ADC
add with carry

N V Z C

arg	hex	len	time
#n	$69	2	2
ZP	$65	2	3
ZP,X	$75	2	4
ABS	$6D	3	4
ABS,X	$7D	3	4+
ABS,Y	$79	3	4+
(ZP,X)	$61	2	6
(ZP),Y	$71	2	5+

AND
bitwise AND with accumulator

N Z

arg	hex	len	time
#n	$29	2	2
ZP	$25	2	3
ZP,X	$35	2	4
ABS	$2D	3	4
ABS,X	$3D	3	4+
ABS,Y	$39	3	4+
(ZP,X)	$21	2	6
(ZP),Y	$31	2	5+

ASL
arithmetic shift left

N Z C

arg	hex	len	time
A	$0A	1	2
ZP	$06	2	5
ZP,X	$16	2	6
ABS	$0E	3	6
ABS,X	$1E	3	7

BIT
test bits

N V Z

arg	hex	len	time
ZP	$24	2	3
ABS	$2C	3	4

BRK
break

B (on stack)

arg	hex	len	time
rel	$00	2	7

CMP
compare accumulator

N Z C

arg	hex	len	time
#n	$C9	2	2
ZP	$C5	2	3
ZP,X	$D5	2	4
ABS	$CD	3	4
ABS,X	$DD	3	4+
ABS,Y	$D9	3	4+
(ZP,X)	$C1	2	6
(ZP),Y	$D1	2	5+

	Z	C
A<M	0	0
A=M	1	1
A>M	0	1

CPX
compare X register

N Z C

arg	hex	len	time
#n	$E0	2	2
ZP	$E4	2	3
ABS	$EC	3	4

CPY
compare Y register

N Z C

arg	hex	len	time
#n	$C0	2	2
ZP	$C4	2	3
ABS	$CC	3	4

DEC
decrement memory

N Z

arg	hex	len	time
ZP	$C6	2	5
ZP,X	$D6	2	6
ABS	$CE	3	6
ABS,X	$DE	3	7

EOR
bitwise exclusive OR

N Z

arg	hex	len	time
#n	$49	2	2
ZP	$45	2	3
ZP,X	$55	2	4
ABS	$4D	3	4
ABS,X	$5D	3	4+
ABS,Y	$59	3	4+
(ZP,X)	$41	2	6
(ZP),Y	$51	2	5+

INC
increment memory

N Z

arg	hex	len	time
ZP	$E6	2	5
ZP,X	$F6	2	6
ABS	$EE	3	6
ABS,X	$FE	3	7

JMP
jump

arg	hex	len	time
ABS	$4C	3	3
(ABS)	$6C	3	5

JSR
jump to subroutine

arg	hex	len	time
ABS	$20	3	6

LDA
load accumulator

N Z

arg	hex	len	time
#n	$A9	2	2
ZP	$A5	2	3
ZP,X	$B5	2	4
ABS	$AD	3	4
ABS,X	$BD	3	4+
ABS,Y	$B9	3	4+
(ZP,X)	$A1	2	6
(ZP),Y	$B1	2	5+

LDX
load X register

N Z

arg	hex	len	time
#n	$A2	2	2
ZP	$A6	2	3
ZP,Y	$B6	2	4
ABS	$AE	3	4
ABS,Y	$BE	3	4+

LDY
load Y register

N Z

arg	hex	len	time
#n	$A0	2	2
ZP	$A4	2	3
ZP,X	$B4	2	4
ABS	$AC	3	4
ABS,X	$BC	3	4+

LSR
logical shift right

N Z C

arg	hex	len	time
A	$4A	1	2
ZP	$46	2	5
ZP,X	$56	2	6
ABS	$4E	3	6
ABS,X	$5E	3	7

NOP
no operation

arg	hex	len	time
-	$EA	1	2

ORA
bitwise OR with accumulator

N Z

arg	hex	len	time
#n	$09	2	2
ZP	$05	2	3
ZP,X	$15	2	4
ABS	$0D	3	4
ABS,X	$1D	3	4+
ABS,Y	$19	3	4+
(ZP,X)	$01	2	6
(ZP),Y	$11	2	5+

ROL
rotate left

N Z C

arg	hex	len	time
A	$2A	1	2
ZP	$26	2	5
ZP,X	$36	2	6
ABS	$2E	3	6
ABS,X	$3E	3	7

ROR
rotate right

N Z C

arg	hex	len	time
A	$6A	1	2
ZP	$66	2	5
ZP,X	$76	2	6
ABS	$6E	3	6
ABS,X	$7E	3	7

RTI
return from interrupt

All (from stack)

arg	hex	len	time
-	$40	1	6

RTS
return from subroutine

arg	hex	len	time
-	$60	1	6

SBC
subtract with carry

N V Z C

arg	hex	len	time
#n	$E9	2	2
ZP	$E5	2	3
ZP,X	$F5	2	4
ABS	$ED	3	4
ABS,X	$FD	3	4+
ABS,Y	$F9	3	4+
(ZP,X)	$E1	2	6
(ZP),Y	$F1	2	5+

STA
store accumulator

arg	hex	len	time
ZP	$85	2	3
ZP,X	$95	2	4
ABS	$8D	3	4
ABS,X	$9D	3	5
ABS,Y	$99	3	5
(ZP,X)	$81	2	6
(ZP),Y	$91	2	6

STX
store X register

arg	hex	len	time
ZP	$86	2	3
ZP,Y	$96	2	4
ABS	$8E	3	4

STY
store Y register

arg	hex	len	time
ZP	$84	2	3
ZP,X	$94	2	4
ABS	$8C	3	4

Branch Instructions:

BPL
branch on plus

arg	hex	len	time
rel	$10	2	2/3+

BMI
branch on minus

arg	hex	len	time
rel	$30	2	2/3+

BVC
branch on overflow clear

arg	hex	len	time
rel	$50	2	2/3+

BVS
branch on overflow set

arg	hex	len	time
rel	$70	2	2/3+

BCC
branch on carry clear

arg	hex	len	time
rel	$90	2	2/3+

BCS
branch on carry set

arg	hex	len	time
rel	$B0	2	2/3+

BNE
branch on not equal

arg	hex	len	time
rel	$D0	2	2/3+

BEQ
branch on equal

arg	hex	len	time
rel	$F0	2	2/3+

Flag Instructions:

CLC
clear carry

C

arg	hex	len	time
-	$18	1	2

SEC
set carry

C

arg	hex	len	time
-	$38	1	2

CLI
clear interrupt

I

arg	hex	len	time
-	$58	1	2

SEI
set interrupt

I

arg	hex	len	time
-	$78	1	2

CLV
clear overflow

V

arg	hex	len	time
-	$B8	1	2

CLD
clear decimal

D

arg	hex	len	time
-	$D8	1	2

SED
set decimal

D

arg	hex	len	time
-	$F8	1	2

Stack Instructions:

TXS
transfer X to SP

arg	hex	len	time
-	$9A	1	2

TSX
transfer SP to X

N Z

arg	hex	len	time
-	$BA	1	2

PHA
push accumulator

arg	hex	len	time
-	$48	1	3

PLA
pull accumulator

N Z

arg	hex	len	time
-	$68	1	4

PHP
push processor status

arg	hex	len	time
-	$08	1	3

PLP
pull processor status

All (from stack)

arg	hex	len	time
-	$28	1	4

Register Instructions:

TAX
transfer A to X

N Z

arg	hex	len	time
-	$AA	1	2

TXA
transfer X to A

N Z

arg	hex	len	time
-	$8A	1	2

TAY
transfer A to Y

N Z

arg	hex	len	time
-	$A8	1	2

TYA
transfer Y to A

N Z

arg	hex	len	time
-	$98	1	2

DEX
decrement X

N Z

arg	hex	len	time
-	$CA	1	2

INX
increment X

N Z

arg	hex	len	time
-	$E8	1	2

DEY
decrement Y

N Z

arg	hex	len	time
-	$88	1	2

INY
increment Y

N Z

arg	hex	len	time
-	$C8	1	2

6502 Registers:

SP - Stack Pointer
PC - Program Counter
X - Index Register
Y - Index Register
A - Accumulator
F - Flag Register (Processor Status)

N	V	*	B	D	I	Z	C

Branch Execution Time:

A branch that is not taken requires two machine cycles. Add one if the branch is taken and add another if the branch crosses the page boundary.

Vectors:

$FFFA - NMI
$FFFC - RESET
$FFFE - IRQ