in	ASCII ohr	ne Vornullen wande	In													
Ausführu	ngszeit	6303	352 - 7310	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12 1	3 14
in µ	s	6809	352 - 4440	Fehlermeldungen	0			0		0	0					0	
Funktion:	abgespei Die Anza chen und	ichert. Die Ope Ihl der signifika I Dezimalpunkt	I in einen ASCII-String umg rranden OP1 und OP2 werd Inten Ziffern ist auf max. 7 I) lang sein. Vorlaufende Nu Ilt werden, wird der String n	den nicht verändert. beschränkt. Der ASCII-St ullen werden durch Leerz	ring	j kai	nn c	lahe	er ma	ax.	9 Z	eich	nen	(inkl	I. Vo		

Übergabe: R Zieladresse des ASCII-Strings

7 6 5 4 3 2 1 0

Länge des gesamten ASCII-Strings - Anzahl der Dezimalstellen

ERA Format des ASCII-Strings:

Ergebnis: ERA verändert

ERB Länge des ASCII-Strings

N, Z

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. : : LRK C 0250 Zieladresse für den ASCII-String in das Indexregister R laden Länge des ASCII-Strings (5 signifikante Stellen + Vorzeichen + Dezimalpunkt = 7) -Anzahl der Dezimalstellen (2) Format des ASCII-Strings in ERA übergeben LAD # \$72 CFA OP1 = 32.12332.12" OP1 = -0.1"- 0.10" OP1 = 4.874.87" OP1 = 2300.25 => " >>> >>"

"-<<<.<"

=>

OP1 = -1000.25

Ausführungszeit in µs 6809 310 - 7190 Mögliche Fehlermeldungen 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		6303	310 - 7190		Mögliche	1 2	3	4	5	6 7	8	19	101	11 1	2 1	3 1/
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 Länge des gesamten ASCII-Strings Länge des gesamten ASCII-Strings Anzahl der Dezimalstellen ERA Format des ASCII-Strings: T, 6, 5, 4, 3, 2, 1 Ergebnis: ERA verändert ERB Länge des ASCII-Strings					_		Ť							-	\pm	
Ergebnis: ERA verändert ERB Länge des ASCII-Strings Länge des gesamten ASCII-Strings Anzahl der Dezimalstellen 7, 6, 5, 4, 3, 2, 1, 0	abgespei Die Anza chen und	chert. Die Ope hl der signifika Dezimalpunkt	eranden OP1 und (Inten Ziffern ist au i) lang sein. Vorlau	OP2 werde of max. 7 be ufende Nul	en nicht verändert. eschränkt. Der ASCII-St llen werden nicht unterdi	ring ka	nn c	laher	m	ax. 9 Z	Zeich	nen	(ink	I. Vor		
ERB Länge des ASCII-Strings					Anz	0	_				Strii	ngs				
	ERB	Länge des														
		abgespei Die Anza chen und Format de R ERA ERA ERB	Der Inhalt von OP1 wird abgespeichert. Die Ope Die Anzahl der signifikaten und Dezimalpunkt Format dargestellt werd R Zieladress ERA Format der ERA verändert ERB Länge des	Der Inhalt von OP1 wird in einen ASCII-S abgespeichert. Die Operanden OP1 und Die Anzahl der signifikanten Ziffern ist au chen und Dezimalpunkt) lang sein. Vorlat Format dargestellt werden, wird der Strin R Zieladresse des ASCII-String ERA Format des ASCII-Strings: ERA verändert ERB Länge des ASCII-Strings	Der Inhalt von OP1 wird in einen ASCII-String umge abgespeichert. Die Operanden OP1 und OP2 werd Die Anzahl der signifikanten Ziffern ist auf max. 7 b chen und Dezimalpunkt) lang sein. Vorlaufende Nul Format dargestellt werden, wird der String mit ">" ar R Zieladresse des ASCII-Strings ERA Format des ASCII-Strings: 7,6,	Der Inhalt von OP1 wird in einen ASCII-String umgewandelt und ab der im I abgespeichert. Die Operanden OP1 und OP2 werden nicht verändert. Die Anzahl der signifikanten Ziffern ist auf max. 7 beschränkt. Der ASCII-St chen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden nicht unterdi Format dargestellt werden, wird der String mit ">" aufgefüllt. R Zieladresse des ASCII-Strings ERA Format des ASCII-Strings: 7, 6, 5, 4, 3, 2, 1, 0 ERA verändert ERB Länge des ASCII-Strings	Der Inhalt von OP1 wird in einen ASCII-String umgewandelt und ab der im Indexre abgespeichert. Die Operanden OP1 und OP2 werden nicht verändert. Die Anzahl der signifikanten Ziffern ist auf max. 7 beschränkt. Der ASCII-String katchen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden nicht unterdrückt. Format dargestellt werden, wird der String mit ">" aufgefüllt. R Zieladresse des ASCII-Strings Länge des Anzahl de ERA Format des ASCII-Strings: 7, 6, 5, 4, 3, 2, 1, 0 ERA verändert ERB Länge des ASCII-Strings	Der Inhalt von OP1 wird in einen ASCII-String umgewandelt und ab der im Indexregiste abgespeichert. Die Operanden OP1 und OP2 werden nicht verändert. Die Anzahl der signifikanten Ziffern ist auf max. 7 beschränkt. Der ASCII-String kann ochen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden nicht unterdrückt. Kann Format dargestellt werden, wird der String mit ">" aufgefüllt. R Zieladresse des ASCII-Strings Länge des ges Anzahl der Des ERA verändert ERB Länge des ASCII-Strings	Der Inhalt von OP1 wird in einen ASCII-String umgewandelt und ab der im Indexregister R a abgespeichert. Die Operanden OP1 und OP2 werden nicht verändert. Die Anzahl der signifikanten Ziffern ist auf max. 7 beschränkt. Der ASCII-String kann daher chen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden nicht unterdrückt. Kann OP1 Format dargestellt werden, wird der String mit ">" aufgefüllt. R Zieladresse des ASCII-Strings Länge des gesamte Anzahl der Dezimal ERA verändert ERB Länge des ASCII-Strings	Der Inhalt von OP1 wird in einen ASCII-String umgewandelt und ab der im Indexregister R and abgespeichert. Die Operanden OP1 und OP2 werden nicht verändert. Die Anzahl der signifikanten Ziffern ist auf max. 7 beschränkt. Der ASCII-String kann daher met chen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden nicht unterdrückt. Kann OP1 ni Format dargestellt werden, wird der String mit ">" aufgefüllt. R Zieladresse des ASCII-Strings Länge des gesamten Anzahl der Dezimalste ERA verändert ERB Länge des ASCII-Strings	Der Inhalt von OP1 wird in einen ASCII-String umgewandelt und ab der im Indexregister R angegebe 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 Zehen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden nicht unterdrückt. Kann OP1 nicht im Format dargestellt werden, wird der String mit ">" aufgefüllt. R Zieladresse des ASCII-Strings Länge des gesamten ASCII-Anzahl der Dezimalstellen ERA verändert ERB Länge des ASCII-Strings	Der Inhalt von OP1 wird in einen ASCII-String umgewandelt und ab der im Indexregister R angegebener 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 Zeicl chen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden nicht unterdrückt. Kann OP1 nicht im gev Format dargestellt werden, wird der String mit ">" aufgefüllt. R Zieladresse des ASCII-Strings Länge des gesamten ASCII-Strings ERA Format des ASCII-Strings: 7, 6, 5, 4, 3, 2, 1, 0 ERA verändert ERB Länge des ASCII-Strings	Der Inhalt von OP1 wird in einen ASCII-String umgewandelt und ab der im Indexregister R angegebenen Ad 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 chen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden nicht unterdrückt. Kann OP1 nicht im gewüns Format dargestellt werden, wird der String mit ">" aufgefüllt. R Zieladresse des ASCII-Strings Länge des gesamten ASCII-Strings Länge des gesamten ASCII-Strings ERA verändert ERB Länge des ASCII-Strings	Der Inhalt von OP1 wird in einen ASCII-String umgewandelt und ab der im Indexregister R angegebenen Adress 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 (ink chen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden nicht unterdrückt. Kann OP1 nicht im gewünscht Format dargestellt werden, wird der String mit ">" aufgefüllt. R Zieladresse des ASCII-Strings Länge des gesamten ASCII-Strings Länge des gesamten ASCII-Strings ERA verändert ERB Länge des ASCII-Strings	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. Vor chen und Dezimalpunkt) lang sein. Vorlaufende Nullen werden nicht unterdrückt. Kann OP1 nicht im gewünschten ASFormat dargestellt werden, wird der String mit ">" aufgefüllt. R Zieladresse des ASCII-Strings Länge des gesamten ASCII-Strings Länge des gesamten ASCII-Strings ERA verändert ERB Länge des ASCII-Strings	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. R Zieladresse des ASCII-Strings Länge des gesamten ASCII-Strings Länge des gesamten ASCII-Strings ERA verändert ERB Länge des ASCII-Strings

Beispiel: 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. : : LRK C 0100 Zieladresse für den ASCII-String in das Indexregister R laden Länge des ASCII-Strings (4 signifikante Stellen + Vorzeichen + Dezimalpunkt = 6) Anzahl der Dezimalstellen (1) Format des ASCII-Strings in ERA übergeben LAD # \$61 CFA 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 2 3 5 6 8 9 10 11 12 13 14 Ausführungszeit 6303 570 - 7140 Möaliche 4 6809 267 - 4140 Fehlermeldungen in µs 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). VM Mantissenvorzeichen (Leerzeichen oder <->) Mantisse 7 signifikante Ziffern + Dezimalpunkt Leerzeichen ASCII-Zeichen <E> Exponentyorzeichen (<+> oder <->)

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

Exp.

Übergabe: R Zieladresse des ASCII-Strings

Länge des gesamten ASCII-Strings Anzahl der Dezimalstellen

Exponent (zwei Ziffern)

ERA Format des ASCII-Strings:

Ergebnis:

FRA verändert

FRB Länge des ASCII-Strings

Ζ entsprechend dem Ergebnis der Routine

E VE Exp.

Ν ungültig Beispiel: OP1 soll in einen ASCII-String mit Exponentdarstellung umgewandelt und ab C0500 abgespeichert werden.

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

CIA	Binär i	n ASCII of	nne Vornullen	n wande	eln														
Ausführu	ngszeit	6303	380 - 2020		Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µ	s	6809	357 - 1370		Fehlermeldungen				0										1
Funktion:	chert. Die Die Anza Vorzeiche	e Operanden C hl der signifika en und Dezima	PP1 und OP2 werd Inten Ziffern ist au Alpunkt) lang sein.	den nicht v f max. 9 b Vorlaufen	delt und ab der im Index erändert. eschränkt. Der ASCII-St de Nullen werden durch n, wird der String mit ">"	ring Lee	kar rze	n d	lahe en e	r m	ax. 1 tzt. K	1 Z	Zeic	hen	ı (in	kl.			
Übergabe:	C1046& R ERA	Zieladress	sse der Binärzahl se des ASCII-Strin es ASCII-Strings:	7 6	Anz						en Si ellen		en						
Ergebnis:	ERB ERA ERB N, Z	verändert	er Binärzahl: s gesamten ASCII-			mat nge d						1 =	= At = In = 2 = 4	tege Byte	er e				

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. : Quelladresse der Binärzahl LRK C 0100 C 1046 Abspeichern der Quelladresse in der 8 Bit Speicherstelle C1046 =R C 3000 Zieladresse für den ASCII-String LRK Anzahl der signifikanten Stellen Anzahl der Dezimalstellen Format des ASCII-Strings in Ergebnisregister ERA übergeben LAD # \$62 Format der Binärzahl: 1 => Integer Länge der Binärzahl: 1 => 4 Byte Format der Binärzahl in Ergebnisregister ERB übergeben # %00000011 LB Aufruf der Routine CIA Integer-Lang = 56499 564.99"" Integer-Lang = -23 0 23"

=>

" >>>>.

Integer-Lang = 1000000

CIA0	Binär i	n ASCII m	it Vornullen	wandelr	1												
Ausführu	ngszeit	6303	310 - 1960		Mögliche	1 2	2 3	4	5	6	7	8	9	10	11	12	13 14
in µ	s	6809	312 - 1320		Fehlermeldungen			0									
Funktion:	chert. Die Die Anza Vorzeich	e Operanden C hl der signifika en und Dezima	DP1 und OP2 wer anten Ziffern ist a alpunkt) lang seir	rden nicht v uf max. 9 b n. Vorlaufer	ndelt und ab der im Index rerändert. eschränkt. Der ASCII-St ide Nullen werden nicht ing mit ">" bzw. "<" aufg	ring k unter	ann	dahe	er m	ax. 1	1 Z	'eich	nen	(in	kl.		
Übergabe:	C1046& R ERA	Zieladres	sse der Binärzah se des ASCII-Stri es ASCII-Strings:	-	Anz	zahl d zahl d					elle	en					
	ERB	Format de	er Binärzahl:	7 6		mat d					1 = 0 =	= Ab = Int = 2 E = 4 E	ege Byte	er e			
Ergebnis:	ERA ERB N, Z	verändert Länge des 0	s gesamten ASC	II-Strings													

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. : Quelladresse der Binärzahl LRK C 0100 C 1046 Abspeichern der Quelladresse in der 8 Bit Speicherstelle C1046 =R C 0200 Zieladresse für den ASCII-String LRK Anzahl der signifikanten Stellen Anzahl der Dezimalstellen Format des ASCII-Strings in Ergebnisregister ERA übergeben LAD # \$83 Format der Binärzahl: 0 => Absolut Länge der Binärzahl: 1 => 4 Byte Format der Binärzahl in Ergebnisregister ERB übergeben # %00000001 LB Aufruf der Routine CIA Absolut-Lang = 56499 " 00056.499" Absolut-Lang = -23 "-00000.023" Absolut-Lang = 1000000 " 01000.000" => Absolut-Lang = 100000000 " >>>> >>

=>

Ausführungszeit in μs 6303 192 - 1180 Mögliche Fehlermeldungen 1 2 3 4 5 6 7 8 9 10 11 12 13 1 Funktion: Eine 3 Byte Binärzahl wird in eine 3 Byte BCD-Zahl umgewandelt. Es ist nur möglich Zahlen im Bereich von 0 bis 999999 in eine 3 Byte BCD-Zahl umzuwandeln. Liegt die Binärzahl außerhalb dieses Bereiches, wird der Fehler 4 (CONV_OVERFLOW) gemeldet und alle BCD-Stellen auf 9 gesetzt (BCD-Zahl: 999999). Die Operanden OP1 und OP2 werden nicht verändert.	CBCD	Binär i	in BCD war	ndeln													
Funktion: Eine 3 Byte Binärzahl wird in eine 3 Byte BCD-Zahl umgewandelt. Es ist nur möglich Zahlen im Bereich von 0 bis 999999 in eine 3 Byte BCD-Zahl umzuwandeln. Liegt die Binärzahl außerhalb dieses Bereiches, wird der Fehler 4 (CONV_OVERFLOW) gemeldet und alle BCD-Stellen auf 9 gesetzt (BCD-Zahl: 999999). Die Operanden OP1 und OP2	Ausführu	ngszeit	6303	192 - 1180	Mögliche	1	2	3	4 !	5 6	7	8	9	10	11 1	2 13	14
eine 3 Byte BCD-Zahl umzuwandeln. Liegt die Binärzahl außerhalb dieses Bereiches, wird der Fehler 4 (CONV_OVERFLOW) gemeldet und alle BCD-Stellen auf 9 gesetzt (BCD-Zahl: 999999). Die Operanden OP1 und OP2	in µ	s	6809	227 - 852	Fehlermeldungen)								
	Funktion:	eine 3 By (CONV_	, /te BCD-Zahl u OVERFLOW) g	mzuwandeln. Liegt die Bi gemeldet und alle BCD-St	närzahl außerhalb dieses l	Bere	eich	es, v	vird (der F	ehle	er 4					n

Übergabe: ERD Quelladresse der Binärzahl

R Zieladresse der BCD-Zahl

Ergebnis: ERD niederwertigen zwei Byte der BCD-Zahl

N, Z entsprechend dem Inhalt von ERD

Beispiel: Binärzahl = 450

=>

BCD-Zahl = \$

Binärzahl = 1956 =>

BCD-Zahl = \$

Binärzahl = 1000000

BCD-Zahl = \$ 9 9 9 9 9

Die Binärzahl, die in den Speicherstellen C0100 bis C0102 gespeichert ist, soll in eine BCD-Zahl umgewandelt werden. Das Ergebnis soll ab der Speicherstelle C0290 abgespeichert werden.

:
LRK C 0100 Quelladresse der Binärzahl
DXR Quelladresse nach ERD laden
LRK C 0290 Zieladresse für BCD-Zahl
CBCD

:

CBIN

BCD in Binär wandeln

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

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 => 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

:

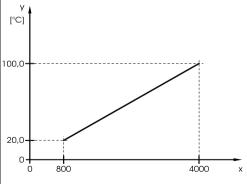
Umwandlung: Binär <=> Physikalisch (Normierung)

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:

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:

1.
$$200 = 800k + d$$

II.
$$1000 = 4000k + d$$

II.
$$1000 = 4000k + 200 - 800k$$

$$800 = 3200k$$

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

$$d = 200 - 800k = 0$$

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_i/y_i)

 $und \left(x_2/y_2\right) der \ Berechnung \ zu \ Grunde \ liegen. \ Das \ Zahlenformat \ (Integer-Kurz \ oder \ IEEE-Format) \ der \ angegebenen$

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:

$$y = kx + d \qquad => \qquad x = \frac{y - d}{k}$$

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,/y,)

und (x₂/y₂) 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 2500 - 6700 2 3 5 6 8 9 10 11 12 13 14 Ausführungszeit 6303 Möaliche 4 6809 1200 - 4200 Fehlermeldungen \mathbf{O} in µs Funktion: Diese Routine hat zwei Funktionen: 1. Berechnung der Faktoren k und d (im IEEE-Format) aus zwei Geradenstützpunkten (x,/y,) und (x,/y,). 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.

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

Funktion 0 = 1. Funktion 1 = 2. Funktion 2 = Integer-Kurz 1 = IEEE-Format Von x bzw. d: 0 = Integer-Kurz 1 = IEEE-Format Von y bzw. d: 0 = Integer-Kurz 1 = IEEE-Forma

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

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: Inhalt Ergebnisregister ERB von 0xxxxx00 0xxxxx01 0xxxxx10 0xxxxx11 1xxxxx00 1xxxxx01 1xxxxx10 1xxxxx11 Speicher Cxxxx Integer-Kurz Integer-Kurz IFFF IFFF Integer-Kurz Integer-Kurz IFFF IFFF Cxxxx Cxxxx Integer-Kurz IEEE Integer-Kurz IEEE Cxxxx Cxxxx d d IFFF Integer-Kurz Integer-Kurz Integer-Kurz IFFF Cxxxx ر2 Cxxxx IEEE Integer-Kurz Integer-Kurz Cxxxx + Cxxxx + Cxxxx + IEEE IEEE Cxxxx + Cxxxx + Integer-Kurz Cxxxx + IEEE Cxxxx + Cxxxx + Cxxxx

CBPQ	Wandle	e Binär =>	Physikalisch, schn	ell														
Ausführung	szeit	6303	780 - 1700	Mögliche	1	2	3	4	5	6	7	8	9	10	11 1	2 1	13 1	1
in µs		6809	580 - 1100	Fehlermeldungen	0					0	О				(5		1

Funktion: Umrechnung einer Binärzahl (Integer-Kurz) in einen physikalischen Wert nach der Formel y = kx + d. Die Faktoren **k** und **d**

liegen bereits als Fließkommazahlen (IEEE-Format, z.B. durch die Routine CBPP ermittelt) vor.

Übergabe: ERD Binärzahl *x* R Quelladress

Quelladresse der Geradenfaktoren k und d (IEEE-Format)

Ergebnis: ERD Physikalischer Wert y (Integer-Kurz)
OP1 Physikalischer Wert y (IEEE-Format)

R verändert

N, Z entsprechend dem Inhalt von ERD

CPBQ Wand	le Physikal	isch => Binär, schn	ell													
Ausführungszeit	6303	780 - 1500	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13 14
in µs	6809	580 - 950	Fehlermeldungen	0					0	0					0	

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

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)

Quelladresse der Geradenfaktoren k und d (IEEE-Format) R

Ergebnis: ERD Binärwert x (Integer-Kurz) OP1 Binärwert x (IEEE-Format)

R verändert

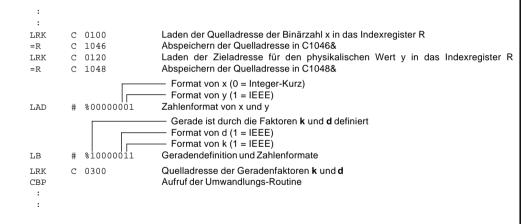
N. Z

entsprechend dem Inhalt von ERD

СВР	Wandle	e Binär =>	Physikalisch														
Ausführungszeit		6303	3400 - 8300	Mögliche	1	2	3	4 5	6	7	8	9	10	11	12	13	14
in µs		6809	ca. 4000	Fehlermeldungen	0				0	0					0		
Funktion:	gerade w Da die Be Aufruf er errechne	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 \mathbf{k} und \mathbf{d} definiert. Da die Berechnung der Faktoren \mathbf{k} und \mathbf{d} aus den Geradenstützpunkten relativ lange dauert, sollte sie nicht bei jedem Aufruf erfolgen. Die Faktoren \mathbf{k} und \mathbf{d} sollten deshalb mit der Routine CBP in einem Initialisierungs-Programmteil errechnet und zwischengespeichert werden. Bei Aufruf der Routine CBP werden diese bereits errechneten Faktoren zur Verfügung gestellt.															
Übergabe:	C1046& C1048& R ERA	Zieladress															
	ERB	7 6 5 4	3 2 1 0	Geradendefinition: Zahlenformat von x bzw. k: Zahlenformat von y bzw. d: Wahl der Geradendef	finitio	1 0 1 0	= di = In = IE = In = IE	urch z urch G iteger- EEE-Fo iteger- EEE-Fo der Za	erac Kurz orma Kurz orma	denf <u>z</u> it <u>z</u> it	akto	oren					

Ergebnis:	ERD OP1 R	Physikalischer Wert <i>y</i> (Integer-Kurz) Physikalischer Wert <i>y</i> (IEEE-Format) 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.

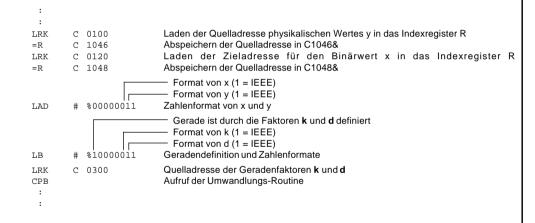


СРВ	Wandle	e Physikali	sch => Binär													
Ausführungszeit		6303	3400 - 8300	Mögliche	1	2	3	4 5	6	7	8	9 1	0	11 ′	12	13 14
in µs		6809	ca. 4000	Fehlermeldungen	О				0	0				(Э	
Funktion:	Umrechn	ung eines phy	sikalisch normierten Eingab	ewertes nach der Forme	ı x le	/ - d	<u>_</u> .									
	Die Umw d definier Da die Be Aufruf erf errechne	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& C1048& R	Zieladress	Zah													
	ERB	7 6 5 4	Zah	adendefinition: lenformat von x bzw. k: lenformat von y bzw. d: Wahl der Geradendef	0 = durch zwei Geradenstützpunkte 1 = durch Geradenfaktoren k und d 0 = Integer-Kurz 1 = IEEE-Format 0 = Integer-Kurz 1 = IEEE-Format finition und der Zahlenformate											

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.



CIM	Wandle	e Zoll => N	lillimeter															
Ausführu	ngszeit	6303	307 - 472	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µ	s	6809	269 - 368	Fehlermeldungen													0	0

Funktion: Eine auf Zoll bezogene Zahl im Integer-Lang Format wird in Millimeter umgerechnet (1 Zoll = 25.4 mm). Die Genauigkeits-

stufen in Bezug auf das Maßsystem können in 10er-Potenzschritten definiert werden. Um Umrechnungsungenauigkeiten zu vermeiden, sollten alle Maßzahlen innerhalb eines Programmes auf die selbe Genauigkeit bezogen werden.

Die Operanden OP1 und OP2 werden nicht verändert.

Übergabe: C1046& Quelladresse der Zoll-Zahl (Integer-Lang) R

Zieladresse der mm-Zahl (Integer-Lang) Exponent der Zoll-Zahl (0 bis 5)

ERA Exponent der mm-Zahl (0 bis 3) FRB

Ergebnis: ERD niederwertige zwei Byte der mm-Zahl N. Z

entsprechend dem Inhalt von ERD

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 im 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			Aufruf der Routine
:			

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

CMI	Wandle	e Millimete	r => Zoll															
Ausführu	ngszeit	6303	307 - 472	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µ	s	6809	267 - 369	Fehlermeldungen													O	0

Funktion: Eine auf Millimeter bezogene Zahl im Integer-Lang Format wird in Zoll umgerechnet (1 mm = 0.03937 Zoll). Die Genauig-

keitsstufen in Bezug auf das Maßsystem können in 10er-Potenzschritten definiert werden. Um Umrechnungsungenauigkeiten zu vermeiden, sollten alle Maßzahlen innerhalb eines Programmes auf die selbe Genauigkeit bezogen werden.

Die Operanden OP1 und OP2 werden nicht verändert.

Übergabe: C1046& Quelladresse der mm-Zahl (Integer-Lang) R Zieladresse der Zoll-Zahl (Integer-Lang)

ERA Exponent der mm-Zahl (0 bis 3)

Exponent der Zoll-Zahl (0 bis 5) FRB

Ergebnis: FRD niederwertige zwei Byte der Zoll-Zahl N. Z

entsprechend dem Inhalt von ERD

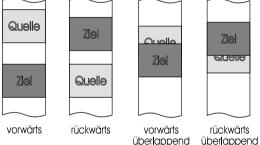
Beispiel:

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			Aufruf der Routine
:			

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 siehe Tabelle 2 3 5 6 8 9 10 11 12 13 14 Ausführungszeit 6303 Möaliche 4 6809 siehe Tabelle Fehlermeldungen in µs Funktion: 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: Beispiel: Die Tabelle TEXT soll in den Speicher ab C0100



kopiert werden.

Zieladresse LRK C 0100 =R C 1048 # 000 LAD

ERD enthält die Länge SPII TEXT der Tabelle und R die Startadresse der Tabelle

FCOP Kopieren der Tabelle

	Ausführungs	zeiten in µs
	6303	6809
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

Übergabe:

Ergebnis:

C1048& ERD

Zieladresse des Speicherbereiches Länge des Speicherbereiches in Byte Quelladresse des Speicherbereiches

FRD N, Z, C ungültig

verändert

FSMB	Speich	erbereich	mit 1 Byte-Wert init	ialisieren														
Ausführur	ngszeit	6303	48 + L*12	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µ:	s	6809	94 + L*7,5	Fehlermeldungen														
				•	_				_			_			_	_	_	$\boldsymbol{\neg}$

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: FRD verändert

N, Z, C ungültig

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

:

1 Byte-Wert in ERA laden LAD # 255

C 1052 1 Byte-Wert in C1052 an die Routine übergeben Länge des Speicherbereiches in Byte (300) LD # 00300 Anfangsadresse des Speicherbereiches LRK C 3000

FSMB

227

FSMW Speicherbereich mit 2 Byte-Wert initialisieren

Ausführungszeit	6303	40 + L*14	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	94 + L*8,5	Fehlermeldungen														

Funktion: Alle Speicherstellen des angegebenen Bereiches (Anfangsadresse, Länge des Bereiches in Worten) werden mit einem 2

Byte-Wert geladen.

Übergabe: C1052& 2 Byte-Wert

> ERD Länge des Speicherbereiches in Worten Anfangsadresse des Speicherbereiches R

Ergebnis: FRD verändert N, Z, C ungültig

Beispiel: Die Doppelspeicherstellen im Bereich C3000 bis C3299 sollen mit dem Wert 1000 (\$03E8) geladen werden:

2 Byte-Wert (Wort) in ERA laden LD # 1000

2 Byte-Wert (Wort) in C1052 an die Routine übergeben =D C 1052 Länge des Speicherbereiches in Worten (150 Worte => 300 Byte)

LD # 00150 Anfangsadresse des Speicherbereiches

C 3000 LRK

FSMW

FCLR	Speich	erbereich	löschen															
Ausführun	gszeit	6303	48 + L*12	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	5	6809	94 + L*7,5	Fehlermeldungen														

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

Ausführungszeit	6303	201 - 223	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs	6809	195 - 207	Fehlermeldungen												0		

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:

Springe, wenn	Sprungbefehl	
OP1 = OP2	SP0 (BEQ)	
OP1 ≠ OP2	SN0 (BNE)	
OP1 < OP2	SP< (BCS)	
OP1 ≤ OP2	J<= (BLS)	
OP1 > OP2	SP> (BHI)	
OP1 ≥ OP2	JC0 (BCC)	

MHIL	Begrer	Begrenzen auf Obergrenze; Wenn OP1 > OP2, dann OP2 ⇒ OP1																
Ausführu	ngszeit	6303	215 - 271	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
in µs		6809	205 - 237	Fehlermeldungen												0		

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.

gesetzt, wenn OP1 mit OP2 geladen wurde.

MLOL	Begrer	Begrenzen auf Untergrenze; Wenn OP1 < OP2, dann OP2 ⇒ OP1																
Ausführun	ngszeit	6303	215 - 271	Mögliche	1	2	3	4	5	6	7	8	9	10	11	12	13 ′	14
in µs		6809	205 - 237	Fehlermeldungen												0		

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

geladen

Übergabe: OP1 Wert, der begrenzt werden soll

OP2 Untergrenze

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:

ERA	7 0 PTR F/REG REG	ERB	7 0 EI D CODE			
PTR	Zeiger auf externen Operanden 00 = Indexregister R 01 = Indexregister Y¹) 10 = User-Stackpointer¹)	EI D	Befehlsart: 0 = Zwei interne Operanden 1 = Ein externer Operand Laden/Abspeichern von internem Operanden:			
F/REG	Wenn der Befehl zwei interne Operanden hat: F/REG = Register-Nr. des 2. Operanden (OP2²) Wenn der Befehl einen externen Operanden hat: F/REG = Formatcode (siehe Tabelle in Abschnitt	0 = Laden 1 = Abspeichern E Befehlscode (siehe "Befehlssatz des 68881")				
REG	"Zahlenformate") Register-Nr. des 1. Operanden (OP1 ²)	15	8 7 0 Startadresse des externen Operanden			
			Indexregister R, Y oder User-Stackpointer ¹⁾			

¹⁾ Indexregister Y und User-Stackpointer k\u00f6nnen ab Version 5.0 des B&R PROgrammierSYStemes verwendet werden.

Beispiel eines APU-Aufrufes:

LRK	C	2000	Zeiger auf externen Operanden
LAD	#	%00100011	Datenformat "Word Integer"
LB	#	%10xxxxxx	Ein externer Operand, xxxxxx = Befehl
MAT			

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

5.1.3. Zahlenformate

FORMAT	Bezeichnung	Länge	Formatcode 1)
7 0 8 Bits	BYTE INTEGER	8 Bit / 1 Byte	110
15 0 16 Bits	WORD INTEGER	16 Bit / 2 Byte	100
31 0 32 Bits	LONG INTEGER	32 Bit / 4 Byte	000
30 22 0 8 Bit Exponent 23 Bit Mantisse Mantissenvorzeichen	SINGLE REAL	32 Bit / 4 Byte	001
62 51 0 11 Bit Exponent 52 Bit Mantisse Mantissenvorzeichen	DOUBLE REAL	64 Bit / 8 Byte	101
62 80 63 0 15 Bit Exp. Null 64 Bit Mantisse Mantissenvorzeichen Dezimalpunkt (implizit)	EXTENDED REAL	96 Bit / 12 Byte	010
95 91 80 67 0 3 Digit Null 17 Digit Mantisse Exponent Dezimalpunkt (implizit) Null Exponentvorzeichen	PACKED DEZIMAL REAL	96 Bit / 12 Byte	011
Exponentvorzeichen Mantissenvorzeichen			

5.1.4. Befehlssatz des 68881 Arithmetikprozessors

Befehl	Bezeichnung	Beschreibung	Code (hex.)	Code (binär)
ABS	Absolutwert	OP1 := 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 := atanh (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
ETOXM1	e ^{x-1}	OP1 := e ^{OP1-1}	\$08	%001000
GETEXP		OP1 := Exponent (OP1)	\$1E	%011110
GETMAN		OP1 := Mantisse (OP1)	\$1F	%011111
INT	Integerfunktion	OP1 := int (OP1)	\$01	%000001
INTRZ	Integer mit Rundung	OP1 := 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 := In (OP1)	\$14	%010100

LOGNP1		OP1 := In (OP1 + 1)	\$06	%000110
MOD	Modulofunktion	OP1 := mod (OP1)	\$21	%100001
MOVE	Laden oder Auslesen		\$00	%000000
MOVECR	Laden mit Konstante	OP1 := const.	\$3B	%111011
MUL	Multiplikation	Multiplikation OP1 := OP1 * OP2		
NEG	Negation	OP1 := 0 - OP1	\$1A	%011010
SCALE		OP1 := OP1 * 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); Reg. n := cos(OP1)	\$3n	%110nnn
SINH	Sinus Hyperbolicus	OP1 := sinh (OP1)	\$02	%000010
SQRT	Quadratwurzel	OP1 := 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 µs.

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	Р	\$30	In(2)	\$36	10 ⁸	\$3C	10512
\$0B	log ₁₀ (2)	\$31	In(10)	\$37	10 ¹⁶	\$3D	101024
\$0C	е	\$32	100	\$38	1032	\$3E	102048
\$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	10⁴	\$3B	10256		

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 1

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:

SOB Zeichen ausgeben SIB Zeichen einlesen

SC Schnittstellenstatus anfordern

SF 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

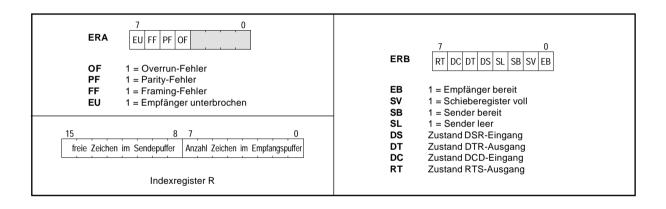
5.2.2. SIB - Zeichen einlesen

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			
	kein gültiges Z	eichen empfangen		
	Z = 1	Z = 1		
		Übertragungsfehler; ERA enthält einen Fehlercode:		
gültiges Zeichen empfangen und in ERA gespeichert	Empfangspuffer leer	7 0 EU FF PF OF		
		OF 1 = Overrun-Fehler PF 1 = Parität-Fehler FF 1 = Framing-Fehler EU 1 = Empfänger unterbrochen		

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

 Das Systemmodul einer älteren B&R PROgrammierSYStem-Version gegen ein Systemmodul der Version 3.1 oder h\u00f6her 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:

ERA	7 0 0 RS _{PP} RS	BAUD		ERB	7 EI SI SB P DI	OE DB	
RS _{PP} 1)			ers für Betrieb des PP60 setzer mit RS485-Verhal-	DB	Datenbits:	00 = 5 Bit 01 = 6 Bit	10 = 7 Bit 11 = 8 Bit
	ten.				Parität	1 = ungerade	0 = gerade
				DI 3)	DSR aktiv	1 = inaktiv	0 = aktiv
RS	1 = RS485 Mode 2)			P	Parität ein/aus	1 = aus	0 = ein
	0 = RS422/RS232 N	/lode		SB	Stop-Bits	1 = 2 Stop-Bits	0 = 1 Stop-Bit
Baud	Baudrate	Baud	Baudrate	SI EI	Sender-Interrupt Empfänger-Interrupt	1 = freigegeben 1 = freigegeben	0 = gesperrt 0 = gesperrt
0000		1000	1200 Baud				• .
0001	50 Baud	1001	1800 Baud	3) nur bei	Punkt zu Punkt Verbindu	ng (RS232 <=> RS23	verwendet.
0010	75 Baud	1010	2400 Baud			•	
0011	110 Baud	1011	3600 Baud				
0100	135 Baud	1100	4800 Baud				
0101	150 Baud	1101	7200 Baud				
0110	300 Baud	1110	9600 Baud	1	5	8 7	0
0111	600 Baud Systemmodul Version	1111	19200 Baud	l	änge Sendepuffer (1 bis	255) Länge Empfangs	puffer (1 bis 255)
	•	•					
	n RS485 Mode schaltet sich der Sender automatisch vom Bus, wenn 30				Inc	lexregister R	
ms kein	Zeichen gesendet wir	d.					

SF - Manuelle Bedienung der Handshake-Leitungen und Sende-/Empfangspuffer

7 0 1 0 0 0 0 0 0 0	LAD SF	# \$80	RTS-Leitung auf Low schalten. ¹⁾
7 0 1 0 0 0 0 0 1	LAD SF	# \$81	RTS-Leitung auf High Schalten. 1)
7 0 1 0 0 0 0 1 0	LAD SF	# \$82	Automatische DTR-Behandlung ein (Nach Power-ON ist diese immer ausgeschaltet!)
7 0 1 0 0 0 0 1 1	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.
7 0 1 0 0 0 0 1 0 0	LAD SF	# \$84	Sende- und Empfangspuffer löschen (Pointer zurücksetzen). Der Busy-Status wird gelöscht, falls er nicht mit dem Befehl \$83 gesperrt wurde.
7 0 1 0 0 0 1 0 1	LAD SF	# \$85	Empfangspuffer löschen (Pointer zurücksetzen). Der Busy-Status wird gelöscht, falls er nicht mit dem Befehl \$83 gesperrt wurde.
7 0 1 0 0 0 0 1 1 0	LAD SF	# \$86	Sendepuffer löschen (Pointer zurücksetzen).

¹⁾ 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).

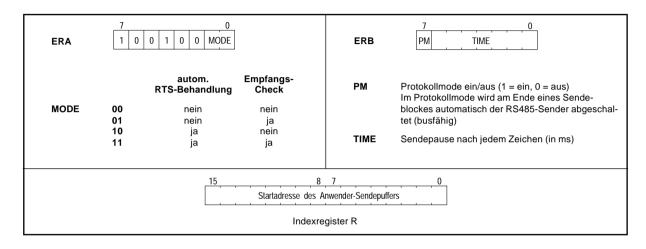
Das Blocksenden/-Empfangen kann wahlweise auch mit dem B&R Standardprotokoll erfolgen (MININET-Protokoll). Dieser Protokollmode wird z.B. benötigt, wenn mit einem BRRT28-Bedientableau oder einem BRMEC Massenspeicher kommuniziert wird. In diesem Fall müssen die Einträge im Sendepuffer (Frames) der Syntax des B&R Protokolls entsprechen:

Befehl vom Master	STX LEN NODE INDEX DATA CHK
Antwort vom Slave ohne Daten (kurze Antwort)	ACK
Antwort vom Slave mit Daten (lange Antwort)	STX LEN NODE INDEX DATA CHK
STX Startzeicher	en, das den Beginn eines Frames anzeigt (\$02)
LENLänge des g	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 (=> STX) als Datenbyte zu ermöglichen.	
	e (Checksum) über den Frame. Diese Prüfsumme wird automatisch vom stem errechnet und muß nicht vom Anwender eingetragen werden.
ACK Bestätigung	g, daß ein Frame fehlerfrei empfangen wurde (\$06)

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 Sendeblöcke im Protokollmode gesendet werden sollen
- 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:

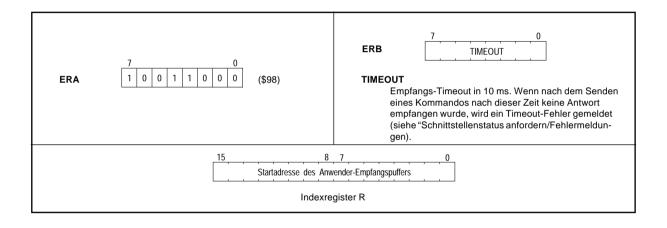
Die Anwenderschnittstelle wird für Blockmode ohne B&R-Protokoll initialisiert. Der User-Sendepuffer beginnt bei C 3000. Da der Empfänger nicht über Handshake-Leitungen verfügt, wird "autom. RTS-Behandlung" ausgeschaltet und nach jedem Zeichen eine 2 ms-Pause eingelegt. Baudrate, Stop-Bits und Parity 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

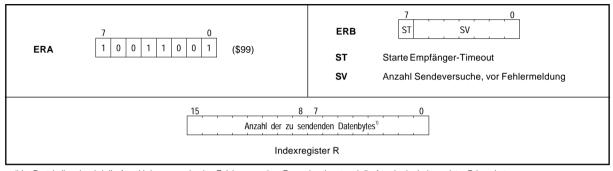
Zwischen dem SF-Aufruf und dem Starten der Timerinterruptroutine mit "SPU \$US2" dürfen keine Befehle stehen, die das Indexregister oder ERA verändern.

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.



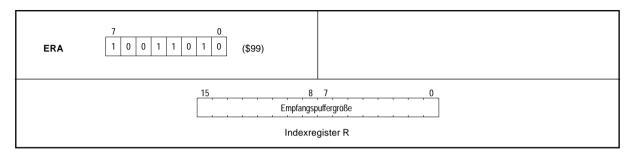
SF - Daten im Blockmode senden



 $^{^{\}circ}$ 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:

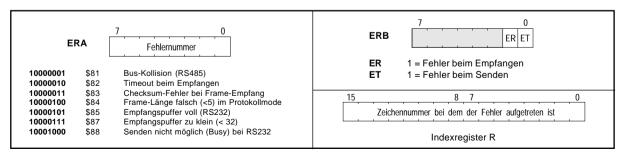
LD	#	00128	Empfangspuffergröße
DXR			
LAD	#	\$9A	Kommando "Daten empfangen"
SF			

SF - Schnittstellenstatus anfordern (\$9F)

Mit diesem Kommando werden Informationen über den Zustand der Schnittstelle angefordert (nur bei Blockmode/Protokollmode relevant).

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)

RS232 RS	
RXD	SR

Punkt zu Punkt RS232 <=> ECINT1 (SIB/SOB, Blockmode)

RS232		ECINT'
TXD RXD DTR GND	*	——————————————————————————————————————

RS485 (SIB/SOB, Blockmode)

RS485	RS485
DATA -	——— DATA
DATA —	<u>DATA</u>

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 1), DTR 2) und RTS 2) 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 Schnittstellenbau-

steins 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:

STX LEN NODE INDEX DATADATA

Die Checksumme wird vom Betriebssystem errechnet. Ebenso werden die Füllbytes (nach jedem \$02 im

Datenstrom) vom Betriebssystem eingefügt.

¹⁾ siehe "SF - Schnittstelle initialisieren"

²⁾ siehe "SF - Manuelle Bedienung der Handschake-Leitungen"

b) RS232 auf Buskoppler

Schnittstelleninitialisierung: RS Bit = 1 (RS485)

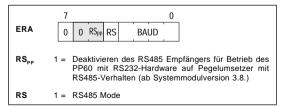
SIB/SOB:

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.



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 BEDIENUNG!!

Blockmode mit Protokoll:

Im Sendepuffer müssen folgende Einträge sein:

STX LEN NODE INDEX DATADATA

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 BEDIENUNG!!

Das RS_{PP} Bit muß 0 sein.

Blockmode mit Protokoll:

DSR, DTR, RTS: wie

wie Blockmode ohne Protokoll

Im Sendepuffer müssen folgende Einträge sein:

STX LEN NODE INDEX DATADATA

Die Checksumme wird vom Betriebssystem errechnet. Ebenso werden die Füllbytes (nach jedem \$02 im Datenstrom) vom Betriebssystem eingefügt.

Das RS_{PP} Bit muß 0 sein.

6. ANHANG

6.1. Alphabetische Übersicht der B&R Mnemonics

+	92	COA	175	LE!	46	SEI	173
++B	95	СОВ	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	SNOL	150
=D	54	END	162	MAB	60	SP<	148
=R	55	EXG	68	MAC	62	SP <l< td=""><td>150</td></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
В	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	JC0L	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	
ABX	104	
ADCA	93	
ADCB	95	
ADDA	92	
ADDB	94	
ADDD	96	
AIM	84	
ANDA	82	
ANDB	83	
ASL	132	
ASLA	133	
ASLB	134	
ASLD	135	
BCC	148	
BCCL	150	
BCS	148	
BCSL	150	
BEQ	148	
BEQL	150	
BHI	148	
BHIL	150	
BITA	115	
BITB	116	
BLS	148	
BLSL	150	
BMI	148	
BMIL	150	
BNE	148	
BNEL BPL	150 148	
BPLL	150	
CBA	108	
CLC	170	
OLO	170	

CLI	172
CLR	166
CLRA	167
CLRB	168
CMPA	109
CMPB	110
COM	174
COMA	175
COMB	176
CPX	111
CPX#	112
CPY	113
CPY#	114
DAA	177
DEC	125
DECA	126
DECB	127
DES	129
DEX	128
EIM	90
END	162
EORA	88
ORB	89
EXG	68
NC	120
NCA	121
NCB	122
NS	124
NX	123
JMP	160
ISR	156
.DAA	34
DAB	35
_DD	36

LDK	37
LDL	38
LDS	45
LDX	39
LDX#	40
LDXL	41
LDY	42
LDY#	43
LDYL	44
LEA!	46
LEAU	47
LEAX	48
LEAY	49
LSR	136
LSRA	137
LSRB	138
LSRD	139
MUL	105
NOP	161
OIM	87
ORAA	85
ORAB	86
PRS	164
PSH	70
PSHA	74
PSHB	76
PSHX	78
PUL	72
PULA	75
PULB	77
PULX	79
ROL	140
ROLA	141
ROLB	142

ROR	143
RORA	144
RORB	145
RST	165
RTS	158
SBA	103
SBCA	99
SBCB	101
SEC	171
SEI	173
SET	169
SK0	152
SK1	154
STAA	52
STAB	53
STAD	54
STS	57
STX	55
STY	56
SUBA	98
SUBB	100
SUBD	102
TAB	60
TAP	62
TBA	61
TFR	67
TIM	117
TPA	63
TSX	64
TXS	65
XGDX	66

6.3. Alphabetische Übersicht der Mathematik-Routinen

6.4. Alphabetische Übersicht der B&R Kurz Mnemonics

CAF CBCD CBIN CBP CBPP CBPQ CFA CFAO CFEA CIAO CIAO	199 210 211 218 214 216 200 202 204 206 208
CBPP	214
CBPQ	216
CFA	200
CFA0	202
CFEA	204
CIA	206
CIA0	208
CIM	222
CMI	224
CPB	220
CPBQ	217
FCLR	229
FCOP	226
FM2B	197
FM3B	197
FM4B	198
FSMB	227
FSMW	228
LAL1	187
LAL2	187
LAW1	188
LAW2	188
LF1	191

LF2	191
LIL1	189
LIL2	189
LIW1	190
LIW2	190
MADD	183
MCMP	230
MCOP	186
MDIV	184
MEXG	186
MHIL	231
MLOL	232
MMUL	184
MSGN	185
MSQR	185
MSUB	183
RFM1	196
RFM2	196
RFM3	196
SAL	192
SAW	192
SFM1	195
SFM2	195
SFM3	195
SFX	194
SIL	193
SIW	193

	99	
++	93	
Α	93	
С	166	
E	88	
1	52	
J<	148	
J>	148	
J0	148	
14	4.40	

A	175
В	176
	34
)	85
,	164
:	165
:L	140
lR	143
l	82
'	109

6.5. Stichwortverzeichhnis

Α		E		S	
Absolute Adressierung Adressierungsarten Absolute Adressierung	21 12, 17 21	Einfachgenaues Fließkommaformat	181	Schnittstellenbefehle SC - Schnittstellenstatus anfordern SF - Schnittstellenfunktionen	241 243 244
Direkte Adressierung Implizite Adressierung Indirekte Adressierung Indizierte Adressierung Negation Relative Adressierung Unmittelbare Adressierung Adreßvorwahlen Arithmetikbefehle Arithmetikprozessor	19 17 30 25 32 29 23 13 233 233	Half-Carry IEEE-Format Implizite Adressierung Indirekte Adressierung Indizierte Adressierung IMZ Mask	181 17 30 25 14	Blockmode initialisieren Daten im Blockmode empfangen Daten im Blockmode senden Manuelle Bedienung der Handshakeleitungen und Sende-/Empfangspuffer Schnittstelle initialisieren Schnittstellenstatus anfordern (\$9F) SIB - Zeichen einlesen SOB - Zeichen ausgeben Statusregister	247 253 252 246 245 254 242 241 14
В		M		Syntax	9
Befehlsabkürzung Befehlssatz des 68881 - Arithmetikprozessors	10 236	Mathematik-Routinen Mnemonic	179 10	U Unmittelbare Adressierung	23
C Carry-Flag CPU-Typ CPU-Typ A CPU-Typ B D Direkte Adressierung	14 9 15 15	Negation Negativ-Flag O Opcode Overflow-Flag R	32 14 12 14	Zahlenformate Absolut mit Vorzeichen Einfachgenaues Fließkommaformat IEEE-Format Integer Zahlenformate (APU) Zero-Flag	181 182 181 181 182 235
		Relative Adressierung	29		