Befehl	Mnemonics Erklärung	Beschreibung		Adressierungsart	Beschreibung Adressierungsart	Bytelänge	Taktzyklen		beinflußte Register (NV-BDIZC)
Ladebefehle									
LDA #$3A	LoaD Akkumulator	Akkumulator direkt laden	$A9	unmittelbar	Einen festen Wert direkt in den Akku schreiben.	2	2		N Z
LDA $082D		Akkumulator absolut laden	$AD	absolut	Einen Wert von der angegebenen Adresse in den Akku laden.	3	4		N Z
LDA $082D,X		Akkumulator absolut mit X laden	$BD	absolut X-indiziert	Einen Wert von (angegebene Adresse + X-Register) in den Akku laden.	3	4-5	Beim Überschreiten der Page-Grenze werden 5 Taktzyklen benötigt, sonst 4.	N Z
LDA $082D,Y		Akkumulator absolut mit Y laden	$B9	absolut Y-indiziert	Einen Wert von (angegebene Adresse + Y-Register) in den Akku laden.	3	4-5	Beim Überschreiten der Page-Grenze werden 5 Taktzyklen benötigt, sonst 4.	N Z
LDA $FB		Akkumulator von Zero-Page laden	$A5	Zero-Page	Einen Wert von der angegebenen Zero-Page Adresse in den Akku laden.	2	3		N Z
LDA $FB,X		Akkumulator von Zero-Page mit X laden	$B5	Zero-Page X-indiziert	Einen Wert von (angegebener Zero-Page Adresse + X-Register) in den Akku laden.	2	4		N Z
LDA ($FB,X)		Akkumulator indirekt mit X-indiziert laden	$A1	indirekt X-indiziert	Einen Wert von der Adresse, die an der (angegebener Zero-Page Adresse + X-Register) steht, in den Akku laden.	2	6		N Z
LDA ($FB),Y		Akkumulator indirekt mit Y-nach-indiziert laden	$B1	indirekt Y-nach-indiziert	Einen Wert von (der Adresse die an der angegebener Zero-Page Adresse steht) + Y-Register in den Akku laden.	2	5-6	Beim Überschreiten der Page-Grenze werden 6 Taktzyklen benötigt, sonst 5.	N Z
LDX #$3A	LoaD X-Register	X-Register direkt laden	$A2	unmittelbar	Einen festen Wert direkt in das X-Register schreiben.	2	2		N Z
LDX $082D		X-Register absolut laden	$AE	absolut	Wert von der angegebenen Adresse in das X-Register schreiben.	3	4		N Z
LDX $082D,Y		X-Register absolut Y-indiziert laden	$BE	absolut Y-indiziert	Wert von der angegebenen Adresse+Y-Register in das X-Register schreiben.	3	4-5	Beim Überschreiten der Page-Grenze werden 5 Taktzyklen benötigt, sonst 4.	N Z
LDX $FB		X-Register von der Zero-Page laden	$A6	Zero-Page	Wert von der angegebenen Zero-Page-Adresse in das X-Register schreiben.	2	3		N Z
LDX $FB,Y		X-Register von Zero-Page Y-indiziert laden	$B6	Zero-Page Y-indiziert	Wert von der angegebenen Zero-Page-Adresse+Y-Register in das X-Register schreiben.	2	4		N Z
LDY #$3A	LoaD Y-Register	Y-Register direkt laden	$A0	unmittelbar	Einen festen Wert direkt ins Y-Register schreiben.	2	2		N Z
LDY $082D		Y-Register absolut laden	$AC	absolut	Wert von der angegebenen Adresse in das Y-Register schreiben.	3	4		N Z
LDY $082D,X		Y-Register absolut X-indiziert laden	$BC	absolut X-indiziert	Wert von der angegebenen Adresse+X-Register in das Y-Register schreiben.	3	4-5	Beim Überschreiten der Page-Grenze werden 5 Taktzyklen benötigt, sonst 4.	N Z
LDY $FB		Y-Register von der Zero-Page laden	$A4	Zero-Page	Wert von der angegebenen Zero-Page-Adresse in das Y-Register schreiben.	2	3		N Z
LDY $FB,X		Y-Register von Zero-Page X-indiziert laden	$B4	Zero-Page X-indiziert	Wert von der angegebenen Zero-Page-Adresse+X-Register in das Y-Register schreiben.	2	4		N Z
Speicherbefehle									
STA $082D	STore Akkumulator	Akkumulator an absoluter Adresse speichern	$8D	absolut	Den Inhalt des Akkus in der angegebenen Speicherstelle speichern.	3	4		
STA $03FF,X		Akkumulator absolut mit X speichern	$9D	absolut, X	Den Inhalt des Akkus in der angegebenen Speicherstelle unter hinzunahme des X-Registers speichern.	3	5		
STA $03FF,Y		Akkumulator absolut mit Y speichern	$99	absolut, Y	Den Inhalt des Akkus in der angegebenen Speicherstelle unter hinzunahme des Y-Registers speichern.	3	5		
STA $FB		Akkumulator auf der Zero-Page speichern	$85	Zero-Page	Den Inhalt des Akkus in der angegebenen Speicherstelle der Zero-Page speichern.	2	3		
STA $FB,X		Akkumulator auf der Zero-Page mit X speichern	$95	Zero-Page, X	Den Inhalt des Akkus in der angegebenen Speicherstelle der Zero-Page-Adresse plus X-Register speichern.	2	4		
STA ($FB,X)		Akkumulator indirekt x-indiziert speichern	$81	indirekt X-indiziert	Den Akku an der Adresse speichern, die an der (angegebener Zero-Page Adresse + X-Register) steht.	2	4		
STA ($FB),Y		Akku indirekt Y-nach-indiziert speichern	$91	indirekt Y-nach-indiziert	Den Inhalt des Akkus in der Adresse (die sich an der angegebenen Zero-Page-Adresse befindet)+Y-Register speichern.	2	6		
STX $082D	STore X-Register	X-Register an absoluter Adresse speichern	$8E	absolut	Den Inhalt des X-Register an der angegebenen Speicherstelle speichern.	3	4		
STX $FB		X-Register auf der Zero-Page speichern	$86	Zero-Page	Den Inhalt des X-Registers in der angegebenen Speicherstelle der Zero-Page speichern.	2	3		
STX $FB,Y		X-Register auf der Zero-Page mit Y speichern	$96	Zero-Page, Y	Den Inhalt des X-Registers in der angegebenen Speicherstelle der Zero-Page-Adresse plus Y-Register speichern.	2	4		
STY $082D	STore Y-Register	Y-Register an absoluter Adresse speichern	$8C	absolut	Den Inhalt des Y-Register an der angegebenen Speicherstelle speichern.	3	4		
STY $FB		Y-Register auf der Zero-Page speichern	$84	Zero-Page	Den Inhalt des Y-Registers in der angegebenen Speicherstelle der Zero-Page speichern.	2	3		
STY $FB,X		Y-Register auf der Zero-Page mit X speichern	$94	Zero-Page, X	Den Inhalt des Y-Registers in der angegebenen Speicherstelle der Zero-Page-Adresse plus X-Register speichern.	2	4		
Transferbefehle									
TAX	Transfer Akku to X-Register	Akku ins X-Register kopieren	$AA	implizit	Den Inhalt des Akkumulator in das X-Register kopieren.	1	2		N Z
TXA	Transfer X-Register to Akku	X-Register in den Akku kopieren	$8A	implizit	Den Inhalt des X-Registers in den Akkumulator kopieren.	1	2		N Z
TAY	Transfer Akku to Y-Register	Akku ins Y-Register kopieren	$A8	implizit	Den Inhalt des Akkumulator in das Y-Register kopieren.	1	2		N Z
TYA	Transfer Y-Register to Akku	Y-Register in den Akku kopieren	$98	implizit	Den Inhalt des Y-Registers in den Akkumulator kopieren.	1	2		N Z
TSX	Transfer SP to X-Register	Stackpointer ins X-Register kopieren	$BA	implizit	Den Inhalt des Stackpointers (SP) in das X-Register kopieren.	1	2		N Z
TXS	Transfer X-Register to SP	X-Register in den Stackpointer kopieren	$9A	implizit	Den Inhalt des X-Registers in den Stackpointer (SP) kopieren.	1	2		

Befehl	Mnemonics Erklärung	Beschreibung		Adressierungsart	Beschreibung Adressierungsart	Bytelänge	Taktzyklen		beinflußte Register
Rechenbefehle									
ADC #$3B	**AD**d with **C**arry	Addiere zu Akku...	$69	unmittelbar	Den angegebenen Wert zum Akku hinzuaddieren.	2	2		N V Z C
ADC $082E		Addiere zu Akku...	$6D	absolut	Den Wert an der angegebenen absoluten Adresse zum Akku hinzuaddieren.	3	4		N V Z C
ADC $082E,X		Addiere zu Akku...	$7D	absolut X-indiziert	Den Wert an der angegebenen Adresse + X-Register zum Akku hinzuaddieren.	3	4-5	Beim Überschreiten der Page-Grenze werden 5 Taktzyklen benötigt, sonst 4.	N V Z C
ADC $082E,Y		Addiere zu Akku...	$79	absolut X-indiziert	Den Wert an der angegebenen Adresse + Y-Register zum Akku hinzuaddieren.	3	4-5	Beim Überschreiten der Page-Grenze werden 5 Taktzyklen benötigt, sonst 4.	N V Z C
ADC $FB		Addiere zu Akku...	$65	Zero-Page	Den Wert an der angegebenen Zero-Page-Adresse zum Akku hinzuaddieren.	2	3		N V Z C
ADC $FB,X		Addiere zu Akku...	$75	Zero-Page X-indiziert	Den Wert an der angegebenen Zero-Page-Adresse + X-Register zum Akku hinzuaddieren.	2	4		N V Z C
ADC ($FB,X)		Addiere zu Akku...	$61	indirekt X-indiziert	Den Wert, der Adresse, an der angegebenen Zero-Page-Adresse + X-Register, zu finden ist, zum Akku hinzuaddieren.	2	6		N V Z C
ADC ($FB),Y		Addiere zu Akku...	$71	indirekt Y-nach-indiziert	Den Wert, der an der Adresse, die an der angegebenen Zero-Page-Adresse liegt, zzgl. Y-Register zu finden ist, zum Akku hinzuaddieren.	2	5-6	Beim Überschreiten der Page-Grenze werden 6 Taktzyklen benötigt, sonst 5.	N V Z C
SBC #$3B	**SU**btract with **C**arry	Subtrahiere vom Akku...	$E9	unmittelbar	Den angegebenen Wert vom Akku subtrahieren.	2	2		N V Z C
SBC $082E		Subtrahiere vom Akku...	$ED	absolut	Den Wert an der angegebenen absoluten Adresse vom Akku subtrahieren.	3	4		N V Z C
SBC $082E,X		Subtrahiere vom Akku...	$FD	absolut X-indiziert	Den Wert an der angegebenen Adresse + X-Register vom Akku subtrahieren.	3	4-5	Beim Überschreiten der Page-Grenze werden 5 Taktzyklen benötigt, sonst 4.	N V Z C
SBC $082E,Y		Subtrahiere vom Akku...	$F9	absolut X-indiziert	Den Wert an der angegebenen Adresse + Y-Register vom Akku subtrahieren.	3	4-5	Beim Überschreiten der Page-Grenze werden 5 Taktzyklen benötigt, sonst 4.	N V Z C
SBC $FB		Subtrahiere vom Akku...	$E5	Zero-Page	Den Wert an der angegebenen Zero-Page-Adresse vom Akku subtrahieren.	2	3		N V Z C
SBC $FB,X		Subtrahiere vom Akku...	$F5	Zero-Page X-indiziert	Den Wert an der angegebenen Zero-Page-Adresse + X-Register vom Akku subtrahieren.	2	4		N V Z C
SBC ($FB,X)		Subtrahiere vom Akku...	$E1	indirekt X-indiziert	Den Wert, der Adresse, an der angegebenen Zero-Page-Adresse + X-Register, zu finden ist, vom Akku subtrahieren.	2	6		N V Z C
SBC ($FB),Y		Subtrahiere vom Akku...	$F1	indirekt Y-nach-indiziert	Den Wert, der an der Adresse, die an der angegebenen Zero-Page-Adresse liegt, zzgl. Y-Register zu finden ist, vom Akku subtrahieren.	2	5-6	Beim Überschreiten der Page-Grenze werden 6 Taktzyklen benötigt, sonst 5.	N V Z C
DEC $082D	**DEC**rement	absolute Speicherstelle verringern	$CE	absolut	Den Inhalt der angegebenen Adresse um 1 verringern.	3	6		N Z
DEC $082D,X		absolute Adresse, X-indiziert verringern	$DE	absolut, X	Den Inhalt (der angegebenen Adresse+X-Register) um 1 verringern.	3	7		N Z
DEC $FB		Zero-Page-Speicherstelle verringern	$C6	Zero-Page	Den Inhalt der angegebenen Zero-Page-Adresse um 1 verringern.	2	5		N Z
DEC $FB,X		Zero-Page X-indiziert verringern	$D6	Zero-Page, X	Den Inhalt (der angegebenen Zero-Page-Adresse+X-Register) um 1 verringern.	2	6		N Z
DEX	**DE**crement **X**-Register	X-Register verringern	$CA	implizit	Den Inhalt des X-Registers um 1 verringern.	1	2		N Z
DEY	**DE**crement **Y**-Register	Y-Register verringern	$88	implizit	Den Inhalt des Y-Registers um 1 verringern.	1	2		N Z
INC $082D	**INC**rement	absolute Speicherstelle erhöhen	$EE	absolut	Den Inhalt der angegebenen Adresse um 1 erhöhen.	3	6		N Z
INC $082D,X		absolute Adresse, X-indiziert erhöhen	$FE	absolut, X	Den Inhalt (der angegebenen Adresse+X-Register) um 1 erhöhen.	3	7		N Z
INC $FB		Zero-Page-Speicherstelle erhöhen	$E6	Zero-Page	Den Inhalt der angegebenen Zero-Page-Adresse um 1 erhöhen.	2	5		N Z
INC $FB,X		Zero-Page X-indiziert erhöhen	$F6	Zero-Page, X	Den Inhalt (der angegebenen Zero-Page-Adresse+X-Register) um 1 erhöhen.	2	6		N Z
INX	**IN**crement **X**-Register	X-Register erhöhen	$E8	implizit	Den Inhalt des X-Registers um 1 erhöhen.	1	2		N Z
INY	**IN**crement **Y**-Register	Y-Register erhöhen	$C8	implizit	Den Inhalt des Y-Registers um 1 erhöhen.	1	2		N Z
Rotations- / Verschiebebefehle									
ASL	**A**rithmetic **S**hift **L**eft	Den Akku-Inhalt bitweise nach links verschieben.	$0A	Akku	Verschiebt den Akku-Inhalt bitweise nach links. Von rechts wird mit einer Null aufgefüllt, das links herausfallende Bit landet im Carry-Flag.	1	2		N Z C
ASL $082D		Den Inhalt des Bytes an der absoluten Adresse bitweise nach links verschieben.	$0E	absolut	Verschiebt das Byte an der absoluten Adresse bitweise nach links. Von rechts wird mit einer Null aufgefüllt, das links herausfallende Bit landet im Carry-Flag.	3	6		N Z C
ASL $082D,X		Den Inhalt des Bytes an der absoluten Adresse + X bitweise nach links verschieben.	$1E	absolut X-indiziert	Verschiebt das Byte an der absoluten Adresse + X bitweise nach links. Von rechts wird mit einer Null aufgefüllt, das links herausfallende Bit landet im Carry-Flag.	3	7		N Z C
ASL $FB		Den Inhalt des Bytes an der Zero-Page-Adresse bitweise nach links verschieben.	$06	Zero-Page	Verschiebt das Byte an der Zero-Page-Adresse bitweise nach links. Von rechts wird mit einer Null aufgefüllt, das links herausfallende Bit landet im Carry-Flag.	2	5		N Z C
ASL $FB,X		Den Inhalt des Bytes an der Zero-Page-Adresse + X bitweise nach links verschieben.	$16	Zero-Page X-indiziert	Verschiebt das Byte an der Zero-Page-Adresse + X bitweise nach links. Von rechts wird mit einer Null aufgefüllt, das links herausfallende Bit landet im Carry-Flag.	2	6		N Z C
LSR	**L**ogical **S**hift **R**ight	Den Akku-Inhalt bitweise nach rechts verschieben.	$4A	Akku	Verschiebt den Akku-Inhalt bitweise nach rechts. Von links wird mit einer Null aufgefüllt, das rechts herausfallende Bit landet im Carry-Flag.	1	2		N Z C
LSR $082D		Den Inhalt des Bytes an der absoluten Adresse bitweise nach rechts verschieben.	$4E	absolut	Verschiebt das Byte an der absoluten Adresse bitweise nach rechts. Von links wird mit einer Null aufgefüllt, das rechts herausfallende Bit landet im Carry-Flag.	3	6		N Z C
LSR $082D,X		Den Inhalt des Bytes an der absoluten Adresse + X bitweise nach rechts verschieben.	$5E	absolut X-indiziert	Verschiebt das Byte an der absoluten Adresse + X bitweise nach rechts. Von links wird mit einer Null aufgefüllt, das rechts herausfallende Bit landet im Carry-Flag.	3	7		N Z C

Befehl	Mnemonics Erklärung	Beschreibung		Adressierungsart	Beschreibung Adressierungsart	Bytelänge	Taktzyklen		beinflußte Register
LSR $FB		Den Inhalt des Bytes an der Zero-Page-Adresse bitweise nach rechts verschieben.	$46	Zero-Page	Verschiebt das Byte an der Zero-Page-Adresse bitweise nach rechts. Von links wird mit einer Null aufgefüllt, das rechts herausfallende Bit landet im Carry-Flag.	2	5		N Z C
LSR $FB,X		Den Inhalt des Bytes an der Zero-Page-Adresse + X bitweise nach rechts verschieben.	$56	Zero-Page X-indiziert	Verschiebt das Byte an der Zero-Page-Adresse + X bitweise nach rechts. Von links wird mit einer Null aufgefüllt, das rechts herausfallende Bit landet im Carry-Flag.	2	6		N Z C
ROL	**RO**tate **L**eft	Den Akku-Inhalt bitweise nach links verschieben, von rechts wird das Carry-Flag eingeschoben.	$2A	Akku	ROL Verschiebt den Akku-Inhalt bitweise nach links. Von rechts wird mit dem Carry-Flag aufgefüllt, das links herausfallende Bit landet zum Schluß wieder im Carry-Flag.	1	2		N Z C
ROL $082D		Den Inhalt des Bytes an der absoluten Adresse bitweise nach links verschieben, von rechts das C-Flag einschieben.	$2E	Akku	ROL Bitweises verschieben nach links. Von rechts wird mit dem Carry-Flag aufgefüllt, das links herausfallende Bit landet zum Schluß wieder im Carry-Flag.	3	6		N Z C
ROL $082D,X		Den Inhalt des Bytes an der absoluten Adresse+X bitweise nach links verschieben und von rechts das C-Flag einschieben.	$3E	Akku	ROL Bitweises verschieben nach links. Von rechts wird mit dem Carry-Flag aufgefüllt, das links herausfallende Bit landet zum Schluß wieder im Carry-Flag.	3	7		N Z C
ROL $FB		Den Inhalt des Bytes an der Zero-Page-Adresse bitweise nach links verschieben und von rechts das Carry-Flag einschieben.	$26	Akku	ROL Bitweises verschieben nach links. Von rechts wird mit dem Carry-Flag aufgefüllt, das links herausfallende Bit landet zum Schluß wieder im Carry-Flag.	2	5		N Z C
ROL $FB,X		Den Inhalt des Bytes an der Zero-Page-Adresse+X bitweise nach links verschieben und das C-Flag von rechts einschieben.	$36	Akku	ROL Bitweises verschieben nach links. Von rechts wird mit dem Carry-Flag aufgefüllt, das links herausfallende Bit landet zum Schluß wieder im Carry-Flag.	2	6		N Z C
ROR	**RO**tate **R**ight	Den Akku-Inhalt bitweise nach rechts verschieben und das Carry-Flag von links einschieben.	$6A	Akku	Verschiebt den Akku-Inhalt bitweise nach rechts. Von links wird mit dem Carry-Flag aufgefüllt, das rechts herausfallende Bit landet zum Schluß wieder im Carry-Flag.	1	2		N Z C
ROR $082D		Den Inhalt des Bytes an der absoluten Adresse bitweise nach rechts verschieben und das Carry-Flag von links einschieben.	$6E	Akku	ROR Bitweises verschieben nach rechts. Von links wird mit dem Carry-Flag aufgefüllt, das rechts herausfallende Bit landet zum Schluß wieder im Carry-Flag.	3	6		N Z C
ROR $082D,X		Den Inhalt des Bytes an der absoluten Adresse+X bitweise nach rechts verschieben, von links das C-Flag einschieben.	$7E	Akku	ROR Bitweises verschieben nach rechts. Von links wird mit dem Carry-Flag aufgefüllt, das rechts herausfallende Bit landet zum Schluß wieder im Carry-Flag.	3	7		N Z C
ROR $FB		Den Inhalt des Bytes an der Zero-Page-Adresse bitweise nach rechts verschieben und von links das Carry-Flag einschieben.	$66	Akku	ROR Bitweises verschieben nach rechts. Von links wird mit dem Carry-Flag aufgefüllt, das rechts herausfallende Bit landet zum Schluß wieder im Carry-Flag.	2	5		N Z C
ROR $FB,X		Den Inhalt des Bytes an der Zero-Page-Adresse+X bitweise nach rechts verschieben, das C-Flag von links einschieben.	$76	Akku	ROR Bitweises verschieben nach rechts. Von links wird mit dem Carry-Flag aufgefüllt, das rechts herausfallende Bit landet zum Schluß wieder im Carry-Flag.	2	6		N Z C
Boolschebefehle									
AND #$3A	**AND**	direkt bitweises UND mit dem Akku	$29	direkt	Den Inhalt des Akkus mit dem angegebenen Wert bitweise Verknüpfen und das Ergebnis im Akku ablegen.	2	2		N Z
AND $083A		bitweises UND mit der absoluten Adresse und dem Akku	$2D	absolut	Den Inhalt des Akkus mit dem Byte an der angegebenen Adresse bitweise UND-Verknüpfen und das Ergebnis im Akku ablegen.	3	4		N Z
AND $083A,X		bitweises UND mit der absoluten Adresse+X und dem Akku	$3D	absolut, X	Den Inhalt des Akkus mit dem Byte an der angegebenen Adresse+X-Register bitweise UND-Verknüpfen und das Ergebnis im Akku ablegen.	3	4-5	Normalerweise werden 4 Taktzyklen benötigt, findet eine Page-Überschreitung statt sind es 5!	N Z
AND $083A,Y		bitweises UND mit der absoluten Adresse+Y und dem Akku	$39	absolut, Y	Den Inhalt des Akkus mit dem Byte an der angegebenen Adresse+Y-Register bitweise UND-Verknüpfen und das Ergebnis im Akku ablegen.	3	4-5	Normalerweise werden 4 Taktzyklen benötigt, findet eine Page-Überschreitung statt sind es 5!	N Z
AND $FB		bitweises UND mit der Zero-Page-Adresse und dem Akku	$25	Zero-Page	Den Inhalt des Akkus mit dem Byte an der angegebenen Zero-Page-Adresse bitweise UND-Verknüpfen und das Ergebnis im Akku ablegen.	2	3		N Z
AND $FB,X		bitweises UND mit der Zero-Page-Adresse+X und dem Akku	$35	Zero-Page, X	Den Inhalt des Akkus mit dem Byte an (der angegebenen Zero-Page-Adresse + X-Register) bitweise UND-Verknüpfen und das Ergebnis im Akku ablegen.	2	4		N Z
AND ($FB,X)		bitweises UND mit der indirekt+X-Adresse und dem Akku	$21	indirekt, X-indiziert	Den Inhalt des Akkus mit dem Byte, das an der Adresse (angegebenen Zero-Page-Adresse + X-Register) liegt, bitweise UND-Verknüpfen und das Ergebnis im Akku ablegen.	2	6		N Z
AND ($FB),Y		bitweises UND mit der indirekt-Y-nach-indizierten-Adresse und dem Akku	$31	indirekt, Y-nach-indiziert	Den Inhalt des Akkus mit dem Byte, das an (der Adresse die an (der angegebenen Zero-Page-Adresse zufinden ist) + Y-Register) liegt, bitweise UND-Verknüpfen und das Ergebnis im Akku ablegen.	2	5-6	Normalerweise werden 5 Taktzyklen benötigt, findet eine Page-Überschreitung statt sind es 6!	N Z
EOR #$3A	**E**xclusive **OR**	direkt bitweises EXKLUSIV ODER mit dem Akku	$49	direkt	Den Inhalt des Akkus mit dem angegebenen Wert bitweise Verknüpfen und das Ergebnis im Akku ablegen.	2	2		N Z
EOR $083A		bitweises EXKLUSIV ODER mit der absoluten Adresse und dem Akku	$4D	absolut	Den Inhalt des Akkus mit dem Byte an der angegebenen Adresse bitweise EXCLUSIV-ODER-Verknüpfen und das Ergebnis im Akku ablegen.	3	4		N Z
EOR $083A,X		bitweises EXKLUSIV ODER mit der absoluten Adresse+X und dem Akku	$5D	absolut, X	Den Inhalt des Akkus mit dem Byte an der angegebenen Adresse+X-Register bitweise EXKLUSIV-ODER-Verknüpfen und das Ergebnis im Akku ablegen.	3	4-5	Normalerweise werden 4 Taktzyklen benötigt, findet eine Page-Überschreitung statt sind es 5!	N Z
EOR $083A,Y		bitweises EXKLUSIV ODER mit der absoluten Adresse+Y und dem Akku	$59	absolut, Y	Den Inhalt des Akkus mit dem Byte an der angegebenen Adresse+Y-Register bitweise EXKLUSIV-ODER-Verknüpfen und das Ergebnis im Akku ablegen.	3	4-5	Normalerweise werden 4 Taktzyklen benötigt, findet eine Page-Überschreitung statt sind es 5!	N Z
EOR $FB		bitweises EXKLUSIV ODER mit der Zero-Page-Adresse und dem Akku	$45	Zero-Page	Den Inhalt des Akkus mit dem Byte an der angegebenen Zero-Page-Adresse bitweise EXKLUSIV-ODER-Verknüpfen und das Ergebnis im Akku ablegen.	2	3		N Z
EOR $FB,X		bitweises EXKLUSIV ODER mit der Zero-Page-Adresse+X und dem Akku	$55	Zero-Page, X	Den Inhalt des Akkus mit dem Byte an (der angegebenen Zero-Page-Adresse + X-Register) bitweise EXKLUSIV-ODER-Verknüpfen und das Ergebnis im Akku ablegen.	2	4		N Z

Befehl	Mnemonics Erklärung	Beschreibung		Adressierungsart	Beschreibung Adressierungsart	Bytelänge	Taktzyklen		beinflußte Register
EOR ($FB,X)		bitweises EXKLUSIV ODER mit der indirekt+X-Adresse und dem Akku	$41	indirekt, X-indiziert	Den Inhalt des Akkus mit dem Byte, das an der Adresse (angegebenen Zero-Page-Adresse + X-Register) liegt, bitweise EXKLUSIV-ODER-Verknüpfen und das Ergebnis im Akku ablegen.	2	6		N Z
EOR ($FB),Y		bitweises EXKLUSIV ODER mit der indirekt-Y-nach-indizierten-Adresse und dem Akku	$51	indirekt, Y-nach-indiziert	Den Inhalt des Akkus mit dem Byte, das an (der Adresse die an (der angegebenen Zero-Page-Adresse zufinden ist) + Y-Register) liegt, bitweise EXKLUSIV-ODER-Verknüpfen und das Ergebnis im Akku ablegen.	2	5-6	Normalerweise werden 5 Taktzyklen benötigt, findet eine Page-Überschreitung statt sind es 6!	N Z
ORA #$3A	**OR** with **A**kku	direkt bitweises ODER mit dem Akku	$09	direkt	Den Inhalt des Akkus mit dem angegebenen Wert bitweise Verknüpfen und das Ergebnis im Akku ablegen.	2	2		N Z
ORA $083A		bitweises ODER mit der absoluten Adresse und dem Akku	$0D	absolut	Den Inhalt des Akkus mit dem Byte an der angegebenen Adresse bitweise ODER-Verknüpfen und das Ergebnis im Akku ablegen.	3	4		N Z
ORA $083A,X		bitweises ODER mit der absoluten Adresse+X und dem Akku	$1D	absolut, X	Den Inhalt des Akkus mit dem Byte an der angegebenen Adresse+X-Register bitweise ODER-Verknüpfen und das Ergebnis im Akku ablegen.	3	4-5	Normalerweise werden 4 Taktzyklen benötigt, findet eine Page-Überschreitung statt sind es 5!	N Z
ORA $083A,Y		bitweises ODER mit der absoluten Adresse+Y und dem Akku	$19	absolut, Y	Den Inhalt des Akkus mit dem Byte an der angegebenen Adresse+Y-Register bitweise ODER-Verknüpfen und das Ergebnis im Akku ablegen.	3	4-5	Normalerweise werden 4 Taktzyklen benötigt, findet eine Page-Überschreitung statt sind es 5!	N Z
ORA $FB		bitweises ODER mit der Zero-Page-Adresse und dem Akku	$05	Zero-Page	Den Inhalt des Akkus mit dem Byte an der angegebenen Zero-Page-Adresse bitweise ODER-Verknüpfen und das Ergebnis im Akku ablegen.	2	3		N Z
ORA $FB,X		bitweises ODER mit der Zero-Page-Adresse+X und dem Akku	$15	Zero-Page, X	Den Inhalt des Akkus mit dem Byte an (der angegebenen Zero-Page-Adresse + X-Register) bitweise ODER-Verknüpfen und das Ergebnis im Akku ablegen.	2	4		N Z
ORA ($FB,X)		bitweises ODER mit der indirekt+X-Adresse und dem Akku	$01	indirekt, X-indiziert	Den Inhalt des Akkus mit dem Byte, das an der Adresse (angegebenen Zero-Page-Adresse + X-Register) liegt, bitweise ODER-Verknüpfen und das Ergebnis im Akku ablegen.	2	6		N Z
ORA ($FB),Y		bitweises ODER mit der indirekt-Y-nach-indizierten-Adresse und dem Akku	$11	indirekt, Y-nach-indiziert	Den Inhalt des Akkus mit dem Byte, das an (der Adresse die an (der angegebenen Zero-Page-Adresse zufinden ist) + Y-Register) liegt, bitweise ODER-Verknüpfen und das Ergebnis im Akku ablegen.	2	5-6	Normalerweise werden 5 Taktzyklen benötigt, findet eine Page-Überschreitung statt sind es 6!	N Z
Vergleichsbefehle									
BIT$0821	**BIT** test	bitweises UND zwischen Akku und der absoluten Adresse	$2C	absolut	Den Wert im Akku und an der angegebenen Adresse bitweise UND-Verknüpfen. OHNE ein Ergebnis zu speichern, es werden nur die Flags gesetzt!	3	4		N V Z
BIT $FB		bitweises UND zwischen Akku und der Zero-Page-Adresse	$24	Zero-Page	Den Wert im Akku und an der angegebenen Zero-Page-Adresse bitweise UND-Verknüpfen. OHNE ein Ergebnis zu speichern, es werden nur die Flags gesetzt!	2	3		N V Z
CMP #$3C	**C**o**MP**are	Vergleiche Akku mit...	$C9	unmittelbar	Den Akku mit dem angegebenen Wert vergleichen!	2	2		N Z C
CMP $0821		Vergleiche Akku mit...	$CD	absolut	Den Akku mit dem an der absoluten Adresse befindlichen Wert vergleichen!	3	4		N Z C
CMP $0821,X		Vergleiche Akku mit...	$DD	absolut X-indiziert	Den Akku mit dem an der absoluten Adresse + X-Register befindlichen Wert vergleichen!	3	4-5	Beim Überschreiten der Page-Grenze werden 5 Taktzyklen benötigt, sonst 4.	N Z C
CMP $0821,Y		Vergleiche Akku mit...	$D9	absolut Y-indiziert	Den Akku mit dem an der absoluten Adresse + Y-Register befindlichen Wert vergleichen!	3	4-5	Beim Überschreiten der Page-Grenze werden 5 Taktzyklen benötigt, sonst 4.	N Z C
CMP $FB		Vergleiche Akku mit...	$C5	Zero-Page	Den Akku mit dem an der Zero-Page-Adresse befindlichen Wert vergleichen!	2	3		N Z C
CMP $FB,X		Vergleiche Akku mit...	$D5	Zero-Page X-indiziert	Den Akku mit dem an der Zero-Page-Adresse + X-Register befindlichen Wert vergleichen!	2	4		N Z C
CMP ($FB,X)		Vergleiche Akku mit...	$C1	indirekt X-indiziert	Den Akku mit dem Wert vergleichen, der an der Adresse zu finden ist, die an der Zero-Page-Adresse + X-Register liegt!	2	6		N Z C
CMP ($FB),Y		Vergleiche Akku mit...	$D1	indirekt Y-nach-indiziert	Den Akku mit dem Wert vergleichen, den man an der Adresse+Y-Register findet, wobei diese Adresse wiederum an der angegebenen Zero-Page-Adresse liegt.	2	5-6	Beim Überschreiten der Page-Grenze werden 6 Taktzyklen benötigt, sonst 5.	N Z C
CPX #$3C	**C**om**P**are **X**-Register	Vergleiche X-Register mit...	$E0	unmittelbar	Das X-Register mit dem angegebenen Wert vergleichen!	2	2		N Z C
CPX $0821		Vergleiche X-Register mit...	$EC	absolut	Das X-Register mit dem an der absoluten Adresse befindlichen Wert vergleichen!	3	4		N Z C
CPX $FB		Vergleiche X-Register mit...	$E4	Zero-Page	Das X-Register mit dem an der Zero-Page-Adresse befindlichen Wert vergleichen!	2	3		N Z C
CPY#$3C	**C**om**P**are **Y**-Register	Vergleiche Y-Register mit...	$C0	unmittelbar	Das Y-Register mit dem angegebenen Wert vergleichen!	2	2		N Z C
CPY $0821		Vergleiche Y-Register mit...	$CC	absolut	Das Y-Register mit dem an der absoluten Adresse befindlichen Wert vergleichen!	3	4		N Z C
CPY $FB		Vergleiche Y-Register mit...	$C4	Zero-Page	Das Y-Register mit dem an der Zero-Page-Adresse befindlichen Wert vergleichen!	2	3		N Z C
Sprungbefehle									
BCC $080D	**B**ranch on **C**arry **C**lear	Springe wenn das Carry-Flag gelöscht ist.	$90	relativ	Wenn C=0, springe zur angegebenen Adresse. Da es sich um eine relative Adressierung handelt, darf das Ziel max. +127 bzw. -128 Bytes entfernt liegen!	2	2-4	2 Taktzyklen werden benötigt, wenn nicht gesprungen werden muss, bei einem Sprung sind es 3 und muss über die Page-Grenze gesprungen werden sogar 4 Taktzyklen.	prüft C=0
BCS $080D	**B**ranch on **C**arry **S**et	Springe wenn das Carry-Flag gesetzt ist.	$B0	relativ	Wenn C=1, springe zur angegebenen Adresse. Da es sich um eine relative Adressierung handelt, darf das Ziel max. +127 bzw. -128 Bytes entfernt liegen!	2	2-4	2 Taktzyklen werden benötigt, wenn nicht gesprungen werden muss, bei einem Sprung sind es 3 und muss über die Page-Grenze gesprungen werden sogar 4 Taktzyklen.	prüft C=1
BVC $080D	**B**ranch on o**V**erflow **C**lear	Springe wenn das Overflow-Flag gelöscht ist.	$50	relativ	Wenn V=0, springe zur angegebenen Adresse. Da es sich um eine relative Adressierung handelt, darf das Ziel max. +127 bzw. -128 Bytes entfernt liegen!	2	2-4	2 Taktzyklen werden benötigt, wenn nicht gesprungen werden muss, bei einem Sprung sind es 3 und muss über die Page-Grenze gesprungen werden sogar 4 Taktzyklen.	prüft V=0

Befehl	Mnemonics Erklärung	Beschreibung		Adressierungsart	Beschreibung Adressierungsart	Bytelänge	Taktzyklen		beinflußte Register
BVS $080D	**B**ranch on o**V**erflow **S**et	Springe wenn das Overflow-Flag gesetzt ist.	$70	relativ	Wenn V=1, springe zur angegebenen Adresse. Da es sich um eine relative Adressierung handelt, darf das Ziel max. +127 bzw. -128 Bytes entfernt liegen!	2	2-4	2 Taktzyklen werden benötigt, wenn nicht gesprungen werden muss, bei einem Sprung sind es 3 und muss über die Page-Grenze gesprungen werden sogar 4 Takzyklen.	prüft V=1
BEQ $080D	**B**ranch on **EQ**ual	Springe wenn gleich	$F0	relativ	Wenn gleich, springe zur angegebenen Adresse. Da es sich um eine relative Adressierung handelt, darf das Ziel max. +127 bzw. -128 Bytes entfernt liegen!	2	2-4	2 Taktzyklen werden benötigt, wenn nicht gesprungen werden muss, bei einem Sprung sind es 3 und muss über die Page-Grenze gesprungen werden sogar 4 Takzyklen.	prüft Z=1
BNE $080D	**B**ranch on **N**ot **E**qual	Springe wenn ungleich	$D0	relativ	Wenn ungleich, springe zur angegebenen Adresse. Da es sich um eine relative Adressierung handelt, darf das Ziel max. +127 bzw. -128 Bytes entfernt liegen!	2	2-4	2 Taktzyklen werden benötigt, wenn nicht gesprungen werden muss, bei einem Sprung sind es 3 und muss über die Page-Grenze gesprungen werden sogar 4 Takzyklen.	prüft Z=0
BPL $080D	**B**ranch on **PL**us	Springe wenn positiv	$10	relativ	Wenn positiv, springe zur angegebenen Adresse. Da es sich um eine relative Adressierung handelt, darf das Ziel max. +127 bzw. -128 Bytes entfernt liegen!	2	2-4	2 Taktzyklen werden benötigt, wenn nicht gesprungen werden muss, bei einem Sprung sind es 3 und muss über die Page-Grenze gesprungen werden sogar 4 Takzyklen.	prüft N=0
BMI $080D	**B**ranch on **Mi**nus	Springe wenn negativ	$30	relativ	Wenn negativ, springe zur angegebenen Adresse. Da es sich um eine relative Adressierung handelt, darf das Ziel max. +127 bzw. -128 Bytes entfernt liegen!	2	2-4	2 Taktzyklen werden benötigt, wenn nicht gesprungen werden muss, bei einem Sprung sind es 3 und muss über die Page-Grenze gesprungen werden sogar 4 Takzyklen.	prüft N=1
JMP $080D	**J**u**MP** to	Springe nach	$4C	absolut	Springe direkt zur angegebenen Adresse	3	3		
JMP ($08E2)		Springe indirekt nach	$6C	indirekt	Springe zur Adresse, die an der angegebenen Adresse gespeichert ist.	3	6	Gebt ihr etwas wie ($C0FF) beim JMP-Befehl an, dann führt euer Sprung nicht zum gewünschten Ziel! Da eine Adresse zwei Byte benötigt verteilt sich diese hier über eine Page-Grenze. Wenn das der Fall ist, wird das LSB zwar vom $C0FF genommen. Aber das MSB sucht der Przessor an $C000	
JSR $080D	**J**ump to **SU**broutine	Springe zum Unterprogramm	$20	implizit	Springt zum Unterprogramm an der angegebenen Stelle. Dabei wir die Rücksprungadresse für RTS auf dem Stack gespeichert.	3	6		
RTS	**R**e**T**urn from **S**ubroutine	Springe zurück zum Aufrufer	$60	implizit	Springt zur aktuellen Adresse, die auf dem Stack liegt.	1	6		
RTI	**R**e**T**urn from **I**nterrupt	Interrupt verlassen	$40	implizit	Beendet die aktuelle Interrupt-Routine, holt die Statusregister vom Stack und setzt das Programm an der auf dem Stack hinterlegten Adresse fort.	1	6		holt SR vom Stack
Stackbefehle									
PHA	**P**us**H** **A**kku on stack	Akku-Inhalt auf den Stack legen	$48	implizit	Kopiert den Akku auf den Stack	1	3		
PHP	**P**us**H** **P**rocessor statusregister on stack	Statusregister auf den Stack legen	$08	implizit	Kopiert das Statusregister (SR) auf den Stack	1	3		
PLA	**P**u**L**l **A**kku from stack	Akku-Inhalt vom Stack holen	$68	implizit	Kopiert das Byte von der aktuellen Stackposition in den Akku	1	4		N Z
PLP	**P**u**L**l **P**rocessor status from stack	Statusregister vom Stack holen	$28	implizit	Setzt die Flags im Statusregister (SR) mit den Bits aus dem Byte von der aktuellen Stackposition	1	4		ALLE!
Flag-Befehle									
CLC	**CL**ear **C**arry	Carry-Flag auf 0 setzen	$18	implizit	Das Carry-Flag auf 0 setzen.	1	2		C
CLD	**CL**ear **D**ecimal	Decimal-Flag auf 0 setzen	$D8	implizit	Das Decimal-Flag auf 0 setzen.	1	2		D
CLI	**CL**ear **I**nterrupt-Flag	Interrupt-Flag auf 0 setzen	$58	implizit	Um Interrupts wieder zu erlauben, I-Flag auf 0 setzen.	1	2		I
CLV	**CL**ear o**V**erflow-Flag	Overflow-Flag auf 0 setzen	$B8	implizit	Das V-Flag wieder auf 0 setzen.	1	2		V
SEC	**SE**t **C**arry	Carry-Flag auf 1 setzen	$38	implizit	Das Carry-Flag auf 1 setzen.	1	2		C
SED	**SE**t **D**ecimal	Decimal-Flag auf 1 setzen	$F8	implizit	Das Decimal-Flag auf 1 setzen.	1	2		D
SEI	**SE**t **I**nterrupt-Flag	Interrupt-Flag auf 1 setzen	$78	implizit	Um Interrupts zu verhindern, I-Flag auf 1 setzen.	1	2		I
Restliche Befehle									
BRK	**BR**ea**K**	Software-Interrupt auslösen	$00	implizit		1	7		B I
NOP	**N**o **OP**eration	Nichts machen	$EA	implizit		1	2		

Illigale-OpCodes

alternatives Mnemonic:	Befehl	Mnemonics Erklärung	Beschreibung		Adressierungsart	Beschreibung Adressierungsart	Bytelänge	Taktzyklen		beinflußte Register
ASR	ALR #$A5	AND+LSR	Führt per AND eine UND-Verknüpfung mit dem angegebenen Wert und dem Akku durch. Abschließend wird per LSR der Akku bitweise nach rechts verschoben.	$4B	unmittelbar		2	2		N Z C
	ARR #$A5	AND+ROR	Führt per AND eine UND-Verknüpfung mit dem angegebenen Wert und dem Akku durch. Abschließend wird per ROR der Akku bitweise nach rechts rotiert.	$6B	unmittelbar		2	2		N Z C
SLO	ASO $082E	ASL+ORA	Kombination von ASL + ORA. Dabei wird das über die Adressierung zufindene Byte zunächst per ASL bitweise nach links verschoben und anschließend mit dem Akku ODER-Verknüpft.	$0F	absolut		3	6		N Z C
	ASO $082E,X	ASL+ORA		$1F	absolut X-indiziert		3	7		N Z C
	ASO $082E,Y	ASL+ORA		$1B	absolut Y-indiziert		3	7		N Z C
	ASO $FB	ASL+ORA		$07	Zero-Page		2	5		N Z C
	ASO $FB,X	ASL+ORA		$17	Zero-Page X-indiziert		2	6		N Z C
	ASO ($FB,X)	ASL+ORA		$03	indirekt X-indiziert		2	8		N Z C
	ASO ($FB),Y	ASL+ORA		$13	indirekt Y-nach-indiziert		2	8		N Z C
SAX AAX	AXS $082E	Akku+XReg+Store	Die Inhalte von Akku und X-Register werden UND-Verknüpft, aber OHNE eines der beiden Register zu ändern! Das Ergbnis wird dann an der angegebenen Adresse abgelegt. Die Flags im Statusregister (SR) bleiben ebenfalls unverändert!	$8F	absolut		3	4		
	ASX $FB	Akku+XReg+Store		$87	Zero-Page		2	3		
	ASX $FB,X	Akku+XReg+Store		$97	Zero-Page X-indiziert		2	4		
	ASX ($FB,X)	Akku+XReg+Store		$83	indirekt X-indiziert		2	6		
DCP	DCM $082E	DEC+CMP	Verringert das Byte an der angegebenen Speicherstelle mit DEC und vergleicht dieses dann mittels CMP mit dem Akku.	$CF	absolut		3	6		N Z C
	DCM $082E,X	DEC+CMP		$1F	absolut X-indiziert		3	7		N Z C
	DCM $082E,Y	DEC+CMP		$1B	absolut Y-indiziert		3	7		N Z C
	DCM $FB	DEC+CMP		$07	Zero-Page		2	5		N Z C
	DCM $FB,X	DEC+CMP		$17	Zero-Page X-indiziert		2	6		N Z C
	DCM ($FB,X)	DEC+CMP		$03	indirekt X-indiziert		2	8		N Z C
	DCM ($FB),Y	DEC+CMP		$13	indirekt Y-nach-indiziert		2	8		N Z C
ISB ISC	INS $082E	INC+SBC	Erhöht das Byte an der angegebenen Speicherstelle mit INC und subtrahiert es dann mit SBC vom Akku.	$EF	absolut		3	6		N Z C
	INS $082E,X	INC+SBC		$1F	absolut X-indiziert		3	7		N Z C
	INS $082E,Y	INC+SBC		$1B	absolut Y-indiziert		3	7		N Z C
	INS $FB	INC+SBC		$07	Zero-Page		2	5		N Z C
	INS $FB,X	INC+SBC		$17	Zero-Page X-indiziert		2	6		N Z C
	INS ($FB,X)	INC+SBC		$03	indirekt X-indiziert		2	8		N Z C
	INS ($FB),Y	INC+SBC		$13	indirekt Y-nach-indiziert		2	8		N Z C
	LAX $082E	LDA+LDX	LAX lädt das Byte von der angegebenen Adresse gleichzeitig in den Akku und ins X-Register.	$AF	absolut		3	4		N Z
	LAX $082E,Y	LDA+LDX		$BF	absolut Y-indiziert		3	4-5		N Z
	LAX $FB	LDA+LDX		$A7	Zero-Page		2	3		N Z
	LAX $FB,X	LDA+LDX		$B7	Zero-Page X-indiziert		2	4-5	Beim Überschreiten der Page-Grenze werden 5 Taktzyklen benötigt, sonst 4.	N Z
	LAX ($FB,X)	LDA+LDX		$A3	indirekt X-indiziert		2	6		N Z
	LAX ($FB),Y	LDA+LDX		$B3	indirekt Y-nach-indiziert		2	5-6	Beim Überschreiten der Page-Grenze werden 6 Taktzyklen benötigt, sonst 5.	N Z
SRE	LSE $082E	LSR+EOR	Kombination von LSR + EOR. Dabei wird das über die Adressierung zufindene Byte zunächst per LSR bitweise nach rechts verschoben und anschließend mit dem Akku exklusiv-ODER-Verknüpft.	$4F	absolut		3	6		N Z C
	LSE $082E,X	LSR+EOR		$5F	absolut X-indiziert		3	7		N Z C
	LSE $082E,Y	LSR+EOR		$5B	absolut Y-indiziert		3	7		N Z C
	LSE $FB	LSR+EOR		$47	Zero-Page		2	5		N Z C
	LSE $FB,X	LSR+EOR		$57	Zero-Page X-indiziert		2	6		N Z C
	LSE ($FB,X)	LSR+EOR		$43	indirekt X-indiziert		2	8		N Z C
	LSE ($FB),Y	LSR+EOR		$53	indirekt Y-nach-indiziert		2	8		N Z C
NOOP ($1A)	NOP	No OPeration	Weitere OpCodes für den bekannten NOP-Befehl.	$1A,$3A, $5A,$7A,$DA, $FA	implizit		1	2		

alternatives Mnemonic:	Befehl	Mnemonics Erklärung	Beschreibung		Adressierungsart	Beschreibung Adressierungsart	Bytelänge	Taktzyklen		beinflußte Register
LXA ATX	OAL #$A5	ORA+AND+LDX	Führt zuerst per ORA eine ODER-Verknüpfung mit dem Akku und dem Wert #$EE durch. Dann wird der Akku durch AND mit dem angegebenen Wert UND-Verknüpft und anschließend ins X-Register kopiert TAX.	$AB	unmittelbar		2	2		N Z
	RLA $082E	ROL+AND	Kombination von ROL + AND. Dabei wird das über die Adressierung zufindene Byte zunächst per ROL bitweise nach links verschoben (von rechts wird das C-Flag eingeschoben) und anschließend mit dem Akku UND-Verknüpft.	$2F	absolut		3	6		N Z C
	RLA $082E,X	ROL+AND		$3F	absolut X-indiziert		3	7		N Z C
	RLA $082E,Y	ROL+AND		$3B	absolut Y-indiziert		3	7		N Z C
	RLA $FB	ROL+AND		$27	Zero-Page		2	5		N Z C
	RLA $FB,X	ROL+AND		$37	Zero-Page X-indiziert		2	6		N Z C
	RLA ($FB,X)	ROL+AND		$23	indirekt X-indiziert		2	8		N Z C
	RLA ($FB),Y	ROL+AND		$33	indirekt Y-nach-indiziert		2	8		N Z C
	RRA $082E	ROR+ADC	RRA kombiniert ein ROR mit einem ADC. Dabei wird das über die Adressierung zufindene Byte zunächst per ROR bitweise nach rechts verschoben (das herausfallende Bit landet im C-Flag) und wird anschließend durch ADC mit dem Akku addiert (dabei wird natürlich auch das Carry-Flag vom ROR beachtet).	$6F	absolut		3	6		N Z
	RRA $082E,X	ROR+ADC		$7F	absolut X-indiziert		3	7		N Z
	RRA $082E,Y	ROR+ADC		$7B	absolut Y-indiziert		3	7		N Z
	RRA $FB	ROR+ADC		$67	Zero-Page		2	5		N Z
	RRA $FB,X	ROR+ADC		$77	Zero-Page X-indiziert		2	6		N Z
	RRA ($FB,X)	ROR+ADC		$63	indirekt X-indiziert		2	8		N Z
	RRA ($FB),Y	ROR+ADC		$73	indirekt Y-nach-indiziert		2	8		N Z
SBX AXS	SAX #$A5	Subtract+AND+TAX	Hier wird zu erst eine UND-Verknüfung zwischen dem Akku und dem X-Register vorgenommen. Dann wird vom Ergebnis der UND-Verknüpfung der angegebenen Wert abgezogen (ohne das Carry-Flag zu beachten!) und das Ergebnis im X-Register gespeichert. Das C-Flag kann aber durch die Subtraktion wiederum gesetzt werden, das OVerflow-Flag bleibt allerdings, ebenso wie der Akku, unverändert!	$CB	unmittelbar		2	2		N Z
NOOP #$00 ($80) DOP / NOP	SKB	SKip next Byte	Überspringt einfach das nächste Byte.	$80, $82, $89, $C2, $E2, $04, $14, $34, $44, $54, $64, $74, $D4, $F4	implizit		2	2-4	$80;$82;$89;$C2;E2 = NOOP #$FF (2 Taktzyklen) $04;44;64 = NOOP $FF (3 Taktzyklen) $14;$34;$54;$74;$D4 = NOOP,X (4 Taktzyklen)	
NOOP $FFFF ($0C) TOP / NOP	SKW	SKip next Word	Überspringt das nächste Word, die nächsten zwei Byte.	$0C,$1C, $3C,$5C, $7C,$DC, $FC	implizit	Es muß im Vice Monitor eine Argument größer 255 ($FF) angeben werden. Ansonsten wird NOOP #$00 ($80) angenommen. $0C = NOOP $FFFF / alle anderen ergeben NOOP $FFFF,X	3	4-5	Der OpCode $0C benötigt 4, die anderen 5.	
ANE	XAA #$A5	TXA+AND	Kopiert den Inhalt des X-Register mit TAX in den Akku und führt dann eine UND-Verknüpfung mit dem angegebenen Wert und dem Akku durch.	$8B	unmittelbar		2	2		N Z

ANE (grünes Mnemonics) muß im WinVice Monitor als illegaler Opcode eingegeben werden. Der in Klammern angegebene HEX-Wert wird von Winvice erzeugt.

Quellen:

http://www.retro-programming.de/?page_id=373
http://nesdev.com/undocumented_opcodes.txt

Danksagung:

Ich möchte Jörn Kierstein danken. Er ermöglichte mir durch seine Arbeit diese Tabelle zuerstellen.
http://www.retro-programming.de/

Erstellt: 22.01.2015