6502 INSTRUCTION SET

Instructions often frighten microcomputer users who are new to programming. Taken in isolation, the operations involved in the execution of a single instruction are usually easy to follow. The purpose of this chapter is to isolate and explain those operations.

Why are the instructions of a microcomputer referred to as an instruction "set"? Because the microcomputer designer selects (or at least should select) the instructions with great care; it must be easy to execute complex operations as a sequence of simple events, each of which is represented by one instruction from a well-designed instruction "set".

Remaining consistent with An Introduction to Microcomputers: Volume 2, Table 3-4 summarizes the 6502 microcomputer instruction set, with similar instructions grouped together. Individual instructions are listed numerically by object code in Table 3-5 and in alphabetical order by instruction mnemonic in Table 3-6. Table 3-6 also compares the 6800 instruction set with that of the 6502. We will discuss the 6800 and 6502 much later in this chapter, after detailing the 6502 instruction set.

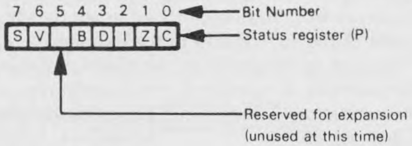
In addition to simply stating what each instruction does, the individual instruction descriptions discuss the purpose of the instruction within normal programming logic.

ABBREVIATIONS

These are the abbreviations used in this chapter:

The registers:

A	Accumulator
X	Index Register X
Y	Index Register Y
PC	Program Counter
SP	Stack Pointer
P	Status register, with bits assigned as follows:



Statuses:

S	Sign or Negative status
V	Overflow status
B	Break status
D	Decimal Mode status
I	Interrupt Disable status
Z	Zero status
C	Carry status

Symbols in the column labeled STATUS:

(blank)	Operation does not affect status
X	Operation affects status
0	Operation clears status
1	Operation sets status
6	Operation reflects bit 6 of memory location
7	Operation reflects bit 7 of memory location
addr	8 bits of absolute or base address
[addr+1,addr]	The address constructed from the contents of memory locations addr and addr+1. This address is used in post-indexed indirect addressing.
addr16	16 bits of absolute or base address
data	8 bits of immediate data
disp	An 8-bit, signed address displacement
label	16-bit absolute address, destination of Jump or Jump-to-Subroutine
PC(HI)	The high-order 8 bits of the Program Counter
PC(LO)	The low-order 8 bits of the Program Counter
pp	The second byte of a two- or three-byte instruction object code
qq	The third byte of a three-byte object code
[ ]	Contents of the memory location designated inside the brackets. For example, [FFFE] represents the contents of memory location FFFE$_{16}$; [addr16+X] represents the contents of the location addressed by adding the contents of register X to addr16; [SP] represents the value at the top of the Stack (contents of the memory location addressed by the Stack Pointer).
[[ ]]	Indirect addressing: the contents of the memory byte addressed by the contents of the memory location designated within the inner brackets. For example, [[addr+X]] represents the contents of a memory location addressed via pre-indexed indirect addressing.
+	Addition — either unsigned binary addition or BCD addition, depending on the condition of the Decimal Mode status.
−	Binary or BCD subtraction, performed by adding the twos complement of the subtrahend to the minuend.
—	The ones complement of the quantity denoted beneath the bar; for example, $\overline{A}$ represents the complement of the contents of the Accumulator; $\overline{C}$ represents the complement of the value of the Carry status.
$\wedge$	Logical AND
$\vee$	Logical OR
$\veebar$	Logical Exclusive-OR
←	Data is transferred in the direction of the arrow.

INSTRUCTION MNEMONICS

Table 3-4 summarizes the 6502 instruction set. The INSTRUCTION column shows the instruction mnemonic (LDA, STA, CLC) and the operands, if any, used with the instruction mnemonic.

The fixed part of an assembly language instruction is shown in UPPER CASE. The variable part (immediate data, address, or label) is shown in lower case.

If a mnemonic has more than one type of operand, each type is listed separately without repeating the mnemonic. For instance, some examples of the format entry

```
        STX
             addr
             addr,Y
             addr16
are:    STX  $75
        STX  $60,Y
        STX  $4276
```

INSTRUCTION OBJECT CODES

For instruction bytes without variations, object codes are represented as two hexadecimal digits (e.g., 8A). For instruction bytes with variations, the object code is shown as eight binary digits (e.g., 101aaa01).

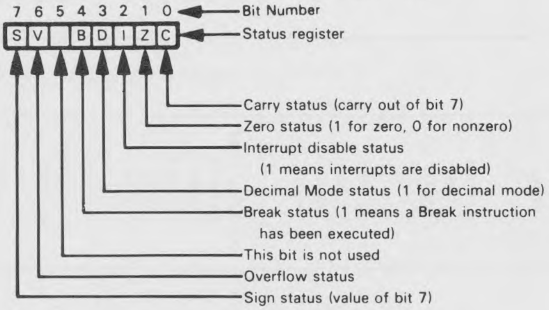
The object code and instruction length in bytes is shown in Table 3-4 for each instruction variation. Table 3-5 lists the object codes in numerical order, and Table 3-6 shows the corresponding object codes for the mnemonics, listed in alphabetical order.

INSTRUCTION EXECUTION TIMES

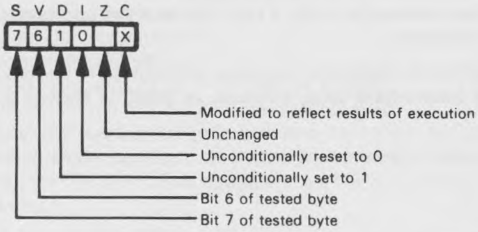
Table 3-4 lists the instruction execution times in numbers of clock periods. Actual execution time can be derived by dividing the given number of clock periods by the clock speed. For example, for an instruction that requires 5 clock periods, a 2 MHz clock will result in a 2.5 microsecond execution time.

STATUS

The status flags are stored in the Status register (P) as follows:


7 6 5 4 3 2 1 0
Bit Number
S V B D I Z C
Status register
Carry status (carry out of bit 7)
Zero status (1 for zero, 0 for nonzero)
Interrupt disable status (1 means interrupts are disabled)
Decimal Mode status (1 for decimal mode)
Break status (1 means a Break instruction has been executed)
This bit is not used
Overflow status
Sign status (value of bit 7)

In the individual instruction descriptions, the effect of instruction execution on status is illustrated as follows:



STATUS CHANGES WITH INSTRUCTION EXECUTION

An X identifies a status that is set or reset. A 0 identifies a status that is always cleared. A 1 identifies a status that is always set. A blank means the status does not change. The numbers 7 and 6 show that the flag contains the value of bit 7 or bit 6 of the byte tested by the instruction.

Table 3-4. A Summary of the 6502 Instruction Set

Type	Instruction	Object Code	Bytes	Clock Periods	Status						Operation Performed	
					S	V	D	I	Z	C		
I/O and Primary Memory Reference	LDA										Load Accumulator from memory.	
	addr	A5 pp	2	3	X				X		A←[addr]	Zero page direct
	addr,X	B5 pp	2	4	X				X		A←[addr+X]	Zero page indexed
	(addr,X)	A1 pp	2	6	X				X		A←[[addr+X]]	Pre-indexed indirect
	(addr),Y	B1 pp	2	5*	X				X		A←[[addr+1,addr]+Y]	Post-indexed indirect
	addr16	AD ppqq	3	4	X				X		A←[addr16]	Extended direct
	addr16,X or Y	11011x01 ppqq	3	4*	X				X		A←[addr16+X] or A←[addr16+Y]	Absolute indexed
	STA										Store Accumulator to memory.	
	addr	85 pp	2	3							[addr]←A	Zero page direct
	addr,X	95 pp	2	4							[addr+X]←A	Zero page indexed
	(addr,X)	81 pp	2	6							[[addr+X]]←A	Pre-indexed indirect
	(addr),Y	91 pp	2	6							[[addr+1,addr]+Y]←A	Post-indexed direct
	addr16	8D ppqq	3	4							[addr16]←A	Extended direct
	addr16,X or Y	10011x01 ppqq	3	5							[addr16+X]←A or [addr16+Y]←A	Absolute indexed
	LDX										Load Index Register X from memory. Index through Register Y only.	
	addr	A6 pp	2	3	X				X		X←[addr]	Zero page direct
	addr,Y	B6 pp	2	4	X				X		X←[addr+Y]	Zero page indexed
	addr16	AE ppqq	3	4	X				X		X←[addr16]	Extended direct
	addr16,Y	BE ppqq	3	4*	X				X		X←[addr16+Y]	Absolute indexed
	STX										Store Index Register X to memory. Index through Register Y only.	
	addr	86 pp	2	3							[addr]←X	Zero page direct
	addr,Y	96 pp	2	4							[addr+Y]←X	Zero page indexed
	addr16	8E ppqq	3	4							[addr16]←X	Extended direct
	LDY										Load Index Register Y from memory. Index through Register X only.	
	addr	A4 pp	2	3	X				X		Y←[addr]	Zero page direct
	addr,X	B4 pp	2	4	X				X		Y←[addr+X]	Zero page indexed
	addr16	AC ppqq	3	4	X				X		Y←[addr16]	Extended direct
	addr16,X	BC ppqq	3	4	X				X		Y←[addr16+X]	Absolute indexed

* Add one clock period if page boundary is crossed. In the object code, "x" designates the Index register: x = 0 for Register Y, x = 1 for Register X.

Table 3-4. A Summary of the 6502 Instruction Set (Continued)

Type	Instruction	Object Code	Bytes	Clock Periods	Status						Operation Performed	
					S	V	D	I	Z	C		
I/O and Primary Memory Reference (Continued)	STY										Store Index Register Y to memory. Index through Register X only.	
	addr	84 pp	2	3							[addr]←Y	Zero page direct
	addr,X	94 pp	2	4							[addr+X]←Y	Zero page indexed
	addr16	8C ppqq	3	4							[addr16]←Y	Extended direct
Secondary Memory Reference (Memory Operate)	ADC										Add contents of memory location, with carry, to those of Accumulator.	
	addr	65 pp	2	3	X	X			X	X	A←A+[addr]+C	Zero page direct
	addr,X	75 pp	2	4	X	X			X	X	A←A+[addr+X]+C	Zero page indexed
	(addr,X)	61 pp	2	6	X	X			X	X	A←A+[[addr+X]]+C	Pre-indexed indirect
	(addr),Y	71 pp	2	5*	X	X			X	X	A←A+[[addr+1, addr]+Y]+C	Post-indexed indirect
	addr16	6D ppqq	3	4	X	X			X	X	A←A+[addr16]+C	Extended direct
	addr16,X or Y	01111x01 ppqq	3	4*	X	X			X	X	A←A+[addr16+X]+C or A←A+[addr16+Y]+C (Zero flag is not valid in Decimal Mode).	Absolute indexed
	AND										AND contents of Accumulator with those of memory location.	
	addr	25 pp	2	3	X				X		A←A∧[addr]	Zero page direct
	addr,X	35 pp	2	4	X				X		A←A∧[addr+X]	Zero page indexed
	(addr,X)	21 pp	2	6	X				X		A←A∧[[addr+X]]	Pre-indexed indirect
	(addr),Y	31 pp	2	5*	X				X		A←A∧[[addr+1, addr]+Y]	Post-indexed indirect
	addr16	2D ppqq	3	4	X				X		A←A∧[addr16]	Extended direct
	addr16,X or Y	00111x01 ppqq	3	4*	X				X		A←A∧[addr16+X] or A←A∧[addr16+Y]	Absolute indexed
	BIT										AND contents of Accumulator with those of memory location. Only the status bits are affected.	
	addr	24 pp	2	3	7	6			X		A∧[addr]	Zero page direct
	addr16	2C ppqq	3	4	7	6			X		A∧[addr16]	Extended direct

* Add one clock period if page boundary is crossed. In the object code, "x" designates the Index register: x = 0 for Register Y, x = 1 for Register X.

Table 3-4. A Summary of the 6502 Instruction Set (Continued)

Type	Instruction	Object Code	Bytes	Clock Periods	Status						Operation Performed	
					S	V	D	I	Z	C		
Secondary Memory Reference (Memory Operate) (Continued)	CMP										Compare contents of Accumulator with those of memory location. Only the status bits are affected.	
	addr	C5 pp	2	3	X				X	X	A−[addr]	Zero page direct
	addr,X	D5 pp	2	4	X				X	X	A−[addr+X]	Zero page indexed
	(addr,X)	C1 pp	2	6	X				X	X	A−[[addr+X]]	Pre-indexed indirect
	(addr),Y	D1 pp	2	5*	X				X	X	A−[[addr+1, addr]+Y]	Post-indexed indirect
	addr16	CD ppqq	3	4	X				X	X	A−[addr16]	Extended direct
	addr16,X or Y	11011x01 ppqq	3	4*	X				X	X	A−[addr16+X] or A−[addr16+Y]	Absolute indexed
	EOR										Exclusive-OR contents of Accumulator with those of memory location.	
	addr	45 pp	2	3	X				X		A←A⊻[addr]	Zero page direct
	addr,X	55 pp	2	4	X				X		A←A⊻[addr+X]	Zero page indexed
	(addr,X)	41 pp	2	6	X				X		A←A⊻[[addr+X]]	Pre-indexed indirect
	(addr),Y	51 pp	2	5*	X				X		A←A⊻[[addr+1, addr]+Y]	Post-indexed indirect
	addr16	4D ppqq	3	4	X				X		A←A⊻[addr16]	Extended direct
	addr16,X or Y	01011x01 ppqq	3	4*	X				X		A←A⊻[addr16+X] or A←A⊻[addr16+Y]	Absolute indexed
	ORA										OR contents of Accumulator with those of memory location.	
	addr	05 pp	2	3	X				X		A←AV[addr]	Zero page direct
	addr,X	15 pp	2	4	X				X		A←AV[addr+X]	Zero page indexed
	(addr,X)	01 pp	2	6	X				X		A←AV[[addr+X]]	Pre-indexed indirect
	(addr),Y	11 pp	2	5*	X				X		A←AV[[addr+1, addr]+Y]	Post-indexed indirect
	addr16	0D ppqq	3	4	X				X		A←AV[addr16]	Extended direct
	addr16,X or Y	00011x01 ppqq	3	4*	X				X		A←AV[addr16+X] or A←AV[addr16+Y]	Absolute indexed

* Add one clock period if page boundary is crossed. In the object code, "x" designates the Index register: x = 0 for Register Y, x = 1 for Register X.

Table 3-4. A Summary of the 6502 Instruction Set (Continued)

Type	Instruction	Object Code	Bytes	Clock Periods	Status						Operation Performed
					S	V	D	I	Z	C	
Secondary Memory Reference (Memory Operate) (Continued)	SBC										Subtract contents of memory location, with borrow, from contents of Accumultor.
	addr	E5 pp	2	3	X	X			X	X	A ← A−[addr]−$\overline{C}$ Zero page direct
	addr,X	F5 pp	2	4	X	X			X	X	A ← A−[addr+X]−$\overline{C}$ Zero page indexed
	(addr,X)	E1 pp	2	6	X	X			X	X	A ← A−[[addr+X]]−$\overline{C}$ Pre-indexed indirect
	(addr),Y	F1 pp	2	5*	X	X			X	X	A ← A−[[addr+1,addr]+Y]−$\overline{C}$ Post-indexed indirect
	addr16	ED ppqq	3	4	X	X			X	X	A ← A−[addr16]−$\overline{C}$ Extended direct
	addr16,X or Y	11111x01 ppqq	3	4*	X	X			X	X	A ← A−[addr16+X]−$\overline{C}$ or A ← A−[addr16+Y]−$\overline{C}$ Absolute indexed (Note that Carry value is the complement of the borrow.)
	INC										Increment contents of memory location. Index through Register X only.
	addr	E6 pp	2	5	X				X		[addr]←[addr]+1 Zero page direct
	addr,X	F6 pp	2	6	X				X		[addr+X]←[addr+X]+1 Zero page indexed
	addr16	EE ppqq	3	6	X				X		[addr16]←[addr16]+1 Extended direct
	addr16,X	FE ppqq	3	7	X				X		[addr16+X]←[addr16+X]+1 Absolute indexed
	DEC										Decrement contents of memory location. Index through Register X only.
	addr	C6 pp	2	5	X				X		[addr]←[addr]−1 Zero page direct
	addr,X	D6 pp	2	6	X				X		[addr+X]←[addr+X]−1 Zero page indexed
	addr16	CE ppqq	3	6	X				X		[addr16]←[addr16]−1 Extended direct
	addr16,X	DE ppqq	3	7	X				X		[addr16+X]←[addr16+X]−1 Absolute indexed
	CPX										Compare contents of X register with those of memory location. Only the status flags are affected.
	addr	E4 pp	2	3	X				X	X	X−[addr] Zero page direct
	addr16	EC ppqq	3	4	X				X	X	X−[addr16] Extended direct
	CPY										Compare contents of Y register with those of memory location. Only the status flags are affected.
	addr	C4 pp	2	3	X				X	X	Y−[addr] Zero page direct
	addr16	CC ppqq	3	4	X				X	X	Y−[addr16] Extended direct

* Add one clock period if page boundary is crossed. In the object code, "x" designates the Index register: x = 0 for Register Y, x = 1 for Register X.

Table 3-4. A Summary of the 6502 Instruction Set (Continued)

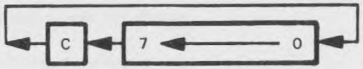
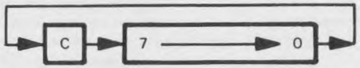
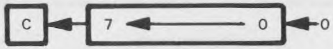
Type	Instruction	Object Code	Bytes	Clock Periods	Status						Operation Performed
					S	V	D	I	Z	C	
Secondary Memory Reference (Memory Operate) (Continued)	ROL										Rotate contents of memory location one bit left through Carry. Index through Register X only.
	addr	26 pp	2	5	X				X	X	[addr]
	addr,X	36 pp	2	6	X				X	X	[addr+X]
	addr16	2E ppqq	3	6	X				X	X	[addr16]
	addr16,X	3E ppqq	3	7	X				X	X	[addr16+X]
											
	ROR										Rotate contents of memory location one bit right, through Carry. Index through Register X only.
	addr	66 pp	2	5	X				X	X	[addr]
	addr,X	76 pp	2	6	X				X	X	[addr+X]
	addr16	6E pp	3	6	X				X	X	[addr16]
	addr16,X	7E ppqq	3	7	X				X	X	[addr16+X]
											
	ASL										Arithmetic shift left contents of memory location. Index through Register X only.
	addr	06 pp	2	5	X				X	X	[addr]
	addr,X	16 pp	2	6	X				X	X	[addr+X]
	addr16	0E ppqq	3	6	X				X	X	[addr16]
	addr16,X	1E ppqq	3	7	X				X	X	[addr16+X]
											

Table 3-4. A Summary of the 6502 Instruction Set (Continued)

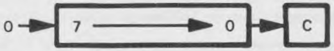
Type	Instruction	Object Code	Bytes	Clock Periods	Status						Operation Performed
					S	V	D	I	Z	C	
Secondary Memory Ref. (Memory Operate) (Cont.)	LSR										Logical shift right contents of memory location. Index through Register X only.
	addr	46 pp	2	5	0				X	X	[addr]
	addr,X	56 pp	2	6	0				X	X	[addr+X]
	addr16	4E ppqq	3	6	0				X	X	[addr16]
	addr16,X	5E ppqq	3	7	0				X	X	[addr16,X] 
Immediate	LDA data	A9 pp	2	2	X				X		Load Accumulator with immediate data. A←data
	LDX data	A2 pp	2	2	X				X		Load Index Register X with immediate data. X←data
	LDY data	A0 pp	2	2	X				X		Load Index Register Y with immediate data. Y←data

Table 3-4. A Summary of the 6502 Instruction Set (Continued)

Type	Instruction	Object Code	Bytes	Clock Periods	Status						Operation Performed
					S	V	D	I	Z	C	
Immediate Operate	ADC data	69 pp	2	2	X	X			X	X	Add immediate with Carry, to Accumulator. The Zero flag is not valid in Decimal Mode. A←A+data+C
	AND data	29 pp	2	2	X				X		AND immediate with Accumulator. A←AΛdata
	CMP data	C9 pp	2	2	X				X	X	Compare immediate with Accumulator. Only the status flags are affected. A−data
	EOR data	49 pp	2	2	X				X		Exclusive-OR immediate with Accumulator. A←A ⊻ data
	ORA data	09 pp	2	2	X				X		OR immediate with Accumulator. A←A V data
	SBC data	E9 pp	2	2	X	X			X	X	Subtract immediate, with borrow, from Accumulator. A←A−data−$\bar{C}$ (Note that Carry value is the complement of the borrow.)
	CPX data	E0 pp	2	2	X				X	X	Compare immediate with Index Register X. Only the status flags are affected. X−data
	CPY data	C0 pp	2	2	X				X	X	Compare immediate with Index Register Y. Only the status flags are affected. Y−data
Jump	JMP label (label)	4C ppqq 6C ppqq	3 3	3 5							Jump to new location, using extended or indirect addressing. PC←label or PC←[label]

Table 3-4. A Summary of the 6502 Instruction Set (Continued)

Type	Instruction	Object Code	Bytes	Clock Periods	Status						Operation Performed
					S	V	D	I	Z	C	
Branch on Condition											Note the following for all Branch-on-Condition instructions: If the condition is satisfied, the displacement is added to the Program Counter after the Program Counter has been incremented to point to the instruction following the Branch instruction.
	BCC disp	90 pp	2	2**							Branch relative if Carry flag is cleared. If C=0, then PC←PC+disp
	BCS disp	B0 pp	2	2**							Branch relative if Carry flag is set. If C=1, then PC←PC+disp
	BEQ disp	F0 pp	2	2**							Branch relative if result is equal to zero. If Z=1, then PC←PC+disp
	BMI disp	30 pp	2	2**							Branch relative if result is negative. If S=1, then PC←PC+disp
	BNE disp	D0 pp	2	2**							Branch relative if result is not zero. If Z=0, then PC←PC+disp
	BPL disp	10 pp	2	2**							Branch relative if result is positive. If S=0, then PC←PC+disp
	BVC disp	50 pp	2	2**							Branch relative if Overflow flag is cleared. If V = 0, then PC←PC+disp
	BVS disp	70 pp	2	2**							Branch relative if Overflow flag is set. If V=1, then PC←PC+disp

**Add one clock period if branch occurs to location in same page; add two clock periods if branch to another page occurs.

Table 3-4. A Summary of the 6502 Instruction Set (Continued)

Type	Instruction	Object Code	Bytes	Clock Periods	Status						Operation Performed
					S	V	D	I	Z	C	
Subroutine Call and Return	JSR label	20 ppqq	3	6							Jump to subroutine beginning at address given in bytes 2 and 3 of the instruction. Note that the stored Program Counter points to the last byte of the JSR instruction. [SP]←PC(HI) [SP−1]←PC(LO) SP←SP−2 PC←label
	RTS	60	1	6							Return from subroutine, incrementing Program Counter to point to the instruction after the JSR which called the routine. PC(LO)←[SP+1] PC(HI)←[SP+2] SP←SP+2 PC←PC+1
Register-Register Move	TAX	AA	1	2	X				X		Move Accumulator contents to Index Register X. X←A
	TXA	8A	1	2	X				X		Move contents of Index Register X to Accumulator. A←X
	TAY	A8	1	2	X				X		Move Accumulator contents to Index Register Y. Y←A
	TYA	98	1	2	X				X		Move contents of Index Register Y to Accumulator. A←Y
	TSX	BA	1	2	X				X		Move contents of Stack Pointer to Index Register X. X←SP
	TXS	9A	1	2							Move contents of Index Register X to Stack Pointer. SP←X

Table 3-4. A Summary of the 6502 Instruction Set (Continued)

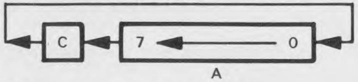
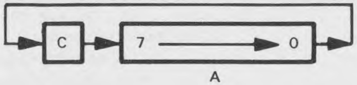
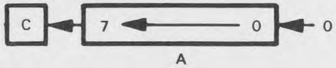
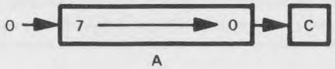
Type	Instruction	Object Code	Bytes	Clock Periods	Status						Operation Performed
					S	V	D	I	Z	C	
Register Operate	DEX	CA	1	2	X				X		Decrement contents of Index Register X. X←X−1
	DEY	88	1	2	X				X		Decrement contents of Index Register Y. Y←Y−1
	INX	E8	1	2	X				X		Increment contents of Index Register X. X←X+1
	INY	C8	1	2	X				X		Increment contents of Index Register Y. Y←Y+1
	ROL A	2A	1	2	X				X	X	Rotate contents of Accumulator left through Carry. 
	ROR A	6A	1	2	X				X	X	Rotate contents of Accumulator right, through Carry. 
	ASL A	0A	1	2	X				X	X	Arithmetic shift left contents of Accumulator. 
	LSR A	4A	1	2	0				X	X	Logical shift right contents of Accumulator. 

Table 3-4. A Summary of the 6502 Instruction Set (Continued)

Type	Instruction	Object Code	Bytes	Clock Periods	Status						Operation Performed
					S	V	D	I	Z	C	
Stack	PHA	48	1	3							Push Accumulator contents onto Stack. [SP]←A SP←SP−1
	PLA	68	1	4	X				X		Load Accumulator from top of Stack ("Pull"). A←[SP+1] SP←SP+1
	PHP	08	1	3							Push Status register contents onto Stack. [SP]←P SP←SP−1
	PLP	28	1	4	X	X	X	X	X	X	Load Status register from top of Stack ("Pull"). P←[SP+1] SP←SP+1
Interrupt	CLI	58	1	2				0			Enable interrupts by clearing interrupt disable bit of Status register. ı←0
	SEI	78	1	2				1			Disable interrupts I←1
	RTI	40	1	6	X	X	X	X	X	X	Return from interrupt; restore Status P←[SP+1] PC(LO)←[SP+2] PC(HI)←[SP+3] SP←SP+3 PC←PC+1
	BRK	00	1	7				1			Programmed interrupt. BRK cannot be disabled. The Program Counter is incremented twice before it is saved on the Stack. [SP]←PC(HI) [SP−1]←PC(LO) [SP−2]←P SP←SP−3 PC(HI)←[FFFF] PC(LO)←[FFFE] I←1 B←1

Table 3-4. A Summary of the 6502 Instruction Set (Continued)

Type	Instruction	Object Code	Bytes	Clock Periods	Status						Operation Performed
					S	V	D	I	Z	C	
Status	CLC	18	1	2						0	Clear Carry flag C←0
	SEC	38	1	2						1	Set Carry flag C←1
	CLD	D8	1	2			0				Clear Decimal Mode D←0
	SED	F8	1	2			1				Set Decimal Mode D←1
	CLV	B8	1	2		0					Clear Overflow flag V←0
	NOP	EA	1	2							No Operation

Table 3-5. 6502 Instruction Object Codes in Numerical Order

Object Code	Instruction	
00	BRK	
01 pp	ORA	(addr,X)
05 pp	ORA	addr
06 pp	ASL	addr
08	PHP	
09 pp	ORA	data
0A	ASL	A
0D ppqq	ORA	addr16
0E ppqq	ASL	addr16
10 pp	BPL	disp
11 pp	ORA	(addr),Y
15 pp	ORA	addr,X
16 pp	ASL	addr,X
18	CLC	
19 ppqq	ORA	addr16,Y
1D ppqq	ORA	addr16,X
1E ppqq	ASL	addr16,X
20 ppqq	JSR	label
21 pp	AND	(addr,X)
24 pp	BIT	addr
25 pp	AND	addr
26 pp	ROL	addr
28	PLP	
29 pp	AND	data
2A	ROL	A
2C ppqq	BIT	addr16
2D ppqq	AND	addr16
2E ppqq	ROL	addr16
30 pp	BMI	disp
31 pp	AND	(addr),Y
35 pp	AND	addr,X
36 pp	ROL	addr,X
38	SEC	
39 ppqq	AND	addr16,Y
3D ppqq	AND	addr16,X
3E ppqq	ROL	addr16,X
40	RTI	
41 pp	EOR	(addr,X)
45 pp	EOR	addr
46 pp	LSR	addr
48	PHA	
49 pp	EOR	data
4A	LSR	A
4C ppqq	JMP	label
4D ppqq	EOR	addr16
4E ppqq	LSR	addr16
50 pp	BVC	disp
51 pp	EOR	(addr),Y
55 pp	EOR	addr,X
56 pp	LSR	addr,X
58	CLI	
59 ppqq	EOR	addr16,Y
5D ppqq	EOR	addr16,X
5E ppqq	LSR	addr16,X
60	RTS	
61 pp	ADC	(addr,X)
65 pp	ADC	addr
66 pp	ROR	addr
68	PLA	
69 pp	ADC	data
6A	ROR	A
6C ppqq	JMP	(label)
6D ppqq	ADC	addr16
6E ppqq	ROR	addr16
70 pp	BVS	disp
71 pp	ADC	(addr),Y
75 pp	ADC	addr,X
76 pp	ROR	addr,X
78	SEI	
79 ppqq	ADC	addr16,Y
7D ppqq	ADC	addr16,X
7E ppqq	ROR	addr16,X
81 pp	STA	(addr,X)
84 pp	STY	addr
85 pp	STA	addr
86 pp	STX	addr
88	DEY	
8A	TXA	
8C ppqq	STY	addr16
8D ppqq	STA	addr16
8E ppqq	STX	addr16
90 pp	BCC	disp
91 pp	STA	(addr),Y
94 pp	STY	addr,X
95 pp	STA	addr,X
96 pp	STX	addr,Y
98	TYA	
99 ppqq	STA	addr16,Y
9A	TXS	
9D ppqq	STA	addr16,X
A0 pp	LDY	data
A1 pp	LDA	(addr,X)
A2 pp	LDX	data
A4 pp	LDY	addr
A5 pp	LDA	addr
A6 pp	LDX	addr
A8	TAY	
A9 pp	LDA	data
AA	TAX	
AC ppqq	LDY	addr16
AD ppqq	LDA	addr16
AE ppqq	LDX	addr16
B0 pp	BCS	disp
B1 pp	LDA	(addr),Y
B4 pp	LDY	addr,X
B5 pp	LDA	addr,X
B6 pp	LDX	addr,Y
B8	CLV	
B9 ppqq	LDA	addr16,Y
BA	TSX	
BC ppqq	LDY	addr16,X
BD ppqq	LDA	addr16,X
BE ppqq	LDX	addr16,Y
C0 pp	CPY	data
C1 pp	CMP	(addr,X)
C4 pp	CPY	addr

Table 3-5. 6502 Instruction Object Codes in Numerical Order (Continued)

Object Code	Instruction	
C5 pp	CMP	addr
C6 pp	DEC	addr
C8	INY	
C9 pp	CMP	data
CA	DEX	
CC ppqq	CPY	addr16
CD ppqq	CMP	addr16
CE ppqq	DEC	addr16
D0 pp	BNE	disp
D1 pp	CMP	(addr),Y
D5 pp	CMP	addr,X
D6 pp	DEC	addr,X
D8	CLD	
D9 ppqq	CMP	addr16,Y
DD ppqq	CMP	addr16,X
DE ppqq	DEC	addr16,X
E0 pp	CPX	data
E1 pp	SBC	(addr,X)
E4 pp	CPX	addr
E5 pp	SBC	addr
E6 pp	INC	addr
E8	INX	
E9 pp	SBC	data
EA	NOP	
EC ppqq	CPX	addr16
ED ppqq	SBC	addr16
EE ppqq	INC	addr16
F0 pp	BEQ	disp
F1 pp	SBC	(addr),Y
F5 pp	SBC	addr,X
F6 pp	INC	addr,X
F8	SED	
F9 ppqq	SBC	addr16,Y
FD ppqq	SBC	addr16,X
FE ppqq	INC	addr16,X

The following symbols are used in the object codes in Table 3-6.

Address-mode Selection:

aaa		
	000	pre-indexed indirect - (addr,X)
	001	direct - addr
	010	immediate - data
	011	extended direct - addr16
	100	post-indexed indirect - (addr),Y
	101	base page indexed - addr,X
	110	absolute indexed - addr16,Y
	111	absolute indexed - addr16,X
bb		
	00	direct - addr
	01	extended direct - addr16
	10	base page indexed - addr,X
	11	absolute indexed - addr16,X
bbb		
	001	direct - addr
	010	accumulator - A
	011	extended direct - addr16
	101	base page indexed - addr,X; addr,Y in STX
	111	absolute indexed - addr16,X; addr16,Y in STX
cc		
	00	immediate - data
	01	direct - addr
	11	extended direct - addr16
ddd		
	000	immediate - data
	001	direct - addr
	011	extended direct - addr16
	101	base page indexed - addr,Y in LDX; addr,X in LDY
	111	absolute indexed - addr16,Y in LDX; addr16,X in LDY
pp	the second byte of a two- or three-byte instruction	
qq	the third byte of a three-byte instruction	
x	one bit choosing the address mode:	
	0	direct - addr
	1	extended direct - addr16
y	one bit choosing the JMP address mode:	
	0	extended direct - label
	1	indirect - (label)

Table 3-6. Summary of 6502 Object Codes with 6800 Mnemonics

Mnemonic	Operand	Object Code	Bytes	Clock Periods	MC6800 Instruction
ADC		011aaa01			ADCA
	data	pp	2	2	data8
	addr	pp	2	3	addr8
	addr,X	pp	2	4*	index
	(addr,X)	pp	2	6	
	(addr),Y	pp	2	5*	
	addr16	ppqq	3	4	addr16
	addr16,X	ppqq	3	4*	
	addr16,Y	ppqq	3	4*	
AND		001aaa01			ANDA
	data	pp	2	2	data8
	addr	pp	2	3	addr8
	addr,X	pp	2	4	index
	(addr,X)	pp	2	6	
	(addr),Y	pp	2	5*	
	addr16	ppqq	3	4	addr16
	addr16,X	ppqq	3	4*	
	addr16,Y	ppqq	3	4*	
ASL	A	000bbb10	1	2	ASLA
	addr	pp	1	5	
	addr,X	pp	2	6	ASL index
	addr16	ppqq	3	6	ASL addr16
	addr16,X	ppqq	3	7	
BCC	disp	90 pp	2	2**	BCC disp
BCS	disp	B0 pp	2	2**	BCS disp
BEQ	disp	F0 pp	2	2**	BEQ disp
BIT		0010x100			BITA
	addr	pp	2	3	addr8
	addr16	ppqq	3	4	addr16
BMI	disp	30 pp	2	2**	BMI disp
BNE	disp	D0 pp	2	2**	BNE disp
BPL	disp	10 pp	2	2**	BPL disp
BRK		00	1	7	(SWI)
BVC	disp	50 pp	2	2**	BVC disp
BVS	disp	70 pp	2	2**	BVS disp
CLC		18	1	2	CLC
CLD		D8	1	2	
CLI		58	1	2	CLI
CLV		B8	1	2	CLV

*Add one clock period if page boundary is crossed.

**Add one clock period if branch occurs to location in same page; add two clock periods if branch to another page occurs.

Table 3-6. Summary of 6502 Object Codes with 6800 Mnemonics (Continued)

Mnemonic	Operand	Object Code	Bytes	Clock Periods	MC6800 Instruction
CMP		110aaa01			CMPA
	data	pp	2	2	data8
	addr	pp	2	3	addr8
	addr,X	pp	2	4	index
	(addr,X)	pp	2	6	
	(addr),Y	pp	2	5*	
	addr16	ppqq	3	4	addr16
	addr16,X	ppqq	3	4*	
	addr16,Y	ppqq	3	4*	
CPX		1110cc00			CPX
	data	pp	2	2	data8
	addr	pp	2	3	addr8
	addr16	ppqq	3	4	addr16
CPY		1100cc00			
	data	pp	2	2	
	addr	pp	2	3	
	addr16	ppqq	3	4	
DEC		110bb110			DEC
	addr	pp	2	5	
	addr,X	pp	2	6	index
	addr16	ppqq	3	6	addr16
	addr16,X	ppqq	3	7	
DEX		CA	1	2	DEX
DEY		88	1	2	
EOR		010aaa01			EORA
	data	pp	2	2	data8
	addr	pp	2	3	addr8
	addr,X	pp	2	4	index
	(addr,X)	pp	2	6	
	(addr),Y	pp	2	5*	
	addr16	ppqq	3	4	addr16
	addr16,X	ppqq	3	4*	
	addr16,Y	ppqq	3	4*	
INC		111bb110			INC
	addr	pp	2	5	
	addr,X	pp	2	6	index
	addr16	ppqq	3	6	addr16
	addr16,X	ppqq	3	7	
INX		E8	1	2	INX
INY		C8	1	2	
JMP		01y01100			JMP
	label	ppqq	3	3	addr16
	(label)	ppqq	3	5	
JSR	label	20 ppqq	3	6	JSR addr16

*Add one clock period if page boundary is crossed.

**Add one clock period if branch occurs to location in same page; add two clock periods if branch to another page occurs.

Table 3-6. Summary of 6502 Object Codes with 6800 Mnemonics (Continued)

Mnemonic	Operand	Object Code	Bytes	Clock Periods	MC6800 Instruction
LDA		101aaa01			LDAA
	data	pp	2	2	data8
	addr	pp	2	3	addr8
	addr,X	pp	2	4	index
	(addr,X)	pp	2	6	
	(addr),Y	pp	2	5*	
	addr16	ppqq	3	4	addr16
	addr16,X	ppqq	3	4*	
	addr16,Y	ppqq	3	4*	
LDX		101ddd10			LDX
	data	pp	2	2	(data8)
	addr	pp	2	3	addr8
	addr,Y	pp	2	4	(index)
	addr16	ppqq	3	4	addr16
	addr16,Y	ppqq	3	4*	
LDY		101ddd00			
	data	pp	2	2	
	addr	pp	2	3	
	addr,X	pp	2	4	
	addr16	ppqq	3	4	
	addr16,X	ppqq	3	4*	
LSR	A	010bbb10	1	2	LSRA
	addr	pp	2	5	
	addr,X	pp	2	6	LSR index
	addr16	ppqq	3	6	LSR addr16
	addr16,X	ppqq	3	7	
NOP		EA	1	2	NOP
ORA		000aaa01			ORAA
	data	pp	2	2	data8
	addr	pp	2	3	addr8
	addr,X	pp	2	4	index
	(addr,X)	pp	2	6	
	(addr),Y	pp	2	5*	
	addr16	ppqq	3	4	addr16
	addr16,X	ppqq	3	4*	
	addr16,Y	ppqq	3	4*	
PHA		48	1	3	PSHA
PHP		08	1	3	
PLA		68	1	4	PULA
PLP		28	1	4	
ROL	A	001bbb10	1	2	ROLA
	addr	pp	2	5	
	addr,X	pp	2	6	ROL index
	addr16	ppqq	3	6	ROL addr16
	addr16,X	ppqq	3	7	

*Add one clock period if page boundary is crossed.

**Add one clock period if branch occurs to location in same page; add two clock periods if branch to another page occurs.

Table 3-6. Summary of 6502 Object Codes with 6800 Mnemonics (Continued)

Mnemonic	Operand	Object Code	Bytes	Clock Periods	MC6800 Instruction
ROR	A	011bbb10	1	2	RORA
	addr	pp	2	5	
	addr,X	pp	2	6	ROR index
	addr16	ppqq	3	6	ROR addr16
	addr16,X	ppqq	3	7	
RTI		40	1	6	RTI
RTS		60	1	6	RTS
SBC		111aaa01			SBCA
	data	pp	2	2	data8
	addr	pp	2	3	addr8
	addr,X	pp	2	4	index
	(addr,X)	pp	2	6	
	(addr),Y	pp	2	5*	
	addr16	ppqq	3	4	addr16
	addr16,X	ppqq	3	4*	
	addr16,Y	ppqq	3	4*	
SEC		38	1	2	SEC
SED		F8	1	2	
SEI		78	1	2	SEI
STA		100aaa01			STAA
	addr	pp	2	3	addr8
	addr,X	pp	2	4	index
	(addr,X)	pp	2	6	
	(addr),Y	pp	2	6	
	addr16	ppqq	3	4	addr16
	addr16,X	ppqq	3	5	
	addr16,Y	ppqq	3	5	
STX		100bb110			STX
	addr	pp	2	3	addr8
	addr,Y	pp	2	4	(index)
	addr16	ppqq	3	4	addr16
STY		100bb100			
	addr	pp	2	3	
	addr,X	pp	2	4	
	addr16	ppqq	3	4	
TAX		AA	1	2	
TAY		A8	1	2	
TSX		BA	1	2	TSX
TXA		8A	1	2	
TXS		9A	1	2	TXS
TYA		98	1	2	

*Add one clock period if page boundary is crossed.

**Add one clock period if branch occurs to location in same page; add two clock periods if branch to another page occurs.