

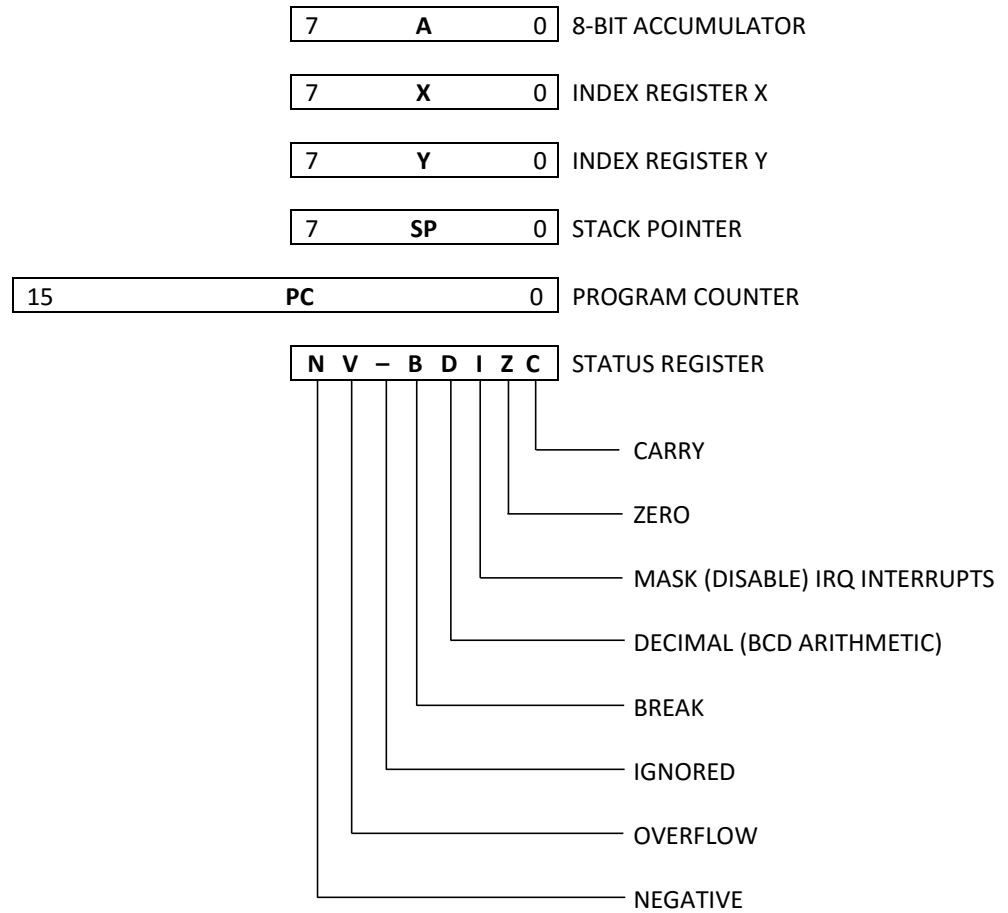
6502 Reference Guide

MOS TECHNOLOGY

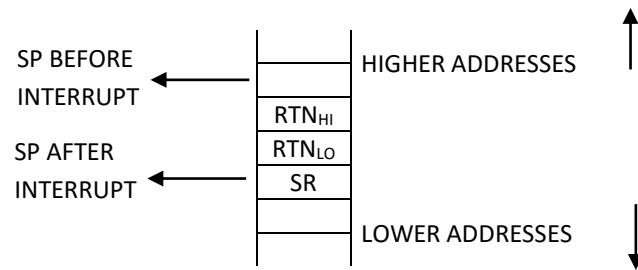
ALEX BUSMAN

6502 INSTRUCTION SET

PROGRAMMING MODEL



STACK AND MEMORY LAYOUT



INTERRUPT VECTOR LOCATIONS

\$FFFE, \$FFFF	IRQ/BRK: Interrupt Request or Break
\$FFFC, \$FFFD	RESET: Power-On (POR) or External Reset
\$FFFA, \$FFFB	NMI: Non-maskable Interrupt

NOTATION USED IN INSTRUCTION SET SUMMARY

CPU Register Notation

Accumulator – A or a
 Index Register X – X or x
 Index Register Y – Y or y
 Stack Pointer – SP, sp, or s
 Program Counter – PC, pc, or p
 Status Register/Condition Code Register – SR, CCR, c

Explanation of Italic Expressions in Source Form Column

opr8i – 8-bit immediate value
opr8a – 8-bit address used with zero page address mode
opr16a – 16-bit address value
opr8 – Any integer in the range -128 ... +127
opr16 – Any integer in the range -32,768 ... +65,535
xy – Index X or Index Y

Operators

+ – Addition
 - – Subtraction
 · – Logical AND
 + – Logical OR (inclusive)
 ⊕ – Logical exclusive OR
 : – Concatenate
 ⇒ – Transfer

Address Mode Notation

- ACC – Accumulator; no operands; operation performed on accumulator.
- IMP – Implied; no operands in object code
- IMM – Immediate; operand in object code
- ZER – Zero Page; operand is the lower byte of an address from \$0000 to \$00FF
- ABS – Absolute; operand is a 16-bit address
- REL – Two's complement relative offset; for branch instructions
- IDX1 – 8-bit signed offset from X or Y; 1 extension byte
- IDX2 – 16-bit signed offset from X or Y; 2 extensions bytes
- [IDX1] – Indexed-indirect; 8-bit offset from X or Y

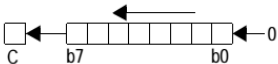
Machine Coding

- ee – High-order byte of a 16-bit constant offset for indexed addressing.
- ff – Low-order eight bits of an 8-bit signed constant offset for indexed addressing, or low-order byte of a 16-bit constant offset for indexed addressing.
- hh – High-order byte of a 16-bit address.
- ii – 8-bit immediate data value.
- ll – Low-order byte of a 16-bit extended address.
- rr – Signed relative offset \$80 (-128) to \$7F (+127). Offset relative to the byte following the relative offset byte.
- zz – 8-bit zero page address \$0000 to \$00FF. (High byte assumed to be \$00).


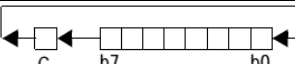
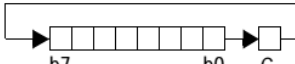
Condition Codes Columns

- – Status bit not affected by operation.
- 0 – Status bit cleared by operation.
- 1 – Status bit set by operation.
- Δ – Status bit affected by operation.

INSTRUCTION SET SUMMARY

Source Form	Operation	Addr. Mode	Machine Coding (hex)	B D I	N Z V C	Cycles
ADC #opr8i ADC opr8a ADC opr16a ADC oprx8,x ADC oprx16,x ADC oprx16,y ADC (opr8,x) ADC (opr8),y	$(A) + (M) + C \Rightarrow A$ Add Memory to Accumulator with Carry	IMM ZER ABS IDX1 IDX2 IDX2 [IDX1] [IDX1]	69 ii 65 zz 6D hh ll 75 ff 7D ee ff 79 ee ff 61 ff 71 ff	---	$\Delta \Delta \Delta \Delta$	2 3 4 4 4* 4* 6 5*
AND #opr8i AND opr8a AND opr16a AND oprx8,x AND oprx16,x AND oprx16,y AND (opr8,x) AND (opr8),y	$(A) \cdot (M) \Rightarrow A$ AND Memory with Accumulator	IMM ZER ABS IDX1 IDX2 IDX2 [IDX1] [IDX1]	29 ii 25 zz 2D hh ll 35 ff 3D ee ff 39 ee ff 21 ff 31 ff	---	$\Delta \Delta --$	2 3 4 4 4* 4* 6 5*
ASL A ASL opr8a ASL opr16a ASL oprx8,x ASL oprx16,x	 Arithmetic Shift Left (Memory or A)	ACC ZER ABS IDX1 IDX2	0A 06 zz 0E hh ll 16 ff 1E ee ff	---	$\Delta \Delta - \Delta$	2 5 6 6 7
BCC rel8	Branch if Carry Clear (if C = 0)	REL	90 rr	---	----	2**
BCS rel8	Branch if Carry Set (if C = 1)	REL	B0 rr	---	----	2**
BEQ rel8	Branch if Equal (if Z = 1)	REL	F0 rr	---	----	2**
BIT opr8a BIT opr16a	$(A) \cdot (M)$ Test Bits in Memory with Accumulator	ZER ABS	24 zz 2C hh ll	---	$\Delta \Delta \Delta -$	3 4
BMI rel8	Branch if Minus (if N = 1)	REL	30 rr	---	----	2**
BNE rel8	Branch if Not Equal (if Z = 0)	REL	D0 rr	---	----	2**
BPL rel8	Branch if Plus (if N = 0)	REL	10 rr	---	----	2**
BRK	Force Break	IMP	00	1-1	----	7
BVC rel8	Branch if Overflow Bit Clear (if V = 0)	REL	50 rr	---	----	2**
BVS rel8	Branch if Overflow Bit Set (if V = 1)	REL	70 rr	---	----	2**
CLC	$0 \Rightarrow C$ Clear Carry Flag	IMP	18	---	---0	2
CLD	$0 \Rightarrow D$ Clear Decimal Mode	IMP	D8	-0-	----	2
CLI	$0 \Rightarrow I$ Clear Interrupt Disable Bit	IMP	58	--0	----	2
CLV	$0 \Rightarrow V$ Clear Overflow Flag	IMP	B8	---	--0-	2
CMP #opr8i CMP opr8a CMP opr16a CMP oprx8,x CMP oprx16,x CMP oprx16,y CMP (opr8,x) CMP (opr8),y	$(A) - (M)$ Compare Memory with Accumulator	IMM ZER ABS IDX1 IDX2 IDX2 [IDX1] [IDX1]	C9 ii C5 zz CD hh ll D5 ff DD ee ff D9 ee ff C1 ff D1 ff	---	$\Delta \Delta \Delta -$	2 3 4 4 4* 4* 6 5*

Source Form	Operation	Addr. Mode	Machine Coding (hex)	B D I	N Z V C	Cycles
CPX #opr8i CPX opr8a CPX opr16a	(X) – (M) Compare Memory and Index X	IMM ZER ABS	E0 ii E4 zz EC hh ll	---	Δ Δ Δ –	2 3 4
CPY #opr8i CPY opr8a CPY opr16a	(Y) – (M) Compare Memory and Index Y	IMM ZER ABS	C0 ii C4 zz CC hh ll	---	Δ Δ Δ –	2 3 4
DEC opr8a DEC opr16a DEC oprx8,x DEC oprx16,x	(M) – \$01 ⇒ M Decrement Memory by 1	ZER ABS IDX1 IDX2	C6 zz CE hh ll D6 ff DE ee ff	---	Δ Δ --	5 6 6 7
DEX	(X) – \$01 ⇒ X Decrement Index X by 1	IMP	CA	---	Δ Δ --	2
DEY	(Y) – \$01 ⇒ Y Decrement Index Y by 1	IMP	88	---	Δ Δ --	2
EOR #opr8i EOR opr8a EOR opr16a EOR oprx8,x EOR oprx16,x EOR oprx16,y EOR (oprx8,x) EOR (oprx8),y	(A) ⊕ (M) ⇒ A Exclusive-OR Memory with Accumulator	IMM ZER ABS IDX1 IDX2 IDX2 [IDX1] [IDX1]	49 ii 45 zz 4D hh ll 55 ff 5D ee ff 59 ee ff 41 ff 51 ff	---	Δ Δ --	2 3 4 4 4* 4* 6 5*
INC opr8a INC opr16a INC oprx8,x INC oprx16,x	(M) + \$01 ⇒ M Increment Memory by 1	ZER ABS IDX1 IDX2	E6 zz EE hh ll F6 ff FE ee ff	---	Δ Δ --	5 6 6 7
INX	(X) + \$01 ⇒ X Increment Index X by 1	IMP	E8	---	Δ Δ --	2
INY	(Y) + \$01 ⇒ Y Increment Index Y by 1	IMP	C8	---	Δ Δ --	2
JMP opr16a JMP (opr16a)	Jump to New Location Routine Address ⇒ PC	ABS IND	4C hh ll 6C ee ff	---	----	3 5
JSR opr16a	Jump to Subroutine Saving Return Address (SP) – 2 ⇒ SP RTN _{HI} :RTN _{LO} ⇒ M _(SP+2) :M _(SP+1) Subroutine address ⇒ PC	ABS	20 hh ll	---	----	6
LDA #opr8i LDA opr8a LDA opr16a LDA oprx8,x LDA oprx16,x LDA oprx16,y LDA (oprx8,x) LDA (oprx8),y	(M) ⇒ A Load Accumulator with Memory	IMM ZER ABS IDX1 IDX2 IDX2 [IDX1] [IDX1]	A9 ii A5 zz AD hh ll B5 ff BD ee ff B9 ee ff A1 ff B1 ff	---	Δ Δ --	2 3 4 4 4* 4* 6 5*
LDX #opr8i LDX opr8a LDX opr16a LDX oprx8,y LDX oprx16,y	(M) ⇒ X Load Index X with Memory	IMM ZER ABS IDX1 IDX2	A2 ii A6 zz AE hh ll B6 ff BE ee ff	---	Δ Δ --	2 3 4 4 4*

Source Form	Operation	Addr. Mode	Machine Coding (hex)	B D I	N Z V C	Cycles
LDY #opr8i LDY opr8a LDY opr16a LDY oprx8,x LDY oprx16,x	$(M) \Rightarrow Y$ Load Index Y with Memory	IMM ZER ABS IDX1 IDX2	A0 ii A4 zz AC hh ll B4 ff BC ee ff	---	$\Delta \Delta --$	2 3 4 4 4*
LSR A LSR opr8a LSR opr16a LSR oprx8,x LSR oprx16,x	 Logical Shift Right	ACC ZER ABS IDX1 IDX2	4A 46 zz 4E hh ll 56 ff 5E ee ff	---	$-\Delta -\Delta$	2 5 6 6 7
NOP	No Operation	IMP	EA	---	----	2
ORA #opr8i ORA opr8a ORA opr16a ORA oprx8,x ORA oprx16,x ORA oprx16,y ORA (oprx8,x) ORA (oprx8),y	$(A) + (M) \Rightarrow A$ Logical OR Memory with Accumulator	IMM ZER ABS IDX1 IDX2 IDX2 [IDX1] [IDX1]	09 ii 05 zz 0D hh ll 15 ff 1D ee ff 19 ee ff 01 ff 11 ff	---	$\Delta \Delta --$	2 3 4 4 4* 4* 6 5*
PHA	$(A) \Rightarrow M_{(SP)}$ $(SP) - 1 \Rightarrow SP$ Push Accumulator on Stack	IMP	48	---	----	3
PHP	$(SR) \Rightarrow M_{(SP)}$ $(SP) - 1 \Rightarrow SP$ Push Processor Status on Stack	IMP	08	---	----	3
PLA	$(SP) + 1 \Rightarrow SP$ $(M_{(SP)}) \Rightarrow A$ Pull Accumulator from Stack	IMP	68	---	$\Delta \Delta --$	4
PLP	$(SP) + 1 \Rightarrow SP$ $(M_{(SP)}) \Rightarrow SR$ Pull Processor Status from Stack	IMP	28	$\Delta \Delta \Delta$	$\Delta \Delta \Delta \Delta$	4
ROL A ROL opr8a ROL opr16a ROL oprx8,x ROL oprx16,x	 Rotate Left through Carry	ACC ZER ABS IDX1 IDX2	2A 26 zz 2E hh ll 36 ff 3E ee ff	---	$\Delta \Delta -\Delta$	2 5 6 6 7
ROR A ROR opr8a ROR opr16a ROR oprx8,x ROR oprx16,x	 Rotate Right through Carry	ACC ZER ABS IDX1 IDX2	6A 66 zz 6E hh ll 76 ff 7E ee ff	---	$\Delta \Delta -\Delta$	2 5 6 6 7
RTI	$(M_{(SP)}) \Rightarrow SR; (SP) + 1 \Rightarrow SP$ $(M_{(SP)}:M_{(SP+1)}) \Rightarrow PC_{HI}:PC_{LO}; (SP) + 2 \Rightarrow SP$ Return from Interrupt	IMP	40	$\Delta \Delta \Delta$	$\Delta \Delta \Delta \Delta$	6
RTS	$(M_{(SP)}:M_{(SP+1)}) \Rightarrow PC_{HI}:PC_{LO}; (SP) + 2 \Rightarrow SP$ Return from Subroutine	IMP	60	---	----	6

Source Form	Operation	Addr. Mode	Machine Coding (hex)	B D I	N Z V C	Cycles
SBC #opr8i SBC opr8a SBC opr16a SBC oprx8,x SBC oprx16,x SBC oprx16,y SBC (oprx8,x) SBC (oprx8),y	(A) – (M) – (C) ⇒ A Subtract Memory from A with Borrow	IMM ZER ABS IDX1 IDX2 IDX2 [IDX1] [IDX1]	E9 ii E5 zz ED hh ll F5 ff FD ee ff F9 ee ff E1 ff F1 ff	---	Δ Δ Δ Δ	2 3 4 4 4* 4* 6 5*
SEC	1 ⇒ C Set Carry Flag	IMP	38	---	---1	2
SED	1 ⇒ D Set Decimal Mode	IMP	F8	-1-	----	2
SEI	1 ⇒ I Set Interrupt Disable Bit	IMP	78	--1	----	2
STA opr8a STA opr16a STA oprx8,x STA oprx16,x STA oprx16,y STA (oprx8,x) STA (oprx8),y	(A) ⇒ M Store Accumulator in Memory	ZER ABS IDX1 IDX2 IDX2 [IDX1] [IDX1]	85 zz 8D hh ll 95 ff 9D ee ff 99 ee ff 81 ff 91 ff	---	----	3 4 4 5 5 6 6
STX opr8a STX opr16a STX oprx8,y	(X) ⇒ M Store Index X in Memory	ZER ABS IDX1	86 zz 8E hh ll 96 ff	---	----	3 4 4
STY opr8a STY opr16a STY oprx8,x	(Y) ⇒ M Store Index Y in Memory	ZER ABS IDX1	84 zz 8C hh ll 94 ff	---	----	3 4 4
TAX	(A) ⇒ X Transfer Accumulator to Index X	IMP	AA	---	Δ Δ --	2
TAY	(A) ⇒ Y Transfer Accumulator to Index Y	IMP	A8	---	Δ Δ --	2
TSX	(SP) ⇒ X Transfer Stack Pointer to Index X	IMP	BA	---	Δ Δ --	2
TXA	(X) ⇒ A Transfer Index X to Accumulator	IMP	8A	---	Δ Δ --	2
TXS	(X) ⇒ SP Transfer Index X to Stack Pointer	IMP	9A	---	Δ Δ --	2
TYA	(Y) ⇒ A Transfer Index Y to Accumulator	IMP	98	---	Δ Δ --	2

* Add 1 to cycles if page boundary is crossed

** Add 1 to cycles if branch occurs on same page; Add 2 to cycles if branch occurs on different page.