

BRK (00)		Cycles: 7	Size: 2*
Implied			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1		Inc. PC.	
1.2	Read PC	To be discarded.	
2.1		Inc. PC.	
2.2	Write S \$0100	Push PC.H onto stack.	
3.1			
3.2	Write (S -1) \$0100	Push PC.L onto stack.	
4.1		Use Interrupt flags to decide which vector to use. RESET=\$FFFC NMI=\$FFFA IRQ/BRK=\$FFFE	
4.2	Write (S -2) \$0100	Push P onto stack with B flag if software BRK.	
5.1		Decrease S by 3.	
5.2	Read Vector Low	Store to Address.L.	
6.1		Set I , unset B .	
6.2	Read Vector High	Store to Address.H.	
7.1		Enable writes to PC (disabled by NMI/IRQ). Copy Address to PC.	
7.2	Read PC	Store as OpCode.	
+X.1			

* Concerning the BRK instruction, you should also note that although its second byte is basically a “don’t care” byte – that is, it can have any value - the BRK (and COP instruction as well) is a two-byte instruction, the second byte sometimes is used as a signature byte to determine the nature of the BRK being executed. When an RTI instruction is executed, control always returns to the second byte past the BRK opcode. — assembly-programming-manual-for-w65c816.pdf

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	00	1	BRK	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	00	1	BRK	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	BRK	T2
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	BRK	T2
4	01fd	20	0		0003	aa	00	00	fd	nv-BdiZc	BRK	T3
4	01fd	00	0		0003	aa	00	00	fd	nv-BdiZc	BRK	T3
5	01fc	20	0		0003	aa	00	00	fd	nv-BdiZc	BRK	T4
5	01fc	03	0		0003	aa	00	00	fd	nv-BdiZc	BRK	T4
6	01fb	20	0		0003	aa	00	00	fd	nv-BdiZc	BRK	T5
6	01fb	32	0		0003	aa	00	00	fd	nv-BdiZc	BRK	T5
7	fffe	00	1		0003	aa	00	00	fa	nv-BdiZc	BRK	
7	fffe	00	1		0003	aa	00	00	fa	nv-BdiZc	BRK	
8	ffff	00	1		0003	aa	00	00	fa	nv-BdIZc	BRK	T0
8	ffff	00	1		0003	aa	00	00	fa	nv-BdIZc	BRK	T0
9	0000	58	1	CLI	0000	aa	00	00	fa	nv-BdIZc	BRK	T1
9	0000	58	1	CLI	0000	aa	00	00	fa	nv-BdIZc	BRK	T1

ORA (01)	Cycles: 6	Size: 2
Indirect, X (Read)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Operand.
2.1		Inc. PC.
2.2	Read Operand	
3.1		Set Pointer to (Operand+X) & \$FF.
3.2	Read Pointer	Store to Address.L.
4.1		
4.2	Read (Pointer+1) & \$FF	Store to Address.H.
5.1		
5.2	Read Address	Store as Operand.
6.1		* Perform A=A Operand , set N and Z accordingly.
6.2	Read PC	Store as OpCode.
+X.1		A and N, Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	01	1	ORA (zp,X)	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	01	1	ORA (zp,X)	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	ORA (zp,X)	T2
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	ORA (zp,X)	T2
4	0020	22	1		0003	aa	00	00	fd	nv-BdiZc	ORA (zp,X)	T3
4	0020	22	1		0003	aa	00	00	fd	nv-BdiZc	ORA (zp,X)	T3
5	0020	22	1		0003	aa	00	00	fd	nv-BdiZc	ORA (zp,X)	T4
5	0020	22	1		0003	aa	00	00	fd	nv-BdiZc	ORA (zp,X)	T4
6	0021	00	1		0003	aa	00	00	fd	nv-BdiZc	ORA (zp,X)	T5
6	0021	00	1		0003	aa	00	00	fd	nv-BdiZc	ORA (zp,X)	T5
7	0022	ff	1		0003	aa	00	00	fd	nv-BdiZc	ORA (zp,X)	T0
7	0022	ff	1		0003	aa	00	00	fd	nv-BdiZc	ORA (zp,X)	T0
8	0003	10	1	BPL	0003	aa	00	00	fd	nv-BdiZc	ORA (zp,X)	T1
8	0003	10	1	BPL	0003	aa	00	00	fd	nv-BdiZc	ORA (zp,X)	T1
9	0004	00	1		0004	ff	00	00	fd	Nv-Bdizc	BPL	T2
9	0004	00	1		0004	ff	00	00	fd	Nv-Bdizc	BPL	T2

JAM (02)	Cycles: ∞	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	
2.1		
2.2	Read \$FFFF	
3.1		
3.2	Read \$FFFE	
4.1		
4.2	Read \$FFFE	
5.1		Repeat 5.1 and 5.2 forever.
5.2	Read \$FFFF	

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	

SLO (03)	Cycles: 8	Size: 2
Indirect, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Operand.
2.1		Inc. PC.
2.2	Read Operand	
3.1		Set Pointer to (Operand+X) & \$FF.
3.2	Read Pointer	Store to Address.L.
4.1		
4.2	Read (Pointer+1) & \$FF	Store to Address.H.
5.1		
5.1	Read Address	Store as Operand.
6.1		
6.2	Write Address	Write unmodified Operand.
7.1		* Set C if high bit of Operand is set. Perform Operand=Operand << 1, A=A Operand, set N and Z based off A .
7.2	Write Address	Write modified Operand.
8.1		
8.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	03	1	unknown	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	03	1	unknown	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	unknown	T2
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	unknown	T2
4	0020	22	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T3
4	0020	22	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T3
5	0020	22	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T4
5	0020	22	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T4
6	0021	00	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T5
6	0021	00	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T5
7	0022	ff	1		0003	aa	00	00	fd	nv-BdiZc	unknown	
7	0022	ff	1		0003	aa	00	00	fd	nv-BdiZc	unknown	
8	0022	ff	0		0003	aa	00	00	fd	nv-BdiZc	unknown	
8	0022	ff	0		0003	aa	00	00	fd	nv-BdiZc	unknown	
9	0022	ff	0		0003	aa	00	00	fd	Nv-BdizC	unknown	T0
9	0022	fe	0		0003	aa	00	00	fd	Nv-BdizC	unknown	T0
10	0003	10	1	BPL	0003	aa	00	00	fd	Nv-BdizC	unknown	T1
10	0003	10	1	BPL	0003	aa	00	00	fd	Nv-BdizC	unknown	T1
11	0004	00	1		0004	fe	00	00	fd	Nv-BdizC	BPL	T2
11	0004	00	1		0004	fe	00	00	fd	Nv-BdizC	BPL	T2

NOP (04)		Cycles: 3	Size: 2
Zero Page (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		Store as OpCode.	
3.2	Read PC		
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	04	1	unknown	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	04	1	unknown	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	unknown	T2
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	unknown	T2
4	0020	22	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T0
4	0020	22	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T0
5	0003	10	1	BPL	0003	aa	00	00	fd	nv-BdiZc	unknown	T1
5	0003	10	1	BPL	0003	aa	00	00	fd	nv-BdiZc	unknown	T1
6	0004	00	1		0004	aa	00	00	fd	nv-BdiZc	BPL	T2
6	0004	00	1		0004	aa	00	00	fd	nv-BdiZc	BPL	T2

ORA (05)		Cycles: 3	Size: 2
Zero Page (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		* Perform A=A Operand, set N and Z accordingly.	
3.2	Read PC	Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	05	1	ORA zp	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	05	1	ORA zp	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	ORA zp	T2
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	ORA zp	T2
4	0020	44	1		0003	aa	00	00	fd	nv-BdiZc	ORA zp	T0
4	0020	44	1		0003	aa	00	00	fd	nv-BdiZc	ORA zp	T0
5	0003	10	1	BPL	0003	aa	00	00	fd	nv-BdiZc	ORA zp	T1
5	0003	10	1	BPL	0003	aa	00	00	fd	nv-BdiZc	ORA zp	T1
6	0004	00	1		0004	ee	00	00	fd	Nv-Bdizc	BPL	T2
6	0004	00	1		0004	ee	00	00	fd	Nv-Bdizc	BPL	T2

ASL (06)	Cycles: 5	Size: 2
Zero Page (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.
2.1		Inc. PC.
2.2	Read Address	Store as Operand.
3.1		
3.2	Write Address	Write unmodified Operand.
4.1		Set C if high bit of Operand is set. Perform Operand=Operand << 1, set N and Z accordingly.
4.2	Write Address	Write modified Operand.
5.1		
5.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	06	1	ASL zp	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	06	1	ASL zp	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	ASL zp	T2
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	ASL zp	T2
4	0020	44	1		0003	aa	00	00	fd	nv-BdiZc	ASL zp	T3
4	0020	44	1		0003	aa	00	00	fd	nv-BdiZc	ASL zp	T3
5	0020	44	0		0003	aa	00	00	fd	nv-BdiZc	ASL zp	T4
5	0020	44	0		0003	aa	00	00	fd	nv-BdiZc	ASL zp	T4
6	0020	44	0		0003	aa	00	00	fd	Nv-Bdizc	ASL zp	T0
6	0020	88	0		0003	aa	00	00	fd	Nv-Bdizc	ASL zp	T0
7	0003	10	1	BPL	0003	aa	00	00	fd	Nv-Bdizc	ASL zp	T1
7	0003	10	1	BPL	0003	aa	00	00	fd	Nv-Bdizc	ASL zp	T1
8	0004	00	1		0004	aa	00	00	fd	Nv-Bdizc	BPL	T2
8	0004	00	1		0004	aa	00	00	fd	Nv-Bdizc	BPL	T2

SLO (07)	Cycles: 5	Size: 2
Zero Page (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.
2.1		Inc. PC.
2.2	Read Address	Store as Operand.
3.1		
3.2	Write Address	Write unmodified Operand.
4.1		Set C if high bit of Operand is set. * Perform Operand=Operand << 1, A = A Operand, set N and Z based off A .
4.2	Write Address	Write modified Operand.
5.1		
5.2	Read PC	Store as OpCode.
+X.1		A and N , Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	07	1	unknown	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	07	1	unknown	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	unknown	T2
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	unknown	T2
4	0020	88	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T3
4	0020	88	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T3
5	0020	88	0		0003	aa	00	00	fd	nv-BdiZc	unknown	T4
5	0020	88	0		0003	aa	00	00	fd	nv-BdiZc	unknown	T4
6	0020	88	0		0003	aa	00	00	fd	nv-BdiZc	unknown	T0
6	0020	10	0		0003	aa	00	00	fd	nv-BdiZc	unknown	T0
7	0003	10	1	BPL	0003	aa	00	00	fd	nv-BdiZc	unknown	T1
7	0003	10	1	BPL	0003	aa	00	00	fd	nv-BdiZc	unknown	T1
8	0004	00	1		0004	ba	00	00	fd	Nv-BdiZc	BPL	T2
8	0004	00	1		0004	ba	00	00	fd	Nv-BdiZc	BPL	T2

PHP (08)		Cycles: 3	Size: 1
Implied			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		To be discarded.	
2.1			
2.2		Push the status register with B and M set.	
3.1		Dec. S .	
3.2	Read PC	Store as OpCode.	
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	08	1	PHP	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	08	1	PHP	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	PHP	T2
3	0002	20	1		0002	aa	00	00	fd	nv-BdiZc	PHP	T2
4	01fd	20	0		0002	aa	00	00	fd	nv-BdiZc	PHP	T0
4	01fd	32	0		0002	aa	00	00	fd	nv-BdiZc	PHP	T0
5	0002	20	1	JSR Abs	0002	aa	00	00	fc	nv-BdiZc	PHP	T1
5	0002	20	1	JSR Abs	0002	aa	00	00	fc	nv-BdiZc	PHP	T1
6	0003	10	1		0003	aa	00	00	fc	nv-BdiZc	JSR Abs	T2
6	0003	10	1		0003	aa	00	00	fc	nv-BdiZc	JSR Abs	T2

ORA (09)		Cycles: 2	Size: 2
Immediate (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Operand.	
2.1		Inc.PC. * Perform A=A Operand, set N and Z accordingly.	
2.2		Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	09	1	ORA #	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	09	1	ORA #	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	ff	1		0002	aa	00	00	fd	nv-BdiZc	ORA #	T0+T2
3	0002	ff	1		0002	aa	00	00	fd	nv-BdiZc	ORA #	T0+T2
4	0003	10	1	BPL	0003	aa	00	00	fd	nv-BdiZc	ORA #	T1
4	0003	10	1	BPL	0003	aa	00	00	fd	nv-BdiZc	ORA #	T1
5	0004	00	1		0004	ff	00	00	fd	Nv-Bdizc	BPL	T2
5	0004	00	1		0004	ff	00	00	fd	Nv-Bdizc	BPL	T2

ASL (0A)	Cycles: 2	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1	Read PC	Inc. PC.
1.2		To be discarded.
2.1		Set C if high bit of Operand is set. Perform A=A << 1 , set N and Z accordingly.
2.2		Store as OpCode.
+X.1		A and C , N , Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	0a	1	ASL	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	0a	1	ASL	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	98	1		0002	aa	00	00	fd	nv-BdiZc	ASL	T0+T2
3	0002	98	1		0002	aa	00	00	fd	nv-BdiZc	ASL	T0+T2
4	0002	98	1	TYA	0002	aa	00	00	fd	nv-BdiZc	ASL	T1
4	0002	98	1	TYA	0002	aa	00	00	fd	nv-BdiZc	ASL	T1
5	0003	10	1		0003	54	00	00	fd	nv-BdiZc	TYA	T0+T2
5	0003	10	1		0003	54	00	00	fd	nv-BdiZc	TYA	T0+T2

ANC (0B)		Cycles: 2	Size: 2
Immediate (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Operand.	
2.1		Inc. PC. * Perform A=A & Operand , set C , N , and Z accordingly (C is set using logic for N).	
2.2		Store as OpCode.	
+X.1		** A and C , N , Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

** Visual6502 incorrectly omits the update of A; flags are updated based off the incorrect A value.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	0b	1	unknown	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	0b	1	unknown	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	0f	1		0002	aa	00	00	fd	nv-BdiZc	unknown	T0+T2
3	0002	0f	1		0002	aa	00	00	fd	nv-BdiZc	unknown	T0+T2
4	0003	10	1	BPL	0003	aa	00	00	fd	nv-BdiZc	unknown	T1
4	0003	10	1	BPL	0003	aa	00	00	fd	nv-BdiZc	unknown	T1
5	0004	00	1		0004	aa	00	00	fd	Nv-BdizC	BPL	T2
5	0004	00	1		0004	aa	00	00	fd	Nv-BdizC	BPL	T2

ANC (ANC2, ANA, ANB)

Type: Combination of an immediate and an implied command (Sub-instructions: AND, ASL/ROL)

Op.	Mnemonic	Function	Size	Cycles	N	V	-	B	D	I	Z	C
\$0B	ANC #imm	A = A & #{imm}	2	2	0						0	0
\$2B	ANC #imm	A = A & #{imm}	2	2	0						0	0

Operation: ANDs the contents of the A register with an immediate value and then moves bit 7 of A into the Carry flag.

- This opcode works basically identically to AND #imm. except that the Carry flag is set to the same state that the Negative flag is set to. (bit 7 is put into the carry, as if the ASL/ROL would have been executed)

— NoMoreSecrets-NMOS6510UnintendedOpcodes-20232412.pdf

NOP (0C)		Cycles: 4	Size: 3
Absolute (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.L.	
2.1		Inc. PC.	
2.2		Store as Address.H.	
3.1		Inc. PC.	
3.2		Store as Operand.	
4.1		Store as OpCode.	
4.2			
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	0c	1	unknown	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	0c	1	unknown	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	0f	1		0002	aa	00	00	fd	nv-BdiZc	unknown	T2
3	0002	0f	1		0002	aa	00	00	fd	nv-BdiZc	unknown	T2
4	0003	10	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T3
4	0003	10	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T3
5	100f	00	1		0004	aa	00	00	fd	nv-BdiZc	unknown	T0
5	100f	00	1		0004	aa	00	00	fd	nv-BdiZc	unknown	T0
6	0004	00	1	BRK	0004	aa	00	00	fd	nv-BdiZc	unknown	T1
6	0004	00	1	BRK	0004	aa	00	00	fd	nv-BdiZc	unknown	T1
7	0005	4c	1		0005	aa	00	00	fd	nv-BdiZc	BRK	T2
7	0005	4c	1		0005	aa	00	00	fd	nv-BdiZc	BRK	T2

ORA (0D)		Cycles: 4	Size: 3
Absolute (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.L.	
2.1		Inc. PC.	
2.2		Store as Address.H.	
3.1		Inc. PC.	
3.2	Read Address	Store as Operand.	
4.1	Read PC	* Perform A=A Operand, set N and Z accordingly.	
4.2		Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	0d	1	ORA Abs	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	0d	1	ORA Abs	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	1f	1		0002	aa	00	00	fd	nv-BdiZc	ORA Abs	T2
3	0002	1f	1		0002	aa	00	00	fd	nv-BdiZc	ORA Abs	T2
4	0003	01	1		0003	aa	00	00	fd	nv-BdiZc	ORA Abs	T3
4	0003	01	1		0003	aa	00	00	fd	nv-BdiZc	ORA Abs	T3
5	011f	0f	1		0004	aa	00	00	fd	nv-BdiZc	ORA Abs	T0
5	011f	0f	1		0004	aa	00	00	fd	nv-BdiZc	ORA Abs	T0
6	0004	a9	1	LDA #	0004	aa	00	00	fd	nv-BdiZc	ORA Abs	T1
6	0004	a9	1	LDA #	0004	aa	00	00	fd	nv-BdiZc	ORA Abs	T1
7	0005	4c	1		0005	af	00	00	fd	Nv-Bdizc	LDA #	T0+T2
7	0005	4c	1		0005	af	00	00	fd	Nv-Bdizc	LDA #	T0+T2

ASL (0E)	Cycles: 6	Size: 3
Absolute (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC.
3.2	Read Address	Store as Operand.
4.1		
4.2	Write Address	Write unmodified Operand.
5.1		Set C if high bit of Operand is set. Perform Operand=Operand << 1, set N and Z accordingly.
5.2	Write Address	Write modified Operand.
6.1		
6.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	0e	1	ASL Abs	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	0e	1	ASL Abs	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	1f	1		0002	aa	00	00	fd	nv-BdiZc	ASL Abs	T2
3	0002	1f	1		0002	aa	00	00	fd	nv-BdiZc	ASL Abs	T2
4	0003	01	1		0003	aa	00	00	fd	nv-BdiZc	ASL Abs	T3
4	0003	01	1		0003	aa	00	00	fd	nv-BdiZc	ASL Abs	T3
5	011f	0f	1		0004	aa	00	00	fd	nv-BdiZc	ASL Abs	T4
5	011f	0f	1		0004	aa	00	00	fd	nv-BdiZc	ASL Abs	T4
6	011f	0f	0		0004	aa	00	00	fd	nv-BdiZc	ASL Abs	T5
6	011f	0f	0		0004	aa	00	00	fd	nv-BdiZc	ASL Abs	T5
7	011f	0f	0		0004	aa	00	00	fd	nv-Bdizc	ASL Abs	T0
7	011f	1e	0		0004	aa	00	00	fd	nv-Bdizc	ASL Abs	T0
8	0004	a9	1	LDA #	0004	aa	00	00	fd	nv-Bdizc	ASL Abs	T1
8	0004	a9	1	LDA #	0004	aa	00	00	fd	nv-Bdizc	ASL Abs	T1
9	0005	4c	1		0005	aa	00	00	fd	nv-Bdizc	LDA #	T0+T2
9	0005	4c	1		0005	aa	00	00	fd	nv-Bdizc	LDA #	T0+T2

SLO (0F)	Cycles: 6	Size: 3
Absolute (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC.
3.2	Read Address	Store as Operand.
4.1		
4.2	Write Address	Write unmodified Operand.
5.1		Set C if high bit of Operand is set. * Perform Operand=Operand << 1, A=A Operand, set N and Z based off A .
5.2	Write Address	Write modified Operand.
6.1		
6.2	Read PC	Store as OpCode.
+X.1		A and N, Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0001	0f	1	unknown	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
2	0001	0f	1	unknown	0001	aa	00	00	fd	nv-BdiZc	CLI	T1
3	0002	1f	1		0002	aa	00	00	fd	nv-BdiZc	unknown	T2
3	0002	1f	1		0002	aa	00	00	fd	nv-BdiZc	unknown	T2
4	0003	01	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T3
4	0003	01	1		0003	aa	00	00	fd	nv-BdiZc	unknown	T3
5	011f	1e	1		0004	aa	00	00	fd	nv-BdiZc	unknown	T4
5	011f	1e	1		0004	aa	00	00	fd	nv-BdiZc	unknown	T4
6	011f	1e	0		0004	aa	00	00	fd	nv-BdiZc	unknown	T5
6	011f	1e	0		0004	aa	00	00	fd	nv-BdiZc	unknown	T5
7	011f	1e	0		0004	aa	00	00	fd	nv-Bdizc	unknown	T0
7	011f	3c	0		0004	aa	00	00	fd	nv-Bdizc	unknown	T0
8	0004	a9	1	LDA #	0004	aa	00	00	fd	nv-Bdizc	unknown	T1
8	0004	a9	1	LDA #	0004	aa	00	00	fd	nv-Bdizc	unknown	T1
9	0005	4c	1		0005	be	00	00	fd	Nv-Bdizc	LDA #	T0+T2
9	0005	4c	1		0005	be	00	00	fd	Nv-Bdizc	LDA #	T0+T2

BPL (10)	Cycles: 2-4	Size: 2
Branch Relative		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1	Read PC	Inc. PC. Check condition (N == 0).
1.2		Store as Operand. Treat as signed 16-bit (Op16=i16(i8(Operand))).
2.1		Inc. PC. If not jumping, end (next half-cycle is 4.2)
2.2		If (PC+Op16).H != PC.H, end after PC.L fix (next half-cycle is 4.2). PC.L=PC.L+Operand.
3.1		
3.2	Read PC	PC.H=previous “(PC+Op16).H” value.
4.1	Read PC	Store as OpCode.
4.2		
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	10	1	BPL	0002	80	00	00	fd	Nv-BdIZc	LDA #	T1
2	0002	10	1	BPL	0002	80	00	00	fd	Nv-BdIZc	LDA #	T1
3	0003	fc	1		0003	80	00	00	fd	Nv-BdIZc	BPL	T2
3	0003	fc	1		0003	80	00	00	fd	Nv-BdIZc	BPL	T2
4	0004	00	1	BRK	0004	80	00	00	fd	Nv-BdIZc	BPL	
4	0004	00	1	BRK	0004	80	00	00	fd	Nv-BdIZc	BPL	
5	0005	4c	1		0005	80	00	00	fd	Nv-BdIZc	BRK	T2
5	0005	4c	1		0005	80	00	00	fd	Nv-BdIZc	BRK	T2

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	10	1	BPL	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
2	0002	10	1	BPL	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
3	0003	fc	1		0003	00	00	00	fd	nv-BdIZc	BPL	T2
3	0003	fc	1		0003	00	00	00	fd	nv-BdIZc	BPL	T2
4	0004	00	1		0004	00	00	00	fd	nv-BdIZc	BPL	T3
4	0004	00	1		0004	00	00	00	fd	nv-BdIZc	BPL	T3
5	0000	a9	1	LDA #	0000	00	00	00	fd	nv-BdIZc	BPL	
5	0000	a9	1	LDA #	0000	00	00	00	fd	nv-BdIZc	BPL	
6	0001	00	1		0001	00	00	00	fd	nv-BdIZc	LDA #	T0+T2
6	0001	00	1		0001	00	00	00	fd	nv-BdIZc	LDA #	T0+T2

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	00fc	10	1	BPL	00fc	00	00	00	fd	nv-BdIZc	JMP Abs	T1
5	00fc	10	1	BPL	00fc	00	00	00	fd	nv-BdIZc	JMP Abs	T1
6	00fd	1c	1		00fd	00	00	00	fd	nv-BdIZc	BPL	T2
6	00fd	1c	1		00fd	00	00	00	fd	nv-BdIZc	BPL	T2
7	00fe	22	1		00fe	00	00	00	fd	nv-BdIZc	BPL	T3
7	00fe	22	1		00fe	00	00	00	fd	nv-BdIZc	BPL	T3
8	001a	00	1		001a	00	00	00	fd	nv-BdIZc	BPL	T0
8	001a	00	1		001a	00	00	00	fd	nv-BdIZc	BPL	T0
9	011a	00	1	BRK	011a	00	00	00	fd	nv-BdIZc	BPL	T1
9	011a	00	1	BRK	011a	00	00	00	fd	nv-BdIZc	BPL	T1
10	011b	00	1		011b	00	00	00	fd	nv-BdIZc	BRK	T2
10	011b	00	1		011b	00	00	00	fd	nv-BdIZc	BRK	T2

ORA (11)		Cycles: 5-6	Size: 2
Indirect, Y (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Pointer.	
2.1		Inc. PC.	
2.2		Store to Address.L.	
3.1	Read Pointer		
3.2		Store to Address.H.	
4.1		Final=Address+Y. Address.L = Final.L.	
4.2		Store as Operand. If Address.H == Final.H, skip the next cycle (next half-cycle is 6.1).	
5.1	Read Address	Address.H = Final.H (fixes high byte of address).	
5.2		Store as Operand.	
6.1		* Perform A=A Operand , set N and Z accordingly.	
6.2		Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	00fc	11	1	ORA (zp),Y	00fc	aa	00	4c	fd	nv-BdIzc	JMP Abs	T1
5	00fc	11	1	ORA (zp),Y	00fc	aa	00	4c	fd	nv-BdIzc	JMP Abs	T1
6	00fd	ee	1		00fd	aa	00	4c	fd	nv-BdIzc	ORA (zp),Y	T2
6	00fd	ee	1		00fd	aa	00	4c	fd	nv-BdIzc	ORA (zp),Y	T2
7	00ee	22	1		00fe	aa	00	4c	fd	nv-BdIzc	ORA (zp),Y	T3
7	00ee	22	1		00fe	aa	00	4c	fd	nv-BdIzc	ORA (zp),Y	T3
8	00ef	01	1		00fe	aa	00	4c	fd	nv-BdIzc	ORA (zp),Y	T4
8	00ef	01	1		00fe	aa	00	4c	fd	nv-BdIzc	ORA (zp),Y	T4
9	016e	ff	1		00fe	aa	00	4c	fd	nv-BdIzc	ORA (zp),Y	T0
9	016e	ff	1		00fe	aa	00	4c	fd	nv-BdIzc	ORA (zp),Y	T0
10	00fe	4a	1	LSR	00fe	aa	00	4c	fd	nv-BdIzc	ORA (zp),Y	T1
10	00fe	4a	1	LSR	00fe	aa	00	4c	fd	nv-BdIzc	ORA (zp),Y	T1
11	00ff	01	1		00ff	ff	00	4c	fd	Nv-BdIzc	LSR	T0+T2
11	00ff	01	1		00ff	ff	00	4c	fd	Nv-BdIzc	LSR	T0+T2

JAM (12)	Cycles: ∞	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	
2.1		
2.2	Read \$FFFF	
3.1		
3.2	Read \$FFFE	
4.1		
4.2	Read \$FFFE	
5.1		Repeat 5.1 and 5.2 forever.
5.2	Read \$FFFF	

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	

SLO (13)	Cycles: 8	Size: 2
Indirect, Y (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Pointer.
2.1		Inc. PC.
2.2	Read Pointer	Store to Address.L.
3.1		
3.2	Read (Pointer+1) & \$FF	Store to Address.H.
4.1		Final=Address+Y. Address.L = Final.L.
4.2	Read Address	Store as Operand.
5.1		Address.H = Final.H (fixes high byte of address).
5.2	Read Address	Store as Operand.
6.1		
6.2	Write Address	Write unmodified Operand.
7.1		* Set C if high bit of Operand is set. Perform Operand=Operand << 1, A =A Operand, set N and Z based off A .
7.2	Write Address	Write modified Operand.
8.1		
8.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	00fc	13	1	unknown	00fc	aa	00	4c	fd	nv-BdIzc	JMP Abs	T1
5	00fc	13	1	unknown	00fc	aa	00	4c	fd	nv-BdIzc	JMP Abs	T1
6	00fd	ee	1		00fd	aa	00	4c	fd	nv-BdIzc	unknown	T2
6	00fd	ee	1		00fd	aa	00	4c	fd	nv-BdIzc	unknown	T2
7	00ee	22	1		00fe	aa	00	4c	fd	nv-BdIzc	unknown	T3
7	00ee	22	1		00fe	aa	00	4c	fd	nv-BdIzc	unknown	T3
8	00ef	01	1		00fe	aa	00	4c	fd	nv-BdIzc	unknown	T4
8	00ef	01	1		00fe	aa	00	4c	fd	nv-BdIzc	unknown	T4
9	016e	ff	1		00fe	aa	00	4c	fd	nv-BdIzc	unknown	T5
9	016e	ff	1		00fe	aa	00	4c	fd	nv-BdIzc	unknown	T5
10	016e	ff	1		00fe	aa	00	4c	fd	nv-BdIzc	unknown	
10	016e	ff	1		00fe	aa	00	4c	fd	nv-BdIzc	unknown	
11	016e	ff	0		00fe	aa	00	4c	fd	nv-BdIzc	unknown	
11	016e	ff	0		00fe	aa	00	4c	fd	nv-BdIzc	unknown	
12	016e	ff	0		00fe	aa	00	4c	fd	Nv-BdIzC	unknown	T0
12	016e	fe	0		00fe	aa	00	4c	fd	Nv-BdIzC	unknown	T0
13	00fe	4a	1	LSR	00fe	aa	00	4c	fd	Nv-BdIzC	unknown	T1
13	00fe	4a	1	LSR	00fe	aa	00	4c	fd	Nv-BdIzC	unknown	T1
14	00ff	01	1		00ff	fe	00	4c	fd	Nv-BdIzC	LSR	T0+T2
14	00ff	01	1		00ff	fe	00	4c	fd	Nv-BdIzC	LSR	T0+T2

NOP (14)		Cycles: 4	Size: 2
Zero Page, X (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Pointer.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		Set Address to (Pointer+X) & \$FF.	
3.2		Store as Operand.	
4.1		Store as OpCode.	
4.2			
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
3	0002	14	1	unknown	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
3	0002	14	1	unknown	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	unknown	T2
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	unknown	T2
5	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T3
5	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T3
6	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T0
6	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T0
7	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	NV-BdIzc	unknown	T1
7	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	NV-BdIzc	unknown	T1
8	0005	50	1		0005	aa	00	00	fd	NV-BdIzc	ORA (zp,X)	T2
8	0005	50	1		0005	aa	00	00	fd	NV-BdIzc	ORA (zp,X)	T2

ORA (15)	Cycles: 4	Size: 2
Zero Page, X (Read)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Pointer.
2.1		Inc. PC.
2.2	Read Pointer	Store as Operand.
3.1		Set Address to (Pointer+X) & \$FF.
3.2	Read Address	Store as Operand.
4.1		* Perform A=A Operand , set N and Z accordingly.
4.2	Read PC	Store as OpCode.
+X.1		A and N, Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	00fc	15	1	ORA zp,X	00fc	aa	00	01	fd	nv-BdIzc	JMP Abs	T1
5	00fc	15	1	ORA zp,X	00fc	aa	00	01	fd	nv-BdIzc	JMP Abs	T1
6	00fd	ee	1		00fd	aa	00	01	fd	nv-BdIzc	ORA zp,X	T2
6	00fd	ee	1		00fd	aa	00	01	fd	nv-BdIzc	ORA zp,X	T2
7	00ee	2f	1		00fe	aa	00	01	fd	nv-BdIzc	ORA zp,X	T3
7	00ee	2f	1		00fe	aa	00	01	fd	nv-BdIzc	ORA zp,X	T3
8	00ee	2f	1		00fe	aa	00	01	fd	nv-BdIzc	ORA zp,X	T0
8	00ee	2f	1		00fe	aa	00	01	fd	nv-BdIzc	ORA zp,X	T0
9	00fe	4a	1	LSR	00fe	aa	00	01	fd	nv-BdIzc	ORA zp,X	T1
9	00fe	4a	1	LSR	00fe	aa	00	01	fd	nv-BdIzc	ORA zp,X	T1
10	00ff	01	1		00ff	af	00	01	fd	Nv-BdIzc	LSR	T0+T2
10	00ff	01	1		00ff	af	00	01	fd	Nv-BdIzc	LSR	T0+T2

ASL (16)	Cycles: 6	Size: 2
Zero Page, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Pointer.
2.1		Inc. PC.
2.2	Read Pointer	Store as Operand.
3.1		Set Address to (Pointer+X) & \$FF.
3.2	Read Address	Store as Operand.
4.1		
4.2	Write Address	Write unmodified Operand.
5.1		Set C if high bit of Operand is set. Perform Operand=Operand << 1, set N and Z accordingly.
5.2	Write Address	Write modified Operand.
6.1		
6.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	00fc	16	1	ASL zp,X	00fc	aa	00	01	fd	nv-BdIzc	JMP Abs	T1
5	00fc	16	1	ASL zp,X	00fc	aa	00	01	fd	nv-BdIzc	JMP Abs	T1
6	00fd	ee	1		00fd	aa	00	01	fd	nv-BdIzc	ASL zp,X	T2
6	00fd	ee	1		00fd	aa	00	01	fd	nv-BdIzc	ASL zp,X	T2
7	00ee	2f	1		00fe	aa	00	01	fd	nv-BdIzc	ASL zp,X	T3
7	00ee	2f	1		00fe	aa	00	01	fd	nv-BdIzc	ASL zp,X	T3
8	00ee	2f	1		00fe	aa	00	01	fd	nv-BdIzc	ASL zp,X	T4
8	00ee	2f	1		00fe	aa	00	01	fd	nv-BdIzc	ASL zp,X	T4
9	00ee	2f	0		00fe	aa	00	01	fd	nv-BdIzc	ASL zp,X	T5
9	00ee	2f	0		00fe	aa	00	01	fd	nv-BdIzc	ASL zp,X	T5
10	00ee	2f	0		00fe	aa	00	01	fd	nv-BdIzc	ASL zp,X	T0
10	00ee	5e	0		00fe	aa	00	01	fd	nv-BdIzc	ASL zp,X	T0
11	00fe	4a	1	LSR	00fe	aa	00	01	fd	nv-BdIzc	ASL zp,X	T1
11	00fe	4a	1	LSR	00fe	aa	00	01	fd	nv-BdIzc	ASL zp,X	T1
12	00ff	01	1		00ff	aa	00	01	fd	nv-BdIzc	LSR	T0+T2
12	00ff	01	1		00ff	aa	00	01	fd	nv-BdIzc	LSR	T0+T2

SLO (17)	Cycles: 6	Size: 2
Zero Page, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Pointer.
2.1		Inc. PC.
2.2	Read Pointer	Store as Operand.
3.1		Set Address to (Pointer+X) & \$FF.
3.2	Read Address	Store as Operand.
4.1		
4.2	Write Address	Write unmodified Operand.
5.1		* Set C if high bit of Operand is set. Perform Operand=Operand << 1, A=A Operand, set N and Z based off A .
5.2	Write Address	Write modified Operand.
6.1		
6.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	00fc	17	1	unknown	00fc	aa	00	01	fd	nv-BdIzc	JMP Abs	T1
5	00fc	17	1	unknown	00fc	aa	00	01	fd	nv-BdIzc	JMP Abs	T1
6	00fd	ee	1		00fd	aa	00	01	fd	nv-BdIzc	unknown	T2
6	00fd	ee	1		00fd	aa	00	01	fd	nv-BdIzc	unknown	T2
7	00ee	5e	1		00fe	aa	00	01	fd	nv-BdIzc	unknown	T3
7	00ee	5e	1		00fe	aa	00	01	fd	nv-BdIzc	unknown	T3
8	00ee	5e	1		00fe	aa	00	01	fd	nv-BdIzc	unknown	T4
8	00ee	5e	1		00fe	aa	00	01	fd	nv-BdIzc	unknown	T4
9	00ee	5e	0		00fe	aa	00	01	fd	nv-BdIzc	unknown	T5
9	00ee	5e	0		00fe	aa	00	01	fd	nv-BdIzc	unknown	T5
10	00ee	5e	0		00fe	aa	00	01	fd	Nv-BdIzc	unknown	T0
10	00ee	bc	0		00fe	aa	00	01	fd	Nv-BdIzc	unknown	T0
11	00fe	4a	1	LSR	00fe	aa	00	01	fd	Nv-BdIzc	unknown	T1
11	00fe	4a	1	LSR	00fe	aa	00	01	fd	Nv-BdIzc	unknown	T1
12	00ff	01	1		00ff	be	00	01	fd	Nv-BdIzc	LSR	T0+T2
12	00ff	01	1		00ff	be	00	01	fd	Nv-BdIzc	LSR	T0+T2

CLC (18)	Cycles: 2	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1	Read PC	Inc. PC.
1.2		To be discarded.
2.1		Clear the C status bit.
2.2		Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
7	00fc	18	1	CLC	00fc	aa	00	00	fd	nv-BdIZC	JMP Abs	T1
7	00fc	18	1	CLC	00fc	aa	00	00	fd	nv-BdIZC	JMP Abs	T1
8	00fd	ee	1		00fd	aa	00	00	fd	nv-BdIZC	CLC	T0+T2
8	00fd	ee	1		00fd	aa	00	00	fd	nv-BdIZC	CLC	T0+T2
9	00fd	ee	1	INC Abs	00fd	aa	00	00	fd	nv-BdIZc	CLC	T1
9	00fd	ee	1	INC Abs	00fd	aa	00	00	fd	nv-BdIZc	CLC	T1
10	00fe	4a	1		00fe	aa	00	00	fd	nv-BdIZc	INC Abs	T2
10	00fe	4a	1		00fe	aa	00	00	fd	nv-BdIZc	INC Abs	T2

ORA (19)	Cycles: 4-5	Size: 3
Absolute, Y (Read)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC. Final=Address+Y. Address.L = Final.L.
3.2	Read Address	Store as Operand. If Address.H == Final.H, skip the next cycle (next half-cycle is 5.1).
4.1		Address.H = Final.H (fixes high byte of address).
4.2	Read Address	Store as Operand.
5.1		* Perform A=A Operand , set N and Z accordingly.
5.2	Read PC	Store as OpCode.
+X.1		A and N, Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	00fc	19	1	ORA Abs,Y	00fc	aa	00	38	fd	nv-BdIzc	JMP Abs	T1
5	00fc	19	1	ORA Abs,Y	00fc	aa	00	38	fd	nv-BdIzc	JMP Abs	T1
6	00fd	ff	1		00fd	aa	00	38	fd	nv-BdIzc	ORA Abs,Y	T2
6	00fd	ff	1		00fd	aa	00	38	fd	nv-BdIzc	ORA Abs,Y	T2
7	00fe	00	1		00fe	aa	00	38	fd	nv-BdIzc	ORA Abs,Y	T3
7	00fe	00	1		00fe	aa	00	38	fd	nv-BdIzc	ORA Abs,Y	T3
8	0037	00	1		00ff	aa	00	38	fd	nv-BdIzc	ORA Abs,Y	T4
8	0037	00	1		00ff	aa	00	38	fd	nv-BdIzc	ORA Abs,Y	T4
9	0137	0f	1		00ff	aa	00	38	fd	nv-BdIzc	ORA Abs,Y	T0
9	0137	0f	1		00ff	aa	00	38	fd	nv-BdIzc	ORA Abs,Y	T0
10	00ff	18	1	CLC	00ff	aa	00	38	fd	nv-BdIzc	ORA Abs,Y	T1
10	00ff	18	1	CLC	00ff	aa	00	38	fd	nv-BdIzc	ORA Abs,Y	T1
11	0100	00	1		0100	af	00	38	fd	Nv-BdIzc	CLC	T0+T2
11	0100	00	1		0100	af	00	38	fd	Nv-BdIzc	CLC	T0+T2

NOP (1A)		Cycles: 2	Size: 1
Implied			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		To be discarded.	
2.1			
2.2		Store as OpCode.	
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	00fc	1a	1	unknown	00fc	aa	00	38	fd	nv-BdIzc	JMP Abs	T1
5	00fc	1a	1	unknown	00fc	aa	00	38	fd	nv-BdIzc	JMP Abs	T1
6	00fd	18	1		00fd	aa	00	38	fd	nv-BdIzc	unknown	T0+T2
6	00fd	18	1		00fd	aa	00	38	fd	nv-BdIzc	unknown	T0+T2
7	00fd	18	1	CLC	00fd	aa	00	38	fd	nv-BdIzc	unknown	T1
7	00fd	18	1	CLC	00fd	aa	00	38	fd	nv-BdIzc	unknown	T1
8	00fe	00	1		00fe	aa	00	38	fd	nv-BdIzc	CLC	T0+T2
8	00fe	00	1		00fe	aa	00	38	fd	nv-BdIzc	CLC	T0+T2

SLO (1B)	Cycles: 7	Size: 3
Absolute, Y (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC. Final=Address+Y. Address.L = Final.L.
3.2	Read Address	Store as Operand.
4.1		Address.H = Final.H (fixes high byte of address).
4.2	Read Address	Store as Operand.
5.1		
5.2	Write Address	Write unmodified Operand.
6.1		Set C if high bit of Operand is set. * Perform Operand=Operand << 1, A = A Operand, set N and Z based off A .
6.2	Write Address	Write modified Operand.
7.1		
7.2	Read PC	Store as OpCode.
+X.1		A and N , Z applied.

* Setting of A/flags is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	00fc	1b	1	unknown	00fc	aa	00	38	fd	nv-BdIzc	JMP Abs	T1
5	00fc	1b	1	unknown	00fc	aa	00	38	fd	nv-BdIzc	JMP Abs	T1
6	00fd	18	1		00fd	aa	00	38	fd	nv-BdIzc	unknown	T2
6	00fd	18	1		00fd	aa	00	38	fd	nv-BdIzc	unknown	T2
7	00fe	00	1		00fe	aa	00	38	fd	nv-BdIzc	unknown	T3
7	00fe	00	1		00fe	aa	00	38	fd	nv-BdIzc	unknown	T3
8	0050	0f	1		00ff	aa	00	38	fd	nv-BdIzc	unknown	T4
8	0050	0f	1		00ff	aa	00	38	fd	nv-BdIzc	unknown	T4
9	0050	0f	1		00ff	aa	00	38	fd	nv-BdIzc	unknown	T5
9	0050	0f	1		00ff	aa	00	38	fd	nv-BdIzc	unknown	T5
10	0050	0f	0		00ff	aa	00	38	fd	nv-BdIzc	unknown	
10	0050	0f	0		00ff	aa	00	38	fd	nv-BdIzc	unknown	
11	0050	0f	0		00ff	aa	00	38	fd	nv-BdIzc	unknown	T0
11	0050	1e	0		00ff	aa	00	38	fd	nv-BdIzc	unknown	T0
12	00ff	18	1	CLC	00ff	aa	00	38	fd	nv-BdIzc	unknown	T1
12	00ff	18	1	CLC	00ff	aa	00	38	fd	nv-BdIzc	unknown	T1
13	0100	00	1		0100	be	00	38	fd	Nv-BdIzc	CLC	T0+T2
13	0100	00	1		0100	be	00	38	fd	Nv-BdIzc	CLC	T0+T2

NOP (1C)		Cycles: 4-5	Size: 3
Absolute, X (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.L.	
2.1		Inc. PC.	
2.2		Store as Address.H.	
3.1		Inc. PC. Final=Address+X. Address.L = Final.L.	
3.2	Read Address	Store as Operand. If Address.H == Final.H, skip the next cycle (next half-cycle is 5.1).	
4.1	Read Address	Address.H = Final.H (fixes high byte of address).	
4.2		Store as Operand.	
5.1			
5.2	Read PC	Store as OpCode.	
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	00fc	1c	1	unknown	00fc	aa	00	38	fd	nv-BdIzc	JMP Abs	T1
5	00fc	1c	1	unknown	00fc	aa	00	38	fd	nv-BdIzc	JMP Abs	T1
6	00fd	18	1		00fd	aa	00	38	fd	nv-BdIzc	unknown	T2
6	00fd	18	1		00fd	aa	00	38	fd	nv-BdIzc	unknown	T2
7	00fe	00	1		00fe	aa	00	38	fd	nv-BdIzc	unknown	T3
7	00fe	00	1		00fe	aa	00	38	fd	nv-BdIzc	unknown	T3
8	0018	00	1		00ff	aa	00	38	fd	nv-BdIzc	unknown	T0
8	0018	00	1		00ff	aa	00	38	fd	nv-BdIzc	unknown	T0
9	00ff	18	1	CLC	00ff	aa	00	38	fd	nv-BdIzc	unknown	T1
9	00ff	18	1	CLC	00ff	aa	00	38	fd	nv-BdIzc	unknown	T1
10	0100	00	1		0100	aa	00	38	fd	nv-BdIzc	CLC	T0+T2
10	0100	00	1		0100	aa	00	38	fd	nv-BdIzc	CLC	T0+T2

ORA (1D)	Cycles: 4-5	Size: 3
Absolute, X (Read)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1	Read PC	Inc. PC.
1.2		Store as Address.L.
2.1		Inc. PC.
2.2		Store as Address.H.
3.1		Inc. PC. Final=Address+X. Address.L = Final.L.
3.2	Read Address	Store as Operand. If Address.H == Final.H, skip the next cycle (next half-cycle is 5.1).
4.1	Read Address	Address.H = Final.H (fixes high byte of address).
4.2		Store as Operand.
5.1		* Perform A=A Operand , set N and Z accordingly.
5.2		Store as OpCode.
+X.1		A and N, Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	00fc	1d	1	ORA Abs,X	00fc	aa	1f	00	fd	nv-BdIzc	JMP Abs	T1
5	00fc	1d	1	ORA Abs,X	00fc	aa	1f	00	fd	nv-BdIzc	JMP Abs	T1
6	00fd	ff	1		00fd	aa	1f	00	fd	nv-BdIzc	ORA Abs,X	T2
6	00fd	ff	1		00fd	aa	1f	00	fd	nv-BdIzc	ORA Abs,X	T2
7	00fe	00	1		00fe	aa	1f	00	fd	nv-BdIzc	ORA Abs,X	T3
7	00fe	00	1		00fe	aa	1f	00	fd	nv-BdIzc	ORA Abs,X	T3
8	001e	00	1		00ff	aa	1f	00	fd	nv-BdIzc	ORA Abs,X	T4
8	001e	00	1		00ff	aa	1f	00	fd	nv-BdIzc	ORA Abs,X	T4
9	011e	00	1		00ff	aa	1f	00	fd	nv-BdIzc	ORA Abs,X	T0
9	011e	00	1		00ff	aa	1f	00	fd	nv-BdIzc	ORA Abs,X	T0
10	00ff	18	1	CLC	00ff	aa	1f	00	fd	nv-BdIzc	ORA Abs,X	T1
10	00ff	18	1	CLC	00ff	aa	1f	00	fd	nv-BdIzc	ORA Abs,X	T1
11	0100	00	1		0100	aa	1f	00	fd	Nv-BdIzc	CLC	T0+T2
11	0100	00	1		0100	aa	1f	00	fd	Nv-BdIzc	CLC	T0+T2

ASL (1E)	Cycles: 7	Size: 3
Absolute, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC. Final=Address+X. Address.L = Final.L.
3.2	Read Address	Store as Operand.
4.1		Address.H = Final.H (fixes high byte of address).
4.2	Read Address	Store as Operand.
5.1		
5.2	Write Address	Write unmodified Operand.
6.1		Set C if high bit of Operand is set. Perform Operand=Operand << 1, set N and Z accordingly.
6.2	Write Address	Write modified Operand.
7.1		
7.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	00fc	1e	1	ASL Abs,X	00fc	aa	1f	00	fd	nv-BdIzc	JMP Abs	T1
5	00fc	1e	1	ASL Abs,X	00fc	aa	1f	00	fd	nv-BdIzc	JMP Abs	T1
6	00fd	ff	1		00fd	aa	1f	00	fd	nv-BdIzc	ASL Abs,X	T2
6	00fd	ff	1		00fd	aa	1f	00	fd	nv-BdIzc	ASL Abs,X	T2
7	00fe	00	1		00fe	aa	1f	00	fd	nv-BdIzc	ASL Abs,X	T3
7	00fe	00	1		00fe	aa	1f	00	fd	nv-BdIzc	ASL Abs,X	T3
8	001e	00	1		00ff	aa	1f	00	fd	nv-BdIzc	ASL Abs,X	T4
8	001e	00	1		00ff	aa	1f	00	fd	nv-BdIzc	ASL Abs,X	T4
9	011e	7f	1		00ff	aa	1f	00	fd	nv-BdIzc	ASL Abs,X	T5
9	011e	7f	1		00ff	aa	1f	00	fd	nv-BdIzc	ASL Abs,X	T5
10	011e	7f	0		00ff	aa	1f	00	fd	nv-BdIzc	ASL Abs,X	
10	011e	7f	0		00ff	aa	1f	00	fd	nv-BdIzc	ASL Abs,X	
11	011e	7f	0		00ff	aa	1f	00	fd	Nv-BdIzc	ASL Abs,X	T0
11	011e	fe	0		00ff	aa	1f	00	fd	Nv-BdIzc	ASL Abs,X	T0
12	00ff	18	1	CLC	00ff	aa	1f	00	fd	Nv-BdIzc	ASL Abs,X	T1
12	00ff	18	1	CLC	00ff	aa	1f	00	fd	Nv-BdIzc	ASL Abs,X	T1
13	0100	00	1		0100	aa	1f	00	fd	Nv-BdIzc	CLC	T0+T2
13	0100	00	1		0100	aa	1f	00	fd	Nv-BdIzc	CLC	T0+T2

SLO (1F)	Cycles: 7	Size: 3
Absolute, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC. Final=Address+X. Address.L = Final.L.
3.2	Read Address	Store as Operand.
4.1		Address.H = Final.H (fixes high byte of address).
4.2	Read Address	Store as Operand.
5.1		
5.2	Write Address	Write unmodified Operand.
6.1		* Set C if high bit of Operand is set. Perform Operand=Operand << 1, A = A Operand, set N and Z based off A .
6.2	Write Address	Write modified Operand.
7.1		
7.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
7	00fc	1f	1	unknown	00fc	00	1f	00	fd	nv-BdIZc	LDA #	T1
7	00fc	1f	1	unknown	00fc	00	1f	00	fd	nv-BdIZc	LDA #	T1
8	00fd	ff	1		00fd	00	1f	00	fd	nv-BdIZc	unknown	T2
8	00fd	ff	1		00fd	00	1f	00	fd	nv-BdIZc	unknown	T2
9	00fe	00	1		00fe	00	1f	00	fd	nv-BdIZc	unknown	T3
9	00fe	00	1		00fe	00	1f	00	fd	nv-BdIZc	unknown	T3
10	001e	00	1		00ff	00	1f	00	fd	nv-BdIZc	unknown	T4
10	001e	00	1		00ff	00	1f	00	fd	nv-BdIZc	unknown	T4
11	011e	fe	1		00ff	00	1f	00	fd	nv-BdIZc	unknown	T5
11	011e	fe	1		00ff	00	1f	00	fd	nv-BdIZc	unknown	T5
12	011e	fe	0		00ff	00	1f	00	fd	nv-BdIZc	unknown	
12	011e	fe	0		00ff	00	1f	00	fd	nv-BdIZc	unknown	
13	011e	fe	0		00ff	00	1f	00	fd	Nv-BdIzC	unknown	T0
13	011e	fc	0		00ff	00	1f	00	fd	Nv-BdIzC	unknown	T0
14	00ff	18	1	CLC	00ff	00	1f	00	fd	Nv-BdIzC	unknown	T1
14	00ff	18	1	CLC	00ff	00	1f	00	fd	Nv-BdIzC	unknown	T1
15	0100	00	1		0100	fc	1f	00	fd	Nv-BdIzC	CLC	T0+T2
15	0100	00	1		0100	fc	1f	00	fd	Nv-BdIzC	CLC	T0+T2

JSR (20)	Cycles: 6	Size: 3
Absolute		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read S \$0100	
3.1		
3.2	Write S \$0100	Push PC.H onto stack.
4.1		
4.2	Write (S -1) \$0100	Push PC.L onto stack.
5.1		
5.2	Read PC	Store as Address.H.
6.1		PC=Address. Decrease S by 2.
6.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	20	1	JSR Abs	0002	aa	1f	00	fd	nv-BdIzc	LDX #	T1
2	0002	20	1	JSR Abs	0002	aa	1f	00	fd	nv-BdIzc	LDX #	T1
3	0003	fa	1		0003	aa	1f	00	fd	nv-BdIzc	JSR Abs	T2
3	0003	fa	1		0003	aa	1f	00	fd	nv-BdIzc	JSR Abs	T2
4	01fd	00	1		0004	aa	1f	00	fa	nv-BdIzc	JSR Abs	T3
4	01fd	00	1		0004	aa	1f	00	fa	nv-BdIzc	JSR Abs	T3
5	01fd	00	0		0004	aa	1f	00	fa	nv-BdIzc	JSR Abs	T4
5	01fd	00	0		0004	aa	1f	00	fa	nv-BdIzc	JSR Abs	T4
6	01fc	00	0		0004	aa	1f	00	fa	nv-BdIzc	JSR Abs	T5
6	01fc	04	0		0004	aa	1f	00	fa	nv-BdIzc	JSR Abs	T5
7	0004	00	1		0004	aa	1f	00	fa	nv-BdIzc	JSR Abs	T0
7	0004	00	1		0004	aa	1f	00	fa	nv-BdIzc	JSR Abs	T0
8	00fa	a9	1	LDA #	00fa	aa	1f	00	fb	nv-BdIzc	JSR Abs	T1
8	00fa	a9	1	LDA #	00fa	aa	1f	00	fb	nv-BdIzc	JSR Abs	T1
9	00fb	00	1		00fb	aa	1f	00	fb	nv-BdIzc	LDA #	T0+T2
9	00fb	00	1		00fb	aa	1f	00	fb	nv-BdIzc	LDA #	T0+T2

AND (21)		Cycles: 6	Size: 2
Indirect, X (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Operand.	
2.1		Inc. PC.	
2.2		Read Operand	
3.1		Set Pointer to (Operand+X) & \$FF.	
3.2		Read Pointer	
4.1		Store to Address.L.	
4.2		Read (Pointer+1) & \$FF	
5.1		Store to Address.H.	
5.2		Read Address	
6.1	Read PC	* Perform A=A & Operand, set N and Z accordingly.	
6.2		Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0004	21	1	AND (zp,X)	0004	7f	0f	00	fd	nv-BdIzc	LDX #	T1
4	0004	21	1	AND (zp,X)	0004	7f	0f	00	fd	nv-BdIzc	LDX #	T1
5	0005	0e	1		0005	7f	0f	00	fd	nv-BdIzc	AND (zp,X)	T2
5	0005	0e	1		0005	7f	0f	00	fd	nv-BdIzc	AND (zp,X)	T2
6	000e	00	1		0006	7f	0f	00	fd	nv-BdIzc	AND (zp,X)	T3
6	000e	00	1		0006	7f	0f	00	fd	nv-BdIzc	AND (zp,X)	T3
7	001d	80	1		0006	7f	0f	00	fd	nv-BdIzc	AND (zp,X)	T4
7	001d	80	1		0006	7f	0f	00	fd	nv-BdIzc	AND (zp,X)	T4
8	001e	00	1		0006	7f	0f	00	fd	nv-BdIzc	AND (zp,X)	T5
8	001e	00	1		0006	7f	0f	00	fd	nv-BdIzc	AND (zp,X)	T5
9	0080	88	1		0006	7f	0f	00	fd	nv-BdIzc	AND (zp,X)	T0
9	0080	88	1		0006	7f	0f	00	fd	nv-BdIzc	AND (zp,X)	T0
10	0006	4c	1	JMP Abs	0006	7f	0f	00	fd	Nv-BdIzc	AND (zp,X)	T1
10	0006	4c	1	JMP Abs	0006	7f	0f	00	fd	Nv-BdIzc	AND (zp,X)	T1
11	0007	00	1		0007	08	0f	00	fd	nv-BdIzc	JMP Abs	T2
11	0007	00	1		0007	08	0f	00	fd	nv-BdIzc	JMP Abs	T2

JAM (22)	Cycles: ∞	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	
2.1		
2.2	Read \$FFFF	
3.1		
3.2	Read \$FFFE	
4.1		
4.2	Read \$FFFE	
5.1		Repeat 5.1 and 5.2 forever.
5.2	Read \$FFFF	

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	

RLA (23)	Cycles: 8	Size: 2
Indirect, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Operand.
2.1		Inc. PC.
2.2	Read Operand	
3.1		Set Pointer to (Operand+X) & \$FF.
3.2	Read Pointer	Store to Address.L.
4.1		
4.2	Read (Pointer+1) & \$FF	Store to Address.H.
5.1		
5.2	Read Address	Store as Operand.
6.1		
6.2	Write Address	Write unmodified Operand.
7.1		* Perform Operand=(Operand << 1) C, A=A & Operand, set N and Z based off A. Set C if high bit of Operand was set before the shift operation.
7.2	Write Address	Write modified Operand.
8.1		
8.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0004	23	1	unknown	0004	7f	0f	00	fd	nv-BdIzc	LDX #	T1
4	0004	23	1	unknown	0004	7f	0f	00	fd	nv-BdIzc	LDX #	T1
5	0005	0e	1		0005	7f	0f	00	fd	nv-BdIzc	unknown	T2
5	0005	0e	1		0005	7f	0f	00	fd	nv-BdIzc	unknown	T2
6	000e	00	1		0006	7f	0f	00	fd	nv-BdIzc	unknown	T3
6	000e	00	1		0006	7f	0f	00	fd	nv-BdIzc	unknown	T3
7	001d	80	1		0006	7f	0f	00	fd	nv-BdIzc	unknown	T4
7	001d	80	1		0006	7f	0f	00	fd	nv-BdIzc	unknown	T4
8	001e	00	1		0006	7f	0f	00	fd	nv-BdIzc	unknown	T5
8	001e	00	1		0006	7f	0f	00	fd	nv-BdIzc	unknown	T5
9	0080	88	1		0006	7f	0f	00	fd	nv-BdIzc	unknown	
9	0080	88	1		0006	7f	0f	00	fd	nv-BdIzc	unknown	
10	0080	88	0		0006	7f	0f	00	fd	nv-BdIzc	unknown	
10	0080	88	0		0006	7f	0f	00	fd	nv-BdIzc	unknown	
11	0080	88	0		0006	7f	0f	00	fd	nv-BdIzc	unknown	T0
11	0080	10	0		0006	7f	0f	00	fd	nv-BdIzc	unknown	T0
12	0006	4c	1	JMP Abs	0006	7f	0f	00	fd	nv-BdIzc	unknown	T1
12	0006	4c	1	JMP Abs	0006	7f	0f	00	fd	nv-BdIzc	unknown	T1
13	0007	00	1		0007	10	0f	00	fd	nv-BdIzc	JMP Abs	T2
13	0007	00	1		0007	10	0f	00	fd	nv-BdIzc	JMP Abs	T2

BIT (24)		Cycles: 3	Size: 2
Zero Page (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		V =(Operand & (1 << 6)) != 0. N =(Operand & (1 << 7)) != 0. Z =(Operand & A) == 0.	
3.2	Read PC	Store as OpCode.	
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	24	1	BIT zp	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
2	0002	24	1	BIT zp	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	BIT zp	T2
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	BIT zp	T2
4	0010	e8	1		0004	00	00	00	fd	nv-BdIZc	BIT zp	T0
4	0010	e8	1		0004	00	00	00	fd	nv-BdIZc	BIT zp	T0
5	0004	a9	1	LDA #	0004	00	00	00	fd	NV-BdIZc	BIT zp	T1
5	0004	a9	1	LDA #	0004	00	00	00	fd	NV-BdIZc	BIT zp	T1
6	0005	4c	1		0005	00	00	00	fd	NV-BdIZc	LDA #	T0+T2
6	0005	4c	1		0005	00	00	00	fd	NV-BdIZc	LDA #	T0+T2

AND (25)		Cycles: 3	Size: 2
Zero Page (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		* Perform A=A & Operand , set N and Z accordingly.	
3.2	Read PC	Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	25	1	AND zp	0002	01	00	00	fd	nv-BdIzc	LDA #	T1
2	0002	25	1	AND zp	0002	01	00	00	fd	nv-BdIzc	LDA #	T1
3	0003	10	1		0003	01	00	00	fd	nv-BdIzc	AND zp	T2
3	0003	10	1		0003	01	00	00	fd	nv-BdIzc	AND zp	T2
4	0010	e8	1		0004	01	00	00	fd	nv-BdIzc	AND zp	T0
4	0010	e8	1		0004	01	00	00	fd	nv-BdIzc	AND zp	T0
5	0004	a9	1	LDA #	0004	01	00	00	fd	Nv-BdIzc	AND zp	T1
5	0004	a9	1	LDA #	0004	01	00	00	fd	Nv-BdIzc	AND zp	T1
6	0005	4c	1		0005	00	00	00	fd	nv-BdIZc	LDA #	T0+T2
6	0005	4c	1		0005	00	00	00	fd	nv-BdIZc	LDA #	T0+T2

ROL (26)		Cycles: 5	Size: 2
Zero Page (Read/Modify/Write)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		Write unmodified Operand.	
3.2	Write Address		
4.1	Write Address	Perform Operand=(Operand << 1) C , set N and Z accordingly.	
4.2		Set C if high bit of Operand was set before the shift operation.	
5.1		Write modified Operand.	
5.2	Read PC	Store as OpCode.	
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
6	0004	26	1	ROL zp	0004	aa	01	00	fd	Nv-BdIZc	AND zp	T1
6	0004	26	1	ROL zp	0004	aa	01	00	fd	Nv-BdIZc	AND zp	T1
7	0005	4c	1		0005	a8	01	00	fd	Nv-BdIZc	ROL zp	T2
7	0005	4c	1		0005	a8	01	00	fd	Nv-BdIZc	ROL zp	T2
8	004c	00	1		0006	a8	01	00	fd	Nv-BdIZc	ROL zp	T3
8	004c	00	1		0006	a8	01	00	fd	Nv-BdIZc	ROL zp	T3
9	004c	00	0		0006	a8	01	00	fd	Nv-BdIZc	ROL zp	T4
9	004c	00	0		0006	a8	01	00	fd	Nv-BdIZc	ROL zp	T4
10	004c	00	0		0006	a8	01	00	fd	nv-BdIZc	ROL zp	T0
10	004c	00	0		0006	a8	01	00	fd	nv-BdIZc	ROL zp	T0
11	0006	45	1	EOR zp	0006	a8	01	00	fd	nv-BdIZc	ROL zp	T1
11	0006	45	1	EOR zp	0006	a8	01	00	fd	nv-BdIZc	ROL zp	T1
12	0007	00	1		0007	a8	01	00	fd	nv-BdIZc	EOR zp	T2
12	0007	00	1		0007	a8	01	00	fd	nv-BdIZc	EOR zp	T2

RLA (27)	Cycles: 5	Size: 2
Zero Page (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.
2.1		Inc. PC.
2.2	Read Address	Store as Operand.
3.1		
3.2	Write Address	Write unmodified Operand.
4.1		* Perform $\text{Operand} = (\text{Operand} \ll 1) \mid \mathbf{C}$, $\mathbf{A} = \mathbf{A} \& \text{Operand}$, set \mathbf{N} and \mathbf{Z} based off \mathbf{A} . Set \mathbf{C} if high bit of Operand was set before the shift operation.
4.2	Write Address	Write modified Operand.
5.1		
5.2	Read PC	Store as OpCode.
+X.1		\mathbf{A} applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
6	0004	27	1	unknown	0004	aa	01	00	fd	Nv-BdIZc	AND zp	T1
6	0004	27	1	unknown	0004	aa	01	00	fd	Nv-BdIZc	AND zp	T1
7	0005	4c	1		0005	a8	01	00	fd	Nv-BdIZc	unknown	T2
7	0005	4c	1		0005	a8	01	00	fd	Nv-BdIZc	unknown	T2
8	004c	00	1		0006	a8	01	00	fd	Nv-BdIZc	unknown	T3
8	004c	00	1		0006	a8	01	00	fd	Nv-BdIZc	unknown	T3
9	004c	00	0		0006	a8	01	00	fd	Nv-BdIZc	unknown	T4
9	004c	00	0		0006	a8	01	00	fd	Nv-BdIZc	unknown	T4
10	004c	00	0		0006	a8	01	00	fd	nv-BdIZc	unknown	T0
10	004c	00	0		0006	a8	01	00	fd	nv-BdIZc	unknown	T0
11	0006	45	1	EOR zp	0006	a8	01	00	fd	nv-BdIZc	unknown	T1
11	0006	45	1	EOR zp	0006	a8	01	00	fd	nv-BdIZc	unknown	T1
12	0007	00	1		0007	00	01	00	fd	nv-BdIZc	EOR zp	T2
12	0007	00	1		0007	00	01	00	fd	nv-BdIZc	EOR zp	T2

PLP (28)	Cycles: 4	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	To be discarded.
2.1		
2.2	Read S \$0100	
3.1		Inc. S .
3.2	Read S \$0100	Store as Operand.
4.1		P =(Operand & ~ B) M (the reserved bit).
4.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
3	0002	28	1	PLP	0002	aa	a6	00	fd	Nv-BdIzc	LDX zp	T1
3	0002	28	1	PLP	0002	aa	a6	00	fd	Nv-BdIzc	LDX zp	T1
4	0003	10	1		0003	aa	a6	00	fd	Nv-BdIzc	PLP	T2
4	0003	10	1		0003	aa	a6	00	fd	Nv-BdIzc	PLP	T2
5	01fd	00	1		0003	aa	a6	00	fd	Nv-BdIzc	PLP	T3
5	01fd	00	1		0003	aa	a6	00	fd	Nv-BdIzc	PLP	T3
6	01fe	01	1		0003	aa	a6	00	fe	Nv-BdIzc	PLP	T0
6	01fe	01	1		0003	aa	a6	00	fe	Nv-BdIzc	PLP	T0
7	0003	10	1	BPL	0003	aa	a6	00	fe	nv-BdizC	PLP	T1
7	0003	10	1	BPL	0003	aa	a6	00	fe	nv-BdizC	PLP	T1
8	0004	27	1		0004	aa	a6	00	fe	nv-BdizC	BPL	T2
8	0004	27	1		0004	aa	a6	00	fe	nv-BdizC	BPL	T2

AND (29)		Cycles: 2	Size: 2
Immediate (Read)			
Cycle	R/W		Desc
-X.2	Read PC		Store as OpCode.
1.1	Read PC		Inc. PC.
1.2			Store as Operand.
2.1			Inc. PC. * Perform A=A & Operand , set N and Z accordingly.
2.2			Store as OpCode.
+X.1			A and N, Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	29	1	AND #	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	29	1	AND #	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	33	1		0003	ff	00	00	fd	Nv-BdIzc	AND #	T0+T2
3	0003	33	1		0003	ff	00	00	fd	Nv-BdIzc	AND #	T0+T2
4	0004	a6	1	LDX zp	0004	ff	00	00	fd	nv-BdIzc	AND #	T1
4	0004	a6	1	LDX zp	0004	ff	00	00	fd	nv-BdIzc	AND #	T1
5	0005	4c	1		0005	33	00	00	fd	nv-BdIzc	LDX zp	T2
5	0005	4c	1		0005	33	00	00	fd	nv-BdIzc	LDX zp	T2

ROL (2A)		Cycles: 2	Size: 1
Implied (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2			
2.1		Perform Operand=(Operand << 1) C , set N and Z accordingly.	
2.2		Set C if high bit of Operand was set before the shift operation.	
+X.1	Read PC	Store as OpCode.	

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
3	0002	2a	1	ROL	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
3	0002	2a	1	ROL	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
4	0003	56	1		0003	aa	00	00	fd	NV-BdIzc	ROL	T0+T2
4	0003	56	1		0003	aa	00	00	fd	NV-BdIzc	ROL	T0+T2
5	0003	56	1	LSR zp,X	0003	aa	00	00	fd	NV-BdIzc	ROL	T1
5	0003	56	1	LSR zp,X	0003	aa	00	00	fd	NV-BdIzc	ROL	T1
6	0004	01	1		0004	54	00	00	fd	nV-BdIzc	LSR zp,X	T2
6	0004	01	1		0004	54	00	00	fd	nV-BdIzc	LSR zp,X	T2

ANC (2B)		Cycles: 2	Size: 2
Immediate (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Operand.	
2.1		Inc. PC. * Perform A=A & Operand , set C , N , and Z accordingly (C is set using logic for N).	
2.2		Store as OpCode.	
+X.1		** A and C , N , Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

** Visual6502 incorrectly omits the update of A; flags are updated based on the incorrect A value.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	2b	1	unknown	0002	0f	00	00	fd	nv-BdIzc	LDA #	T1
2	0002	2b	1	unknown	0002	0f	00	00	fd	nv-BdIzc	LDA #	T1
3	0003	33	1		0003	0f	00	00	fd	nv-BdIzc	unknown	T0+T2
3	0003	33	1		0003	0f	00	00	fd	nv-BdIzc	unknown	T0+T2
4	0004	a6	1	LDX zp	0004	0f	00	00	fd	nv-BdIzc	unknown	T1
4	0004	a6	1	LDX zp	0004	0f	00	00	fd	nv-BdIzc	unknown	T1
5	0005	4c	1		0005	0f	00	00	fd	nv-BdIzc	LDX zp	T2
5	0005	4c	1		0005	0f	00	00	fd	nv-BdIzc	LDX zp	T2

ANC (ANC2, ANA, ANB)

Type: Combination of an immediate and an implied command (Sub-instructions: AND, ASL/ROL)

Op.	Mnemonic	Function	Size	Cycles	N	V	-	B	D	I	Z	C
\$0B	ANC #imm	A = A & #{imm}	2	2	o						o	o
\$2B	ANC #imm	A = A & #{imm}	2	2	o						o	o

Operation: ANDs the contents of the A register with an immediate value and then moves bit 7 of A into the Carry flag.

- This opcode works basically identically to AND #imm. except that the Carry flag is set to the same state that the Negative flag is set to. (bit 7 is put into the carry, as if the ASL/ROL would have been executed)

— NoMoreSecrets-NMOS6510UnintendedOpcodes-20232412.pdf

BIT (2C)		Cycles: 4	Size: 3
Absolute (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.L.	
2.1		Inc. PC.	
2.2		Store as Address.H.	
3.1		Inc. PC.	
3.2	Read Address	Store as Operand.	
4.1	Read PC	V=(Operand & (1 << 6)) != 0.	
		N=(Operand & (1 << 7)) != 0.	
4.2		Z=(Operand & A) == 0.	
+X.1		Store as OpCode.	

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	2c	1	BIT Abs	0002	0f	00	00	fd	nv-BdIzc	LDA #	T1
2	0002	2c	1	BIT Abs	0002	0f	00	00	fd	nv-BdIzc	LDA #	T1
3	0003	10	1		0003	0f	00	00	fd	nv-BdIzc	BIT Abs	T2
3	0003	10	1		0003	0f	00	00	fd	nv-BdIzc	BIT Abs	T2
4	0004	00	1		0004	0f	00	00	fd	nv-BdIzc	BIT Abs	T3
4	0004	00	1		0004	0f	00	00	fd	nv-BdIzc	BIT Abs	T3
5	0010	e8	1		0005	0f	00	00	fd	nv-BdIzc	BIT Abs	T0
5	0010	e8	1		0005	0f	00	00	fd	nv-BdIzc	BIT Abs	T0
6	0005	4c	1	JMP Abs	0005	0f	00	00	fd	NV-BdIzc	BIT Abs	T1
6	0005	4c	1	JMP Abs	0005	0f	00	00	fd	NV-BdIzc	BIT Abs	T1
7	0006	45	1		0006	0f	00	00	fd	NV-BdIzc	JMP Abs	T2
7	0006	45	1		0006	0f	00	00	fd	NV-BdIzc	JMP Abs	T2

AND (2D)		Cycles: 4	Size: 3
Absolute (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.L.	
2.1		Inc. PC.	
2.2		Store as Address.H.	
3.1		Inc. PC.	
3.2	Read Address	Store as Operand.	
4.1	Read PC	* Perform A=A & Operand, set N and Z accordingly.	
4.2		Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	2d	1	AND Abs	0002	0f	00	00	fd	nv-BdIzc	LDA #	T1
2	0002	2d	1	AND Abs	0002	0f	00	00	fd	nv-BdIzc	LDA #	T1
3	0003	10	1		0003	0f	00	00	fd	nv-BdIzc	AND Abs	T2
3	0003	10	1		0003	0f	00	00	fd	nv-BdIzc	AND Abs	T2
4	0004	00	1		0004	0f	00	00	fd	nv-BdIzc	AND Abs	T3
4	0004	00	1		0004	0f	00	00	fd	nv-BdIzc	AND Abs	T3
5	0010	88	1		0005	0f	00	00	fd	nv-BdIzc	AND Abs	T0
5	0010	88	1		0005	0f	00	00	fd	nv-BdIzc	AND Abs	T0
6	0005	4c	1	JMP Abs	0005	0f	00	00	fd	Nv-BdIzc	AND Abs	T1
6	0005	4c	1	JMP Abs	0005	0f	00	00	fd	Nv-BdIzc	AND Abs	T1
7	0006	45	1		0006	08	00	00	fd	nv-BdIzc	JMP Abs	T2
7	0006	45	1		0006	08	00	00	fd	nv-BdIzc	JMP Abs	T2

ROL (2E)	Cycles: 6	Size: 3
Absolute (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC.
3.2	Read Address	Store as Operand.
4.1		
4.2	Write Address	Write unmodified Operand.
5.1		Perform Operand=(Operand << 1) C , set N and Z accordingly. Set C if high bit of Operand was set before the shift operation.
5.2	Write Address	Write modified Operand.
6.1		
6.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
3	0002	2e	1	ROL Abs	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
3	0002	2e	1	ROL Abs	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	ROL Abs	T2
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	ROL Abs	T2
5	0004	01	1		0004	aa	00	00	fd	NV-BdIzc	ROL Abs	T3
5	0004	01	1		0004	aa	00	00	fd	NV-BdIzc	ROL Abs	T3
6	01fb	8f	1		0005	aa	00	00	fd	NV-BdIzc	ROL Abs	T4
6	01fb	8f	1		0005	aa	00	00	fd	NV-BdIzc	ROL Abs	T4
7	01fb	8f	0		0005	aa	00	00	fd	NV-BdIzc	ROL Abs	T5
7	01fb	8f	0		0005	aa	00	00	fd	NV-BdIzc	ROL Abs	T5
8	01fb	8f	0		0005	aa	00	00	fd	nV-BdIzC	ROL Abs	T0
8	01fb	1e	0		0005	aa	00	00	fd	nV-BdIzC	ROL Abs	T0
9	0005	50	1	BVC	0005	aa	00	00	fd	nV-BdIzC	ROL Abs	T1
9	0005	50	1	BVC	0005	aa	00	00	fd	nV-BdIzC	ROL Abs	T1
10	0006	2e	1		0006	aa	00	00	fd	nV-BdIzC	BVC	T2
10	0006	2e	1		0006	aa	00	00	fd	nV-BdIzC	BVC	T2

RLA (2F)	Cycles: 6	Size: 3
Absolute (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC.
3.2	Read Address	Store as Operand.
4.1		
4.2	Write Address	Write unmodified Operand.
5.1		* Perform Operand=(Operand << 1) C , A=A & Operand, set N and Z based off A . Set C if high bit of Operand was set before the shift operation.
5.2	Write Address	Write modified Operand.
6.1		
6.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	2f	1	unknown	0003	0f	00	00	fd	nv-BdIzC	SEC	T1
4	0003	2f	1	unknown	0003	0f	00	00	fd	nv-BdIzC	SEC	T1
5	0004	2e	1		0004	0f	00	00	fd	nv-BdIzC	unknown	T2
5	0004	2e	1		0004	0f	00	00	fd	nv-BdIzC	unknown	T2
6	0005	00	1		0005	0f	00	00	fd	nv-BdIzC	unknown	T3
6	0005	00	1		0005	0f	00	00	fd	nv-BdIzC	unknown	T3
7	002e	7f	1		0006	0f	00	00	fd	nv-BdIzC	unknown	T4
7	002e	7f	1		0006	0f	00	00	fd	nv-BdIzC	unknown	T4
8	002e	7f	0		0006	0f	00	00	fd	nv-BdIzC	unknown	T5
8	002e	7f	0		0006	0f	00	00	fd	nv-BdIzC	unknown	T5
9	002e	7f	0		0006	0f	00	00	fd	Nv-BdIzc	unknown	T0
9	002e	ff	0		0006	0f	00	00	fd	Nv-BdIzc	unknown	T0
10	0006	00	1	BRK	0006	0f	00	00	fd	Nv-BdIzc	unknown	T1
10	0006	00	1	BRK	0006	0f	00	00	fd	Nv-BdIzc	unknown	T1
11	0007	00	1		0007	0f	00	00	fd	nv-BdIzc	BRK	T2
11	0007	00	1		0007	0f	00	00	fd	nv-BdIzc	BRK	T2

BMI (30)	Cycles: 2-4	Size: 2
Branch Relative		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1	Read PC	Inc. PC. Check condition (N == 1).
1.2		Store as Operand. Treat as signed 16-bit (Op16=i16(i8(Operand))).
2.1		Inc. PC. If not jumping, end (next half-cycle is 4.2)
2.2		If (PC+Op16).H != PC.H, end after PC.L fix (next half-cycle is 4.2). PC.L=PC.L+Operand.
3.1		
3.2		
4.1		PC.H=previous “(PC+Op16).H” value.
4.2		Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	30	1	BMI	0003	0f	00	00	fd	Nv-BdIzC	SEC	T1
4	0003	30	1	BMI	0003	0f	00	00	fd	Nv-BdIzC	SEC	T1
5	0004	2e	1		0004	0f	00	00	fd	Nv-BdIzC	BMI	T2
5	0004	2e	1		0004	0f	00	00	fd	Nv-BdIzC	BMI	T2
6	0005	00	1	BRK	0005	0f	00	00	fd	Nv-BdIzC	BMI	
6	0005	00	1	BRK	0005	0f	00	00	fd	Nv-BdIzC	BMI	
7	0006	00	1		0006	0f	00	00	fd	Nv-BdIzC	BRK	T2
7	0006	00	1		0006	0f	00	00	fd	Nv-BdIzC	BRK	T2

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	30	1	BMI	0003	ff	00	00	fd	Nv-BdIzC	SEC	T1
4	0003	30	1	BMI	0003	ff	00	00	fd	Nv-BdIzC	SEC	T1
5	0004	2e	1		0004	ff	00	00	fd	Nv-BdIzC	BMI	T2
5	0004	2e	1		0004	ff	00	00	fd	Nv-BdIzC	BMI	T2
6	0005	00	1		0005	ff	00	00	fd	Nv-BdIzC	BMI	T3
6	0005	00	1		0005	ff	00	00	fd	Nv-BdIzC	BMI	T3
7	0033	00	1	BRK	0033	ff	00	00	fd	Nv-BdIzC	BMI	
7	0033	00	1	BRK	0033	ff	00	00	fd	Nv-BdIzC	BMI	
8	0034	00	1		0034	ff	00	00	fd	Nv-BdIzC	BRK	T2
8	0034	00	1		0034	ff	00	00	fd	Nv-BdIzC	BRK	T2

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
7	00fc	30	1	BMI	00fc	ff	00	00	fd	Nv-BdIzC	JMP Abs	T1
7	00fc	30	1	BMI	00fc	ff	00	00	fd	Nv-BdIzC	JMP Abs	T1
8	00fd	2e	1		00fd	ff	00	00	fd	Nv-BdIzC	BMI	T2
8	00fd	2e	1		00fd	ff	00	00	fd	Nv-BdIzC	BMI	T2
9	00fe	00	1		00fe	ff	00	00	fd	Nv-BdIzC	BMI	T3
9	00fe	00	1		00fe	ff	00	00	fd	Nv-BdIzC	BMI	T3
10	002c	00	1		002c	ff	00	00	fd	Nv-BdIzC	BMI	T0
10	002c	00	1		002c	ff	00	00	fd	Nv-BdIzC	BMI	T0
11	012c	00	1	BRK	012c	ff	00	00	fd	Nv-BdIzC	BMI	T1
11	012c	00	1	BRK	012c	ff	00	00	fd	Nv-BdIzC	BMI	T1
12	012d	00	1		012d	ff	00	00	fd	Nv-BdIzC	BRK	T2
12	012d	00	1		012d	ff	00	00	fd	Nv-BdIzC	BRK	T2

AND (31)		Cycles: 5-6	Size: 2
Indirect, Y (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Pointer.	
2.1		Inc. PC.	
2.2		Store to Address.L.	
3.1	Read Pointer		
3.2		Store to Address.H.	
4.1		Final=Address+Y. Address.L = Final.L.	
4.2		Store as Operand. If Address.H == Final.H, skip the next cycle (next half-cycle is 6.1).	
5.1	Read Address	Address.H = Final.H (fixes high byte of address).	
5.2		Store as Operand.	
6.1		* Perform A=A & Operand , set N and Z accordingly.	
6.2		Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	31	1	AND (zp),Y	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
4	0003	31	1	AND (zp),Y	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	AND (zp),Y	T2
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	AND (zp),Y	T2
6	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	AND (zp),Y	T3
6	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	AND (zp),Y	T3
7	00fd	2e	1		0005	aa	00	ff	fd	Nv-BdIzC	AND (zp),Y	T4
7	00fd	2e	1		0005	aa	00	ff	fd	Nv-BdIzC	AND (zp),Y	T4
8	2e2f	00	1		0005	aa	00	ff	fd	Nv-BdIzC	AND (zp),Y	T5
8	2e2f	00	1		0005	aa	00	ff	fd	Nv-BdIzC	AND (zp),Y	T5
9	2f2f	00	1		0005	aa	00	ff	fd	Nv-BdIzC	AND (zp),Y	T0
9	2f2f	00	1		0005	aa	00	ff	fd	Nv-BdIzC	AND (zp),Y	T0
10	0005	00	1	BRK	0005	aa	00	ff	fd	nv-BdIZC	AND (zp),Y	T1
10	0005	00	1	BRK	0005	aa	00	ff	fd	nv-BdIZC	AND (zp),Y	T1
11	0006	00	1		0006	00	00	ff	fd	nv-BdIZC	BRK	T2
11	0006	00	1		0006	00	00	ff	fd	nv-BdIZC	BRK	T2

JAM (32)	Cycles: ∞	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	
2.1		
2.2	Read \$FFFF	
3.1		
3.2	Read \$FFFE	
4.1		
4.2	Read \$FFFE	
5.1		Repeat 5.1 and 5.2 forever.
5.2	Read \$FFFF	

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	

RLA (33)	Cycles: 8	Size: 2
Indirect, Y (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Pointer.
2.1		Inc. PC.
2.2	Read Pointer	Store to Address.L.
3.1		
3.2	Read (Pointer+1) & \$FF	Store to Address.H.
4.1		Final=Address+Y. Address.L = Final.L.
4.2	Read Address	Store as Operand.
5.1		Address.H = Final.H (fixes high byte of address).
5.2	Read Address	Store as Operand.
6.1		
6.2	Write Address	Write unmodified Operand.
7.1		* Perform Operand=(Operand << 1) C , A = A & Operand, set N and Z based off A . Set C if high bit of Operand was set before the shift operation.
7.2	Write Address	Write modified Operand.
8.1		
8.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	33	1	unknown	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
4	0003	33	1	unknown	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	unknown	T2
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	unknown	T2
6	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T3
6	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T3
7	00fd	2e	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T4
7	00fd	2e	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T4
8	2e2f	00	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T5
8	2e2f	00	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T5
9	2f2f	00	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	
9	2f2f	00	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	
10	2f2f	00	0		0005	aa	00	ff	fd	Nv-BdIzC	unknown	
10	2f2f	00	0		0005	aa	00	ff	fd	Nv-BdIzC	unknown	
11	2f2f	00	0		0005	aa	00	ff	fd	nv-BdIzc	unknown	T0
11	2f2f	01	0		0005	aa	00	ff	fd	nv-BdIzc	unknown	T0
12	0005	00	1	BRK	0005	aa	00	ff	fd	nv-BdIzc	unknown	T1
12	0005	00	1	BRK	0005	aa	00	ff	fd	nv-BdIzc	unknown	T1
13	0006	00	1		0006	00	00	ff	fd	nv-BdIzc	BRK	T2
13	0006	00	1		0006	00	00	ff	fd	nv-BdIzc	BRK	T2

NOP (34)		Cycles: 4	Size: 2
Zero Page, X (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC Read PC Read Pointer Read Address Read PC	Inc. PC.	
1.2		Store as Pointer.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		Set Address to (Pointer+X) & \$FF.	
3.2		Store as Operand.	
4.1		Store as OpCode.	
4.2			
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	34	1	unknown	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
4	0003	34	1	unknown	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	unknown	T2
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	unknown	T2
6	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T3
6	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T3
7	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T0
7	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T0
8	0005	00	1	BRK	0005	aa	00	ff	fd	Nv-BdIzC	unknown	T1
8	0005	00	1	BRK	0005	aa	00	ff	fd	Nv-BdIzC	unknown	T1
9	0006	00	1		0006	aa	00	ff	fd	Nv-BdIzC	BRK	T2
9	0006	00	1		0006	aa	00	ff	fd	Nv-BdIzC	BRK	T2

AND (35)		Cycles: 4	Size: 2
Zero Page, X (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Pointer.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		Set Address to (Pointer+X) & \$FF.	
3.2	Read Address	Store as Operand.	
4.1	Read PC	* Perform A=A & Operand, set N and Z accordingly.	
4.2		Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	35	1	AND zp,X	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
4	0003	35	1	AND zp,X	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	AND zp,X	T2
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	AND zp,X	T2
6	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	AND zp,X	T3
6	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	AND zp,X	T3
7	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	AND zp,X	T0
7	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	AND zp,X	T0
8	0005	00	1	BRK	0005	aa	00	ff	fd	nv-BdIzC	AND zp,X	T1
8	0005	00	1	BRK	0005	aa	00	ff	fd	nv-BdIzC	AND zp,X	T1
9	0006	00	1		0006	20	00	ff	fd	nv-BdIzC	BRK	T2
9	0006	00	1		0006	20	00	ff	fd	nv-BdIzC	BRK	T2

ROL (36)	Cycles: 6	Size: 2
Zero Page, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Pointer.
2.1		Inc. PC.
2.2	Read Pointer	Store as Operand.
3.1		Set Address to (Pointer+X) & \$FF.
3.2	Read Address	Store as Operand.
4.1		
4.2	Write Address	Write unmodified Operand.
5.1		Perform Operand=(Operand << 1) C, set N and Z accordingly. Set C if high bit of Operand was set before the shift operation.
5.2	Write Address	Write modified Operand.
6.1		
6.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	36	1	ROL zp,X	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
4	0003	36	1	ROL zp,X	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	ROL zp,X	T2
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	ROL zp,X	T2
6	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	ROL zp,X	T3
6	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	ROL zp,X	T3
7	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	ROL zp,X	T4
7	00fc	30	1		0005	aa	00	ff	fd	Nv-BdIzC	ROL zp,X	T4
8	00fc	30	0		0005	aa	00	ff	fd	Nv-BdIzC	ROL zp,X	T5
8	00fc	30	0		0005	aa	00	ff	fd	Nv-BdIzC	ROL zp,X	T5
9	00fc	30	0		0005	aa	00	ff	fd	nv-BdIzc	ROL zp,X	T0
9	00fc	61	0		0005	aa	00	ff	fd	nv-BdIzc	ROL zp,X	T0
10	0005	00	1	BRK	0005	aa	00	ff	fd	nv-BdIzc	ROL zp,X	T1
10	0005	00	1	BRK	0005	aa	00	ff	fd	nv-BdIzc	ROL zp,X	T1
11	0006	00	1		0006	aa	00	ff	fd	nv-BdIzc	BRK	T2
11	0006	00	1		0006	aa	00	ff	fd	nv-BdIzc	BRK	T2

RLA (37)	Cycles: 6	Size: 2
Zero Page, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Pointer.
2.1		Inc. PC.
2.2	Read Pointer	Store as Operand.
3.1		Set Address to (Pointer+X) & \$FF.
3.2	Read Address	Store as Operand.
4.1		
4.2	Write Address	Write unmodified Operand.
5.1		* Perform Operand=(Operand << 1) C, A=A & Operand, set N and Z based off A. Set C if high bit of Operand was set before the shift operation.
5.2	Write Address	Write modified Operand.
6.1		
6.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	37	1	unknown	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
4	0003	37	1	unknown	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	unknown	T2
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	unknown	T2
6	00fc	61	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T3
6	00fc	61	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T3
7	00fc	61	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T4
7	00fc	61	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T4
8	00fc	61	0		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T5
8	00fc	61	0		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T5
9	00fc	61	0		0005	aa	00	ff	fd	Nv-BdIzc	unknown	T0
9	00fc	c3	0		0005	aa	00	ff	fd	Nv-BdIzc	unknown	T0
10	0005	00	1	BRK	0005	aa	00	ff	fd	Nv-BdIzc	unknown	T1
10	0005	00	1	BRK	0005	aa	00	ff	fd	Nv-BdIzc	unknown	T1
11	0006	00	1		0006	82	00	ff	fd	Nv-BdIzc	BRK	T2
11	0006	00	1		0006	82	00	ff	fd	Nv-BdIzc	BRK	T2

SEC (38)	Cycles: 2	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	To be discarded.
2.1		Set the C status bit.
2.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	38	1	SEC	0002	aa	00	ff	fd	Nv-BdIzc	LDY #	T1
2	0002	38	1	SEC	0002	aa	00	ff	fd	Nv-BdIzc	LDY #	T1
3	0003	39	1		0003	aa	00	ff	fd	Nv-BdIzc	SEC	T0+T2
3	0003	39	1		0003	aa	00	ff	fd	Nv-BdIzc	SEC	T0+T2
4	0003	39	1	AND Abs,Y	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
4	0003	39	1	AND Abs,Y	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	AND Abs,Y	T2
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	AND Abs,Y	T2

AND (39)		Cycles: 4-5	Size: 3
Absolute, Y (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1		Inc. PC.	
1.2	Read PC	Store as Address.L.	
2.1		Inc. PC.	
2.2	Read PC	Store as Address.H.	
3.1		Inc. PC. Final=Address+Y. Address.L = Final.L.	
3.2	Read Address	Store as Operand. If Address.H == Final.H, skip the next cycle (next half-cycle is 5.1).	
4.1		Address.H = Final.H (fixes high byte of address).	
4.2	Read Address	Store as Operand.	
5.1		* Perform A=A & Operand , set N and Z accordingly.	
5.2	Read PC	Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	39	1	AND Abs,Y	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
4	0003	39	1	AND Abs,Y	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	AND Abs,Y	T2
5	0004	fc	1		0004	aa	00	ff	fd	Nv-BdIzC	AND Abs,Y	T2
6	0005	00	1		0005	aa	00	ff	fd	Nv-BdIzC	AND Abs,Y	T3
6	0005	00	1		0005	aa	00	ff	fd	Nv-BdIzC	AND Abs,Y	T3
7	00fb	00	1		0006	aa	00	ff	fd	Nv-BdIzC	AND Abs,Y	T4
7	00fb	00	1		0006	aa	00	ff	fd	Nv-BdIzC	AND Abs,Y	T4
8	01fb	00	1		0006	aa	00	ff	fd	Nv-BdIzC	AND Abs,Y	T0
8	01fb	00	1		0006	aa	00	ff	fd	Nv-BdIzC	AND Abs,Y	T0
9	0006	00	1	BRK	0006	aa	00	ff	fd	nv-BdIzC	AND Abs,Y	T1
9	0006	00	1	BRK	0006	aa	00	ff	fd	nv-BdIzC	AND Abs,Y	T1
10	0007	00	1		0007	00	00	ff	fd	nv-BdIzC	BRK	T2
10	0007	00	1		0007	00	00	ff	fd	nv-BdIzC	BRK	T2

NOP (3A)		Cycles: 2	Size: 1
Implied			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		To be discarded.	
2.1			
2.2		Store as OpCode.	
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	3a	1	unknown	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
4	0003	3a	1	unknown	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
5	0004	a9	1		0004	aa	00	ff	fd	Nv-BdIzC	unknown	T0+T2
5	0004	a9	1		0004	aa	00	ff	fd	Nv-BdIzC	unknown	T0+T2
6	0004	a9	1	LDA #	0004	aa	00	ff	fd	Nv-BdIzC	unknown	T1
6	0004	a9	1	LDA #	0004	aa	00	ff	fd	Nv-BdIzC	unknown	T1
7	0005	00	1		0005	aa	00	ff	fd	Nv-BdIzC	LDA #	T0+T2
7	0005	00	1		0005	aa	00	ff	fd	Nv-BdIzC	LDA #	T0+T2

RLA (3B)	Cycles: 7	Size: 3
Absolute, Y (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC. Final=Address+Y. Address.L = Final.L.
3.2	Read Address	Store as Operand.
4.1		Address.H = Final.H (fixes high byte of address).
4.2	Read Address	Store as Operand.
5.1		
5.2	Write Address	Write unmodified Operand.
6.1		* Perform Operand=(Operand << 1) C, A=A & Operand, set N and Z based off A. Set C if high bit of Operand was set before the shift operation.
6.2	Write Address	Write modified Operand.
7.1		
7.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	3b	1	unknown	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
4	0003	3b	1	unknown	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
5	0004	a9	1		0004	aa	00	ff	fd	Nv-BdIzC	unknown	T2
5	0004	a9	1		0004	aa	00	ff	fd	Nv-BdIzC	unknown	T2
6	0005	00	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T3
6	0005	00	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T3
7	00a8	00	1		0006	aa	00	ff	fd	Nv-BdIzC	unknown	T4
7	00a8	00	1		0006	aa	00	ff	fd	Nv-BdIzC	unknown	T4
8	01a8	00	1		0006	aa	00	ff	fd	Nv-BdIzC	unknown	T5
8	01a8	00	1		0006	aa	00	ff	fd	Nv-BdIzC	unknown	T5
9	01a8	00	0		0006	aa	00	ff	fd	Nv-BdIzC	unknown	
9	01a8	00	0		0006	aa	00	ff	fd	Nv-BdIzC	unknown	
10	01a8	00	0		0006	aa	00	ff	fd	nv-BdIzc	unknown	T0
10	01a8	01	0		0006	aa	00	ff	fd	nv-BdIzc	unknown	T0
11	0006	00	1	BRK	0006	aa	00	ff	fd	nv-BdIzc	unknown	T1
11	0006	00	1	BRK	0006	aa	00	ff	fd	nv-BdIzc	unknown	T1
12	0007	00	1		0007	00	00	ff	fd	nv-BdIzc	BRK	T2
12	0007	00	1		0007	00	00	ff	fd	nv-BdIzc	BRK	T2

NOP (3C) Cycles: 4-5		Size: 3
Absolute, X (Read)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC. Final=Address+X. Address.L = Final.L.
3.2	Read Address	Store as Operand. If Address.H == Final.H, skip the next cycle (next half-cycle is 5.1).
4.1		Address.H = Final.H (fixes high byte of address).
4.2	Read Address	Store as Operand.
5.1		
5.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	3c	1	unknown	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
4	0003	3c	1	unknown	0003	aa	00	ff	fd	Nv-BdIzC	SEC	T1
5	0004	a9	1		0004	aa	00	ff	fd	Nv-BdIzC	unknown	T2
5	0004	a9	1		0004	aa	00	ff	fd	Nv-BdIzC	unknown	T2
6	0005	00	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T3
6	0005	00	1		0005	aa	00	ff	fd	Nv-BdIzC	unknown	T3
7	00a9	00	1		0006	aa	00	ff	fd	Nv-BdIzC	unknown	T0
7	00a9	00	1		0006	aa	00	ff	fd	Nv-BdIzC	unknown	T0
8	0006	00	1	BRK	0006	aa	00	ff	fd	Nv-BdIzC	unknown	T1
8	0006	00	1	BRK	0006	aa	00	ff	fd	Nv-BdIzC	unknown	T1
9	0007	00	1		0007	aa	00	ff	fd	Nv-BdIzC	BRK	T2
9	0007	00	1		0007	aa	00	ff	fd	Nv-BdIzC	BRK	T2

AND (3D) Cycles: 4-5		Size: 3
Absolute, X (Read)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1	Read PC	Inc. PC.
1.2		Store as Address.L.
2.1		Inc. PC.
2.2		Store as Address.H.
3.1		Inc. PC. Final=Address+X. Address.L = Final.L.
3.2	Read Address	Store as Operand. If Address.H == Final.H, skip the next cycle (next half-cycle is 5.1).
4.1	Read Address	Address.H = Final.H (fixes high byte of address).
4.2		Store as Operand.
5.1		* Perform A=A & Operand , set N and Z accordingly.
5.2	Read PC	Store as OpCode.
+X.1		A and N, Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	0003	3d	1	AND Abs,X	0003	aa	33	00	fd	nv-BdIzC	SEC	T1
5	0003	3d	1	AND Abs,X	0003	aa	33	00	fd	nv-BdIzC	SEC	T1
6	0004	a9	1		0004	aa	33	00	fd	nv-BdIzC	AND Abs,X	T2
6	0004	a9	1		0004	aa	33	00	fd	nv-BdIzC	AND Abs,X	T2
7	0005	00	1		0005	aa	33	00	fd	nv-BdIzC	AND Abs,X	T3
7	0005	00	1		0005	aa	33	00	fd	nv-BdIzC	AND Abs,X	T3
8	00dc	f0	1		0006	aa	33	00	fd	nv-BdIzC	AND Abs,X	T0
8	00dc	f0	1		0006	aa	33	00	fd	nv-BdIzC	AND Abs,X	T0
9	0006	00	1	BRK	0006	aa	33	00	fd	Nv-BdIzC	AND Abs,X	T1
9	0006	00	1	BRK	0006	aa	33	00	fd	Nv-BdIzC	AND Abs,X	T1
10	0007	00	1		0007	a0	33	00	fd	Nv-BdIzC	BRK	T2
10	0007	00	1		0007	a0	33	00	fd	Nv-BdIzC	BRK	T2

ROL (3E)		Cycles: 7	Size: 3
Absolute, X (Read/Modify/Write)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.L.	
2.1	Read PC	Inc. PC.	
2.2		Store as Address.H.	
3.1	Read Address	Inc. PC. Final=Address+X. Address.L = Final.L.	
3.2		Store as Operand.	
4.1	Read Address	Address.H = Final.H (fixes high byte of address).	
4.2		Store as Operand.	
5.1	Write Address	Write unmodified Operand.	
5.2		Perform Operand=(Operand << 1) C, set N and Z accordingly.	
6.1	Write Address	Set C if high bit of Operand was set before the shift operation.	
6.2		Write modified Operand.	
7.1	Read PC		
7.2		Store as OpCode.	
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	0003	3e	1	ROL Abs,X	0003	aa	33	00	fd	nv-BdIzC	SEC	T1
5	0003	3e	1	ROL Abs,X	0003	aa	33	00	fd	nv-BdIzC	SEC	T1
6	0004	a9	1		0004	aa	33	00	fd	nv-BdIzC	ROL Abs,X	T2
6	0004	a9	1		0004	aa	33	00	fd	nv-BdIzC	ROL Abs,X	T2
7	0005	00	1		0005	aa	33	00	fd	nv-BdIzC	ROL Abs,X	T3
7	0005	00	1		0005	aa	33	00	fd	nv-BdIzC	ROL Abs,X	T3
8	00dc	f0	1		0006	aa	33	00	fd	nv-BdIzC	ROL Abs,X	T4
8	00dc	f0	1		0006	aa	33	00	fd	nv-BdIzC	ROL Abs,X	T4
9	00dc	f0	1		0006	aa	33	00	fd	nv-BdIzC	ROL Abs,X	T5
9	00dc	f0	1		0006	aa	33	00	fd	nv-BdIzC	ROL Abs,X	T5
10	00dc	f0	0		0006	aa	33	00	fd	nv-BdIzC	ROL Abs,X	
10	00dc	f0	0		0006	aa	33	00	fd	nv-BdIzC	ROL Abs,X	
11	00dc	f0	0		0006	aa	33	00	fd	Nv-BdIzC	ROL Abs,X	T0
11	00dc	e1	0		0006	aa	33	00	fd	Nv-BdIzC	ROL Abs,X	T0
12	0006	00	1	BRK	0006	aa	33	00	fd	Nv-BdIzC	ROL Abs,X	T1
12	0006	00	1	BRK	0006	aa	33	00	fd	Nv-BdIzC	ROL Abs,X	T1
13	0007	00	1		0007	aa	33	00	fd	Nv-BdIzC	BRK	T2
13	0007	00	1		0007	aa	33	00	fd	Nv-BdIzC	BRK	T2

RLA (3F)		Cycles: 7	Size: 3
Absolute, X (Read/Modify/Write)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.L.	
2.1	Read PC	Inc. PC.	
2.2		Store as Address.H.	
3.1	Read Address	Inc. PC. Final=Address+X. Address.L = Final.L.	
3.2		Store as Operand.	
4.1	Read Address	Address.H = Final.H (fixes high byte of address).	
4.2		Store as Operand.	
5.1	Write Address	Write unmodified Operand.	
5.2		* Perform Operand=(Operand << 1) C, A=A & Operand, set N and Z based off A. Set C if high bit of Operand was set before the shift operation.	
6.1			
6.2	Write Address	Write modified Operand.	
7.1	Read PC	Store as OpCode.	
7.2			
+X.1		A applied.	

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
5	0003	3f	1	unknown	0003	aa	33	00	fd	nv-BdIzC	SEC	T1
5	0003	3f	1	unknown	0003	aa	33	00	fd	nv-BdIzC	SEC	T1
6	0004	a9	1		0004	aa	33	00	fd	nv-BdIzC	unknown	T2
6	0004	a9	1		0004	aa	33	00	fd	nv-BdIzC	unknown	T2
7	0005	00	1		0005	aa	33	00	fd	nv-BdIzC	unknown	T3
7	0005	00	1		0005	aa	33	00	fd	nv-BdIzC	unknown	T3
8	00dc	e1	1		0006	aa	33	00	fd	nv-BdIzC	unknown	T4
8	00dc	e1	1		0006	aa	33	00	fd	nv-BdIzC	unknown	T4
9	00dc	e1	1		0006	aa	33	00	fd	nv-BdIzC	unknown	T5
9	00dc	e1	1		0006	aa	33	00	fd	nv-BdIzC	unknown	T5
10	00dc	e1	0		0006	aa	33	00	fd	nv-BdIzC	unknown	
10	00dc	e1	0		0006	aa	33	00	fd	nv-BdIzC	unknown	
11	00dc	e1	0		0006	aa	33	00	fd	Nv-BdIzC	unknown	T0
11	00dc	c3	0		0006	aa	33	00	fd	Nv-BdIzC	unknown	T0
12	0006	00	1	BRK	0006	aa	33	00	fd	Nv-BdIzC	unknown	T1
12	0006	00	1	BRK	0006	aa	33	00	fd	Nv-BdIzC	unknown	T1
13	0007	00	1		0007	82	33	00	fd	Nv-BdIzC	BRK	T2
13	0007	00	1		0007	82	33	00	fd	Nv-BdIzC	BRK	T2

RTI (40)	Cycles: 6	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read S \$0100	
3.1		
3.2	Read (S +1) \$0100	Store as Tmp.
4.1		Copy all but the M and B flags from Tmp to P (M and B remain unchanged on P).
4.2	Read (S +2) \$0100	Store as Address.L.
5.1		Increase S by 3.
5.2	Read S \$0100	Store as Address.H.
6.1		PC=Address.
6.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	40	1	RTI	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
2	0002	40	1	RTI	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	RTI	T2
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	RTI	T2
4	01fd	ff	1		0004	00	00	00	fd	nv-BdIZc	RTI	T3
4	01fd	ff	1		0004	00	00	00	fd	nv-BdIZc	RTI	T3
5	01fe	88	1		0004	00	00	00	fd	nv-BdIZc	RTI	T4
5	01fe	88	1		0004	00	00	00	fd	nv-BdIZc	RTI	T4
6	01ff	12	1		0004	00	00	00	fd	Nv-BDizc	RTI	T5
6	01ff	12	1		0004	00	00	00	fd	Nv-BDizc	RTI	T5
7	0100	00	1		0004	00	00	00	00	Nv-BDizc	RTI	T0
7	0100	00	1		0004	00	00	00	00	Nv-BDizc	RTI	T0
8	0012	e6	1	INC zp	0012	00	00	00	00	Nv-BDizc	RTI	T1
8	0012	e6	1	INC zp	0012	00	00	00	00	Nv-BDizc	RTI	T1
9	0013	0f	1		0013	00	00	00	00	Nv-BDizc	INC zp	T2
9	0013	0f	1		0013	00	00	00	00	Nv-BDizc	INC zp	T2

EOR (41)	Cycles: 6	Size: 2
Indirect, X (Read)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Operand.
2.1		Inc. PC.
2.2	Read Operand	
3.1		Set Pointer to (Operand+X) & \$FF.
3.2	Read Pointer	Store to Address.L.
4.1		
4.2	Read (Pointer+1) & \$FF	Store to Address.H.
5.1		
5.2	Read Address	Store as Operand.
6.1		* Perform $A=A \wedge \text{Operand}$, set N and Z accordingly.
6.2	Read PC	Store as OpCode.
+X.1		A and N, Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	41	1	EOR (zp,X)	0003	1f	00	00	fd	nv-BdIzC	SEC	T1
4	0003	41	1	EOR (zp,X)	0003	1f	00	00	fd	nv-BdIzC	SEC	T1
5	0004	a9	1		0004	1f	00	00	fd	nv-BdIzC	EOR (zp,X)	T2
5	0004	a9	1		0004	1f	00	00	fd	nv-BdIzC	EOR (zp,X)	T2
6	00a9	55	1		0005	1f	00	00	fd	nv-BdIzC	EOR (zp,X)	T3
6	00a9	55	1		0005	1f	00	00	fd	nv-BdIzC	EOR (zp,X)	T3
7	00a9	55	1		0005	1f	00	00	fd	nv-BdIzC	EOR (zp,X)	T4
7	00a9	55	1		0005	1f	00	00	fd	nv-BdIzC	EOR (zp,X)	T4
8	00aa	00	1		0005	1f	00	00	fd	nv-BdIzC	EOR (zp,X)	T5
8	00aa	00	1		0005	1f	00	00	fd	nv-BdIzC	EOR (zp,X)	T5
9	0055	f8	1		0005	1f	00	00	fd	nv-BdIzC	EOR (zp,X)	T0
9	0055	f8	1		0005	1f	00	00	fd	nv-BdIzC	EOR (zp,X)	T0
10	0005	00	1	BRK	0005	1f	00	00	fd	nv-BdIzC	EOR (zp,X)	T1
10	0005	00	1	BRK	0005	1f	00	00	fd	nv-BdIzC	EOR (zp,X)	T1
11	0006	00	1		0006	e7	00	00	fd	Nv-BdIzC	BRK	T2
11	0006	00	1		0006	e7	00	00	fd	Nv-BdIzC	BRK	T2

JAM (42)	Cycles: ∞	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	
2.1		
2.2	Read \$FFFF	
3.1		
3.2	Read \$FFFE	
4.1		
4.2	Read \$FFFE	
5.1		Repeat 5.1 and 5.2 forever.
5.2	Read \$FFFF	

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	

SRE (43)	Cycles: 8	Size: 2
Indirect, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Operand.
2.1		Inc. PC.
2.2	Read Operand	
3.1		Set Pointer to (Operand+X) & \$FF.
3.2	Read Pointer	Store to Address.L.
4.1		
4.2	Read (Pointer+1) & \$FF	Store to Address.H.
5.1		
5.1	Read Address	Store as Operand.
6.1		Set C if lowest bit of Operand is set.
6.2	Write Address	Write unmodified Operand.
7.1		* Perform Operand=Operand >> 1, A = A ^ Operand, set N and Z based off A .
7.2	Write Address	Write modified Operand.
8.1		
8.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	43	1	unknown	0003	1f	00	00	fd	nv-BdIzC	SEC	T1
4	0003	43	1	unknown	0003	1f	00	00	fd	nv-BdIzC	SEC	T1
5	0004	a9	1		0004	1f	00	00	fd	nv-BdIzC	unknown	T2
5	0004	a9	1		0004	1f	00	00	fd	nv-BdIzC	unknown	T2
6	00a9	55	1		0005	1f	00	00	fd	nv-BdIzC	unknown	T3
6	00a9	55	1		0005	1f	00	00	fd	nv-BdIzC	unknown	T3
7	00a9	55	1		0005	1f	00	00	fd	nv-BdIzC	unknown	T4
7	00a9	55	1		0005	1f	00	00	fd	nv-BdIzC	unknown	T4
8	00aa	00	1		0005	1f	00	00	fd	nv-BdIzC	unknown	T5
8	00aa	00	1		0005	1f	00	00	fd	nv-BdIzC	unknown	T5
9	0055	f8	1		0005	1f	00	00	fd	nv-BdIzC	unknown	
9	0055	f8	1		0005	1f	00	00	fd	nv-BdIzC	unknown	
10	0055	f8	0		0005	1f	00	00	fd	nv-BdIzC	unknown	
10	0055	f8	0		0005	1f	00	00	fd	nv-BdIzC	unknown	
11	0055	f8	0		0005	1f	00	00	fd	nv-BdIzC	unknown	T0
11	0055	7c	0		0005	1f	00	00	fd	nv-BdIzC	unknown	T0
12	0005	00	1	BRK	0005	1f	00	00	fd	nv-BdIzC	unknown	T1
12	0005	00	1	BRK	0005	1f	00	00	fd	nv-BdIzC	unknown	T1
13	0006	00	1		0006	63	00	00	fd	nv-BdIzC	BRK	T2
13	0006	00	1		0006	63	00	00	fd	nv-BdIzC	BRK	T2

NOP (44)		Cycles: 3	Size: 2
Zero Page (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		Store as OpCode.	
3.2	Read PC		
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	44	1	unknown	0003	1f	00	00	fd	nv-BdIzC	SEC	T1
4	0003	44	1	unknown	0003	1f	00	00	fd	nv-BdIzC	SEC	T1
5	0004	a9	1		0004	1f	00	00	fd	nv-BdIzC	unknown	T2
5	0004	a9	1		0004	1f	00	00	fd	nv-BdIzC	unknown	T2
6	00a9	55	1		0005	1f	00	00	fd	nv-BdIzC	unknown	T0
6	00a9	55	1		0005	1f	00	00	fd	nv-BdIzC	unknown	T0
7	0005	00	1	BRK	0005	1f	00	00	fd	nv-BdIzC	unknown	T1
7	0005	00	1	BRK	0005	1f	00	00	fd	nv-BdIzC	unknown	T1
8	0006	00	1		0006	1f	00	00	fd	nv-BdIzC	BRK	T2
8	0006	00	1		0006	1f	00	00	fd	nv-BdIzC	BRK	T2

EOR (45)		Cycles: 3	Size: 2
Zero Page (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		* Perform A=A ^ Operand , set N and Z accordingly.	
3.2	Read PC	Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	45	1	EOR zp	0003	1f	00	00	fd	nv-BdIzC	SEC	T1
4	0003	45	1	EOR zp	0003	1f	00	00	fd	nv-BdIzC	SEC	T1
5	0004	9f	1		0004	1f	00	00	fd	nv-BdIzC	EOR zp	T2
5	0004	9f	1		0004	1f	00	00	fd	nv-BdIzC	EOR zp	T2
6	009f	f8	1		0005	1f	00	00	fd	nv-BdIzC	EOR zp	T0
6	009f	f8	1		0005	1f	00	00	fd	nv-BdIzC	EOR zp	T0
7	0005	f8	1	SED	0005	1f	00	00	fd	nv-BdIzC	EOR zp	T1
7	0005	f8	1	SED	0005	1f	00	00	fd	nv-BdIzC	EOR zp	T1
8	0006	00	1		0006	e7	00	00	fd	Nv-BdIzC	SED	T0+T2
8	0006	00	1		0006	e7	00	00	fd	Nv-BdIzC	SED	T0+T2

LSR (46)	Cycles: 5	Size: 2
Zero Page (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.
2.1		Inc. PC.
2.2	Read Address	Store as Operand.
3.1		Set C if lowest bit of Operand is set.
3.2	Write Address	Write unmodified Operand.
4.1		Perform Operand=Operand << 1, set N and Z accordingly.
4.2	Write Address	
5.1		
5.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	46	1	LSR zp	0003	1f	00	00	fd	nv-BdIzC	SEC	T1
4	0003	46	1	LSR zp	0003	1f	00	00	fd	nv-BdIzC	SEC	T1
5	0004	9f	1		0004	1f	00	00	fd	nv-BdIzC	LSR zp	T2
5	0004	9f	1		0004	1f	00	00	fd	nv-BdIzC	LSR zp	T2
6	009f	f8	1		0005	1f	00	00	fd	nv-BdIzC	LSR zp	T3
6	009f	f8	1		0005	1f	00	00	fd	nv-BdIzC	LSR zp	T3
7	009f	f8	0		0005	1f	00	00	fd	nv-BdIzc	LSR zp	T4
7	009f	f8	0		0005	1f	00	00	fd	nv-BdIzc	LSR zp	T4
8	009f	f8	0		0005	1f	00	00	fd	nv-BdIzc	LSR zp	T0
8	009f	7c	0		0005	1f	00	00	fd	nv-BdIzc	LSR zp	T0
9	0005	a0	1	LDY #	0005	1f	00	00	fd	nv-BdIzc	LSR zp	T1
9	0005	a0	1	LDY #	0005	1f	00	00	fd	nv-BdIzc	LSR zp	T1
10	0006	00	1		0006	1f	00	00	fd	nv-BdIzc	LDY #	T0+T2
10	0006	00	1		0006	1f	00	00	fd	nv-BdIzc	LDY #	T0+T2

SRE (47)	Cycles: 5	Size: 2
Zero Page (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.
2.1		Inc. PC.
2.2	Read Address	Store as Operand.
3.1		Set C if lowest bit of Operand is set.
3.2	Write Address	Write unmodified Operand.
4.1		* Perform $\text{Operand} = \text{Operand} \gg 1$, $\text{A} = \text{A} \wedge \text{Operand}$, set N and Z based off A .
4.2	Write Address	Write modified Operand.
5.1		
5.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0003	47	1	unknown	0003	1f	00	00	fd	nv-BdIzC	SEC	T1
4	0003	47	1	unknown	0003	1f	00	00	fd	nv-BdIzC	SEC	T1
5	0004	9f	1		0004	1f	00	00	fd	nv-BdIzC	unknown	T2
5	0004	9f	1		0004	1f	00	00	fd	nv-BdIzC	unknown	T2
6	009f	7e	1		0005	1f	00	00	fd	nv-BdIzC	unknown	T3
6	009f	7e	1		0005	1f	00	00	fd	nv-BdIzC	unknown	T3
7	009f	7e	0		0005	1f	00	00	fd	nv-BdIzc	unknown	T4
7	009f	7e	0		0005	1f	00	00	fd	nv-BdIzc	unknown	T4
8	009f	7e	0		0005	1f	00	00	fd	nv-BdIzc	unknown	T0
8	009f	3f	0		0005	1f	00	00	fd	nv-BdIzc	unknown	T0
9	0005	a0	1	LDY #	0005	1f	00	00	fd	nv-BdIzc	unknown	T1
9	0005	a0	1	LDY #	0005	1f	00	00	fd	nv-BdIzc	unknown	T1
10	0006	00	1		0006	20	00	00	fd	nv-BdIzc	LDY #	T0+T2
10	0006	00	1		0006	20	00	00	fd	nv-BdIzc	LDY #	T0+T2

PHA (48)		Cycles: 3	Size: 1
Implied			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		To be discarded.	
2.1		Write A .	
2.2			
3.1		Store as OpCode.	
3.2			
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	48	1	PHA	0002	fe	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	48	1	PHA	0002	fe	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	10	1		0003	fe	00	00	fd	Nv-BdIzc	PHA	T2
3	0003	10	1		0003	fe	00	00	fd	Nv-BdIzc	PHA	T2
4	01fd	10	0		0003	fe	00	00	fd	Nv-BdIzc	PHA	T0
4	01fd	fe	0		0003	fe	00	00	fd	Nv-BdIzc	PHA	T0
5	0003	10	1	BPL	0003	fe	00	00	fc	Nv-BdIzc	PHA	T1
5	0003	10	1	BPL	0003	fe	00	00	fc	Nv-BdIzc	PHA	T1
6	0004	00	1		0004	fe	00	00	fc	Nv-BdIzc	BPL	T2
6	0004	00	1		0004	fe	00	00	fc	Nv-BdIzc	BPL	T2

EOR (49)		Cycles: 2	Size: 2
Immediate			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Operand.	
2.1		Inc.PC. * Perform $A=A \wedge$ Operand, set N and Z accordingly.	
2.2		Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	49	1	EOR #	0002	80	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	49	1	EOR #	0002	80	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	80	1		0003	80	00	00	fd	Nv-BdIzc	EOR #	T0+T2
3	0003	80	1		0003	80	00	00	fd	Nv-BdIzc	EOR #	T0+T2
4	0004	35	1	AND zp,X	0004	80	00	00	fd	Nv-BdIzc	EOR #	T1
4	0004	35	1	AND zp,X	0004	80	00	00	fd	Nv-BdIzc	EOR #	T1
5	0005	4c	1		0005	00	00	00	fd	nv-BdIZc	AND zp,X	T2
5	0005	4c	1		0005	00	00	00	fd	nv-BdIZc	AND zp,X	T2

LSR (4A)	Cycles: 2	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1	Read PC	Inc. PC.
1.2		To be discarded.
2.1		Set C if low bit of A is set. Perform A=A >> 1 , unset N and set Z accordingly.
2.2		Store as OpCode.
+X.1		A and N, Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	4a	1	LSR	0002	80	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	4a	1	LSR	0002	80	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	35	1		0003	80	00	00	fd	Nv-BdIzc	LSR	T0+T2
3	0003	35	1		0003	80	00	00	fd	Nv-BdIzc	LSR	T0+T2
4	0003	35	1	AND zp,X	0003	80	00	00	fd	Nv-BdIzc	LSR	T1
4	0003	35	1	AND zp,X	0003	80	00	00	fd	Nv-BdIzc	LSR	T1
5	0004	00	1		0004	40	00	00	fd	nv-BdIzc	AND zp,X	T2
5	0004	00	1		0004	40	00	00	fd	nv-BdIzc	AND zp,X	T2

ASR (4B)	Cycles: 2	Size: 2
Immediate		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Operand.
2.1		Inc.PC. * Perform A=A & Operand , A=A >> 1 , set C , N and Z accordingly. C is set if (A & 1) before the shift.
2.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	4b	1	unknown	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	4b	1	unknown	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	0f	1		0003	ff	00	00	fd	Nv-BdIzc	unknown	T0+T2
3	0003	0f	1		0003	ff	00	00	fd	Nv-BdIzc	unknown	T0+T2
4	0004	30	1	BMI	0004	ff	00	00	fd	Nv-BdIzc	unknown	T1
4	0004	30	1	BMI	0004	ff	00	00	fd	Nv-BdIzc	unknown	T1
5	0005	4c	1		0005	7f	00	00	fd	nv-BdIzc	BMI	T2
5	0005	4c	1		0005	7f	00	00	fd	nv-BdIzc	BMI	T2

JMP (4C)		Cycles: 3	Size: 3
Absolute			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.L.	
2.1		Inc. PC.	
2.2		Store as Address.H.	
3.1		Copy Address to PC.	
3.2		Store as OpCode.	
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	4c	1	JMP Abs	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	4c	1	JMP Abs	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	20	1		0003	ff	00	00	fd	Nv-BdIzc	JMP Abs	T2
3	0003	20	1		0003	ff	00	00	fd	Nv-BdIzc	JMP Abs	T2
4	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	JMP Abs	T0
4	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	JMP Abs	T0
5	0120	35	1	AND zp,X	0120	ff	00	00	fd	Nv-BdIzc	JMP Abs	T1
5	0120	35	1	AND zp,X	0120	ff	00	00	fd	Nv-BdIzc	JMP Abs	T1
6	0121	00	1		0121	ff	00	00	fd	Nv-BdIzc	AND zp,X	T2
6	0121	00	1		0121	ff	00	00	fd	Nv-BdIzc	AND zp,X	T2

EOR (4D)		Cycles: 4	Size: 3
Absolute (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.L.	
2.1		Inc. PC.	
2.2		Store as Address.H.	
3.1		Inc. PC.	
3.2	Read Address	Store as Operand.	
4.1	Read PC	* Perform $A=A \wedge \text{Operand}$, set N and Z accordingly.	
4.2		Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	4d	1	EOR Abs	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	4d	1	EOR Abs	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	20	1		0003	ff	00	00	fd	Nv-BdIzc	EOR Abs	T2
3	0003	20	1		0003	ff	00	00	fd	Nv-BdIzc	EOR Abs	T2
4	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	EOR Abs	T3
4	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	EOR Abs	T3
5	0120	35	1		0005	ff	00	00	fd	Nv-BdIzc	EOR Abs	T0
5	0120	35	1		0005	ff	00	00	fd	Nv-BdIzc	EOR Abs	T0
6	0005	20	1	JSR Abs	0005	ff	00	00	fd	Nv-BdIzc	EOR Abs	T1
6	0005	20	1	JSR Abs	0005	ff	00	00	fd	Nv-BdIzc	EOR Abs	T1
7	0006	02	1		0006	ca	00	00	fd	Nv-BdIzc	JSR Abs	T2
7	0006	02	1		0006	ca	00	00	fd	Nv-BdIzc	JSR Abs	T2

LSR (4E)	Cycles: 6	Size: 3
Absolute (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC.
3.2	Read Address	Store as Operand.
4.1		Set C if lowest bit of Operand is set.
4.2	Write Address	Write unmodified Operand.
5.1		Perform Operand=Operand << 1, set N and Z accordingly.
5.2	Write Address	Write modified Operand.
6.1		
6.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
3	0002	4e	1	LSR Abs	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
3	0002	4e	1	LSR Abs	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	LSR Abs	T2
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	LSR Abs	T2
5	0004	01	1		0004	aa	00	00	fd	NV-BdIzc	LSR Abs	T3
5	0004	01	1		0004	aa	00	00	fd	NV-BdIzc	LSR Abs	T3
6	01fb	3f	1		0005	aa	00	00	fd	NV-BdIzc	LSR Abs	T4
6	01fb	3f	1		0005	aa	00	00	fd	NV-BdIzc	LSR Abs	T4
7	01fb	3f	0		0005	aa	00	00	fd	NV-BdIzc	LSR Abs	T5
7	01fb	3f	0		0005	aa	00	00	fd	NV-BdIzc	LSR Abs	T5
8	01fb	3f	0		0005	aa	00	00	fd	nV-BdIzc	LSR Abs	T0
8	01fb	1f	0		0005	aa	00	00	fd	nV-BdIzc	LSR Abs	T0
9	0005	50	1	BVC	0005	aa	00	00	fd	nV-BdIzc	LSR Abs	T1
9	0005	50	1	BVC	0005	aa	00	00	fd	nV-BdIzc	LSR Abs	T1
10	0006	2e	1		0006	aa	00	00	fd	nV-BdIzc	BVC	T2
10	0006	2e	1		0006	aa	00	00	fd	nV-BdIzc	BVC	T2

SRE (4F)	Cycles: 6	Size: 3
Absolute (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC.
3.2	Read Address	Store as Operand.
4.1		Set C if lowest bit of Operand is set.
4.2	Write Address	Write unmodified Operand.
5.1		* Perform Operand=Operand >> 1, A = A ^ Operand, set N and Z based off A .
5.2	Write Address	Write modified Operand.
6.1		
6.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	4f	1	unknown	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	4f	1	unknown	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	20	1		0003	ff	00	00	fd	Nv-BdIzc	unknown	T2
3	0003	20	1		0003	ff	00	00	fd	Nv-BdIzc	unknown	T2
4	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	unknown	T3
4	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	unknown	T3
5	0120	ff	1		0005	ff	00	00	fd	Nv-BdIzc	unknown	T4
5	0120	ff	1		0005	ff	00	00	fd	Nv-BdIzc	unknown	T4
6	0120	ff	0		0005	ff	00	00	fd	Nv-BdIzC	unknown	T5
6	0120	ff	0		0005	ff	00	00	fd	Nv-BdIzC	unknown	T5
7	0120	ff	0		0005	ff	00	00	fd	nv-BdIzC	unknown	T0
7	0120	7f	0		0005	ff	00	00	fd	nv-BdIzC	unknown	T0
8	0005	20	1	JSR Abs	0005	ff	00	00	fd	nv-BdIzC	unknown	T1
8	0005	20	1	JSR Abs	0005	ff	00	00	fd	nv-BdIzC	unknown	T1
9	0006	02	1		0006	80	00	00	fd	Nv-BdIzC	JSR Abs	T2
9	0006	02	1		0006	80	00	00	fd	Nv-BdIzC	JSR Abs	T2

BVC (50)	Cycles: 2-4	Size: 2
Branch Relative		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1	Read PC	Inc. PC. Check condition (V == 0).
1.2		Store as Operand. Treat as signed 16-bit (Op16=i16(i8(Operand))).
2.1		Inc. PC. If not jumping, end (next half-cycle is 4.2)
2.2		If (PC+Op16).H != PC.H, end after PC.L fix (next half-cycle is 4.2). PC.L=PC.L+Operand.
3.1		
3.2		
4.1		PC.H=previous “(PC+Op16).H” value.
4.2		Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
3	0002	50	1	BVC	0002	aa	00	00	fd	nV-BdIzc	BIT zp	T1
3	0002	50	1	BVC	0002	aa	00	00	fd	nV-BdIzc	BIT zp	T1
4	0003	20	1		0003	aa	00	00	fd	nV-BdIzc	BVC	T2
4	0003	20	1		0003	aa	00	00	fd	nV-BdIzc	BVC	T2
5	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	nV-BdIzc	BVC	
5	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	nV-BdIzc	BVC	
6	0005	50	1		0005	aa	00	00	fd	nV-BdIzc	ORA (zp,X)	T2
6	0005	50	1		0005	aa	00	00	fd	nV-BdIzc	ORA (zp,X)	T2

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	50	1	BVC	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	50	1	BVC	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	20	1		0003	ff	00	00	fd	Nv-BdIzc	BVC	T2
3	0003	20	1		0003	ff	00	00	fd	Nv-BdIzc	BVC	T2
4	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	BVC	T3
4	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	BVC	T3
5	0024	16	1	ASL zp,X	0024	ff	00	00	fd	Nv-BdIzc	BVC	
5	0024	16	1	ASL zp,X	0024	ff	00	00	fd	Nv-BdIzc	BVC	
6	0025	32	1		0025	ff	00	00	fd	Nv-BdIzc	ASL zp,X	T2
6	0025	32	1		0025	ff	00	00	fd	Nv-BdIzc	ASL zp,X	T2

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
6	01fb	50	1	BVC	01fb	aa	00	00	fd	nv-BdIzc	JMP Abs	T1
6	01fb	50	1	BVC	01fb	aa	00	00	fd	nv-BdIzc	JMP Abs	T1
7	01fc	7f	1		01fc	aa	00	00	fd	nv-BdIzc	BVC	T2
7	01fc	7f	1		01fc	aa	00	00	fd	nv-BdIzc	BVC	T2
8	01fd	00	1		01fd	aa	00	00	fd	nv-BdIzc	BVC	T3
8	01fd	00	1		01fd	aa	00	00	fd	nv-BdIzc	BVC	T3
9	017c	00	1		017c	aa	00	00	fd	nv-BdIzc	BVC	T0
9	017c	00	1		017c	aa	00	00	fd	nv-BdIzc	BVC	T0
10	027c	00	1	BRK	027c	aa	00	00	fd	nv-BdIzc	BVC	T1
10	027c	00	1	BRK	027c	aa	00	00	fd	nv-BdIzc	BVC	T1
11	027d	00	1		027d	aa	00	00	fd	nv-BdIzc	BRK	T2
11	027d	00	1		027d	aa	00	00	fd	nv-BdIzc	BRK	T2

EOR (51)	Cycles: 5-6	Size: 2
Indirect, Y (Read)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Pointer.
2.1		Inc. PC.
2.2	Read Pointer	Store to Address.L.
3.1		
3.2	Read (Pointer+1) & \$FF	Store to Address.H.
4.1		Final=Address+Y. Address.L = Final.L.
4.2	Read Address	Store as Operand. If Address.H == Final.H, skip the next cycle (next half-cycle is 6.1).
5.1		Address.H = Final.H (fixes high byte of address).
5.2	Read Address	Store as Operand.
6.1		* Perform $A=A \wedge \text{Operand}$, set N and Z accordingly.
6.2	Read PC	Store as OpCode.
+X.1		A and N, Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
3	0002	51	1	EOR (zp),Y	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
3	0002	51	1	EOR (zp),Y	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	EOR (zp),Y	T2
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	EOR (zp),Y	T2
5	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	EOR (zp),Y	T3
5	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	EOR (zp),Y	T3
6	00fc	00	1		0004	aa	00	00	fd	NV-BdIzc	EOR (zp),Y	T4
6	00fc	00	1		0004	aa	00	00	fd	NV-BdIzc	EOR (zp),Y	T4
7	0080	7f	1		0004	aa	00	00	fd	NV-BdIzc	EOR (zp),Y	T0
7	0080	7f	1		0004	aa	00	00	fd	NV-BdIzc	EOR (zp),Y	T0
8	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	NV-BdIzc	EOR (zp),Y	T1
8	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	NV-BdIzc	EOR (zp),Y	T1
9	0005	50	1		0005	d5	00	00	fd	NV-BdIzc	ORA (zp,X)	T2
9	0005	50	1		0005	d5	00	00	fd	NV-BdIzc	ORA (zp,X)	T2

JAM (52)	Cycles: ∞	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	
2.1		
2.2	Read \$FFFF	
3.1		
3.2	Read \$FFFE	
4.1		
4.2	Read \$FFFE	
5.1		Repeat 5.1 and 5.2 forever.
5.2	Read \$FFFF	

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	

SRE (53)	Cycles: 8	Size: 2
Indirect, Y (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Pointer.
2.1		Inc. PC.
2.2	Read Pointer	Store to Address.L.
3.1		
3.2	Read (Pointer+1) & \$FF	Store to Address.H.
4.1		Final=Address+Y. Address.L = Final.L.
4.2	Read Address	Store as Operand.
5.1		Address.H = Final.H (fixes high byte of address).
5.2	Read Address	Store as Operand.
6.1		Set C if lowest bit of Operand is set.
6.2	Write Address	Write unmodified Operand.
7.1		* Perform Operand=Operand >> 1, A = A ^ Operand, set N and Z based off A .
7.2	Write Address	Write modified Operand.
8.1		
8.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
3	0002	53	1	unknown	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
3	0002	53	1	unknown	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	unknown	T2
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	unknown	T2
5	00fb	27	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T3
5	00fb	27	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T3
6	00fc	00	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T4
6	00fc	00	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T4
7	0027	00	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T5
7	0027	00	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T5
8	0027	00	1		0004	aa	00	00	fd	NV-BdIzc	unknown	
8	0027	00	1		0004	aa	00	00	fd	NV-BdIzc	unknown	
9	0027	00	0		0004	aa	00	00	fd	NV-BdIzc	unknown	
9	0027	00	0		0004	aa	00	00	fd	NV-BdIzc	unknown	
10	0027	00	0		0004	aa	00	00	fd	nV-BdIzc	unknown	T0
10	0027	00	0		0004	aa	00	00	fd	nV-BdIzc	unknown	T0
11	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	nV-BdIzc	unknown	T1
11	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	nV-BdIzc	unknown	T1
12	0005	50	1		0005	aa	00	00	fd	NV-BdIzc	ORA (zp,X)	T2
12	0005	50	1		0005	aa	00	00	fd	NV-BdIzc	ORA (zp,X)	T2

NOP (54)		Cycles: 4	Size: 2
Zero Page, X (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Pointer.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		Set Address to (Pointer+X) & \$FF.	
3.2		Store as Operand.	
4.1		Store as OpCode.	
4.2			
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
3	0002	54	1	unknown	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
3	0002	54	1	unknown	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	unknown	T2
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	unknown	T2
5	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T3
5	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T3
6	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T0
6	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	unknown	T0
7	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	NV-BdIzc	unknown	T1
7	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	NV-BdIzc	unknown	T1
8	0005	50	1		0005	aa	00	00	fd	NV-BdIzc	ORA (zp,X)	T2
8	0005	50	1		0005	aa	00	00	fd	NV-BdIzc	ORA (zp,X)	T2

EOR (55)		Cycles: 4	Size: 2
Zero Page, X (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Pointer.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		Set Address to (Pointer+X) & \$FF.	
3.2	Read Address	Store as Operand.	
4.1	Read PC	* Perform A=A ^ Operand , set N and Z accordingly.	
4.2		Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
3	0002	55	1	EOR zp,X	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
3	0002	55	1	EOR zp,X	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	EOR zp,X	T2
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	EOR zp,X	T2
5	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	EOR zp,X	T3
5	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	EOR zp,X	T3
6	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	EOR zp,X	T0
6	00fb	80	1		0004	aa	00	00	fd	NV-BdIzc	EOR zp,X	T0
7	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	NV-BdIzc	EOR zp,X	T1
7	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	NV-BdIzc	EOR zp,X	T1
8	0005	50	1		0005	2a	00	00	fd	nV-BdIzc	ORA (zp,X)	T2
8	0005	50	1		0005	2a	00	00	fd	nV-BdIzc	ORA (zp,X)	T2

LSR (56)	Cycles: 6	Size: 2
Zero Page, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Pointer.
2.1		Inc. PC.
2.2	Read Pointer	Store as Operand.
3.1		Set Address to (Pointer+X) & \$FF.
3.2	Read Address	Store as Operand.
4.1		Set C if lowest bit of Operand is set.
4.2	Write Address	Write unmodified Operand.
5.1		Perform Operand=Operand << 1, set N and Z accordingly.
5.2	Write Address	Write modified Operand.
6.1		
6.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
3	0002	56	1	LSR zp,X	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
3	0002	56	1	LSR zp,X	0002	aa	00	00	fd	NV-BdIzc	BIT zp	T1
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	LSR zp,X	T2
4	0003	fb	1		0003	aa	00	00	fd	NV-BdIzc	LSR zp,X	T2
5	00fb	4f	1		0004	aa	00	00	fd	NV-BdIzc	LSR zp,X	T3
5	00fb	4f	1		0004	aa	00	00	fd	NV-BdIzc	LSR zp,X	T3
6	00fb	4f	1		0004	aa	00	00	fd	NV-BdIzc	LSR zp,X	T4
6	00fb	4f	1		0004	aa	00	00	fd	NV-BdIzc	LSR zp,X	T4
7	00fb	4f	0		0004	aa	00	00	fd	NV-BdIzC	LSR zp,X	T5
7	00fb	4f	0		0004	aa	00	00	fd	NV-BdIzC	LSR zp,X	T5
8	00fb	4f	0		0004	aa	00	00	fd	nV-BdIzC	LSR zp,X	T0
8	00fb	27	0		0004	aa	00	00	fd	nV-BdIzC	LSR zp,X	T0
9	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	nV-BdIzC	LSR zp,X	T1
9	0004	01	1	ORA (zp,X)	0004	aa	00	00	fd	nV-BdIzC	LSR zp,X	T1
10	0005	50	1		0005	aa	00	00	fd	nV-BdIzC	ORA (zp,X)	T2
10	0005	50	1		0005	aa	00	00	fd	nV-BdIzC	ORA (zp,X)	T2

SRE (57)	Cycles: 6	Size: 2
Zero Page, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Pointer.
2.1		Inc. PC.
2.2	Read Pointer	Store as Operand.
3.1		Set Address to (Pointer+X) & \$FF.
3.2	Read Address	Store as Operand.
4.1		Set C if lowest bit of Operand is set.
4.2	Write Address	Write unmodified Operand.
5.1		* Perform Operand=Operand >> 1, A = A ^ Operand, set N and Z based off A .
5.2	Write Address	Write modified Operand.
6.1		
6.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	57	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
2	0002	57	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
3	0003	ff	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
3	0003	ff	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
4	00ff	7f	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
4	00ff	7f	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
5	00ff	7f	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
5	00ff	7f	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
6	00ff	7f	0		0004	00	00	00	fd	nv-BdIZC	unknown	T5
6	00ff	7f	0		0004	00	00	00	fd	nv-BdIZC	unknown	T5
7	00ff	7f	0		0004	00	00	00	fd	nv-BdIzC	unknown	T0
7	00ff	3f	0		0004	00	00	00	fd	nv-BdIzC	unknown	T0
8	0004	20	1	JSR Abs	0004	00	00	00	fd	nv-BdIzC	unknown	T1
8	0004	20	1	JSR Abs	0004	00	00	00	fd	nv-BdIzC	unknown	T1
9	0005	4c	1		0005	3f	00	00	fd	nv-BdIzC	JSR Abs	T2
9	0005	4c	1		0005	3f	00	00	fd	nv-BdIzC	JSR Abs	T2

CLI (58)	Cycles: 2	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	To be discarded.
2.1		Clear the I status bit.
2.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	58	1	CLI	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
2	0002	58	1	CLI	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
3	0003	21	1		0003	00	00	00	fd	nv-BdIZc	CLI	T0+T2
3	0003	21	1		0003	00	00	00	fd	nv-BdIZc	CLI	T0+T2
4	0003	21	1	AND (zp,X)	0003	00	00	00	fd	nv-BdiZc	CLI	T1
4	0003	21	1	AND (zp,X)	0003	00	00	00	fd	nv-BdiZc	CLI	T1
5	0004	20	1		0004	00	00	00	fd	nv-BdiZc	AND (zp,X)	T2
5	0004	20	1		0004	00	00	00	fd	nv-BdiZc	AND (zp,X)	T2

EOR (59)		Cycles: 4-5	Size: 3
Absolute, Y (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.L.	
2.1		Inc. PC.	
2.2		Store as Address.H.	
3.1		Inc. PC. Final=Address+Y. Address.L = Final.L.	
3.2	Read Address	Store as Operand. If Address.H == Final.H, skip the next cycle (next half-cycle is 5.1).	
4.1	Read Address	Address.H = Final.H (fixes high byte of address).	
4.2		Store as Operand.	
5.1		* Perform A=A ^ Operand , set N and Z accordingly.	
5.2	Read PC	Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	59	1	EOR Abs,Y	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	59	1	EOR Abs,Y	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	21	1		0003	ff	00	00	fd	Nv-BdIzc	EOR Abs,Y	T2
3	0003	21	1		0003	ff	00	00	fd	Nv-BdIzc	EOR Abs,Y	T2
4	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	EOR Abs,Y	T3
4	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	EOR Abs,Y	T3
5	0121	7f	1		0005	ff	00	00	fd	Nv-BdIzc	EOR Abs,Y	T0
5	0121	7f	1		0005	ff	00	00	fd	Nv-BdIzc	EOR Abs,Y	T0
6	0005	4c	1	JMP Abs	0005	ff	00	00	fd	Nv-BdIzc	EOR Abs,Y	T1
6	0005	4c	1	JMP Abs	0005	ff	00	00	fd	Nv-BdIzc	EOR Abs,Y	T1
7	0006	02	1		0006	80	00	00	fd	Nv-BdIzc	JMP Abs	T2
7	0006	02	1		0006	80	00	00	fd	Nv-BdIzc	JMP Abs	T2

NOP (5A)		Cycles: 2	Size: 1
Implied			
Cycle	R/W		Desc
-X.2	Read PC		Store as OpCode.
1.1			Inc. PC.
1.2			To be discarded.
2.1			
2.2			Store as OpCode.
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	5a	1	unknown	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	5a	1	unknown	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	21	1		0003	ff	00	00	fd	Nv-BdIzc	unknown	T0+T2
3	0003	21	1		0003	ff	00	00	fd	Nv-BdIzc	unknown	T0+T2
4	0003	21	1	AND (zp,X)	0003	ff	00	00	fd	Nv-BdIzc	unknown	T1
4	0003	21	1	AND (zp,X)	0003	ff	00	00	fd	Nv-BdIzc	unknown	T1
5	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	AND (zp,X)	T2
5	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	AND (zp,X)	T2

SRE (5B)	Cycles: 7	Size: 3
Absolute, Y (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC. Final=Address+Y. Address.L = Final.L.
3.2	Read Address	Store as Operand.
4.1		Address.H = Final.H (fixes high byte of address).
4.2	Read Address	Store as Operand.
5.1		Set C if lowest bit of Operand is set.
5.2	Write Address	Write unmodified Operand.
6.1		* Perform Operand=Operand >> 1, A = A ^ Operand, set N and Z based off A .
6.2	Write Address	Write modified Operand.
7.1		
7.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	5b	1	unknown	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	5b	1	unknown	0002	ff	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	21	1		0003	ff	00	00	fd	Nv-BdIzc	unknown	T2
3	0003	21	1		0003	ff	00	00	fd	Nv-BdIzc	unknown	T2
4	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	unknown	T3
4	0004	01	1		0004	ff	00	00	fd	Nv-BdIzc	unknown	T3
5	0121	ff	1		0005	ff	00	00	fd	Nv-BdIzc	unknown	T4
5	0121	ff	1		0005	ff	00	00	fd	Nv-BdIzc	unknown	T4
6	0121	ff	1		0005	ff	00	00	fd	Nv-BdIzc	unknown	T5
6	0121	ff	1		0005	ff	00	00	fd	Nv-BdIzc	unknown	T5
7	0121	ff	0		0005	ff	00	00	fd	Nv-BdIzc	unknown	
7	0121	ff	0		0005	ff	00	00	fd	Nv-BdIzc	unknown	
8	0121	ff	0		0005	ff	00	00	fd	nv-BdIzc	unknown	T0
8	0121	7f	0		0005	ff	00	00	fd	nv-BdIzc	unknown	T0
9	0005	4c	1	JMP Abs	0005	ff	00	00	fd	nv-BdIzc	unknown	T1
9	0005	4c	1	JMP Abs	0005	ff	00	00	fd	nv-BdIzc	unknown	T1
10	0006	02	1		0006	80	00	00	fd	Nv-BdIzc	JMP Abs	T2
10	0006	02	1		0006	80	00	00	fd	Nv-BdIzc	JMP Abs	T2

NOP (5C)		Cycles: 4-5	Size: 3
Absolute, X (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.L.	
2.1		Inc. PC.	
2.2		Store as Address.H.	
3.1		Inc. PC. Final=Address+X. Address.L = Final.L.	
3.2	Read Address	Store as Operand. If Address.H == Final.H, skip the next cycle (next half-cycle is 5.1).	
4.1	Read Address	Address.H = Final.H (fixes high byte of address).	
4.2		Store as Operand.	
5.1			
5.2	Read PC	Store as OpCode.	
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	5c	1	unknown	0002	aa	ff	00	fd	Nv-BdIzc	LDX #	T1
2	0002	5c	1	unknown	0002	aa	ff	00	fd	Nv-BdIzc	LDX #	T1
3	0003	21	1		0003	aa	ff	00	fd	Nv-BdIzc	unknown	T2
3	0003	21	1		0003	aa	ff	00	fd	Nv-BdIzc	unknown	T2
4	0004	01	1		0004	aa	ff	00	fd	Nv-BdIzc	unknown	T3
4	0004	01	1		0004	aa	ff	00	fd	Nv-BdIzc	unknown	T3
5	0120	00	1		0005	aa	ff	00	fd	Nv-BdIzc	unknown	T4
5	0120	00	1		0005	aa	ff	00	fd	Nv-BdIzc	unknown	T4
6	0220	00	1		0005	aa	ff	00	fd	Nv-BdIzc	unknown	T0
6	0220	00	1		0005	aa	ff	00	fd	Nv-BdIzc	unknown	T0
7	0005	4c	1	JMP Abs	0005	aa	ff	00	fd	Nv-BdIzc	unknown	T1
7	0005	4c	1	JMP Abs	0005	aa	ff	00	fd	Nv-BdIzc	unknown	T1
8	0006	02	1		0006	aa	ff	00	fd	Nv-BdIzc	JMP Abs	T2
8	0006	02	1		0006	aa	ff	00	fd	Nv-BdIzc	JMP Abs	T2

EOR (5D)		Cycles: 4-5	Size: 3
Absolute, X (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.L.	
2.1		Inc. PC.	
2.2		Store as Address.H.	
3.1		Inc. PC. Final=Address+X. Address.L = Final.L.	
3.2	Read Address	Store as Operand. If Address.H == Final.H, skip the next cycle (next half-cycle is 5.1).	
4.1	Read Address	Address.H = Final.H (fixes high byte of address).	
4.2		Store as Operand.	
5.1		* Perform $A=A \wedge \text{Operand}$, set N and Z accordingly.	
5.2	Read PC	Store as OpCode.	
+X.1		A and N, Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	5d	1	EOR Abs,X	0002	aa	ff	00	fd	Nv-BdIzc	LDX #	T1
2	0002	5d	1	EOR Abs,X	0002	aa	ff	00	fd	Nv-BdIzc	LDX #	T1
3	0003	21	1		0003	aa	ff	00	fd	Nv-BdIzc	EOR Abs,X	T2
3	0003	21	1		0003	aa	ff	00	fd	Nv-BdIzc	EOR Abs,X	T2
4	0004	01	1		0004	aa	ff	00	fd	Nv-BdIzc	EOR Abs,X	T3
4	0004	01	1		0004	aa	ff	00	fd	Nv-BdIzc	EOR Abs,X	T3
5	0120	00	1		0005	aa	ff	00	fd	Nv-BdIzc	EOR Abs,X	T4
5	0120	00	1		0005	aa	ff	00	fd	Nv-BdIzc	EOR Abs,X	T4
6	0220	00	1		0005	aa	ff	00	fd	Nv-BdIzc	EOR Abs,X	T0
6	0220	00	1		0005	aa	ff	00	fd	Nv-BdIzc	EOR Abs,X	T0
7	0005	4c	1	JMP Abs	0005	aa	ff	00	fd	Nv-BdIzc	EOR Abs,X	T1
7	0005	4c	1	JMP Abs	0005	aa	ff	00	fd	Nv-BdIzc	EOR Abs,X	T1
8	0006	02	1		0006	aa	ff	00	fd	Nv-BdIzc	JMP Abs	T2
8	0006	02	1		0006	aa	ff	00	fd	Nv-BdIzc	JMP Abs	T2

LSR (5E)	Cycles: 7	Size: 3
Absolute, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC. Final=Address+X. Address.L = Final.L.
3.2	Read Address	Store as Operand.
4.1		Address.H = Final.H (fixes high byte of address).
4.2	Read Address	Store as Operand.
5.1		Set C if lowest bit of Operand is set.
5.2	Write Address	Write unmodified Operand.
6.1		Perform Operand=Operand << 1, set N and Z accordingly.
6.2	Write Address	Write modified Operand.
7.1		
7.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	5e	1	LSR Abs,X	0002	aa	ff	00	fd	Nv-BdIzc	LDX #	T1
2	0002	5e	1	LSR Abs,X	0002	aa	ff	00	fd	Nv-BdIzc	LDX #	T1
3	0003	21	1		0003	aa	ff	00	fd	Nv-BdIzc	LSR Abs,X	T2
3	0003	21	1		0003	aa	ff	00	fd	Nv-BdIzc	LSR Abs,X	T2
4	0004	00	1		0004	aa	ff	00	fd	Nv-BdIzc	LSR Abs,X	T3
4	0004	00	1		0004	aa	ff	00	fd	Nv-BdIzc	LSR Abs,X	T3
5	0020	7f	1		0005	aa	ff	00	fd	Nv-BdIzc	LSR Abs,X	T4
5	0020	7f	1		0005	aa	ff	00	fd	Nv-BdIzc	LSR Abs,X	T4
6	0120	7f	1		0005	aa	ff	00	fd	Nv-BdIzc	LSR Abs,X	T5
6	0120	7f	1		0005	aa	ff	00	fd	Nv-BdIzc	LSR Abs,X	T5
7	0120	7f	0		0005	aa	ff	00	fd	Nv-BdIzc	LSR Abs,X	
7	0120	7f	0		0005	aa	ff	00	fd	Nv-BdIzc	LSR Abs,X	
8	0120	7f	0		0005	aa	ff	00	fd	nv-BdIzc	LSR Abs,X	T0
8	0120	3f	0		0005	aa	ff	00	fd	nv-BdIzc	LSR Abs,X	T0
9	0005	4c	1	JMP Abs	0005	aa	ff	00	fd	nv-BdIzc	LSR Abs,X	T1
9	0005	4c	1	JMP Abs	0005	aa	ff	00	fd	nv-BdIzc	LSR Abs,X	T1
10	0006	02	1		0006	aa	ff	00	fd	nv-BdIzc	JMP Abs	T2
10	0006	02	1		0006	aa	ff	00	fd	nv-BdIzc	JMP Abs	T2

SRE (5F)	Cycles: 7	Size: 3
Absolute, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.L.
2.1		Inc. PC.
2.2	Read PC	Store as Address.H.
3.1		Inc. PC. Final=Address+X. Address.L = Final.L.
3.2	Read Address	Store as Operand.
4.1		Address.H = Final.H (fixes high byte of address).
4.2	Read Address	Store as Operand.
5.1		Set C if lowest bit of Operand is set.
5.2	Write Address	Write unmodified Operand.
6.1		* Perform Operand=Operand >> 1, A = A ^ Operand, set N and Z based off A .
6.2	Write Address	Write modified Operand.
7.1		
7.2	Read PC	Store as OpCode.
+X.1		A applied.

* Setting of A is delayed by 2 cycles until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	5f	1	unknown	0002	aa	ff	00	fd	Nv-BdIzc	LDX #	T1
2	0002	5f	1	unknown	0002	aa	ff	00	fd	Nv-BdIzc	LDX #	T1
3	0003	21	1		0003	aa	ff	00	fd	Nv-BdIzc	unknown	T2
3	0003	21	1		0003	aa	ff	00	fd	Nv-BdIzc	unknown	T2
4	0004	00	1		0004	aa	ff	00	fd	Nv-BdIzc	unknown	T3
4	0004	00	1		0004	aa	ff	00	fd	Nv-BdIzc	unknown	T3
5	0020	7f	1		0005	aa	ff	00	fd	Nv-BdIzc	unknown	T4
5	0020	7f	1		0005	aa	ff	00	fd	Nv-BdIzc	unknown	T4
6	0120	3f	1		0005	aa	ff	00	fd	Nv-BdIzc	unknown	T5
6	0120	3f	1		0005	aa	ff	00	fd	Nv-BdIzc	unknown	T5
7	0120	3f	0		0005	aa	ff	00	fd	Nv-BdIzc	unknown	
7	0120	3f	0		0005	aa	ff	00	fd	Nv-BdIzc	unknown	
8	0120	3f	0		0005	aa	ff	00	fd	nv-BdIzc	unknown	T0
8	0120	1f	0		0005	aa	ff	00	fd	nv-BdIzc	unknown	T0
9	0005	4c	1	JMP Abs	0005	aa	ff	00	fd	nv-BdIzc	unknown	T1
9	0005	4c	1	JMP Abs	0005	aa	ff	00	fd	nv-BdIzc	unknown	T1
10	0006	02	1		0006	b5	ff	00	fd	Nv-BdIzc	JMP Abs	T2
10	0006	02	1		0006	b5	ff	00	fd	Nv-BdIzc	JMP Abs	T2

RTS (60)	Cycles: 6	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	
2.1		Inc. PC.
2.2	Read S \$0100	
3.1		
3.2	Read (S +1) \$0100	Store as Address.L.
4.1		Increase S by 2.
4.2	Read S \$0100	Store as Address.H.
5.1		Copy Address to PC.
5.2	Read PC	
6.1		Inc. PC.
6.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
8	0021	60	1	RTS	0021	aa	ff	00	fb	Nv-BdIzc	JSR Abs	T1
8	0021	60	1	RTS	0021	aa	ff	00	fb	Nv-BdIzc	JSR Abs	T1
9	0022	00	1		0022	aa	ff	00	fb	Nv-BdIzc	RTS	T2
9	0022	00	1		0022	aa	ff	00	fb	Nv-BdIzc	RTS	T2
10	01fb	60	1		0023	aa	ff	00	fb	Nv-BdIzc	RTS	T3
10	01fb	60	1		0023	aa	ff	00	fb	Nv-BdIzc	RTS	T3
11	01fc	04	1		0023	aa	ff	00	fb	Nv-BdIzc	RTS	T4
11	01fc	04	1		0023	aa	ff	00	fb	Nv-BdIzc	RTS	T4
12	01fd	00	1		0023	aa	ff	00	fd	Nv-BdIzc	RTS	T5
12	01fd	00	1		0023	aa	ff	00	fd	Nv-BdIzc	RTS	T5
13	0004	00	1		0004	aa	ff	00	fd	Nv-BdIzc	RTS	T0
13	0004	00	1		0004	aa	ff	00	fd	Nv-BdIzc	RTS	T0
14	0005	4c	1	JMP Abs	0005	aa	ff	00	fd	Nv-BdIzc	RTS	T1
14	0005	4c	1	JMP Abs	0005	aa	ff	00	fd	Nv-BdIzc	RTS	T1
15	0006	02	1		0006	aa	ff	00	fd	Nv-BdIzc	JMP Abs	T2
15	0006	02	1		0006	aa	ff	00	fd	Nv-BdIzc	JMP Abs	T2

ADC (61)	Cycles: 6	Size: 2
Indirect, X (Read)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Operand.
2.1		Inc. PC.
2.2	Read Operand	
3.1		Set Pointer to (Operand+X) & \$FF.
3.2	Read Pointer	Store to Address.L.
4.1		
4.2	Read (Pointer+1) & \$FF	Store to Address.H.
5.1		
5.2	Read Address	Store as Operand.
6.1		* u16 Tmp = A + Operand + (C flag). $V = (\sim(u16(A) \wedge u16(Operand)) \& (u16(A) \wedge Tmp) \& \$0080) \neq 0.$ $A = u8(Tmp).$ $C = Tmp > \$FF.$ $Z = A == \$00.$ $N = (A \& \$80) \neq 0.$
6.2	Read PC	Store as OpCode.
+X.1		A and C, N, V, Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	61	1	ADC (zp,X)	0002	aa	ff	00	fd	Nv-BdIzc	LDX #	T1
2	0002	61	1	ADC (zp,X)	0002	aa	ff	00	fd	Nv-BdIzc	LDX #	T1
3	0003	21	1		0003	aa	ff	00	fd	Nv-BdIzc	ADC (zp,X)	T2
3	0003	21	1		0003	aa	ff	00	fd	Nv-BdIzc	ADC (zp,X)	T2
4	0021	01	1		0004	aa	ff	00	fd	Nv-BdIzc	ADC (zp,X)	T3
4	0021	01	1		0004	aa	ff	00	fd	Nv-BdIzc	ADC (zp,X)	T3
5	0020	23	1		0004	aa	ff	00	fd	Nv-BdIzc	ADC (zp,X)	T4
5	0020	23	1		0004	aa	ff	00	fd	Nv-BdIzc	ADC (zp,X)	T4
6	0021	01	1		0004	aa	ff	00	fd	Nv-BdIzc	ADC (zp,X)	T5
6	0021	01	1		0004	aa	ff	00	fd	Nv-BdIzc	ADC (zp,X)	T5
7	0123	6c	1		0004	aa	ff	00	fd	Nv-BdIzc	ADC (zp,X)	T0
7	0123	6c	1		0004	aa	ff	00	fd	Nv-BdIzc	ADC (zp,X)	T0
8	0004	20	1	JSR Abs	0004	aa	ff	00	fd	Nv-BdIzc	ADC (zp,X)	T1
8	0004	20	1	JSR Abs	0004	aa	ff	00	fd	Nv-BdIzc	ADC (zp,X)	T1
9	0005	4c	1		0005	16	ff	00	fd	nv-BdIzC	JSR Abs	T2
9	0005	4c	1		0005	16	ff	00	fd	nv-BdIzC	JSR Abs	T2

JAM (62)	Cycles: ∞	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	
2.1		
2.2	Read \$FFFF	
3.1		
3.2	Read \$FFFE	
4.1		
4.2	Read \$FFFE	
5.1		Repeat 5.1 and 5.2 forever.
5.2	Read \$FFFF	

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
2	0002	02	1	unknown	0002	00	00	00	fd	nv-BdIZc	LDA #	T1
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
3	0003	10	1		0003	00	00	00	fd	nv-BdIZc	unknown	T2
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
4	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T3
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
5	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T4
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
6	fffe	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	T5
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
7	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
8	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
9	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
10	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	
11	ffff	00	1		0004	00	00	00	fd	nv-BdIZc	unknown	

RRA (63)	Cycles: 8	Size: 2
Indirect, X (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Operand.
2.1		Inc. PC.
2.2	Read Operand	
3.1		Set Pointer to (Operand+X) & \$FF.
3.2	Read Pointer	Store to Address.L.
4.1		
4.2	Read (Pointer+1) & \$FF	Store to Address.H.
5.1		
5.1	Read Address	Store as Operand.
6.1		
6.2	Write Address	Write unmodified Operand.
7.1		* Tmp = (C << 7). Set C if low bit of P is set. Operand=(Operand >> 1) Tmp. A=(A ADC Operand), set C, N, V, and Z based off A.
7.2	Write Address	Write modified Operand.
8.1		
8.2	Read PC	Store as OpCode.
+X.1		A and C, N, V, Z applied.

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	63	1	unknown	0002	f0	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	63	1	unknown	0002	f0	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	10	1		0003	f0	00	00	fd	Nv-BdIzc	unknown	T2
3	0003	10	1		0003	f0	00	00	fd	Nv-BdIzc	unknown	T2
4	0010	19	1		0004	f0	00	00	fd	Nv-BdIzc	unknown	T3
4	0010	19	1		0004	f0	00	00	fd	Nv-BdIzc	unknown	T3
5	0010	19	1		0004	f0	00	00	fd	Nv-BdIzc	unknown	T4
5	0010	19	1		0004	f0	00	00	fd	Nv-BdIzc	unknown	T4
6	0011	00	1		0004	f0	00	00	fd	Nv-BdIzc	unknown	T5
6	0011	00	1		0004	f0	00	00	fd	Nv-BdIzc	unknown	T5
7	0019	ff	1		0004	f0	00	00	fd	Nv-BdIzc	unknown	
7	0019	ff	1		0004	f0	00	00	fd	Nv-BdIzc	unknown	
8	0019	ff	0		0004	f0	00	00	fd	Nv-BdIzc	unknown	
8	0019	ff	0		0004	f0	00	00	fd	Nv-BdIzc	unknown	
9	0019	ff	0		0004	f0	00	00	fd	nv-BdIzc	unknown	T0
9	0019	7f	0		0004	f0	00	00	fd	nv-BdIzc	unknown	T0
10	0004	00	1	BRK	0004	f0	00	00	fd	nv-BdIzc	unknown	T1
10	0004	00	1	BRK	0004	f0	00	00	fd	nv-BdIzc	unknown	T1
11	0005	4c	1		0005	70	00	00	fd	nv-BdIzc	BRK	T2
11	0005	4c	1		0005	70	00	00	fd	nv-BdIzc	BRK	T2

NOP (64)		Cycles: 3	Size: 2
Zero Page (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		Store as OpCode.	
3.2	Read PC		
+X.1			

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	64	1	unknown	0002	f0	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	64	1	unknown	0002	f0	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	10	1		0003	f0	00	00	fd	Nv-BdIzc	unknown	T2
3	0003	10	1		0003	f0	00	00	fd	Nv-BdIzc	unknown	T2
4	0010	19	1		0004	f0	00	00	fd	Nv-BdIzc	unknown	T0
4	0010	19	1		0004	f0	00	00	fd	Nv-BdIzc	unknown	T0
5	0004	a4	1	LDY zp	0004	f0	00	00	fd	Nv-BdIzc	unknown	T1
5	0004	a4	1	LDY zp	0004	f0	00	00	fd	Nv-BdIzc	unknown	T1
6	0005	4c	1		0005	f0	00	00	fd	Nv-BdIzc	LDY zp	T2
6	0005	4c	1		0005	f0	00	00	fd	Nv-BdIzc	LDY zp	T2

ADC (65)		Cycles: 3	Size: 2
Zero Page (Read)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1	Read PC	Inc. PC.	
1.2		Store as Address.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		* u16 Tmp = A + Operand + (C flag). $V = (\sim(u16(A) \wedge u16(Operand)) \& (u16(A) \wedge Tmp) \& \$0080) \neq 0.$ $A = u8(Tmp).$ $C = Tmp > \$FF.$ $Z = A == \$00.$ $N = (A \& \$80) \neq 0.$	
3.2	Read PC	Store as OpCode.	
+X.1		A and C , N , V , Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	65	1	ADC zp	0002	f0	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	65	1	ADC zp	0002	f0	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	10	1		0003	f0	00	00	fd	Nv-BdIzc	ADC zp	T2
3	0003	10	1		0003	f0	00	00	fd	Nv-BdIzc	ADC zp	T2
4	0010	19	1		0004	f0	00	00	fd	Nv-BdIzc	ADC zp	T0
4	0010	19	1		0004	f0	00	00	fd	Nv-BdIzc	ADC zp	T0
5	0004	a4	1	LDY zp	0004	f0	00	00	fd	Nv-BdIzc	ADC zp	T1
5	0004	a4	1	LDY zp	0004	f0	00	00	fd	Nv-BdIzc	ADC zp	T1
6	0005	4c	1		0005	09	00	00	fd	nv-BdIzC	LDY zp	T2
6	0005	4c	1		0005	09	00	00	fd	nv-BdIzC	LDY zp	T2

ROR (66)	Cycles: 5	Size: 2
Zero Page (Read/Modify/Write)		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	Store as Address.
2.1		Inc. PC.
2.2	Read Address	Store as Operand.
3.1		
3.2	Write Address	Write unmodified Operand.
4.1		* Tmp = (C << 7). Set C if low bit of Operand is set. Operand=(Operand >> 1) Tmp. Set N, and Z based off Operand.
4.2	Write Address	Write modified Operand.
5.1		
5.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
4	0002	66	1	ROR zp	0002	aa	00	00	fd	nv-BdIZC	SEC	T1
4	0002	66	1	ROR zp	0002	aa	00	00	fd	nv-BdIZC	SEC	T1
5	0003	10	1		0003	aa	00	00	fd	nv-BdIZC	ROR zp	T2
5	0003	10	1		0003	aa	00	00	fd	nv-BdIZC	ROR zp	T2
6	0010	7f	1		0004	aa	00	00	fd	nv-BdIZC	ROR zp	T3
6	0010	7f	1		0004	aa	00	00	fd	nv-BdIZC	ROR zp	T3
7	0010	7f	0		0004	aa	00	00	fd	nv-BdIZC	ROR zp	T4
7	0010	7f	0		0004	aa	00	00	fd	nv-BdIZC	ROR zp	T4
8	0010	7f	0		0004	aa	00	00	fd	Nv-BdIzC	ROR zp	T0
8	0010	bf	0		0004	aa	00	00	fd	Nv-BdIzC	ROR zp	T0
9	0004	a4	1	LDY zp	0004	aa	00	00	fd	Nv-BdIzC	ROR zp	T1
9	0004	a4	1	LDY zp	0004	aa	00	00	fd	Nv-BdIzC	ROR zp	T1
10	0005	4c	1		0005	aa	00	00	fd	Nv-BdIzC	LDY zp	T2
10	0005	4c	1		0005	aa	00	00	fd	Nv-BdIzC	LDY zp	T2

RRA (67)	Cycles: 5	Size: 2	
Zero Page (Read/Modify/Write)			
Cycle	R/W	Desc	
-X.2	Read PC	Store as OpCode.	
1.1		Inc. PC.	
1.2		Store as Address.	
2.1		Inc. PC.	
2.2		Store as Operand.	
3.1		Write unmodified Operand. * Tmp = (C << 7). Set C if low bit of P is set. Operand=(Operand >> 1) Tmp. A =(A ADC Operand), set C , N , V , and Z based off A .	
3.2			
4.1			
4.2			Write Address
5.1			
5.2		Read PC	
+X.1		A and C , N , V , Z applied.	

* Setting of A/flags is delayed by 1 cycle until the start of the following instruction.

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	67	1	unknown	0002	38	00	00	fd	nv-BdIzc	LDA #	T1
2	0002	67	1	unknown	0002	38	00	00	fd	nv-BdIzc	LDA #	T1
3	0003	10	1		0003	38	00	00	fd	nv-BdIzc	unknown	T2
3	0003	10	1		0003	38	00	00	fd	nv-BdIzc	unknown	T2
4	0010	2f	1		0004	38	00	00	fd	nv-BdIzc	unknown	T3
4	0010	2f	1		0004	38	00	00	fd	nv-BdIzc	unknown	T3
5	0010	2f	0		0004	38	00	00	fd	nv-BdIzc	unknown	T4
5	0010	2f	0		0004	38	00	00	fd	nv-BdIzc	unknown	T4
6	0010	2f	0		0004	38	00	00	fd	nv-BdIzc	unknown	T0
6	0010	17	0		0004	38	00	00	fd	nv-BdIzc	unknown	T0
7	0004	a4	1	LDY zp	0004	38	00	00	fd	nv-BdIzc	unknown	T1
7	0004	a4	1	LDY zp	0004	38	00	00	fd	nv-BdIzc	unknown	T1
8	0005	4c	1		0005	50	00	00	fd	nv-BdIzc	LDY zp	T2
8	0005	4c	1		0005	50	00	00	fd	nv-BdIzc	LDY zp	T2

PLA (68)	Cycles: 4	Size: 1
Implied		
Cycle	R/W	Desc
-X.2	Read PC	Store as OpCode.
1.1		Inc. PC.
1.2	Read PC	To be discarded.
2.1		
2.2	Read S \$0100	Tom Harte tests require this to be stored to A .
3.1		Inc. S .
3.2	Read S \$0100	Store as Operand.
4.1		A =Operand, set N and Z based off A .
4.2	Read PC	Store as OpCode.
+X.1		

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State
2	0002	68	1	PLA	0002	80	00	00	fd	Nv-BdIzc	LDA #	T1
2	0002	68	1	PLA	0002	80	00	00	fd	Nv-BdIzc	LDA #	T1
3	0003	10	1		0003	80	00	00	fd	Nv-BdIzc	PLA	T2
3	0003	10	1		0003	80	00	00	fd	Nv-BdIzc	PLA	T2
4	01fd	00	1		0003	80	00	00	fd	Nv-BdIzc	PLA	T3
4	01fd	00	1		0003	80	00	00	fd	Nv-BdIzc	PLA	T3
5	01fe	00	1		0003	80	00	00	fe	Nv-BdIzc	PLA	T0
5	01fe	00	1		0003	80	00	00	fe	Nv-BdIzc	PLA	T0
6	0003	10	1	BPL	0003	00	00	00	fe	nv-BdIZc	PLA	T1
6	0003	10	1	BPL	0003	00	00	00	fe	nv-BdIZc	PLA	T1
7	0004	00	1		0004	00	00	00	fe	nv-BdIZc	BPL	T2
7	0004	00	1		0004	00	00	00	fe	nv-BdIZc	BPL	T2

