Teil 2: Die Befehle

Abkürzungen

A	Akkumulator
C	Übertrag (carry)
M	Speicheradresse (memory)
P	Statusregister
S	Stapelzeiger
X	Indexregister X
Y	Indexregister Y
ADR	Adresse
DATEN	Adressierte Daten
DISP	Abstand (displacement)
HEX	Hexadezimal
PC	Programmzähler
PCH	Programmzähler, höherwertig (high)
PCL	Programmzähler, niederwertig (low)
STAPEL	Inhalt der Stapelspitze
V	logisches ODER
Λ	logisches UND
∀	logisches EXKLUSIV-ODER
●	ändert
←	Zuweisung (erhält den Wert)
( )	Inhalt von
(M6)	Bitstelle 6 unter Adresse M
b	Einzelbit, besonders angegeben

ADC

Add with carry
Mit Übertrag addieren

Funktion: A ← (A) + DATEN + C

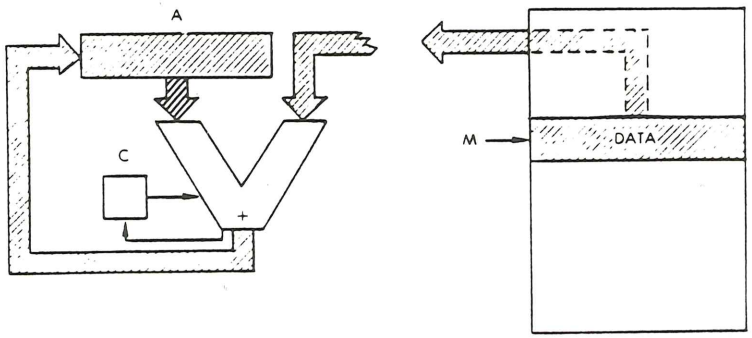
Format:

011bbb01	ADR/DATEN	ADR

Beschreibung:
Addiert den Inhalt der angegebenen Speicheradresse oder Konstanten zum Akkumulator plus dem Wert des Übertragsbits C. Das Ergebnis steht im Akkumulator, der Übertrag in C.

Besonderheiten:
— ADC kann sowohl dezimal als auch binär arbeiten. Der Arbeitsmodus wird durch die Dezimalflagge D bestimmt, die vor der Addition auf den in Frage kommenden Wert gesetzt sein muß.
— Um ohne Übertrag zu addieren, muß vor der Rechnung das Übertragsbit C mit CLC gelöscht werden.

Datenwege:



Addressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0 PAGE	IMMEDIATE	ABS, X	ABS, Y	(IND, X)	(IND), Y	0 PAGE, X	0 PAGE, Y	RELATIVE	(IND), Y
HEX			6D	65	69	7D	79	61	71	75			
BYTES			3	2	2	3	3	2	2	2			
CYCLES			4	3	2	4*	4*	6	5*	4			
bbb			011	001	010	111	110	000	100	101			

* PLUS 1 CYCLE IF CROSSING PAGE BOUNDARY

Flaggen:

N	V	B	D	I	Z	C
●	●				●	●

AND

Logisches UND

Funktion: A ← (A) ∧ DATEN

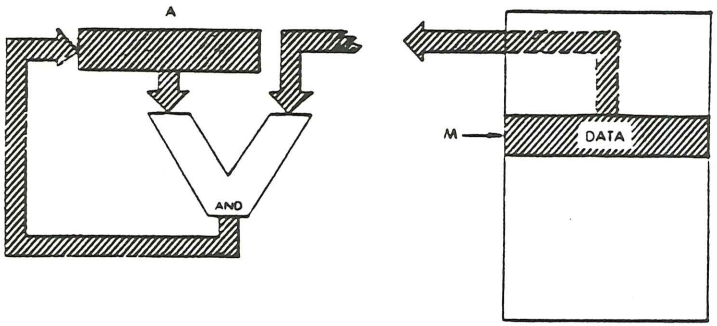
Format:

001bbb01	ADR/DATEN	ADR

Beschreibung:
UND-verknüpft den Akkumulatorinhalt bitweise mit den angegebenen Daten. Das Ergebnis steht im Akkumulator. Die Verknüpfung erfolgt nach der Tabelle:

A\M	0	1
0	0	0
1	0	1

Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS. X	ABS. Y	(IND. X)	(IND), Y	0-PAGE, X	0-PAGE, Y	RELATIVE	INDIRECT
HEX			2D	25	29	3D	39	21	31	35			
BYTES			3	2	2	3	3	2	2	2			
CYCLES			4	3	2	4*	4*	6	5*	4			
bbb			011	001	010	111	110	000	100	101			

*: PLUS 1 CYCLE IF CROSSING PAGE BOUNDARY.

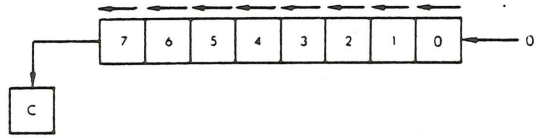
Flaggen:

N	V	B	D	I	Z	C
●					●	

ASL

arithmetic shift left
Arithmetisch links schieben

Funktion:

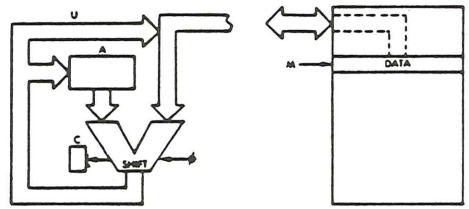


Format:

000bbb10	ADR	ADR

Beschreibung:
Akkumulator oder Speicherstelleninhalt um eine Bitstelle nach links schieben. Bit 0 wird zu „0", Bit 7 wird in das Übertragsbit C geschoben. Das Ergebnis steht in der Datenquelle (Akkumulator oder Speicherstelle).

Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS. X	ABS. Y	(IND. X)	(IND), Y	0-PAGE, X	0-PAGE, Y	RELATIVE	INDIRECT
HEX		0A	0E	06		1E				16			
BYTES		1	3	2		3				2			
CYCLES		2	6	5		7				6			
bbb		010	011	001		111				101			

Flaggen:

N	V	B	D	I	Z	C
●					●	●

BCC

branch on carry clear
Verzweigen bei gelöschtem Übertrag

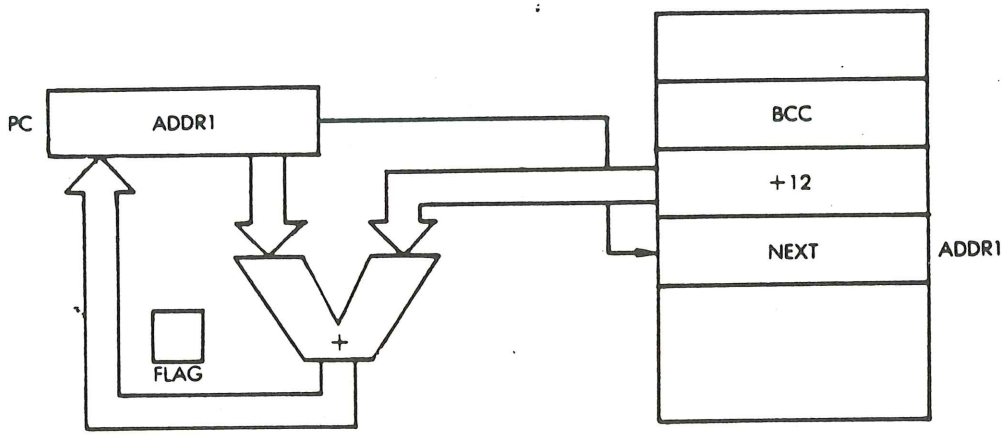
Funktion: Gehe zur angegebenen Adresse, falls C = 0 ist.

Format:

10010000	DISP

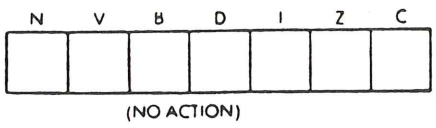
Beschreibung:
Testet die Übertragsflagge C. Ist C = 0, dann verzweigt die Abarbeitung zur nächsten Adresse plus dem angegebenen Abstand (im Zweierkomplementform: – 128 bis + 127). Ist C = 1, so erfolgt keine besondere Aktion. Da die Abstandsangabe zur Adresse des *folgenden* Befehls addiert wird, läßt sich insgesamt ein Raum von – 126 bis + 129 um den BCC-Befehl herum überspringen.

Datenwege:



Adressierungsarten:
Nur Relativadressierung möglich:
HEX = 90, 2 Bytes, 2 Taktzyklen + 1, wenn Verzweigung erfolgt,
+ 2, wenn die Seitengrenze überschritten wird.

Flaggen:



BCS

branch on carry set
Verzweigen bei gesetztem Übertrag

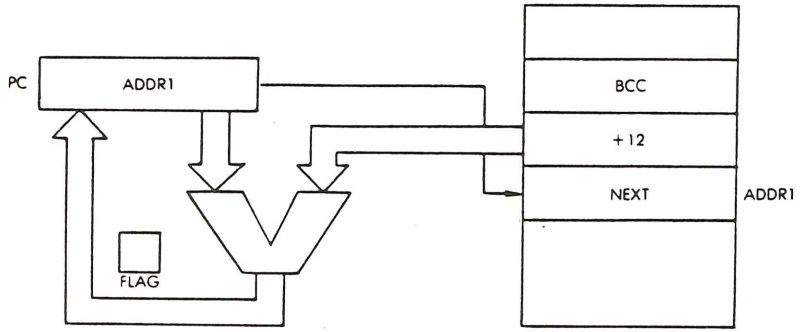
Funktion: Gehe zur angegebenen Adresse, falls C = 1 ist.

Format:

10110000	DISP

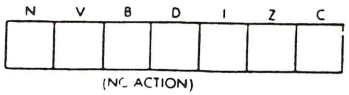
Beschreibung:
Testet die Übertragsflagge C. Ist C = 1, dann verzweigt die Abarbeitung zur nächsten Adresse plus dem angegebenen Abstand (in Zweierkomplementform: – 128 bis + 127). Ist C = 0, so erfolgt keine besondere Aktion. Da die Abstandsangabe zur Adresse des *folgenden* Befehls addiert wird, läßt sich insgesamt ein Raum von – 126 bis + 129 um den BCS-Befehl herum überspringen.

Datenwege:



Adressierungsarten:
Nur Relativadressierung möglich:
HEX = B0, 2 Bytes, 2 Taktzyklen + 1, wenn Verzweigung erfolgt,
+ 2, wenn die Seitengrenze überschritten wird.

Flaggen:



BEQ

branch on equal to zero
Verzweigen falls gleich Null

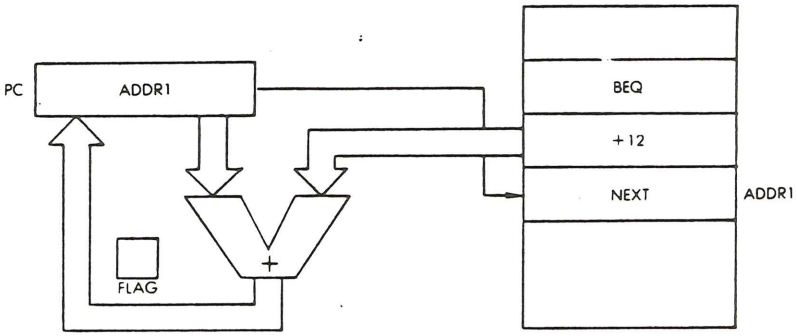
Funktion: Gehe zur angegebenen Adresse, falls Z = 1 ist.

Format:

11110000	DISP

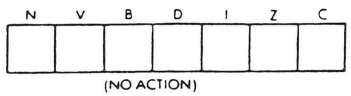
Beschreibung:
Testet die Nullflagge Z. Ist Z = 1, dann verzweigt die Abarbeitung zur nächsten Adresse plus dem angegebenen Abstand (in Zweierkomplementform: – 128 bis + 127). Ist Z = 0, so erfolgt keine besondere Aktion. Da die Abstandsangabe zur Adresse des *folgenden* Befehls addiert wird, läßt sich insgesamt ein Raum von – 126 bis + 129 um den BEQ-Befehl herum überspringen.

Datenwege:



Adressierungsarten:
Nur Relativadressierung möglich:
HEX = F0, 2 Bytes, 2 Taktzyklen + 1, wenn Verzweigung erfolgt,
+ 2, wenn die Seitengrenze überschritten wird.

Flaggen:



BIT

Speicherbits testen

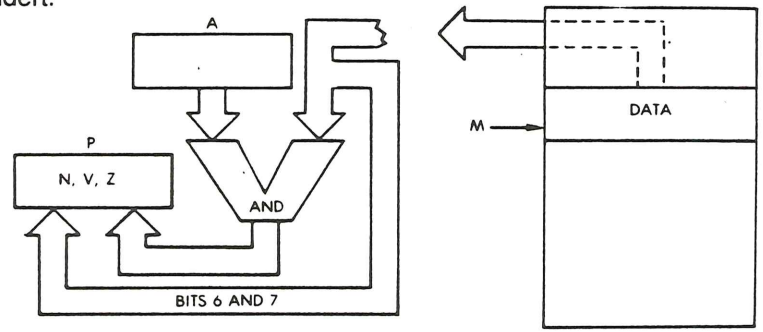
Funktion: $Z \leftarrow (A) \wedge (M), N \leftarrow (M7), V \leftarrow (M6)$

Format:

0010b100	ADR	ADR

Beschreibung:
Die Inhalte von A und M werden UND-verknüpft, das resultierende Byte jedoch nicht festgehalten. Das Vergleichsergebnis steht in der Z-Flagge: Z = 0, wenn alle getesteten Bits gelöscht (= „0") sind; Z = 1 sonst. Zusätzlich werden Bits 6 und 7 der adressierten Speicherstelle in die Flagge V bzw. N im Statusregister P übertragen. Der Akkumulatorinhalt bleibt unverändert.

Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS. X	ABS. Y	(IND, X)	(IND), Y	0-PAGE, X	0-PAGE, Y	RELATIVE	INDIRECT
HEX			2C	24									
BYTES			3	2									
CYCLES			4	3									
bbb			011	001									

Flaggen:

N	V	B	D	I	Z	C
M_7	M_6				●	

Befehlskodes:

Absolut:	00101100	16-Bit-Adresse
	b = 1	HEX = 2C 4 Taktzyklen
Zero-Page:	00100100	8-Bit-Adresse
	b = 0	HEX = 2C 3 Taktzyklen

BMI

branch on minus
Verzweigen falls negativ

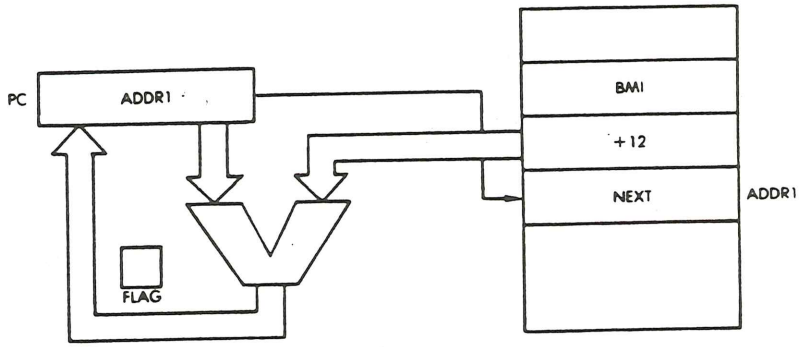
Funktion: Gehe zur angegebenen Adresse, falls N = 1 ist (Ergebnis < 0).

Format:

00110000	DISP

Beschreibung:
Testet die Vorzeichenflagge N. Ist N = 1, dann verzweigt die Abarbeitung zur nächsten Adresse plus dem angegebenen Abstand (in Zweierkomplementform: – 128 bis + 127). Ist N = 0, so erfolgt keine besondere Aktion. Da die Abstandsangabe zur Adresse des *folgenden* Befehls addiert wird, läßt sich insgesamt ein Raum von – 126 bis + 129 um den BMI-Befehl herum überspringen.

Datenwege:



Adressierungsarten:
Nur Relativadressierung möglich:
HEX = 30, 2 Bytes, 2 Taktzyklen + 1, wenn Verzweigung erfolgt,
+ 2, wenn die Seitengrenze überschritten wird.

Flaggen:

N	V	B	D	I	Z	C

(NO ACTION)

BNE

branch on not equal to zero
Verzweigung falls ungleich Null

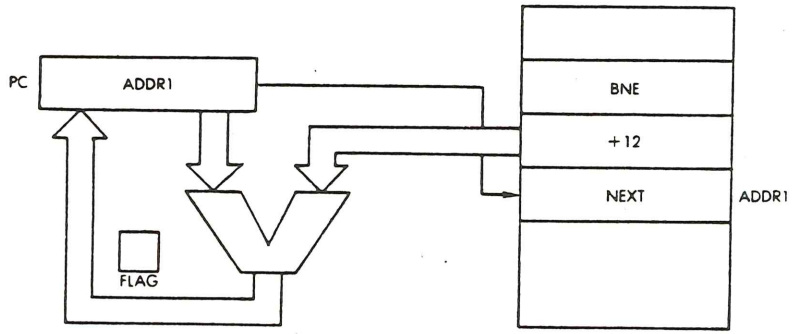
Funktion: Gehe zur angegebenen Adresse, falls Z = 0 ist. (Ergebnis ≠ 0)

Format:

11010000	DISP

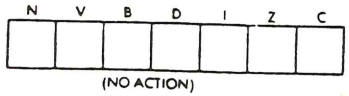
Beschreibung:
Testet die Nullflagge Z. Ist Z = 0, dann verzweigt die Abarbeitung zur nächsten Adresse plus dem angegebenen Abstand (in Zweierkomplementform: – 128 bis + 127). Ist Z = 1, so erfolgt keine besondere Aktion. Da die Abstandsangabe zur Adresse des *folgenden* Befehls addiert wird, läßt sich insgesamt ein Raum von – 126 bis + 129 um den BNE-Befehl herum überspringen.

Datenwege:



Adressierungsarten:
Nur Relativadressierung möglich:
HEX = D0, 2 Bytes, 2 Taktzyklen + 1, wenn Verzweigung erfolgt,
+ 2, wenn die Seitengrenze überschritten wird.

Flaggen:



BPL

branch on plus
Verzweigen falls positiv

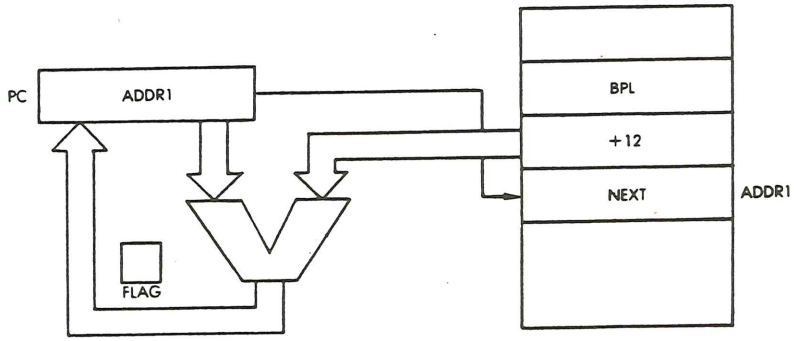
Funktion: Gehe zur angegebenen Adresse, falls N = 0 ist, (Ergebnis ⩾ 0)

Format:

00010000	DISP

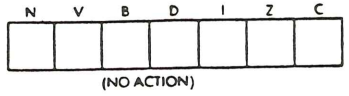
Beschreibung:
Testet die Vorzeichenflagge N. Ist N = 0, dann verzweigt die Abarbeitung zur nächsten Adresse plus dem angegebenen Abstand (in Zweierkomplementform: – 128 bis + 127). Ist N = 1, so erfolgt keine besondere Aktion. Da die Abstandsangabe zur Adresse des *folgenden* Befehls addiert wird, läßt sich insgesamt ein Raum von – 126 bis + 129 um den BPL-Befehl herum überspringen.

Datenwege:



Adressierungsarten:
Nur Relativadressierung möglich:
HEX = 10, 2 Bytes, 2 Taktzyklen + 1, wenn Verzweigung erfolgt,
+ 2, wenn die Seitengrenze überschritten wird.

Flaggen:



BRK

break
Unterbrechung durch Software

Funktion: STAPEL (PC) + 2, STAPEL (P), PC ← (FFFE, FFFF)

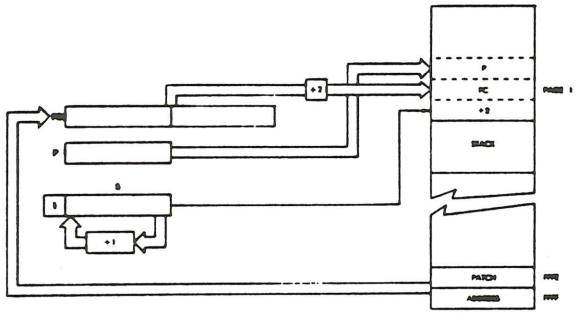
Format:

00000000

Beschreibung:
Arbeitet wie eine Programmunterbrechung: Der Wert des Programmzählers PC wird auf den Stapel gebracht, danach der Inhalt des Statusregisters P. Der Inhalt der Speicherstellen FFFE und FFFF (hexadezimal) wird darauf in den Programmzähler übertragen: (FFFE) kommt in den niederwertigen Teil (PCL), (FFFF) kommt in den höherwertigen Teil (PCH). Der auf den Stapel gebrachte P-Inhalt hat die B-Flagge auf „1" gesetzt, um so BRK von IRQ zu unterscheiden.

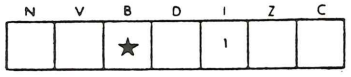
Wichtig:
Anders als im Falle der Unterbrechung durch Hardware wird (PC) + 2 gespeichert. Das wird u. U. nicht auf den nächsten Befehl zeigen, so daß eine Korrektur notwendig werden kann. Man macht das wegen des üblichen Einsatzes von BRK als Ersatz eines Zweibytebefehls beim Austesten von Programmen.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = 00, 1 Byte, 7 Taktzyklen

Flaggen:



Bau: B wird gesetzt, bevor P auf den Stapel kommt

BVC

branch on overflow clear
Verzweigung falls kein Überlauf

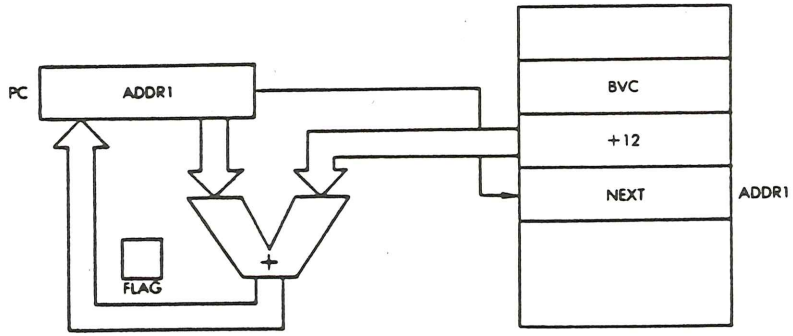
Funktion: Gehe zur angegebenen Adresse, falls V = 0 ist.

Format:

01010000	DISP

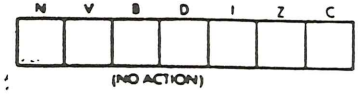
Beschreibung:
Testet die Überlaufsflagge V. Ist V = 0, dann verzweigt die Abarbeitung zur nächsten Adresse plus dem angegebenen Abstand (in Zweierkomplementform: – 128 bis + 127). Ist V = 1, so erfolgt keine besondere Aktion. Da die Abstandsangabe zur Adresse des *folgenden* Befehls addiert wird, läßt sich insgesamt ein Raum von – 126 bis + 129 um den BVC-Befehl herum überspringen.

Datenwege:



Adressierungsarten:
Nur Relativadressierung möglich:
HEX = 50, 2 Bytes, 2 Taktzyklen + 1, wenn Verzweigung erfolgt,
+ 2, wenn die Seitengrenze überschritten wird.

Flaggen:



BVS

branch on overflow set
Verzweigen falls Überlauf

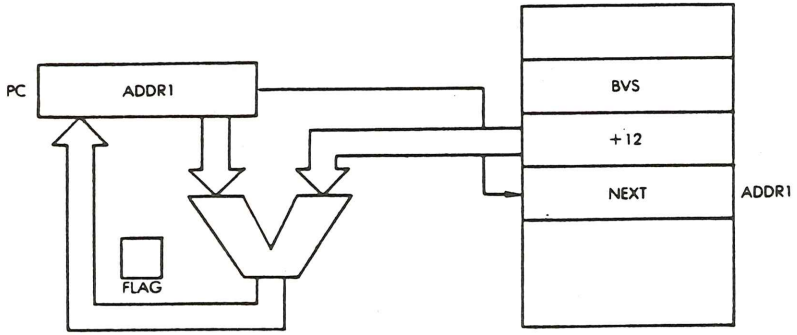
Funktion: Gehe zur angegebenen Adresse, falls V = 1 ist.

Format:

01110000	DISP

Beschreibung:
Testet die Überlaufsflagge V. Ist V = 1, dann verzweigt die Abarbeitung zur nächsten Adresse plus dem angegebenen Abstand (in Zweierkomplementform: – 128 bis + 127). Ist V = 0, so erfolgt keine besondere Aktion. Da die Abstandsangabe zur Adresse des *folgenden* Befehls addiert wird, läßt sich insgesamt ein Raum von – 126 bis + 129 um den BVS-Befehl herum überspringen.

Datenwege:



Adressierungsarten:
Nur Relativadressierung möglich:
HEX = 70, 2 Bytes, 2 Taktzyklen + 1, wenn Verzweigung erfolgt,
+ 2, wenn die Seitengrenze überschritten wird.

Flaggen:

N	V	B	D	I	Z	C

(NO ACTION)

CLC

clear carry
Übertragsflagge löschen

Funktion: C ← 0

Format:

00011000

Beschreibung:
Die Übertragsflagge C wird auf 0 gesetzt. Das ist oft vor einem ADC-Befehl notwendig.

Adressierungsarten:
Nur implizit:
HEX = 18, 1 Byte, 2 Taktzyklen

Flaggen:

N	V	B	D	I	Z	C
						Ø

CLD

clear dezimal mode
Dezimalmodus abschalten

Funktion: D ← 0

Format:

11011000

Beschreibung:
Die Dezimalflagge D wird auf 0 gesetzt. Damit werden die nachfolgenden ADC- und SBC-Befehle binär durchgeführt, bis die Dezimalflagge wieder auf 1 gesetzt worden ist (durch SED).

Adressierungsarten:
Nur implizit:
HEX = D8, 1 Byte, 2 Taktzyklen

Flaggen:

N	V	B	D	I	Z	C
			Ø			

CLI

Clear interrupt mask
Unterbrechungsmaske löschen

Funktion: I ← 0

Format:

01011000

Beschreibung:
Die Unterbrechungsmaske I im Statusregister P wird auf 0 gesetzt. Dadurch werden weitere Programmunterbrechungen möglich gemacht. Ein Programm zur Unterbrechungsverarbeitung muß die I-Flagge immer löschen, soll keine folgende Unterbrechungsanforderung verlorengehen.

Adressierungsarten:
Nur implizit:
HEX = 58, 1 Byte, 2 Taktzyklen

Flaggen:

N	V	B	D	I	Z	C
				Ø		

CLV

Clear overflow flag
Überlaufsflagge löschen

Funktion: V ← 0

Format:

10111000

Beschreibung:
Die Überlaufsflagge V wird auf 0 gesetzt.

Adressierungsarten:
Nur implizit:
HEX = B8, 1 Byte, 2 Taktzyklen

Flaggen:

N	V	B	D	I	Z	C
	Ø					

CMP

compare to accumulator
Mit Akkumulator vergleichen

Funktion: N, Z, C ← (A) − DATEN

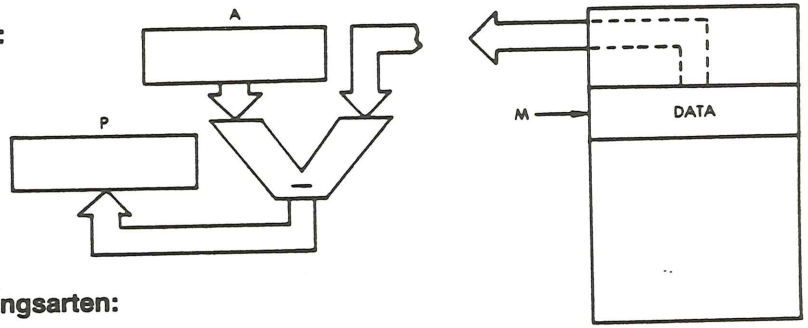
	(A) < DATEN	(A) = DATEN	(A) > DATEN
N:	0	0	1
Z:	0	1	0
C:	1	1	0

Format:

110bbb01	ADR/DATEN	ADR

Beschreibung:
Die adressierten Daten werden von Register A abgezogen, das Ergebnis jedoch nicht gespeichert. Es werden lediglich die drei Flaggen N, Z und C dem Ergebnis entsprechend gesetzt. Z = 1, wenn beide Werte gleich sind. N = 1, wenn der Inhalt von A kleiner als die adressierten Daten ist. C = 1, wenn der Inhalt von A größer oder gleich den angegebenen Daten ist.
Üblicherweise wird der Vergleichsbefehl von einer Verzweigung gefolgt. Dabei entdeckt BEQ Gleichheit, BNE Ungleichheit, BMI oder BCC die Kleiner-als-Bedingung und BEQ gefolgt von BCS oder BPL die Größer-als-Bedingungn. BCS und BPL verzweigen im Falle „größer oder gleich" und BEQ gefolgt von BCC oder BMI entdeckt „kleiner oder gleich".

Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS., X	ABS., Y	(IND., X)	(IND), Y	0-PAGE, X	0-PAGE, Y	RELATIVE	INDIRECT
HEX			CD	C5	C9	DD	D9	C1	D1	D5			
BYTES			3	2	2	3	3	2	2	2			
CYCLES			4	3	2	4*	4*	6	5*	4			
bbb			011	001	010	111	110	000	100	101			

*: PLUS 1 CYCLE IF CROSSING PAGE BOUNDARY.

Flaggen:

N	V	B	D	I	Z	C
●					●	●

CPX

compare to register X
Mit Register X vergleichen

Funktion: N, Z, C ← (X) − DATEN

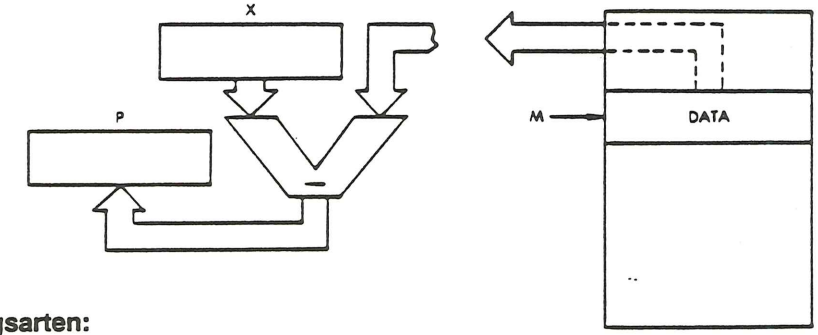
	(A) > DATEN	(A) = DATEN	(A) < DATEN
N:	0	0	1
Z:	0	1	0
C:	1	1	0

Format:

1110bb00	ADR/DATEN	ADR

Beschreibung:
Die adressierten Daten werden von Register X abgezogen, das Ergebnis jedoch nicht gespeichert. Es werden lediglich die drei Flaggen N, Z und C dem Ergebnis entsprechend gesetzt. Z = 1, wenn beide Werte gleich sind. N = 1, wenn der Inhalt von A kleiner als die adressierten Daten ist. C = 1, wenn der Inhalt von A größer oder gleich den angegebenen Daten ist.
Üblicherweise wird der Vergleichsbefehl von einer Verzweigung gefolgt. Dabei entdeckt BEQ Gleichheit, BNE Ungleichheit, BMI oder BCC die Kleiner-als-Bedingung und BEQ gefolgt von BCS oder BPL die Größer-als-Bedingung. BCS und BPL verzweigen im Falle „größer oder gleich" und BEQ gefolgt von BCC oder BMI entdeckt „kleiner oder gleich".

Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS, X	ABS, Y	(IND, X)	(IND), Y	0-PAGE, X	0-PAGE, Y	RELATIVE	INDIRECT
HEX			EC	E4	E0								
BYTES			3	2	2								
CYCLES			4	3	2								
bb			11	01	00								

Flaggen:

N	V	B	D	I	Z	C
					●	●

CPY

compare to register Y
Mit Register Y vergleichen

Funktion: N, Z, C ← (Y) − DATEN

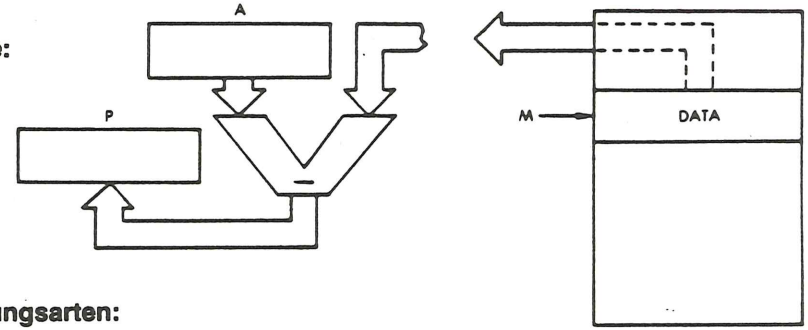
	(A) > DATEN	(A) = DATEN	(A) < DATEN
N:	0	0	1
Z:	0	1	0
C:	1	1	0

Format:

1100bb00	ADR/DATEN	ADR

Beschreibung:
Die adressierten Daten werden von Register Y abgezogen, das Ergebnis jedoch nicht gespeichert. Es werden lediglich die drei Flaggen N, Z und C dem Ergebnis entsprechend gesetzt. Z = 1, wenn beide Werte gleich sind. N = 1, wenn der Inhalt von A kleiner als die adressierten Daten ist. C = 1, wenn der Inhalt von A größer oder gleich den angegebenen Daten ist.
Üblicherweise wird der Vergleichsbefehl von einer Verzweigung gefolgt. Dabei entdeckt BEQ Gleichheit, BNE Ungleichheit, BMI oder BCC die Kleiner-als-Bedingung und BEQ gefolgt von BCS oder BPL die Größer-als-Bedingung. BCS und BPL verzweigen im Falle „größer oder gleich" und BEQ gefolgt von BCC oder BMI entdeckt „kleiner oder gleich".

Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS, X	ABS, Y	(IND, X)	(IND), Y	0-PAGE, X	0-PAGE, Y	RELATIVE	INDIRECT
HEX			CC	C4	C0								
BYTES			3	2	2								
CYCLES			4	3	2								
bb			11	01	00								

Flaggen:

N	V	B	D	I	Z	C
●					●	●

DEC

decrement memory
Speicher dekrementieren

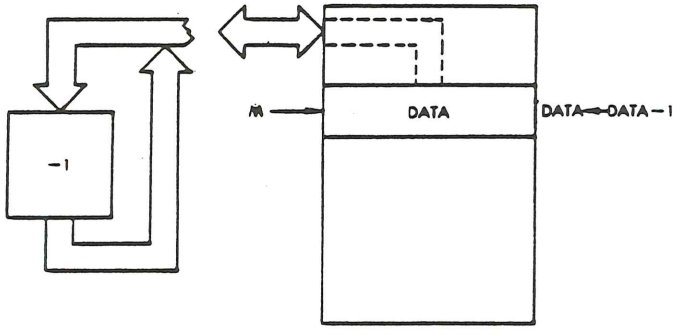
Funktion: M ← (M) − 1

Format:

110bb110	ADR	ADR

Beschreibung:
Der Inhalt der adressierten Speicherstelle wid um 1 heruntergezählt. Das Ergebnis wird an der angegebenen Adresse wieder abgelegt.

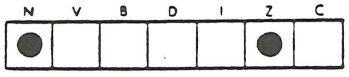
Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS, X	ABS, Y	(IND, X)	(IND), Y	0-PAGE, X	0-PAGE, Y	RELATIVE	INDIRECT
HEX			CE	C6		DE				D6			
BYTES			3	2		3				2			
CYCLES			6	5		7				6			
bb			01	00		11				10			

Flaggen:



DEX

decrement X
Register X dekrementieren

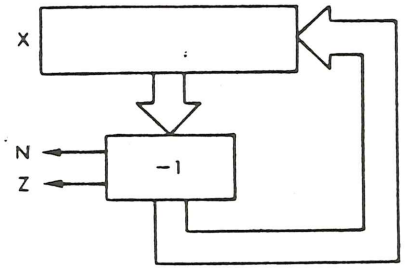
Funktion: X ← (X) − 1

Format:

11001010

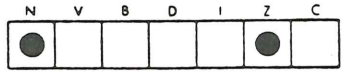
Beschreibung:
Der Inhalt von Register X wird um 1 heruntergezählt. Das gestattet den Einsatz von X als Zähler.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = CA, 1 Byte, 2 Taktzyklen

Flaggen:



DEY

decrement Y
Register Y dekrementieren

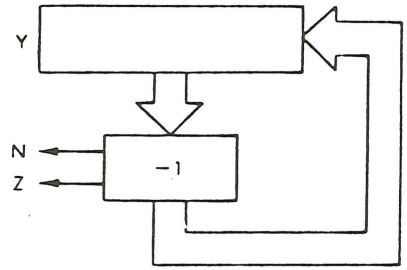
Funktion: Y ← (Y) − 1

Format:

10001000

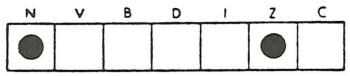
Beschreibung:
Der Inhalt von Register Y wird um 1 heruntergezählt. Das gestattet den Einsatz von Y als Zähler.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = 88, 1 Byte, 2 Taktzyklen

Flaggen:



EOR

exclusive-OR
EXKLUSIV-ODER verknüpfen

Funktion: A ← (A) V Daten

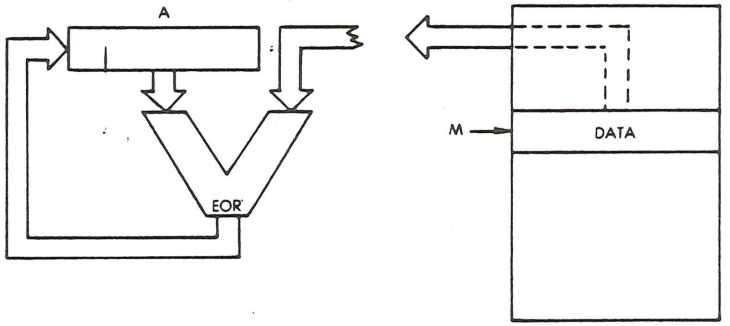
Format:

010bbb01	ADR/DATEN	ADR

Beschreibung:
EXKLUSIV-ODER-verknüpft den Akkumulatorinhalt bitweise mit den angegebenen Daten. Das Ergebnis steht im Akkumulator. Die Verknüpfung erfolgt nach der Tabelle:

Anmerkung:
EOR mit „−1" (11111111) dient zum Komplementieren des Akkumulatorinhalts.

Datenwege:

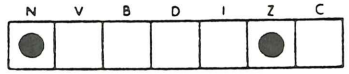


Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS, X	ABS, Y	(IND, X)	(IND), Y	0-PAGE, X	0-PAGE, Y	RELATIVE	INDIRECT
HEX			4D	45	49	5D	59	41	51	55			
BYTES			3	2	2	3	3	2	2	2			
CYCLES			4	3	2	4*	4*	6	5*	4			
bbb			011	001	010	111	110	000	100	101			

*: PLUS 1 CYCLE IF CROSSING PAGE BOUNDARY.

Flaggen:



INC

increment memory
Speicher inkrementieren

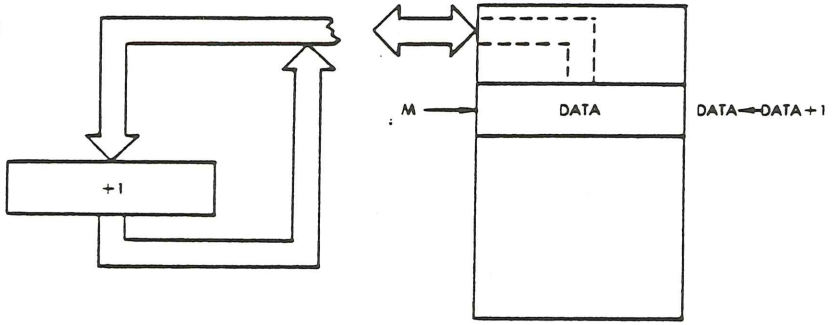
Funktion: M ← (M) + 1

Format:

111bb110	ADR	ADR

Beschreibung:
Der Inhalt der adressierten Speicherstelle wird um 1 weitergezählt. Das Ergebnis wird an der angegebenen Adresse wieder abgelegt.

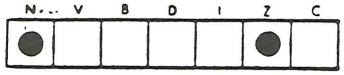
Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS, X	ABS, Y	(IND, X)	(IND, Y)	0-PAGE, X	0-PAGE, Y	RELATIVE	INDIRECT
HEX			EE	E6		FE				F6			
BYTES			3	2		3				2			
CYCLES			6	5		7				6			
bb			01	00		11				10			

Flaggen:



INX

increment X
Register X inkrementieren

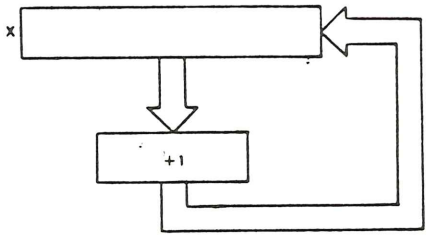
Funktion: X ← (X) + 1

Format:

11101000

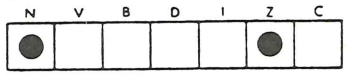
Beschreibung:
Der Inhalt von Register X wird um 1 weitergezählt. Das gestattet den Einsatz von X als Zähler.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = E8, 1 Byte, 2 Taktzyklen

Flaggen:



INY

increment Y
Register Y inkrementieren

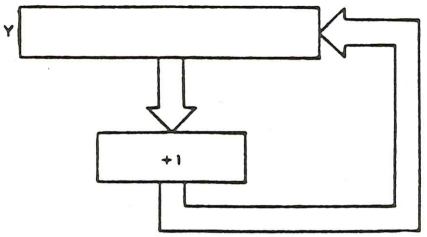
Funktion: Y ← (Y) + 1

Format:

11001000

Beschreibung:
Der Inhalt von Register Y wird um 1 weitergezählt. Das gestattet den Einsatz von Y als Zähler.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = C8, 1 Byte, 2 Taktzyklen

Flaggen:

N	V	B	D	I	Z	C
●					●	

JMP

jump to address
Zur angegebenen Adresse springen

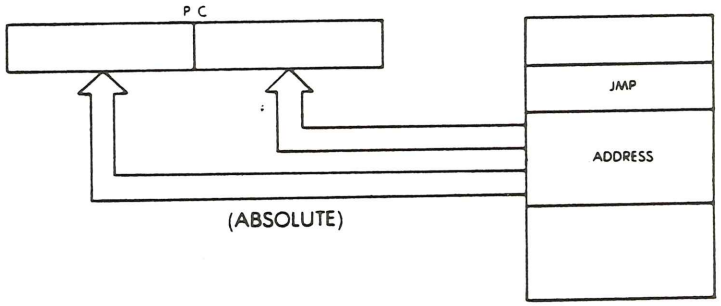
Funktion: PC ← ADR

Format:

01b01100	ADR	ADR

Beschreibung:
Der Programmzähler wird mit der angegebenen Adresse geladen, was einen Sprung im Programm bewirkt. Die Adresse kann absolut oder indirekt angegeben werden.

Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS, X	ABS, Y	(IND, X)	(IND), Y	0-PAGE, X	0-PAGE, Y	RELATIVE	INDIRECT
HEX			4C										6C
BYTES			3										3
CYCLES			3										5
b			0										1

Flaggen:

N	V	B	D	I	Z	C

(NO EFFECT)

JSR

jump to subroutine
Sprung zum Unterprogramm

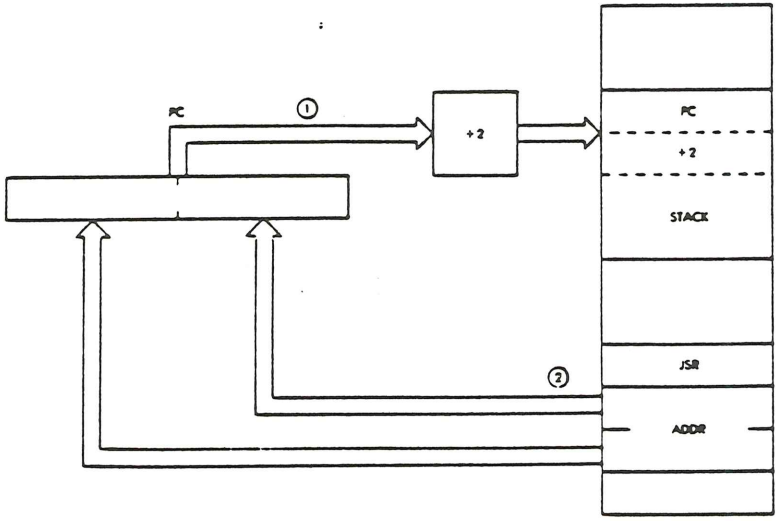
Funktion: STAPEL ← (PC) + 2
PC ← ADR

Format:

00100000	ADR	ADR

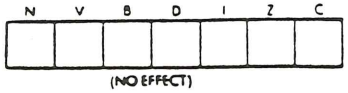
Beschreibung:
Der Programmzählerinhalt plus 2 wird auf den Stapel gebracht. (Das ist die dem JSR-Befehl folgende Adresse minus 1!) Darauf wird die Unterprogrammadresse in den Programmzähler geladen. Man bezeichnet diesen Befehl auch als Unterprogrammaufruf (subroutine CALL).

Datenwege:



Adressierungsarten:
Nur absolut:
HEX = 3 Bytes, 6 Taktzyklen
HEX = 20, 3 Bytes

Flaggen:



LDA

load accumulator
Akkumulator laden

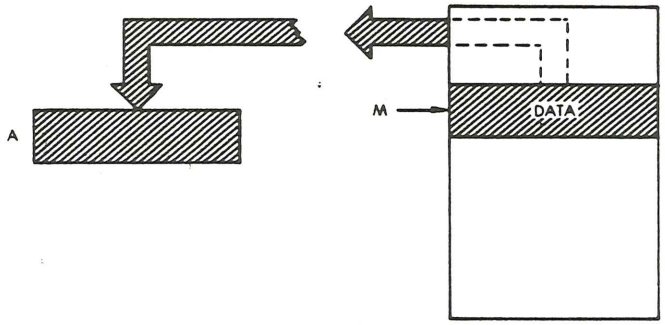
Funktion: A ← Daten

Format:

101bbb01	ADR/DATEN	ADR

Beschreibung:
Der Akkumulator wird mit den neuen Daten geladen.

Datenwege:

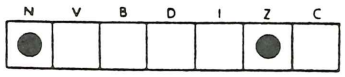


Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS, X	ABS, Y	(IND, X)	(IND), Y	0-PAGE, X	0-PAGE, Y	RELATIVE	INDIRECT
HEX			AD	A5	A9	BD	B9	A1	B1	B5			
BYTES			3	2	2	3	3	2	2	2			
CYCLES			4	3	2	4*	4*	6	5*	4			
bbb			011	001	010	111	110	000	100	101			

*: PLUS 1 CYCLE IF CROSSING PAGE BOUNDARY.

Flaggen:



LDX

load register X
Register X laden

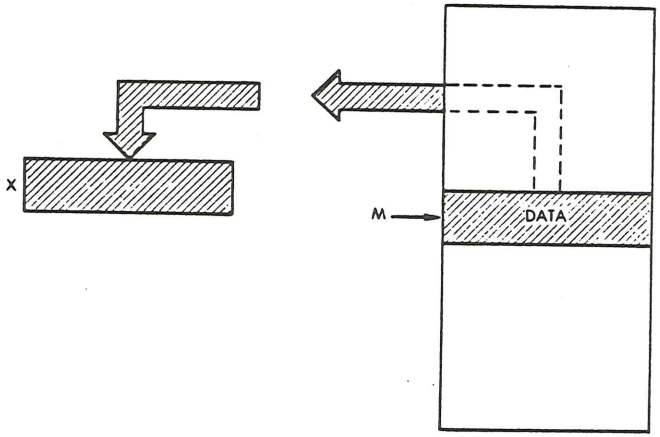
Funktion: X ← DATEN

Format:

101bbb10	ADR/DATEN	ADR

Beschreibung:
Indexregister X wird mit den neuen Daten geladen.

Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0 PAGE	IMMEDIATE	ABS. X	ABS. Y	(IND. X)	(IND), Y	0 PAGE. X	0 PAGE. Y	RELATIVE	INDIRECT
HEX			AE	A6	A2		BE				B6		
BYTES			3	2	2		3				2		
CYCLES			4	3	2		4*				4		
bbb			011	001	000		111				110		

*. PLUS 1 CYCLE IF CROSSING PAGE BOUNDARY.

Flaggen:

N	V	B	D	I	Z	C
●					●	

LDY

load register Y
Register Y laden

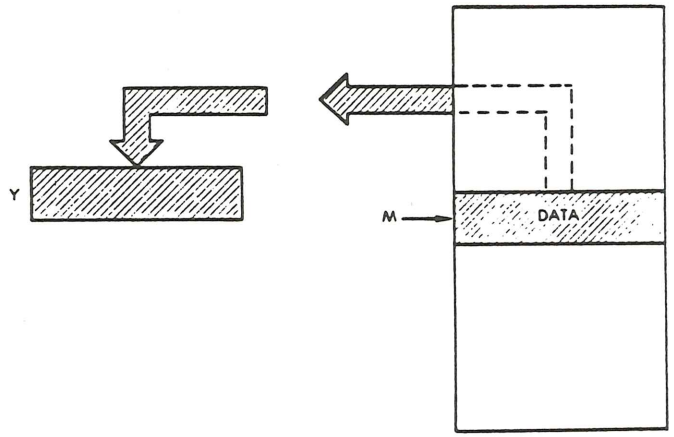
Funktion: Y ← Daten

Format:

101bbb00	ADR/DATEN	ADR

Beschreibung:
Indexregister Y wird mit den neuen Daten geladen.

Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0 PAGE	IMMEDIATE	ABS. X	ABS. Y	(IND. X)	(IND), Y	0 PAGE. X	0 PAGE. Y	RELATIVE	INDIRECT
HEX			AC	A4	A0	BC				B4			
BYTES			3	2	2	3				4			
CYCLES			4	3	2	4*				4			
bbb			011	001	000	111				101			

*. PLUS 1 CYCLE IF CROSSING PAGE BOUNDARY.

Flaggen:

N	V	B	D	I	Z	C
●					●	

NOP

no operation
Keine Operation

Funktion: Keine

Format:

11101010

Beschreibung:
Tut zwei Taktzyklen lang nichts. Man setzt diesen Befehl in erster Linie zur Verlängerung von Warteschleifen und als Platzhalter für möglicherweise einzufügende Programmbefehle ein.

Adressierungsarten:
Nur implizit:
HEX = EA, 1 Byte, 2 Taktzyklen

Flaggen:

N	V	B	D	I	Z	C

(NO ACTION)

ORA

inclusive OR with accumulator
(inklusiv) mit dem Akkumulator ODER-verknüpfen

Funktion: A ← (A) V DATEN

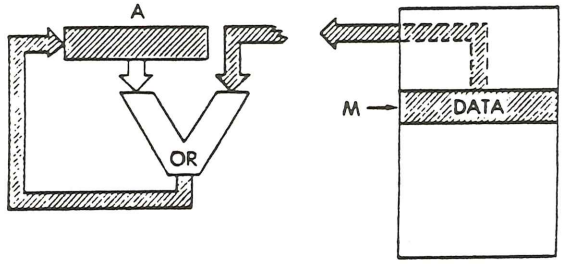
Format:

000bbb01	ADR	ADR

Beschreibung:
ODER-verknüpft den Akkumulatorinhalt bitweise mit den angegebenen Daten. Das Ergebnis steht im Akkumulator. Die Verknüpfung erfolgt nach der Tabelle:

	0	1
0	0	1
1	1	1

Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS. X	ABS. Y	(IND. X)	(IND). Y	0 PAGE. X	0 PAGE. Y	RELATIVE	INDIRECT
HEX			0D	05	09	1D	19	01	11	15			
BYTES			3	2	2	3	3	2	2	2			
CYCLES			4	3	2	4*	4*	6	5*	4			
bbb			011	001	010	111	110	000	100	101			

*: PLUS 1 CYCLE IF CROSSING PAGE BOUNDARY.

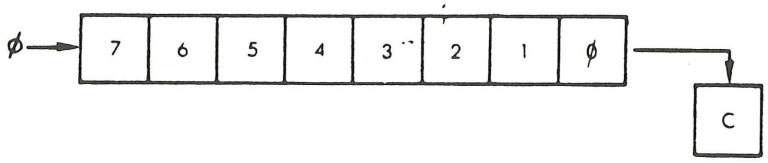
Flaggen:

N	V	B	D	I	Z	C
●					●	

LSR

logical shift right
Logisch rechts schieben

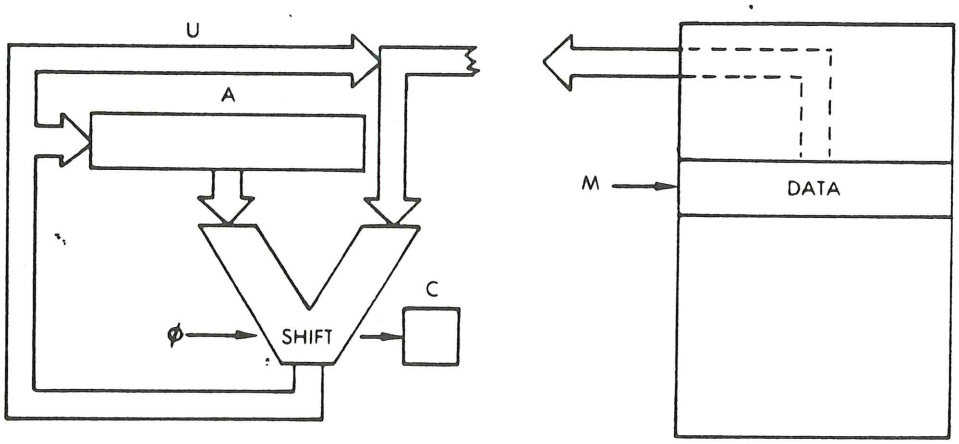
Funktion:



Format:

Beschreibung:
Akkumulator oder Speicherstelleninhalt um eine Bitstelle nach rechts schieben. Bit 7 wird zu „0", Bit 0 wird in das Übertragsbit C geschoben. Das Ergebnis steht in der Datenquelle (Akkumulator oder Speicherstelle).

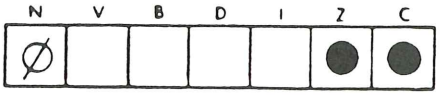
Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0 PAGE	IMMEDIATE	ABS. X	ABS. Y	(IND, X)	(IND), Y	0 PAGE, X	0 PAGE, Y	RELATIVE	INDIRECT
HEX		4A	4E	46		5E				56			
BYTES		1	3	2		3				2			
CYCLES		2	6	5		7				6			
bbb		010	011	001		111				101			

Flaggen:



PHA

push accumulator
Akkumulator auf den Stapel bringen

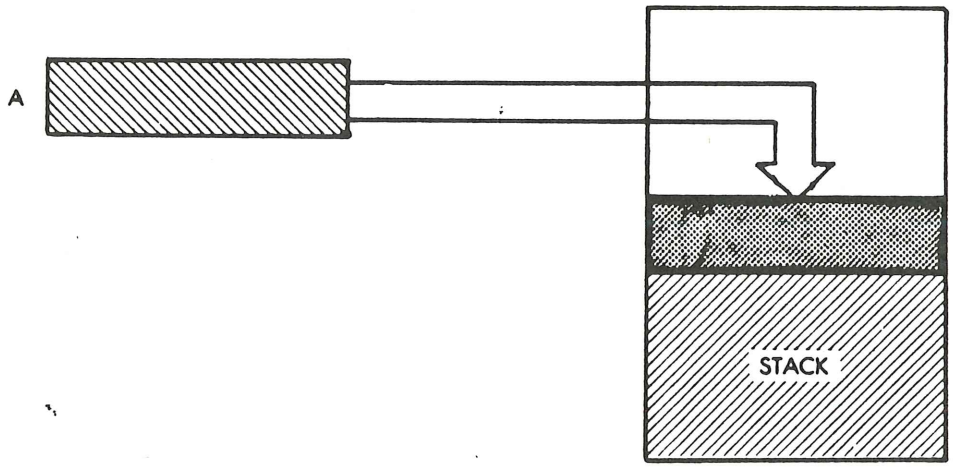
Funktion: STAPEL ← (A), S ← (S) − 1

Format:

01001000

Beschreibung:
Der Akkumualtorinhalt wird auf den Stapel gebracht und der Stapelzeiger S neu gesetzt (der Stapel wächst nach unten). Der Inhalt von A bleibt unverändert.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = 48, 1 Byte, 3 Taktzyklen

Flaggen:

N	V	B	D	I	Z	C

(NO EFFECT)

PHP

push processor status
Statusregister auf den Stapel bringen

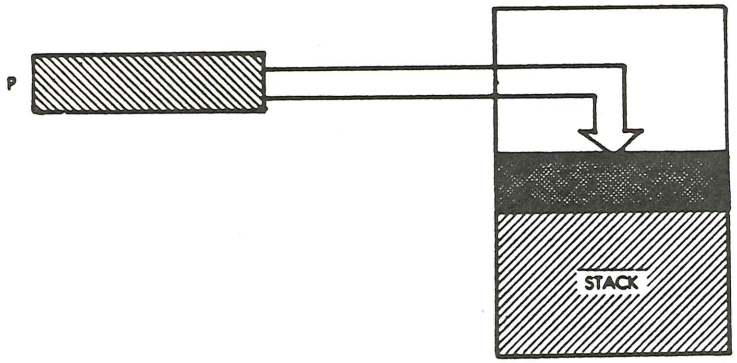
Funktion: STAPEL ← (P), S ← (S) − 1

Format:

00001000

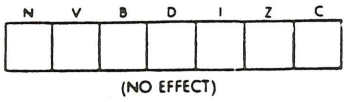
Beschreibung:
Der Statusregisterinhalt wird auf den Stapel gebracht und der Stapelzeiger S neu gesetzt (der Stapel wächst nach unten). Der Inhalt von P bleibt unverändert.

Datenweg:



Adressierungsarten:
Nur implizit:
HEX = 08, 1 Byte, 3 Taktzyklen

Flaggen:



PLA

pull accumulator
Akkumulator vom Stapel holen

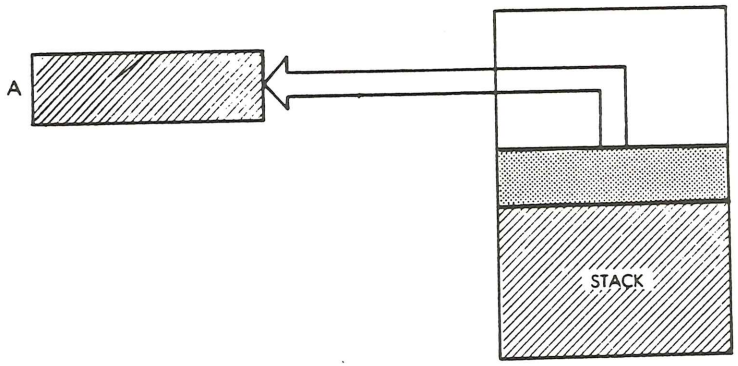
Funktion: A ← (STAPEL), S ← (S) + 1

Format:

01101000

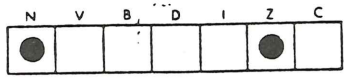
Beschreibung:
Der Akkumulator wird mit dem Inhalt der Stapelspitze geladen und der Stapelzeiger S neu gesetzt (der Stapel wächst nach unten), wodurch ein Stapelelement frei wird.

Datenweg:



Adressierungsarten:
Nur implizit:
HEX = 68, 1 Byte, 4 Taktzyklen

Flaggen:



PLP

pull processor status
Statusregister vom Stapel holen

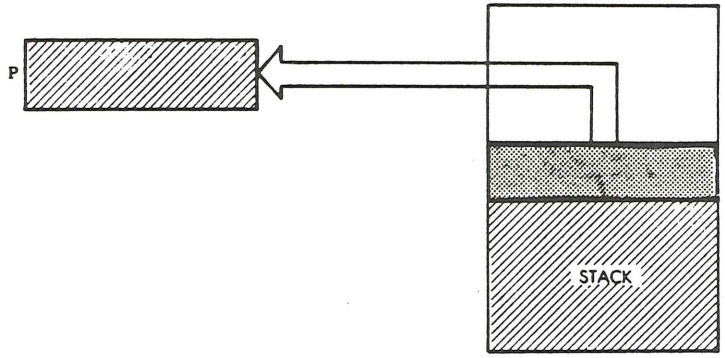
Funktion: P ← (STAPEL), S ← (S) + 1

Format:

00101000

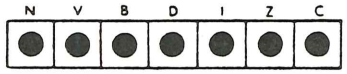
Beschreibung:
Das Statusregister wird mit dem Inhalt der Stapelspitze geladen und der Stapelzeiger S neu gesetzt (der Stapel wächst nach unten), wodurch ein Stapelelement frei wird.

Datenweg:



Adressierungsarten:
Nur implizit:
HEX = 28, 1 Byte, 4 Taktzyklen

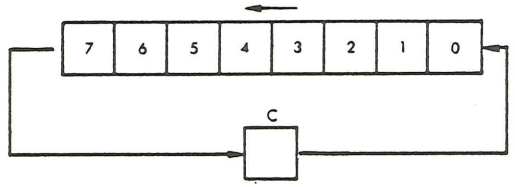
Flaggen:



ROL

rotate left one bit
Um ein Bit nach links rotieren

Funktion:

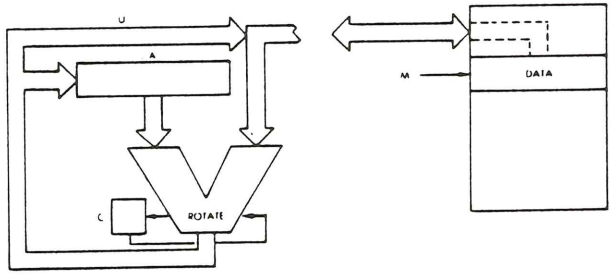


Format:

001bbb10	ADR	ADR

Beschreibung:
Der Inhalt der adressierten Speicherstelle oder des Akkumulators wird um ein Bit nach links verschoben. Dabei kommt der Inhalt des Übertragsbits C in Bit 0 und Bit 7 in C, so daß der 9-Bit-Inhalt von Register und C-Bit um eine Stelle rotiert.

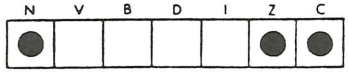
Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS, X	ABS, Y	(IND, X)	(IND), Y	0-PAGE, X	0-PAGE, Y	RELATIVE	INDIRECT
HEX		2A	2E	26		3E				36			
BYTES		1	3	2		3				2			
CYCLES		2	6	5		7				6			
bbb		010	011	001		111				101			

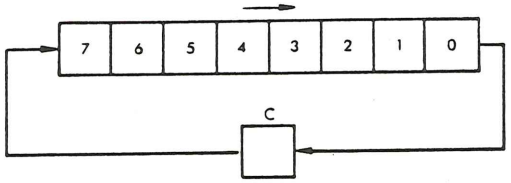
Flaggen:



ROR

rotate right one bit
Um ein Bit nach rechts rotieren

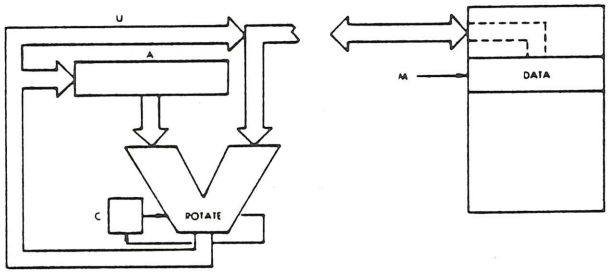
Funktion:



Format:

Beschreibung:
Der Inhalt der adressierten Speicherstelle oder des Akkumulators wird um ein Bit nach rechts verschoben. Dabei kommt der Inhalt des Übertragsbits C in Bit 7 und Bit 0 in C, so daß der 9-Bit-Inhalt von Register und C-Bit um eine Stelle rotiert.

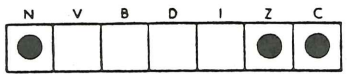
Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0 PAGE	IMMEDIATE	ABS. X	ABS. Y	(IND. X)	(IND), Y	0 PAGE, X	0 PAGE, Y	RELATIVE	INDIRECT
HEX		6A	6E	66		7E			76				
BYTES		1	3	2		3			2				
CYCLES		2	6	5		7			6				
bbb		010	011	001		111			101				

Flaggen:



RTI

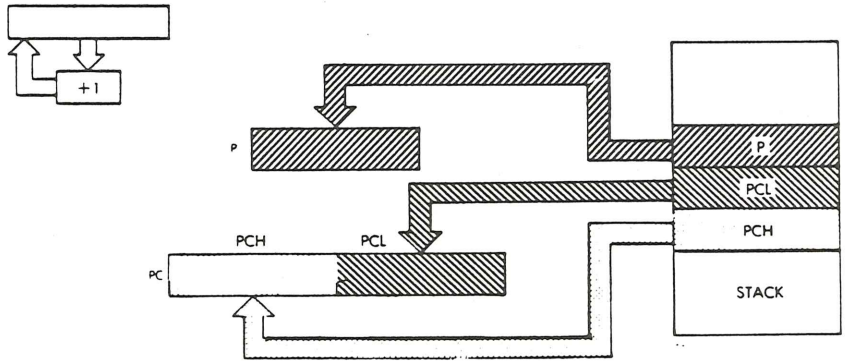
return from interrupt
Rückkehr aus einer Programmunterbrechung

Funktion:
P ← (STAPEL)
S ← (S) + 1
PCL ← (STAPEL)
S ← (S) + 1
PCH ← (STAPEL)
S ← (S) + 1

Format:

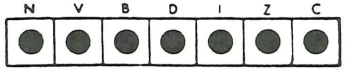
Beschreibung:
Stellt den ursprünglichen Zustand von Statusregister P und Programmzähler PC vor der Programmunterbrechung wieder her. Diese Werte wurden bei der Programmunterbrechung auf den Stapel gebracht. Der Stapelzeiger wird erneuert, womit drei Stapelplätze frei werden.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = 40, 1 Byte, 6 Taktzyklen

Flaggen:



RTS

return from subroutine
Aus einem Unterprogramm zurückkehren

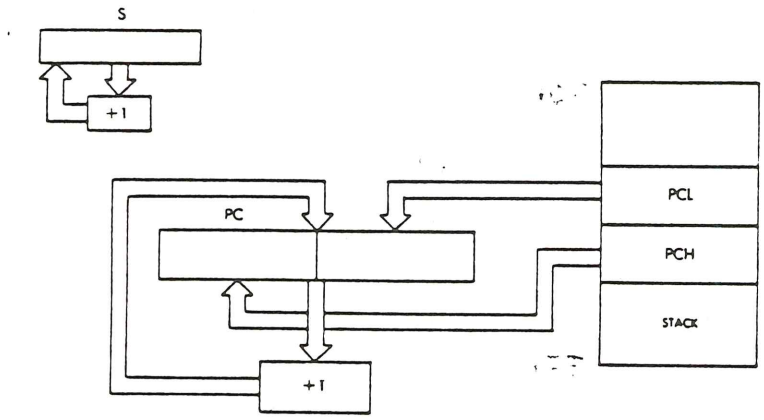
Funktion:
PCL ← (STAPEL)
S ← (S) + 1
PCH ← (STAPEL)
S ← (S) + 1
PC ← (PC) +1

Format:

01100000

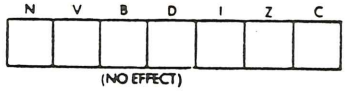
Beschreibung:
Programmzähler vom Stapel zurückholen und auf den nächsten Befehl stellen (d.h. um 1 weiterzählen; JSR speichert auf dem Stapel einen um 1 zu kleinen Wert!). Der Stapelzeiger wird neu gesetzt und gibt so zwei Stapeleinheiten frei.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = 60, 1 Byte, 6 Taktzyklen

Flaggen:



SBC

subtract with carry
Mit Übertrag subtrahieren

Funktion: A ← (A) − DATEN − C

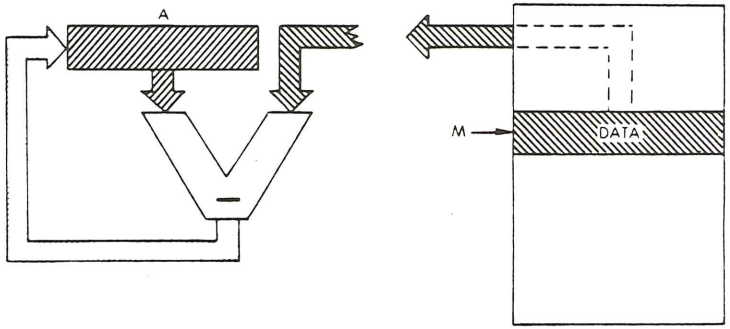
Format:

111bbb01	ADR/DATEN	ADR

Beschreibung:
Subtrahiert den Inhalt der angegebenen Speicheradresse oder Konstanten zum Akkumulator und noch den *komplementierten* Wert des Übertragsbits C. Das Ergebnis steht im Akkumulator, der (komplementierte) Übertrag in C.

Besonderheiten:
— SBC kann sowohl dezimal als auch binär arbeiten. Der Arbeitsmodus wird durch die Dezimalflagge D bestimmt, die vor der Addition auf den in Frage kommenden Wert gesetzt sein muß.
— Um ohne Übertrag zu subtrahieren, muß vor der Rechnung das Übertragsbit C mit SEC gesetzt werden.

Datenwege:

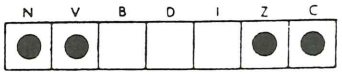


Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0 PAGE	IMMEDIATE	ABS, X	ABS, Y	(IND, X)	(IND), Y	0 PAGE, X	0 PAGE, Y	RELATIVE	INDIRECT
HEX			ED	E5	E9	FD	F9	E1	F1	F5			
BYTES			3	2	2	3	3	2	2	2			
CYCLES			4	3	2	4*	4*	6	5*	4			
bbb			011	001	010	111	110	000	100	101			

* PLUS 1 CYCLE IF CROSSING PAGE BOUNDARY.

Flaggen:



SEC

set carry
Übertragsflagge setzen

Funktion: C ← 1

Format:

00111000

Beschreibung:
Die Übertragsflagge C wird auf 1 gesetzt. Das ist oft vor einem SBC-Befehl notwendig.

Adressierungsarten:
Nur implizit:
HEX = 38, 1 Byte, 2 Taktzyklen

Flaggen:

N	V	B	D	I	Z	C
						1

SED

set decimal mode
Dezimalmodus einschalten

Funktion: D ← 1

Format:

11111000

Beschreibung:
Die Dezimalflagge D wird auf 1 gesetzt. Damit werden die nachfolgenden ADC- und SBC-Befehle dezimal durchgeführt, bis die Dezimalflagge wieder auf 0 gesetzt worden ist (durch CLD).

Adressierungsarten
Nur implizit:
HEX = F8, 1 Byte, 2 Taktzyklen

Flaggen:

N	V	B	D	I	Z	C
			1			

SEI

set interrupt mask
Unterbrechungsmaske setzen

Funktion: I ← 1

Format:

01111000

Beschreibung:
Die Unterbrechungsmaske I im Statusregister P wird auf 1 gesetzt. Dadurch werden weitere Programmunterbrechungen gesperrt. Eine Programmunterbrechung und die allgemeine Rücksetzoperation (reset) durch Hardware setzen I immer auf 1. Um weitere Unterbrechungen zu ermöglichen, muß I ausdrücklich wieder auf 0 gesetzt werden (durch CLI).

Adressierungsarten:
Nur implizit:
HEX = 78, 1 Byte, 2 Taktzyklen

Flaggen:

N	V	B	D	I	Z	C
				1		

STA

store accumulator in memory
Akkumulatorinhalt im Speicher ablegen

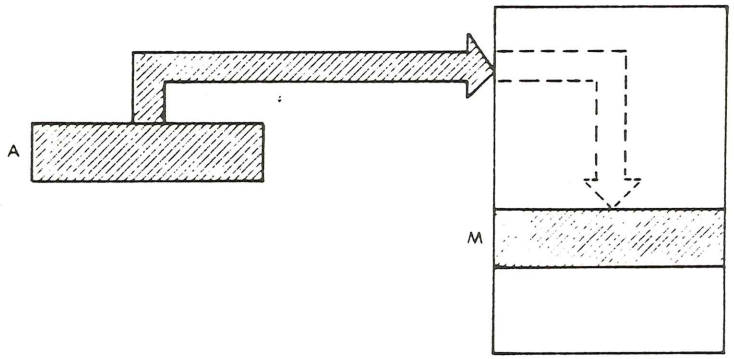
Funktion: M ← (A)

Format:

100bbb01	ADR	ADR

Beschreibung:
Der Akkumulatorinhalt wird in die adressierte Speicherstelle geschrieben. Dabei bleibt der Inhalt von A unverändert.

Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0 PAGE	IMMEDIATE	ABS. X	ABS. Y	(IND. X)	(IND), Y	0 PAGE. X	0 PAGE. Y	RELATIVE	INDIRECT
HEX			8D	85		9D	99	81	91	95			
BYTES			3	2		3	3	2	2	2			
CYCLES			4	3		5	5	6	6	4			
bbb			011	001		111	110	000	100	101			

Flaggen:

N	V	B	D	I	Z	C

(NO EFFECT)

STX

store X in memory
Register X im Speicher ablegen

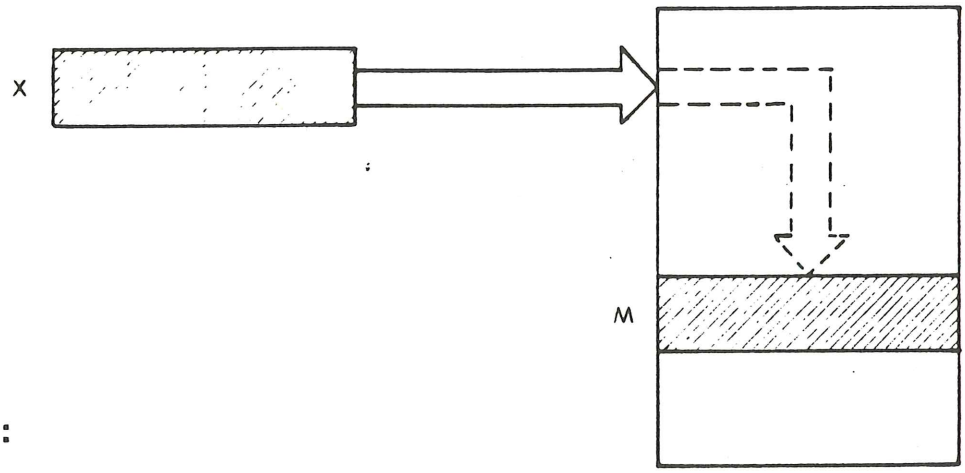
Funktion: M ← (X)

Format:

100bb110	ADR	ADR

Beschreibung:
Der Inhalt von Indexregister X wird an der adressierten Speicherstelle abgelegt. Der Inhalt von X bleibt unverändert.

Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0 PAGE	IMMEDIATE	ABS. X	ABS. Y	(IND, X)	(IND), Y	0 PAGE, X	0 PAGE, Y	RELATIVE	INDIRECT
HEX			8E	86							96		
BYTES			3	2							2		
CYCLES			4	3							4		
bb			01	00							10		

Flaggen:

N	V	B	D	I	Z	C

(NO EFFECT)

Befehlskodes:

Absolut:	10001110 bb = 01	16-Bit-Adresse HEX = 8E 4 Taktzyklen
Zero-Page:	10000110 bb = 00	8-Bit-Adresse HEX = 86 3 Taktzyklen
Zero-Page, X:	10010110 bb = 10	8-Bit-Adresse HEX = 96 4 Taktzyklen

STY

store Y in memory
Register Y im Speicher ablegen

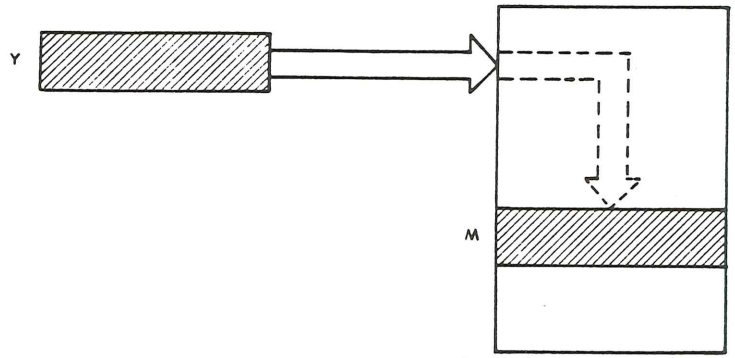
Funktion: M ← (Y)

Format:

100bb100	ADR	ADR

Beschreibung:
Der Inhalt von Indexregister Y wird an der adressierten Speicherstelle abgelegt. Der Inhalt von Y bleibt unverändert.

Datenwege:



Adressierungsarten:

	IMPLIED	ACCUMULATOR	ABSOLUTE	0-PAGE	IMMEDIATE	ABS. X	ABS. Y	(IND. X)	(IND. Y)	0-PAGE, X	0-PAGE, Y	RELATIVE	INDIRECT
HEX			8C	84						94			
BYTES			3	2						4			
CYCLES			4	3						4			
bb			01	00						10			

Flaggen:

N	V	B	D	I	Z	C

(NO EFFECT)

Befehlskodes:

Absolut:	10001100 bb = 01	16-Bit-Adresse HEX = 8C 4 Taktzyklen
Zero-Page:	10000100 bb = 00	8-Bit-Adresse HEX = 84 3 Taktzyklen
Zero-Page, Y:	10010100 bb = 10	8-Bit-Adresse HEX = 94 4 Taktzyklen

TAX

transfer accumulator into X
Akkumulator nach X übertragen

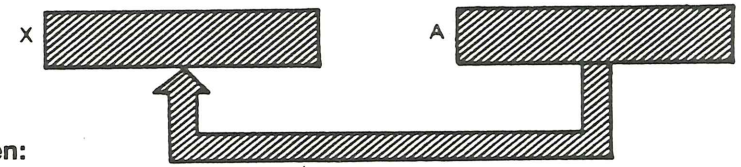
Funktion: X ← (A)

Format:

10101010

Beschreibung:
Kopiert den Akkumulator in Indexregister X. Der Inhalt von A bleibt unverändert.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = AA, 1 Byte, 2 Taktzyklen

Flaggen:

N	V	B	D	I	Z	C
●					●	

TAY

transfer accumulator into Y
Akkumulator nach Y übertragen

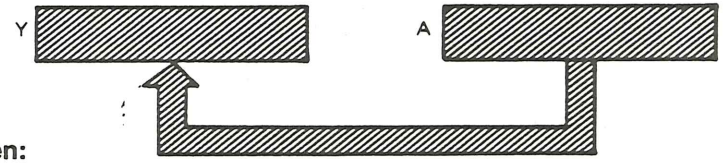
Funktion: Y ← (A)

Format:

10101000

Beschreibung:
Kopiert den Akkumulator in Indexregister Y. Der Inhalt von A bleibt unverändert.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = A8, 1 Byte, 2 Taktzyklen

Flaggen:

N	V	B	D	I	Z	C
●					●	

TSX

transfer S into X
Stapelzeiger S nach X übertragen

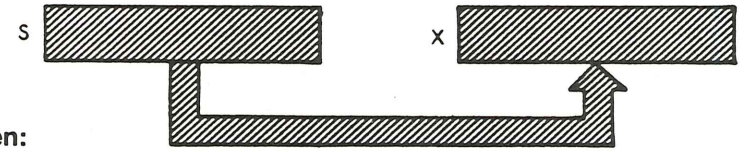
Funktion: X ← (S)

Format:

10111010

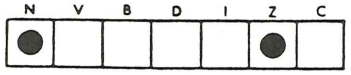
Beschreibung:
Der Inhalt des Stapelzeigers S wird in Indexregister X übertragen. Der Inhalt von S bleibt unverändert.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = BA, 1 Byte, 2 Taktzyklen

Flaggen:



TXA

transfer X into accumulator
Register X in Akkumulator übertragen

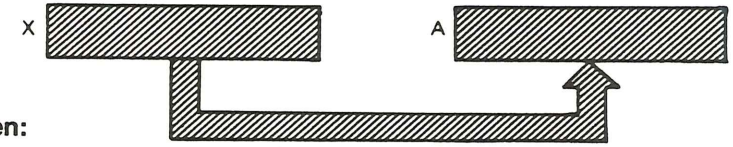
Funktion: A ← (X)

Format:

10001010

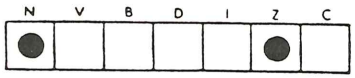
Beschreibung:
Der Inhalt von Indexregister X wird in den Akkumulator übertragen. Dabei bleibt der Inhalt von X unverändert.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = 8A, 1 Byte, 2 Taktzyklen

Flaggen:



TXS

transfer X into S
Register X in den Stapelzeiger S übertragen

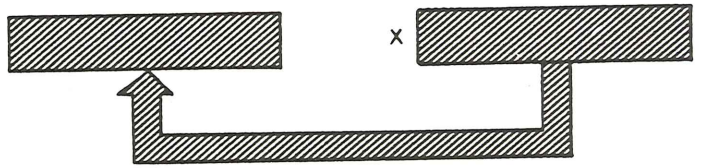
Funktion: S ← (X)

Format:

10011010

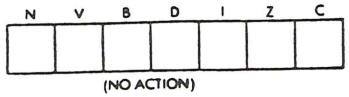
Beschreibung:
Der Inhalt von Indexregister X wird in den Stapelzeiger S übertragen. (Dies ist die einzige Möglichkeit, den Stapelzeiger zu laden!) Dabei bleibt der Inhalt von X unverändert.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = 9A, 1 Byte, 2 Taktzyklen

Flaggen:



TYA

transfer Y into accumulator
Register Y in Akkumulator übertragen

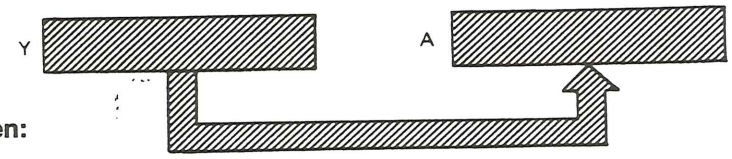
Funktion: A ← (Y)

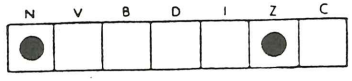
Format:

10011000

Beschreibung:
Der Inhalt von Indexregister Y wird in den Akkumulator übertragen. Dabei bleibt der Inhalt von Y unverändert.

Datenwege:



Adressierungsarten:
Nur implizit:
HEX = 98, 1 Byte, 2 Taktzyklen

Flaggen: