6502 Instruction Set

HI	LO-NIBBLE															
	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00	BRK impl	ORA X,ind	??? ---	??? ---	??? ---	ORA zpg	ASL zpg	??? ---	PHP impl	ORA #	ASL A	??? ---	??? ---	ORA abs	ASL abs	??? ---
10	BPL rel	ORA ind,Y	??? ---	??? ---	??? ---	ORA zpg,X	ASL zpg,X	??? ---	CLC impl	ORA abs,Y	??? ---	??? ---	??? ---	ORA abs,X	ASL abs,X	??? ---
20	JSR abs	AND X,ind	??? ---	??? ---	BIT zpg	AND zpg	ROL zpg	??? ---	PLP impl	AND #	ROL A	??? ---	BIT abs	AND abs	ROL abs	??? ---
30	BMI rel	AND ind,Y	??? ---	??? ---	??? ---	AND zpg,X	ROL zpg,X	??? ---	SEC impl	AND abs,Y	??? ---	??? ---	??? ---	AND abs,X	ROL abs,X	??? ---
40	RTI impl	EOR X,ind	??? ---	??? ---	??? ---	EOR zpg	LSR zpg	??? ---	PHA impl	EOR #	LSR A	??? ---	JMP abs	EOR abs	LSR abs	??? ---
50	BVC rel	EOR ind,Y	??? ---	??? ---	??? ---	EOR zpg,X	LSR zpg,X	??? ---	CLI impl	EOR abs,Y	??? ---	??? ---	??? ---	EOR abs,X	LSR abs,X	??? ---
60	RTS impl	ADC X,ind	??? ---	??? ---	??? ---	ADC zpg	ROR zpg	??? ---	PLA impl	ADC #	ROR A	??? ---	JMP ind	ADC abs	ROR abs	??? ---
70	BVS rel	ADC ind,Y	??? ---	??? ---	??? ---	ADC zpg,X	ROR zpg,X	??? ---	SEI impl	ADC abs,Y	??? ---	??? ---	??? ---	ADC abs,X	ROR abs,X	??? ---
80	??? ---	STA X,ind	??? ---	??? ---	STY zpg	STA zpg	STX zpg	??? ---	DEY impl	??? ---	TXA impl	??? ---	STY abs	STA abs	STX abs	??? ---
90	BCC rel	STA ind,Y	??? ---	??? ---	STY zpg,X	STA zpg,X	STX zpg,Y	??? ---	TYA impl	STA abs,Y	TXS impl	??? ---	??? ---	STA abs,X	??? ---	??? ---
A0	LDY #	LDA X,ind	LDX #	??? ---	LDY zpg	LDA zpg	LDX zpg	??? ---	TAY impl	LDA #	TAX impl	??? ---	LDY abs	LDA abs	LDX abs	??? ---
B0	BCS rel	LDA ind,Y	??? ---	??? ---	LDY zpg,X	LDA zpg,X	LDX zpg,Y	??? ---	CLV impl	LDA abs,Y	TSX impl	??? ---	LDY abs,X	LDA abs,X	LDX abs,Y	??? ---
C0	CPY #	CMP X,ind	??? ---	??? ---	CPY zpg	CMP zpg	DEC zpg	??? ---	INY impl	CMP #	DEX impl	??? ---	CPY abs	CMP abs	DEC abs	??? ---
D0	BNE rel	CMP ind,Y	??? ---	??? ---	??? ---	CMP zpg,X	DEC zpg,X	??? ---	CLD impl	CMP abs,Y	??? ---	??? ---	??? ---	CMP abs,X	DEC abs,X	??? ---
E0	CPX #	SBC X,ind	??? ---	??? ---	CPX zpg	SBC zpg	INC zpg	??? ---	INX impl	SBC #	NOP impl	??? ---	CPX abs	SBC abs	INC abs	??? ---
F0	BEQ rel	SBC ind,Y	??? ---	??? ---	??? ---	SBC zpg,X	INC zpg,X	??? ---	SED impl	SBC abs,Y	??? ---	??? ---	??? ---	SBC abs,X	INC abs,X	??? ---

Address Modes:

A	....	Accumulator	OPC A	*operand is AC*
abs	....	absolute	OPC $HHLL	*operand is address $HHLL*
abs,X	....	absolute, X-indexed	OPC $HHLL,X	*operand is address incremented by X with carry*
abs,Y	....	absolute, Y-indexed	OPC $HHLL,Y	*operand is address incremented by Y with carry*
#	....	immediate	OPC #$BB	*operand is byte (BB)*
impl	....	implied	OPC	*operand implied*
ind	....	indirect	OPC ($HHLL)	*operand is effective address; effective address is value of address*
X,ind	....	X-indexed, indirect	OPC ($BB,X)	*operand is effective zeropage address; effective address is byte (BB) incremented by X without carry*
ind,Y	....	indirect, Y-indexed	OPC ($LL),Y	*operand is effective address incremented by Y with carry; effective address is word at zeropage address*

rel	relative	OPC $BB	*branch target is PC + offset (BB), bit 7 signifies negative offset*
zpg	zeropage	OPC $LL	*operand is of address; address hibyte = zero ($00xx)*
zpg,X	zeropage, X-indexed	OPC $LL,X	*operand is address incremented by X; address hibyte = zero ($00xx); no page transition*
zpg,Y	zeropage, Y-indexed	OPC $LL,Y	*operand is address incremented by Y; address hibyte = zero ($00xx); no page transition*

Instructions by Name:

ADC	add with carry
AND	and (with accumulator)
ASL	arithmetic shift left
BCC	branch on carry clear
BCS	branch on carry set
BEQ	branch on equal (zero set)
BIT	bit test
BMI	branch on minus (negative set)
BNE	branch on not equal (zero clear)
BPL	branch on plus (negative clear)
BRK	interrupt
BVC	branch on overflow clear
BVS	branch on overflow set
CLC	clear carry
CLD	clear decimal
CLI	clear interrupt disable
CLV	clear overflow
CMP	compare (with accumulator)
CPX	compare with X
CPY	compare with Y
DEC	decrement
DEX	decrement X
DEY	decrement Y
EOR	exclusive or (with accumulator)
INC	increment
INX	increment X
INY	increment Y
JMP	jump
JSR	jump subroutine
LDA	load accumulator
LDY	load X
LDY	load Y
LSR	logical shift right

NOP	no operation
ORA	or with accumulator
PHA	push accumulator
PHP	push processor status (SR)
PLA	pull accumulator
PLP	pull processor status (SR)
ROL	rotate left
ROR	rotate right
RTI	return from interrupt
RTS	return from subroutine
SBC	subtract with carry
SEC	set carry
SED	set decimal
SEI	set interrupt disable
STA	store accumulator
STX	store X
STY	store Y
TAX	transfer accumulator to X
TAY	transfer accumulator to Y
TSX	transfer stack pointer to X
TXA	transfer X to accumulator
TXS	transfer X to stack pointer
TYA	transfer Y to accumulator

Registers:

PC	program counter	(16 bit)
AC	accumulator	(8 bit)
X	X register	(8 bit)
Y	Y register	(8 bit)
SR	status register [NV-BDIZC]	(8 bit)
SP	stack pointer	(8 bit)

SR Flags (bit 7 to bit 0):

N	Negative
V	Overflow
-	ignored
B	Break
D	Decimal (use BCD for arithmetics)

I .... Interrupt (IRQ disable)

Z .... Zero

C .... Carry

Processor Stack:
LIFO, top down, 8 bit range, 0x0100 - 0x01FF

Bytes, Words, Addressing:
8 bit bytes, 16 bit words in lobyte-hibyte representation (Little-Endian).
16 bit address range, operands follow instruction codes.

Vendor:
MOS Technology, 1975

APPENDIX A: 6502 Instructions in Detail

ADC Add Memory to Accumulator with Carry

```
     A + M + C -> A, C                N Z C I D V
                                      + + + - - +
```

addressing	assembler	opc	bytes	cyles
immidiate	ADC #oper	69	2	2
zeropage	ADC oper	65	2	3
zeropage,X	ADC oper,X	75	2	4
absolute	ADC oper	6D	3	4
absolute,X	ADC oper,X	7D	3	4*
absolute,Y	ADC oper,Y	79	3	4*
(indirect,X)	ADC (oper,X)	61	2	6
(indirect),Y	ADC (oper),Y	71	2	5*

AND AND Memory with Accumulator

```
     A AND M -> A                     N Z C I D V
                                      + + - - - -
```

addressing	assembler	opc	bytes	cyles
immidiate	AND #oper	29	2	2
zeropage	AND oper	25	2	3
zeropage,X	AND oper,X	35	2	4

absolute	AND oper	2D	3	4
absolute,X	AND oper,X	3D	3	4*
absolute,Y	AND oper,Y	39	3	4*
(indirect,X)	AND (oper,X)	21	2	6
(indirect),Y	AND (oper),Y	31	2	5*

ASL Shift Left One Bit (Memory or Accumulator)

```
C <- [76543210] <- 0             N Z C I D V
                                 + + + - - -
```

addressing	assembler	opc	bytes	cyles
accumulator	ASL A	0A	1	2
zeropage	ASL oper	06	2	5
zeropage,X	ASL oper,X	16	2	6
absolute	ASL oper	0E	3	6
absolute,X	ASL oper,X	1E	3	7

BCC Branch on Carry Clear

```
branch on C = 0                  N Z C I D V
                                 - - - - - -
```

addressing	assembler	opc	bytes	cyles
relative	BCC oper	90	2	2**

BCS Branch on Carry Set

```
branch on C = 1                  N Z C I D V
                                 - - - - - -
```

addressing	assembler	opc	bytes	cyles
relative	BCS oper	B0	2	2**

BEQ Branch on Result Zero

```
branch on Z = 1                  N Z C I D V
                                 - - - - - -
```

addressing	assembler	opc	bytes	cyles
relative	BEQ oper	F0	2	2**

BIT Test Bits in Memory with Accumulator

bits 7 and 6 of operand are transfered to bit 7 and 6 of SR (N,V);
the zeroflag is set to the result of operand AND accumulator.

```
A AND M, M7 -> N, M6 -> V        N Z C I D V
                                M7 + - - - M6
```

```
     addressing    assembler    opc  bytes  cyles
     --------------------------------------------
     zeropage      BIT oper      24    2     3
     absolute      BIT oper      2C    3     4


BMI  Branch on Result Minus

     branch on N = 1                  N Z C I D V
                                      - - - - - -

     addressing    assembler    opc  bytes  cyles
     --------------------------------------------
     relative      BMI oper      30    2     2**


BNE  Branch on Result not Zero

     branch on Z = 0                  N Z C I D V
                                      - - - - - -

     addressing    assembler    opc  bytes  cyles
     --------------------------------------------
     relative      BNE oper      D0    2     2**


BPL  Branch on Result Plus

     branch on N = 0                  N Z C I D V
                                      - - - - - -

     addressing    assembler    opc  bytes  cyles
     --------------------------------------------
     relative      BPL oper      10    2     2**


BRK  Force Break

     interrupt,                       N Z C I D V
     push PC+2, push SR               - - - 1 - -

     addressing    assembler    opc  bytes  cyles
     --------------------------------------------
     implied       BRK           00    1     7


BVC  Branch on Overflow Clear

     branch on V = 0                  N Z C I D V
                                      - - - - - -

     addressing    assembler    opc  bytes  cyles
     --------------------------------------------
     relative      BVC oper      50    2     2**


BVS  Branch on Overflow Set

     branch on V = 1                  N Z C I D V
```

-	-	-	-	-	-

addressing	assembler	opc	bytes	cyles
relative	BVC oper	70	2	2**

CLC Clear Carry Flag

0 -> C

N	Z	C	I	D	V
-	-	0	-	-	-

addressing	assembler	opc	bytes	cyles
implied	CLC	18	1	2

CLD Clear Decimal Mode

0 -> D

N	Z	C	I	D	V
-	-	-	-	0	-

addressing	assembler	opc	bytes	cyles
implied	CLD	D8	1	2

CLI Clear Interrupt Disable Bit

0 -> I

N	Z	C	I	D	V
-	-	-	0	-	-

addressing	assembler	opc	bytes	cyles
implied	CLI	58	1	2

CLV Clear Overflow Flag

0 -> V

N	Z	C	I	D	V
-	-	-	-	-	0

addressing	assembler	opc	bytes	cyles
implied	CLV	B8	1	2

CMP Compare Memory with Accumulator

A - M

N	Z	C	I	D	V
+	+	+	-	-	-

addressing	assembler	opc	bytes	cyles
immidiate	CMP #oper	C9	2	2
zeropage	CMP oper	C5	2	3
zeropage,X	CMP oper,X	D5	2	4
absolute	CMP oper	CD	3	4
absolute,X	CMP oper,X	DD	3	4*

absolute,Y	CMP oper,Y	D9	3	4*
(indirect,X)	CMP (oper,X)	C1	2	6
(indirect),Y	CMP (oper),Y	D1	2	5*

CPX Compare Memory and Index X

X - M

N	Z	C	I	D	V
+	+	+	-	-	-

addressing	assembler	opc	bytes	cyles
immidiate	CPX #oper	E0	2	2
zeropage	CPX oper	E4	2	3
absolute	CPX oper	EC	3	4

CPY Compare Memory and Index Y

Y - M

N	Z	C	I	D	V
+	+	+	-	-	-

addressing	assembler	opc	bytes	cyles
immidiate	CPY #oper	C0	2	2
zeropage	CPY oper	C4	2	3
absolute	CPY oper	CC	3	4

DEC Decrement Memory by One

M - 1 -> M

N	Z	C	I	D	V
+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
zeropage	DEC oper	C6	2	5
zeropage,X	DEC oper,X	D6	2	6
absolute	DEC oper	CE	3	3
absolute,X	DEC oper,X	DE	3	7

DEX Decrement Index X by One

X - 1 -> X

N	Z	C	I	D	V
+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
implied	DEC	CA	1	2

DEY Decrement Index Y by One

Y - 1 -> Y

N	Z	C	I	D	V
+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles

implied	DEC	88	1	2

EOR Exclusive-OR Memory with Accumulator

A EOR M -> A	N	Z	C	I	D	V
	+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
immidiate	EOR #oper	49	2	2
zeropage	EOR oper	45	2	3
zeropage,X	EOR oper,X	55	2	4
absolute	EOR oper	4D	3	4
absolute,X	EOR oper,X	5D	3	4*
absolute,Y	EOR oper,Y	59	3	4*
(indirect,X)	EOR (oper,X)	41	2	6
(indirect),Y	EOR (oper),Y	51	2	5*

INC Increment Memory by One

M + 1 -> M	N	Z	C	I	D	V
	+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
zeropage	INC oper	E6	2	5
zeropage,X	INC oper,X	F6	2	6
absolute	INC oper	EE	3	6
absolute,X	INC oper,X	FE	3	7

INX Increment Index X by One

X + 1 -> X	N	Z	C	I	D	V
	+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
implied	INX	E8	1	2

INY Increment Index Y by One

Y + 1 -> Y	N	Z	C	I	D	V
	+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
implied	INY	C8	1	2

JMP Jump to New Location

(PC+1) -> PCL (PC+2) -> PCH	N	Z	C	I	D	V
	-	-	-	-	-	-

addressing	assembler	opc	bytes	cyles

absolute	JMP oper	4C	3	3
indirect	JMP (oper)	6C	3	5

JSR Jump to New Location Saving Return Address

push (PC+2),
(PC+1) -> PCL
(PC+2) -> PCH

N	Z	C	I	D	V
-	-	-	-	-	-

addressing	assembler	opc	bytes	cyles
absolute	JSR oper	20	3	6

LDA Load Accumulator with Memory

M -> A

N	Z	C	I	D	V
+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
immidiate	LDA #oper	A9	2	2
zeropage	LDA oper	A5	2	3
zeropage,X	LDA oper,X	B5	2	4
absolute	LDA oper	AD	3	4
absolute,X	LDA oper,X	BD	3	4*
absolute,Y	LDA oper,Y	B9	3	4*
(indirect,X)	LDA (oper,X)	A1	2	6
(indirect),Y	LDA (oper),Y	B1	2	5*

LDX Load Index X with Memory

M -> X

N	Z	C	I	D	V
+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
immidiate	LDX #oper	A2	2	2
zeropage	LDX oper	A6	2	3
zeropage,Y	LDX oper,Y	B6	2	4
absolute	LDX oper	AE	3	4
absolute,Y	LDX oper,Y	BE	3	4*

LDY Load Index Y with Memory

M -> Y

N	Z	C	I	D	V
+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
immidiate	LDY #oper	A0	2	2
zeropage	LDY oper	A4	2	3
zeropage,X	LDY oper,X	B4	2	4
absolute	LDY oper	AC	3	4
absolute,X	LDY oper,X	BC	3	4*

LSR Shift One Bit Right (Memory or Accumulator)

0 -> [76543210] -> C

N	Z	C	I	D	V
-	+	+	-	-	-

addressing	assembler	opc	bytes	cyles
accumulator	LSR A	4A	1	2
zeropage	LSR oper	46	2	5
zeropage,X	LSR oper,X	56	2	6
absolute	LSR oper	4E	3	6
absolute,X	LSR oper,X	5E	3	7

NOP No Operation

N	Z	C	I	D	V
-	-	-	-	-	-

addressing	assembler	opc	bytes	cyles
implied	NOP	EA	1	2

ORA OR Memory with Accumulator

A OR M -> A

N	Z	C	I	D	V
+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
immidiate	ORA #oper	09	2	2
zeropage	ORA oper	05	2	3
zeropage,X	ORA oper,X	15	2	4
absolute	ORA oper	0D	3	4
absolute,X	ORA oper,X	1D	3	4*
absolute,Y	ORA oper,Y	19	3	4*
(indirect,X)	ORA (oper,X)	01	2	6
(indirect),Y	ORA (oper),Y	11	2	5*

PHA Push Accumulator on Stack

push A

N	Z	C	I	D	V
-	-	-	-	-	-

addressing	assembler	opc	bytes	cyles
implied	PHA	48	1	3

PHP Push Processor Status on Stack

push SR

N	Z	C	I	D	V
-	-	-	-	-	-

addressing	assembler	opc	bytes	cyles

implied	PHP	08	1	3

PLA Pull Accumulator from Stack

```
pull A                          N Z C I D V
                                + + - - - -
```

addressing	assembler	opc	bytes	cyles
implied	PLA	68	1	4

PLP Pull Processor Status from Stack

```
pull SR                         N Z C I D V
                                from stack
```

addressing	assembler	opc	bytes	cyles
implied	PHP	28	1	4

ROL Rotate One Bit Left (Memory or Accumulator)

```
C <- [76543210] <- C            N Z C I D V
                                + + + - - -
```

addressing	assembler	opc	bytes	cyles
accumulator	ROL A	2A	1	2
zeropage	ROL oper	26	2	5
zeropage,X	ROL oper,X	36	2	6
absolute	ROL oper	2E	3	6
absolute,X	ROL oper,X	3E	3	7

ROR Rotate One Bit Right (Memory or Accumulator)

```
C -> [76543210] -> C            N Z C I D V
                                + + + - - -
```

addressing	assembler	opc	bytes	cyles
accumulator	ROR A	6A	1	2
zeropage	ROR oper	66	2	5
zeropage,X	ROR oper,X	76	2	6
absolute	ROR oper	6E	3	6
absolute,X	ROR oper,X	7E	3	7

RTI Return from Interrupt

```
pull SR, pull PC                N Z C I D V
                                from stack
```

addressing	assembler	opc	bytes	cyles

implied	RTI	40	1	6

RTS Return from Subroutine

pull PC, PC+1 -> PC

N	Z	C	I	D	V
-	-	-	-	-	-

addressing	assembler	opc	bytes	cyles
implied	RTS	60	1	6

SBC Subtract Memory from Accumulator with Borrow

A - M - C -> A

N	Z	C	I	D	V
+	+	+	-	-	+

addressing	assembler	opc	bytes	cyles
immidiate	SBC #oper	E9	2	2
zeropage	SBC oper	E5	2	3
zeropage,X	SBC oper,X	F5	2	4
absolute	SBC oper	ED	3	4
absolute,X	SBC oper,X	FD	3	4*
absolute,Y	SBC oper,Y	F9	3	4*
(indirect,X)	SBC (oper,X)	E1	2	6
(indirect),Y	SBC (oper),Y	F1	2	5*

SEC Set Carry Flag

1 -> C

N	Z	C	I	D	V
-	-	1	-	-	-

addressing	assembler	opc	bytes	cyles
implied	SEC	38	1	2

SED Set Decimal Flag

1 -> D

N	Z	C	I	D	V
-	-	-	-	1	-

addressing	assembler	opc	bytes	cyles
implied	SED	F8	1	2

SEI Set Interrupt Disable Status

1 -> I

N	Z	C	I	D	V
-	-	-	1	-	-

addressing	assembler	opc	bytes	cyles
implied	SEI	78	1	2

STA Store Accumulator in Memory

A -> M

N	Z	C	I	D	V
-	-	-	-	-	-

addressing	assembler	opc	bytes	cyles
zeropage	STA oper	85	2	3
zeropage,X	STA oper,X	95	2	4
absolute	STA oper	8D	3	4
absolute,X	STA oper,X	9D	3	5
absolute,Y	STA oper,Y	99	3	5
(indirect,X)	STA (oper,X)	81	2	6
(indirect),Y	STA (oper),Y	91	2	6

STX Store Index X in Memory

X -> M

N	Z	C	I	D	V
-	-	-	-	-	-

addressing	assembler	opc	bytes	cyles
zeropage	STX oper	86	2	3
zeropage,Y	STX oper,Y	96	2	4
absolute	STX oper	8E	3	4

STY Sore Index Y in Memory

Y -> M

N	Z	C	I	D	V
-	-	-	-	-	-

addressing	assembler	opc	bytes	cyles
zeropage	STY oper	84	2	3
zeropage,X	STY oper,X	94	2	4
absolute	STY oper	8C	3	4

TAX Transfer Accumulator to Index X

A -> X

N	Z	C	I	D	V
+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
implied	TAX	AA	1	2

TAY Transfer Accumulator to Index Y

A -> Y

N	Z	C	I	D	V
+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
implied	TAY	A8	1	2

TSX Transfer Stack Pointer to Index X

SP -> X

N	Z	C	I	D	V
+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
implied	TSX	BA	1	2

TXA Transfer Index X to Accumulator

X -> A

N	Z	C	I	D	V
+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
implied	TXA	8A	1	2

TXS Transfer Index X to Stack Register

X -> SP

N	Z	C	I	D	V
+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
implied	TXS	9A	1	2

TYA Transfer Index Y to Accumulator

Y -> A

N	Z	C	I	D	V
+	+	-	-	-	-

addressing	assembler	opc	bytes	cyles
implied	TYA	98	1	2

* add 1 to cycles if page boundery is crossed

** add 1 to cycles if branch occurs on same page
add 2 to cycles if branch occurs to different page

Legend to Flags:

+	....	modified
-	....	not modified
1	....	set
0	....	cleared
M6	....	memory bit 6
M7	....	memory bit 7

Note on assembler syntax:
Most assemblers employ "OPC *oper" for forced zeropage addressing.

APENDIX B: The 65xx-Family:

Type	Features, Comments
6502	NMOS, 16 bit address bus, 8 bit data bus
6502A	accelerated version of 6502
6502C	accelerated version of 6502, CMOS
65C02	16 bit version, additional instructions and address modes
6503, 6505, 6506	12 bit address bus [4 KiB]
6504	13 bit address bus [8 KiB]
6507	13 bit address bus [8 KiB], no interrupts
6509	20 bit address bus [1 MiB] by bankswitching
6510	as 6502 with additional 6 bit I/O-port
6511	integrated micro controler with I/O-port, serial interface, and RAM (Rockwell)
65F11	as 6511, integrated FORTH interpreter
7501	as 6502, HMOS
8500	as 6510, CMOS
8502	as 6510 with switchable 2 MHz option, 7 bit I/O-port
65816 (65C816)	16 bit registers and ALU, 24 bit address bus [16 MiB], up to 24 MHz (Western Design Center)
65802 (65C802)	as 65816, pin compatible to 6502, 64 KiB address bus, up to 16 MHz

Disclaimer:
Errors excepted. The information is provided for free and AS IS, therefore without any warranty;
without even the implied warranty of merchantability or fitness for a particular purpose.

See also:
>> Virtual 6502 (6502/6510 emulator)
>> 6502 Assembler
>> 6502 Disassembler

Presented by virtual 6502, e-tradion.net.