Befehlssatz

MC68HC08

Microcontroller

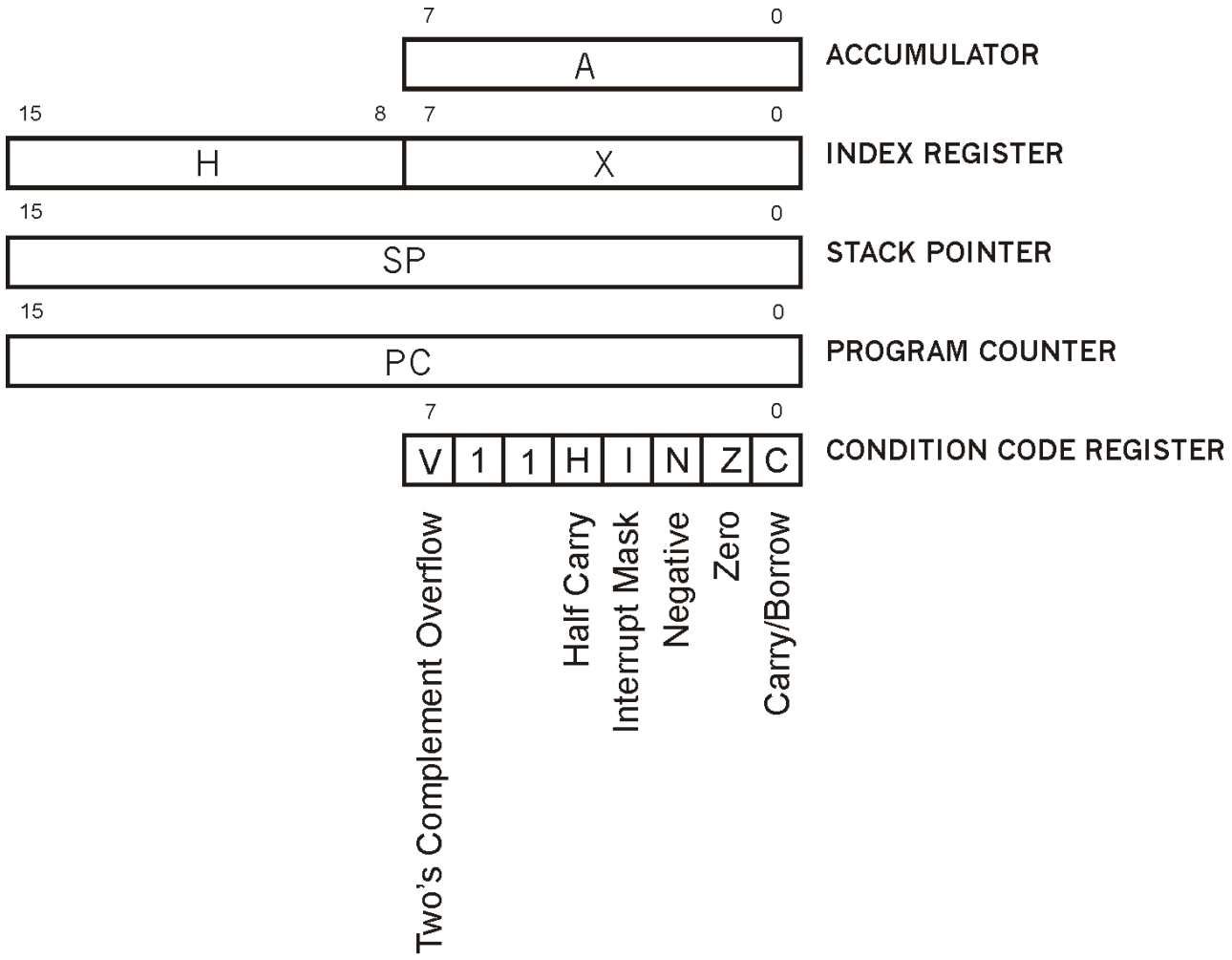
SYSTECH J.Schnyder GmbH
6540 Castaneda

2003

Erstellt:	22.12.2003	
Ergänzt:	23.12.2003	
	05.01.2004	
	06.01.2004	
	12.01.2004	
	13.01.2004	
	19.01.2004	
	20.01.2004	
	26.01.2004	V0.1 Rohfassung

Inhalt:

Programmierungs Modell



zum Inhaltsverzeichnis

Symbole und Abkürzungen

Condition Code Register

		Bit-Nummer	Aktion/Bemerkung
V	Überlauf Indikator	7	
I	Interrupt Masken Bit	3	1 ≙ Intrrupts sind gesperrt
Z	Zero Bit	1	1 ≙ Resultat ist 0
H	Halb-Carry Bit	4	1 ≙ Nibble Übertrag
N	Negativ Indikator	2	1 ≙ Zahl ist negativ (Bit 7 = 1)
C	Carry/Borrow Bit	0	1 ≙ Byte Übertrag

-	Bit nicht geändert
X	Bit undefiniert
0	Bit gelöscht (0)
1	Bit gesetzt (1)
	Bit gesetzt oder gelöscht

zum Inhaltsverzeichnis

Adressierungs Arten

INH	Inherent	(keine Operanden)
IMM	8-bit immediate (unmittelbar)	(ii)
DIR	8-bit direkt	(dd, dd rr)
EXT	16-bit erweitert	(hh ll)
IX	16-bit indiziert ohne Offset	(keine Operanden)
IX+	16-bit indiziert ohne Offset, vorangehendes Inkrement	
IX1	16-bit indiziert mit 8-bit Offset	(ff)
IX1+	16-bit indiziert mit 8-bit Offset, vorangehendes Inkrement	
IX2	16-bit indiziert mit 16-bit Offset	(ee ff)
REL	8-bit relativer Offset	(rr)
DD	direkt zu direkt	
IMD	immediate zu direkt	
IX+D	16-bit indiziert, vorangehendes Inkrement der Quelle nach direkt adressiert	
DIX+	direkt adressierte Quelle nach 16-bit indiziert, vorangehendes Inkrement	
SP1	Stack-Pointer mit 8-bit Offset	
SP2	Stack-Pointer mit 16-bit Offset	

M	Speicherstelle
rel	relativ Offset
opr	Operand

zum Inhaltsverzeichnis

A

ADC Add with Carry

Addiert den Inhalt des C-Bits zur Summe von A und dem entsprechenden Operanden. Das Resultat befindet sich in A. Diese Operation eignet sich für Additionen von Operanden, die grösser als 8-bit lang sind.

A (A) + (M) + (C)

VHINZC
–

ADC #opr	**IMM**	**A9**	**ii**	**2**
ADC opr	**DIR**	**B9**	**dd**	**3**
ADC opr	**EXT**	**C9**	**hh ll**	**4**
ADC opr,X	**IX2**	**D9**	**ee ff**	**4**
ADC opr,X	**IX1**	**E9**	**ff**	**3**
ADC ,X	**IX**	**F9**		**2**
ADC opr,SP	**SP1**	**9EE9**	**ff**	**4**
ADC opr,SP	**SP2**	**9ED9**	**ee ff**	**5**

Beispiel ADC zum Inhaltsverzeichnis

ADD Add without Carry

Addiert den Inhalt von A und den Inhalt des entsprechenden Operanden.
Das Resultat befindet sich in A.

A (A) + (M)

VHINZC
–

ADD #opr	**IMM**	**AB**	**ii**	**2**
ADD opr	**DIR**	**BB**	**dd**	**3**
ADD opr	**EXT**	**CB**	**hh ll**	**4**
ADD opr,X	**IX2**	**DB**	**ee ff**	**4**
ADD opr,X	**IX1**	**EB**	**ff**	**3**
ADD ,X	**IX**	**FB**		**2**
ADD opr,SP	**SP1**	**9EEB**	**ff**	**4**
ADD opr,SP	**SP2**	**9EDB**	**ee ff**	**5**

Beispiel ADD zum Inhaltsverzeichnis

AIS Add Immediate Value (Signed) to Stack Pointer

Addiere Immediate Wert (8-bit) zum Stack Pointer.
Der IMM-Wert ist eine 8-bit vorzeichenbehaftete Zahl welche auf 16 Bit erweitert und zum Stack Pointer addiert wird. Der AIS-Befehl kann verwendet werden, um Stack-Puffer zu erstellen oder zu löschen, welche temporäre Variable enthalten.

Diese Instruktion beeinflusst keine Bits des CCR, so dass Statusinformationen einer Subroutine oder C-Funktion gelesen oder übergeben werden können, ohne dass die Reservierung oder Freigabe von lokalen Variablen diese Statusinformationen überschreibt.

SP SP + (16 ≪ M)

```
VHINZC
------
```

AIS #opr	**IMM**	**A7**	**ii**	**2**

Beispiel AIS zum Inhaltsverzeichnis

AIX Add Immediate Value (Signed) to Index Register (H:X)

Addiere Immediate Wert (8-bit) zum Index Register (H:X), welches mit dem H- und dem X-Register gebildet wird. Der IMM-Wert ist eine 8-bit vorzeichenbehaftete Zahl welche auf 16 Bit erweitert und zum Index Register addiert wird.

Diese Instruktion beeinflusst keine CCR-Bits, so dass Index-Register-Berechnungen den Programm-Ablauf, welcher von den Status-Bits abhängt, nicht beeinflussen.

H:X (H:X) + (16 ≪ M)

```
VHINZC
-------
```

AIX #opr	**IMM**	**AF**	**ii**	**2**

Beispiel AIX zum Inhaltsverzeichnis

AND Logical AND

Logische UND Verknüpfung von A und dem angegebenen Operanden

A (A) & (M)

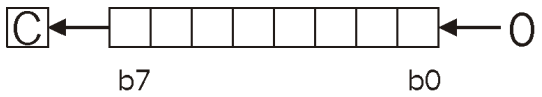
```
VHINZC
 --
```

AND #opr	**IMM**	**A4**	**ii**	**2**
AND opr	**DIR**	**B4**	**dd**	**3**
AND opr	**EXT**	**C4**	**hh ll**	**4**
AND opr,X	**IX2**	**D4**	**ee ff**	**4**
AND opr,X	**IX1**	**E4**	**ff**	**3**
AND ,X	**IX**	**F4**		**2**
AND opr,SP	**SP1**	**9EE4**	**ff**	**4**
AND opr,SP	**SP2**	**9ED4**	**ee ff**	**5**

Beispiel AND zum Inhaltsverzeichnis

ASL Arithmetic Shift Left (Same as LSL)

Schiebt alle Bits des angegebenen Registers um eine Stelle nach links, wobei Bit 0 auf 0 gesetzt wird und Bit 7 ins Carry-Bit des CCR geschoben wird. Diese Operation ist mathematisch eine Multiplikation mit Zwei. Das V-Bit zeig an, ob das Vorzeichen des Resultats geändert hat.



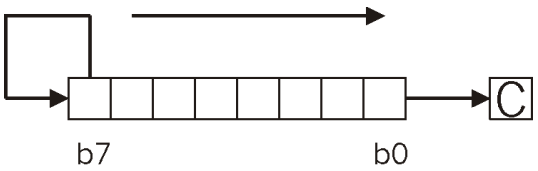
VHINZC
--

ASL opr	**DIR**	**38**	**dd**	**4**	
ASLA		**DIR**	**48**		**1**
ASLX		**EXT**	**58**		**1**
ASL opr,X	**IX1**	**68**	**ff**	**4**	
ASL ,X	**IX**	**78**		**2**	
ASL opr,SP	**SP1**	**9E68**	**ff**	**5**	

Beispile ASL zum Inhaltsverzeichnis

ASR Arithmetic Shift Right

Schiebt alle Bits des angegebenen Registers um eine Stelle nach rechts, wobei Bit 7 seinen Wert beibehält und Bit 0 ins Carry-Bit des CCR geschoben wird. Diese Operation dividiert eine zweierkomplement Zahl durch zwei ohne das Vorzeichen zu ändern. Das C-Bit kann zur Rundung verwendet werden.



VHINZC
--

ASR opr	**DIR**	**37**	**dd**	**4**	
ASRA		**DIR**	**47**		**1**
ASRX		**EXT**	**57**		**1**
ASR opr,X	**IX1**	**67**	**ff**	**4**	
ASR ,X	**IX**	**77**		**2**	
ASR opr,SP	**SP1**	**9E67**	**ff**	**5**	

Beispiel ASR zum Inhaltsverzeichnis

B

BCC Branch if Carry Bit Clear

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Carry-Bit gelöscht ist.

PC (PC) + $0002 + rel ? (C) = 0

```
VHINZC
------
```

BCC rel	**REL**	**24**	**rr**	**3**

Beispiel BCC zum Inhaltsverzeichnis

BCLR Clear Bit n in Memory

Löschen eines Bits im Speicher

Mn (0)

```
VHINZC
------
```

BCLR n,opr	**DIR(b0)**	**11**	**dd**	**4**
	DIR(b1)	**13**	**dd**	**4**
	DIR(b2)	**15**	**dd**	**4**
	DIR(b3)	**17**	**dd**	**4**
	DIR(b4)	**19**	**dd**	**4**
	DIR(b5)	**1B**	**dd**	**4**
	DIR(b6)	**1D**	**dd**	**4**
	DIR(b7)	**1F**	**dd**	**4**

Beispiel BCLR zum Inhaltsverzeichnis

BCS Branch if Carry Bit Set (Same as BLO)

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Carry-Bit gesetzt ist.

PC (PC) + $0002 + rel ? (C) = 1

```
VHINZC
------
```

BCS rel	**REL**	**25**	**rr**	**3**

Beispiel BCS zum Inhaltsverzeichnis

BEQ Branch if Equal

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Resultat der vorangegangenen Operation Null ist. (Das Zero-Bit ist gesetzt)

PC (PC) + $0002 + rel ? (Z) = 1

```
VHINZC
------
```

BEQ rel **REL** **27** **rr** **3**

Beispiel BEQ zum Inhaltsverzeichnis

BGE Branch if Greater Than or Equal To (Signed Operands)

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Resultat der vorangegangenen Operation grösser oder gleich ist.

Achtung; diese Funktion nur bei vorzeichenbehafteten Operationen anwenden!

PC (PC) + $0002 + rel ? (N ⊕ V) = 0

```
VHINZC
------
```

BGE rel **REL** **90** **rr** **3**

Beispiel BGE zum Inhaltsverzeichnis

BGT Branch if Greater Than (Signed Operands)

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Resultat der vorangegangenen Operation grösser ist.

Achtung; diese Funktion nur bei vorzeichenbehafteten Operationen anwenden!

PC (PC) + $0002 + rel ? (Z) | (N ⊕ V) = 0

```
VHINZC
------
```

BGT rel **REL** **92** **rr** **3**

Beispiel BGT zum Inhaltsverzeichnis

BHCC Branch if Half Carry Bit Clear

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Halb-Carry- Bit gelöscht ist.

PC (PC) + $0002 + rel ? (H) = 0

```
VHINZC
------
```

BHCC rel **REL** **28** **rr** **3**

Beispiel BHCC zum Inhaltsverzeichnis

BHCS Branch if Half Carry Bit Set

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Halb-Carry- Bit gelöscht ist.

PC (PC) + $0002 + rel ? (H) = 0

```
VHINZC
------
```

BHCS	**rel**	**REL**	**29**	**rr**	**3**

Beispiel BHCS zum Inhaltsverzeichnis

BHI Branch if Higher

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Resultat der vorangegangenen Operation grösser ist.

PC (PC) + $0002 + rel ? (C) | (Z) = 0

```
VHINZC
------
```

BHI	**rel**	**REL**	**22**	**rr**	**3**

Beispiel BHI zum Inhaltsverzeichnis

BHS Branch if Higher or Same (Same as BCC)

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Resultat der vorangegangenen Operation grösser ist.

PC (PC) + $0002 + rel ? (C) = 0

```
VHINZC
------
```

BHS	**rel**	**REL**	**24**	**rr**	**3**

Beispiel BHS zum Inhaltsverzeichnis

BIH Branch if /IRQ Pin High

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn der /IRQ-Eingang auf 1 ist.

PC (PC) + $0002 + rel ? /IRQ = 1

```
VHINZC
------
```

BIH	**rel**	**REL**	**2F**	**rr**	**3**

Beispiel BIH zum Inhaltsverzeichnis

BIL Branch if /IRQ Pin Low

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn der /IRQ-Eingang auf 1 ist.

PC (PC) + $0002 + rel ? /IRQ = 0

```
VHINZC
------
```

BIL	**rel**	**REL**	**2E**	**rr**	**3**

Beispiel BIL zum Inhaltsverzeichnis

BIT Bit Test

Bitweises Vergleichen des Akkumulators mit dem angegebenen Operatoren (Memory)

(A) & (M)

```
VHINZC
 --
```

BIT	**#opr**	**IMM**	**A5**	**ii**	**2**
BIT	**opr**	**DIR**	**B5**	**dd**	**3**
BIT	**opr**	**EXT**	**C5**	**hh ll**	**4**
BIT	**opr,X**	**IX2**	**D5**	**ee ff**	**4**
BIT	**opr,X**	**IX1**	**E5**	**ff**	**3**
BIT	**,X**	**IX**	**F5**		**2**
BIT	**opr,SP**	**SP1**	**9EE5**	**ff**	**4**
BIT	**opr,SP**	**SP2**	**9ED5**	**ee ff**	**5**

Beispiel BIT zum Inhaltsverzeichnis

BLE Branch if Less Than or Equal To (Signed Operands)

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Resultat der vorangegangenen Operation kleiner oder gleich ist.

Achtung; diese Funktion nur bei vorzeichenbehafteten Operationen anwenden!

PC (PC) + $0002 + rel ? (Z) | (N ⊕ V) = 1

```
VHINZC
------
```

BLE	**rel**	**REL**	**93**	**rr**	**3**

Beispiel BLE zum Inhaltsverzeichnis

BLO Branch if Lower (Same as BCS)

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Carry-Bit gesetzt ist. (Das Resultat der vorangegangenen Operation ist kleiner)

PC (PC) + $0002 + rel ? (C) = 1

```
VHINZC
------
```

BLO	**rel**	**REL**	**25**	**rr**	**3**

Beispiel BLO zum Inhaltsverzeichnis

BLS Branch if Lower or Same

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Resultat der vorangegangenen Operation kleiner oder gleich ist.

PC (PC) + $0002 + rel ? (C) | (Z) = 1

```
VHINZC
------
```

BLS rel	**REL**	**23**	**rr**	**3**

Beispiel BLS zum Inhaltsverzeichnis

BLT Branch if Less Than (Signed Operands)

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Resultat der vorangegangenen Operation kleiner ist.

Achtung; diese Funktion nur bei vorzeichenbehafteten Operationen anwenden!

PC (PC) + $0002 + rel ? (N ⊕ V) = 1

```
VHINZC
------
```

BLT rel	**REL**	**91**	**rr**	**3**

Beispiel BLT zum Inhaltsverzeichnis

BMC Branch if Interrupt Mask Clear

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das I-Bit im CCR gelöscht ist.

PC (PC) + $0002 + rel ? (I) = 0

```
VHINZC
------
```

BMC rel	**REL**	**2C**	**rr**	**3**

Beispiel BMC zum Inhaltsverzeichnis

BMI Branch if Minus

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Resultat der vorangegangenen Operation negativ ist.

PC (PC) + $0002 + rel ? (N) = 1

```
VHINZC
------
```

BMI rel	**REL**	**2B**	**rr**	**3**

Beispiel BMI zum Inhaltsverzeichnis

BMS Branch if Interrupt Mask Set

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das I-Bit im CCR gesetzt ist.

PC (PC) + $0002 + rel ? (I) = 1

```
VHINZC
------
```

BMS rel	**REL**	**2D**	**rr**	**3**

Beispiel BMS zum Inhaltsverzeichnis

BNE Branch if Not Equal

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Resultat der vorangegangenen Operation ungleich ist.

PC (PC) + $0002 + rel ? (Z) = 0

```
VHINZC
------
```

BNE rel	**REL**	**26**	**rr**	**3**

Beispiel BNE zum Inhaltsverzeichnis

BPL Branch if Plus

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn das Resultat der vorangegangenen Operation positiv ist.

PC (PC) + $0002 + rel ? (N) = 0

```
VHINZC
------
```

BPL rel	**REL**	**2A**	**rr**	**3**

Beispiel BPL zum Inhaltsverzeichnis

BRA Branch Always

Verzweige immer zur aktuellen Adresse+2 und angegebenem Offset.

PC (PC) + $0002 + rel

```
VHINZC
------
```

BRA rel	**REL**	**20**	**rr**	**3**

Beispiel BRA zum Inhaltsverzeichnis

BRCLR Branch if Bit n in Memory Clear

Verzweige zur aktuellen Adresse+2 und angegebenem Offset wenn Bit n der angegebenen Speicherstelle gelöscht ist.

Achtung! der Adressbereich muss zwischen 0 und 255 liegen.

Dieser Befehl ist ein Lesen-Ändern-Schreiben-Befehl (Read Modify Write) und kann bei Speicherstellen, die beim Lesen und Schreiben auf unterschiedliche Register zugreifen unerwartete Resultate liefern!

PC (PC) + $0003 + rel ? (Mn) = 0

```
VHINZC
-----
```

BRCLR	**n,opr,rel**	**DIR(b0)**	**01**	**dd rr**	**5**
		DIR(b1)	**03**	**dd rr**	**5**
		DIR(b2)	**05**	**dd rr**	**5**
		DIR(b3)	**07**	**dd rr**	**5**
		DIR(b4)	**09**	**dd rr**	**5**
		DIR(b5)	**0B**	**dd rr**	**5**
		DIR(b6)	**0D**	**dd rr**	**5**
		DIR(b7)	**0F**	**dd rr**	**5**

Beispiel BRCLR zum Inhaltsverzeichnis

BRN Branch Never

Verzweige nie.
(Dieser Befehl kann als Ersatz für andere Bxx-Befehle verwendet werden)

PC (PC) + $0002

```
VHINZC
------
```

BRN	**rel**	**REL**	**21**	**rr**	**3**

Beispiel BRN zum Inhaltsverzeichnis

BRSET Branch if Bit n in Memory Set

Verzweige zur aktuellen Adresse+3 und angegebenem Offset wenn Bit n der angegebenen Speicherstelle gesetzt ist.

Achtung! der Adressbereich muss zwischen 0 und 255 liegen.
Dieser Befehl ist ein Lesen-Ändern-Schreiben-Befehl (Read Modify Write) und kann bei Speicherstellen, die beim Lesen und Schreiben auf unterschiedliche Register zugreifen unerwartete Resultate liefern!

PC (PC) + $0003 + rel ? (Mn) = 1

```
VHINZC
-----
```

BRSET	**n,opr,rel**	**DIR(b0)**	**00**	**dd rr**	**5**
		DIR(b1)	**02**	**dd rr**	**5**
		DIR(b2)	**04**	**dd rr**	**5**
		DIR(b3)	**06**	**dd rr**	**5**
		DIR(b4)	**08**	**dd rr**	**5**
		DIR(b5)	**0A**	**dd rr**	**5**
		DIR(b6)	**0C**	**dd rr**	**5**
		DIR(b7)	**0E**	**dd rr**	**5**

Beispiel BRSET zum Inhaltsverzeichnis

BSET Set Bit n in Memory

Löschen eines Bits im Speicher

Mn (1)

```
VHINZC
------
```

BSET	**n,opr**	**DIR(b0)**	**10**	**dd**	**4**
		DIR(b1)	**12**	**dd**	**4**
		DIR(b2)	**14**	**dd**	**4**
		DIR(b3)	**16**	**dd**	**4**
		DIR(b4)	**18**	**dd**	**4**
		DIR(b5)	**1A**	**dd**	**4**
		DIR(b6)	**1C**	**dd**	**4**
		DIR(b7)	**1E**	**dd**	**4**

Beispiel BSET zum Inhaltsverzeichnis

BSR Branch to Subroutine

Verzweige zum Unterprogramm an der aktuellen Adresse+2 und angegebenem Offset. Der Programm-Counter wird im Stack gespeichert.

PC (PC) + $0002; push (PCL)
SP (SP) - $0001; push (PCH)
SP (SP) - $0001
PC (PC) + rel

```
VHINZC
------
```

BSR	**rel**	**REL**	**AD**	**rr**	**4**

Beispiel BSR zum Inhaltsverzeichnis

C

CBEQ Compare and Branch if Equal

Vergleiche Akkumulator mit der angegebenen Speicherstelle und verzweige bei gleichem Inhalt zur aktuellen Adresse+(Befehlslänge) und angegebenem Offset.

Achtung! Speicherstellen ausserhalb des Adressbereiches 0..255 können nur mit Hilfe des X-Registers adressiert werden!

PC	**(PC) + $0003 + rel ? (A) - (M) = $00**	CBEQ	opr,rel
PC	**(PC) + $0003 + rel ? (A) - (M) = $00**	CBEQA	#opr,rel
PC	**(PC) + $0003 + rel ? (X) - (M) = $00**	CBEQX	#opr,rel
PC	**(PC) + $0003 + rel ? (A) - (M) = $00**	CBEQ	opr,X+,rel
PC	**(PC) + $0002 + rel ? (A) - (M) = $00**	CBEQ	X+,rel
PC	**(PC) + $0004 + rel ? (A) - (M) = $00**	CBEQ	opr,SP,rel

```
VHINZC
------
```

CBEQ	**opr,rel**	**DIR**	**31**	**dd rr**	**5**
CBEQA	**#opr,rel**	**IMM**	**41**	**ii rr**	**4**
CBEQX	**#opr,rel**	**IMM**	**51**	**ii rr**	**4**
CBEQ	**opr,X+,rel**	**IX1+**	**61**	**rr**	**5**
CBEQ	**X+,rel**	**IX+**	**71**	**ff rr**	**4**
CBEQ	**opr,SP,rel**	**SP1**	**9E61**	**ff rr**	**6**

Beispiel CBEQ zum Inhaltsverzeichnis

CLC Clear Carry Bit

Löscht das Carry Bit.

C 0

```
VHINZC
-----0
```

CLC	**INH**	**98**	**1**

Beispiel CLC zum Inhaltsverzeichnis

CLI Clear Interrupt Mask Bit

Löscht das Interrupt Mask Bit.

I 0

```
VHINZC
--0---
```

CLI	**INH**	**9A**	**2**

Beispiel CLI zum Inhaltsverzeichnis

CLR Clear

Löscht (setzt auf $00) die angegebene Speicherstelle oder das angegebene Register.

M	**$00**	CLR	opr
A	**$00**	CLRA	
X	**$00**	CLRX	
H	**$00**	CLRH	
M	**$00**	CLR	opr,X
M	**$00**	CLR	,X
M	**$00**	CLR	opr,SP

```
VHINZC
0--01-
```

CLR	**opr**	**DIR**	**3F**	**dd**	**3**
CLRA		**INH**	**4F**		**1**
CLRX		**INH**	**5F**		**1**
CLRH		**INH**	**8C**		**1**
CLR	**opr,X**	**IX1**	**6F**	**ff**	**3**
CLR	**,X**	**IX**	**7F**		**2**
CLR	**opr,SP**	**SP1**	**9E61**	**ff**	**4**

Beispiel CLR zum Inhaltsverzeichnis

CMP Compare Accumulator with Memory

Vergleichen des Akkumulators mit dem angegebenen Operatoren (Memory)

(A) - (M)

```
VHINZC
 --
```

CMP #opr	**IMM**	**A1**	**ii**	**2**
CMP opr	**DIR**	**B1**	**dd**	**3**
CMP opr	**EXT**	**C1**	**hh ll**	**4**
CMP opr,X	**IX2**	**D1**	**ee ff**	**4**
CMP opr,X	**IX1**	**E1**	**ff**	**3**
CMP ,X	**IX**	**F1**		**2**
CMP opr,SP	**SP1**	**9EE1**	**ff**	**4**
CMP opr,SP	**SP2**	**9ED1**	**ee ff**	**5**

Beispiel CMP zum Inhaltsverzeichnis

COM Complement (One's Complement)

Bildet das Einer-Komplement des angegebenen Operators.

M	**/M = $FF - (M)**	COM	opr
A	**/A = $FF - (A)**	COMA	
X	**/X = $FF - (X)**	COMX	
M	**/M = $FF - (M)**	COM	opr,X
M	**/M = $FF - (M)**	COM	,X
M	**/M = $FF - (M)**	COM	opr,SP

```
VHINZC
0--  1
```

COM	**opr**	**DIR**	**33**	**dd**	**4**

COMA		**INH**	**43**		**1**
COMX		**INH**	**53**		**1**
COM	**opr,X**	**IX1**	**63**	**ff**	**4**
COM	**,X**	**IX**	**73**		**3**
COM	**opr,SP**	**SP1**	**9E63**	**ff**	**5**

Beispiel COM zum Inhaltsverzeichnis

CPHX Compare Index Register (H:X) with Memory

Vergleichen des Index-Registers mit der nachfolgenden 2Byte-Konstante oder mit dem Wert, der durch den nachfolgenden Operator adressiert wird.

(H:X) - (M:M + $0001)

```
VHINZC
 --
```

CPHX	**#opr**	**IMM**	**65**	**ii ii+1**	**3**
CPHX	**opr**	**DIR**	**75**	**dd**	**4**

Beispiel CPHX zum Inhaltsverzeichnis

CPX Compare X (Index Register Low) with Memory

Vergleichen des X-Registers (Low-Byte des Index-Registers) mit dem angegebenen Operatoren (Memory)

(X) - (M)

```
VHINZC
 --
```

CPX	**#opr**	**IMM**	**A3**	**ii**	**2**
CPX	**opr**	**DIR**	**B3**	**dd**	**3**
CPX	**opr**	**EXT**	**C3**	**hh ll**	**4**
CPX	**opr,X**	**IX2**	**D3**	**ee ff**	**4**
CPX	**opr,X**	**IX1**	**E3**	**ff**	**3**
CPX	**,X**	**IX**	**F3**		**2**
CPX	**opr,SP**	**SP1**	**9EE3**	**ff**	**4**
CPX	**opr,SP**	**SP2**	**9ED3**	**ee ff**	**5**

Beispiel CPX zum Inhaltsverzeichnis

D

DAA Decimal Adjust Accumulator

Justiert den Inhalt des Akkumulators und das Carry-Bit nach AND oder ADC Operationen mit BCD-codierten Werten, so dass sich eine korrekte BCD-Summe im Akkumulator befindet und das Carry-Bit entsprechend gesetzt ist. Der Zustand des Half-Carry-Bits beeinflusst diese Operation (siehe untenstehende Tabelle)

$(A)_{10}$

```
VHINZC
X--
```

DAA INH 72 2

DAA Funktions-Zusammenstellung

1	2	3	4	5	6
Anfangswert C-Bit	Wert A [7:4]	Anfangswert H-Bit	Wert A [3:0]	Korrektu-Faktor	Korrigiertes C-Bit
0	37994	0	37994	0	0
0	37993	0	A-f	6	0
0	37994	1	37988	6	0
0	A-F	0	37994	60	1
0	9-F	0	A-F	66	1
0	A-F	1	37988	66	1
1	37987	0	37994	60	1
1	37987	0	A-F	66	1
1	37988	1	37988	66	1

Beispiel DAA zum Inhaltsverzeichnis

DBNZ Decrement and Branch if Not Zero

Subtrahiert 1vom Inhalt des entsprechenden Registers und verzweigt anschliessend zur aktuellen Adresse+Befehlslänge plus angegebenem Offset wenn das Resultat nicht $00 ist. DBNZ beeinflusst nur das diederwertige Byte des Index-Registers (H:X); das höherwertige Byte wird nicht verändert.

A (A) - $0001 oder M (M) - $0001 oder X (X) - $0001

PC (PC) + $0003 + rel if (result) ≠ $00	DBNZ opr,rel	
	DBNZ opr,X,rel	oder
PC (PC) + $0002 + rel if (result) ≠ $00	DBNZA, DBNZX	
	DBNZ X,rel	oder
PC (PC) + $0004 + rel if (result) ≠ $00	DBNZ opr,SP,rel	

```
VHINZC
------
```

DBNZ	**opr,rel**	**DIR**	**3B**	**dd rr**	**5**
DBNZA	**rel**	**IMM**	**4B**	**rr**	**3**
DBNZX	**rel**	**IMM**	**5B**	**rr**	**3**
DBNZ	**opr,X+,rel**	**IX1+**	**6B**	**ff rr**	**5**
DBNZ	**X+,rel**	**IX+**	**7B**	**rr**	**4**
DBNZ	**opr,SP,rel**	**SP1**	**9E6B**	**ff rr**	**6**

Beispiel DBNZ zum Inhaltsverzeichnis

DEC Decrement

Subtrahiert 1 vom Inhalt des entsprechenden Registers. Die V-, N- und Z-Bits im CCR werden dem Resultat entsprechend gesetzt. Das C-bit im CCR-Register wird nicht geändert. Verzweigungs-Instruktionen wie BLS, BLO, BHS und BHI sind nach DEC nicht sinnvoll.

M	**(M) - $01**	DEC	opr
A	**(A) - $01**	DECA	
X	**(X) - $01**	DECX	
M	**(M) - $01**	DEC	opr,X
M	**(M) - $01**	DEC	,X
M	**(M) - $01**	DEC	opr,SP

```
VHINZC
  -
```

DEC opr	**DIR**	**3A**	**dd**	**4**
DECA	**INH**	**4A**		**1**
DECX	**INH**	**5A**		**1**
DEC opr,X	**IX1**	**6A**	**ff**	**4**
DEC ,X	**IX**	**7A**		**3**
DEC opr,SP	**SP1**	**9E6A**	**ff**	**5**

Beispiel DEC zum Inhaltsverzeichnis

DIV Divide

Dividiert eine 16-Bit Zahl ohne Vorzeichen in den Registern H und A durch einen 8-bit Divisor im X-Register. Der Quotient befindet sich nach der Operation im Akkumulator und der Rest im H-Register. Der Divisor im X-Register bleibt unverändert.

Ein Überlauf (Quotient > $FF) oder eine Division durch 0 setzt das C-Bit, wobei Quotient und Rest unbestimmt sind.

A (H:A)/(X)
H Reminder

```
VHINZC
----
```

DIV	**INH**	**52**	**7**

Beispiel DIV zum Inhaltsverzeichnis

E

EOR Exclusive OR Memory with Accumulator

Führt eine logische exklusiv-ODER Operation von Akkumulator und dem angegebenen Operanden aus. Jedes Bit des Akkumulators ist das Resultat einer bitweisen exklusiv-ODER Verknüpfung von A und dem angegebenen Operator.

A (A) ⊕ (M)

```
VHINZC
0--  -
```

EOR #opr	**IMM**	**A8**	**ii**	**2**
EOR opr	**DIR**	**B8**	**dd**	**3**
EOR opr	**EXT**	**C8**	**hh ll**	**4**
EOR opr,X	**IX2**	**D8**	**ee ff**	**4**
EOR opr,X	**IX1**	**E8**	**ff**	**3**
EOR ,X	**IX**	**F8**		**2**
EOR opr,SP	**SP1**	**9EE8**	**ff**	**4**
EOR opr,SP	**SP2**	**9ED8**	**ee ff**	**5**

Beispiel EOR zum Inhaltsverzeichnis

I

INC Increment

Addiert 1 zum Inhalt des entsprechenden Registers. Die V-, N- und Z-Bits im CCR werden dem Resultat entsprechend gesetzt. Das C-bit im CCR-Register wird nicht geändert. Verzweigungs-Instruktionen wie BLS, BLO, BHS und BHI sind nach DEC icht sinnvoll.

M	**(M) + $01**	INC	opr
A	**(A) + $01**	INCA	
X	**(X) + $01**	INCX	
M	**(M) + $01**	INC	opr,X
M	**(M) + $01**	INC	,X
M	**(M) + $01**	INC	opr,SP

VHINZC
\-

INC	**opr**	**DIR**	**3C**	**dd**	**4**
INCA		**INH**	**4C**		**1**
INCX		**INH**	**5C**		**1**
INC	**opr,X**	**IX1**	**6C**	**ff**	**4**
INC	**,X**	**IX**	**7C**		**3**
INC	**opr,SP**	**SP1**	**9E6C**	**ff**	**5**

Beispiel INC zum Inhaltsverzeichnis

J

JMP Jump

Ein Sprung zur Instruktion an der effektiven Adresse wird durchgeführt. Das Format der effektiven Adresse muss den Regeln der erweiterten, direkten oder indizierten Adressierung entsprechen.

PC effective Address

VHINZC

JMP opr	**DIR**	**BC**	**dd**	**2**
JMP opr	**EXT**	**CC**	**hh ll**	**3**
JMP opr,X	**IX2**	**DC**	**ee ff**	**4**
JMP opr,X	**IX1**	**EC**	**ff**	**3**
JMP ,X	**IX**	**FC**		**2**

Beispiel JMP zum Inhaltsverzeichnis

JSR Jump to Subroutine

Der Programm-Zähler wird um n erhöht, so dass er auf die nächste der JSR folgende Instruktion zeigt (n= 1,2,3 abhängig vom Adressierungs-Mode). Der Programm-Zähler wird anschliessend 8-bitweise auf dem Stack gespeichert und zwar das niederwertige Byte zuerst. Danach erfolgt ein Sprung zur angegebenen effektiven Adresse. Das Format der effektiven Adresse muss den Regeln der erweiterten, direkten oder indizierten Adressierung entsprechen.

PC (PC) +n (n= 1,2,3 abhängig vom Adress-Mode)
Push (PCL); SP (SP) - $0001
Push (PCH); SP (SP) -$0001
PC effective Adress

VHINZC

JMP opr	**DIR**	**BD**	**dd**	**4**
JMP opr	**EXT**	**CD**	**hh ll**	**5**
JMP opr,X	**IX2**	**DD**	**ee ff**	**6**
JMP opr,X	**IX1**	**ED**	**ff**	**5**
JMP ,X	**IX**	**FD**		**4**

Beispiel JSR zum Inhaltsverzeichnis

L

LDA Load Accumulator from Memory

Lädt den Inhalt der angegebenen Speicherstelle in den Akkumulator. Die N- und Z-Bits des CCR werden entsprechend dem Dateninhalt gesetzt oder gelöscht. Das V-Bit wird gelöscht. Dies ermöglicht bedingte Verzweigungen nach dem Laden ohne zusätzliche Tests durchzuführen.

A (M)

```
VHINZC
  –
```

LDA #opr	**IMM**	**A6**	**ii**	**2**
LDA opr	**DIR**	**B6**	**dd**	**3**
LDA opr	**EXT**	**C6**	**hh ll**	**4**
LDA opr,X	**IX2**	**D6**	**ee ff**	**4**
LDA opr,X	**IX1**	**E6**	**ff**	**3**
LDA ,X	**IX**	**F6**		**2**
LDA opr,SP	**SP1**	**9EE6**	**ff**	**4**
LDA opr,SP	**SP2**	**9ED6**	**ee ff**	**5**

Beispiel LDA zum Inhaltsverzeichnis

LDHX Load Index Reister from Memory

Lädt den Inhalt der angegebenen Speicherstelle in das Index-Register (H:X). Die N- und Z-Bits des CCR werden entsprechend dem Dateninhalt gesetzt oder gelöscht. Das V-Bit wird gelöscht. Dies ermöglicht bedingte Verzweigungen nach dem Laden ohne zusätzliche Tests durchzuführen.

H:X (M:M + $0001)

```
VHINZC
  –
```

LDHX	**#opr**	**IMM**	**45**	**ii kk**	**3**
LDHX	**opr**	**DIR**	**55**	**dd**	**4**

Beispiel LDHX zum Inhaltsverzeichnis

LDX Load X (Index Register Low) from Memory

Lädt den Inhalt der angegebenen Speicherstelle in das X-Register. Die N- und Z-Bits des CCR werden entsprechend dem Dateninhalt gesetzt oder gelöscht. Das V-Bit wird gelöscht. Dies ermöglicht bedingte Verzweigungen nach dem Laden ohne zusätzliche Tests durchzuführen.

X (M)

```
VHINZC
  –
```

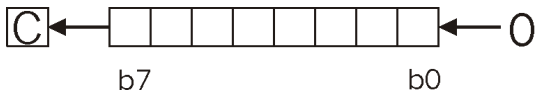
LDX #opr	**IMM**	**AE**	**ii**	**2**
LDX opr	**DIR**	**BE**	**dd**	**3**
LDX opr	**EXT**	**CE**	**hh ll**	**4**
LDX opr,X	**IX2**	**DE**	**ee ff**	**4**

LDX opr,X	**IX1**	**EE**	**ff**	**3**
LDX ,X	**IX**	**FE**		**2**
LDX opr,SP	**SP1**	**9EEE**	**ff**	**4**
LDX opr,SP	**SP2**	**9EDE**	**ee ff**	**5**

Beispiel LDX zum Inhaltsverzeichnis

LSL Logical Shift Left (Same As ASL)

Schiebt alle Bits des angegebenen Registers um eine Stelle nach links, wobei Bit 0 auf 0 gesetzt wird und Bit 7 ins Carry-Bit des CCR geschoben wird. Diese Operation ist mathematisch eine Multiplikation mit Zwei. Das V-Bit zeig an, ob das Vorzeichen des Resultats geändert hat.



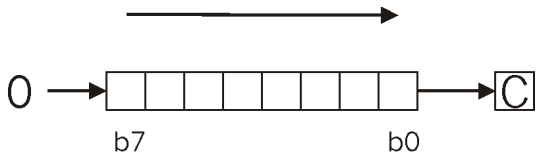
VHINZC
--

LSL opr	**DIR**	**38**	**dd**	**4**
LSLA	**INH**	**48**		**1**
LSLX	**INH**	**58**		**1**
LSL opr,X	**IX1**	**68**	**ff**	**4**
LSL ,X	**IX**	**78**		**3**
LSL opr,SP	**SP1**	**9E68**	**ff**	**5**

Beispiel LSL zum Inhaltsverzeichnis

LSR Logical Shift Right

Schiebt alle Bits des angegebenen Operators um eine Stelle nach Rechts, wobei Bit 7 mit 0 geladen wird und Bit 0 ins C-Bit des CCR geschoben wird.



VHINZC
--

LSR opr	**DIR**	**34**	**dd**	**4**
LSRA	**INH**	**44**		**1**
LSRX	**INH**	**54**		**1**
LSR opr,X	**IX1**	**64**	**ff**	**4**
LSR ,X	**IX**	**74**		**3**
LSR opr,SP	**SP1**	**9E64**	**ff**	**5**

Beispiel LSR zum Inhaltsverzeichnis

M

MOV Move

Kopiert ein Datenbyte von einer Quelladresse nach eine Zieladresse. Die Daten werden während des Kopierens bewertet und das CCR wird entsprechend bearbeitet. Die Quelldaten werden nicht verändert und der Akkumulator wird nicht beeinflusst.

Folgende vier Adressierungs-Mode sind möglich:

1. IMM/DIR kopiert ein unmittelbare folgendes Byte in eine direkt adressierte Speicherstelle.
2. DIR/DIR kopiert eine direkte adressierte Speicherstelle in eine andere direkt adressierte Speicherstelle.
3: IX+/DIR kopiert ein durch H:X adressiertes Byte in eine direkt adressierte Speicherstelle. H:X wird nach der Operation um 1 erhöht.
4. DIR/IX+ kopiert ein Byte einer direkt adressierten Speicherstelle in eine durch H:X adressierte Speicherstelle. H:X wird nach der Operation um 1 erhöht.

$(M)_{destination}$ $(M)_{source}$
H:X (H:X) + $0001 in IX+D and DIX+ Mode

VHINZC

MOV opr,opr	**DD**	**4E**	**dd dd**	**5**
MOV opr,X+	**DIX+**	**5E**	**dd**	**4**
MOV #opr,opr	**IMD**	**6E**	**ii dd**	**4**
MOV X+,opr	**IX+D**	**7E**	**dd**	**4**

Beispiel MOV zum Inhaltsverzeichnis

MUL Unsigned Multiply

Multipliziert den 8-bit Wert im X-Register mit dem 8-bit Wert im Akkumulator. Das Resultat, eine 16-bit Zahl ohne Vorzeichen, wird im X-Register und im Akkumulator abgelegt. Das X-Register enthält das höherwertige Byte und der Akkumulator das niederwertige Byte des Resultats.

X:A (X) x (A)

VHINZC

MUL	**INH**	**42**		**5**

Beispiel MUL zum Inhaltsverzeichnis

N

NEG Negate (Two's Complement)

Ersetzt den Inhalt on A, X oder M mit deren Zweierkomplement. Der Wert $80 wird nicht verändert! Diese Funktion ist equivalent zur Subtraktion von A, X oder M von $00.

M	**-(M)**	**= $00 - (M)**	NEG	opr
A	**-(A)**	**= $00 - (A)**	NEGA	
X	**-(X)**	**= $00 - (X)**	NEGX	
M	**-(M)**	**= $00 - (M)**	NEG	opr,X
M	**-(M)**	**= $00 - (M)**	NEG	,X
M	**-(M)**	**= $00 - (M)**	NEG	opr,SP

VHINZC
--

NEG	**opr**	**DIR**	**30**	**dd**	**4**
NEGA		**INH**	**40**		**1**
NEGX		**INH**	**50**		**1**
NEG	**opr,X**	**IX1**	**60**	**ff**	**4**
NEG	**,X**	**IX**	**70**		**3**
NEG	**opr,SP**	**SP1**	**9E60**	**ff**	**5**

Beispiel NEG zum Inhaltsverzeichnis

NOP No Operation

Diese 1byte Instruktion erledigt nichts weiter als die CPU einen Taktzyklus lang zu beschäftigen, während der Programm-Zähler um Eins erhöht wird. Es werden weder CCR-Flags noch sonstige Inhalte verändert.

None

VHINZC

NOP	**INH**	**9D**	**1**

Beispiel NOP zum Inhaltsverzeichnis

NSA Nibble Swap Accumulator

Tauscht das obere Nibble(4 Bits) mit dem unteren Nibble des Akkumulators. Die NSA Instruktion wird für die effiziente Speicherung bei BCD-Operationen verwendet.

A (A[3:0]:A[7:4])

VHINZC

NSA	**INH**	**62**	**3**

Beispiel NSA zum Inhaltsverzeichnis

O

ORA Inclusive OR Accumulator and Memory

Führt eine logische ODER-Verknüpfung zwischen Akkumulator und dem angegebenen Operanden durch. Das Resultat wird im Akkumulator abgelegt. Jedes Bit im Akkumulator ist das Resultat der ODER-Verknüpfung der jeweiligen Bits im Akkumulator und dem angegebenen Operanden vor der Operation.

A (A) | (M)

```
VHINZC
  -
```

ORA #opr	**IMM**	**AA**	**ii**	**2**
ORA opr	**DIR**	**BA**	**dd**	**3**
ORA opr	**EXT**	**CA**	**hh ll**	**4**
ORA opr,X	**IX2**	**DA**	**ee ff**	**4**
ORA opr,X	**IX1**	**EA**	**ff**	**3**
ORA ,X	**IX**	**FA**		**2**
ORA opr,SP	**SP1**	**9EEA**	**ff**	**4**
ORA opr,SP	**SP2**	**9ED6**	**ee ff**	**5**

Beispiel ORA zum Inhaltsverzeichnis

P

PSHA Push Accumulator onto Stack

Der Inhalt des Akkumulators wird auf die im Stack-Pointer enthaltene Adresse gespeichert. Anschliessend wird der Stack-Pointer um Eins erniedrigt. Der Inhalt des Akkumulators bleibt unverändert.

Push (A); SP (SP) - $0001

VHINZC

PSHA	**INH**	**87**	**2**

Beispiel PSHA zum Inhaltsverzeichnis

PSHH Push Index Register onto Stack

Der Inhalt des H-Registers wird auf die im Stack-Pointer enthaltene Adresse gespeichert. Anschliessend wird der Stack-Pointer um Eins erniedrigt. Der Inhalt des H-Registers bleibt unverändert.

Push (H); SP (SP) - $0001

VHINZC

PSHH	**INH**	**8B**	**2**

Beispiel PSHH zum Inhaltsverzeichnis

PSHX Push Index Register X onto Stack

Der Inhalt des X-Registers wird auf die im Stack-Pointer enthaltene Adresse gespeichert. Anschliessend wird der Stack-Pointer um Eins erniedrigt. Der Inhalt des X-Registers bleibt unverändert.

Push (X); SP (SP) - $0001

VHINZC

PSHX	**INH**	**89**	**2**

Beispiel PSHX zum Inhaltsverzeichnis

PULA Pull Accumulator from Stack

Der Stack-Pointer wird um Eins erhöht, um auf den letzten Operanden auf dem Stack zu verweisen. Anschliessend wird der Inhalt der Speicherstelle, auf die der Stack-Pointer zeigt in den Akkumulator geladen.

SP (SP) + $0001; Pull (A)

VHINZC

PULA	**INH**	**86**	**2**

Beispiel PULA zum Inhaltsverzeichnis

PULH Pull H from Stack

Der Stack-Pointer wird um Eins erhöht, um auf den letzten Operanden auf dem Stack zu verweisen. Anschliessend wird der Inhalt der Speicherstelle, auf die der Stack-Pointer zeigt in das H-Register geladen.

SP (SP) + $0001; Pull (H)

VHINZC

PULH	**INH**	**8A**	**2**

Beispiel PULH zum Inhaltsverzeichnis

PULX Pull X from Stack

Der Stack-Pointer wird um Eins erhöht, um auf den letzten Operanden auf dem Stack zu verweisen. Anschliessend wird der Inhalt der Speicherstelle, auf die der Stack-Pointer zeigt in das X-Register geladen.

SP (SP) + $0001; Pull (X)

VHINZC

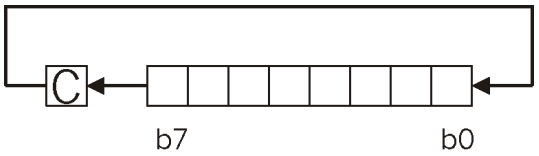
PULX	**INH**	**88**	**2**

Beispiel PULX zum Inhaltsverzeichnis

R

ROL Rotate Left through Carry

Schiebt alle Bits des gewählten Operanden um eine Stelle nach links. Bit 0 wird vom C-Bit des CCR übernommen. Das C-Bit wird vom höchstwertigen Bit des gewählten Operanden geladen. Diese Instruktion beinhaltet das Carry-Bit um die Erweiterung von Schiebe- und Rotations-Manipulationen auf mehrere Bytes zu ermöglichen. Zum Beispiel erlaubt die Sequenz ASL LOW, ROL MID, ROL HIGH das Schieben eines 24-bit Wertes um eine Stelle nach links, wobei LOW, MID und HIGH dem untern, mittleren und oberen Byte des 24-bit Wertes entsprechen.



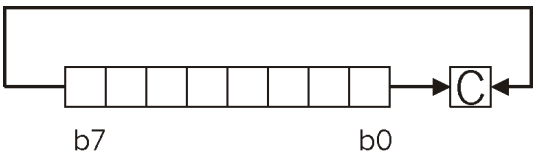
VHINZC
--

ROL opr	**DIR**	**39**	**dd**	**4**
ROLA	**INH**	**49**		**1**
ROLX	**INH**	**59**		**1**
ROL opr,X	**IX1**	**69**	**ff**	**4**
ROL ,X	**IX**	**79**		**3**
ROL opr,SP	**SP1**	**9E69**	**ff**	**5**

Beispiel ROL zum Inhaltsverzeichnis

ROR Rotate Right through Carry

Schiebt alle Bits des gewählten Operanden um eine Stelle nach rechts. Bit 7 wird vom C-Bit des CCR übernommen. Das C-Bit wird vom Bit 0 des gewählten Operanden geladen. Diese Instruktion beinhaltet das Carry-Bit um die Erweiterung von Schiebe- und Rotations-Manipulationen auf mehrere Bytes zu ermöglichen. Zum Beispiel erlaubt die Sequenz LSR LOW, ROR MID, ROR HIGH das Schieben eines 24-bit Wertes um eine Stelle nach rechts, wobei LOW, MID und HIGH dem untern, mittleren und oberen Byte des 24-bit Wertes entsprechen.



VHINZC
--

ROR opr	**DIR**	**36**	**dd**	**4**
RORA	**INH**	**46**		**1**
RORX	**INH**	**56**		**1**
ROR opr,X	**IX1**	**66**	**ff**	**4**
ROR ,X	**IX**	**76**		**3**
ROR opr,SP	**SP1**	**9E66**	**ff**	**5**

Beispiel ROR zum Inhaltsverzeichnis

RSP Reset Stack Pointer

Setzt den Stack-Pointer auf $FF.
Verwenden Sie diese, aus der HC05-Familie stammenden, Instruktion nich um den Stack-Pointer in HC08-Microcontrollern zu initialisieren. Da in vielen HC08-Microcontrollern der Speicherbereich $00FF übersteigt, ist es besser den Stack-Pointer kurz nach einem Reset auf die höchste Adresse zu setzen.

zum Beispiel:

```
LDHX    #RAM_END+1
TXS
```

SP $FF

VHINZC

RSP	**INH**	**9C**	**1**

Beispiel RSP zum Inhaltsverzeichnis

RTI Return from Interrupt

Das CCR-Register, der Akkumulator, das X-Register und der Programm-Zähler werden vom Stack zurückgeladen und somit in der Zustand vor dem Sprung zur Interrupt-Routine gesetzt. Das I-Bit im CCR wird gelöscht sofern das auf dem Stack gespeicherte entsprechende Bit gelöscht ist (Normalfall).

SP (SP) + $0001; Pull (CCR)
SP (SP) + $0001; Pull (A)
SP (SP) + $0001; Pull (X)
SP (SP) + $0001; Pull (PCH)
SP (SP) + $0001; Pull (PCL)

VHINZC

RTI	**INH**	**80**	**7**

Beispiel RTI zum Inhaltsverzeichnis

RTS Return from Subroutine

Der Stack wird um Eins erhöht und die Speicherstelle, welche der Stack-Pointer adressiert, wird in das höherwertige Byte des Programm-Zählers geladen. Danach wird der Stack-Pointer nochmals um Eins erhöht und die Speicherstelle, welche der Stack-Pointer adressiert, wird in das niederwertige Byte des Programm-Zählers geladen. Das Programm wird daraufhin an der vom Stack geladenen Adresse weiter ausgeführt.

SP (SP) + $0001; Pull (PCH)
SP (SP) + $0001; Pull (PCL)

VHINZC

RTS	**INH**	**81**	**4**

S

SBC Subtract with Carry

Subtrahiert den Inhalt des angegebenen Operarors und den Inhalt des C-Bits vom Akkumulator. Das Resultat wird im Akkumulator gespeichert. Diese Instruktion ist hilfreich für Subtraktionen mit Werten, die grösser als 8-bit enthalten.

A (A) - (M) - (C)

VHINZC
–

SBC #opr	**IMM**	**A2**	**ii**	**2**
SBC opr	**DIR**	**B2**	**dd**	**3**
SBC opr	**EXT**	**C2**	**hh ll**	**4**
SBC opr,X	**IX2**	**D2**	**ee ff**	**4**
SBC opr,X	**IX1**	**E2**	**ff**	**3**
SBC ,X	**IX**	**F2**		**2**
SBC opr,SP	**SP1**	**9EE2**	**ff**	**4**
SBC opr,SP	**SP2**	**9ED2**	**ee ff**	**5**

Beispiel SBC zum Inhaltsverzeichnis

SEC Set Carry Bit

Setzt das C-Bit im CCR. Diese Instruktion kann benützt werden um das C-Bit vor der Ausführung eines Verzweigungs- oder Rotation-Befehls in den gewünschten Zustand zu setzen.

C bit 1

VHINZC

SEC	**INH**	**9C**		**1**

Beispiel SEC zum Inhaltsverzeichnis

SEI Set Interrupt Mask Bit

Setzt das I-Bit im CCR. Der Microcontroller verarbeitet keine Interrupts, wenn das I-Bit gesetzt (maskiert) ist. Das I-Bit wird am Ende des SEI-Zyklus gesetzt dies kann zu spät sein un einen Interrupt zu stoppen, wenn dieser während der Ausführung der SEI-Instruktion erfolgt.

I bit 1

VHINZC

SEI	**INH**	**9B**		**1**

Beispiel SEI zum Inhaltsverzeichnis

STA Store Accumulator in Memory

Speichert den Inhalt des Akkumulators in die angegebene Speicherstelle. Die N- und Z-Bits des CCR werden entsprechend dem Dateninhalt gesetzt oder gelöscht. Das V-Bit wird gelöscht. Dies ermöglicht bedingte Verzweigungen nach dem Laden ohne zusätzliche Tests durchzuführen.

M (A)

VHINZC

STA opr	**DIR**	**B7**	**dd**	**3**
STA opr	**EXT**	**C7**	**hh ll**	**4**
STA opr,X	**IX2**	**D7**	**ee ff**	**4**
STA opr,X	**IX1**	**E7**	**ff**	**3**
STA ,X	**IX**	**F7**		**2**
STA opr,SP	**SP1**	**9EE7**	**ff**	**4**
STA opr,SP	**SP2**	**9ED7**	**ee ff**	**5**

Beispiel STA zum Inhaltsverzeichnis

STHX Store Index Register (H:X)

Speichert den Inhalt des H-Registers in die durch den Operatoren adressierte Speicherstelle und den Inhalt des X-Registers in die durch den Operatoren adressierte Speicherstelle+1. Die N- und Z-Bits des CCR werden entsprechend dem Dateninhalt gesetzt oder gelöscht. Das V-Bit wird gelöscht. Dies ermöglicht bedingte Verzweigungen nach dem Laden ohne zusätzliche Tests durchzuführen.

(M:M + $0001) (H:X)

VHINZC

STHX opr	**DIR**	**35**	**dd**	**4**

Beispiel STHX zum Inhaltsverzeichnis

STOP Enable /IRQ Pin; Stop Oscillator

Vermindert die Stromaufnahme durch das eliminieren aller dynamischen Vorgänge. Der externe Interrupt-Pin wird freigegeben (IRQ) und das I-Bit wird gelöscht um den externen Interrupt freizugeben.
Wird nun am /Reset- oder /IRQ-Pin logisch 0 erkannt, wird der Oszillator gestartet. Nach 4095 Prozessor-Zyklen wird der Reset-Vektor oder der IRQ-Vektor gelesen und die entsprechend Interrupt-Routine ausgeführt.

I bit 0; stop oscillator

VHINZC

STOP	**INH**	**8E**		**1**

Beispiel STOP zum Inhaltsverzeichnis

STX Store X (Index Register Low) in Memory

Speichert den Inhalt des X-Registers in der angegebenen Speicherstelle. Der Inhalt des X-Registers bleibt unverändert. Die N- und Z-Bits des CCR werden entsprechend dem

Dateninhalt gesetzt oder gelöscht. Das V-Bit wird gelöscht. Dies ermöglicht bedingte Verzweigungen nach dem Laden ohne zusätzliche Tests durchzuführen.

M (X)

VHINZC

STX opr	**DIR**	**BF**	**dd**	**3**
STX opr	**EXT**	**CF**	**hh ll**	**4**
STX opr,X	**IX2**	**DF**	**ee ff**	**4**
STX opr,X	**IX1**	**EF**	**ff**	**3**
STX ,X	**IX**	**FF**		**2**
STX opr,SP	**SP1**	**9EEF**	**ff**	**4**
STX opr,SP	**SP2**	**9EDF**	**ee ff**	**5**

Beispiel STX zum Inhaltsverzeichnis

SUB Subtract

Subtrahiert den Inhalt des angegebenen Operarors vom Akkumulator. Das Resultat wird im Akkumulator gespeichert.

A (A) - (M)

VHINZC
–

SUB #opr	**IMM**	**A0**	**ii**	**2**
SUB opr	**DIR**	**B0**	**dd**	**3**
SUB opr	**EXT**	**C0**	**hh ll**	**4**
SUB opr,X	**IX2**	**D0**	**ee ff**	**4**
SUB opr,X	**IX1**	**E0**	**ff**	**3**
SUB ,X	**IX**	**F0**		**2**
SUB opr,SP	**SP1**	**9EE0**	**ff**	**4**
SUB opr,SP	**SP2**	**9ED0**	**ee ff**	**5**

Beispiel SUB zum Inhaltsverzeichnis

SWI Software Interrupt

Der Programm-Zähler wird um Eins erhöht. Der Programm-Zähler, X-Register und Akkumulator werden auf den Stack geschrieben, ebenso das CCR. Das I-Bit wird gesetzt und der SWI-Vektor in den Programm-Zähler geladen. Diese Instruktion ist nicht mit dem I-Bit maskierbar!

PC (PC) + $0001
Push (PCL) ;SP (SP) - $0001
Push (PCH) ;SP (SP) - $0001
Push (X) ;SP (SP) - $0001
Push (A) ;SP (SP) - $0001
Push (CCR) ;SP (SP) - $0001
I bit 1
PCH ($FFFC)
PCL ($FFFD)

VHINZC

SWI	**INH**	**83**	**9**
Beispiel SWI			zum Inhaltsverzeichnis

T

TAP Transfer Accumulator to CCR

Transferiert den Inhalt des Akkumulators ins CCR. Der Inhalt des Akkumulators wird nicht verändert. Wenn diese Instruktion ein Wechsel der I-Bits von 0 auf 1 verursacht, so wird ein zusätzlicher Bus-Zyklus eingeleitet. Dies verhindert, dass die auf die TAP-Instruktion folgende Instruktion immer ausgeführt wird, selbst wenn eine Interrupt-Anforderung ausgelöst wurde.

CCR (A)

VHINZC

TAP	**INH**	**84**	**2**

Beispiel TAP zum Inhaltsverzeichnis

TAX Transfer Accumulator to X (Indes Register Low)

Lädt das X-Register mit dem Inhalt der Akkumulators. Der Inhalt des Akkumulators bleibt unverändert.

X (A)

VHINZC

TAX	**INH**	**97**	**1**

Beispiel TAX zum Inhaltsverzeichnis

TPA Transfer CCR to Accumulator

Verschiebt den Inhalt des CCR in den Akkumulator.

A (CCR)

VHINZC

TPA	**INH**	**85**	**1**

Beispiel TPA zum Inhaltsverzeichnis

TST Test for Negative or Zero

Setzt das N- und das Z-Bit des CCR entsprechend dem Inhalts des angegebenen Operators. Der Inhalt des angegebenen Operators wird nicht verändert.

(A)	**- $00**	TSTA		
(X)	**- $00**	TSTX		
(M)	**- $00**	TST	opr	oder
		TST	opr,X	oder
		TST,X		oder
		TST	opr,SP	

VHINZC

TST opr	**DIR**	**3D**	**dd**	**3**
TSTA	**INH**	**4D**		**1**
TSTX	**INH**	**5D**		**1**
TST opr,X	**IX1**	**6D**	**ff**	**3**
TST ,X	**IX**	**7D**		**2**
TST opr,SP	**SP1**	**9E6D**	**ff**	**4**

Beispiel TST zum Inhaltsverzeichnis

TSX Transfer Stack Pointer to Index Register

Lädt das Index-Register (H:X) mit Eins plus den Inhalt des Stack-Pointers (SP). Der Inhalt des Stack-Pointers bleibt unverändert. Nach einer TSX-Instruktion zeigt das Index-Register auf den letzten auf dem Stack gespeicherten Wert.

H:X (SP) + $0001

VHINZC

TSX	**INH**	**95**		**2**

Beispiel TSX zum Inhaltsverzeichnis

TXA Transfer X (Index Register Low) to Accumulator

Lädt den Akkumulator mit dem Inhalt des X-Registers. Der Inhalt des X-Registers bleibt unverändert.

A (X)

VHINZC

TXA	**INH**	**9F**		**1**

Beispiel TXA zum Inhaltsverzeichnis

TXS Transfer Index Register to Stack Pointer

Lädt den Stack-Pointer mit dem Inhalt des Index-Registers (H:X) minus Eins. Der Inhalt des Index-Registers bleibt unverändert.

SP (H:X) - $0001

VHINZC

TXS	**INH**	**94**		**2**

Beispiel TXS zum Inhaltsverzeichnis

W

WAIT Enable Interrupts; Stop Processor

Vermindert die Stromaufnahme durch das eliminieren einiger dynamischen Vorgänge. Der Timer, der Timer-Vorteiler und die integrierte- Peripherie-Bausteine (falls freigegeben) funktionieren weiterhin, da sie als potentielle Interrupt-Quelle benötigt werden. Das I-Bit wird gelöscht um die Interrupts freizugeben und der Prozessor-Takt wird gesperrt.
Wird nun am /Reset- oder /IRQ-PIn logisch 0 oder ein freigegebener Interrupt erkannt, wird der Prozessor-Takt freigegeben. Der Reset-, IRQ- oder Interrupt-Vektor wird gelesen und die entsprechend Interrupt-Routine ausgeführt.

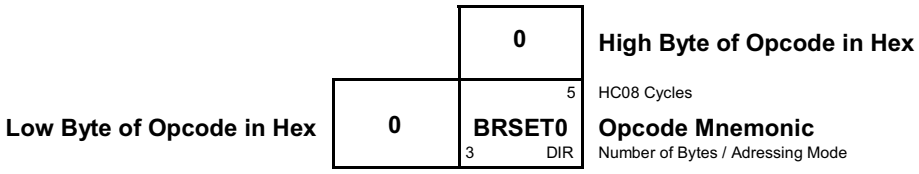
I bit 0; inhibit CPU clocking until interrupted

VHINZC

WAIT	**INH**	**8F**	**1**

Beispiel WAIT zum Inhaltsverzeichnis

OPCODE MAP



	Bit Manipulation		Branch	Read-Modify-Write						
	DIR	DIR	REL	DIR	INH	INH	IX1	SP1	IX	
HIGH / LOW	0	1	2	3	4	5	6	9000000	7	
0	5 BRSET0 3 DIR	4 BSET0 2 DIR	3 BRA 2 REL	4 NEG 2 DIR	1 NEGA 1 INH	1 NEGX 1 INH	4 NEG 2 IX1	5 NEG 3 SP1	3 NEG 1 IX	
1	5 BRCLR0 3 DIR	4 BCLR0 2 DIR	3 BRN 2 REL	5 CBEQ 3 DIR	4 CBEQA 3 IMM	4 CBEQX 3 IMM	5 CBEQ 3 IX1+	6 CBEQ 4 SP1	4 CBEQ 2 IX+	
2	5 BRSET1 3 DIR	4 BSET1 2 DIR	3 BHI 2 REL		5 MUL 1 INH	7 DIV 1 INH	3 NSA 1 INH		3 DAA 1 IX	
3	5 BRCLR1 3 DIR	4 BCLR1 2 DIR	3 BLS 2 REL	4 COM 2 DIR	1 COMA 1 INH	1 COMX 1 INH	4 COM 2 IX1	5 COM 3 SP1	3 COM 1 IX	
4	5 BRSET2 3 DIR	4 BSET2 2 DIR	3 BCC 2 REL	4 LSR 2	1 LSRA 1 INH	1 LSRX 1 INH	4 LSR 1 INH	5 LSR 2 IX1	3 LSR 1 IX	
5	5 BRCLR2 3 DIR	4 BCLR2 2 DIR	3 BCS 2 REL	4 COM 2 DIR	1 COMA 2 IMN	1 COMX 2 DIR	4 COM 3 IMM		3 COM 2 DIR	
6	5 BRSET3 3 DIR	4 BSET3 2 DIR	3 BNE 2 REL	4 ROR 2 DIR	1 RORA 1 INH	1 LSRX 1 INH	4 LSR 1 INH	5 LSR 2 IX1	3 LSR 1 IX	
7	5 BRCLR3 3 DIR	4 BCLR3 2 DIR	3 BEQ 2 REL	4 ASR 2	1 ASRA 1 INH	1 ASRX 1 INH	4 ASR 2 IX1	5 ASR 3	3 ASR 1 IX	
8	5 BRSET4 3 DIR	4 BSET4 2 DIR	3 BHCC 2 REL	4 LSL 2 DIR	1 LSLA 1 INH	1 LSLX 1 INH	4 LSL 2 IX1	5 LSL 3 SP1	3 LSL 1 IX	
9	5 BRCLR4 3 DIR	4 BCLR4 2 DIR	3 BHCS 2 REL	4 ROL 2 DIR	1 ROLA 1 INH	1 ROLX 1 INH	4 ROL 2 IX1	5 ROL 3	3 ROL 1 IX	
A	5 BRSET5 3 DIR	4 BSET5 2 DIR	3 BPL 2 REL	4 DEC 2 DIR	1 DECA 1 INH	1 DECX 1 INH	4 DEC 2 IX1	5 DEC 3 SP1	3 DEC 1 IX	
B	5 BRCLR5 3 DIR	4 BCLR5 2 DIR	3 BMI 2 REL	5 DBNZ 3 DIR	3 DBNZA 2 INH	3 DBNZX 2 INH	5 DBNZ 3 IX1	6 DBNZ 4 SP1	4 DBNZ 2 IX	
C	5 BRSET6 3 DIR	4 BSET6 2 DIR	3 BMC 2 REL	4 INC 2 DIR	1 INCA 1 INH	1 INCX 1 INH	4 INC 2 IX1	5 INC 3 SP1	3 INC 1 IX	
D	5 BRCLR6 3 DIR	4 BCLR6 2 DIR	3 BMC 2 REL	3 TST 2 DIR	1 TSTA 1 INH	1 TSTX 1 INH	3 TST 2 IX1	4 TST 3 SP1	2 TST 1 IX	
E	5 BRSET7 3 DIR	4 BSET7 2 DIR	3 BIL 2 REL		4 MOV 3 DD	4 MOV 2 DIX+	4 MOV 3 IMD		4 MOV 2 IX+D	
F	5 BRCLR7 3 DIR	4 BCLR7 2 DIR	3 BIH 2 REL	3 CLR 2	1 CLRA 1 INH	1 CLRX 1 INH	3 CLR 2 IX1	4 CLR 3 SP1	2 CLR 1 IX	

zum Inhaltsverzeichnis

	Control		Register/Memory							
	INH	INH	IMM	DIR	EXT	IX2	SP2	IX1	SP1	IX
HIGH / LOW	8	9	A	B	C	D	9ED	E	9EE	F
0	7 **RTI** 1 INH	3 **BGE** 2 REL	2 **SUB** 2 IMM	3 **SUB** 2 DIR	4 **SUB** 3 EXT	4 **SUB** 3 IX2	5 **SUB** 4 SP2	3 **SUB** 2 IX1	4 **SUB** 3 SP1	2 **SUB** 1 IX
1	4 **RTS** 1 INH	3 **BLT** 2 REL	2 **CMP** 2 IMM	3 **CMP** 2 DIR	4 **CMP** 3 EXT	4 **CMP** 3 IX2	5 **CMP** 4 SP2	3 **CMP** 2 IX1	4 **CMP** 3 SP1	2 **CMP** 1 IX
2		3 **BGT** 2 REL	2 **SBC** 2 IMM	3 **SBC** 2 DIR	4 **SBC** 3 EXT	4 **SBC** 3 IX2	5 **SBC** 4 SP2	3 **SBC** 2 IX1	4 **SBC** 3 SP1	2 **SBC** 1 IX
3	9 **SWI** 1 INH	3 **BLE** 2 REL	2 **CPX** 2 IMM	3 **CPX** 2 DIR	4 **CPX** 3 EXT	4 **CPX** 3 IX2	5 **CPX** 4 SP2	3 **CPX** 2 IX1	4 **CPX** 3 SP1	2 **CPX** 1 IX
4	2 **TAP** 1 INH	2 **TXS** 1 INH	2 **AND** 2 IMM	3 **AND** 2 DIR	4 **AND** 3 EXT	4 **AND** 3 IX2	5 **AND** 4 SP2	3 **AND** 2 IX1	4 **AND** 3 SP1	2 **AND** 1 IX
5	1 **TPA** 1 INH	2 **TSX** 1 INH	2 **BIT** 2 IMM	3 **BIT** 2 DIR	4 **BIT** 3 EXT	4 **BIT** 3 IX2	5 **BIT** 4 SP2	3 **BIT** 2 IX1	4 **BIT** 3 SP1	2 **BIT** 1 IX
6	2 **PULA** 1 INH		2 **LDA** 2 IMM	3 **LDA** 2 DIR	4 **LDA** 3 EXT	4 **LDA** 3 IX2	5 **LDA** 4 SP2	3 **LDA** 2 IX1	4 **LDA** 3 SP1	2 **LDA** 1 IX
7	2 **PSHA** 1 INH	1 **TAX** 1 INH	2 **AIS** 2 IMM	3 **STA** 2 DIR	4 **STA** 3 EXT	4 **STA** 3 IX2	5 **STA** 4 SP2	3 **STA** 2 IX1	4 **STA** 3 SP1	2 **STA** 1 IX
8	2 **PULX** 1 INH	1 **CLC** 1 INH	2 **EOR** 2 IMM	3 **EOR** 2 DIR	4 **EOR** 3 EXT	4 **EOR** 3 IX2	5 **EOR** 4 SP2	3 **EOR** 2 IX1	4 **EOR** 3 SP1	2 **EOR** 1 IX
9	2 **PSHX** 1 INH	1 **SEC** 1 INH	2 **ADC** 2 IMM	3 **ADC** 2 DIR	4 **ADC** 3 EXT	4 **ADC** 3 IX2	5 **ADC** 4 SP2	3 **ADC** 2 IX1	4 **ADC** 3 SP1	2 **ADC** 1 IX
A	2 **PULH** 1 INH	2 **CLI** 1 INH	2 **ORA** 2 IMM	3 **ORA** 2 DIR	4 **ORA** 3 EXT	4 **ORA** 3 IX2	5 **ORA** 4 SP2	3 **ORA** 2 IX1	4 **ORA** 3 SP1	2 **ORA** 1 IX
B	2 **PSHH** 1 INH	2 **SEI** 1 INH	2 **ADD** 2 IMM	3 **ADD** 2 DIR	4 **ADD** 3 EXT	4 **ADD** 3 IX2	5 **ADD** 4 SP2	3 **ADD** 2 IX1	4 **ADD** 3 SP1	2 **ADD** 1 IX
C	1 **CLRH** 1 INH	1 **RSP** 1 INH		3 **JMP** 2 DIR	5 **JMP** 3 EXT	4 **JMP** 3 IX2		3 **JMP** 2 IX1		2 **JMP** 1 IX
D		1 **NOP** 1 INH	4 **BSR** 2 IMM	3 **JSR** 2 DIR	4 **JSR** 3 EXT	5 **JSR** 3 IX2		5 **JSR** 2 IX1		4 **JSR** 1 IX
E	1 **STOP** 1 INH	Prebyte for stack indexed instructions	2 **LDX** 2 IMM	3 **LDX** 2 DIR	4 **LDX** 3 EXT	4 **LDX** 3 IX2	5 **LDX** 4 SP2	3 **LDX** 2 IX1	4 **LDX** 3 SP1	2 **LDX** 1 IX
F	1 **WAIT** 1 INH	1 **TXA** 1 INH	2 **AIX** 2 IMM	3 **STX** 2 DIR	4 **STX** 3 EXT	4 **STX** 3 IX2	5 **STX** 4 SP2	3 **STX** 2 IX1	4 **STX** 3 SP1	2 **STX** 1 IX

INH	Inherent	REL	Relative	SP1	Stack Pointer, 8-Bit Offset
IMM	Immediate	IX	Indexed, No Offset	SP2	Stack Pointer, 16-Bit Offset
DIR	Direct	IX1	Indexed, 8-Bit Offset	IX+	Indexed, No Offset with Post Increment
EXT	Extended	IX2	Indexed, 16-Bit Offset		
DD	Direct-Direct	IMD	Immediate-Direct	IX1+	Index, 1-Byte Offset with Post Increment
IX+D	Indexed-Direct	DIX+	Direct-Indexed		

zum Inhaltsverzeichnis

Beispiele:

ADC

vorher VHINZC
A $0F H:X $4502 SP $00FF PC $2000 CC %1XX011

ADC #$02

nacher VHINZC
A $12 H:X $4502 SP $00FF PC $2002 CC %0XX000

zum Inhaltsverzeichnis

ADD

vorher VHINZC
A $0F H:X $4502 SP $00FF PC $2000 CC %1XX011

ADD #$02

nacher VHINZC
A $11 H:X $4502 SP $00FF PC $2002 CC %0XX000

zum Inhaltsverzeichnis

AIS

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %XXXXXX

AIS #$04

nacher VHINZC
A $XX H:X $4502 SP $00F4 PC $2002 CC %XXXXXX

zum Inhaltsverzeichnis

AIX

vorher VHINZC
A $XX H:X $4502 SP $00FF PC $2000 CC %XXXXXX

AIX #$02

nacher VHINZC
A $XX H:X $4504 SP $00FF PC $2002 CC %XXXXXX

zum Inhaltsverzeichnis

AND

vorher VHINZC
A $0F H:X $4502 SP $00FF PC $2000 CC %0XX101

AND #$F0

nacher VHINZC
A $00 H:X $4502 SP $00FF PC $2002 CC %0XX010

zum Inhaltsverzeichnis

ASL

vorher VHINZC
A $08 H:X $4502 SP $00FF PC $2000 CC %0XX101

ASLA

nacher VHINZC

A $10 H:X $4502 SP $00FF PC $2001 CC %0XX000

zum Inhaltsverzeichnis

ASR

vorher VHINZC
A $0F H:X $4502 SP $00FF PC $2000 CC %0XX101

ASRA

nacher VHINZC
A $07 H:X $4502 SP $00FF PC $2001 CC %0XX001

zum Inhaltsverzeichnis

BCC

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %XXXXX0

BCC $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %XXXXX0

zum Inhaltsverzeichnis

BCLR

vorher VHINZC
A $0F H:X $4502 SP $00FF PC $2000 CC %XXXXX0
M$4000 %00000010

BCLR 2,$4000

nacher VHINZC
A $0F H:X $4502 SP $00FF PC $2004 CC %XXXXX0
M$4000 %00000000

zum Inhaltsverzeichnis

BCS

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %XXXXX1

BCS $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %XXXXX1

zum Inhaltsverzeichnis

BEQ

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %XXXXX1

BEQ $06

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2008 CC %XXXXX1

zum Inhaltsverzeichnis

BGE

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %0XX0X0

BGE $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %0XX0X0

zum Inhaltsverzeichnis

BGT

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %0XX000

BGT $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %0XX000

zum Inhaltsverzeichnis

BHCC

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %01X000

BHCC $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %01X000

zum Inhaltsverzeichnis

BHCS

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %01X000

BHCS $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %01X000

zum Inhaltsverzeichnis

BHI

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %0XX000

BHI $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %0XX000

zum Inhaltsverzeichnis

BHS

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %0XX000

BHS $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %0XX000

zum Inhaltsverzeichnis

BIH

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %01X000
/IRQ = 1

BIH $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %01X000
/IRQ = 1

zum Inhaltsverzeichnis

BIL

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %01X000
/IRQ = 0

BIL $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %01X000
/IRQ = 0

zum Inhaltsverzeichnis

BIT

vorher VHINZC
A $01 H:X $4502 SP $00F0 PC $2000 CC %XXX00X

BIT #$01

nacher VHINZC
A $01 H:X $4502 SP $00F0 PC $2002 CC %0XX01X

zum Inhaltsverzeichnis

BLE

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %0XX01X

BGT $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %0XX01X

zum Inhaltsverzeichnis

BLO

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %XXXXX1

BCS $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %XXXXX1

zum Inhaltsverzeichnis

BLS

vorher VHINZC

A $XX H:X $4502 SP $00F0 PC $2000 CC %XXXX01

BCS $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %XXXX01

zum Inhaltsverzeichnis

BLT

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %0XX1XX

BGT $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %0XX1XX

zum Inhaltsverzeichnis

BMC

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %XX0XXX

BMC $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %XX0XXX

zum Inhaltsverzeichnis

BMI

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %XXX1XX

BMI $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %XXX1XX

zum Inhaltsverzeichnis

BMS

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %XX1XXX

BMC $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %XX1XXX

zum Inhaltsverzeichnis

BNE

vorher VHINZC
A $XX H:X $4502 SP $00F0 PC $2000 CC %XXXX0X

BNE $04

nacher VHINZC
A $XX H:X $4502 SP $00F0 PC $2006 CC %XXXX0X

zum Inhaltsverzeichnis

BPL

```
        vorher                                              VHINZC
        A $XX     H:X $4502   SP $00F0   PC $2000           CC %XXX0XX

        BPL       $04

        nacher                                              VHINZC
A $XX   H:X $4502         SP $00F0   PC $2006               CC %XXX0XX
```

zum Inhaltsverzeichnis

BRA

```
        vorher                                              VHINZC
        A $XX     H:X $4502   SP $00F0   PC $2000           CC %XXXXXX

        BRA       $04

        nacher                                              VHINZC
A $XX   H:X $4502         SP $00F0   PC $2006               CC %XXXXXX
```

zum Inhaltsverzeichnis

BRCLR

```
        vorher                                              VHINZC
A $0F   H:X $4502         SP $00FF   PC $2000               CC %XXXXX0
        M$4000    %00000100

        BRCLR     2,$4000,$4

        nacher                                              VHINZC
        A $0F     H:X $4502   SP $00FF   PC $2007           CC %XXXXX0
        M$4000    %00000100
```

zum Inhaltsverzeichnis

BRN

```
        vorher                                              VHINZC
        A $XX     H:X $4502   SP $00F0   PC $2000           CC %XXXXXX

        BRN       $04

        nacher                                              VHINZC
A $XX   H:X $4502         SP $00F0   PC $2002               CC %XXXXXX
```

zum Inhaltsverzeichnis

BRSET

```
        vorher                                              VHINZC
A $0F   H:X $4502         SP $00FF   PC $2000               CC %XXXXX0
        M$4000    %00000010

        BRSET     2,$4000,$4

        nacher                                              VHINZC
        A $0F     H:X $4502   SP $00FF   PC $2007           CC %XXXXX0
        M$4000    %00000010
```

zum Inhaltsverzeichnis

BSET

```
        vorher                                                  VHINZC
A $0F   H:X $4502       SP $00FF    PC $2000            CC %XXXXX0
        M$4000  %00000000

        BSET    2,$4000

        nacher                                                  VHINZC
        A $0F   H:X $4502   SP $00FF    PC $2004            CC %XXXXX0
        M$4000  %00000010
```

zum Inhaltsverzeichnis

BSR

```
        vorher                                                  VHINZC
        A $XX   H:X $4502   SP $00F0    PC $2000            CC %XXXXXX

        BSR     $08

        nacher                                                  VHINZC
A $XX   H:X $4502       SP $00ED    PC $200A            CC %XXXXXX
```

zum Inhaltsverzeichnis

CBEQ

```
        vorher                                                  VHINZC
A $0F   H:X $4000       SP $00FF    PC $2000            CC %XXXXXX
        M$4000  $0F

        CBEQ    X,$9

        nacher                                                  VHINZC
        A $0F   H:X $4000   SP $00FF    PC $200B            CC %XXXXXX
        M$4000  $0F
```

zum Inhaltsverzeichnis

CLC

```
        vorher                                                  VHINZC
        A $XX   H:X $4502   SP $00F0    PC $2000            CC %XXXXXX

        CLC

        nacher                                                  VHINZC
A $XX   H:X $4502       SP $00F0    PC $2001            CC %XXXXX0
```

zum Inhaltsverzeichnis

CLI

```
        vorher                                                  VHINZC
        A $XX   H:X $4502   SP $00F0    PC $2000            CC %XXXXXX

        CLI

        nacher                                                  VHINZC
A $XX   H:X $4502       SP $00F0    PC $2001            CC %XX0XXX
```

zum Inhaltsverzeichnis

CLR

```
        vorher                                                  VHINZC
        A $1F   H:X $4502   SP $00F0    PC $2000            CC %XXXXXX
```

```
        CLRA

        nacher                                                  VHINZC
A $00   H:X $4502       SP $00F0    PC $2001                    CC %0XX01X
```

zum Inhaltsverzeichnis

CMP

```
        vorher                                                  VHINZC
        A $04       H:X $4502   SP $00F0    PC $2000            CC %XXXXXX

        CMPA    #$04

        nacher                                                  VHINZC
A $04   H:X $4502       SP $00F0    PC $2002                    CC %0XX01X
```

zum Inhaltsverzeichnis

COM

```
        vorher                                                  VHINZC
        A $04       H:X $4502   SP $00F0    PC $2000            CC %XXXXXX

        COMA

        nacher                                                  VHINZC
A $FB   H:X $4502       SP $00F0    PC $2002                    CC %0XX001
```

zum Inhaltsverzeichnis

CPHX

```
        vorher                                                  VHINZC
A $0F   H:X $4000       SP $00FF    PC $2000                    CC %XXXXXX
        M$60:$61 $4000

        CBEQ    X,$9

        nacher                                                  VHINZC
        A $0F       H:X $4000   SP $00FF    PC $2003            CC %0XX010
        M$60:$61 $4000
```

zum Inhaltsverzeichnis

CPX

```
        vorher                                                  VHINZC
        A $04       H:X $4504   SP $00F0    PC $2000            CC %XXXXXX

        CMPA    #$04

        nacher                                                  VHINZC
A $04   H:X $4504       SP $00F0    PC $2002                    CC %0XX01X
```

zum Inhaltsverzeichnis

DAA

```
        LDA     #$78                ;A=$78
        ADD     #$49                ;A=$78+$49 = $C1; C=0, H=1

        vorher                                                  VHINZC
        A $C1       H:X $4504   SP $00F0    PC $2000            CC %X1XXX0

        DAA                         ;add $66; A=$27; C=1 {=127 BCD}

        nacher                                                  VHINZC
```

A $27 H:X $4504 SP $00F0 PC $2001 CC %00X001

zum Inhaltsverzeichnis

DBNZ

vorher VHINZC
A $01 H:X $4504 SP $00F0 PC $2000 CC %XXXXXX

DBNZA #$04

nacher VHINZC
A $00 H:X $4504 SP $00F0 PC $2006 CC %XXXXXX

zum Inhaltsverzeichnis

DEC

vorher VHINZC
A $01 H:X $4504 SP $00F0 PC $2000 CC %XXXXXX

DECA

nacher VHINZC
A $00 H:X $4504 SP $00F0 PC $2001 CC %0XX01X

zum Inhaltsverzeichnis

DIV

```
LDA     #20             ;20/4       (8Bit/8Bit)
CLRH
LDX     #4
```

vorher VHINZC
A $14 H:X $0004 SP $00F0 PC $2000 CC %XXXXXX

DIV

nacher VHINZC
A $05 H:X $0004 SP $00F0 PC $2001 CC %XXXX00

zum Inhaltsverzeichnis

EOR

vorher VHINZC
A $01 H:X $4504 SP $00F0 PC $2000 CC %XXXXXX

EOR #$03

nacher VHINZC
A $02 H:X $4504 SP $00F0 PC $2002 CC %0XX00X

zum Inhaltsverzeichnis

INC

vorher VHINZC
A $01 H:X $4504 SP $00F0 PC $2000 CC %XXXXXX

INCA

nacher VHINZC
A $02 H:X $4504 SP $00F0 PC $2001 CC %0XX00X

zum Inhaltsverzeichnis

JMP

```
vorher                                          VHINZC
A $01     H:X $4504   SP $00F0   PC $2000       CC %XXXXXX

JMP       $1000

nacher                                          VHINZC
A $02     H:X $4504   SP $00F0   PC $1000       CC %XXXXXX
```

zum Inhaltsverzeichnis

JSR

```
vorher                                          VHINZC
A $01     H:X $4504   SP $00F0   PC $2000       CC %XXXXXX

JSR       $1000

nacher                                          VHINZC
A $02     H:X $4504   SP $00EE   PC $1000       CC %XXXXXX
```

zum Inhaltsverzeichnis

LDA

```
vorher                                          VHINZC
A $XX     H:X $4502   SP $00FF   PC $2000       CC %XXXXXX
M$4000    $55

LDA       $4000

nacher                                          VHINZC
A $55     H:X $4502   SP $00FF   PC $2003       CC %0XX00X
M$4000    $55
```

zum Inhaltsverzeichnis

LDHX

```
vorher                                          VHINZC
A $0F     H:X $2222   SP $00FF   PC $2000       CC %XXXXXX
M$60:$61  $4000

LDHX      $60

nacher                                          VHINZC
A $0F     H:X $4000   SP $00FF   PC $2003       CC %0XX01X
M$60:$61  $4000
```

zum Inhaltsverzeichnis

LDX

```
vorher                                          VHINZC
A $XX     H:X $4502   SP $00FF   PC $2000       CC %XXXXXX
M$4000    $55

LDX       $4000

nacher                                          VHINZC
A $55     H:X $4555   SP $00FF   PC $2003       CC %0XX00X
M$4000    $55
```

zum Inhaltsverzeichnis

LSL

```
vorher                                                    VHINZC
A $08     H:X $4502   SP $00FF   PC $2000                 CC %0XX101

LSLA

nacher                                                    VHINZC
A $10     H:X $4502   SP $00FF   PC $2001                 CC %0XX000
```

zum Inhaltsverzeichnis

LSR

```
vorher                                                    VHINZC
A $0F     H:X $4502   SP $00FF   PC $2000                 CC %XXXXXX

LSRA

nacher                                                    VHINZC
A $07     H:X $4502   SP $00FF   PC $2001                 CC %0XX001
```

zum Inhaltsverzeichnis

MOV

```
vorher                                                    VHINZC
A $0F     H:X $4502   SP $00FF   PC $2000                 CC %XXXXXX
PORTA     $XX

MOV       #$AA,PORTA

nacher                                                    VHINZC
A $0F     H:X $4502   SP $00FF   PC $2003                 CC %0XX000
PORTA     $AA
```

zum Inhaltsverzeichnis

MUL

```
vorher                                                    VHINZC
A $0F     H:X $0002   SP $00FF   PC $2000                 CC %XXXXXX

MUL                                  ;2x15=30

nacher                                                    VHINZC
A $1E     H:X $0000   SP $00FF   PC $2001                 CC %X0XXX0
```

zum Inhaltsverzeichnis

NEG

```
vorher                                                    VHINZC
A $0F     H:X $0002   SP $00FF   PC $2000                 CC %XXXXXX

NEGX

nacher                                                    VHINZC
A $0F     H:X $00FE SP $00FF     PC $2001                 CC %X0X000
```

zum Inhaltsverzeichnis

NOP

```
vorher                                                    VHINZC
```

A $0F H:X $0002 SP $00FF PC $2000 CC %XXXXXX

NEGX

nacher VHINZC
A $0F H:X $0002 SP $00FF PC $2001 CC %XXXXXX

zum Inhaltsverzeichnis

NSA

vorher VHINZC
A $0F H:X $0002 SP $00FF PC $2000 CC %XXXXXX

NSA

nacher VHINZC
A $F0 H:X $0002 SP $00FF PC $2001 CC %XXXXXX

zum Inhaltsverzeichnis

ORA

vorher VHINZC
A $0F H:X $0002 SP $00FF PC $2000 CC %XXXXXX

ORA #$80

nacher VHINZC
A $8F H:X $0002 SP $00FF PC $2002 CC %0XX00X

zum Inhaltsverzeichnis

PSHA

vorher VHINZC
A $0F H:X $0002 SP $00FF PC $2000 CC %XXXXXX
M$00FF $XX

PSHA

nacher VHINZC
A $0F H:X $0002 SP $00FE PC $2001 CC %XXXXXX
M$00FF $0F

zum Inhaltsverzeichnis

PSHH

vorher VHINZC
A $0F H:X $1002 SP $00FF PC $2000 CC %XXXXXX
M$00FF $XX

PSHA

nacher VHINZC
A $0F H:X $1002 SP $00FE PC $2001 CC %XXXXXX
M$00FF $10

zum Inhaltsverzeichnis

PSHX

vorher VHINZC
A $0F H:X $1002 SP $00FF PC $2000 CC %XXXXXX
M$00FF $XX

PSHX

```
nacher                                                  VHINZC
A $0F     H:X $1002   SP $00FE   PC $2001               CC %XXXXXX
M$00FF    $02
```

zum Inhaltsverzeichnis

PULA

```
vorher                                                  VHINZC
A $XX     H:X $1002   SP $00FF   PC $2000               CC %XXXXXX
M$00FF    $02
```

PULA

```
nacher                                                  VHINZC
A $02     H:X $1002   SP $00FF   PC $2001               CC %XXXXXX
M$00FF    $02
```

zum Inhaltsverzeichnis

PULH

```
vorher                                                  VHINZC
A $0F     H:X $XX02 SP $00FF     PC $2000               CC %XXXXXX
M$00FF    $10
```

PULH

```
nacher                                                  VHINZC
A $0F     H:X $1002   SP $00FF   PC $2001               CC %XXXXXX
M$00FF    $10
```

zum Inhaltsverzeichnis

PULX

```
vorher                                                  VHINZC
A $0F     H:X $10XX SP $00FF     PC $2000               CC %XXXXXX
M$00FF    $02
```

PULX

```
nacher                                                  VHINZC
A $0F     H:X $1002   SP $00FF   PC $2001               CC %XXXXXX
M$00FF    $02
```

zum Inhaltsverzeichnis

ROL

```
vorher                                                  VHINZC
A $48     H:X $4502   SP $00FF   PC $2000               CC %0XX101
```

ROLA

```
nacher                                                  VHINZC
A $81     H:X $4502   SP $00FF   PC $2001               CC %0XX000
```

zum Inhaltsverzeichnis

ROR

```
vorher                                                  VHINZC
A $48     H:X $4502   SP $00FF   PC $2000               CC %0XX101
```

ROLA

nacher VHINZC
A $A4 H:X $4502 SP $00FF PC $2001 CC %0XX000

zum Inhaltsverzeichnis

RSP

vorher VHINZC
A $XX H:X $4502 SP $00E1 PC $2000 CC %0XX101

RSP

nacher VHINZC
A $XX H:X $4502 SP $00FF PC $2001 CC %0XX000

besser:

vorher VHINZC
A $XX H:X $4502 SP $00E1 PC $2000 CC %XXXXXX

```
RAM_END     EQU  $01FF

LDHX        RAM_END+1
TXS
```

nacher VHINZC
A $XX H:X $01FF SP $01FF PC $2004 CC %0XX000

zum Inhaltsverzeichnis

RTI

vorher VHINZC
A $XX H:X $XXXX SP $00F0 PC $XXXX CC %XXXXXX

RTI

nacher VHINZC
A $A4 H:X $XX02 SP $00F5 PC $2000 CC %000100

zum Inhaltsverzeichnis

RTS

vorher VHINZC
A $XX H:X $XXXX SP $00F0 PC $XXXX CC %XXXXXX

RTI

nacher VHINZC
A $XX H:X $XXXX SP $00F2 PC $2000 CC %XXXXXX

zum Inhaltsverzeichnis

SBC

vorher VHINZC
A $08 H:X $4502 SP $00FF PC $2000 CC %1XX011

SBC #$02

nacher VHINZC
A $05 H:X $4502 SP $00FF PC $2002 CC %0XX000

zum Inhaltsverzeichnis

SEC

```
        vorher                                                  VHINZC
A $XX   H:X $XXXX SP $00FF   PC $2000                   CC %XXXXXX

        SEC

        nacher                                                  VHINZC
A $XX   H:X $XXXX SP $00FF   PC $2001                   CC %XXXXX1
```

zum Inhaltsverzeichnis

SEI

```
        vorher                                                  VHINZC
A $XX   H:X $XXXX SP $00FF   PC $2000                   CC %XXXXXX

        SEI

        nacher                                                  VHINZC
A $XX   H:X $XXXX SP $00FF   PC $2001                   CC %XX1XXX
```

zum Inhaltsverzeichnis

STA

```
        vorher                                                  VHINZC
A $55   H:X $4502       SP $00FF   PC $2000             CC %XXXXXX
        M$4000  $XX

        STA     $4000

        nacher                                                  VHINZC
        A $55   H:X $4502  SP $00FF   PC $2003          CC %0XX00X
        M$4000  $55
```

zum Inhaltsverzeichnis

STHX

```
        vorher                                                  VHINZC
A $55   H:X $4502       SP $00FF   PC $2000             CC %XXXXXX
        M$0040:0041     $XX:XX

        STHX    $40

        nacher                                                  VHINZC
        A $55   H:X $4502  SP $00FF   PC $2003          CC %0XX000
        M$0040:0041     $45:02
```

zum Inhaltsverzeichnis

STOP

```
        vorher                                                  VHINZC
A $55   H:X $4502       SP $00FF   PC $2000             CC %XXXXXX

        STOP

        nacher                                                  VHINZC
        A $55   H:X $4502  SP $00FF   PC $2001          CC %XX0XXX
```

/IRQ-Pin freigegeben! Oszillator gestoppt!

zum Inhaltsverzeichnis

STX

vorher VHINZC
A $55 H:X $4502 SP $00FF PC $2000 CC %XXXXXX
M$4000 $XX

STX $4000

nacher VHINZC
A $55 H:X $4502 SP $00FF PC $2003 CC %0XX00X
M$4000 $02

zum Inhaltsverzeichnis

SUB

vorher VHINZC
A $08 H:X $4502 SP $00FF PC $2000 CC %1XX011

SUB #$02

nacher VHINZC
A $06 H:X $4502 SP $00FF PC $2002 CC %0XX000

zum Inhaltsverzeichnis

SWI

vorher VHINZC
A $08 H:X $4502 SP $00FF PC $2000 CC %1XX011
M$FFFC:FFFD $38:00

SWI #$02

nacher VHINZC
A $06 H:X $4502 SP $00FF PC $3800 CC %XX1XXX
M$FFFC:FFFD $38:00

zum Inhaltsverzeichnis

TAP

vorher VHINZC
A $06 H:X $4502 SP $00FF PC $2000 CC %XXXXXX

TAP

nacher VHINZC
A $06 H:X $4502 SP $00FF PC $2001 CC %000110

zum Inhaltsverzeichnis

TAX

vorher VHINZC
A $06 H:X $4502 SP $00FF PC $2000 CC %XXXXXX

TAX

nacher VHINZC
A $06 H:X $4506 SP $00FF PC $2001 CC %XXXXXX

zum Inhaltsverzeichnis

TPA

vorher VHINZC
A $XX H:X $4502 SP $00FF PC $2000 CC %001010

TAX

nacher VHINZC
A $0A H:X $4502 SP $00FF PC $2001 CC %XXXXXX

zum Inhaltsverzeichnis

TST

vorher VHINZC
A $06 H:X $4502 SP $00FF PC $2000 CC %XXXXXX

TSTA

nacher VHINZC
A $06 H:X $4502 SP $00FF PC $2001 CC %0XX00X

zum Inhaltsverzeichnis

TSX

vorher VHINZC
A $06 H:X $4502 SP $00F0 PC $2000 CC %XXXXXX

TSTA

nacher VHINZC
A $06 H:X $00F1 SP $00F0 PC $2001 CC %XXXXXX

zum Inhaltsverzeichnis

TXA

vorher VHINZC
A $06 H:X $4502 SP $00F0 PC $2000 CC %XXXXXX

TSTA

nacher VHINZC
A $02 H:X $4502 SP $00F0 PC $2001 CC %XXXXXX

zum Inhaltsverzeichnis

TXS

vorher VHINZC
A $06 H:X $0102 SP $00F0 PC $2000 CC %XXXXXX

TSTA

nacher VHINZC
A $06 H:X $0102 SP $0101 PC $2001 CC %XXXXXX

zum Inhaltsverzeichnis

WAIT

vorher VHINZC
A $55 H:X $4502 SP $00FF PC $2000 CC %XXXXXX

STOP

nacher VHINZC
A $55 H:X $4502 SP $00FF PC $2001 CC %XX0XXX

Interrupts freigegeben! CPU-Takt gestoppt!

zum Inhaltsverzeichnis