4510 INSTRUCTION SET

Instruction Timing

Note that the number of cycles depends on the speed setting of the processor: Some instructions take more or fewer cycles when the processor is running at full-speed, or a C65 compatibility 3.5MHz speed, or at C64 compatibility 1MHz/2MHz speed. More detailed information on this is listed under each each instruction's information, but the high-level view is:

- When the processor is running at 1MHz, all instructions take at least two cycles, and dummy cycles are re-inserted into Read-Modify-Write instructions, so that all instructions take exactly the same number of cycles as on a 6502.
- The Read-Modify-Write instructions and all instructions that read a value from memory all require an extra cycle when operating at full speed, to allow signals to propagate within the processor.
- The Read-Modify-Write instructions require an additional cycle if the operand is $D019, as the dummy write is performed in this case. This is to improve compatibility with C64 software that frequently uses this "bug" of the 6502 to more rapidly acknowledge VIC-II interrupts.
- Page-crossing and branch-taking penalties do not apply when the processor is running at full speed.
- Many instructions require fewer cycles when the processor is running at full speed, as generally most non-bus cycles are removed. For example, Pushing and Pulling values to and from the stack requires only 2 cycles, instead of the 4 that that the 6502 requires for these instructions.

Opcode Table

The coloured cells indicate an extended 45GS02 Opcode. A Q pseudo register opcode is marked blue, a base-page indirect Z indexed opcode that can use 32-bit pointers is cyan.

	$x0	$x1	$x2
$0x	size *cyc* OPC mode	size *cyc* QOP Q mode	size *cyc* FARQ Q IbpZ

The letters attached to the cycle count have the following meaning:

	Meaning
b	Add one cycle if branch is taken. Add one more cycle if branch taken crosses a page boundary.
d	Subtract one cycle when CPU is at 3.5MHz.
i	Add one cycle if clock speed is at 40 MHz.
m	Subtract non-bus cycles when at 40MHz.
p	Add one cycle if indexing crosses a page boundary.
r	Add one cycle if clock speed is at 40 MHz.
s	Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

Opcode Table 4510/45GS02

	$x0	$x1	$x2	$x3	$x4	$x5	$x6	$x7
$0x	1 BRK imp *7*	2 ORA bpiX *6rp*	1 CLE imp *1*	1 SEE imp *1*	2 TSB bp *3r*	2 ORA Q bp *3r*	2 ASL Q bp *4r*	2 RMB0 bp *4rb*
$1x	2 BPL rel *2b*	2 ORA ibpY *5rp*	2 ORA Q ibpZ *5rp*	3 BPL relfar *3b*	2 TRB bp *5r*	2 ORA bpX *3r*	2 ASL Q bpX *4r*	2 RMB1 bp *4rb*
$2x	3 JSR abs *5s*	2 AND bpiX *5rp*	3 JSR ind *5r*	3 JSR indX *5rp*	2 BIT Q bp *3r*	2 AND Q bp *3r*	2 ROL Q bp *4r*	2 RMB2 bp *4r*
$3x	2 BMI rel *2r*	2 AND ibpY *5rp*	2 AND Q ibpZ *5rp*	3 BMI relfar *3b*	2 BIT bpX *3rp*	2 AND bpX *4rp*	2 ROL Q bpX *5rp*	2 RMB3 bp *4r*
$4x	1 RTI imp *6m*	2 EOR bpiX *5r*	1 NEG acc *1*	1 ASR Q acc *1*	2 ASR Q bp *4r*	2 EOR Q bp *3r*	2 LSR Q bp *4r*	2 RMB4 bp *4r*
$5x	2 BVC rel *2b*	2 EOR ibpY *5rp*	2 EOR Q ibpZ *5rp*	3 BVC relfar *3b*	2 ASR Q bpX *5rp*	2 EOR bpX *3p*	2 LSR Q bpX *3rp*	2 RMB5 bp *4r*
$6x	1 RTS imp *6m*	2 ADC bpiX *5r*	2 RTS imm *4*	3 BSR relfar *3b*	2 STZ bp *3*	2 ADC Q bp *3r*	2 ROR Q bp *5r*	2 RMB6 bp *5r*
$7x	2 BVS rel *2b*	2 ADC ibpY *5rp*	2 ADC Q ibpZ *5rp*	3 BVS relfar *3b*	2 STZ bpX *3*	2 ADC bpX *3r*	2 ROR Q bpX *5rmdp*	2 RMB7 bp *4r*
$8x	2 BRA rel *2b*	2 STA bpiX *5p*	2 STA ispY *6p*	3 BRA relfar *3b*	2 STY bp *3*	2 STA Q bp *3*	2 STX bp *3*	2 SMB0 bp *4r*
$9x	2 BCC rel *2b*	2 STA ibpY *5p*	2 STA Q ibpZ *5p*	3 BCC relfar *3b*	2 STY bpX *3p*	2 STA bpX *3p*	2 STX bpY *3p*	2 SMB1 bp *4r*
$Ax	2 LDY imm *2*	2 LDA bpiX *5rp*	2 LDX imm *2*	2 LDZ imm *2*	2 LDY bp *3r*	2 LDA Q bp *3r*	2 LDX bp *3r*	2 SMB2 bp *4r*
$Bx	2 BCS rel *2b*	2 LDA ibpY *5rp*	2 LDA Q ibpZ *5rp*	3 BCS relfar *3b*	2 LDY bpX *3rp*	2 LDA bpX *3rp*	2 LDX bpY *5rp*	2 SMB3 bp *4r*
$Cx	2 CPY imm *2*	2 CMP bpiX *5rp*	2 CPZ imm *2*	2 DEW bp *7mdr*	2 CPY bp *3r*	2 CMP Q bp *3r*	2 DEC Q bp *5mdr*	2 SMB4 bp *4r*
$Dx	2 BNE rel *2b*	2 CMP ibpY *5rp*	2 CMP Q ibpZ *5rp*	3 BNE relfar *3b*	2 CPZ bp *3r*	2 CMP bpX *3rp*	2 DEC Q bpX *5mdrp*	2 SMB5 bp *4r*
$Ex	2 CPX imm *2*	2 SBC bpiX *3pm*	2 LDA ispY *6rmp*	2 INW bp *7mdr*	2 CPX bp *3r*	2 SBC Q bp *3r*	2 INC Q bp *5mdr*	2 SMB6 bp *4r*
$Fx	2 BEQ rel *2b*	2 SBC ibpY *3rp*	2 SBC Q ibpZ *5rp*	3 BEQ relfar *3b*	3 PHW im16 *5m*	2 SBC bpX *3rp*	2 INC Q bpX *5dmrp*	2 SMB7 bp *4r*

Opcode Table 4510/45GS02

$x8	$x9	$xA	$xB	$xC	$xD	$xE	$xF	
1 *2* PHP imp	2 *2* ORA imm	1 *1* ASL Q acc	1 *1* TSY imp	3 *5r* TSB abs	3 *4r* ORA Q abs	3 *5r* ASL Q abs	3 *0rb* BBR0 bpr8	$0x
1 *1* CLC imp	3 *4r* ORA absY	1 *1* INC Q acc	1 *1* INZ imp	3 *4r* TRB abs	3 *4rp* ORA absX	3 *5rp* ASL Q absX	3 *5b* BBR1 bpr8	$1x
1 *4m* PLP imp	2 *2* AND imm	1 *1* ROL Q acc	1 *1* TYS imp	3 *4r* BIT Q abs	3 *4r* AND Q abs	3 *5r* ROL Q abs	3 *5b* BBR2 bpr8	$2x
1 *1* SEC imp	3 *4r* AND absY	1 *1* DEC Q acc	1 *1* DEZ imp	3 *4rp* BIT absX	3 *4rp* AND absX	3 *5rp* ROL Q absX	3 *4b* BBR3 bpr8	$3x
1 *2* PHA imp	2 *2* EOR imm	1 *1* LSR Q acc	1 *1* TAZ imp	3 *3* JMP abs	3 *4r* EOR Q abs	3 *5r* LSR Q abs	3 *4rb* BBR4 bpr8	$4x
1 *1* CLI imp	3 *4rp* EOR absY	1 *2* PHY imp	1 *1* TAB imp	1 *1* MAP imp	3 *4rp* EOR absX	3 *5rp* LSR Q absX	3 *4rb* BBR5 bpr8	$5x
1 *4m* PLA imp	2 *2* ADC imm	1 *1* ROR Q acc	1 *1* TZA imp	3 *5r* JMP ind	3 *4r* ADC Q abs	3 *6r* ROR Q abs	3 *4rb* BBR6 bpr8	$6x
1 *1s* SEI imp	3 *4r* ADC absY	1 *4m* PLY imp	1 *1* TBA imp	3 *6mp* JMP indX	3 *4r* ADC absX	3 *5rmdp* ROR Q absX	3 *4br* BBR7 bpr8	$7x
1 *1s* DEY imp	2 *2* BIT imm	1 *1* TXA imp	3 *4p* STY absX	3 *4* STY abs	3 *4* STA Q abs	3 *4* STX abs	3 *4br* BBS0 bpr8	$8x
1 *1* TYA imp	3 *4p* STA absY	1 *1* TXS imp	3 *4p* STX absY	3 *4* STZ abs	3 *4p* STA absX	3 *4p* STZ absX	3 *4br* BBS1 bpr8	$9x
1 *1* TAY imp	2 *2* LDA imm	1 *1* TAX imp	3 *4r* LDZ abs	3 *4r* LDY abs	3 *4r* LDA Q abs	3 *4r* LDX abs	3 *4br* BBS2 bpr8	$Ax
1 *1* CLV imp	3 *4rp* LDA absY	1 *1* TSX imp	3 *4rp* LDZ absX	3 *4rp* LDY absX	3 *4rp* LDA absX	3 *4rp* LDX absY	3 *4br* BBS3 bpr8	$Bx
1 *1s* INY imp	2 *2* CMP imm	1 *1s* DEX imp	3 *7rmd* ASW abs	3 *4r* CPY abs	3 *4r* CMP Q abs	3 *6mdr* DEC Q abs	3 *4br* BBS4 bpr8	$Cx
1 *1* CLD imp	3 *4rp* CMP absY	1 *3m* PHX imp	1 *3m* PHZ imp	3 *4r* CPZ abs	3 *4rp* CMP absX	3 *6mdrp* DEC Q absX	3 *4br* BBS5 bpr8	$Dx
1 *1s* INX imp	2 *2* SBC imm	1 *1* EOM imp	3 *5rmd* ROW abs	3 *4r* CPX abs	3 *4r* SBC Q abs	3 *6dmr* INC Q abs	3 *4br* BBS6 bpr8	$Ex
1 *1s* SED imp	3 *4rp* SBC absY	1 *4m* PLX imp	1 *4m* PLZ imp	3 *7m* PHW abs	3 *4rp* SBC absX	3 *6drp* INC Q absX	3 *4br* BBS7 bpr8	$Fx

ADC

This instruction adds the argument and the Carry Flag to the contents of the Accumulator Register. If the D flag is set, then the addition is performed using Binary Coded Decimal.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The V flag will be set if the result has a different sign to both of the arguments, otherwise it will be cleared. If the flag is set, this indicates that a signed overflow has occurred.
- The C flag will be set if the unsigned result is >255, or >99 if the D flag is set.

ADC : Add with carry **4510**

$A \leftarrow A + M + C$

N	Z	I	C	D	V	E
+	+	·	+	·	+	·

Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**	
(indirect,X)	ADC ($nn,X)	61	2	5	*r*
base-page	ADC $nn	65	2	3	*r*
immediate 8bit	ADC #$nn	69	2	2	
absolute	ADC $nnnn	6D	3	4	*r*
(indirect),Y	ADC ($nn),Y	71	2	5	*pr*
(indirect),Z	ADC ($nn),Z	72	2	5	*pr*
base-page,X	ADC $nn,X	75	2	3	*r*
absolute,Y	ADC $nnnn,Y	79	3	4	*r*
absolute,X	ADC $nnnn,X	7D	3	4	*r*

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

AND

This instruction performs a binary AND operation of the argument with the accumulator, and stores the result in the accumulator. Only bits that were already set in the accumulator, and that are set in the argument will be set in the accumulator on completion.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

AND : Binary AND **4510**

A ← A *AND* M

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
(indirect,X)	AND ($nn,X)	21	2	5	*pr*
base-page	AND $nn	25	2	3	*r*
immediate 8bit	AND #$nn	29	2	2	
absolute	AND $nnnn	2D	3	4	*r*
(indirect),Y	AND ($nn),Y	31	2	5	*pr*
(indirect),Z	AND ($nn),Z	32	2	5	*pr*
base-page,X	AND $nn,X	35	2	4	*pr*
absolute,Y	AND $nnnn,Y	39	3	4	*r*
absolute,X	AND $nnnn,X	3D	3	4	*pr*

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

ASL

This instruction shifts either the Accumulator or contents of the provided memory location one bit left. Bit 0 will be set to zero, and the bit 7 will be shifted out into the Carry Flag

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The C flag will be set if bit 7 of the value was set, prior to being shifted.

ASL : Arithmetic Shift Left Memory or Accumulator					**4510**
A ← A≪1 or M ← M≪1					
			N Z I C D V E		
			+ + · + · · ·		
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**	
base-page	ASL $nn	06	2	4	^{r}
accumulator	ASL A	0A	1	1	^{s}
absolute	ASL $nnnn	0E	3	5	^{r}
base-page,X	ASL $nn,X	16	2	4	^{r}
absolute,X	ASL $nnnn,X	1E	3	5	^{pr}

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.
s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

ASR

This instruction shifts either the Accumulator or contents of the provided memory location one bit right. Bit 7 is considered to be a sign bit, and is preserved. The contents of bit 0 will be shifted out into the Carry Flag

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The C flag will be set if bit 0 of the value was set, prior to being shifted.

ASR : Arithmetic Shift Right					**4510**
A ← A>>1 *or* M ← M>>1					
			N Z I C D V E		
			+ + · + · · ·		
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**	
accumulator	ASR A	43	1	1	^{s}
base-page	ASR $nn	44	2	4	^{r}
base-page,X	ASR $nn,X	54	2	5	^{pr}

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.
s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

ASW

This instruction shifts a 16-bit value in memory left one bit.

For example, if location $1234 contained $87 and location $1235 contained $A9, ASW $1234 would result in location $1234 containing $0E and location $1235 containing $53, and the Carry Flag being set.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The C flag will be set if bit 7 of the upper byte was set, prior to being shifted, otherwise it will be cleared.

ASW : Arithmetic Shift Word Left **4510**

$M \leftarrow M \ll 1$

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
absolute	ASW $nnnn	CB	3	7 ^{dmr}

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
r Add one cycle if clock speed is at 40 MHz.

BBR0

This instruction branches to the indicated address if bit 0 is clear in the indicated base-page memory location.

BBR0 : Branch on Bit 0 Reset **4510**

$M(0)=0 \implies PC \leftarrow PC + R8$

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page+rel	BBR0 $nn,$rr	0F	3	0 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BBR1

This instruction branches to the indicated address if bit 1 is clear in the indicated base-page memory location.

BBR1 : Branch on Bit 1 Reset — **4510**

M(1)=0 $\Longrightarrow$ PC $\leftarrow$ PC + R8

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page+rel	BBR1 \$nn,\$rr	1F	3	5 [b]

[b] Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.

BBR2

This instruction branches to the indicated address if bit 2 is clear in the indicated base-page memory location.

BBR2 : Branch on Bit 2 Reset — **4510**

M(2)=0 $\Longrightarrow$ PC $\leftarrow$ PC + R8

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page+rel	BBR2 \$nn,\$rr	2F	3	5 [b]

[b] Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.

BBR3

This instruction branches to the indicated address if bit 3 is clear in the indicated base-page memory location.

BBR3 : Branch on Bit 3 Reset — **4510**

M(3)=0 $\Longrightarrow$ PC $\leftarrow$ PC + R8

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page+rel	BBR3 \$nn,\$rr	3F	3	4 [b]

[b] Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.

BBR4

This instruction branches to the indicated address if bit 4 is clear in the indicated base-page memory location.

BBR4 : Branch on Bit 4 Reset **4510**

M(4)=0 $\Longrightarrow$ PC $\leftarrow$ PC + R8

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page+rel	BBR4 \$nn,\$rr	4F	3	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BBR5

This instruction branches to the indicated address if bit 5 is clear in the indicated base-page memory location.

BBR5 : Branch on Bit 5 Reset **4510**

M(5)=0 $\Longrightarrow$ PC $\leftarrow$ PC + R8

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page+rel	BBR5 \$nn,\$rr	5F	3	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BBR6

This instruction branches to the indicated address if bit 6 is clear in the indicated base-page memory location.

BBR6 : Branch on Bit 6 Reset				**4510**
M(6)=0 $\Longrightarrow$ PC $\leftarrow$ PC + R8				
			N Z I C D V E	
			. . . .	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
base-page+rel	BBR6 $nn,$rr	6F	3	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BBR7

This instruction branches to the indicated address if bit 7 is clear in the indicated base-page memory location.

BBR7 : Branch on Bit 7 Reset				**4510**
M(7)=0 $\Longrightarrow$ PC $\leftarrow$ PC + R8				
			N Z I C D V E	
			. . . .	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
base-page+rel	BBR7 $nn,$rr	7F	3	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BBS0

This instruction branches to the indicated address if bit 0 is set in the indicated base-page memory location.

BBS0 : Branch on Bit 0 Set				**4510**
M(0)=1 $\Longrightarrow$ PC $\leftarrow$ PC + R8				
			N Z I C D V E	
			. . . .	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
base-page+rel	BBS0 $nn,$rr	8F	3	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BBS1

This instruction branches to the indicated address if bit 1 is set in the indicated base-page memory location.

BBS1 : Branch on Bit 1 Set **4510**

M(1)=1 $\Longrightarrow$ PC $\leftarrow$ PC + R8

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page+rel	BBS1 $nn,$rr	9F	3	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BBS2

This instruction branches to the indicated address if bit 2 is set in the indicated base-page memory location.

BBS2 : Branch on Bit 2 Set **4510**

M(2)=1 $\Longrightarrow$ PC $\leftarrow$ PC + R8

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page+rel	BBS2 $nn,$rr	AF	3	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BBS3

This instruction branches to the indicated address if bit 3 is set in the indicated base-page memory location.

BBS3 : Branch on Bit 3 Set				**4510**
M(3)=1 $\Longrightarrow$ PC $\leftarrow$ PC + R8				
			N Z I C D V E	
			· · · · · · ·	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
base-page+rel	BBS3 \$nn,\$rr	BF	3	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BBS4

This instruction branches to the indicated address if bit 4 is set in the indicated base-page memory location.

BBS4 : Branch on Bit 4 Set				**4510**
M(4)=1 $\Longrightarrow$ PC $\leftarrow$ PC + R8				
			N Z I C D V E	
			· · · · · · ·	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
base-page+rel	BBS4 \$nn,\$rr	CF	3	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BBS5

This instruction branches to the indicated address if bit 5 is set in the indicated base-page memory location.

BBS5 : Branch on Bit 5 Set				**4510**
M(5)=1 $\Longrightarrow$ PC $\leftarrow$ PC + R8				
			N Z I C D V E	
			· · · · · · ·	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
base-page+rel	BBS5 \$nn,\$rr	DF	3	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BBS6

This instruction branches to the indicated address if bit 6 is set in the indicated base-page memory location.

BBS6 : Branch on Bit 6 Set **4510**

$M(6)=1 \implies PC \leftarrow PC + R8$

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page+rel	BBS6 $nn,$rr	EF	3	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BBS7

This instruction branches to the indicated address if bit 7 is set in the indicated base-page memory location.

BBS7 : Branch on Bit 7 Set **4510**

$M(7)=1 \implies PC \leftarrow PC + R8$

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page+rel	BBS7 $nn,$rr	FF	3	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BCC

This instruction branches to the indicated address if the Carry Flag is clear.

BCC : Branch on Carry Flag Clear **4510**

C=0 $\implies$ PC $\leftarrow$ PC + R8 or PC $\leftarrow$ PC + R16

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
relative	BCC $rr	90	2	2 [b]
16-bit relative	BCC $rrrr	93	3	3 [b]

[b] Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.

BCS

This instruction branches to the indicated address if the Carry Flag is set.

BCS : Branch on Carry Flag Set **4510**

C=1 $\implies$ PC $\leftarrow$ PC + R8 or PC $\leftarrow$ PC + R16

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
relative	BCS $rr	B0	2	2 [b]
16-bit relative	BCS $rrrr	B3	3	3 [b]

[b] Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.

BEQ

This instruction branches to the indicated address if the Zero Flag is set. BEQ stands for branch if equal, because a CMP will result in the zero flag being set if the operants are equal.

BEQ : Branch on Zero Flag Set **4510**

Z=1 $\implies$ PC $\leftarrow$ PC + R8 or PC $\leftarrow$ PC + R16

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
relative	BEQ $rr	F0	2	2 [b]
16-bit relative	BEQ $rrrr	F3	3	3 [b]

[b] Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.

BIT

This instruction is used to test the bits stored in a memory location or the immediate argument of the opcode.

Bits 6 and 7 of the memory location's contents are directly copied into the Overflow Flag and Negative Flag. The Zero Flag is set or cleared based on the result of performing the binary AND of the Accumulator Register and the contents of the indicated memory location.

The immediate test will set the N and V flags with valid states (treating the argument as the memory value), which was not the case with the earlier 65C02 implementation.

Side effects

- The N flag will be set if the bit 7 of the memory location is set, otherwise it will be cleared.
- The V flag will be set if the bit 6 of the memory location is set, otherwise it will be cleared.
- The Z flag will be set if the result of A *AND* M is zero, otherwise it will be cleared.

BIT : Perform Bit Test **4510**

N ← M(7), V ← M(6), Z ← A *AND* M

N	Z	I	C	D	V	E
+	+	·	·	·	+	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page	BIT $nn	24	2	3	r
absolute	BIT $nnnn	2C	3	4	r
base-page,X	BIT $nn,X	34	2	3	pr
absolute,X	BIT $nnnn,X	3C	3	4	pr
immediate 8bit	BIT #$nn	89	2	2	

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BMI

This instruction branches to the indicated address if the Negative Flag is set. BMI stands for branch on minus.

BMI : Branch on Negative Flag Set **4510**

N=1 $\implies$ PC $\leftarrow$ PC + R8 *or* PC $\leftarrow$ PC + R16

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles	
relative	BMI $rr	30	2	2	*r*
16-bit relative	BMI $rrrr	33	3	3	*b*

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BNE

This instruction branches to the indicated address if the Zero Flag is clear. BNE stands for Branch if not equal, because a CMP will result in the zero flag being cleared if the operants are not equal.

BNE : Branch on Zero Flag Clear **4510**

Z=0 $\implies$ PC $\leftarrow$ PC + R8 *or* PC $\leftarrow$ PC + R16

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles	
relative	BNE $rr	D0	2	2	*b*
16-bit relative	BNE $rrrr	D3	3	3	*b*

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.

BPL

This instruction branches to the indicated address if the Negative Flag is clear. BPL stands for branch on plus.

BPL : Branch on Negative Flag Clear **4510**

N=0 $\implies$ PC $\leftarrow$ PC + R8 *or* PC $\leftarrow$ PC + R16

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles	
relative	BPL $rr	10	2	2	*b*
16-bit relative	BPL $rrrr	13	3	3	*b*

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.

BRA

This instruction branches to the indicated address.

BRA : Branch Unconditionally **4510**

PC ← PC + R8 *or* PC ← PC + R16

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
relative	BRA $rr	80	2	2 [b]
16-bit relative	BRA $rrrr	83	3	3 [b]

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.

BRK

The break command causes the microprocessor to go through an interrupt sequence under program control. The address of the BRK instruction + 2 is pushed to the stack along with the status register with the Break flag set. This allows the interrupt service routine to distinguish between IRQ events and BRK events. For example:

```
PLA          ; load status
PHA          ; restore stack
AND #$10     ; mask break flag
BNE DO_BREAK ; -> it was a BRK
...          ; else continue with IRQ server
```

Cite from: MCS6500 Microcomputer Family Programming Manual, January 1976, Second Edition, MOS Technology Inc., Page 144:

"The BRK is a single byte instruction and its addressing mode is Implied."

There are debates, that BRK could be seen as a two byte instruction with the addressing mode immediate, where the operand byte is discarded. The byte following the BRK could then be used as a call argument for the break handler. Commodore however used the BRK, as stated in the manual, as a single byte instruction, which breaks into the ML monitor, if present. These builtin monitors decremented the stacked PC, so that it could be used to return or jump directly to the code byte after the BRK.

BRK : Break to Interrupt **4510**

STACK ← PC + 2; PC ← ($FFFE)

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	BRK	00	1	7

BSR

This instruction branches to the indicated address, saving the address of the following instruction on the stack, so that the routine can be returned from using an RTS instruction.

This instruction is helpful for using relocatable code, as it provides a relative-addressed alternative to JSR.

BSR : Branch Sub-Routine **4510**

STACK $\leftarrow$ PC + len(V), PC $\leftarrow$ PC + V

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
16-bit relative	BSR $rrrr	63	3	3 [b]

[b] Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.

BVC

This instruction branches to the indicated address if the Overflow (V) Flag is clear.

BVC : Branch on Overflow Flag Clear **4510**

V=0 $\Longrightarrow$ PC $\leftarrow$ PC + R8 *or* PC $\leftarrow$ PC + R16

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
relative	BVC $rr	50	2	2 [b]
16-bit relative	BVC $rrrr	53	3	3 [b]

[b] Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.

BVS

This instruction branches to the indicated address if the Overflow (V) Flag is set.

BVS : Branch on Overflow Flag Set **4510**

V=1 $\Longrightarrow$ PC $\leftarrow$ PC + R8 *or* PC $\leftarrow$ PC + R16

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
relative	BVS $rr	70	2	2	b
16-bit relative	BVS $rrrr	73	3	3	b

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.

CLC

This instruction clears the Carry Flag.

Side effects

- The C flag is cleared.

CLC : Clear Carry Flag **4510**

C $\leftarrow$ 0

N	Z	I	C	D	V	E
·	·	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
implied	CLC	18	1	1	s

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

CLD

This instruction clears the Decimal Flag. Arithmetic operations will use normal binary arithmetic, instead of Binary-Coded Decimal (BCD).

Side effects

- The D flag is cleared.

CLD : Clear Decimal Flag **4510**

D $\leftarrow$ 0

N	Z	I	C	D	V	E
·	·	·	·	+	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
implied	CLD	D8	1	1	s

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

CLE

This instruction clears the Extended Stack Disable Flag. This causes the stack to be able to exceed 256 bytes in length, by allowing the processor to modify the value of the high byte of the stack address (SPH).

Side effects

- The E flag is cleared.

CLE : Clear Extended Stack Disable Flag **4510**

E ← 0

N	Z	I	C	D	V	E
·	·	·	·	·	·	+

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	CLE	02	1	1 s

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

CLI

This instruction clears the Interrupt Disable Flag. Interrupts will now be able to occur.

Side effects

- The I flag is cleared.

CLI : Clear Interrupt Disable Flag **4510**

I ← 0

N	Z	I	C	D	V	E
·	·	+	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	CLI	58	1	1 s

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

CLV

This instruction clears the Overflow Flag.

Side effects

- The V flag is cleared.

CLV : Clear Overflow Flag **4510**

V $\leftarrow$ 0

N	Z	I	C	D	V	E
·	·	·	·	·	+	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
implied	CLV	B8	1	1	^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

CMP

This instruction performs A − M, and sets the processor flags accordingly, but does not modify the contents of the Accumulator Register.

Side effects

- The N flag will be set if the result of A − M is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The C flag will be set if the result of A − M is zero or positive, i.e., if A is not less than M, otherwise it will be cleared.
- The Z flag will be set if the result of A − M is zero, otherwise it will be cleared.

CMP : Compare Accumulator **4510**

N,C,Z $\Leftarrow$ [A − M]

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
(indirect,X)	CMP ($nn,X)	C1	2	5	^{pr}
base-page	CMP $nn	C5	2	3	^{r}
immediate 8bit	CMP #$nn	C9	2	2	
absolute	CMP $nnnn	CD	3	4	^{r}
(indirect),Y	CMP ($nn),Y	D1	2	5	^{pr}
(indirect),Z	CMP ($nn),Z	D2	2	5	^{pr}
base-page,X	CMP $nn,X	D5	2	3	^{pr}
absolute,Y	CMP $nnnn,Y	D9	3	4	^{pr}
absolute,X	CMP $nnnn,X	DD	3	4	^{pr}

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

CPX

This instruction performs X − M, and sets the processor flags accordingly, but does not modify the contents of the Accumulator Register.

Side effects

- The N flag will be set if the result of X − M is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The C flag will be set if the result of X − M is zero or positive, i.e., if X is not less than M, otherwise it will be cleared.
- The Z flag will be set if the result of X − M is zero, otherwise it will be cleared.

CPX : Compare X Register **4510**

N,C,Z ⇐ [X − M]

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
immediate 8bit	CPX #$nn	E0	2	2	
base-page	CPX $nn	E4	2	3	r
absolute	CPX $nnnn	EC	3	4	r

r Add one cycle if clock speed is at 40 MHz.

CPY

This instruction performs Y − M, and sets the processor flags accordingly, but does not modify the contents of the Accumulator Register.

Side effects

- The N flag will be set if the result of Y − M is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The C flag will be set if the result of Y − M is zero or positive, i.e., if Y is not less than M, otherwise it will be cleared.
- The Z flag will be set if the result of Y − M is zero, otherwise it will be cleared.

CPY : Compare Y Register **4510**

N,C,Z ⇐ [Y − M]

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
immediate 8bit	CPY #$nn	C0	2	2	
base-page	CPY $nn	C4	2	3	r
absolute	CPY $nnnn	CC	3	4	r

r Add one cycle if clock speed is at 40 MHz.

CPZ

This instruction performs Z − M, and sets the processor flags accordingly, but does not modify the contents of the Accumulator Register.

Side effects

- The N flag will be set if the result of Z − M is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The C flag will be set if the result of Z − M is zero or positive, i.e., if Z is not less than M, otherwise it will be cleared.
- The Z flag will be set if the result of Z − M is zero, otherwise it will be cleared.

CPZ : Compare Z Register **4510**

N,C,Z ⇐ [Z − M]

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
immediate 8bit	CPZ #$nn	C2	2	2	
base-page	CPZ $nn	D4	2	3	r
absolute	CPZ $nnnn	DC	3	4	r

r Add one cycle if clock speed is at 40 MHz.

DEC

This instruction decrements the Accumulator Register or indicated memory location.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.

- The Z flag will be set if the result is zero, otherwise it will be cleared.

DEC : Decrement **4510**

A ← A − 1 *or* M ← M − 1

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
accumulator	DEC A	3A	1	1	*s*
base-page	DEC $nn	C6	2	5	*dmr*
absolute	DEC $nnnn	CE	3	6	*dmr*
base-page,X	DEC $nn,X	D6	2	5	*dmpr*
absolute,X	DEC $nnnn,X	DE	3	6	*dmpr*

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.
s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

DEW

This instruction decrements the indicated memory word in the Base Page. The low numbered address contains the least significant bits. For example, if memory location $12 contains $78 and memory location $13 contains $56, the instruction DEW $12 would cause memory location $12 to be set to $77.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

DEW : Decrement Memory Word **4510**

M16 ← M16 − 1

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page	DEW $nn	C3	2	7	*dmr*

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
r Add one cycle if clock speed is at 40 MHz.

DEX

This instruction decrements the X Register.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

DEX : Decrement X Register **4510**

$X \leftarrow X - 1$

			N	Z	I	C	D	V	E
			+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	DEX	CA	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

DEY

This instruction decrements the Y Register.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

DEY : Decrement Y Register **4510**

$Y \leftarrow Y - 1$

			N	Z	I	C	D	V	E
			+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	DEY	88	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

DEZ

This instruction decrements the Z Register.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

DEZ : Decrement Z Register **4510**

$Z \leftarrow Z - 1$

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	DEZ	3B	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

EOM

In contrast with the 6502, the NOP instruction on the 45GS02 performs two additional roles when in 4502 mode.

First, indicate the end of a memory mapping sequence caused by a MAP instruction, allowing interrupts to occur again.

Second, it instructs the processor that if the following instruction uses Base-Page Indirect Z Indexed addressing, that the processor should use a 32-bit pointer instead of a 16-bit 6502 style pointer. Such 32-bit addresses are unaffected by C64, C65 or MEGA65 memory banking. This allows fast and easy access to the entire address space of the MEGA65 without having to perform or be aware of any banking, or using the DMA controller. This addressing mode causes a two cycle penalty, caused by the time required to read the extra two bytes of the pointer.

NOTE: please take care if you use EOM/NOP after a Hypervisor Call for delay, as this might change your next instruction. CLV can be used as an alternative.

Side effects

- Removes the prohibition on all interrupts caused by the the MAP instruction, allowing Non-Maskable Interrupts to again occur, and IRQ interrupts, if the Interrupt Disable Flag is not set.

EOM : End of Mapping Sequence / No-Operation **4510**

Special

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	EOM	EA	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

EOR

This instruction performs a binary exclusive-OR operation of the argument with the accumulator, and stores the result in the accumulator. Only bits that were already set in the accumulator, or that are set in the argument will be set in the accumulator on completion, but not both.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

EOR : Binary Exclusive OR **4510**

A ← A *XOR* M

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
(indirect,X)	EOR ($nn,X)	41	2	5	r
base-page	EOR $nn	45	2	3	r
immediate 8bit	EOR #$nn	49	2	2	
absolute	EOR $nnnn	4D	3	4	r
(indirect),Y	EOR ($nn),Y	51	2	5	pr
(indirect),Z	EOR ($nn),Z	52	2	5	pr
base-page,X	EOR $nn,X	55	2	3	p
absolute,Y	EOR $nnnn,Y	59	3	4	pr
absolute,X	EOR $nnnn,X	5D	3	4	pr

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

INC

This instruction increments the Accumulator Register or indicated memory location.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

INC : Increment Memory or Accumulator **4510**

A ← A + 1 *or* M ← M + 1

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
accumulator	INC A	1A	1	1	*s*
base-page	INC $nn	E6	2	5	*dmr*
absolute	INC $nnnn	EE	3	6	*dmr*
base-page,X	INC $nn,X	F6	2	5	*dmpr*
absolute,X	INC $nnnn,X	FE	3	6	*dpr*

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.
s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

INW

This instruction increments the indicated memory word in the Base Page. The low numbered address contains the least significant bits. For example, if memory location $12 contains $78 and memory location $13 contains $56, the instruction INW $12 would cause memory location $12 to be set to $79.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

INW : Increment Memory Word **4510**

M16 ← M16 + 1

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page	INW $nn	E3	2	7	*dmr*

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
r Add one cycle if clock speed is at 40 MHz.

INX

This instruction increments the X Register, i.e., adds 1 to it.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

INX : Increment X Register **4510**

X ← X + 1

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	INX	E8	1	1 [s]

[s] Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

INY

This instruction increments the Y Register, i.e., adds 1 to it.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

INY : Increment Y Register **4510**

Y ← Y + 1

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	INY	C8	1	1 [s]

[s] Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

INZ

This instruction increments the Z Register, i.e., adds 1 to it.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

INZ : Increment Z Register **4510**

Z ← Y + 1

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
implied	INZ	1B	1	1	s

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

JMP

This instruction sets the Program Counter (PC) Register to the address indicated by the instruction, causing execution to continue from that address.

JMP : Jump to Address **4510**

PC ← M2:M1

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
absolute	JMP $nnnn	4C	3	3	
(indirect)	JMP ($nnnn)	6C	3	5	r
(indirect,X)	JMP ($nnnn,X)	7C	3	6	mp

m Subtract non-bus cycles when at 40MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

JSR

This instruction saves the address of the instruction following the JSR instruction onto the stack, and then sets the Program Counter (PC) Register to the address indicated by the instruction, causing execution to continue from that address. Because the return address has been saved on the stack, the RTS instruction can be used to return from the called sub-routine and resume execution following the JSR instruction.

NOTE: This instruction actually pushes the address of the last byte of the JSR instruction onto the stack. The RTS instruction naturally is aware of this, and increments the address on popping it from the stack, before setting the Program Counter (PC) register.

JSR : Jump to Sub-Routine					**4510**
PC ← M2:M1, STACK ← PCH:PCL					
			N Z I C D V E		
			· · · · · · ·		
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**	
absolute	JSR $nnnn	20	3	5	
(indirect)	JSR ($nnnn)	22	3	5	r
(indirect,X)	JSR ($nnnn,X)	23	3	5	pr

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

LDA

This instruction loads the Accumulator Register with the indicated value, or with the contents of the indicated location.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

LDA : Load Accumulator					**4510**
A ← M					
			N Z I C D V E		
			+ + · · · · ·		
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**	
(indirect,X)	LDA ($nn,X)	A1	2	5	pr
base-page	LDA $nn	A5	2	3	r
immediate 8bit	LDA #$nn	A9	2	2	
absolute	LDA $nnnn	AD	3	4	r
(indirect),Y	LDA ($nn),Y	B1	2	5	pr
(indirect),Z	LDA ($nn),Z	B2	2	5	pr
base-page,X	LDA $nn,X	B5	2	3	pr
absolute,Y	LDA $nnnn,Y	B9	3	4	pr
absolute,X	LDA $nnnn,X	BD	3	4	pr
(immediate,SP),Y	LDA ($nn,SP),Y	E2	2	6	mpr

m Subtract non-bus cycles when at 40MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

LDX

This instruction loads the X Register with the indicated value, or with the contents of the indicated location.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

LDX : Load X Register **4510**

X ← M

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
immediate 8bit	LDX #$nn	A2	2	2	
base-page	LDX $nn	A6	2	3	r
absolute	LDX $nnnn	AE	3	4	r
base-page,Y	LDX $nn,Y	B6	2	5	pr
absolute,Y	LDX $nnnn,Y	BE	3	4	pr

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

LDY

This instruction loads the Y Register with the indicated value, or with the contents of the indicated location.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

LDY : Load Y Register **4510**

Y ← M

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
immediate 8bit	LDY #$nn	A0	2	2	
base-page	LDY $nn	A4	2	3	r
absolute	LDY $nnnn	AC	3	4	r
base-page,X	LDY $nn,X	B4	2	3	pr
absolute,X	LDY $nnnn,X	BC	3	4	pr

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

LDZ

This instruction loads the Z Register with the indicated value, or with the contents of the indicated location.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

LDZ : Load Z Register **4510**

Z ← M

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
immediate 8bit	LDZ #$nn	A3	2	2	
absolute	LDZ $nnnn	AB	3	4	r
absolute,X	LDZ $nnnn,X	BB	3	4	pr

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

LSR

This instruction shifts either the Accumulator or contents of the provided memory location one bit right. Bit 7 will be set to zero, and the bit 0 will be shifted out into the Carry Flag

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The C flag will be set if bit 0 of the value was set, prior to being shifted.

LSR : Logical Shift Right **4510**

A ← A≫1, C ← A(0) *or* M ← M≫1, C ← M(0)

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page	LSR $nn	46	2	4	*r*
accumulator	LSR A	4A	1	1	*s*
absolute	LSR $nnnn	4E	3	5	*r*
base-page,X	LSR $nn,X	56	2	3	*pr*
absolute,X	LSR $nnnn,X	5E	3	5	*pr*

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.
s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

MAP

This instruction sets the C65 or MEGA65 style memory map, depending on the values in the Accumulator, X, Y and Z registers.

Care should be taken to ensure that after the execution of an MAP instruction that appropriate memory is mapped at the location of the following instruction. Failure to do so will result in unpredictable results.

Further information on this instruction is available in Appendix K.

Side effects

- The memory map is immediately changed to that requested.
- All interrupts, including Non-Maskable Interrupts (NMIs) are blocked from occurring until an EOM (NOP) instruction is encountered.

MAP : Set Memory Map				**4510**
Special				
			N Z I C D V E	
			· · · · · · ·	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
implied	MAP	5C	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

NEG

This instruction replaces the contents of the Accumulator Register with the two's-complement of the contents of the Accumulator Register.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

NEG : Negate Accumulator				**4510**
A ← (A *XOR* \$FF) + 1				
			N Z I C D V E	
			+ + · · · · ·	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
accumulator	NEG A	42	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

ORA

This instruction performs a binary OR operation of the argument with the accumulator, and stores the result in the accumulator. Only bits that were already set in the accumulator, or that are set in the argument will be set in the accumulator on completion, or both.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

ORA : Binary OR **4510**

A ← A *OR* M

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
(indirect,X)	ORA ($nn,X)	01	2	6	*pr*
base-page	ORA $nn	05	2	3	*r*
immediate 8bit	ORA #$nn	09	2	2	
absolute	ORA $nnnn	0D	3	4	*r*
(indirect),Y	ORA ($nn),Y	11	2	5	*pr*
(indirect),Z	ORA ($nn),Z	12	2	5	*pr*
base-page,X	ORA $nn,X	15	2	3	*r*
absolute,Y	ORA $nnnn,Y	19	3	4	*r*
absolute,X	ORA $nnnn,X	1D	3	4	*pr*

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

PHA

This instruction pushes the contents of the Accumulator Register onto the stack, and decrements the value of the Stack Pointer by 1.

PHA : Push Accumulator Register onto the Stack **4510**

STACK ← A, SP ← SP − 1

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	PHA	48	1	2

PHP

This instruction pushes the contents of the Processor Flags onto the stack, and decrements the value of the Stack Pointer by 1.

PHP : Push Processor Flags onto the Stack **4510**

STACK ← P, SP ← SP − 1

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	PHP	08	1	2

PHW

This instruction pushes either a 16-bit literal value or the memory word indicated onto the stack, and decrements the value of the Stack Pointer by 2.

PHW : Push Word onto the Stack **4510**

STACK $\leftarrow$ M1:M2, SP $\leftarrow$ SP $-$ 2

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
immediate 16bit	PHW #$nnnn	F4	3	5	m
absolute	PHW $nnnn	FC	3	7	m

m Subtract non-bus cycles when at 40MHz.

PHX

This instruction pushes the contents of the X Register onto the stack, and decrements the value of the Stack Pointer by 1.

PHX : Push X Register onto the Stack **4510**

STACK $\leftarrow$ X, SP $\leftarrow$ SP $-$ 1

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
implied	PHX	DA	1	3	m

m Subtract non-bus cycles when at 40MHz.

PHY

This instruction pushes the contents of the Y Register onto the stack, and decrements the value of the Stack Pointer by 1.

PHY : Push Y Register onto the Stack **4510**

STACK $\leftarrow$ Y, SP $\leftarrow$ SP $-$ 1

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	PHY	5A	1	2

PHZ

This instruction pushes the contents of the Z Register onto the stack, and decrements the value of the Stack Pointer by 1.

PHZ : Push Z Register onto the Stack **4510**

STACK ← z, SP ← SP − 1

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
implied	PHZ	DB	1	3	m

m Subtract non-bus cycles when at 40MHz.

PLA

This instruction replaces the contents of the Accumulator Register with the top value from the stack, and increments the value of the Stack Pointer by 1.

PLA : Pull Accumulator Register from the Stack **4510**

A ← STACK, SP ← SP + 1

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
implied	PLA	68	1	4	m

m Subtract non-bus cycles when at 40MHz.

PLP

This instruction replaces the contents of the Processor Flags with the top value from the stack, and increments the value of the Stack Pointer by 1.

NOTE: This instruction does NOT replace the Extended Stack Disable Flag (E Flag), or the Software Interrupt Flag (B Flag)

PLP : Pull Processor Flags from the Stack **4510**

A ← STACK, SP ← SP + 1

N	Z	I	C	D	V	E
+	+	+	+	+	+	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
implied	PLP	28	1	4	m

m Subtract non-bus cycles when at 40MHz.

PLX

This instruction replaces the contents of the X Register with the top value from the stack, and increments the value of the Stack Pointer by 1.

PLX : Pull X Register from the Stack **4510**

X ← STACK, SP ← SP + 1

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
implied	PLX	FA	1	4	m

m Subtract non-bus cycles when at 40MHz.

PLY

This instruction replaces the contents of the Y Register with the top value from the stack, and increments the value of the Stack Pointer by 1.

PLY : Pull Y Register from the Stack **4510**

Y ← STACK, SP ← SP + 1

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
implied	PLY	7A	1	4	m

m Subtract non-bus cycles when at 40MHz.

PLZ

This instruction replaces the contents of the Z Register with the top value from the stack, and increments the value of the Stack Pointer by 1.

PLZ : Pull Z Register from the Stack **4510**

Z ← STACK, SP ← SP + 1

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
implied	PLZ	FB	1	4	m

m Subtract non-bus cycles when at 40MHz.

RMB0

This instruction clears bit zero of the indicated address. No flags are modified, regardless of the result.

RMB0 : Reset Bit 0 in Base Page **4510**

M(0) ← 0

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page	RMB0 $nn	07	2	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

RMB1

This instruction clears bit 1 of the indicated address. No flags are modified, regardless of the result.

RMB1 : Reset Bit 1 in Base Page **4510**

M(1) ← 0

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page	RMB1 $nn	17	2	4 ^{br}

b Add one cycle if branch is taken.
Add one more cycle if branch taken crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

RMB2

This instruction clears bit 2 of the indicated address. No flags are modified, regardless of the result.

RMB2 : Reset Bit 2 in Base Page **4510**

M(2) ← 0

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page	RMB2 $nn	27	2	4 ^{r}

r Add one cycle if clock speed is at 40 MHz.

RMB3

This instruction clears bit 3 of the indicated address. No flags are modified, regardless of the result.

RMB3 : Reset Bit 3 in Base Page **4510**

$M(3) \leftarrow 0$

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page	RMB3 $nn	37	2	4 ^{r}

r Add one cycle if clock speed is at 40 MHz.

RMB4

This instruction clears bit 4 of the indicated address. No flags are modified, regardless of the result.

RMB4 : Reset Bit 4 in Base Page **4510**

$M(4) \leftarrow 0$

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page	RMB4 $nn	47	2	4 ^{r}

r Add one cycle if clock speed is at 40 MHz.

RMB5

This instruction clears bit 5 of the indicated address. No flags are modified, regardless of the result.

RMB5 : Reset Bit 5 in Base Page **4510**

$M(5) \leftarrow 0$

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page	RMB5 $nn	57	2	4 ^{r}

r Add one cycle if clock speed is at 40 MHz.

RMB6

This instruction clears bit 6 of the indicated address. No flags are modified, regardless of the result.

RMB6 : Reset Bit 6 in Base Page					**4510**
$M(6) \leftarrow 0$					
			N Z I C D V E		
			. . . .		
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**	
base-page	RMB6 $nn	67	2	5	^{r}

r Add one cycle if clock speed is at 40 MHz.

RMB7

This instruction clears bit 7 of the indicated address. No flags are modified, regardless of the result.

RMB7 : Reset Bit 7 in Base Page					**4510**
$M(7) \leftarrow 0$					
			N Z I C D V E		
			. . . .		
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**	
base-page	RMB7 $nn	77	2	4	^{r}

r Add one cycle if clock speed is at 40 MHz.

ROL

This instruction shifts either the Accumulator or contents of the provided memory location one bit left. Bit 0 will be set to the current value of the Carry Flag, and the bit 7 will be shifted out into the Carry Flag

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The C flag will be set if bit 7 of the value was set, prior to being shifted.

ROL : Rotate Left Memory or Accumulator **4510**

M ← M≪1, C ← M(7), M(0) ← C

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page	ROL $nn	26	2	4	*r*
accumulator	ROL A	2A	1	1	*s*
absolute	ROL $nnnn	2E	3	5	*r*
base-page,X	ROL $nn,X	36	2	5	*pr*
absolute,X	ROL $nnnn,X	3E	3	5	*pr*

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.
s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

ROR

This instruction shifts either the Accumulator or contents of the provided memory location one bit right. Bit 7 will be set to the current value of the Carry Flag, and the bit 0 will be shifted out into the Carry Flag

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The C flag will be set if bit 7 of the value was set, prior to being shifted.

ROR : Rotate Right Memory or Accumulator **4510**

M ← M≫1, C ← M(0), M(7) ← C

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page	ROR $nn	66	2	5	*r*
accumulator	ROR A	6A	1	1	*s*
absolute	ROR $nnnn	6E	3	6	*r*
base-page,X	ROR $nn,X	76	2	5	*dmpr*
absolute,X	ROR $nnnn,X	7E	3	5	*dmpr*

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.
s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

ROW

This instruction rotates the contents of the indicated memory word one bit left. Bit 0 of the low byte will be set to the current value of the Carry Flag, and the bit 7 of the high byte will be shifted out into the Carry Flag

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The C flag will be set if bit 7 of the upper byte was set, prior to being shifted.

ROW : Rotate Word Left **4510**

M2:M1 $\leftarrow$ M2:M1$\ll$1, C $\leftarrow$ M2(7), M1(0) $\leftarrow$ C

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
absolute	ROW $nnnn	EB	3	5	dmr

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
r Add one cycle if clock speed is at 40 MHz.

RTI

This instruction pops the processor flags from the stack, and then pops the Program Counter (PC) register from the stack, allowing an interrupted program to resume.

- The 6502 Processor Flags are restored from the stack.
- Neither the B (Software Interrupt) nor E (Extended Stack) flags are set by this instruction.

RTI : Return From Interrupt **4510**

P $\leftarrow$ STACK, PC $\leftarrow$ STACK

N	Z	I	C	D	V	E
+	+	+	+	+	+	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
implied	RTI	40	1	6	m

m Subtract non-bus cycles when at 40MHz.

RTS

This instruction adds an optional argument to the Stack Pointer (SP) Register, and then pops the Program Counter (PC) register from the stack, allowing a routine to return to its caller.

RTS : Return From Subroutine **4510**

PC ← STACK *or* PC ← STACK + M, SP ← SP − 2

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	RTS	60	1	6 ^{m}
immediate 8bit	RTS #$nn	62	2	4

m Subtract non-bus cycles when at 40MHz.

SBC

This instruction performs A $-$ M $-$ 1 $+$ C, and sets the processor flags accordingly. The result is stored in the Accumulator Register.

NOTE: If the D flag is set, then the addition is performed using binary Coded Decimal.

Side effects

- The N flag will be set if the result of A − M is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The C flag will be set if the result of A − M is zero or positive, i.e., if A is not less than M, otherwise it will be cleared.
- The V flag will be set if the result has a different sign to both of the arguments, otherwise it will be cleared. If the flag is set, this indicates that a signed overflow has occurred.
- The Z flag will be set if the result of A − M is zero, otherwise it will be cleared.

SBC : Subtract With Carry $A \leftarrow - M - 1 + C$			**4510** N Z I C D V E: + + · + · + ·		
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**	
(indirect,X)	SBC ($nn,X)	E1	2	3	mp
base-page	SBC $nn	E5	2	3	r
immediate 8bit	SBC #$nn	E9	2	2	
absolute	SBC $nnnn	ED	3	4	r
(indirect),Y	SBC ($nn),Y	F1	2	3	pr
(indirect),Z	SBC ($nn),Z	F2	2	5	pr
base-page,X	SBC $nn,X	F5	2	3	pr
absolute,Y	SBC $nnnn,Y	F9	3	4	pr
absolute,X	SBC $nnnn,X	FD	3	4	pr

m Subtract non-bus cycles when at 40MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

SEC

This instruction sets the Carry Flag.

Side effects

- The C flag is set.

SEC : Set Carry Flag $C \leftarrow 1$			**4510** N Z I C D V E: · · · + · · ·		
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**	
implied	SEC	38	1	1	s

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

SED

This instruction sets the Decimal Flag. Binary arithmetic will now use Binary-Coded Decimal (BCD) mode.

NOTE: The C64's interrupt handler does not clear the Decimal Flag, which makes it dangerous to set the Decimal Flag without first setting the Interrupt Disable Flag.

Side effects

- The D flag is set.

SED : Set Decimal Flag **4510**

$D \leftarrow 1$

N	Z	I	C	D	V	E
·	·	·	·	+	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	SED	F8	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

SEE

This instruction sets the Extended Stack Disable Flag. This causes the stack to operate as on the 6502, i.e., limited to a single page of memory. The page of memory in which the stack is located can still be modified by setting the Stack Pointer High (SPH) Register.

Side effects

- The E flag is set.

SEE : Set Extended Stack Disable Flag **4510**

$E \leftarrow 1$

N	Z	I	C	D	V	E
·	·	·	·	·	·	+

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	SEE	03	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

SEI

This instruction sets the Interrupt Disable Flag. Normal (IRQ) interrupts will no longer be able to occur. Non-Maskable Interrupts (NMI) will continue to occur, as their name suggests.

Side effects

- The I flag is set.

SEI : Set Interrupt Disable Flag **4510**

I ← 1

N	Z	I	C	D	V	E
·	·	+	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	SEI	78	1	1 [s]

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

SMB0

This instruction sets bit zero of the indicated address. No flags are modified, regardless of the result.

SMB0 : Set Bit 0 in Base Page **4510**

M(0) ← 1

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page	SMB0 $nn	87	2	4 [r]

r Add one cycle if clock speed is at 40 MHz.

SMB1

This instruction sets bit 1 of the indicated address. No flags are modified, regardless of the result.

SMB1 : Set Bit 1 in Base Page **4510**

M(1) ← 1

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page	SMB1 $nn	97	2	4 [r]

r Add one cycle if clock speed is at 40 MHz.

SMB2

This instruction sets bit 2 of the indicated address. No flags are modified, regardless of the result.

SMB2 : Set Bit 2 in Base Page				**4510**
M(2) ← 1				
			N Z I C D V E	
			· · · · · · ·	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
base-page	SMB2 $nn	A7	2	4 ^{r}

r Add one cycle if clock speed is at 40 MHz.

SMB3

This instruction sets bit 3 of the indicated address. No flags are modified, regardless of the result.

SMB3 : Set Bit 3 in Base Page				**4510**
M(3) ← 1				
			N Z I C D V E	
			· · · · · · ·	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
base-page	SMB3 $nn	B7	2	4 ^{r}

r Add one cycle if clock speed is at 40 MHz.

SMB4

This instruction sets bit 4 of the indicated address. No flags are modified, regardless of the result.

SMB4 : Set Bit 4 in Base Page				**4510**
M(4) ← 1				
			N Z I C D V E	
			· · · · · · ·	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
base-page	SMB4 $nn	C7	2	4 ^{r}

r Add one cycle if clock speed is at 40 MHz.

SMB5

This instruction sets bit 5 of the indicated address. No flags are modified, regardless of the result.

SMB5 : Set Bit 5 in Base Page **4510**

M(5) $\leftarrow$ 1

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page	SMB5 $nn	D7	2	4 ^{r}

r Add one cycle if clock speed is at 40 MHz.

SMB6

This instruction sets bit 6 of the indicated address. No flags are modified, regardless of the result.

SMB6 : Set Bit 6 in Base Page **4510**

M(6) $\leftarrow$ 1

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page	SMB6 $nn	E7	2	4 ^{r}

r Add one cycle if clock speed is at 40 MHz.

SMB7

This instruction sets bit 7 of the indicated address. No flags are modified, regardless of the result.

SMB7 : Set Bit 7 in Base Page **4510**

M(7) $\leftarrow$ 1

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
base-page	SMB7 $nn	F7	2	4 ^{r}

r Add one cycle if clock speed is at 40 MHz.

STA

This instruction stores the contents of the Accumulator Register into the indicated location.

STA : Store Accumulator **4510**

M ← A

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles	
(indirect,X)	STA ($nn,X)	81	2	5	p
(immediate,SP),Y	STA ($nn,SP),Y	82	2	6	p
base-page	STA $nn	85	2	3	
absolute	STA $nnnn	8D	3	4	
(indirect),Y	STA ($nn),Y	91	2	5	p
(indirect),Z	STA ($nn),Z	92	2	5	p
base-page,X	STA $nn,X	95	2	3	p
absolute,Y	STA $nnnn,Y	99	3	4	p
absolute,X	STA $nnnn,X	9D	3	4	p

p Add one cycle if indexing crosses a page boundary.

STX

This instruction stores the contents of the X Register into the indicated location.

STX : Store X Register **4510**

M ← X

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page	STX $nn	86	2	3	
absolute	STX $nnnn	8E	3	4	
base-page,Y	STX $nn,Y	96	2	3	p
absolute,Y	STX $nnnn,Y	9B	3	4	p

p Add one cycle if indexing crosses a page boundary.

STY

This instruction stores the contents of the Y Register into the indicated location.

STY : Store Y Register **4510**

M ← Y

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page	STY $nn	84	2	3	
absolute,X	STY $nnnn,X	8B	3	4	*p*
absolute	STY $nnnn	8C	3	4	
base-page,X	STY $nn,X	94	2	3	*p*

p Add one cycle if indexing crosses a page boundary.

STZ

This instruction stores the contents of the Z Register into the indicated location.

STZ : Store Z Register **4510**

M ← Z

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page	STZ $nn	64	2	3	
base-page,X	STZ $nn,X	74	2	3	
absolute	STZ $nnnn	9C	3	4	
absolute,X	STZ $nnnn,X	9E	3	4	*p*

p Add one cycle if indexing crosses a page boundary.

TAB

This instruction sets the Base Page register to the contents of the Accumulator Register. This allows the relocation of the 6502's Zero-Page into any page of memory.

TAB : Transfer Accumulator into Base Page Register **4510**

B ← A

N	Z	I	C	D	V	E
·	·	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
implied	TAB	5B	1	1	*s*

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

TAX

This instruction loads the X Register with the contents of the Accumulator Register.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

TAX : Transfer Accumulator Register into the X Register **4510**

X ← A

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	TAX	AA	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

TAY

This instruction loads the Y Register with the contents of the Accumulator Register.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

TAY : Transfer Accumulator Register into the Y Register **4510**

Y ← A

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	TAY	A8	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

TAZ

This instruction loads the Z Register with the contents of the Accumulator Register.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

TAZ : Transfer Accumulator Register into the Z Register **4510**

Z ← A

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	TAZ	4B	1	1 s

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

TBA

This instruction loads the Accumulator Register with the contents of the Base Page Register.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

TBA : Transfer Base Page Register into the Accumulator **4510**

A ← B

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	TBA	7B	1	1 s

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

TRB

This instruction performs a binary AND of the negation of the Accumulator Register and the indicated memory location, storing the result there. That is, any bits set in the Accumulator Register will be reset in the indicated memory location.

It also performs a test for any bits in common between the accumulator and indicated memory location. This can be used to construct simple shared-memory multi-processor systems, by providing an atomic means of setting a semaphore or acquiring a lock.

Side effects

- The Z flag will be set if the binary AND of the Accumulator Register and contents of the indicated memory location prior are zero, prior to the execution of the instruction.

TRB : Test and Reset Bit **4510**

M ← M *AND* (*NOT* A)

N	Z	I	C	D	V	E
·	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page	TRB $nn	14	2	5	r
absolute	TRB $nnnn	1C	3	4	r

r Add one cycle if clock speed is at 40 MHz.

TSB

This instruction performs a binary OR of the Accumulator Register and the indicated memory location, storing the result there. That is, any bits set in the Accumulator Register will be set in the indicated memory location.

It also performs a test for any bits in common between the accumulator and indicated memory location. This can be used to construct simple shared-memory multi-processor systems, by providing an atomic means of setting a semaphore or acquiring a lock.

Side effects

- The Z flag will be set if the binary AND of the Accumulator Register and contents of the indicated memory location prior are zero, prior to the execution of the instruction.

TSB : Test and Set Bit **4510**

M ← M *OR* A

N	Z	I	C	D	V	E
·	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page	TSB $nn	04	2	3	r
absolute	TSB $nnnn	0C	3	5	r

r Add one cycle if clock speed is at 40 MHz.

TSX

This instruction loads the X Register with the contents of the Stack Pointer Low (SPL) Register.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

TSX : Transfer Stack Pointer Low Register into the X Register **4510**

X ← SPL

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	TSX	BA	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

TSY

This instruction loads the Y Register with the contents of the Stack Pointer High (SPH) Register.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

TSY : Transfer Stack Pointer High Register into the Y Register **4510**

Y ← SPH

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	TSY	0B	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

TXA

This instruction loads the Accumulator Register with the contents of the X Register.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

TXA : Transfer X Register into the Accumulator Register				4510
A ← X				
			N Z I C D V E	
			+ + · · · · ·	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
implied	TXA	8A	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

TXS

This instruction sets the low byte of the Stack Pointer (SPL) register to the contents of the X Register.

TXS : Transfer X Register into Stack Pointer Low Register				4510
SPL ← X				
			N Z I C D V E	
			· · · · · · ·	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
implied	TXS	9A	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

TYA

This instruction loads the Accumulator Register with the contents of the Y Register.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

TYA : Transfer Y Register into the Accumulator Register				4510
A ← Y				
			N Z I C D V E	
			+ + · · · · ·	
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**
implied	TYA	98	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

TYS

This instruction sets the high byte of the Stack Pointer (SPH) register to the contents of the Y Register. This allows changing the memory page where the stack is located (if the Extended Stack Disable Flag (E) is set), or else allows setting the current Stack Pointer to any page in memory, if the Extended Stack Disable Flag (E) is clear.

TYS : Transfer Y Register into Stack Pointer High Register **4510**

SPH ← Y

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	TYS	2B	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

TZA

This instruction loads the Accumulator Register with the contents of the Z Register.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

TZA : Transfer Z Register into the Accumulator Register **4510**

A ← Z

N	Z	I	C	D	V	E
+	+	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles
implied	TZA	6B	1	1 ^{s}

s Instruction requires 2 cycles when CPU is run at 1MHz or 2MHz.

45GS02 COMPOUND INSTRUCTIONS

As the 4510 has no unallocated opcodes, the 45GS02 uses compound instructions to implement its extension. These compound instructions consist of one or more single byte instructions placed immediately before a conventional instruction. These prefixes instruct the 45GS02 to treat the following instruction differently, as described in Chapter/Appendix K on page K-3.

You can find them highlighted in the 4510 Opcode Table (see Chapter/Appendix L on page L-60)

ADC

This instruction adds the argument and the Carry Flag to the contents of the Accumulator Register. If the D flag is set, then the addition is performed using Binary Coded Decimal.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The V flag will be set if the result has a different sign to both of the arguments, otherwise it will be cleared. If the flag is set, this indicates that a signed overflow has occurred.
- The C flag will be set if the unsigned result is >255, or >99 if the D flag is set.

ADC : Add with carry **45GS02**

A ← A + M + C

N	Z	I	C	D	V	E
+	+	·	+	·	+	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
[indirect],Z	ADC [$nn],Z	EA 72	3	7	ipr

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

ADCQ

This instruction adds the argument and the Carry Flag to the contents of the 32-bit Q Pseudo Register.

NOTE: the indicated memory location is treated as the first byte of a 32-bit little-endian value.

NOTE: If the D flag is set, the operation is undefined and subject to change.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.

- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The V flag will be set if the result has a different sign to both of the arguments, otherwise it will be cleared. If the flag is set, this indicates that a signed overflow has occurred.
- The C flag will be set if the unsigned result is $\geq 2^{32}$.

ADCQ : Add with carry Quad **45GS02**

Q ← Q + M + C

N	Z	I	C	D	V	E
+	+	·	+	·	+	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page quad	ADCQ $nn	42 42 65	4	8	r
absolute quad	ADCQ $nnnn	42 42 6D	5	9	r
(indirect quad)	ADCQ ($nn)	42 42 72	4	10	ipr
[indirect quad]	ADCQ [$nn]	42 42 EA 72	5	13	ipr

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

AND

This instruction performs a binary AND operation of the argument with the accumulator, and stores the result in the accumulator. Only bits that were already set in the accumulator, and that are set in the argument will be set in the accumulator on completion.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

AND : Binary AND **45GS02**

A ← A *AND* M

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
[indirect],Z	AND [$nn],Z	EA 32	3	7	ipr

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

ANDQ

This instruction performs a binary AND operation of the argument with the Q pseudo register, and stores the result in the accumulator. Only bits that were already set in the Q pseudo register, and that are set in the argument will be set in the Q pseudo register on completion.

NOTE: the indicated memory location is treated as the first byte of a 32-bit little-endian value.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

ANDQ : Binary AND Quad **45GS02**

Q ← Q *AND* M

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page quad	ANDQ $nn	42 42 25	4	8	r
absolute quad	ANDQ $nnnn	42 42 2D	5	9	r
(indirect quad)	ANDQ ($nn)	42 42 32	4	10	ipr
[indirect quad]	ANDQ [$nn]	42 42 EA 32	5	13	ipr

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

ASLQ

This instruction shifts either the Q pseudo-register or contents of the provided memory location and following three one bit left, treating them as holding a little-endian 32-bit value. Bit 0 will be set to zero, and the bit 31 will be shifted out into the Carry Flag

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The C flag will be set if bit 31 of the value was set, prior to being shifted, otherwise it will be cleared.

ASLQ : Arithmetic Shift Left Quad **45GS02**

Q ← Q<<1 *or* M ← M<<1

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page quad	ASLQ $nn	42 42 06	4	12	*dmr*
Q Pseudo Register	ASLQ Q	42 42 0A	3	3	
absolute quad	ASLQ $nnnn	42 42 0E	5	13	*dmr*
base-page quad,X	ASLQ $nn,X	42 42 16	4	12	*dmpr*
absolute quad,X	ASLQ $nnnn,X	42 42 1E	5	13	*dmpr*

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

ASRQ

This instruction shifts either the Q pseudo-register or contents of the provided memory location and following three one bit right, treating them as holding a little-endian 32-bit value. Bit 31 is considered to be a sign bit, and is preserved. The content of bit 0 will be shifted out into the Carry Flag

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The C flag will be set if bit 0 of the value was set, prior to being shifted, otherwise it will be cleared.

ASRQ : Arithmetic Shift Right Quad **45GS02**

Q ← Q>>1 *or* M ← M>>1

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
Q Pseudo Register	ASRQ Q	42 42 43	3	3	
base-page quad	ASRQ $nn	42 42 44	4	12	*dmr*
base-page quad,X	ASRQ $nn,X	42 42 54	4	12	*dmpr*

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

BITQ

This instruction is used to test the bits stored in a memory location and following three, treating them as holding a little-endian 32-bit value. Bits 30 and 31 of the memory location's contents are directly copied into the Overflow Flag and Negative Flag. The Zero Flag is set or cleared based on the result of performing the binary AND of the Q Register and the contents of the indicated memory location.

Side effects

- The N flag will be set if the bit 31 of the memory location is set, otherwise it will be cleared.
- The V flag will be set if the bit 30 of the memory location is set, otherwise it will be cleared.
- The Z flag will be set if the result of Q *AND* M is zero, otherwise it will be cleared.

BITQ : Perform Bit Test Quad **45GS02**

N ← M(31), V ← M(30), Z ← Q *AND* M

N	Z	I	C	D	V	E
+	+	·	·	·	+	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page quad	BITQ $nn	42 42 24	4	8	r
absolute quad	BITQ $nnnn	42 42 2C	5	9	r

r Add one cycle if clock speed is at 40 MHz.

CMP

This instruction performs A − M, and sets the processor flags accordingly, but does not modify the contents of the Accumulator Register.

Side effects

- The N flag will be set if the result of A − M is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The C flag will be set if the result of A − M is zero or positive, i.e., if A is not less than M, otherwise it will be cleared.
- The Z flag will be set if the result of A − M is zero, otherwise it will be cleared.

CMP : Compare Accumulator **45GS02**

N,C,Z ⇐ [A − M]

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
[indirect],Z	CMP [$nn],Z	EA D2	3	7	*ipr*

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

CMPQ

This instruction performs Q − M, and sets the processor flags accordingly, but does not modify the contents of the Q Register.

NOTE: the indicated memory location is treated as the first byte of a 32-bit little-endian value.

Side effects

- The N flag will be set if the result of A − M is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The C flag will be set if the result of A − M is zero or positive, i.e., if A is not less than M, otherwise it will be cleared.
- The Z flag will be set if the result of A − M is zero, otherwise it will be cleared.

CMPQ : Compare Q Pseudo Register **45GS02**

N,C,Z ⇐ [Q − M]

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page quad	CMPQ $nn	42 42 C5	4	8	*r*
absolute quad	CMPQ $nnnn	42 42 CD	5	9	*r*
(indirect quad)	CMPQ ($nn)	42 42 D2	4	10	*ipr*
[indirect quad]	CMPQ [$nn]	42 42 EA D2	5	13	*ipr*

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

DEQ

This instruction decrements the Q psuedo register or indicated memory location.

NOTE: the indicated memory location is treated as the first byte of a 32-bit little-endian value.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

DEQ : Decrement Quad **45GS02**

$Q \leftarrow Q - 1$ *or* $M \leftarrow M - 1$

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
Q Pseudo Register	DEQ Q	42 42 3A	3	3	
base-page quad	DEQ $nn	42 42 C6	4	12	dmr
absolute quad	DEQ $nnnn	42 42 CE	5	13	dmr
base-page quad,X	DEQ $nn,X	42 42 D6	4	12	$dmpr$
absolute quad,X	DEQ $nnnn,X	42 42 DE	5	13	$dmpr$

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

EOR

This instruction performs a binary exclusive-OR operation of the argument with the accumulator, and stores the result in the accumulator. Only bits that were already set in the accumulator, or that are set in the argument will be set in the accumulator on completion, but not both.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

EOR : Binary Exclusive OR **45GS02**

A ← A *XOR* M

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
[indirect],Z	EOR [$nn],Z	EA 52	3	7	^{ipr}

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

EORQ

This instruction performs a binary exclusive-OR operation of the argument with the Q pseudo register, and stores the result in the Q pseudo register. Only bits that were already set in the Q pseudo register, or that are set in the argument will be set in the accumulator on completion, but not bits that were set in both.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

EORQ : Binary Exclusive OR Quad **45GS02**

Q ← Q *XOR* M

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page quad	EORQ $nn	42 42 45	4	8	^{r}
absolute quad	EORQ $nnnn	42 42 4D	5	9	^{r}
(indirect quad)	EORQ ($nn)	42 42 52	4	10	^{ipr}
[indirect quad]	EORQ [$nn]	42 42 EA 52	5	13	^{ipr}

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

INQ

This instruction increments the Q pseudo register or indicated memory location.

Note that the indicated memory location is treated as the first byte of a 32-bit little-endian value.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

INQ : Increment Memory or Accumulator **45GS02**

Q ← Q + 1 *or* M ← M + 1

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
Q Pseudo Register	INQ Q	42 42 1A	3	3	
base-page quad	INQ $nn	42 42 E6	4	13	*dmr*
absolute quad	INQ $nnnn	42 42 EE	5	14	*dmr*
base-page quad,X	INQ $nn,X	42 42 F6	4	13	*dmpr*
absolute quad,X	INQ $nnnn,X	42 42 FE	5	14	*dpr*

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

LDA

This instruction loads the Accumulator Register with the indicated value, or with the contents of the indicated location.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

LDA : Load Accumulator **45GS02**

A ← M

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
[indirect],Z	LDA [$nn],Z	EA B2	3	7	*ipr*

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

LDQ

This instruction loads the Q pseudo register with the indicated value, or with the contents of the indicated location. As the Q register is an alias for A, X, Y and Z used together, this operation will set those four registers. A contains the least significant bits, X the next least significant, then Y, and Z contains the most significant bits.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

LDQ : Load Q Pseudo Register **45GS02**

Q ← M

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page quad	LDQ $nn	42 42 A5	4	8	*r*
absolute quad	LDQ $nnnn	42 42 AD	5	9	*r*
(indirect quad),Z	LDQ ($nn),Z	42 42 B2	4	10	*ipr*
[indirect quad],Z	LDQ [$nn],Z	42 42 EA B2	5	13	*ipr*

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

LSRQ

This instruction shifts either the Q pseudo register or contents of the provided memory location one bit right. Bit 31 will be set to zero, and the bit 0 will be shifted out into the Carry Flag.

Note that the memory address is treated as the first address of a little-endian encoded 32-bit value.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The C flag will be set if bit 0 of the value was set, prior to being shifted, otherwise it will be cleared.

LSRQ : Logical Shift Right Quad					**45GS02**
Q ← Q≫1, C ← A(0) *or* M ← M≫1					
			N Z I C D V E		
			+ + · + · · ·		
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**	
base-page quad	LSRQ $nn	42 42 46	4	12	dmr
Q Pseudo Register	LSRQ Q	42 42 4A	3	3	
absolute quad	LSRQ $nnnn	42 42 4E	5	13	dmr
base-page quad,X	LSRQ $nn,X	42 42 56	4	12	$dmpr$
absolute quad,X	LSRQ $nnnn,X	42 42 5E	5	13	$dmpr$

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

ORA

This instruction performs a binary OR operation of the argument with the accumulator, and stores the result in the accumulator. Only bits that were already set in the accumulator, or that are set in the argument will be set in the accumulator on completion, or both.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

ORA : Binary OR					**45GS02**
A ← A *OR* M					
			N Z I C D V E		
			+ + · · · · ·		
Addressing Mode	**Assembly**	**Code**	**Bytes**	**Cycles**	
[indirect],Z	ORA [$nn],Z	EA 12	3	7	ipr

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

ORQ

This instruction performs a binary OR operation of the argument with the Q pseudo register, and stores the result in the Q pseudo register. Only bits that were already set in the Q pseudo register, or that are set in the argument, or both, will be set in the Q pseudo register on completion.

Note that this operation treats the memory address as the first address of a 32-bit little-endian value. That is, the memory address and the three following will be used.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.

ORQ : Binary OR Quad **45GS02**

$Q \leftarrow Q\ OR\ M$

N	Z	I	C	D	V	E
+	+	·	·	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page quad	ORQ $nn	42 42 05	4	8	^{r}
absolute quad	ORQ $nnnn	42 42 0D	5	9	^{r}
(indirect quad)	ORQ ($nn)	42 42 12	4	10	^{pr}
[indirect quad]	ORQ [$nn]	42 42 EA 12	5	13	^{pr}

p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

RESQ

These extended opcodes are reserved, and their function is undefined and subject to change in future revisions of the 45GS02. They should therefore not be used in any program.

RESQ : Reserved extended opcode **45GS02**

UNDEFINED

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles	
(indirect quad,X)	RESQ ($nn,X)	42 42 01	4	10	^{ipr}
(indirect quad),Y	RESQ ($nn),Y	42 42 11	4	10	^{ipr}
base-page quad,X	RESQ $nn,X	42 42 15	4	8	^{pr}
absolute quad,Y	RESQ $nnnn,Y	42 42 19	5	9	^{pr}
absolute quad,X	RESQ $nnnn,X	42 42 1D	5	9	^{pr}
(indirect quad,X)	RESQ ($nn,X)	42 42 21	4	10	^{ir}
(indirect quad),Y	RESQ ($nn),Y	42 42 31	4	10	^{ipr}
base-page quad,X	RESQ $nn,X	42 42 34	4	8	^{pr}
base-page quad,X	RESQ $nn,X	42 42 35	4	8	^{pr}
absolute quad,Y	RESQ $nnnn,Y	42 42 39	5	10	^{pr}
absolute quad,X	RESQ $nnnn,X	42 42 3C	5	9	^{pr}
absolute quad,X	RESQ $nnnn,X	42 42 3D	5	10	^{pr}
(indirect quad,X)	RESQ ($nn,X)	42 42 41	4	10	^{ipr}
(indirect quad),Y	RESQ ($nn),Y	42 42 51	4	10	^{ipr}
base-page quad,X	RESQ $nn,X	42 42 55	4	8	^{pr}
absolute quad,Y	RESQ $nnnn,Y	42 42 59	5	9	^{pr}
absolute quad,X	RESQ $nnnn,X	42 42 5D	5	9	^{pr}
(indirect quad,X)	RESQ ($nn,X)	42 42 61	4	10	^{ir}
(indirect quad),Y	RESQ ($nn),Y	42 42 71	4	10	^{ipr}
base-page quad,X	RESQ $nn,X	42 42 75	4	8	^{pr}
absolute quad,Y	RESQ $nnnn,Y	42 42 79	5	10	^{pr}
absolute quad,X	RESQ $nnnn,X	42 42 7D	5	10	^{pr}

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

ROLQ

This instruction shifts either the Q pseudo register or contents of the provided memory location one bit left. Bit 0 will be set to the current value of the Carry Flag, and the bit 31 will be shifted out into the Carry Flag.

NOTE: The memory address is treated as the first address of a little-endian encoded 32-bit value.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The C flag will be set if bit 31 of the value was set, prior to being shifted, otherwise it will be cleared.

ROLQ : Rotate Left Quad **45GS02**

M ← M≪1, C ← M(31), M(0) ← C

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page quad	ROLQ $nn	42 42 26	4	12	dmr
Q Pseudo Register	ROLQ Q	42 42 2A	3	3	
absolute quad	ROLQ $nnnn	42 42 2E	5	13	dmr
base-page quad,X	ROLQ $nn,X	42 42 36	4	12	$dmpr$
absolute quad,X	ROLQ $nnnn,X	42 42 3E	5	13	$dmpr$

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

RORQ

This instruction shifts either the Q pseudo register or contents of the provided memory location one bit right. Bit 31 will be set to the current value of the Carry Flag, and the bit 0 will be shifted out into the Carry Flag

Note that the address is treated as the first address of a little-endian 32-bit value.

Side effects

- The N flag will be set if the result is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The Z flag will be set if the result is zero, otherwise it will be cleared.
- The C flag will be set if bit 31 of the value was set, prior to being shifted, otherwise it will be cleared.

RORQ : Rotate Right Quad **45GS02**

$M \leftarrow M \gg 1, C \leftarrow M(0), M(31) \leftarrow C$

N	Z	I	C	D	V	E
+	+	·	+	·	·	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page quad	RORQ $nn	42 42 66	4	12	dmr
Q Pseudo Register	RORQ Q	42 42 6A	3	3	
absolute quad	RORQ $nnnn	42 42 6E	5	13	dmr
base-page quad,X	RORQ $nn,X	42 42 76	4	12	$dmpr$
absolute quad,X	RORQ $nnnn,X	42 42 7E	5	13	$dmpr$

d Subtract one cycle when CPU is at 3.5MHz.
m Subtract non-bus cycles when at 40MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

RSVQ

These extended opcodes are reserved, and their function is undefined and subject to change in future revisions of the 45GS02. They should therefore not be used in any program.

RSVQ : Reserved extended opcode **45GS02**
UNDEFINED

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles	
(indirect quad,X)	RSVQ ($nn,X)	42 42 81	4	10	^{ip}
(indirect quad,SP),Y	RSVQ ($nn,SP),Y	42 42 82	4	10	^{ip}
(indirect quad),Y	RSVQ ($nn),Y	42 42 91	4	10	^{ip}
base-page quad,X	RSVQ $nn,X	42 42 95	4	8	^{p}
absolute quad,Y	RSVQ $nnnn,Y	42 42 99	5	9	^{p}
absolute quad,X	RSVQ $nnnn,X	42 42 9D	5	9	^{p}
(indirect quad,X)	RSVQ ($nn,X)	42 42 A1	4	10	^{ipr}
(indirect quad,X)	RSVQ ($nn,X)	42 42 C1	4	10	^{ipr}
(indirect quad),Y	RSVQ ($nn),Y	42 42 D1	4	10	^{ipr}
base-page quad,X	RSVQ $nn,X	42 42 D5	4	8	^{pr}
absolute quad,Y	RSVQ $nnnn,Y	42 42 D9	5	9	^{pr}
absolute quad,X	RSVQ $nnnn,X	42 42 DD	5	9	^{pr}
(indirect quad,X)	RSVQ ($nn,X)	42 42 E1	4	10	^{ipr}
(indirect quad),Y	RSVQ ($nn),Y	42 42 F1	4	10	^{ipr}
base-page quad,X	RSVQ $nn,X	42 42 F5	4	8	^{pr}
absolute quad,Y	RSVQ $nnnn,Y	42 42 F9	5	8	^{pr}
absolute quad,X	RSVQ $nnnn,X	42 42 FD	5	9	^{pr}

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

SBC

This instruction performs A − M − 1 + C, and sets the processor flags accordingly. The result is stored in the Accumulator Register.

NOTE: If the D flag is set, then the addition is performed using binary Coded Decimal.

Side effects

- The N flag will be set if the result of A − M is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The C flag will be set if the result of A − M is zero or positive, i.e., if A is not less than M, otherwise it will be cleared.
- The V flag will be set if the result has a different sign to both of the arguments, otherwise it will be cleared. If the flag is set, this indicates that a signed overflow has occurred.

- The Z flag will be set if the result of A − M is zero, otherwise it will be cleared.

SBC : Subtract With Carry **45GS02**

A ← − M − 1 + C

N	Z	I	C	D	V	E
+	+	·	+	·	+	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
[indirect],Z	SBC [$nn],Z	EA F2	3	0	

SBCQ

This instruction performs Q − M − 1 + C, and sets the processor flags accordingly. The result is stored in the Q pseudo register.

NOTE: the indicated memory location is treated as the first byte of a 32-bit little-endian value.

NOTE: If the D flag is set, the operation is undefined and subject to change.

Side effects

- The N flag will be set if the result of A − M is negative, i.e. has it's most significant bit set, otherwise it will be cleared.
- The C flag will be set if the result of A − M is zero or positive, i.e., if A is not less than M, otherwise it will be cleared.
- The V flag will be set if the result has a different sign to both of the arguments, otherwise it will be cleared. If the flag is set, this indicates that a signed overflow has occurred.
- The Z flag will be set if the result of A − M is zero, otherwise it will be cleared.

SBCQ : Subtract With Carry Quad **45GS02**

Q ← Q − M − 1 + C

N	Z	I	C	D	V	E
+	+	·	+	·	+	·

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page quad	SBCQ $nn	42 42 E5	4	8	r
absolute quad	SBCQ $nnnn	42 42 ED	5	9	r
(indirect quad)	SBCQ ($nn)	42 42 F2	4	10	ipr
[indirect quad]	SBCQ [$nn]	42 42 EA F2	5	13	ipr

i Add one cycle if clock speed is at 40 MHz.
p Add one cycle if indexing crosses a page boundary.
r Add one cycle if clock speed is at 40 MHz.

STA

This instruction stores the contents of the Accumulator Register into the indicated location.

STA : Store Accumulator **45GS02**

M ← A

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles	
[indirect],Z	STA [$nn],Z	EA 92	3	8	^{ip}

i Add one cycle if clock speed is at 40 MHz.

p Add one cycle if indexing crosses a page boundary.

STQ

This instruction stores the contents of the Q pseudo register into the indicated location.

As Q is composed of A, X, Y and Z, this means that these four registers will be written to the indicated memory location through to the indicated memory location plus 3, respectively.

STQ : Store Q **45GS02**

M ← A, M+1 ← X, M+2 ← Y, M+3 ← Z

N	Z	I	C	D	V	E
.	.	.	.	.	.	.

Addressing Mode	Assembly	Code	Bytes	Cycles	
base-page quad	STQ $nn	42 42 85	4	8	
absolute quad	STQ $nnnn	42 42 8D	5	9	
(indirect quad)	STQ ($nn)	42 42 92	4	10	^{ip}
[indirect quad]	STQ [$nn]	42 42 EA 92	5	13	^{ip}

i Add one cycle if clock speed is at 40 MHz.

p Add one cycle if indexing crosses a page boundary.